Pnina Soffer
Erik Proper (Eds.)

LNBIP 72

Information Systems Evolution

CAiSE Forum 2010
Hammamet, Tunisia, June 2010
Selected Extended Papers

Lecture Notes
in Business Information Processing 72

Series Editors

Wil van der Aalst
Eindhoven Technical University, The Netherlands

John Mylopoulos
University of Trento, Italy

Michael Rosemann
Queensland University of Technology, Brisbane, Qld, Australia

Michael J. Shaw
University of Illinois, Urbana-Champaign, IL, USA

Clemens Szyperski
Microsoft Research, Redmond, WA, USA

Pnina Soffer Erik Proper (Eds.)

Information Systems Evolution

CAiSE Forum 2010
Hammamet, Tunisia, June 7-9, 2010
Selected Extended Papers

Springer

Volume Editors

Pnina Soffer
University of Haifa
Carmel Mountain, 31905 Haifa, Israel
E-mail: spnina@is.haifa.ac.il

Erik Proper
Public Research Centre Henri Tudor
29, avenue John F. Kennedy, 1855 Luxembourg-Kirchberg, Luxembourg
E-mail: erik.proper@tudor.lu

Library of Congress Control Number: 2010940735

ACM Computing Classification (1998): J.1, H.3.5, H.4.1, D.2

ISSN 1865-1348
ISBN-10 3-642-17721-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-17721-7 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

springer.com

© Springer-Verlag Berlin Heidelberg 2011
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper 06/3180 5 4 3 2 1 0

Preface

The CAiSE series of conferences, which started in 1989, provides an established venue in the domain of information systems engineering for presenting and exchanging results of design-oriented research in information systems. In 2010, the 22nd CAiSE conference, organized in Hammamet, Tunisia, continued this tradition. The overarching theme of CAiSE 2010 was 'Information Systems Evolution'.

While the CAiSE conference itself focuses on papers that report on well-matured research, the CAiSE forum was created specifically to provide a platform to publish early results, as well as to report on tool development based-on/supporting research activities. As such, the CAiSE Forum aims at the presentation of fresh ideas, new concepts, as well as the demonstration of new and innovative systems, tools and applications. The Forum sessions at the conference were shaped in such a way to facilitate the interaction, discussion, and exchange of ideas among presenters and participants.

This year marked a further step in the maturation of the CAiSE Forum. Thus far, the CAiSE Forum only featured an informal proceedings format. This format really supported the informal and future focused style of the CAiSE Forum. However, to make the value of the Forum even more explicit and tangible, it was decided to create formal proceedings for the best contributions submitted to the CAiSE forum.

Out of the 32 submissions to the Forum, 25 submissions were invited to be presented at the conference. Finally, 22 of these contributions were accepted for the proceedings featuring extended versions of the original contributions, also taking the feedback from the original reviewers into due consideration.

October 2010

Pnina Soffer
Erik Proper

Organization

Workshop Co-chairs

Pnina Soffer	University of Haifa, Israel
Erik Proper	Public Research Centre – Henri Tudor, Luxembourg and Radboud University Nijmegen, The Netherlands

Program Committee

Alexander Dreiling	SAP Research, Australia
Anna Queralt	Universitat Politécnica de Catalunya, Spain
Atonia Albani	University of St. Gallen, Switzerland
Bas van der Raadt	Ernst & Young Advisory, The Netherlands
Bas van Gils	BiZZdesign, The Netherlands
Camille Salinesi	University of Paris 1, France
Frank Harmsen	Ernst & Young Advisory and University of Maastricht, The Netherlands
Hajo Reijers	Eindhoven University of Technology, The Netherlands
Hans Mulder	VIA Groep, The Netherlands and University of Antwerp, Belgium
Inge van de Weerd	Utrecht University, The Netherlands
Iris Reinhartz-Berger	University of Haifa, Israel
Irit Hadar	University of Haifa, Israel
Jan Mendling	Humboldt-Universität zu Berlin, Germany
Jan Recker	Queensland University of Technology, Australia
Johan Versendaal	University of Utrecht and University of Applied Sciences Utrecht, The Netherlands
Jolita Ralyteé	University of Geneva, Switzerland
Kai Riemer	The University of Sydney, Australia
Linda Terlouw	ICRIS and Delft University of Technology, The Netherlands
Marta Indulska	University of Queensland, Australia
Martin Op 't Land	Capgemini and Delft University of Technology, The Netherlands
Mathias Ekstedt	Royal Institute of Technology, Sweden
Michael zur Muehlen	Stevens Institute of Technology, USA
Michael Rosemann	Queensland University of Technology, Australia
Nelly Condori Fernández	Universidad Politécnica de Valencia, Spain

Nikolaus Müssigmann	University of Applied Sciences Augsburg, Germany
Olga De Troyer	Vrije Universiteit Brussel, Belgium
Pär Ågerfalk	Uppsala University, Sweden
Sergio España	Universidad Politécnica de Valencia, Spain
Sietse Overbeek	Delft University of Technology, The Netherlands
Stijn Hoppenbrouwers	Radboud University Nijmegen, The Netherlands

Table of Contents

Managing Personal Information through Information Components

Stefania Leone, Matthias Geel, and Moira C. Norrie

Institute for Information Systems, ETH Zurich,
CH-8092 Zurich, Switzerland
{leone,geel,norrie}@inf.ethz.ch

Abstract. We introduce the concept of information components and show how it can allow non-expert users to construct their personal information spaces by selecting, customising and composing components defined by the system or other users. The system presented is based on a plug-and-play concept where new user-defined applications can be created and integrated into the portal-style interface based on default templates which can easily be customised by the users.

Keywords: information components, personal information management, pluggable interface architectures.

1 Introduction

The term Web 2.0 refers to a new generation of Web-based applications that empower the user in the creation and management of content and services. Combined with the concepts of portals, widgets and mashups, some users have evolved to so-called advanced Web users [1] that not only manage and share their own data, but also integrate a wide range of external data and services. At the same time, they are encouraged to collaborate in a range of ways—including the community-based development of application libraries offered by sites such as Facebook.

It is therefore not surprising that users are increasingly turning to Web 2.0 sites for the management of personal information. However, this can in turn create its own problems in terms of losing control of one's own data and increased fragmentation of information across a range of Web 2.0 applications and desktop applications. At the same time, while sites such as Facebook provide a very large collection of applications for the management of personal information such as contacts, photo albums, places visited and virtual bookshelves, it is not possible to personalise these or combine them in flexible ways.

Our goal was to adopt concepts from Web 2.0 to empower users in the management of all of their personal information by allowing them to customise and compose *information components*. Instead of offering units of reuse at the interface or service level, we offer them at the database level, thereby allowing users to focus on their data requirements and to extend, customise, group or associate data items as they choose. To demonstrate the feasibility of the approach, we

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 1–14, 2010.
© Springer-Verlag Berlin Heidelberg 2010

have developed the PIM 2.0 platform based on an extension of an object database system that supports the information component concept. PIM 2.0 offers a Web-based pluggable interface architecture capable of automatically generating the portal-style interfaces based on default or selected templates which can easily be customised.

In this paper, we will focus on the technological level of our approach. We begin in Sect. 2 with a discussion of the background before introducing our notion of information components in Sect. 3. We then present our approach by first describing the process of extending a personal information space through the composition of components in Sect. 4 and then discussing the system architecture in Sect. 5. Implementation details of our PIM 2.0 system are given in Sect. 6. A discussion of the current status of our work and future directions of research is given in Sect. 7. Concluding remarks are given in Sect. 8.

2 Background

With the dramatic increase in the volume of personal data stored by users in recent years, there have been various proposals to build tools on top of existing file systems and desktop applications to extract and integrate data from different sources in order to meet a user's information needs, e.g. [2,3]. In the position paper by Franklin, Halevy and Maier [4], they propose a notion of *dataspace systems* where traditional database functionality such as metadata management, indexing and query processing can be used alongside traditional file systems and applications to support the administration, discovery and enhancement of personal data. Personal information management (PIM) solutions based on this vision aim to integrate all of a user's data and provide support for associating data of various types and from various sources, mostly following the vision of trails proposed in [5]. These associations may be automatically created as well as user-defined [3,6].

Generally, PIM systems proposed in research tend to either use a predefined PIM domain model e.g. [7,3,8], or work according to a "no-schema" or "schema-later" approach e.g. [6,9,2]. On the interface level, they mostly offer a generic browser where a user can browse data via associations. In Haystack [2], entities are self-rendering in that they know how to render and display themselves and offer a context menu with the operations for that entity.

While these systems help users work with information fragmented across existing applications, they do not provide the basic infrastructure to enable users to easily design, build and customise their own personal information space. We believe that, with the right tools, users could be put in control of what data they manage and how they manage it. Our general aim is not dissimilar to that of MashArt [1,10] where they provide a platform that enables advanced Web users rather than developers to create their own applications through the composition of user interface (UI), application and data components. The main difference is that they focus on the integration of existing services, whereas we want to empower users in the creation as well as the composition of components. Further,

our components are *information components* rather than *services* and integration is done at the database level. Therefore, whereas MashArt uses event-based composition, the composition of information components involves creating new data structures that integrate and augment existing metadata and data. Last but not least, we support the automatic generation of the UI based on a portal-style of interface familiar from Web 2.0 applications.

Web 2.0 has already had a tremendous impact in terms of how people use the Web to communicate, collaborate and manage personal data. Its success can provide valuable lessons in how to provide easy-to-use, customisable and extensible platforms for PIM. In particular, users have become familiar with the plug-and-play style of interfaces offered by portals as well as the plug-and-play style of applications offered by social networking sites such as Facebook which now offers thousands of applications developed by the user community. They are also becoming increasingly familiar with the notions of widgets that allow small, lightweight applications to be integrated into Web pages and Web-based mashups that allow the integration of services within a client. Taken together with the notions of user-generated content underlying Web 2.0, users are increasingly becoming empowered to manage their own information needs. However, on the negative side, the increased usage of Web 2.0 applications for PIM has drastic consequences in terms of loss of user's control over their own data and also information fragmentation [11,12].

We have adopted features of Web 2.0 to develop a platform for PIM that allows users to define and manage their own personal information space by creating, sharing, customising and composing *information components*. These components define the data structures, application logic and interaction logic that support some particular information management task. Being able to build a personal information space in a plug-and-play manner through the selection and composition of these components allows even non-expert users to profit from the experience of more advanced users and have fine-grained control over their PIM.

While reuse in databases has been considered at the architectural level in terms of component database systems [13] and also at the data level in terms of various forms of data integration services [14], little attention has been given to reuse at the database schema level to support reuse in the design and development of applications. An exception is the work of Thalheim [15] where he proposed the use of composable sub schematas to enhance the management of large and complex database schemas. In contrast, we focus on reuse as a means of allowing non-expert users to create a customised personal information space. We achieve this by providing them with a Web-based pluggable interface with an embedded set of graphical tools that enables them to create their personal information space through the selection, customisation and composition of components that are usually small and simple. We note that in the case of object databases where the database schema corresponds to class definitions that include application logic and also possibly presentation and interaction logic, schema reuse can have a significant impact. However, we believe that reuse is

important even for relational or XML databases with no application logic integrated into the schema to enable non-expert users to design and develop their own personal information spaces.

3 Approach

Information components are intended to be the basic units of reuse in information systems at both the schema and data level. Fig. 1 gives an overview of the information component meta model. An information space consists of a set of information components where each information component can define both metadata and data. An information component may reuse data and metadata from other components and may expose data and metadata for reuse. The reuse is reflected by the import aggregation, which defines the structures that have been imported from other components. In turn, the export aggregation specifies the data and metadata that are shared for reuse in new components. When creating a new component, one can import and reuse structures from existing components, if desired. If an information component exports only metadata, reuse is only at the schema level to support the design of an information space. Optionally, an information component may export data in addition to metadata which allows the reuse of data.

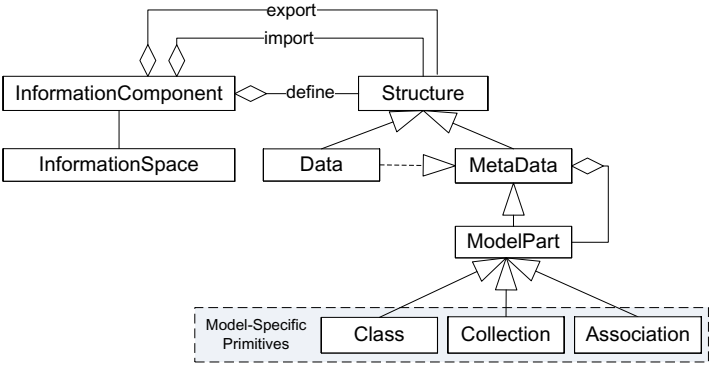

Fig. 1. Information components metamodel

Our prototype was developed using an object database based on the OM model [16] and therefore the metadata of a component is defined in terms of the model primitives which are classes, collections and associations. We show this in Fig. 1 as a particular set of model parts to indicate that our notion of information components could be applied to other models.

Our approach allows users to construct their personal information space by defining their own components through a process of selecting, extending and combining existing components. A core set of system-defined components are provided to support the basic, common information management tasks in PIM

systems such as the management of contacts and we show in the next section how a user can use these as the starting point for developing their own PIM applications. In addition, users can also reuse applications defined by other users based on a global registry.

4 Application Development Process

Having introduced the concept of information components, we now describe how this can be used to build a personal information space through the composition of information components. Assume a user has a simple contact application and a picture management application and would like to tag pictures with contacts. We will now illustrate how the user could extend their personal information space with this functionality by creating a new information component from the composition of the existing components.

An application consists of an information component together with an associated user interface. Figure 2 gives an overview of the steps involved in building an application. Before detailing the steps, we note that some tasks are carried out by the user while others are done automatically by the system. To create a new application, the user basically models the application domain and associates domain concepts to templates. The system then generates both the database representation of the domain model and the user interface based on the domain concept-template assignment specified by the user and deploys the application into the user interface. Our PIM 2.0 system has a portal-like interface and we provide an Application Manager application embedded into the PIM portal that supports all of these operations.

① shows the PIM 2.0 UI with three applications. In order to create a new application, a user first creates a new information component ② by modelling the application domain. A user may choose to reuse existing component parts and/or specify new domain concepts. Note that an information component can have arbitrarily complex models, but may also represent a single domain concept.

In our example, we assume the reuse of two information components, the *Contacts* component and the *Pictures* component. According to the constructs of the OM data model, a component will consist of one or more collections of a particular membertype plus associations between collections. Cardinality constraints can be specified over associations. In addition, the OM model can represent rich classification structures through the use of subcollection and subtyping relationships together with classification constraints such as disjoint and cover. It is beyond the scope of this paper to describe the OM model in full. Details of the model are given in [16].

In the following examples, we use an abbreviated form of the data definition language associated with the OM model for the sake of simplicity. In particular, we will specify the collections and their membertypes together, although we note that it is possible in the OM model to define multiple collections with the same membertype and therefore they are usually defined separately.

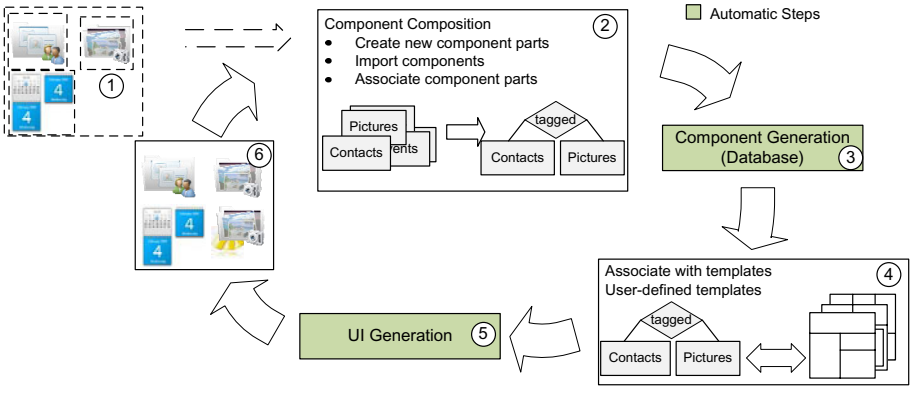

Fig. 2. Process of composing information components

Assume that the *Contacts* component has a single collection `contacts` defined as:

```
contacts(name:string, birthday:date, phone:set, address:set)
```

Further, assume that the *Pictures* component is defined as:

```
pictures(picture:uri, caption:string)
albums(name:string)
picturesInAlbums(picture:ref, album:ref)
```

where `picturesInAlbums` is an association between `pictures` and `albums`. To create a *PictureTagging* component, the user could choose to import `contacts` from the *Contacts* component and `pictures` from the *Pictures* component and associate these two parts with a `tagged` association ②, defined as follows:

```
Contacts.contacts(name:string, birthday:date, phone:set, address:set)
Pictures.pictures(picture:uri, caption:string)
tagged(picture:ref, contact:ref)
```

Note that the names of collections, types, attributes etc. may be qualified with their component names to ensure uniqueness. However, we also provide options for the user to rename objects during the composition process in order to simplify the interface of the new component.

Figure 3 shows a screenshot of the component composition process in the Application Manager. Users can drag and drop component parts into the composer area of the Application Manager to reuse them. New classes, collections and associations can be created using the menu on the left. For the *PictureTagging* component, the `contacts` collection from the *Contacts* component and the `pictures` collection from the *Pictures* component have been associated with a `tagged` association in the composer area. Once defined, the component model is automatically created in the database ③.

To create the interface of the new component, the user can associate the component parts to structural templates that define how the data will be displayed ④. These templates define what should be displayed in terms of attributes

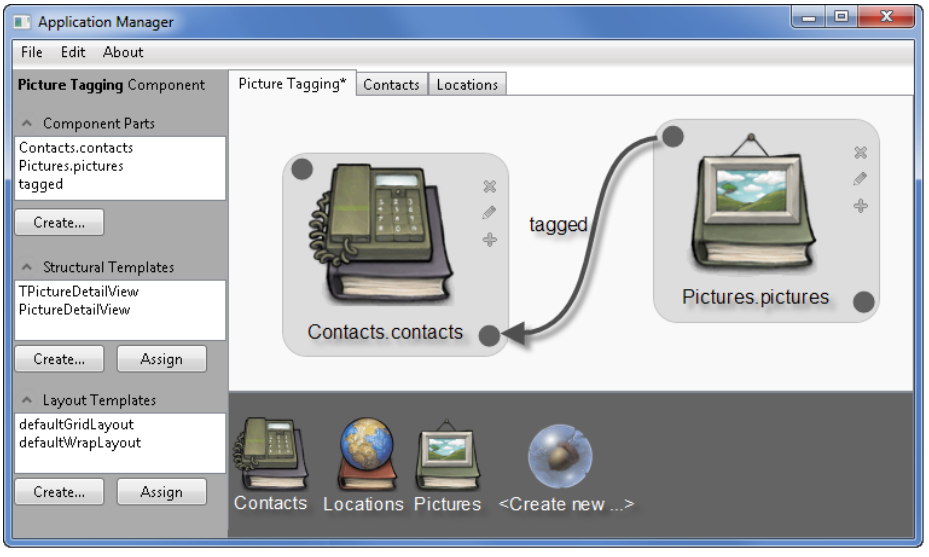

Fig. 3. Application Manager

and associated entities, the order in which these should be displayed and also the operations offered through context menus and buttons. To facilitate the process of defining the user interface, we defined standard templates for displaying collections and objects in read or write modes. Further, standard structural templates can be extended or new ones created through the Application Manager by users defining views through the simple selection and ordering of object attributes. Also, context menus to select from various standard options are available where appropriate.

The actual layout and positioning of data is defined in separate layout templates that are applied upon interface generation. Note that users can create their own layout templates or extend existing ones. During UI generation, a view is generated which includes the layout as well as the structural information and represents the actual UI through which the user interacts with the data.

The default collection template displays a collection as a list and the user has to specify the object attributes to appear in that list. For example, they might specify that the `contacts` collection be displayed as a list of surnames followed by forenames. By default, collection views are always in write mode, so that objects can be selected, added to and removed from the list.

The default structural templates represent objects in a generic way. It is easy for users to create their own custom templates. For example, they might specify that in the picture detail view, the actual picture together with a caption should be displayed rather than the URL. To support such customisations, templates can be created which specify a set of applicable types and users are presented with a choice of presentations. The picture detail template would have the form:

```
<view name='PictureDetailView' mode='read'>
  <layout source='defaultGridLayout'>
  <compatibletypes>
    <type name='pictures'/>
  </compatibletypes>
  <attributes>
    <attribute name='picture' resource='attributes/picture'/>
    <attribute name='caption' value='attributes/caption'/>
  </attributes>
</view>
```

Attribute `picture` is a resource which means that the value is the path to the resource to be displayed, while the attribute caption is a value that can be displayed directly. This template uses a default layout template *defaultGridLayout*, but a user can also define their own layout template or extend the default ones.

A user may wish to reuse the customised templates of imported components, extending them to cater for new data or functionality. For example, in the picture tagging application, the user might want to display the names of tagged persons along with the picture and caption. This could be done by creating a template that extends the `PictureDetailView` template as follows:

```
<view name=''TPictureDetailView'' extends name=''PictureDetailView''>
  ...
  <attributes>
    <attribute name = 'Tagged'
       value='associations/tagged/contacts/attributes/name' type='set'/>
  </attributes>
</view>
```

The previous template is extended with an attribute that provides a set of names of the people related by the 'tagged' association of the *PictureTagging* component as specified by a path expression. By declaring `type='set'`, we indicate that the navigation may yield a set of values all of which should be displayed. After associating the model with the templates, the actual views are generated by combining the structural and layout templates ⑤. The application is then deployed into the portal. In our example, the PIM portal is extended and features the additional picture tagging application ⑥.

5 Architecture

Having described the general process of composing information components, we now look in more detail at how our approach works in practice in terms of supporting the local and global reuse of components. We start by looking at the structure of the PIM 2.0 system which manages and provides access to a user's personal information space based on the concept of information components presented in the previous sections.

Figure 4 provides an overview of the PIM 2.0 structure. The system has two main parts—the Interface Manager and the Component Manager. The Component Manager is responsible for the creation and manipulation of information

components as well as the management of the template assignments. The Interface Manager manages the template repository where the default structural and layout templates as well as user-defined templates are stored. Structural templates are written in an implementation-independent XML dialect whereas layout templates are written in an implementation-dependent manner since they are part of the underlying UI technology. The specific technologies used and implementation details are given in the next section.

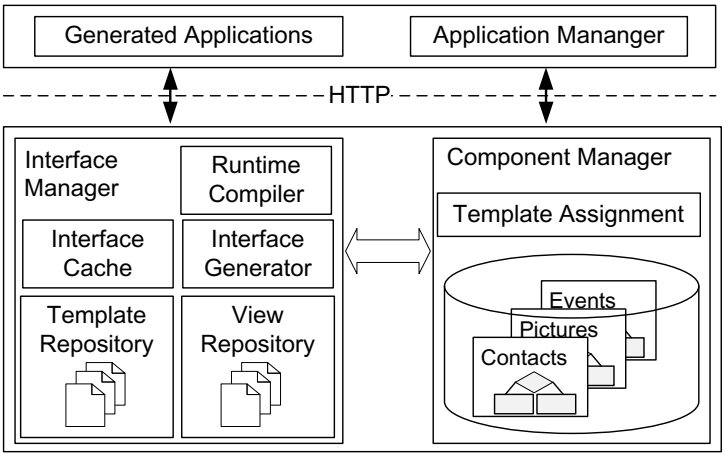

Fig. 4. Structure of PIM 2.0 system

The user interface consists of two parts, the actual portal-like user interface which is the set of applications to manage personal data and the Application Manager that allows users to create new applications by defining new information components. The Application Manager exhibits the functionality presented in Sect. 4, namely component design, template design and component-template assignment. As mentioned previously, the Application Manager is provided as an application embedded in a user's PIM portal.

Since there are many common information management tasks in PIM systems such as the management of contacts and todo lists, we offer a core set of system-defined components for these tasks. This means that the user can start by selecting the components that they require from this set. However, it is simple for them to create new customised components based on these by selecting which parts of the core components they want to use and/or extending them with additional information. We refer to this as *local reuse* and it often occurs as part of the evolution of a personal information space and not just the initial design. This means that it is common that not only the metadata, but also the data is integrated as part of the composition process. Data integration usually takes the form of creating new objects, collections or associations that enable data from the imported components to be aggregated or linked together.

While users could use the system to design and build their personal information space based on a set of pre-defined components, one of the central ideas behind our approach is that they can benefit from the experience and expertise of others through the reuse of components developed within the user community. We refer to this as *global reuse*. In addition to the local repository of components and associated templates shown in Fig. 4, there is a global registry where users can make their information components available to others. A component can be registered together with one or more associated structural and view templates. A user can therefore search for components and templates in the global registry and select among the variants offered for a particular information management task. Since information components are typically rather simple, it is not difficult for a user to select a component based on keywords, description and inspection. Clearly there are also issues related to trust and security and we are currently investigating a combination of technical approaches based on making PIM 2.0 a secure, controlled environment and ones of trust commonly adopted in Web 2.0 platforms such as Facebook.

6 Implementation

In this section, we describe how we implemented the PIM 2.0 system. The first step was to implement an object database system that supported the concept of information components defined in this paper. In a second step, we implemented a Web application on top of the database that allows users to manage their personal information space by creating, composing and accessing information components through a portal-style interface.

Figure 5 gives an overview of our layered system architecture. For each layer, we indicate the technologies used. The database layer consists of an object database that has been extended with the information component concept. We have developed our own object database system, OMS Avon, which is based on the OM model and implemented in Java [17]. The information component concept is implemented as a meta model extension module [18]. Since OMS Avon bootstraps its meta model, we were able to extend the Avon meta model with the information component meta model and thus have the information component concepts as metadata concepts natively in the database.

In OMS Avon, meta model concepts are manipulated via corresponding CRUD classes. These implement the basic data management operations of create, read, update and delete for the various meta model concepts as well as providing additional methods that implement more sophisticated operations. We extended the set of CRUD classes for types, collections, associations and constraints provided by Avon with an information component CRUD class, that supports the creation, retrieval, deletion, composition and manipulation of information components. Upon creation, the `import` method supports the importation and thus composition of components, while the `export` method exposes component structures. The actual data is accessed via the `ObjectsCrud` class which supports the functionality of creating, retrieving and deleting objects, as well as retrieving and updating attribute values.

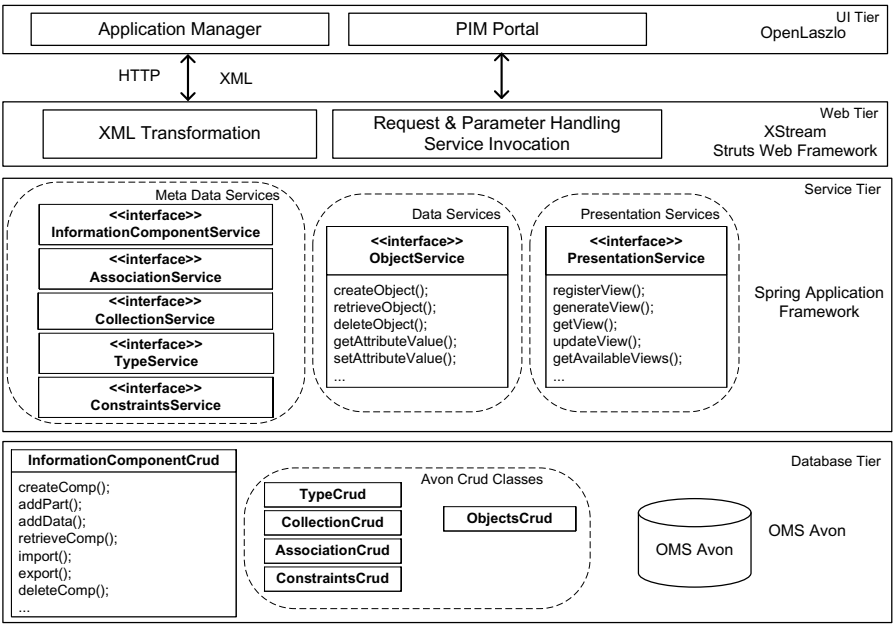

Fig. 5. Layered architecture of framework

These CRUD classes are used by the service tier to manage information components. The service tier exposes the information component concepts as a Web service. For each meta model concept, a service class has been created which offers methods to deal with these concepts. These service classes contain additional application logic needed by the Application Manager. Additionally, this tier introduces model classes that correspond to entities from the database schema such as collections and associations. These are used to transfer data between the server and client via the Web service interface. All model classes are simple Java classes and follow the JavaBeans specification. The service tier separates the services in three classes. *Metadata Services* provide methods to deal with information components and allow the definition of domain models using OM concepts. *Data Services* are used to create and manipulate domain objects i.e. the actual data. The *Presentation Service* offers methods to manage the view definitions and generation as well as functionality to register structural and layout templates for specific types.

The Web service tier mediates data between the service tier on the server-side and the Web-based UI on the client side. It has been decoupled from the service tier in order to allow for different Web service implementations. This tier processes URL requests to the Web service and serves responses back to the clients. It was implemented using the Struts Web application framework and uses XStream to marshal the Java model classes to XML and back.

The Web client offers a portal interface to a user's PIM system, where components are represented as portal applications. The Application Manager is also

a portal application embedded within the PIM interface and is used to create, compose and manage applications. The UI has been implemented using the OpenLaszlo Web application framework[1]. The OpenLaszlo architecture allows Web applications to be specified using a declarative XML syntax, Openlaszlo XML (LZX), that is then compiled to Flash on-the-fly. As an immediate benefit, this architecture allows us to compile automatically generated OpenLaszlo applications dynamically at runtime. We make use of this functionality to automatically load newly created applications into the PIM interface upon invocation of the view generation process on the server side.

The Presentation Service on the server-side orchestrates the whole interface generation process and upon generation invocation lets the Interface Manager retrieve the required template files from the file system. It then invokes the View Generator that composes these templates and injects additional declarative statements where necessary, such as bindings of XML data to structural OpenLaszlo elements via XPath. The resulting OpenLaszlo files are then stored back to the file system. These OpenLaszlo XML files are ready to be compiled by OpenLaszlo as soon as they are requested for the first time.

7 Discussion

Our ultimate goal is to enable users to design and build their own personal information spaces according to their specific information requirements. To achieve this, we propose an approach that allows users to construct their personal information space in an incremental way based on the reuse and composition of both data and metadata shared with other users. These ideas are similar to the idea of crowdsourced systems [19], where members of the user community develop and share applications that can be reused by other members of the community. While ideally regular users would be supported in composing their information spaces, in reality, our system addresses the needs of advanced Web users who are familiar with Web 2.0 platforms and have worked with technologies such as widgets and mashups.

While most PIM systems are either based on predefined, no-schema or schema-later approaches, we combine the advantages of all of these and offer a user a set of existing PIM components that can either be imported from a global registry or are already present in a user's local information space. The user can then extend or compose these to create new information components according to their information needs as they evolve. Our studies of existing PIM systems and also various Web 2.0 platforms such as Facebook shows that PIM application schemas tend to be rather small and simple. Users therefore tend to find PIM schemas easy to understand and our initial experiences with PIM 2.0 suggest that they have no problems to work with and compose our information components. However, this is something that requires detailed studies in the future.

Metadata and data reuse is currently done by reference in the case of local reuse and by copy in the case of global reuse. In earlier work [11], we investigated the

[1] `http://www.openlaszlo.org/`

problem of data replication across a user's information space and Web 2.0 applications and proposed a framework that supports the synchronisation of local data with replications of that data on social networking sites. This is closely related to the importation of global components and the desire to stay 'synchronised', especially when importing not only schema, but also data. For example, one could import a movie database from a global provider such as IMDB, where a user extends the movie database with personal ratings and comments. A user, however, would want to have an up-to-date movie database, which asks for periodical synchronisation with the original source. We are currently investigating different options of how to support reuse within and across information spaces.

While our prototype implementation of PIM 2.0 has shown the feasibility of our approach within a user's information space, we are currently investigating several issues related to global reuse. As already mentioned, one area of research is to investigate issues of security and trust. While we believe that it is important to have mechanisms that ensure for example that users' personal data cannot be accessed by other parties unless explicitly shared, we also think it is important to consider the notions of trust common in Web 2.0 platforms and the extent to which these can be applied in more general settings of PIM.

Finally, we report on our experiences concerning the particular choice of technologies used to implement PIM 2.0. As described earlier, we used OpenLaszlo for the UI of our system. OpenLaszlo is one example of a Rich Internet Application (RIA) framework and we have also experimented with alternatives. Our experience has shown that while RIA frameworks are well suited to rapid prototyping, they are not as flexible and dynamic as expected, especially in terms of special purpose UI widgets and non-standard interaction models that are not inherent in the system. As a result, while OpenLaszlo enabled us to successfully demonstrate the concept, the current version of PIM 2.0 is rather limited in terms of rich user interaction and therefore not really suitable for carrying out detailed user studies. We are therefore investigating alternative implementation strategies.

8 Conclusion

We argue that users should be empowered in the management of their personal data by providing tools that can enable them to design and build PIM systems adapted to their personal information requirements. We have presented the concept of information components as a mechanism for allowing users to construct their personal information space in a plug-and-play style of composing schemas and data. By supporting reuse within and across PIM systems, we believe that advanced Web users can create and share components with other users, while non-expert users can still benefit from the expertise and experience of the community similar to collaboration evident in many Web 2.0 communities.

References

1. Daniel, F., Casati, F., Benatallah, B., Shan, M.C.: Hosted Universal Composition: Models, Languages and Infrastructure in mashArt. In: Laender, A.H.F., Castano, S., Dayal, U., Casati, F., de Oliveira, J.P.M. (eds.) ER 2009. LNCS, vol. 5829, pp. 428–443. Springer, Heidelberg (2009)
2. Karger, D.R.: Haystack: Per-User Information Environments. In: Kaptelinin, V., Czerwinski, M. (eds.) Beyond the Desktop Metaphor: Designing Integrated Digital Work Environments, 1st edn., vol. 1, pp. 49–100. The MIT Press, Cambridge (2007)
3. Dong, X., Halevy, A.Y.: A Platform for Personal Information Management and Integration. In: Proc. CIDR 2005, Asilomar, CA, USA (2005)
4. Franklin, M., Halevy, A., Maier, D.: From Databases to Dataspaces: A New Abstraction for Information Management. ACM SIGMOD Record (December 2005)
5. Bush, V.: As We May Think. The Atlantic Monthly 176(1) (1945)
6. Vaz Salles, M.A., Dittrich, J.P., Karakashian, S.K., Girard, O.R., Blunschi, L.: iTrails: Pay-As-You-Go Information Integration in Dataspaces. In: Proc. VLDB 2007, Vienna, Austria (2007)
7. Gemmell, J., Bell, G., Lueder, R.: MyLifeBits: a Personal Database for Everything. ACM Comm. 49(1) (2006)
8. Shoens, K.A., Luniewski, A., Schwarz, P.M., Stamos, J.W., Thomas, J.: The Rufus System: Information Organization for Semi-Structured Data. In: Proc. VLDB 1993, Dublin, Ireland (1993)
9. Freeman, E., Gelernter, D.: Lifestreams: a Storage Model for Personal Data. SIGMOD Record 25(1) (1996)
10. Daniel, F., Yu, J., Benatallah, B., Casati, F., Matera, M., Saint-Paul, R.: Understanding UI Integration: A Survey of Problems, Technologies, and Opportunities. IEEE Internet Computing 11(3) (2007)
11. Leone, S., Grossniklaus, M., Norrie, M.C.: Architecture for Integrating Desktop and Web 2.0 Data Management. In: Proc. IWOOST 2008, Yorktown Heights, USA (2008)
12. Norrie, M.C.: PIM Meets Web 2.0. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) ER 2008. LNCS, vol. 5231, pp. 15–25. Springer, Heidelberg (2008)
13. Dittrich, K.R., Geppert, A. (eds.): Component Database Systems. Morgan Kaufmann, San Francisco (2001)
14. Halevy, A., Rajaraman, A., Ordille, J.: Data Integration: the Teenage Years. In: Proc. VLDB 2006, Seoul, Korea (2006)
15. Thalheim, B.: Component Development and Construction for Database Design. Data & Knowledge Engineering 54(1) (2005)
16. Norrie, M.C.: An Extended Entity-Relationship Approach to Data Management in Object-Oriented Systems. In: Elmasri, R.A., Kouramajian, V., Thalheim, B. (eds.) ER 1993. LNCS, vol. 823. Springer, Heidelberg (1994)
17. Norrie, M.C., Grossniklaus, M., Decurtins, C., de Spindler, A., Vancea, A., Leone, S.: Semantic Data Management for db4o. In: Proc. ICOODB 2009, Berlin, Germany (2009)
18. Grossniklaus, M., Leone, S., de Spindler, A., Norrie, M.C.: Dynamic Metamodel Extension Modules to Support Adaptive Data Management. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 363–377. Springer, Heidelberg (2010)
19. Kazman, R., Chen, H.M.: The Metropolis Model a New Logic for Development of Crowdsourced Systems. ACM Commun. 52(7), 76–84 (2009)

Exploiting Tag Clouds for Database Browsing and Querying

Stefania Leone, Matthias Geel, Corinne Müller, and Moira C. Norrie

Institute for Information Systems, ETH Zurich
CH-8092 Zurich, Switzerland
{leone,geel,norrie}@inf.ethz.ch

Abstract. Querying and browsing of databases is a task exclusively done by experts that have mastered the query language and are familiar with a database's schema. We show how tag clouds can be used alongside more traditional query languages and data visualisation techniques as a means for browsing and querying databases by both experts and non-expert users. Our approach is based on a general, extensible framework that supports different modes of visualisation as well as different database systems. We have validated our prototype with a user study that has shown how non-experts were able to browse and retrieve data that usually would only be possible by means of queries.

Keywords: tag cloud, data visualisation, database interface.

1 Introduction

Tag clouds are becoming extremely popular as a means of providing visual summaries of collections of documents. Although very simple, they can be used to support search, browsing and recognition as well as forming and presenting impressions [1]. The presentation and layout of tags can be controlled so that features such as the size, font and colour can be used to give some measure of the importance of a given tag, while the positioning of tags may be based on pure aesthetics, alphabetical sorting or some form of relationship between tags.

While tag clouds have been widely used in Web 2.0 applications for visualising user-generated tags and folksonomies of web sites such as Flickr[1], more recently they have been introduced into various application domains as a means of summarising and visualising specific domain information. For example, in [2] tag clouds were used as a visual augmentation of employee profile pages, while [3] use them to summarise research activities.

Given the flexibility of tag clouds in terms of information representation together with the simplicity of the associated style of navigation, it is natural that database researchers should consider exploiting the concept of tag clouds to address the longstanding problems of database usability [4]. The use of a query language requires the user to master not only the query language but also the

[1] http://www.flickr.com

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 15–28, 2010.
© Springer-Verlag Berlin Heidelberg 2010

database schema. To allow users to view the data in a natural way, a higher-level presentation of the database content such as a visual schema browser and query interface is needed.

Our goal was to investigate the extent to which tag clouds could be exploited to support database browsing and querying, either by replacing existing query languages and other modes of data visualisation or being used alongside them. Our tag clouds therefore mainly represent data and metadata values rather than terms occurring within the data. To support our investigations, we have developed a general, extensible framework that supports different modes of data visualisation, including customisable tag clouds. We have also designed it so that different types of databases can be accessed, and currently have implementations for both object databases and relational databases.

A key advantage of the tag cloud approach is that it is data-driven rather than schema-driven which is particularly beneficial to users with no experience of databases and query languages. Our initial user studies have shown that even users with low computer literacy and no previous experience of tag clouds were able to find the results of non-trivial queries using our system. At the same time, expert users also gave favourable feedback about the system.

In Sect. 2, we discuss the background to this work before going on to present the details of our approach in Sect. 3. We then provide an overview of the system that we have implemented in Sect. 4. In Sect. 5, we present the user study and its results. Open issues and directions of future research are discussed in Sect. 6 before giving concluding remarks in Sect. 7.

2 Background

While database technology has made great advances in the past decades, database usability has not improved a lot and database systems are still known for being hard to access and query [4]. Query languages such as SQL and XQuery are the current means provided by database systems to allow users to access and retrieve data. While these languages are powerful, they require users to not only master the query language but also to fully comprehend the schema of the database. Thus, at present, the capabilities of database systems can only be fully exploited by expert users. End users usually access only structured data by means of special purpose applications built on top of databases, where the data access is constrained by the specific user interface and the functionality provided. Consequently, few general computer users take the step of actually using database software directly to help them create and access personal information.

It has therefore been a longstanding challenge to develop interfaces that can allow non-expert users to access a database without having detailed knowledge about the underlying technology or schema design. To allow users to view the data in a natural way, a higher-level presentation of the database content, such as a visual query interface, is required. A lot of research has already been done in the area of visual query languages [5], but so far such research has tended to have an impact only in specific application domains. In more recent work [4], Jagadish

et al. presented a system that offers a set of user interfaces for an XML database including a traditional XQuery interface, a keyword search, a natural language interface, a form-based interface and a visual query editor. Since they found that users had difficulties locating elements of interest in the schema tree, they experimented with schema summarisation as well as schema-free approaches.

In contrast, one of the effects of Web 2.0 is that visualisation paradigms such as tag clouds and faceted browsing are now in widespread use for searching large data collections of user-generated content. These interfaces are targeted to support end users in the tasks of browsing and searching data collections in an easy and intuitive way, and are used by non-experts and experts alike. In some cases, a more advanced interface may be provided to support more complex searches, but the simpler visual interfaces will suffice for the majority of queries.

While tag clouds originally emerged for browsing user-generated folksonomies, they have become popular for visualising data of various types. For example, in [2], they describe a social tagging extension to an employee directory application, where employees can tag each other and employee profiles were extended with tag clouds. Each employee profile got enriched with two tag clouds, an incoming and outgoing tag cloud, where the incoming tag cloud visualised the tags assigned to that person from their co-workers and the outgoing tag cloud summarised the terms used by that person to tag their co-workers. Their user studies showed that people tag other people as a form of contact management, to see all the people associated with a project, or to locate experts in a particular area. They also found that the tags were considered by users as giving accurate descriptions of their interests and expertise. Similarly, in [3], we experimented with tag clouds that provide visual summaries of researchers' activities to promote awareness within research groups and institutes. Each researcher was associated with a tag cloud generated automatically based on the documents that they read and wrote. These tag clouds were integrated into an ambient information system deployed within a research group. A user study showed that researchers could recognise other members of their research group based on their tag cloud. At the level of an institute, participants of the study were able to recognise a research group based on an individual's tag cloud but not that individual. This meant that, through a common visualisation scheme, awareness of the activities of individual researchers and research groups could be promoted at a level meaningful to different users.

Tag clouds have also been proposed as a means of summarising and refining the results of keyword searches over structured data as presented in [6,7]. In this case, the term *data cloud* is used to refer to their particular adaptation of tag clouds for summarising keyword search results. Data clouds were implemented as part of CourseRank, a social tool to access official university information and statistics, such as course descriptions, grade distributions and course evaluations, as well as user-generated information, such as course ratings, comments, questions and answers. Students could use the tool to search for classes, give comments and ratings, and also organise their classes into a personalised schedule. An interesting feature of their approach is that since it was developed for

relational databases, the developer of a data cloud application specifies how application entities can be composed from the relations in the database in order that keyword search can be applied to entities rather than simple attributes or tuples. The keyword search is based on a traditional information retrieval approach where entities are considered as documents and attribute values as weighted terms.

Another project that uses tag clouds for summarising query results is PubCloud [8] for searching the PubMed biomedical literature database. In this case, the tag clouds are generated from words extracted from the abstracts returned by the query. In [9], an approach using multiple synchronised tag clouds has been proposed to summarise, browse and compare search results over clinical trial data. This approach is similar to the one of faceted browsing in that browsing and querying are combined and results are refined.

In [6], data clouds for specific application domains rely on entity construction of the underlying data done by the cloud application developer, while [9] is only targeted to the domain of clinical trial data and their specific categorisation scheme. In contrast to these approaches which focus on domain-specific information visualisation of data sources, our aim is to strive for a more general approach, where tag clouds can be exploited to browse data in a generic way by offering synchronised tag clouds for browsing both schema and data.

3 Data Browser

Our general goal was to provide both non-expert and expert users with a user-friendly and intuitive interface to traditional databases by exploiting the concept of tag clouds and adapting them to the process of querying and browsing structured data. In the Web, the tags within a tag cloud are usually hyperlinks that lead to the collection of items that are associated with the tag. Tag clouds are graphically appealing due to different visualisation features. Tag cloud features include text features, such as the tag content, the size, font style and colour as well as the positioning and order of tags in a cloud. A lot of studies, such as [10,1,11] have experimented with tag cloud features and positioning and their impact on users. According to [10,1], font size, font weight and intensity are the most important features. While topic-based layouts of tags can improve search performance for specific search tasks compared to random arrangements, they still perform worse than alphabetic layouts according to [11].

We adapted these concepts to browse structured data where tags represent attribute values. Clicking on a tag initiates a selection for data items with the corresponding attribute value. In the case of object databases, the result would be a collection of objects, while in the case of a relational database it would be a collection of tuples, i.e. a relation. It is possible to form tag clouds from combined attribute values or to combine the display of various attributes by using the tag content to represent one attribute and the colour of a tag to indicate the value of another attribute. Using visual features such as the colour of a tag to represent a specific value of an attribute is appropriate only when the number of distinct values is small.

We use these concepts to browse both data and metadata and have even experimented with a mix of metadata and data within tag clouds as well as with synchronised browsing. We define a data source to be a set of data collections, where each collection contains data items of a specific type. These collections are either class extents or sets of objects of a specific type in object databases, while they are relations in relational databases.

We now explain these concepts further by means of an example based on a database with information about contacts and their locations. Figure 1 gives an overview of the data model using ER notation. Contacts can either be persons or organisations and are located at specific locations. Persons are further categorised into private contacts, ETH persons and persons that previously worked at ETH. Persons work for organisations and organisations can be part of larger organisations. Note that in an object database representation of this model, the metadata for the entities will map into two concepts, namely the object types and the collections of objects. Thus, contacts would be represented by a type `Contact` and a collection `Contacts`.

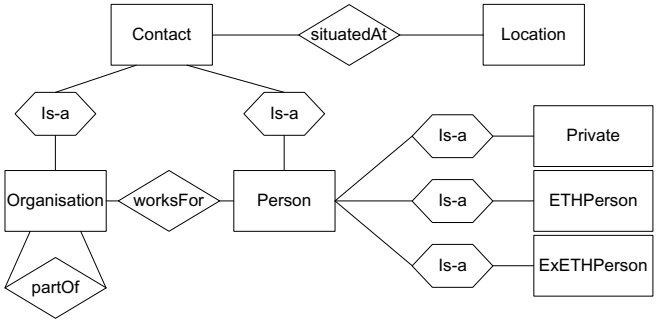

Fig. 1. Contacts Model

We have experimented with different visualisations of both data and metadata. In an earlier version, data was summarised using a tag cloud and metadata collections could be selected by means of drop down menus. In a more recent version, the metadata itself has been visualised by a tag cloud as shown with the screenshot in Figure 2. On the left hand side of the screenshot, the schema tag cloud summarises the names of the various collections of data items within the database. The default is to have the size of the tags represent the relative cardinality of the collection. To distinguish the tag clouds that represent data and metadata, we will refer to these as data clouds and schema clouds, respectivley.

As will be explained later, we implemented several different data adapters on top of relational and object-oriented database systems. A user can start browsing a database either by entering a database-dependent query expression in the window below the tag clouds or by selecting one or more of the tags in the schema cloud. When clicking on a collection tag, the data cloud on the right hand side of the figure is generated. In the example shown in Figure 2, the user has selected the *Contacts* tag in the schema cloud, which triggered the generation

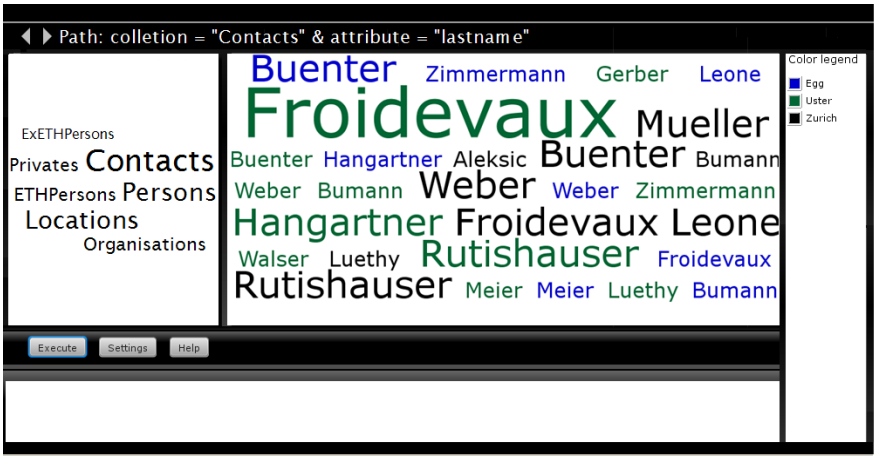

Fig. 2. Schema and Data Browsing

of the data cloud where contacts are summarised according to their last name. This is also indicated by the navigation path shown on top of the screenshot. In addition, contacts can be further grouped by a second attribute, in this case the `city` attribute which is visually represented by different colours. The lighter tag Froidvaux represents the number of objects with last name Froidvaux that live in Uster, while the black tag Froidvaux represents the objects with last name Froidvaux that live in Zurich. As one can see from the colour legend on the right hand side of the figure, each distinct attribute value of the `city` attribute is assigned a specific colour. By default, the first non-unique string attribute of each collection is used for its visualisation as a data cloud. However, the user can also specify this by means of a simple selection of attributes through check boxes. Alternatively, one can display the attributes themselves as a schema cloud and allow the users to select one or more attributes as tags. In this way, we support synchronised browsing across the metadata and data through the adjacent tag clouds.

The size of a tag in the data cloud represents how many data items have that particular attribute value. In this way, a data cloud can be considered as a visualisation of the attribute value frequency. The user can now click on a tag and further refine their selection. When hovering over a `lastname` tag, a user gets detailed information about the number of objects that have this attribute value, or in the case of only a single object, we get the set of attribute values.

A second example of such a user interaction involving the browsing of the schema and the process of refining a query is illustrated in Figure 3. In a first step, the user selects the collection to be queried from a menu. Assume in this case it is the `Contacts` collection. Schema browsing allows the user to then select and combine attributes not only from the currently selected data items, but also from those associated with them. Assume that the user is interested in the `Locations` of contacts, particularly the city names where they live. Then from

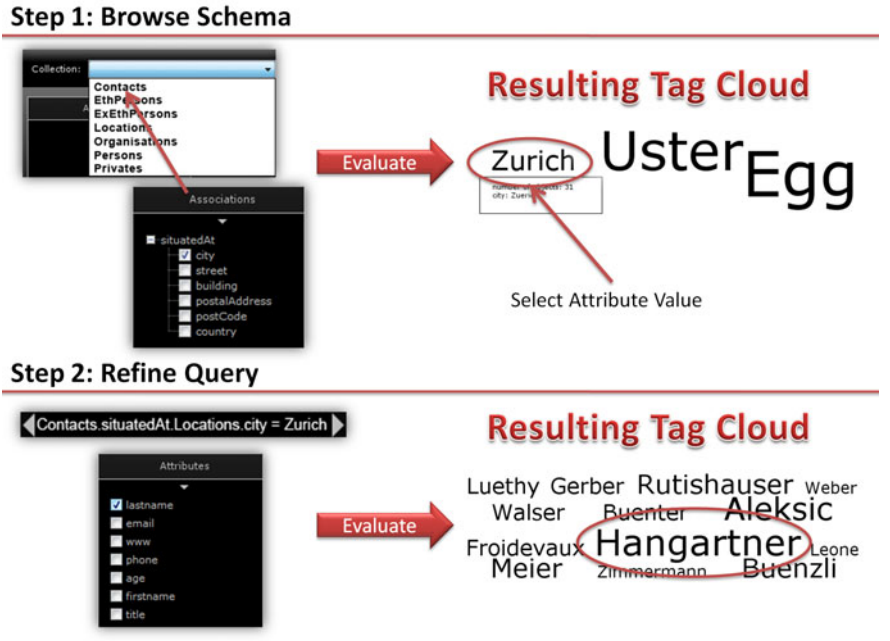

Fig. 3. Interactive Query Formulation

a further menu of associations over `Contacts`, they could select the `SituatedAt` association that relates contacts to locations. The user is then provided with a list of the attributes of `location` objects for them to select. The figure shows the data cloud resulting from a selection of the `city` attribute. The result is displayed as a data cloud whose terms are weighted according to the occurrences of contacts living in the same city. The user now has several possibilities for how to proceed. They could either start exploring other parts of the schema, display a different selection of attributes of the same entities or they can further refine the current query. In the second step, the example illustrates the latter possibility where the query has been constrained to contacts living in Zurich by selecting the corresponding tag within the result of step 1. In this case, the lastname attribute of `Contacts` objects is selected to be displayed in the data cloud. From the resulting data cloud, it is easy for the user to gain an overview of the contacts living in Zurich and to see that Hangartner has the most occurrences.

These two user interaction examples illustrate the kind of queries that can be posed by means of tag cloud browsing. The selection of attributes by the user prior to the creation of the tag cloud corresponds to a concatenation of attribute values before the resulting list is grouped by frequency. A click on one of those tags immediately leads to an extension of the query with a predicate ensuring the currently displayed attribute is fixed to that particular tag value for all subsequent queries. This cycle of attribute selection and predicate generation

can be arbitrarily repeated by the user. However, due to the nature of tag clouds, attributes with a large value range or unique attributes are not very well suited for selection. Furthermore, join operations are supported by explicit traversal of associations when the user navigates between collections. Because of that direct way of interacting with real data, it is not possible to create range queries or any other open queries. For those kinds of queries, we offer a query field. In terms of SQL, the queries that can be answered by our system can be summarised as a selection of concatenated attributes retrieved by (possibly) nested joins with WHERE clauses that contain AND-chained equality predicates. The results are then grouped by frequency and displayed as a tag cloud.

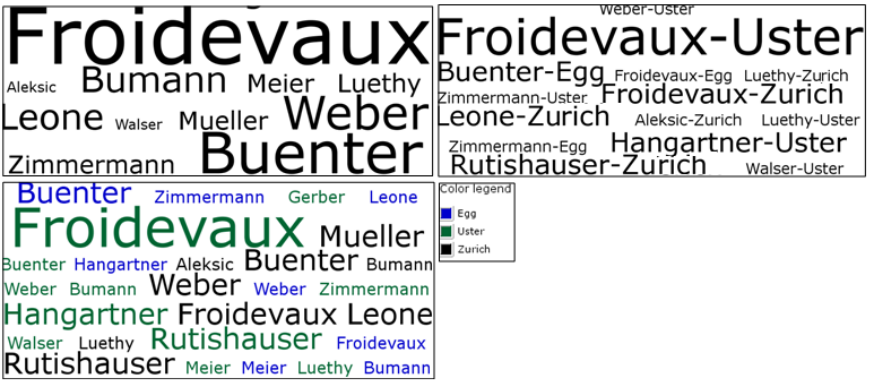

Fig. 4. Exploiting tag cloud features

As already mentioned, we offer different modes for attribute value visualisation and this is depicted in Figure 4. In the data cloud in the upper-left corner, the `Contacts` collection is visualised by last names. In the data cloud in the upper-right corner, the tag content consists of a combination of two attributes, namely the attribute `lastname` of `Contacts` as well as the attribute `city` of the associated location objects. The tags thus represent the number of contacts with a given name that live in the same city. In this example data set, the tag Froidvaux-Zurich represents the set of contacts with last name Froidvaux who live in Zurich. As one can see in this example, more people with the name Froidvaux live in Uster than in Zurich. For the data cloud in the lower-left corner of Figure 4, we added colour as an additional visualisation dimension. The attribute `lastname` is bound to the tag content, while the attribute `city` from the associated location is bound to the colour feature, as already explained before. Note that care has to be taken in choosing the right attributes to bind to the colour feature since, as mentioned previously, it only makes sense if the distinct set of values is small in order that the colour index is of reasonable size and the tag colours are informative.

4 Architecture and Implementation

Since a key objective of this work was to experiment with different tag cloud visualisations and investigate the extent to which these could be used to replace traditional database query languages, it was important that we developed a framework that could serve as an experimental platform. Further, we wanted to be able to experiment with not only different forms of interfaces, but also different database technologies. We therefore implemented a general framework for visualising data from various data sources. The framework is implemented in Java with a JavaFX[2] graphical user interface (GUI). Figure 5 gives an overview of the system architecture. The manager component is the heart of the system and responsible for handling requests from the GUI, forwarding these to the database through the database adapter and invoking the visualisation manager to transform the results into the appropriate visual elements to be returned and displayed in the GUI.

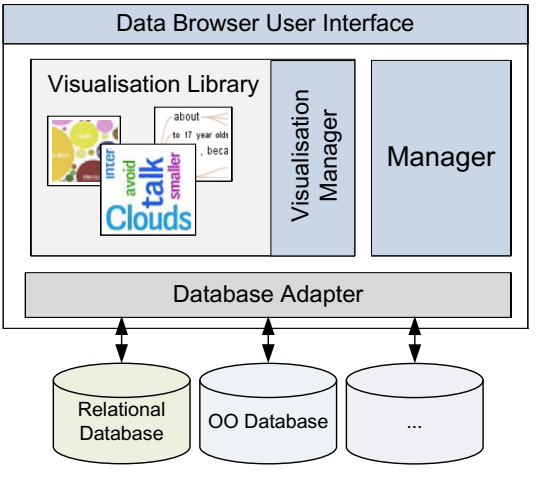

Fig. 5. System Architecture

Our framework is extensible in multiple ways. Firstly, we provide a data adapter interface which can be implemented for any data source. At the moment, we have an implementation for the object databases db4objects[3], OMS Avon[4] and OMSPro[5] as well as a MySQL implementation. Secondly, the visualisation manager can manage different kinds of visualisation techniques. Therefore, we provide a visualisation interface which has to be implemented to add a new technique to the visualisation library. We currently provide two tag cloud versions, an earlier version where metadata is selected from a drop down menu and

[2] http://javafx.com/
[3] http://www.db4o.com/
[4] http://maven.globis.ethz.ch/projects/avon/
[5] http://www.globis.ethz.ch/research/oms/platforms/omspro

a later version that offers synchronised browsing of data and metadata. We have implemented the tag cloud generation algorithm based on [12]. In addition, we are currently working on a bubble chart visualisation.

Our data browser application is flexible and configurable and is currently used as a platform for experimentation in our research group. We have for example integrated it into an ambient information system where researchers were profiled using tag clouds [3].

5 User Study

We undertook a preliminary user study to evaluate our system and test the hypothesis that tag clouds are an easy, suitable and intuitive way for users to browse structured data. We recruited ten participants that regularly browse the Web, from which five were computer science professionals and the other five were regular end users. The five computer science professionals included three undergraduate students, a PhD student and a software engineer. The non-expert users' professions were very diverse, including a carpenter, a shop assistant and a human resource and supply chain manager. The ages ranged from 23 to 58 with three participants in their 50s.

The user study was carried out in individual sessions with each participant. In the first part of each session, we asked the participants for their personal details such as name, age and profession. Additionally, we recorded their computer skills. We were interested in their level of expertise in relational and object database techniques and whether they were familiar with the concept of tag clouds.

In the second part of the session, a participant received a short oral introduction to our data browser. We briefly described the concept of tag clouds to the ones that were not familiar with it and showcased the use of the browser by means of a simple example where the colour feature was used. These oral explanations were intentionally kept very short, since one of our goals was to show that our browser is easy and intuitive to use. The participants were then asked to solve five tasks autonomously by using the data browser. Each of these tasks was targeted at a different degree of difficulty and different concepts of our data browser were needed to answer the question. There was no restriction in the time available to the participants to complete the task.

The participants were asked to browse the contacts database presented in Sect. 3 and find results to the following queries:

- How many persons from ETH have "Bunter" as their last name?
- How many of these persons live in Zurich?
- Get the email addresses of the contacts named "Leone"?
- Where are these contacts situated?
- What is the most frequent age of all contacts?

Finally, the participants had to fill out a questionnaire where we asked about their impressions and experiences with the browser. Questions included whether the short introduction was sufficient help to solve the tasks, whether the tasks

were difficult to solve, and whether they generally liked the browser. For these questions, they could tick yes or no and provide free text comments.

For the study, we used an early version of the data browser, where the metadata was represented in a drop down menu, from which the user could simply select a collection of objects to be displayed as a data cloud. This is conceptually similar to the synchronised version, where a user selects which collection to represent as a data cloud from the schema cloud.

The participants' previous knowledge and the user study results are summarised in Table 1 and 2. Note that the questions are grouped based on their answers' scale type. For the evaluation of the results, we have divided the participants into two distinct categories, one for the five computer science professionals and one for the five non-experts. We asked them to rate their skills in both relational (RDBMS) and object database (OODBMS) techniques on a scale from 1 to 5 where 1 refers to no skills and 5 to expert skills. While computer scientists were highly skilled in both RDBMS and OODBMS technologies, with average skill levels of 4 and 3.6 respectively, non-experts did not have any previous knowledge in that area, except for one participant who occasionally had to work with an MS Access database. The most surprising fact was that nearly all participants solved the five tasks successfully as indicated in Table 1, question 3. While non-experts accomplished the given tasks successfully, one expert had difficulties with two of the five tasks.

Table 1. Interval Scale

#	Question	Category	User Group	Average	Std. Dev.
2.1	RDBMS Knowledge	Skills	non-experts	1.4	0.55
			computer scientists	4	0.71
2.2	OODBMS Knowledge	Skills	non-experts	1	0
			computer scientists	3.6	1.14
3	Queries	Task	non-experts	5	0
			computer scientists	4.6	0.89

Surprisingly, the familiarity of tag clouds was not as high as we had assumed before the study (Table 2, Question 2.3). While the concept was unknown to all the non-experts, four out of five computer scientists were familiar with tag clouds. This is shown by the significance (p-value) of the CHI-square test that indicates a statistically significant relationship between group membership and tag cloud familiarity. For all the other questions, the CHI-square test showed the a user's expertise had no influence on the answers. Regardless of their computer skills and prior tag cloud knowledge, both experts and non-experts found the provided introduction sufficient for solving the given tasks and none of them found the tasks too difficult. The colour feature was used by five participants, that is three non-experts and two expert users. Nine out of ten participants liked to browse the database with our data browser. One participant thought that the data browser has great potential and another participant commented that he

Table 2. Nominal Scale

#	Question	Category	User Group	Yes	No	CHI-Square (p-value)
2.3	Tag Cloud Concept	Skills	non-experts	0	5	0.048
			computer scientists	4	1	
4.1	Sufficient Help	Feedback	non-experts	5	0	1
			computer scientists	4	1	
4.2	Task difficulties	Feedback	non-experts	0	5	1
			computer scientists	0	5	
4.3	Colour Feature	Feedback	non-experts	3	2	1
			computer scientists	2	3	
4.4	Liked browser	Feedback	non-experts	4	1	1
			computer scientists	5	0	

finds the query refinement process very useful. A third participant mentioned that she liked the browser, mainly because one can filter the data by selections, instead of having to write queries.

6 Discussion

Our preliminary user study has shown promising results regarding our hypothesis that the use of tag clouds for the browsing and querying of structured data is easy and intuitive. All non-expert users were able to complete all of the given tasks successfully, even though they were not familiar with the concept of tag clouds before participating in the study. Interestingly, the only user who failed to complete the tasks was a computer science professional who failed to correctly complete two out of the five tasks. These results show that our data browser approach has the potential for providing non-experts with an easy and user-friendly means of accessing traditional databases without prior knowledge of a query language and also the actual database schema. Indeed, these are initial findings only and an extended user study has to be carried out that measures efficiency and effectiveness of our approach compared to conventional tools before our hypothesis can be fully validated. However, we believe that there are a number of open issues that merit further investigation. One of these is the question of how best to exploit tag features other than size, such as the tag colour, to help the user to browse the data. Since only 50% of the participants from both user groups made use of the colour feature to solve the tasks, there is no significant dependency between the use of this feature, the user groups and the accomplishment of the given tasks. This feature would have to be evaluated in a separate study, where more emphasis would be put on exploiting the tag colour for query answering. In addition, we already noted that care has to be taken when using the colour feature, since its use is only helpful if the set of distinct attribute values for a given attribute is rather small and this is something of which users would have to be aware. One participant mentioned that the colour feature is not intuitive and he had problems understanding it in the beginning.

Other issues to investigate include that of using schema clouds alongside data clouds and whether users would find this more intuitive than selection from menus. In fact, it remains to be seen just how much could be expressed using only tag clouds without the need for menus and check lists and whether this would be beneficial or detrimental to the ease of use.

An interesting hypothesis for further investigation is the question of whether the use of tag clouds would enhance developers' efficiency and effectiveness. While we have shown that non-experts and experts are equally capable of solving a set of given tasks using our browser, we only relied on the use of tag clouds without involving any querying. In an additional study targeted at expert users, one could experiment with a combination of a traditional query interface and tag clouds, where a developer could familiarise themself with the schema and data by browsing it using the tag cloud and iteratively constructing queries. One could then test and compare the quality and time used to construct a query using only a query language with the combined approach. In fact, one of the expert users commented that he likes the option for querying and that the study would have been more meaningful to him if they could first solve the tasks with a traditional database interface and then compare it to the data browser.

While the study reported here focused on browsing structured data sets, we are now experimenting with the browsing of both structured and semi-structured data in terms of a publications database. Based on this, we would like to compare our approach with previous approaches such as [9] where synchronised tag clouds were used to explore semi-structured clinical trial data.

Finally, we note that while our work has focussed mainly on object databases to date, concepts similar to those proposed for data clouds in [6,7] could be adopted to return entities rather than attribute values for specific applications relying on relational databases.

7 Conclusion

We have presented a data-driven approach to the browsing and querying of databases based on the visualisation of both data and metadata as tag clouds. We also reported on a preliminary user study involving both expert and non-expert users which demonstrated the feasibility of the approach. While this initial study provided very positive feedback, we are well aware that there are many open issues and consider the results to be a positive indicator of the potential of tag clouds as a visualisation scheme rather than as validation of a paticular interface design. We therefore plan to further experiment with different tag cloud visualisations as well as how these can be combined into a database browsing interface for both semi-structured and structured data.

References

1. Bateman, S., Gutwin, C., Nacenta, M.: Seeing Things in the Clouds: The Effect of Visual Features on Tag Cloud Selections. In: Proc. ACM Conf. on Hypertext and Hypermedia (HT 2008), pp. 193–202 (2008)

2. Farrell, S., Lau, T., Nusser, S., Wilcox, E., Muller, M.: Socially augmenting employee profiles with people-tagging. In: Proc. ACM Symposium on User Interface Software and Technology (UIST 2007), pp. 91–100 (2007)
3. de Spindler, A., Leone, S., Geel, M., Norrie, M.C.: Using tag clouds to promote community awareness in research environments. In: Luo, Y. (ed.) CDVE 2010. LNCS, vol. 6240, pp. 3–10. Springer, Heidelberg (2010)
4. Jagadish, H.V., Chapman, A., Elkiss, A., Jayapandian, M., Li, Y., Nandi, A., Yu, C.: Making Database Systems Usable. In: Proc. ACM SIGMOD Intl. Conf. on Management of Data (SIGMOD 2007), pp. 13–24 (2007)
5. Catarci, T., Costabile, M.F., Levialdi, S., Batini, C.: Visual query systems for databases: A survey. Journal of Visual Languages & Computing 8(2), 215–260 (1997)
6. Koutrika, G., Zadeh, Z.M., Garcia-Molina, H.: Data Clouds: Summarizing Keyword Search Results over Structured Data. In: Proc. 12th Intl. Conf. on Extending Database Technology (EDBT 2009), pp. 391–402 (2009)
7. Koutrika, G., Zadeh, Z.M., Garcia-Molina, H.: CourseCloud: Summarizing and Refining Keyword Searches over Structured Data. In: Proc. 12th Intl. Conf. on Extending Database Technology (EDBT 2009), pp. 1132–1135 (2009)
8. Kuo, B.Y.L., Hentrich, T., Good, B.M., Wilkinson, M.D.: Tag Clouds for Summarizing Web Search Results. In: Proc. Intl. Conf. on World Wide Web (WWW 2007), pp. 1203–1204 (2007)
9. Hernandez, M.E., Falconer, S.M., Storey, M.A., Carini, S., Sim, I.: Synchronized tag clouds for exploring semi-structured clinical trial data. In: Proc. Conf. of the Center for Advanced Studies on Collaborative Research (CASCON 2008), pp. 42–56 (2008)
10. Rivadeneira, A.W., Gruen, D.M., Muller, M.J., Millen, D.R.: Getting our Head in the Clouds: Toward Evaluation Studies of Tag Clouds. In: Proc. Intl. Conf. on Human Factors in Computing Systems (CHI 2007), pp. 995–998 (2007)
11. Schrammel, J., Leitner, M., Tscheligi, M.: Semantically Structured Tag Clouds: An Empirical Evaluation of Clustered Presentation Approaches. In: Proc. Intl. Conf. on Human Factors in Computing Systems (CHI 2009), pp. 2037–2040 (2009)
12. Kaser, O., Lemire, D.: Tag-Cloud Drawing: Algorithms for Cloud Visualization. In: Tagging and Metadata for Social Information Organization Workshop, in Conjunction with WWW 2007 (2007)

A Goal-Oriented Requirements Engineering Method for Business Processes

Ken Decreus and Geert Poels

Faculty of Economics and Business Administration, Ghent University, Belgium
{ken.decreus,geert.poels}@ugent.be

Abstract. Central to the development of BPMS technology was the promotion of a new language, Business Process Modelling Notation (BPMN). The primary goal of BPMN is to provide a common language for describing process behaviour, shareable by business and IT, which includes business users, business analysts, and technical developers. What seems to be missing in the way that business users are supposed to use BPMN, is an explicit consideration of the strategic rationale of having certain business processes as well as support for describing business processes in terms familiar to business people. We extended current work on Goal-Oriented Requirements Engineering (GORE) for business process design, i.e., B-SCP framework [1] and the work of Lapouchnian et al. [2], in order to obtain an appropriate GORE for BPMN modelling method. Our first contribution is the introduction of a B-SCP metamodel, which has been implemented by means of the Eclipse Modelling Framework. Our second contribution is an Eclipse-based B-SCP editor that enables business users to specify their strategic requirements and operational tasks. Our third contribution consists of model transformations to generate BPMN skeletons out of the B-SCP model, which were implemented by means of the Atlas Transformation Language.

Keywords: Goal-Oriented Requirements Engineering, Business Process Modelling, Business-Strategy Context Process, Atlas Transformation Language.

1 Introduction

In the last three decades, an increasing attention to business process change as a factor of organizational success has been witnessed. In particular from 2000 onwards, Business Process Management (BPM) technologies have gained world-wide popularity [3]. One of the main BPM-enabling technologies is the Business Process Management System (BPMS), which Smith and Fingar [4] define as a modelling, integration, and execution environment for the design, manufacture and maintenance of business processes. The importance of BPMS is illustrated by Gartner [5], who predicts that by 2015 30% of business applications will be developed by means of BPMS technology (the prediction probability was 0,9).

Central to the development of BPMS technology was the promotion of a new language, Business Process Modelling Notation (BPMN), which could be used to represent business processes. As given by the BPMN specification [6], the primary goal of BPMN is "to provide a notation that is readily *understandable* by all business

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 29–43, 2010.
© Springer-Verlag Berlin Heidelberg 2010

users, from the business analysts that create the initial drafts of the processes, to the technical developers responsible for implementing the technology that will perform those processes, and finally, to the business people who will manage and monitor those processes." (p1, [6]). Silver stresses the importance of using BPMN as a *common language* between business and IT. Furthermore, Silver [7] distinguishes different types of BPMN modelling, depending on the user category. Firstly, BPMN Level 1, or *descriptive* modelling, is geared towards the business user and offers a basic set of BPMN elements. Secondly, BPMN Level 2, or *analytical* modelling, supports the business analyst in using the complete BPMN notation to describe the activity flow precisely, including the exception paths. These models should be complete and consistent, but not yet contain technical details to make them executable. Thirdly, BPMN Level 3, or *executable* modelling, allows the technical developers to add process data, service interfaces and human task assignment that are needed to execute the BPMN models using BPMS technology.

When we look at BPMN Level 1, the business user is already expected to understand and work with BPMN concepts such as pool, lane, task, subprocess, start event, stop event, exclusive gateways, parallel gateways, sequence flow, and message flow, which are terms that, maybe apart from task, do not belong to the ordinary language used by business people. It is doubtful whether business people (e.g., accountants, marketers, sales people, auditors, finance officers, stock managers, etc.) think of business processes in terms of 'lanes', 'pools', 'gateways', and 'events'. Havey [8] warns that BPMN is not suited for business users, and stresses the importance of capturing requirements based on an approach that business users can understand. Fernandez et al. [9] confirm this finding and state that BPMN scores low on usability for business users.

Business managers also frequently need to deal with complex real-world problems (e.g., how to react to the entry of a new, low-cost service provider in the market?) that require considering simultaneously high-level strategic requirements and low-level operational details. However, Recker [10, 11] states that BPMN currently lacks concepts to support process decomposition and organisational modelling. Recker [10, 11] suggests to use a different, easier, and more business user adapted approach to process modelling with BPMN, by providing dedicated symbols for placing a process into its organisational and hierarchical context.

What seems to be missing in BPMN Level 1, i.e., the way that business users are supposed to use BPMN, is an explicit consideration of the strategic rationale of having certain business processes as well as support for describing business processes in terms familiar to business people. In attempting to deal with this matter, this paper addresses the following research question:

RQ: How can business users design complex business processes in terms of and in correspondence with strategic requirements?

To answer this research question, we developed a new BPMN approach to business process modelling targeted at business users (i.e., BPMN level 1 as referred to by Silver [7]). We assume that the context of our approach consists of a real-world environment in which there is a strategic interest of business users in the design of the business processes. As Wieringa and Heerkens [12] explain, the solution design phase

proposes an improvement to a problematic situation, and is based on specific solution properties. In our approach, the main solution properties are:

(i) consideration of the strategic rationale of having certain business processes, and having them organized in certain ways
(ii) support for describing business processes in terms familiar to business users and linking these business processes to strategic requirements

This paper is structured as follows. Section 2 provides details on the background of our work. Section 3 shows an overview of our approach and introduces the full implementation details of it. Section 4 offers a discussion about our approach. Section 5 concludes this paper and introduces future work.

2 Background

Our approach heavily relies on previous Goal-Oriented Requirements Engineering (GORE) research, which aimed at developing ways to capture high-level strategic business requirements and use them to drive the system development process. To this end, we investigated [13] the GORE and BPM literature to find studies that apply goal-oriented requirements engineering to business process design. We found that methods that apply GORE techniques for business process modelling, generally lack clear mappings between goal concepts and business process concepts and are short of detailed transformation descriptions (for details of this study, see [13]). Therefore, we were not able to reuse their transformations in our research. Some methods, however, provide a sound basis on which we can build our approach, i.e., the B-SCP framework [1] and the work of Lapouchnian et al. [2], which we will briefly introduce in this section.

To start with, the B-SCP framework [1] is a requirements engineering framework for organizational IT that directly addresses an organization's business strategy and the alignment of IT requirements with that strategy. Goal modelling is used to represent business strategy as requirements, and Jackson context diagrams [14] to represent business and system model context. The strategy and context parts are integrated using a problem diagram framework [14]. Strategy is first elicited using VMOST [15], an organizational alignment analysis technique. Then, an i* goal model [16] is constructed using goal modelling rules for organizational motivation proposed by OMG's Business Motivation Model [17]. To refine requirements from a strategic, high-level problem diagram down to the lowest operational level, a progression of problem diagrams is used to represent this top-down hierarchy. In addition, the combined goal and problem diagrams are briefly mapped to Role Activity Diagrams (RAD) [18], but we did not reuse these mappings due to our earlier findings [13].

Next, Lapouchnian et al. [2] propose a requirements-driven method for configuration of high-variability business processes in terms of business priorities. This method is characterized by textual annotations to add control flow detail to goal models, which we will reuse in this paper. For instance, the sequence annotation (";") can be added to AND decomposition to indicate that all the subgoals are to be achieved in sequence from left to right. As we aim at BPMN Level 1 [7], we only consider annotation of sequential AND decomposition, parallel AND decomposition, and OR decomposition. The annotation of control flow is organised per group of

decomposed requirements (e.g., all sub-requirements of one requirement have a sequential AND decomposition), so it is impossible to have different kinds of control flow annotations in the same group of requirements.

3 Overview of Our Approach

Our approach to BPMN business process modelling for business users consists of four steps. First, the business user applies the original B-SCP method [1, 19] and uses our visual editor to create a B-SCP model. Secondly, the business user decides to elaborate the process aspects of a specific part of the B-SCP model, by adding control flow annotations [2] that are needed for BPMN model generation. Thirdly, the business user uses the computer-based model transformations to generate BPMN process model skeletons, and finally, the business analyst takes the BPMN process model skeleton as input for his work and creates a consistent and complete BPMN business process diagram (compatible with BPMS technology).

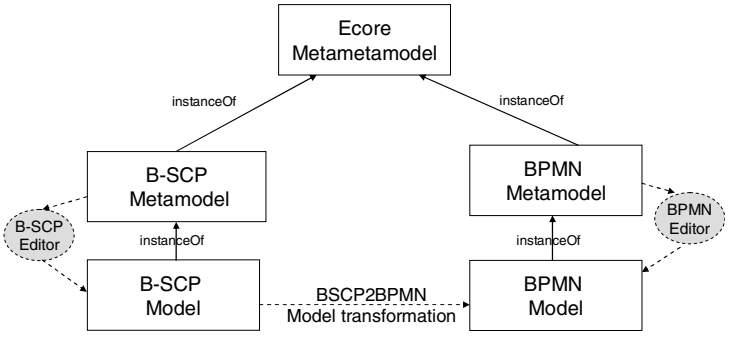

Fig. 1. Overview of GORE for BPMN

To realize our approach, a layered implementation architecture (Fig. 1) was developed in IBM's Eclipse environment [20]. The fundaments of our solution are built upon the different abstraction layers of the OMG Model Driven Architecture [21]. On top, the high-level metametamodel in Ecore (e.g., defining elementary constructs like Class and Relationship) is used to define medium-level metamodels (e.g., containing the instance of a Class called Goal), of which models are defined on the lowest-level (e.g., containing the instance of a Goal called 'Shorten Cash Cycle'). In this paper, we use two different medium-level metamodels, i.e., one metamodel to define strategic business requirements and context (B-SCP) and another metamodel to represent business processes (BPMN). Both metamodels have associated tool support to allow users to visually edit model instances.

The properties of our approach are supported by our implementation as follows. The consideration of the strategic rationale of having certain business processes, and having them organized in certain ways, is supported by our B-SCP metamodel and corresponding B-SCP editor to create B-SCP models. The support for describing business processes in terms familiar to business users and linking these business

processes to strategic requirements is given by the annotation of control flow via the B-SCP editor and by means of the BSCP2BPMN model transformations.

Section 2.1 introduces B-SCP terminology and explains how we created the B-SCP metamodel. Section 2.2 shows the B-SCP editor and clarifies how business users add control flow annotations to B-SCP models in the B-SCP editor. Section 2.3 provides insights into the model transformations from B-SCP to BPMN. To illustrate the concepts, we use a fictive car rental company (EU-Rent) as running example.

3.1 B-SCP Metamodel

The syntax of a language is determined by the set of symbols that compose the language as well as the rules for forming valid combinations of these symbols [22]. The original work on B-SCP [1, 19] defines a set of symbols based on the i* goal language [16] and Jackson problem frames [14], and informally explains the rules for forming valid combinations of these symbols. Our work introduces a B-SCP metamodel in Ecore (Fig. 2) to define the abstract syntax of the B-SCP language.

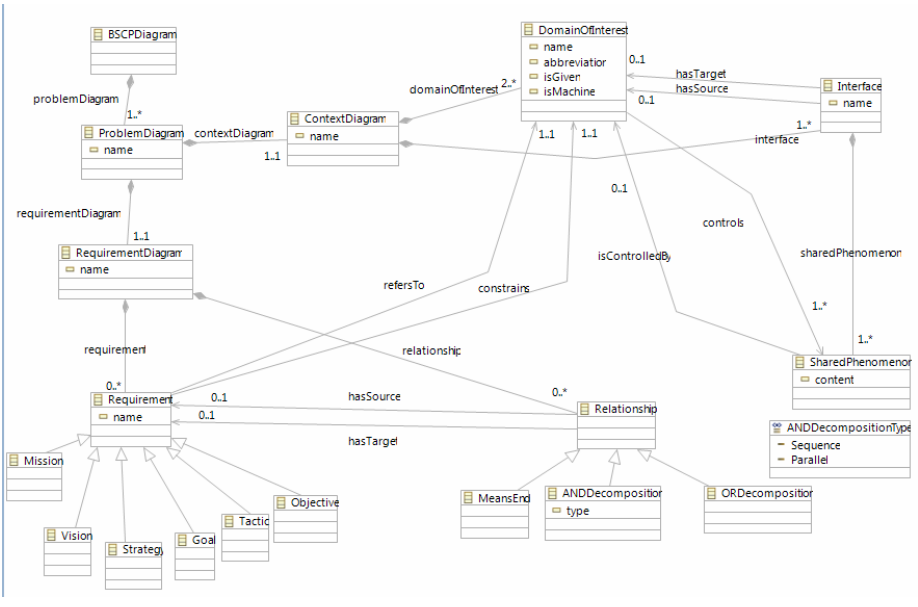

Fig. 2. B-SCP Metamodel in Ecore

A *BSCPDiagram* contains one or more *ProblemDiagrams*, which have each exactly one *RequirementDiagram* and *ContextDiagram*. A *ProblemDiagram* may refine elements of another *ProblemDiagram*, which makes a *BSCPDiagram* an hierarchical structure of *ProblemDiagrams*.

A *RequirementDiagram* can contain many *Requirements*, where *Requirement* is a generalization of *Mission*, *Vision*, *Strategy*, *Goal*, *Tactic* and *Objective*. As defined by the OMG's Business Motivation Model [17], a *Vision* describes the future state of the enterprise, without regard to how it is to be achieved, and *Mission* indicates the

ongoing activity that makes the vision a reality. For instance, EU-Rent could have a vision to 'Be the car rental brand of choice for business users', and a mission to 'Provide car rental service across Europe for both business and personal customers'. Next, a *Goal* indicates what must be satisfied on a continuing basis to effectively attain the vision, and a *Strategy* is a long-term activity designed to achieve a goal. For instance, the goal 'Be a premium brand car rental company' tries to attain EU-Rent's vision, and a strategy 'Target major airports to find business users' supports the achievement of the EU-Rents' goals. Finally, an *Objective* is a specific and measurable statement of intent whose achievement supports a goal, and a *Tactic* is a short-term action designed to achieve an objective. For instance, the objective 'Be rated by AC Nielsen in top 6 car rental companies' supports the EU-Rent's goal to be a premium brand, and the tactic 'Encourage rental extensions' would be a short-term action to score better in listings such as AC Nielsen.

Requirements described in *RequirementDiagrams* are interconnected via *Relationships*, such as *MeansEnd*, *ORDecomposition,* and *ANDDecomposition*. A *MeansEnd* link indicates a relationship between an end and a means for attaining it [1]. For instance, the vision 'Be the car rental brand of choice for business users' is an end supported by its mission 'Provide car rental service across Europe for both business and personal customers' as means. Next, an *ORDecomposition* link indicates that a requirement is fulfilled if at least one of the lower-level requirements are fulfilled [1]. For instance, a tactic 'Handle Rental Extensions' could be fulfilled by lower-level tactics such as 'Use own staff to extend rental' or 'Use airport staff to extend rental'. Finally, an *ANDDecomposition* link indicates that a requirement is fulfilled if all lower-level requirements are fulfilled [1]. In this paper, we distinguish between sequential and parallel fulfilment of *ANDDecomposition* links. For instance, the tactic 'Encourage rental extensions' can be decomposed into two sequential tactics, of which 'Persuade airport customers' is the first in time and 'Handle rental extensions' is the second. In contrast, the tactic 'Persuade airport customers' might be decomposed into parallel tactics that can be executed simultaneously, such as 'Offer extra flight miles' and 'Offer free cabrio upgrade'.

A *ContextDiagram* contains at least two *DomainsOfInterest* and at least one *Interface* to connect a pair of *DomainsOfInterest*. For instance, *DomainOfInterest* EU-Rent has an interface with *DomainsOfInterest* business customer, personal customer and airport. An *Interface* should contain at least one *SharedPhenomenon* that is controlled by a specific *DomainOfInterest*. For instance, domains EU-Rent and airport might share phenomena such as airport location, welcoming of customer, or holiday season.

A domain of interest in the context diagram *describes* a part of the real-world, whereas a requirement *prescribes* the domain of interest in the context diagram. The connection between requirements and context is made by using the *refersTo* and *constrains* relations from a *Requirement* to a *DomainOfInterest*. For instance, the requirement 'Be the car rental brand of choice for business users' *refers to* domain EU-Rent, as this requirement involves the EU-Rent domain without constraining the way that EU-Rent becomes the car rental brand of choice. In contrast, the requirement 'Use own staff' *constrains* the domain EU-Rent Airport centre in making the staffing planning, as this requirement restricts the way that EU-Rent Airport organizes its staffing.

3.2 B-SCP Editor

In order to instantiate B-SCP models from the B-SCP metamodel, modellers need an intuitive and graphical environment. The *graphical modelling framework* [23] project takes an Ecore metamodel (such as our B-SCP metamodel) as an input and offers a step-by-step approach to generate a fully functional graphical editor (details can be found at [24]). In Fig. 3, the running example of EU-Rent is visualised in the Eclipse-based B-SCP editor.

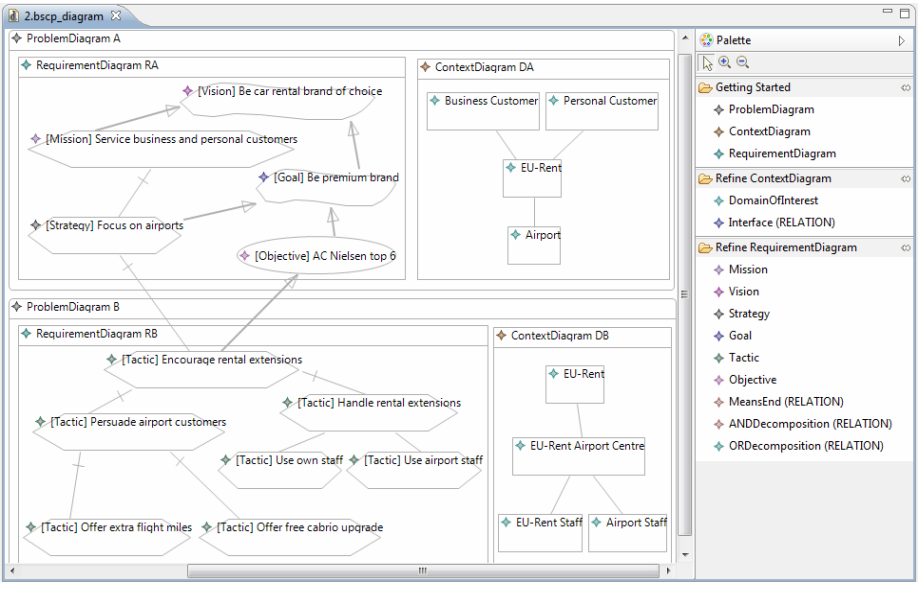

Fig. 3. Eclipse-based Visual B-SCP Editor

At some point in the problem diagram hierarchy, operational details about the business processes are specified. When a business user creates a B-SCP *ProblemDiagram* that solely exists of tactics, we consider this problem diagram to represent (a part of) a business process, so it becomes useful to add control flow annotations. Although Lapouchnian et al. [2] recommend textual annotations to add control flow detail to requirement models, we choose to add such annotations via the *properties* pane of the B-SCP editor to lower the visual complexity of the models. For instance, Fig. 4 shows an OR Decomposition, Fig. 5 illustrates how an AND Decomposition is annotated with sequence, and Fig. 6 displays the parallel AND Decomposition annotation. Note that Lapouchnian et al. [2] advises that control flow annotations should be consistent per group of tactic decompositions. For instance, Fig. 3 – *Encourage rental extensions* has two sequential AND decompositions, Fig. 3 – *Persuade airport customers* has two parallel AND decompositions, and Fig. 3 – *Handle rental extensions* has two OR decompositions.

Fig. 4. Using OR Decomposition

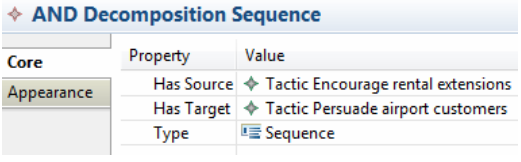

Fig. 5. Setting AND Decomposition to Sequence

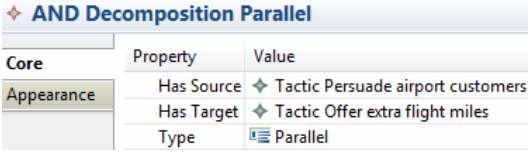

Fig. 6. Setting AND Decomposition to Parallel

3.3 Model Transformation B-SCP to BPMN

When a B-SCP *ProblemDiagram* meets certain transformation criteria, our model transformations can be used to transform this diagram into the skeleton of a BPMN business process diagram that can be further refined by a business process analyst to achieve completeness and consistency. Our transformation criteria are as follows. The requirement diagram (related to the problem diagram to be transformed) should only contain tactics, the top tactic should represent a business process, the control flow annotations should be consistent per group of tactic decompositions, each tactic should refer to or constrain one domain of interest (of the context diagram), and there should be at least one shared phenomenon on each interface between domains of interest (further explanation is given in Section 4.3).

Next, we will elaborate on the B-SCP to BPMN concept mappings that we created, which are implemented by means of the *atlas transformation language* [25] (the implemented rule expressions can be found in Appendix A). In general, Rules 1 to 4 are used to transform the main concepts, Rules 5 to 9 relate to the control flow transformation, and Rule 10 takes care of the generation of message flows. An overview of the BSCP2BPMN transformation rules is given by Table 1.

- Rule 1 transforms a top node in a *RequirementDiagram* (e.g., Fig. 3 – *Encourage rental extensions*) into business process diagram (e.g., the diagram shown in Fig. 7).
- Rule 2 transforms a domain of interest (e.g., Fig. 3 – *EU-Rent*) into a pool, a start event, a sequence edge, and an end event (e.g., Fig. 7 – Labelled with (2)).

- Rule 3 transforms a medium node (e.g., Fig. 3 – *Persuade airport customers*) of a *RequirementDiagram* into a sub-process (e.g., Fig. 7 – Labelled with (3)).
- Rule 4 transforms a leaf node (e.g., Fig. 3 – *Offer extra flight miles*) of a requirement diagram into a task (e.g., Fig. 7 – Labelled with (4)).
- Rule 5 transforms the first occurrence of an OR Decomposition (e.g., Fig. 3 – Link between *Handle rental extensions* and *Use own staff*) into two Gateway Data-Based Exclusive activities and two sequence edges (e.g., Fig. 7 – Labelled with (5)).
- Rule 6 transforms the other occurrences of an OR Decomposition (e.g., Fig. 3 – Link between *Handle rental extensions* and *Use airport staff*) into two sequence edges (e.g., Fig. 7 – Labelled with (6)).
- Rule 7 transforms an AND Decomposition with sequence (e.g., Fig. 3 – Link between *Encourage rental extensions* and *Persuade airport customers*) into two sequence edges (e.g., Fig. 7 – Labelled with (7)).
- Rule 8 transforms the first occurrence of a parallel AND Decomposition (e.g., Fig. 3 – Link between *Persuade airport customers* and *Offer extra flight miles*) into two Gateway Parallel activities and two sequence edges (e.g., Fig. 7 – Labelled with (8)).
- Rule 9 transforms the other occurrences of parallel AND Decomposition (e.g., Fig. 3 - link between *Persuade airport customers* and *Offer free cabrio upgrade*) into two sequence edges (e.g., Fig. 7 – Labelled with (9)).
- Rule 10 transforms a shared phenomenon, between two domains of interest (e.g., Fig. 3 – shared phenomenon x between *EU-Rent* and *EU-Rent Airport Centre*), into a message edge x with a 'send' and 'receive' task and two sequence edges (e.g., Fig. 7 – Labelled with (10)).

Table 1. BSCP2BPMN Concept Mappings

Rule Nr	**B-SCP Concept**	**BPMN Concept**
1	Top node (in Requirement Diagram)	BPMN Diagram
2	Domain Of Interest (in Context Diagram)	Pool
		Start Event
		Sequence Edge
		End Event
3	Medium node (in Requirement Diagram)	SubProcess
4	Leaf node (in Requirement Diagram)	Task
5	OR Decomposition – first occurrence	2 x Gateway Data-Based Exclusive Activity
		2 x Sequence Edge
6	OR Decomposition – other occurrences	2 x Sequence Edge
7	AND Decomposition (Sequence)	Sequence Edge
8	AND Decomposition (Parallel) – first occurrence	2 x Gateway Parallel Activity
		2 x Sequence Edge
9	AND Decomposition (Parallel) – other occurrences	2 x Sequence Edge
10	Shared Phenomenon (between two domains of interest a and b)	'Send' Task (at first pool)
		Sequence Edge (at first pool)
		'Receive' Task (at second pool)
		Sequence Edge (at second pool)
		Messaging Edge from 'Send' to 'Receive' Task

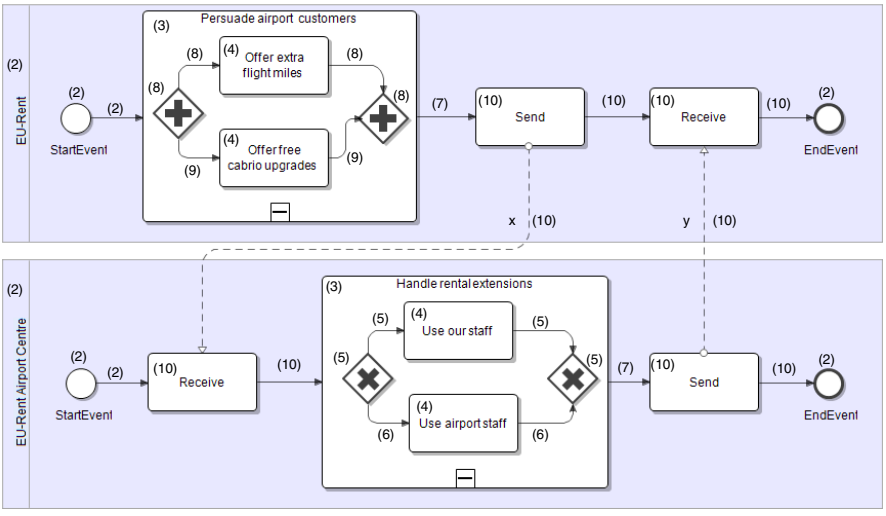

Fig. 7. Resulting BPMN diagram for the B-SCP tactic 'Encourage rental extensions'

4 Discussion

4.1 Using B-SCP as Modelling Aid

Originally, B-SCP was proposed as requirements engineering framework for validating strategic alignment of organisational IT, based on manual interpretation of traceability links between strategy and technology. In this paper, we reused the work on B-SCP for a different purpose, that is as modelling aid for BPMN Level 1, which we interpret as the business process modelling level where business users design complex business processes in terms of and in correspondence with strategic requirements. We believe that B-SCP offers a well-documented and scalable alternative to the currently available modelling methods that combine strategic goals and business processes, which are often i*-based modelling languages [13]. For instance, the B-SCP approach has been successfully implemented at a large e-business initiative of a major Australian financial institution [26]. Few published studies exist on applying the i* goal language into practice, and indications exists that practitioners of large-scale industrial projects are unable to understand i* models well enough to validate the requirements of the system they were building [27]. As the B-SCP framework was proposed to address the known shortcomings of i* and to leverage the existing knowledge of Jacksons's Problem Frames, we considered the B-SCP framework as the starting point of our work.

4.2 Separating Requirements and Business Processes

The differentiation between a goal-oriented requirements language and a business process language is the result of a deliberate design choice. As a modelling language is always conceived with a certain purpose in mind [28], we believe it is easier to

represent goals and business processes using different languages, and to provide model-based translations between these languages, instead of choosing one modelling language to represent both goals and business process concepts. With low modelling complexity (e.g., modelling one clearly understood business process), creating a requirements model could be seen as an overhead cost. But, as real-world business process modelling projects often quickly grow in complexity, business users can use a requirements model as an overview (or one could say, an overarching strategically aligned business process architecture), and generate the required business process models from the requirements model.

4.3 Motivating Our Transformation Criteria

The main motivation of our work is to encourage the technology transfer between RE and industry by lowering the barriers of current GORE research for real-world business users [29]. As a result, our approach tries to find a balance between *rigour* of and *relevance* [30]. Firstly, the restrictions of the B-SCP metamodel and graphical B-SCP editor forces a certain degree of *rigour* in our approach, as the business user must understand and comply to our tool-supported method. Secondly, our approach offers *relevance* to business users by not overloading the business user with technical semantics or formal business rules, as B-SCP is based on strategic management techniques [15] and OMG's general accepted Business Motivation Model (BMM) [17].

Based on these insights, we created our BSCP2BPMN model transformations and decided that OMG's BMM concept *tactic* plays a central role. Instead of forcing the business user in expressing the difference between business processes, subprocesses and activities, the business user just employs *tactics* wherever he specifies how strategic requirements (i.e. *mission*, *vision*, *strategy*, *goal* and *objective*) are or should be achieved. Consequently, the B-SCP model might contain numerous problem diagrams, some without *tactics*, some with both *tactics* and strategic requirements, and some solely consisting out of *tactics*. As our understanding of business process [31] does not allow business users to embed strategic requirements into business process (but only allow users to refer these goals via i* decompositions), only the problem diagrams that solely consists of *tactics* can be syntactically transformed to BPMN skeletons. Finally, to avoid the generation of non-linked BPMN lanes and activities, our transformation criteria state that each tactic should refer to or constrain one domain of interest, and there should be at least one shared phenomenon on each interface between domains of interest.

5 Conclusion, Limitations and Future Work

Central to the development of BPMS technology was the promotion of a new language, Business Process Modelling Notation (BPMN). The primary goal of BPMN is to provide a common language for describing process behaviour, shareable by business and IT, which includes business users, business analysts, and technical developers. What seems to be missing in the way that business users are supposed to use BPMN, is an explicit consideration of the strategic rationale of having certain

business processes as well as support for describing business processes in terms familiar to business people. This paper presents an approach that allows business users to design complex business processes in terms of and in correspondence with strategic requirements. The main claim of this work is two-fold:

(i) The consideration of the strategic rationale of having certain business processes, and having them organized in certain ways, is important for a business user during business process modelling. By extending the work of Bleistein et al. [1], we created a B-SCP metamodel and offered the business user a graphical B-SCP editor (that corresponds to the B-SCP metamodel) to express the strategic rationale and the business processes related to the strategic rationale.
(ii) Support for describing business processes in terms familiar to business users and linking these business processes to strategic requirements. By extending the work of Lapouchnian et al. [2], business users can annotate control flow via the B-SCP editor. Then, specific parts of the B-SCP models can be transformed in corresponding BPMN skeletons by means of the BSCP2BPMN model transformations.

The main limitation of our approach is the lack of full-scale validation and the absence of the reverse transformation (from BPMN to B-SCP). Firstly, a full-scale validation is needed to evaluate the properties of our approach, and to investigate whether these properties contribute positively to BPMN modelling for business users. In order to tackle this shortcoming, we are in the process of applying the Seven-Eleven Japan [32] case exemplar to our approach to investigate the feasibility, and we are conducting case study research [33] at two organisations to discover the added value of our approach. Secondly, our work presents a top-down modelling method, which enables business users to transform parts of B-SCP models into BPMN skeletons, but the reverse transformation (from BPMN to B-SCP) is currently not supported. To this end, our future work will investigate how the PRiM method of Grau et al. [34] could be reused to generate B-SCP requirements from BPMN diagrams. In this context, we will investigate how existing AS-IS or TO-BE business process models can be combined with AS-IS or TO-BE information in B-SCP models. Next, we plan an empirical evaluation of the B-SCP and BPMN editors in terms of their usability, and verification of whether the B-SCP editor has a complete correspondence with the B-SCP method.

References

1. Bleistein, S.J., Cox, K., Verner, J., Phalp, K.T.: B-SCP: A requirements analysis framework for validating strategic alignment of organizational IT based on strategy, context, and process. Information and Software Technology 48, 846–868 (2006)
2. Lapouchnian, A., Yu, Y., Mylopoulos, J.: Requirements-Driven Design and Configuration Management of Business Processes. Business Process Management, 246–261 (2007)
3. Harmon, P.: Business Process Change, Second Edition: A Guide for Business Managers and BPM and Six Sigma Professionals. Morgan Kaufmann, San Francisco (2007)
4. Smith, H., Fingar, P.: Business Process Management (BPM): The Third Wave (2004)

5. Woods, J., Genovese, Y.: Delivery of Commodity Business Applications in a BPMS Does Not Mean You Should Customize the Applications. Gartner Research G00139084 (2006)
6. OMG: BPMN 1.2 specification (2009), http://www.omg.org/spec/BPMN/1.2/
7. Silver, B.: BPMN Method and Style. Cody-Cassidy Press (2009)
8. Havey, M.: Keeping BPM simple for business users: power users beware. BPTrends (January 2006)
9. Fernández, H.F., Palacios-González, E., García-Díaz, V., Pelayo G-Bustelo, B.C., Sanjuán Martínez, O., Cueva Lovelle, J.M.: SBPMN – An easier business process modeling notation for business users. Computer Standards & Interfaces 32, 18–28 (2010)
10. Recker, J.: Opportunities and constraints: the current struggle with BPMN. Business Process Management Journal 16, 181–201 (2010)
11. Muehlen, M., Recker, J.: How much language is enough? Theoretical and practical use of the business process modeling notation. In: Bellahsène, Z., Léonard, M. (eds.) CAiSE 2008. LNCS, vol. 5074, pp. 465–479. Springer, Heidelberg (2008)
12. Wieringa, R., Heerkens, J.: The methodological soundness of requirements engineering papers: a conceptual framework and two case studies. Requirements Engineering 11, 295–307 (2006)
13. Decreus, K., Snoeck, M., Poels, G.: Practical Challenges for Methods Transforming i* Goal Models into Business Process Models. In: 17th IEEE International Requirements Engineering Conference, Atlanta (2009)
14. Jackson, M.: Problem Frames: Analysing and Structuring Software Development Problems. Addison-Wesley, New York (2000)
15. Sondhi, R.: Total Strategy. Airworthy Publications International Ltd. (1999)
16. Yu, E.: Towards modelling and reasoning support for early-phase requirements engineering. In: Proceedings of the Third IEEE International Symposium on Requirements Engineering, pp. 226–235 (1997)
17. OMG: Business Motivation Model (2009), http://www.omg.org/spec/BMM/
18. Ould, M.A.: Business Processes: Modelling and Analysis for Reengineering and Improvement. Wiley, Chichester (1995)
19. Bleistein, S.J., Cox, K., Verner, J.: Validating strategic alignment of organizational IT requirements using goal modeling and problem diagrams. JSS 79, 362–378 (2006)
20. Eclipse: EMF - Eclipse Modelling Framework, http://www.eclipse.org/modeling/emf/
21. OMG: Model Driven Architecture (2009), http://www.omg.org/mda/
22. Guizzardi, G.: On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta) Models. In: Proceeding of the 2007 Conference on Databases and Information Systems IV: Selected Papers from the Seventh International Baltic Conference DB & IS 2006. IOS Press, Amsterdam (2007)
23. Eclipse: GMF - Graphical Modeling Framework, http://www.eclipse.org/modeling/gmf/
24. Decreus, K.: Google Project Site (2010), http://code.google.com/p/bscp2bpmn/
25. Eclipse: ATL - ATLAS Transformation Language, http://www.eclipse.org/m2m/atl/
26. Cox, K., Bleistein, S.J., Reynolds, P., Thorogood, A.: A contingency view of organizational infrastructure requirements engineering. In: ACM SAC OE, France (2006)

27. Maiden, N.A.M., Jones, S.V., Manning, S., Greenwood, J., Renou, L.: Model-driven requirements engineering: Synchronising models in an air traffic management case study. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, pp. 368–383. Springer, Heidelberg (2004)
28. Mylopoulos, J.: Information modeling in the time of the revolution. Information Systems 23, 127–155 (1998)
29. Kaindl, H., Brinkkemper, S., Bubenko Jr., J.A., Farbey, B., Greenspan, S.J., Heitmeyer, C.L., Leite, J.C.S.d.P., Mead, N.R., Mylopoulos, J., Siddiqi, J.: Requirements Engineering and Technology Transfer: Obstacles, Incentives and Improvement Agenda. Requirements Engineering 7, 113–123 (2002)
30. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design Science in Information Systems Research. MIS Quarterly 28, 75–105 (2004)
31. Hammer, M., Champy, J.: Reengineering the corporation : a manifesto for business revolution. Harper Business, New York (1994)
32. Nagayama, K., Weill, P.: Seven Eleven Japan: Reinventing the Retail Business Model. CISR Working Paper 338 - MIT Sloan WP 4485-04 (2004)
33. Yin, R.K.: Case Study Research: Design and Methods, 4th edn., vol. 5. SAGE Publications, Thousand Oaks (2009)
34. Grau, G., Franch, X., Maiden, N.A.M.: PRiM: An i*-based process reengineering method for information systems specification. Information and Software Technology 50, 76–100 (2008)

Appendix A: BSCP2BPMN

```
(1) rule TopNode2BPMNDiagram{
     from a : BSCP!Tactic (a.isUpperTactic())
     to b : BPMN!BpmnDiagram(name <- a.name)}

(2) rule DomainOfInterest2Pool{
     from a : BSCP!DomainOfInterest
     to b : BPMN!Pool(name <- a.name),
        startevent : BPMN!Activity(activityType <- 'EventStartEmpty'),
        endevent : BPMN!Activity(activityType <- 'EventEndEmpty'),
        firstSequence : BPMN!SequenceEdge}

(3) rule MediumNode2SubProcess{
     from a : BSCP!Tactic (a.isMiddleTactic())
     to b : BPMN!SubProcess(name <- a.name)}

(4) rule LeafNode2Task{
     from a : BSCP!Tactic (a.isBottomTactic())
     to b : BPMN!Activity(activityType <- 'Task', name <- a.name)}

(5) rule ORDecomposition_FirstOccurrence{
     from a : BSCP!ORDecomposition(BSCP!ORDecomposition.allInstances()->first())
     to b : BPMN!Activity(activityType <- 'GatewayDataBasedExclusive'),
        c : BPMN!SequenceEdge(iD <- 'Left Conditional Edge'),
        d : BPMN!SequenceEdge(iD <- 'Right Conditional Edge'),
        e : BPMN!Activity(activityType <- 'GatewayDataBasedExclusive'),
        f : BPMN!SequenceEdge(iD <- 'Edge Closing Conditional Construction')}

(6) rule ORDecomposition_OtherOccurrences{
     from a : BSCP!ORDecomposition(not BSCP!ORDecomposition.allInstances()->first())
     to b : BPMN!SequenceEdge(iD <- 'Left Conditional Edge'),
        c : BPMN!SequenceEdge(iD <- 'Right Conditional Edge')}


(7) rule ANDDecomposition_Sequence{
     from a : BSCP!ANDDecomposition(self.type = #SequentialOrder)
     to b : BPMN!SequenceEdge(iD <- 'Sequence Edge')}

(8) rule ANDDecomposition_Parallel_FirstOccurrence{
     from a : BSCP!ANDDecomposition(self.type = #ParallelOrder and
                                    BSCP!ANDDecomposition.allInstances()->first())
     to b : BPMN!Activity(activityType <- 'GatewayParallel'),
        c : BPMN!SequenceEdge(iD <- 'Left Parallel Edge'),
        d : BPMN!SequenceEdge(iD <- 'Right Parallel Edge'),
        e : BPMN!Activity(activityType <- 'GatewayParallel'),
        f : BPMN!SequenceEdge(iD <- 'Edge Closing Parallel Construction')}

(9) rule ANDDecomposition_Parallel_OtherOccurrences{
     from a : BSCP!ANDDecomposition(self.type = #ParallelOrder and not
                                    BSCP!ANDDecomposition.allInstances()->first())
     to b : BPMN!SequenceEdge(iD <- 'Left Parallel Edge'),
        c : BPMN!SequenceEdge(iD <- 'Right Parallel Edge')}

(10) rule SharedPhenomenon2MessagingEdge{
      from a : BSCP!SharedPhenomenon
      to b : BPMN!Activity(activityType <- 'Task', name <- 'Send'),
             BPMN!Activity(activityType <- 'Task', name <- 'Receive'),
             BPMN!SequenceEdge(iD <- 'Sequence Edge'),
             BPMN!SequenceEdge(iD <- 'Sequence Edge'),
             BPMN!MessagingEdge(iD <- 'Messaging Edge')}
```

Foundations of a Reference Model for SOA Governance

Christian Ott[1], Axel Korthaus[2], Tilo Böhmann[3],
Michael Rosemann[2], and Helmut Krcmar[1]

[1] Technische Universität München, Lehrstuhl für Wirtschaftsinformatik, München, Germany
[2] Queensland University of Technology, Business Process Management, Brisbane, Australia
[3] ISS International Business School of Service Management, Hamburg, Germany
christian@coonet.de, axel.korthaus@qut.edu.au,
boehmann@iss-hamburg.de, m.rosemann@qut.edu.au,
krcmar@in.tum.de

Abstract. Although the lack of elaborate governance mechanisms is often seen as the main reason for failures of SOA projects, SOA governance is still very low in maturity. In this paper, we follow a design science approach to address this drawback by presenting a framework that can guide organisations in implementing a governance approach for SOA more successfully. We have reviewed the highly advanced IT governance frameworks Cobit and ITIL and mapped them to the SOA domain. The resulting blueprint for a SOA governance framework was refined based on a detailed literature review, expert interviews and a practical application in a government organisation. The proposed framework stresses the need for business representatives to get involved in SOA decisions and to define benefits ownership for services.

Keywords: Service-Oriented Architecture (SOA), SOA governance.

1 Introduction

Governance has been seen as one of the key success factors of IT for many years and enterprises currently invest considerable resources into the implementation of IT governance frameworks such as Cobit [1, 2]. In their seminal work, [3] define IT governance as the process of "specifying the decision rights and accountability framework to encourage desirable behaviour in the use of IT." The purpose of such a decision rights and accountability framework is to address the three basic questions of IT governance: "What decisions must be made to ensure effective management and use of IT?", "Who should make this decisions?" and "How will these decisions be made and monitored?" [3]. Many enterprises presently face the challenge of developing adequate governance mechanisms for Service-Oriented Architectures (SOAs), which introduce new complexities due to the amount of services to be managed [4]. The SOA paradigm has become widespread and is often considered an important concept to drive the evolution towards an IT architecture focusing on business processes, flexibility and reuse [5, 6, 7]. Moreover, some proponents envision that organisations will begin to open up their architecture to their business ecosystem, i.e. their network of customers, suppliers and even competitors,

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 44–59, 2010.
© Springer-Verlag Berlin Heidelberg 2010

achieving increased interoperability, i.e. the ability to exchange information and to use the information that has been exchanged, through the use of open standards as postulated by the SOA paradigm [8, 9]. The decomposition of today's business applications into reusable business process components that may be marketed to external customers creates novel challenges for IT governance. To date, however, no widely accepted framework for SOA governance has emerged [4]. Given that the lack of a comprehensive governance approach has been cited as the most common reason for failures of post-pilot SOA projects [10], work in this area is highly relevant.

Notwithstanding the urgent need, delineating SOA governance and building a corresponding framework is not an easy task. Already the question of how SOA governance relates to IT governance lacks a consistent answer. While for Malinverno, "SOA governance isn't simply a subset of IT governance" [11], some authors do make this very assumption [12]. For others, SOA governance is an "extension" [13] or "specialisation" [14] of IT governance. In spite of the discussions about a precise definition of the term SOA governance, most authors agree on the basic elements a governance framework should address, namely the organisational structure, processes, policies and metrics [14, 15, 16]. To provide a working definition for the rest of this paper, we build on definitions in [4] and [17]:

SOA governance focuses on the decisions across the entire service lifecycle to enable organisations to realise the benefits of SOA. It is an approach to exercising control and mitigating risk by establishing organisational structures, processes, policies and metrics suitable to ensure that the adoption, implementation, operation and evolution of an SOA is in line with the organisation's strategies and objectives and complies with laws, regulations and best practices.

For reasons of scope, we concentrate on the organisational aspects in this paper by deriving a set of activities and roles that are required in an SOA context and by proposing their responsibilities along the service lifecycle. The resulting framework can guide organisations in designing or evaluating their own governance structure.

The paper is structured as follows. In sections 2 and 3, we point to related work and explicate our research approach. Section 4 outlines the identified activities along the service lifecycle. Section 5 describes the roles involved, to which responsibilities are assigned in section 6. Section 7 includes lessons learned from an application of the framework in a case study. The paper concludes with summary and further research opportunities in section 8.

2 Related Work

The knowledge bases of corporate and IT governance form obvious points of references for research into SOA governance. While from an IT governance perspective, standard works like [3] and well-received frameworks such as Cobit [1] and ITIL [18] are the most prominent examples, the OECD Principles of Corporate Governance are among the most influential guidelines in the area of corporate governance [19].

Academic literature on SOA governance (e.g. [4, 14]) is still relatively scarce, whereas numerous IT solution vendors and analysts have addressed the topic in recent years (e.g. [10, 11, 20, 21, 25, 26, 27]). SOA governance solutions presented by IT vendors tend to address only fragments of a holistic approach. While some IT vendors propose advanced governance models that attend to design-time governance, organisational and business aspects, many of them focus on technical management and run-time governance, i.e. the governance of services that are already in production, and do not address the whole service lifecycle including aspects of adequate design-time governance and higher level corporate governance.

Open standards organisations such as OASIS, OMG and The Open Group have produced a large amount of technical specifications and standards on the subject of SOA, often overlapping in contents. Kreger and Estefan [25] give an overview of the documents for SOA reference models and ontologies, reference architectures, maturity models, SOA modelling profiles, and open standards related to the topic of SOA governance. Not only is the OMG SOA Governance RFP development group [26] exploring the standardisation of SOA governance, the topic has also been included as a chapter in the OASIS Reference Architecture for SOA Foundation [27], and a SOA governance framework is being developed by The Open Group (SOA Governance Framework [28]). These consortia address the topic from different perspectives, and the most promising for the research focus taken in this paper, i.e. organisational aspects of SOA governance, is the work of The Open Group. Their SOA Governance framework, which as at December 2009 has draft status only, however still lacks essential elements such as a specification of detailed accountabilities of roles along the service life cycle.

Bernhardt and Seese [4] propose a conceptual SOA governance framework striving to cover the complete SOA lifecycle. Their approach differs from the one taken in this paper as it uses the standardised OASIS SOA reference model [29] as a starting point for the identification of SOA governance aspects to be considered. They do not make use of empirically tested best practises from related IT governance literature, whereas the framework we propose is primarily derived from the much wider area of successful IT governance frameworks in order to leverage existing knowledge and revise it against the background of SOA-specific characteristics. Bernhardt and Seese [4] have not yet investigated the relationships between their approach and these common IT frameworks.

3 Research Approach

The SOA governance reference framework partly presented in this paper is a “design artefact” in the sense of the design science-based approach to IS research as described in [30]. According to them, IS research is concerned with two design processes, i.e.

- to ‘build’ purposeful artefacts to address heretofore unsolved problems, and
- to ‘evaluate’ these artefacts with respect to the utility provided in solving those problems (cf. Fig. 1).

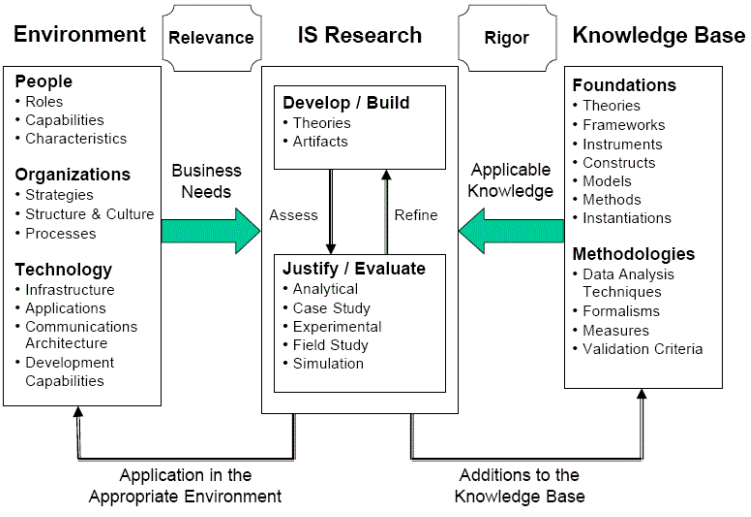

Fig. 1. Information Systems Research Framework according to [30]

Hence, as opposed to behavioural science, design science aims at providing utility and relevance to practice by innovatively designing an artefact that meets an existing business need or "problem" [30]. With regard to the work on SOA governance presented here, the claim of relevance to practice can be justified not only through the Gartner Research study mentioned earlier [10], but also, for example, through a recent SOA governance user survey by Software AG [21]. This survey indicates that most users view SOA governance as important, acknowledge the need for improvement and emphasise the demand for a holistic, business objective-driven lifecycle approach from the start. Rigour in the research process has to be assured by the appropriate application of existing foundations from the knowledge base of the field in the 'build' phase and of suitable methodologies in the 'evaluate' phase [30].

Starting from the existing knowledge base in the 'build' phase of the proposed SOA governance framework, we analysed the widely-used IT governance frameworks Cobit and ITIL and provided an initial evaluation of its utility in a case study in order to derive the core of the SOA governance framework. We selected Cobit and ITIL for the following reasons. Cobit has practically become the global de facto standard for IT control and governance, and most frameworks somewhat align with. ITIL, on the other hand, is a most prominent IT Management framework that primarily defines management and support processes. While ITIL primarily addresses IT efficiency that relates to the effective operation of IT, Cobit is primarily addressing effectiveness and strategy of IT in the context of an organisation, where effectiveness relates to producing a decided, decisive, or desired effect and strategy relates to the strategic planning and adaptation (e.g. of structure or behaviour) that serves the core function of IT to contribute to desired business outcomes.

Mapping the roles and activities proposed by the two frameworks to an SOA environment revealed a need for extensions, as some criteria that are specific to SOAs are not covered in these two popular frameworks (cf. sections 4, 5 and 6). Furthermore, this mapping necessitated a re-naming and re-grouping of activities into a service

lifecycle. In a second step, we conducted a detailed review of literature related to service lifecycle management and SOA governance. Academic articles as well as industry white papers about SOA roles and responsibilities are scarce and often use diverging terminologies or present ideas in an unstructured way, so we focused on the identification of main concepts. We also conducted a series of interviews with carefully selected experts in the field of service management. For the identification of the relevant roles and their responsibilities, we conducted a comprehensive content analysis using published job profiles from Seek.com, Australia's leading recruitment website.

In order to critically evaluate the utility of the framework, we applied it at Landgate, a public sector organisation. Landgate is the Statutory Authority responsible for Western Australia's land and property information and seeks to evolve its IT business applications to implement new services for its clients and to collaborate more closely with partners. The application of the governance framework to Landgate showed how the model supports organisations in identifying new IT management activities when moving into a service-oriented paradigm and which consequences this new paradigm has for the establishment of accountabilities.

4 The Service Lifecycle

4.1 Overview

Cobit and ITIL are very detailed and widely used frameworks that propose a large number of best practises and processes as well as measures, roles and responsibilities to aid management in the planning and organisation, acquisition and implementation, delivery and support, operation, monitoring and evaluation of IT systems. In Cobit alone, there are 197 single steps grouped in 34 processes, which are part of 4 main phases, offering an extensive repository of relevant activities and a highly elaborated set of assignments to roles. Some of the issues covered, such as infrastructure, data or technology and support, will not change significantly independent of the underlying paradigm (e.g. when SOA is replaced by another IT design paradigm) and therefore have not been further analysed. Besides that, the structures of Cobit and ITIL do not allow for an explicit representation of different decision levels. Thus, we looked at management models to find a suitable high-level structure. Drawing from IT-management, we suggest that decision rights can be distributed into distinct layers. Among these, strategy management, portfolio management, program management and project management are mentioned by most authors [31, 32]. Furthermore, operations management had to be considered as well, since governance is not just relevant during the identification and development of services, but for operating services as well (run-time governance). Due to space constraints, this paper covers only three of the five layers (shaded in Fig. 2): service portfolio-, service project- and service operation management.

While acknowledging that there is a broad variety of definitions, we agree with [33] who stress that portfolio management deals with selecting and prioritising the best projects to proceed with. Portfolio management is about choosing the right project, whereas project management is about doing the project right [34]. Hence, in

the portfolio management stage of our proposed framework, the goal is to identify the most relevant services from a larger service portfolio and decide if and when to implement them. Program management, which we do not cover here, represents the connecting link between the two and aligns strategy and execution to deliver the whole SOA by managing interdependent projects [32]. Once a business sponsor has been identified and accepts responsibility for the service, a project is started and the service can be developed. The development process and the publishing or deployment of the service are governed in the service project management stage. Once in place, the operation management of a service covers operation and use, including performance and change management, as well as the retirement phase.

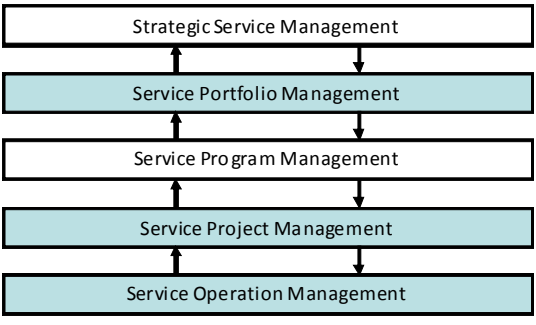

Fig. 2. Layers of management comprising decisions relevant for SOA governance (adapted from [31]). The layers covered in this paper are shaded.

A significant amount of research has been published regarding the lifecycle of a single service (cf. [35] for a comprehensive overview) with a more or less common understanding of what should be part of it. Starting with a service analysis and design phase, most authors include service implementation, service publishing, service operation as well as service retirement or withdrawal. In addition to that, [35] mention a negotiation phase. The latter is primarily relevant if a service or part of its sub-services are provided or sourced externally. For many organisations using SOA today, this is not yet an option but will become more important once emerging service brokers have leveraged the discovery of available services and provide required functions such as pricing and contracting [8]. From today's perspective, it is also not clear to what extent bargaining will happen at all, or if, for example, prices and quality standards are specified solely by the service broker. For these reasons, we have not yet considered negotiation activities in this paper.

4.2 Detailed View

In this section, we focus on the main differences as compared to traditional IT governance by introducing new activities that provide managers with a foundation upon which SOA-related decisions can be based and by discussing those that require changes. Fig. 3 gives an overview and shows how management layers, lifecycle stages and activities are interrelated. The first two columns in Fig. 3 map the three management layers introduced above (service portfolio management, service project

management and service operations management) to the main steps of the service lifecycle (service analysis, service design, service implementation, service publishing, service operation and service retirement). Service analysis occurs both on a portfolio level, e.g. to determine which new services to add to the service portfolio, and on the service project level in order to define the requirements for a single service to be built in the course of the service project. Service design, implementation and publishing are all part of a service project and therefore can be mapped to the service project management level. Service operation and retirement are mapped to the service operations management layer here, although service retirement could also be seen as belonging to the service portfolio view, as it typically requires the assessment of the service to be retired in relation to the other services in the portfolio. The third column of Fig. 3 refers to SOA-specific activities that differ from conventional IT governance-related activities. These activities are part of the service lifecycle step they are aligned with in the diagram (column two). In the following, Fig. 3 will be discussed in detail.

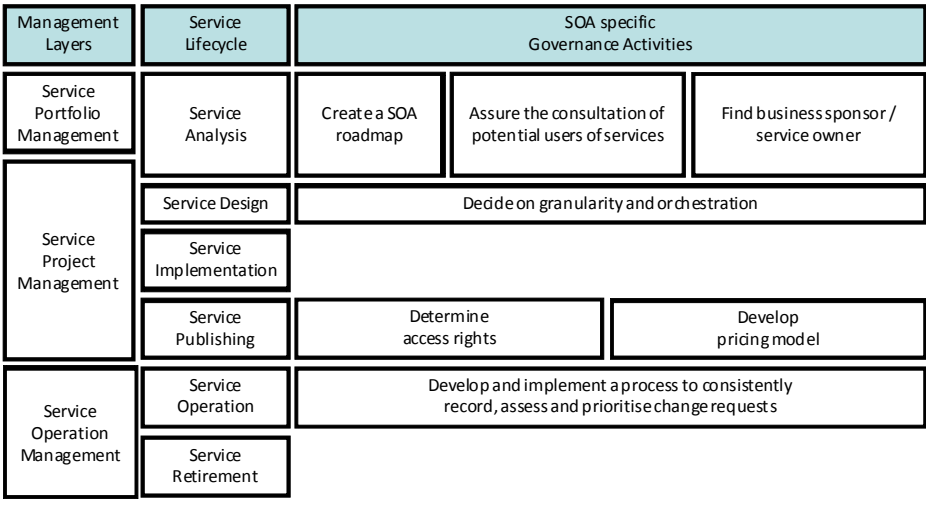

Fig. 3. Interrelationship of management layers, lifecycle stages and the activities addressed in this paper

4.2.1 Service Portfolio Management

As a first step within the service portfolio management phase, a service roadmap is developed by identifying and prioritising service candidates (e.g. by analysing business processes). The proposed services are subsequently analysed further. In this step, all potential users should contribute to the definition of requirements to ensure high reusability of the service. After the feasibility study has yielded a positive outcome and a business case has been developed, identifying a business sponsor who is willing to fund the development and operation of the service [36] is an essential activity before a project can be started. Besides that, portfolio management is also responsible for the development of an overarching service taxonomy and service

descriptions as well as for monitoring across projects. The following items represent a (certainly not exhaustive) list of Cobit activities that need to be adapted to a SOA environment:

- *Create an SOA roadmap:* The implementation of an SOA requires a significant change of both the IT landscape and the mindset of business and IT people within an organisation [37]. Due to time and budget constraints, in most cases a gradual transition towards SOA is more likely to be successful than a "big-bang" implementation. Guidance on where to start and which subsequent steps to follow is therefore essential and should be provided by creating a suitable SOA roadmap. This roadmap suggests a certain sequence in which proposed services should be analysed and developed. Identification and prioritisation of services are therefore key elements of creating a roadmap. Service identification can be conducted in top-down approaches, such as capability analysis or domain decomposition [38], or more bottom-up as in tracing business processes [36]. Prioritisation should be based on estimated business value, reuse potential (e.g. by implementing business process patterns) and IT complexity reduction potential [36]. In many real world organisations, however, services are created out of "immediate needs" (cf. section 5) either due to a lack of coordination between business and IT or simply out of aiming at short term returns. This is not surprising, as budget constraints or other obstacles may prevent a detailed analysis at this stage. In these cases, an evolutionary approach [36] can be helpful, meaning that smaller IT projects with positive business cases are defined that comply with a target application landscape as well. This will balance both short-term financial results and long-term efficiency of the SOA. We believe that the quality of the roadmap will be a crucial determining factor for the effectiveness of the whole SOA investment. The formulation of a SOA roadmap should therefore be seen as a core activity within the governance approach.
- *Assure the consultation of potential users of services:* As suggested by Cobit, all stakeholders should be included in the process (e.g. for determining requirements or assessing risks). In an SOA environment, the consultation of stakeholders becomes a common, yet more complex task, as aiming for reusability of services on a broad basis is seen as one of the core characteristics of an SOA [11, 37]. Nevertheless, many SOA initiatives fail to leverage reuse and therefore do not yield the expected financial results, leading to a drop of management support [11]. While consulting potential users is crucial to the realisation of the expected benefits, it requires a solid ground of knowing who the potential users are, putting even more emphasis on the service identification step.
- *Find business sponsor / service owner:* Another important step refers to the issue of funding [39]. Adapting services to the requirements of different users will be more expensive than developing them for the sole purpose of a single user [40]. In many cases, the benefits might outweigh the cost so that a mechanism is required for identifying those services that are worth adapting. This mechanism, however, cannot make a perfect distinction, as there is uncertainty involved in the estimation of development and maintenance cost and possible revenues. Considering this, an enterprise architect (see section 5) can identify potential users, help them express their needs and recommend a certain design of a service, but should not appoint a

business sponsor or owner. The latter should be found in a less hierarchical manner, because to enable performance measurement and encourage a high quality of decision making, the holder of the decision right should bear the economic risk as well. As multiple ownership would cause an increase in coordination effort, it will be helpful if services are owned by one of the potential users. The enterprise architect can encourage this by promoting a business case for the adapted service. If none of the potential users is willing to sponsor the service, the enterprise architect or another centralised committee could ultimately own the service as well and should therefore be provided with a dedicated budget.

4.2.2 Service Project Management

Most steps of the basic service lifecycle, as mentioned above, are part of service project management. These include analysis, design, implementation and deployment/ publishing. The analysis phase is fragmented, as this task is to a large extent conducted in the portfolio management phase, before a service sponsor can be found. In this paper, we focus on particularly interesting differences compared to traditional software development. We located them in the following activities that belong to the service lifecycle phases "Service Design" and "Service Publishing":

- *Decide on granularity and orchestration:* Although an initial analysis is conducted within the portfolio phase, different options remain for the realisation of the required functionality after a service project has been started. Sub-services that are available from the internal repository or could be bought from a service broker can serve as building blocks and reduce development cost. On the other hand, a finer granularity of service components than proposed by the requirements of internal users might also help promote services and sell them to external customers. Consequently, an optimal level of granularity is no longer just subject to technical requirements but also to market supply and demand. Thus, the availability of and the demand for services both externally and internally determines how fine or coarse a service should be and how atomic services can be combined into molecular services. This is referred to as the "economic level of granularity" [40].
- *Determine access rights:* Before a service is published, access rights need to be specified. This does not just refer to users within the organisation, but, in contrast to traditional software development, also to potential external customers. This is a strategic decision, for if cutting-edge knowledge is made accessible to competitors, comparative advantages might be lost. Therefore, key executives should be responsible for this decision.
- *Develop pricing model:* Among traditional IT cost accounting methods (for an overview see [41]), activity-based costing is seen as one of the most effective representatives [42]. Under the SOA reuse paradigm, where services are shared among several business units or departments, new mechanisms like negotiation [41] between service owners and consumers should be considered. In addition, a pricing model for the external market has to be developed if the service is also offered to external customers. It differs from the internal pricing model as it does not aim at discouraging over- or underutilisation, but aims at maximising profit.

As in the context of Service Portfolio Management, this list should not be considered complete, as there are other important aspects, such as issues regarding service contracts, Service Level Agreements (SLAs), business object governance etc. that require specific approaches.

4.2.3 Service Operation Management

Within operation management, the actual service operation, which involves activities such as training, monitoring of Service Level Agreements (SLAs) and change management, as well as the retirement phase are governed. Incident and capacity management have not been included in the service operation phase as they are not service-specific. Retirement is a responsibility of the portfolio manager; however, it strongly affects the service owner as well. It could therefore be included in the portfolio management phase as well as in the operation management phase. One main difference compared to traditional software development refers to change management. We describe the following activity in detail:

- *Develop and implement a process to consistently record, assess and prioritise change requests:* The change management process in an SOA is complex due to the distance between service providers and service consumers [39] and the high coordination effort that is required as every change affects not just the one who requested the change but the other users as well. Risk assessment should also consider side effects, because if the responses of a service are modified, other services that invoke the changed service may require changes as well [39]. Once the decision has been made and changes are authorised, all customers must be informed about the details and how their service usage requires adaptation.

5 Roles

Most of the roles proposed by Cobit (e.g. Board, CEO, CIO, CFO) are on a top management level. Additionally, architects, developers and operation managers are mentioned, but many roles that become relevant within an SOA environment are not included. Academic articles as well as industry white papers about SOA roles and responsibilities are sparse ([43] and [44] provide comprehensive frameworks, which, however, lack validation). SOA literature with a management or lifecycle focus mentions some additional roles, but mostly in an unstructured or anecdotal way [39]. Due to the lack of widely accepted terminology, definitions and descriptions, comparing or even consolidating different terms is not easy.

In this section, we give a brief overview of roles that are either not mentioned in Cobit or whose focus changes significantly under a SOA paradigm. We conducted a literature review and a comprehensive content analysis of more than 300 published job profiles at Seek.com (keyword: “SOA”), Australia's number one job site with over 100,000 jobs online. We believe that Seek.com provides a comprehensive source for an overview of the roles to be found in contemporary SOA projects in Australia. However, we did not crosscheck the findings by approaching major IT companies directly, which might have further refined the results. Here, we focus on defining the most important and accepted roles and show corresponding references.

- *Business Analyst* (Seek.com, [44, 45, 46]): The business analyst elicits domain knowledge. The business analyst understands the language of business users and providers and can translate the functional and non-functional requirements into processes and services. Among the business analyst's main responsibilities are the identification and analysis of services, but s/he is also consulted for the development of test cases.
- *Enterprise Architect* (Seek.com, [20, 36, 46, 47]): Within a traditional IT context, the enterprise architect focuses on the application of technology to increase operational effectiveness and efficiency, e.g. based on the identification of patterns in business processes [12]. In the more modern, holistic view, the enterprise architect integrates the business plan with the technical capabilities, e.g. by establishing technical policies. Within an SOA context, the enterprise architect is responsible for the development of an SOA framework and strategy. S/he ensures an optimal use as well as the performance level of services. The uptake of dedicated service architects is not visible yet.
- *Service Owner* [20, 36]: Although the service owner is mentioned as a key role, there is no definition of corresponding responsibilities and tasks in any of the literature or the published job profiles we reviewed. We define the service owner as the one who sponsors the development and operation of the service, in other terms, the benefits owner. This might be the business unit that launched the request or a centralised committee if none of the potential users is willing to fund the service or the organisation is structured hierarchically and business units or departments do not hold decision rights for the investment. As the one bearing the financial risk of the service project, the service owner must hold the right to determine a pricing model and "sell" it to other users as well as to make decisions about changes.
- *Service Librarian* [43, 48]: The service librarian is a new role in SOAs. The service librarian is responsible for the service repository and ensures the quality of published (meta-)data about as well as ease of discovery of and access to registered services.
- *Project Manager* [1, 43, 44, 46]: Compared to its traditional counterpart, an SOA project manager needs to plan for much shorter delivery cycles. This role is responsible for defining project plans, implementing the plans and monitoring the project as well as establishing the appropriate service-level agreements and resource usage. With an increased use of aggregated services (composed of other services), the relevance of this role will most likely rise.

6 Assignment of Responsibilities

The assignment of responsibilities calls for a detailed mapping of the involvement of the different roles in the activities of SOA governance. We use so-called RACI charts for each of the management layers in our proposed initial SOA governance framework to show the recommended responsibilities. The RACI charts map activities of the SOA lifecycle to roles of stakeholders in a SOA initiative and propose their responsibilities by specifying which roles are (r)esponsible, (a)ccountable, (c)onsulted or (i)nformed regarding specific activities. Roles are represented as columns and service lifecycle activities as rows. By providing these RACI charts, our framework offers a tangible and easy-to-apply tool for the analysis of responsibilities along the whole service lifecycle.

While a detailed discussion of the RACI charts is beyond the scope of this paper, two aspects of the assignment of responsibilities became particularly prominent. The first aspect is the involvement of top management and business executives in SOA development, the second aspect is the alignment of ownership for individual services. The involvement of business executives documents the degree to which the design of a service-oriented architecture is backed and driven by business concerns. In many organisations, SOA is seen as "yet another way" of software development. Consequently, few responsibilities have been changed since it was introduced. The business potential of this new paradigm is often not realised and SOA remains a means of integration for an organisation's software architecture. If this is to be changed, business representatives, especially business executives, have to be involved in decision making even more than proposed by Cobit for a traditional IT environment [1]. At first sight, this seems to increase the complexity of decision making, which would contradict executives' striving for reduction of information. Yet, management is not required to look at technical details but to understand the business implications. They can provide support for the development of interdepartmental services to leverage the reuse potential of SOA and promote the utilisation of services by selling them to external customers. Within the proposed framework, it is recommended that executives be involved in the development of an SOA roadmap and the prioritisation of services by evaluating the business potential and business value. Moreover, they can help find a business sponsor and should receive accountability for determining access rights. The business executives are expected to evaluate if a service contributes to the competitive advantage of the organisation, which could be lost once the service is offered to competitors.

Turning to ownership, the framework proposes to designate either individual service users or a central committee as service owner. A single owner that bears all cost but also appropriates all benefits of a service has several advantages. Single service ownership facilitates performance management for services and encourages owners to look for business opportunities of their internal processes, turning them into marketable services to expand their business case.

In section 7, we discuss changes to the assignment of responsibilities as compared to what traditional IT governance proposes and present the results of an application of our framework in a practical case.

7 Initial Evaluation of Utility and Lessons Learned

As a first step to validate its utility and applicability and to inform its further development, we discussed our framework with enterprise architects and other employees involved in the SOA initiative at Landgate (www.landgate.wa.gov.au). Landgate started to engage in an SOA initiative in 2006. Since then, the organisation has made some crucial organisational changes to realise SOA benefits, most remarkably the introduction of an Enterprise Architecture Office (EAO). The EAO is responsible for an impact review of proposed service projects as well as for recommending changes that leverage business investments. Thus, compared to other organisations with limited experience in SOA, Landgate's approach can be seen as quite advanced in terms of governance. In the context of this work, Landgate was very much interested in contributing to the development of the SOA governance framework and willing to learn

from that process as well. In the following, we show that the practical application of the framework not only led to an evolution of the framework itself but also helped Landgate to identify alternative approaches to assigning responsibilities within their SOA initiative and clarify the decision rights for key stakeholders.

When we discussed our recommendations and their own approach with Landgate, we perceived the limited support for interdepartmental projects due to the lack of authority and involvement of business executives in the SOA as an issue. Landgate is currently limited to a bottom-up approach to service development due to limited resources, and service development is initially based on "what services the organisation needs". Their approach is primarily project-focused, meaning that priority is given to building services as they are required by larger scale projects, which are product development projects in most cases. An impact analysis based on how the services might be shared is only an afterthought and does not affect portfolio decisions. Once a project has been started, the EAO can make suggestions on how the services should be developed. Now, in a planned re-engineering of core systems, a portfolio and program management approach is anticipated. This discussion with Landgate not only motivated the introduction of the different SOA governance layers (see section 4.1) in our framework but also strengthened the emphasis on the proposed top-executive involvement in developing a SOA roadmap and the recommendation of a more top-down oriented approach to leverage the potential of SOA for reuse and flexibility and thus long-term benefits [29].

A pain point perceived by Landgate was their unsatisfactory solution to the concept of service ownership, which differed significantly from the notion we propose in our framework. First, Landgate determines ownership at the project level. This effectively allows for multiple ownership at the project level as Landgate's scope of projects typically comprises several software services that are jointly offered as one product to their customers. Second, the ownership concept is more seen as a technical or operational role at Landgate. As a consequence, project owners usually do not seek to capture the business benefits of development projects. The direct attribution of benefits is further reduced if the EAO requires a proposed service to be changed in order to increase reusability. The EAO can then sponsor the additional cost. This, however, impedes performance measurement problems, since determining the additional cost as well as the reduction of development cost in reuse scenarios can only be calculated based on budgeted cost, as the actual cost is only known in the scenario that has been chosen. Measuring performance based on budgeted cost is not objective and can be subject to manipulation. Thus, we learned from the observed circumstances that a single service sponsor or benefits owner is required before a project is started. If no sponsor can be found, we recommend that the EAO (in Landgate's case) or another centralised committee should have the budget and authorisation to sponsor and "own" the service. If they can sell the service to the internal users and potentially external customers as well, their performance can be measured based on actual and therefore objective financial data.

8 Summary and Outlook

This paper has presented selected parts of a new framework for SOA governance. We focused on what changes to traditional IT governance approaches are required in order

to utilise the business potential of service-orientation. Initial validation at a Western Australian government agency showed that the framework can assist organisations in evaluating their own governance structure and in identifying the main obstacles to financial returns on their SOA investments. By comparing their own organisational governance model to the roles, activities and their alignment as proposed by our framework, organisations can identify divergences, which might point to weaknesses in their own approach. Once obstacles have been identified, however, major changes within the organisational structure as well as a change in mindset are often required. Therefore, it has to be borne in mind that opposition from within the organisation is likely to arise and that the implementation of required changes might take a considerable amount of time, potentially necessitating the involvement of external consultants with experience in the fields of SOA governance and change management. The proposed framework should be seen as a starting point for the research community and, at this stage, stays below the level of elaboration of its archetypes Cobit and ITIL. Its current limitations include the preliminary empirical evidence in Australia only at this stage, the emphasis on organisational aspects of SOA governance at the expense of other governance aspects such as policies, processes and metrics, and its yet untested economic efficiency. To arrive at a fully-fledged reference model for SOA governance, further work is required to evaluate the framework in real world organisations and to inform its refinement. This will be part of our future work. Moreover, we have started to derive and define a SOA governance meta-model, which will provide a formalised foundation for the framework presented in this paper and will facilitate comparison with related frameworks, the implementation of tool support and the customisation of the framework for organisation-specific needs. In addition to that, we see research opportunities in broadening the scope by integrating the different players of a service ecosystem, such as service brokers, service consumers and service providers, into the model and examine who will have the market power to set standards and force other players to comply with them in an ecosystem environment.

References

1. IT Governance Institute (ITGI). Cobit 4.1. Isaca, Rolling Meadows (2007)
2. Ridley, G., Young, J., Carroll, P.: COBIT and its Utilization: A Framework from the Literature. In: Proc. of the 37th Hawaii Int. Conf. on System Sciences (HICSS 2004), pp. 1–8. IEEE, Los Alamitos (2004)
3. Weill, P., Ross, J.W.: IT Governance: How Top Performers Manage IT Decision Rights for Superior Results. Harvard Business School Press, Boston (2004)
4. Bernhardt, J., Seese, D.: A Conceptual Framework for the Governance of Service-Oriented Architectures. In: Proc. of the 3rd Workshop on Trends in Enterprise Architecture Research (TEAR 2008), Sydney, Australia, December 1 (2008)
5. Erl, T.: SOA Principles of Service Design. Prentice Hall, Upper Saddle River (2008)
6. Barnes, M., Sholler, D., Malinverno, P.: Benefits and Challenges of SOA in Business Terms. Gartner Research, 1–6 (2005)
7. Barnes, M., Malinverno, P.: Learn the Key Success Factors for SOA Deployments. Gartner Research 8 (2005)
8. Barros, A., Dumas, M.: The Rise of Web Service Ecosystems. IT Professional 8(5), 31–37 (2006)

9. Schulz-Hofen, J.: Web Service Middleware - An Infrastructure For Near Future Real Life Web Service Eco-systems. In: Proc. of the IEEE Int. Conf. on Service-Oriented Computing and Applications (SOCA 2007), Newport Beach, California, pp. 261–270. IEEE, Los Alamitos (2007)
10. Malinverno, P.: Service-Oriented Architecture Craves Governance. Gartner Research, 1–6 (January 20, 2006)
11. Malinverno, P.: Gartner Research Index on SOA Governance. Gartner Research, 1–5 (August 16, 2006a)
12. Webmethods. SOA Governance - Enabling Sustainable Success with SOA (October 2006), http://www1.webmethods.com/PDF/whitepapers/SOA_Governance.pdf
13. Holley, K., Palistrant, J., Graham, S.: Effective SOA Governance (2006), ftp://ftp.software.ibm.com/software/rational/web/whitepapers/soagov-mgmt.pdf
14. Schelp, J., Stutz, M.: SOA-Governance. HMD-Praxis der Wirtschaftsinformatik 253, 66–73 (2007)
15. Bell, M.: Service-Oriented Modeling: Service Analysis, Design, and Architecture. Wiley, Hoboken (2008)
16. Shah, R., Bieberstein, N., Bose, S., Fiammante, M., Jones, K.: SOA Project Planning Aspects (2006), http://websphere.sys-con.com/read/168398.htm
17. Dodani, M.H.: Who Took the Cookie from the Cookie Jar? Journal of Object Technology 5(4) (May-June 2006), http://www.jot.fm/issues/issue_2006_05/column3/
18. ITIL. IT Infrastructure Library v3. OGC, TSO (2007), http://www.itil-officialsite.com/home/home.asp
19. OECD. OECD Principles of Corporate Governance. Org. for Economic Co-Operation and Development (2004), http://www.oecd.org/dataoecd/32/18/31557724.pdf
20. Malinverno, P.: Sample Governance Mechanisms for a Service-Oriented Architecture. In: Gartner Research, April 27, pp. 1–5 (2006b)
21. Software AG. Best Practices for SOA Governance – User Survey. SOA Governance Survey (2008), http://www.infoq.com/zones/centrasite/res/download/SOAGovernanceBestPracticesSurvey.pdf
22. Biske, T.: SOA Governance. Packt Publishing (2008)
23. Marks, E.A.: Service-Oriented Architecture (SOA) Governance for the Services Driven Enterprise. Wiley, Chichester (2008)
24. Brown, W.A., Laird, R.G., Gee, C., Mitra, T.: SOA Governance - Achieving and Sustaining Business and IT Agility. IBM Press (2008)
25. Kreger, H., Estefan, J.: Navigating the SOA Open Standards Landscape Around Architecture, White Paper W096, The Open Group (July 2009), http://www.opengroup.org/pubs/catalog/w096.htm
26. OMG. OMG SOA Governance Metamodel and Profile (SGMP) Standard Wiki (2009), http://www.omgwiki.org/soagov/doku.php
27. OASIS. OASIS Reference Architecture for Service-Oriented Architecture, Version 1.0, OASIS Public Review Draft 1 (April 23, 2008), http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-pr-01.pdf
28. The Open Group, The Open Group SOA Governance Framework, Draft Technical Standard (2009), http://www.opengroup.org/projects/soa-governance

29. MacKenzie, C.M., Laskey, K., McCabe, F., Brown, P.F., Metz, R.: OASIS Reference Model for Service Oriented Architecture 1.0 (October 2006), http://docs.oasis-open.org/soa-rm/v1.0/
30. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design Science in Information Systems Research. MIS Quartely 28(1), 75–105 (2004)
31. Morris, P.W.G., Jamieson, A.: Moving from Corporate Strategy to Project Strategy. Project Management Journal 36(4), 5–18 (2005)
32. Martinelli, R., Waddell, J.: Program Management: It's About the Business. PM World Today 4(1), 1–6 (2007)
33. Archer, N.P., Ghasemzadeh, F.: Project Portfolio Selection and Management. In: Morris, P.W.G., Pinto, J.K. (eds.) The Wiley Guide to Management Projects. John Wiley & Sons, NY (2004)
34. Cooke-Davies, T.: Project Success. In: Morris, P.W.G., Pinto, J.K. (eds.) The Wiley Guide to Management Projects. John Wiley & Sons, New York (2004)
35. Riedl, C., Boehmann, T., Rosemann, M., Krcmar, H.: Quality Management in Service Ecosystems. Information Systems & E-Business Management, 1–23 (2008)
36. Deb, M., Helbig, J., Kroll, M., Scherdin, A.: Bringing SOA to Life: The Art and Science of Service Discovery and Design. SOA Web Services Journal 27, 42–47 (2005)
37. Zhao, L.J., Goul, M., Purao, S., Vitharana, P., Wang, H.J.: Impact of Service-Centric Computing on Business and Education. Comm. of the Association for Information Systems 22, 295–310 (2008)
38. Kohlborn, T.: A Consolidated Approach for Service Analysis. Master's Thesis, BPM Group, Faculty of IT, Queensland University of Technology, Brisbane (2008)
39. Schepers, T.G.J., Iacob, M.E., Van Eck, P.A.T.: A Lifecycle Approach to SOA Governance. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 1055–1061. Springer, Heidelberg (2009)
40. Ren, M., Lyytinen, K.: Building Enterprise Architecture Agility and Sustenance with SOA. Comm. of the Association for Information Systems 22, 75–86 (2008)
41. Gerlach, J., Neumann, B., Moldauer, E., Aro, M., Frisby, D.: Determining the Cost of IT Services. Comm. of the ACM 45(9), 61–67 (2002)
42. Ross, J.W., Vitale, M.R., Beath, C.M.: The Untapped Potential of IT Chargeback. MIS Quarterly 23(2), 215–237 (1999)
43. Kajko-Mattsson, M., Lewis, G.A., Smith, D.B.: A Framework for Roles for Development, Evolution and Maintenance of SOA-Based Systems. In: Proc. of the Int. Workshop on Systems Development in SOA Environments (SDSOA 2007), pp. 7–12 (2007)
44. Bieberstein, N.: Service-Oriented Architecture Compass: Business Value, Planning, and Enterprise Road-map. Prentice Hall, Englewood Cliffs (2006)
45. Papazoglou, M.P.: Web Services: Principles and Technology. Prentice Hall, Harlow (2008)
46. Matsumura, M.: Delivering Business Agility with SOA: View from the Corporate Trenches. In: The Online Conference Series on Keys to SOA, pp. 21–30 (2008)
47. Strano, C., Rehmani, Q.: The Role of the Enterprise Architect. Information Systems and E-Business Management 5, 379–396 (2007)
48. Haines, M.N.: The Impact of Service-Oriented Application Development on Software Development Methodology. In: Proc. of the 40th Hawaii Int. Conf. on System Sciences (HICSS 2007), pp. 172–180. IEEE, Los Alamitos (2007)

XES, XESame, and ProM 6

H.M.W. Verbeek, Joos C.A.M. Buijs,
Boudewijn F. van Dongen, and Wil M.P. van der Aalst

Technische Universiteit Eindhoven
Department of Mathematics and Computer Science
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
{h.m.w.verbeek,j.c.a.m.buijs,b.f.v.dongen,w.m.p.v.d.aalst}@tue.nl

Abstract. Process mining has emerged as a new way to analyze business processes based on event logs. These events logs need to be extracted from operational systems and can subsequently be used to discover or check the conformance of processes. ProM is a widely used tool for process mining. In earlier versions of ProM, MXML was used as an input format. In future releases of ProM, a new logging format will be used: the *eXtensible Event Stream* (XES) format. This format has several advantages over MXML. The paper presents two tools that use this format - *XESame* and *ProM 6* - and highlights the main innovations and the role of XES. *XESame* enables domain experts to specify how the event log should be extracted from existing systems and converted to XES. *ProM 6* is a completely new process mining framework based on XES and enabling innovative process mining functionality.

1 Introduction

Unlike classical process analysis tools which are purely model-based (like simulation models), process mining requires event logs. Fortunately, today's systems provide detailed event logs. Process mining has emerged as a way to analyze systems (and their actual use) based on the event logs they produce [1,2,3,4,6,16]. Note that, unlike classical data mining, the focus of process mining is on concurrent processes and not on static or mainly sequential structures. Also note that commercial Business Intelligence (BI for short) tools are not doing any process mining. They typically look at aggregate data seen from an external perspective (including frequencies, averages, utilization and service levels). Unlike BI tools, process mining looks "inside the process" and allows for insights at a much more refined level.

The omnipresence of event logs is an important enabler of process mining, as analysis of run-time behavior is only possible if events are recorded. Fortunately, all kinds of information systems provide such logs, which include classical workflow management systems like FileNet and Staffware, ERP systems like SAP, case handling systems like BPM|one, PDM systems like Windchill, CRM systems like Microsoft Dynamics CRM, and hospital information systems like Chipsoft. These systems provide very detailed information about the activities that have been executed.

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 60–75, 2010.
© Springer-Verlag Berlin Heidelberg 2010

However, also all kinds of embedded systems increasingly log events. An embedded system is a special-purpose system in which the computer is completely encapsulated by or dedicated to the device or system it controls. Examples include medical systems like X-ray machines, mobile phones, car entertainment systems, production systems like wafer steppers, copiers, and sensor networks. Software plays an increasingly important role in such systems and, already today, many of these systems log events. An example is the "CUSTOMerCARE Remote Services Network" of Philips Medical Systems (PMS for short), which is a worldwide internet-based private network that links PMS equipment to remote service centers. Any event that occurs within an X-ray machine (like moving the table or setting the deflector) is recorded and can be analyzed remotely by PMS. The logging capabilities of the machines of PMS illustrate the way in which embedded systems produce event logs.

The MXML format [7] has proven its use as a standard event log format in process mining. However, based on practical experiences with applying MXML in about one hundred organizations, several problems and limitations related to the MXML format have been discovered. One of the main problems is the semantics of additional attributes stored in the event log. In MXML, these are all treated as string values with a key and have no generally understood meaning. Another problem is the nomenclature used for different concepts. This is caused by MXML's assumption that strictly structured process would be stored in this format [10].

To solve the problems encountered with MXML and to create a standard that could also be used to store event logs from many different information systems directly, a new event log format is under development. This new event log format is named XES, which stands for eXtensible Event Stream. Please note that this paper is based on XES definition version 1.0, revision 3, last updated on November 28, 2009. This version serves as input for standardization efforts by the IEEE Task Force Process Mining [13]. Minor changes might be made before the final release and publication of the format.

The remainder of this paper is organized as follows. Section 2 introduces the new event log format XES. Of course, we need to be able to extract XES event logs from arbitrary information systems in the field. For this reason, Section 3 introduces the XES tool XESame. This tool can connect to any ODBC database, and allows the domain expert to provide the details of the desired extraction in a straightforward way. After having obtained an XES event log, we should be able to analyze this log in all kinds of ways. For this reason, Section 4 introduces the XES tool ProM 6, which is the upcoming release of the ProM framework [8]. ProM 6 supports the XES event log format, and provides a completely new process mining framework. Finally, Section 5 concludes the paper.

2 XES: eXtensible Event Stream

To explain the structure of an XES event log, we compare the way a single process containing only a single event is captured in both the MXML and the

Listing 1. Example MXML log

```
<WorkflowLog xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation=
      "http://is.ieis.tue.nl/research/processmining/WorkflowLog.xsd"
    description="Example log">
  <Process id="Order">
    <ProcessInstance id="Order 1" description="instance with Order 1">
      <Data>
        <Attribute name="TotalValue">2142.38<Attribute>
      </Data>
      <AuditTrailEntry>
        <WorkflowModelElement>Create</WorkflowModelElement>
        <EventType>complete</EventType>
        <Originator>Wil<Originator>
        <Timestamp>2009-01-03T15:30:00.000+01:00</Timestamp>
        <Data>
          <Attribute name="currentValue">2142.38</Attribute>
          <Attribute name="requestedBy">Eric</Attribute>
          <Attribute name="supplier">Fluxi Inc.</Attribute>
          <Attribute name="expectedDelivery">
            2009-01-12T12:00:00.000+01:00
          </Attribute>
        </Data>
      </AuditTrailEntry>
    </ProcessInstance>
  </Process>
</WorkflowLog>
```

XES format. The event corresponds to the creation of an order on January 3, 2009 at 15:30 hours CET by the employee called Wil. The total value of the entire order is 2,142.38 Euros, it was requested by a customer called Eric, it is supplied by a company called Fluxi Inc., and delivery is expected on January 12, 2009 at 12:00 hours CET.

Listing 1 shows the way this log is captured in MXML. In MXML, a **ProcessInstance** element captures a single process instance, whereas an **AuditTrailEntry** element captures a single event. Data attributes can be associated to these elements using a **Data** element containing multiple **Attribute** elements. MXML uses a number of predefined MXML attributes:

WorkflowModelElement. This attribute captures the name of the activity that triggered the event.

EventType. This attribute captures the type of the event, like start, complete, suspend, and resume.

Originator. This attribute captures the name of the resource (human or not) who actually executed the activity.

Timestamp. This attribute captures the time at which the event occurred in the system.

Although the meaning of any of these standard attributes is generally well-understood, the meaning of the any of the other, non-standard, attributes is not.

In contrast, Listing 2 shows this log in XES, whereas Fig. 1 shows the XES meta model [11]. Intuitively, the XES **log** element replaces, the MXML **WorfklowLog** element, the **trace** element replaces the **ProcessInstance** element, and the **event** element replaces the **AuditTrailEntry** element. However, there are a number of differences worth mentioning. First of all, in XES the **log**, **trace** and **event** elements only define the structure of the document: they do not contain any information themselves. To store any data in the XES format, attributes are used. Every attribute has a string based key, a known type, and a value of that type. Possible types are **string**, **date**, **integer**, **float** and **boolean**. Note that attributes can have attributes themselves which can be used to provide more specific information.

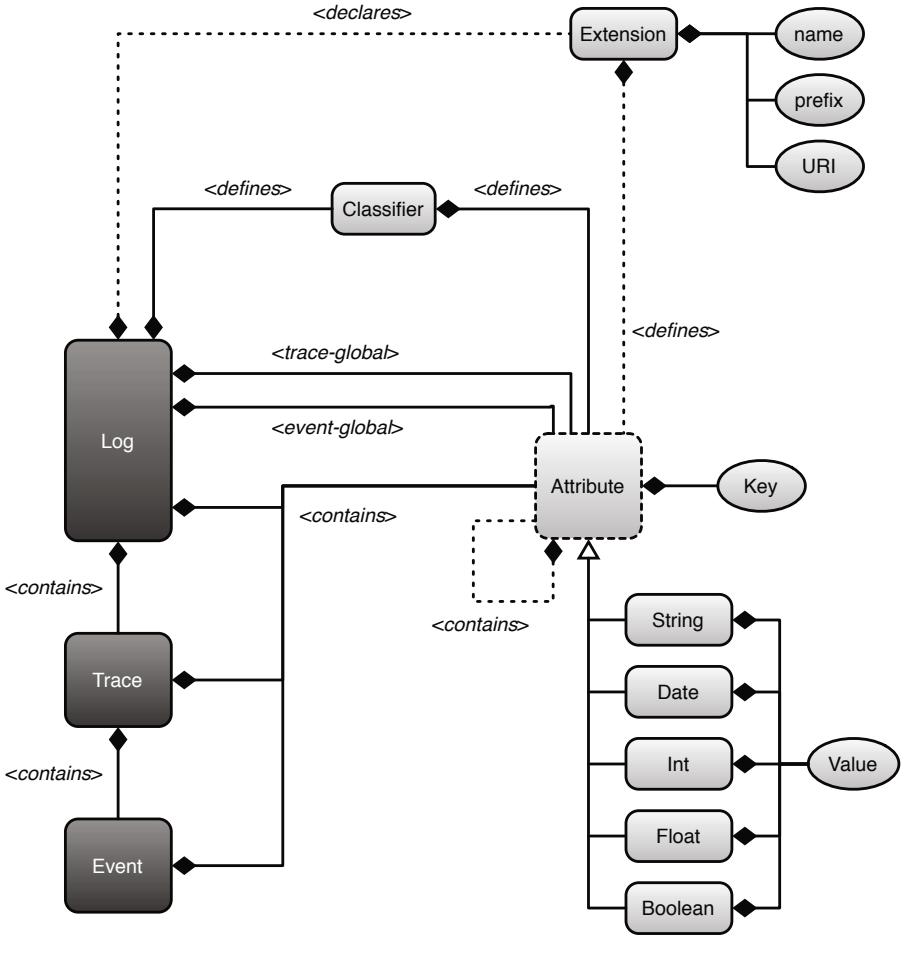

Fig. 1. XES Meta Model

Listing 2. Example XES log

```
<log>
  <extension name="Lifecycle" prefix="lifecycle"
      uri="http://www.xes-standard.org/lifecycle.xesext" />
  <extension name="Time" prefix="time"
      uri="http:///www.xes-standard.org/time.xesext" />
  <extension name="Concept" prefix="concept"
      uri="http:///www.xes-standard.org/concept.xesext" />
  <extension name="Semantic" prefix="semantic"
      uri="http:///www.xes-standard.org/semantic.xesext" />
  <extension name="Organizational" prefix="org"
      uri="http:///www.xes-standard.org/org.xesext" />
  <extension name="Order" prefix="order"
      uri="http://my.company.com/xes/order.xesext" />
  <global scope="trace">
    <string key="concept:name" value="unknown" />
  </global>
  <global scope="event">
    <string key="concept:name" value="unknown" />
    <string key="lifecycle:transition" value="unknown" />
    <string key="org:resource" value="unknown" />
  </global>
  <classifier name="Activity classifier" keys="concept:name lifecycle:transition" />
  <string key="concept:name" value="Example log" />
  <trace>
    <string key="concept:name" value="Order 1" />
    <float key="order:totalValue" value="2142.38" />
    <event>
      <string key="concept:name" value="Create" />
      <string key="lifecycle:transition" value="complete" />
      <string key="org:resource" value="Wil" />
      <date key="time:timestamp" value="2009-01-03T15:30:00.000+01:00" />
      <float key="order:currentValue" value="2142.38" />
      <string key="details" value="Order creation details">
        <string key="requestedBy" value="Eric" />
        <string key="supplier" value="Fluxi Inc." />
        <date key="expectedDelivery" value="2009-01-12T12:00:00.000+01:00" />
      </string>
    </event>
  </trace>
</log>
```

Table 1. List of XES extensions and their attribute keys

Extension	Key	Level	Type	Description
concept	name	log, trace, event	string	Generally understood name.
	instance	event	string	Identifier of the activity whose execution generated the event.
lifecycle	model	log	string	The transactional model used for the lifecycle transition for all events in the log.
	transition	event	string	The lifecycle transition represented by each event (e.g. start, complete, etc.).
organizational	resource	event	string	The name, or identifier, of the resource having triggered the event.
	role	event	string	The role of the resource having triggered the event, within the organizational structure.
	group	event	string	The group within the organizational structure, of which the resource having triggered the event is a member.
time	timestamp	event	date	The date and time, at which the event has occurred.
semantic	modelReference	all	string	Reference to model concepts in an ontology.

The precise semantics of an attribute is defined by its extension, which could be either a standard extension or some user-defined extension. Standard extensions include the *concept* extension, the *lifecycle* extension, the *organizational* extension, the *time* extension, and the *semantic* extension. Table 1 shows an overview of these extensions together with a list of possible keys, the level on which these keys may occur, the value type, and a short description, whereas Listing 3 shows the definition of the concept extension. Note that the semantic extension is inspired by SA-MXML (Semantically Annotated MXML) [15]. Note that in the example of Listing 2 some attributes are defined by an order extension (totalValue and currentValue), where other attributes are not defined by any extension (including details and supplier).

Furthermore, *event classifiers* can be specified in the **log** element which assign an identity to each event. This makes events comparable to other events via their assigned identity. Classifiers are defined via a set of attributes, from which the class identity of an event is derived. A straightforward example of a classifier is the combination of the event name and the lifecycle transition as used in MXML, which is included in Listing 2.

Finally, the fact that certain attributes have well-defined values for every trace and/or event in the log can be specified by the **global** element. For example, in the example shown in Listing 2 the event attributes concept name and lifecycle

Listing 3. Concept extension

```
<xesextension name="Concept" prefix="concept"
    uri="http:///www.xes−standard.org/concept.xesext">
  <log>
    <string key="name">
      <alias mapping="EN" name="Name"/>
      <alias mapping="DE" name="Name"/>
      <alias mapping="FR" name="Appellation"/>
      <alias mapping="ES" name="Nombre"/>
      <alias mapping="PT" name="Nome"/>
    </string>
  </log>
  <trace>
    <string key="name">
      <alias mapping="EN" name="Name"/>
      <alias mapping="DE" name="Name"/>
      <alias mapping="FR" name="Appellation"/>
      <alias mapping="ES" name="Nombre"/>
      <alias mapping="PT" name="Nome"/>
    </string>
  </trace>
  <event>
    <string key="name">
      <alias mapping="EN" name="Name"/>
      <alias mapping="DE" name="Name"/>
      <alias mapping="FR" name="Appellation"/>
      <alias mapping="ES" name="Nombre"/>
      <alias mapping="PT" name="Nome"/>
    </string>
    <string key="instance">
      <alias mapping="EN" name="Instance"/>
      <alias mapping="DE" name="Instanz"/>
      <alias mapping="FR" name="Entit"/>
      <alias mapping="ES" name="Instancia"/>
      <alias mapping="PT" name="Instncia"/>
    </string>
  </event>
</xesextension>
```

transition have well-defined values for every event, which of course is very nice (though not mandatory) as these attributes are used in an event classifier. In case a trace and/or event does not have the attribute, the value of the corresponding **global** attribute will be used. As a result of these **global** elements, plug-ins that require these attributes to have values for every trace and/or event can quickly check whether these attributes indeed have values for every trace and/or event.

3 XESame

Although many information systems record the information required for process mining, chances are that this information is not readily available in the XES format. Since the information is present in the data storage of the information system, it should be possible to reconstruct an event log that contains this information. However, extracting this information from the data storage is likely to be a time consuming task and requires domain knowledge, knowledge which is usually held by domain experts like business analysts.

For the purpose of extracting am (MXML) event log from an information system, the ProM Import Framework [9] was created. Although there is a collection of plug-ins for various systems and data structures, chances are that a new plug-in needs to be written by the domain expert in Java. The main problem with this approach is that one cannot expect the domain expert to have Java programming skills. Therefore, there is a need for a tool that can extract the event log from the information system at hand without the domain expert having to program. This tool is *XESame* [5][1].

XESame provides a generic way for extracting an event log from some data source, and is designed to be easy to use. A key strength of XESame is that no programming skills are required: The entire conversion from data source to event log can be defined through the GUI. We use a simple example to showcase XESame. Table 2 shows the contents of two tables, from which we will generate an event log. Fig. 2 shows the internal representation of the conversion in XESame. Among other things, these details show that:

- The resulting log will use the concept, lifecycle, organizational, time, and semantic extension.
- The resulting log will originate from the **events.csv** table, will have name 'Testlog', and will use the standard lifecycle model.
- The resulting log will contain the standard event classifier.
- In the resulting log, every trace corresponds to an order.
- The resulting log will contain all traces that correspond to orders with **orderID** less than 100.
- Every trace will contain all events that are related to the corresponding order.
- Every event will use the **eventname** field as concept name, the **eventtype** field as lifecycle transition, and the **timestamp** field as time timestamp.
- Every event will use the **usergroup** field from the **users.csv** table as organizational group, where both tables have been linked on the **userId** field.

To demonstrate the applicability of XESame, we have performed two case studies on data from different systems. The first case study was performed on data from an SAP system. This case study showed that a conversion definition could be defined using different tables and columns from the data source. The other case study showed that data exported from a custom system can also be converted to

[1] Note that in [5] the tool is called the XES Mapper instead of XESame.

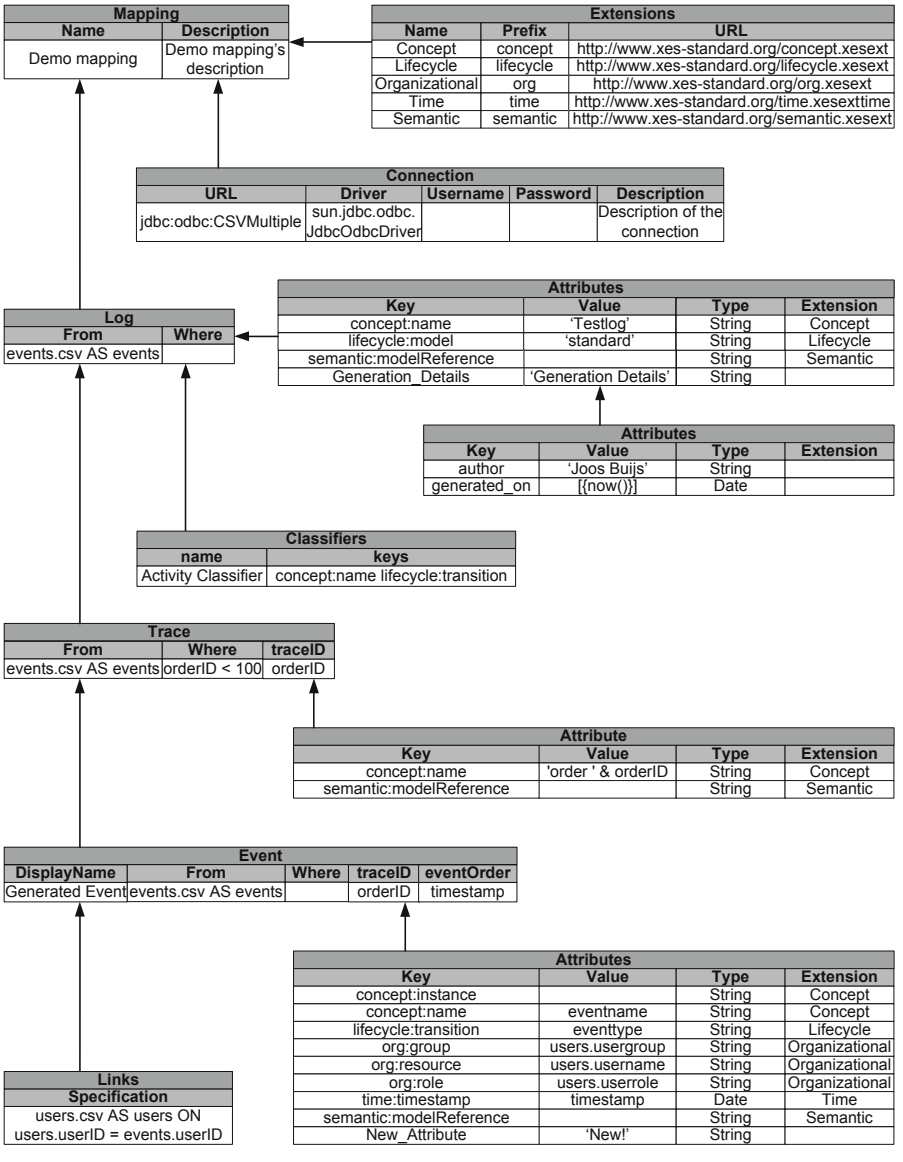

Fig. 2. An example mapping instance

Table 2. Example source data

events.csv

orderID	eventName	timestamp	eventType	userID
1	Create	1-1-2009 10:00	Start	1
1	Create	1-1-2009 11:00	Complete	1
2	Create	1-1-2009 11:00	Start	1
2	Create	1-1-2009 12:00	Complete	1
1	Send	2-1-2009 10:00	Start	2
3	Create	2-1-2009 10:01	Start	1
1	Send	2-1-2009 10:10	Complete	2
3	Create	2-1-2009 11:00	Complete	1
2	Send	2-1-2009 12:00	Start	2
2	Send	2-1-2009 13:00	Complete	2
1	Receive	3-1-2009 10:00	Start	3
1	Receive	3-1-2009 10:05	Complete	3
1	Pay	3-1-2009 15:00	Start	4
2	Pay	3-1-2009 15:00	Start	4
1	Pay	3-1-2009 15:30	Complete	4
2	Pay	3-1-2009 15:30	Complete	4
2	Receive	3-1-2009 17:00	Start	3
2	Receive	3-1-2009 17:05	Complete	3
123	Create	14-2-2009 9:00	Start	1
123	Create	14-2-2009 9:10	Complete	1

users.csv

userID	userName	userGroup	userRole
1	George	Purchase	OrderCreator
2	Ine	Support	Secretary
3	Eric	Warehouse	Reciever
4	Wil	Finance	Payer

a log by XESame. For both case studies, the performance was also investigated and shown to be linear in time with the size of the resulting event log.

4 ProM

After having extracted the event log from the information system, we can analyze the event log using ProM [8], the plugable generic open-source process mining framework. The ProM toolkit has been around for about six years. During this period, the ProM framework has matured to a professional level, which has allowed dozens of developers in different countries to contribute their research in the form of plug-ins. In the end, this resulted in ProM 5.2, which contains 286 plug-ins, and which has been used in over one hundred case studies. Some examples include:

For a provincial office of *Rijkswaterstaat* (the Dutch National Public Works Department), we have analyzed [1] its invoice process and have shown that its bad performance was mainly due to employees working at remote sites. Furthermore, we showed that it is worthwhile to combine different mining perspectives to reach a richer understanding of the process. In this case, for example, the process model revealed the problems (loops), but it took an organizational model to identify the key players, and a case-oriented analysis to understand the impact of these loops on the process performance.

For *ASML* (the leading manufacturer of wafer scanners in the world), we have investigated [17] its test process, and have made concrete suggestions for process improvement, which included reordering of tasks to prevent feedback loops and using idle time for scheduling. However, we also showed that further research is needed to develop process mining techniques that are particularly suitable for analyzing processes like the highly dynamic test process of ASML.

For the Dutch *AMC hospital*, we have shown [14] that we were able to derive understandable models for large groups of patients, which was confirmed by people of the hospital. Nevertheless, we also showed that traditional process mining approaches have problems dealing with unstructured processes as, for example, can be found in a hospital environment.

As XES is a new log format that is still under development, the older versions of ProM do not handle XES logs. Fortunately, the upcoming version of ProM, ProM 6, will be able to handle XES logs.

The fact that ProM 6 can handle XES logs where earlier versions of ProM cannot is not the only difference between ProM 6 and its predecessors (ProM 5.2 and earlier). Although these predecessors have been a huge success in the process mining field, they limited future work for a number of reasons. First and foremost, the earlier versions of ProM did not separate the functionality of a plug-in and its GUI. As a result, a plug-in like the α-miner [3] could not be run without having it popping up dialogs. As a result, it was impossible to run the plug-in on some remote machine, unless there would be somebody at the remote display to deal with these dialogs. Since we are using a dedicated process grid for process mining, this is highly relevant. Second, the distinction between the different kind of plug-ins (mining plug-ins, analysis plug-in, conversion plug-ins, import plug-ins, and export plug-ins) has disappeared; leaving only the concept of a generic plug-in. Third, the concept of an object pool has been introduced: plug-ins take a number of objects from this pool as input, and produce new objects for this pool. Fourth, ProM 6 allows the user to first select a plug-in, and then select the necessary input objects from the pool. As some plug-in can handle different configurations of objects as input, ProM 6 also introduces the concept of plug-in variants. The basic functionality of variants of some plug-in will be identical, but every variant will be able to take a different set of objects as input.

Fig. 3. ProM 6 view on the log

We use the XES event log obtained from XESame, as described in the previous section, to showcase ProM 6. Fig. 3 shows shows some basic characteristics of the log, which includes:

- the number of case (traces) (3),
- the number of events (18),
- the number of event classes (8),
- the number of event types (2),
- the number of originators (4),
- a graphical representation of the distribution of events per case,
- a graphical representation of the distribution of event classes per case, and
- some general information on the log.

Fig. 4 shows the action view of ProM 6, which includes a view on the filtered list of installed plug-ins. For every plug-in this list includes:

- the name of the plug-in (like "Simple Log Filter"),
- the name of the author of the plug-in (like "H.M.W. Verbeek"), and
- a URL where additional information on this plug-in may be found (like "http://www.processmining.org").

Fig. 4. ProM 6 action view

ProM 6 is aware of all required inputs and all output types for every plug-in. As a result, because the user has selected the event log named "Testlog.xes" to be a required input, the list of installed plug-ins show only those plug-ins that can actually take an event log as input. Plug-ins for which all required inputs are satisfied, that is, plug-ins that only need an event as input, are colored green in the plug-in list (like "Simple Log Filter"), whereas plug-ins that require additional inputs are colored yellow (like "Fitness", which also requires a Petri net as input). Note that it is also possible to filter the list of plug-ins on the output types, which allows the user to get quickly to those plug-ins that can produce an object of a type s/he needs.

Fig. 5 shows a dotted chart [18] on the log. This chart shows a graphical view on the log, where all events of a single case are plotted on a horizontal line, where the position on this line corresponds to the timestamp of the event, and where the color of the dot corresponds to the name of the event. For example, the two selected green dots in the middle correspond to two events for the case named "order 2", the first event corresponds to a send activity started by "Ine" on January 2, 2009 at 12:00 PM, whereas the second event corresponds to the matching complete event that occurred an hour later.

Fig. 6 shows the fuzzy model that result from running the "Fuzzy miner" [12] on the example log. This fuzzy model clearly shows that no strict ordering exists between the "Receive" and "Pay" activities.

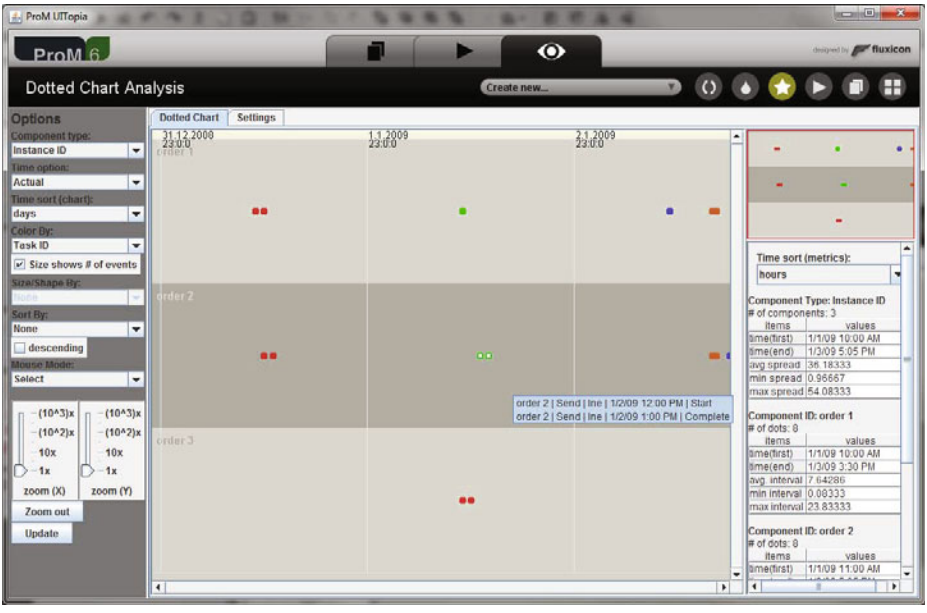

Fig. 5. ProM 6 dotted chart

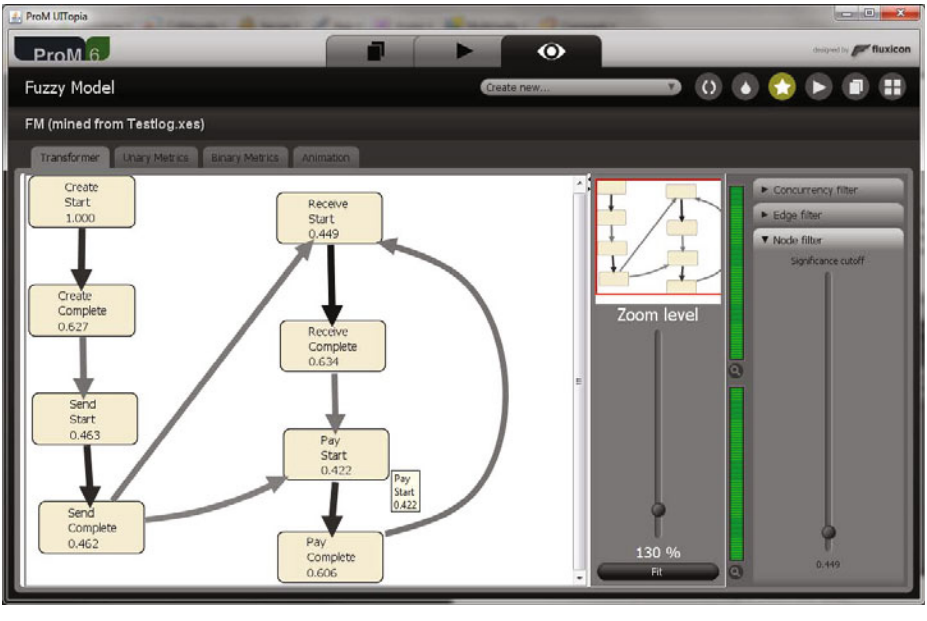

Fig. 6. ProM 6 fuzzy model

5 Conclusions

This paper has introduced the new event log format XES. The XES format enhances the existing MXML [7] in many ways, as is shown in this paper. XES is used as input for standardization efforts within the IEEE Task Force on Process Mining [13].

This paper also introduced a tool that allows the domain expert to extract an XES event log from some existing system. This tool, XESame [5], improves on the ProM Import framework [9] in the way that it is generic, and that it does not require the domain expert to create a Java plug-in for specifying (and executing) the extraction. Instead, XESame allows the domain expert to simply specify from which fields in the database which attributes in the event log should be extracted.

Finally, this paper has introduced a new version of the ProM framework [8], ProM 6. In contrast to earlier versions of ProM, ProM 6 can handle XES event logs, can be executed on remote machines, and can guide the user into selecting the appropriate inputs for a certain plug-in. As a result, it better supports the analysis of event logs than any of the earlier releases did.

ProM 6 will be released in the Summer of 2010, but interested readers may already obtain a prerelease (which also contains XESame) or so-called 'nightly builds' through the Process Mining website (`www.processmining.org`).

Acknowledgements. The authors would like to thank Christian Günther for his work on the XES standard and the new UI of ProM 6.

References

1. van der Aalst, W.M.P., Reijers, H.A., Weijters, A.J.M.M., van Dongen, B.F., Alves de Medeiros, A.K., Song, M., Verbeek, H.M.W.: Business Process Mining: An Industrial Application. Information Systems 32(5), 713–732 (2007)
2. van der Aalst, W.M.P., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.M.M.: Workflow Mining: A Survey of Issues and Approaches. Data and Knowledge Engineering 47(2), 237–267 (2003)
3. van der Aalst, W.M.P., Weijters, A.J.M.M., Maruster, L.: Workflow Mining: Discovering Process Models from Event Logs. IEEE Transactions on Knowledge and Data Engineering 16(9), 1128–1142 (2004)
4. Agrawal, R., Gunopulos, D., Leymann, F.: Mining process models from workflow logs. In: Schek, H.-J., Saltor, F., Ramos, I., Alonso, G. (eds.) EDBT 1998. LNCS, vol. 1377, pp. 469–483. Springer, Heidelberg (1998)
5. Buijs, J.C.A.M.: Mapping Data Sources to XES in a Generic Way. Master's thesis, Eindhoven University of Technology (2010)
6. Datta, A.: Automating the Discovery of As-Is Business Process Models: Probabilistic and Algorithmic Approaches. Information Systems Research 9(3), 275–301 (1998)
7. van Dongen, B.F., van der Aalst, W.M.P.: A Meta Model for Process Mining Data. In: Casto, J., Teniente, E. (eds): Proceedings of the CAiSE 2005 Workshops (EMOI-INTEROP Workshop), vol. 2, pp. 309–320. FEUP, Porto, Portugal (2005)

8. van Dongen, B.F., Alves de Medeiros, A.K., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The ProM Framework: A New Era in Process Mining Tool Support. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 444–454. Springer, Heidelberg (2005)
9. Günther, C.W., van der Aalst, W.M.P.: A Generic Import Framework for Process Event Logs. In: Eder, J., Dustdar, S. (eds.) BPI 2006. LNCS, vol. 4103, pp. 81–92. Springer, Heidelberg (2006)
10. Günther, C.W.: Process Mining in Flexible Environments. PhD thesis, Eindhoven University of Technology, Eindhoven (2009)
11. Günther, C.W.: XES Standard Definition. Fluxicon Process Laboratories (November 2009)
12. Günther, C.W., van der Aalst, W.M.P.: Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 328–343. Springer, Heidelberg (2007)
13. IEEE Task Force on Process Mining, `http://www.win.tue.nl/ieeetfpm`
14. Mans, R.S., Schonenberg, M.H., Song, M., van der Aalst, W.M.P., Bakker, P.J.M.: Application of process mining in healthcare - a case study in a Dutch hospital. In: Fred, A., Filipe, J., Gamboa, H. (eds.) Biomedical Engineering Systems and Technologies. Communications in Computer and Information Science, vol. 25, pp. 425–438. Springer, Heidelberg (2009)
15. Alves de Medeiros, A.K., Pedrinaci, C., van der Aalst, W.M.P., Domingue, J., Song, M., Rozinat, A., Norton, B., Cabral, L.: An Outlook on Semantic Business Process Mining and Monitoring. In: Meersman, R., Tari, Z., Herrero, P. (eds.) SWWS 2007. LNCS, vol. 4806, pp. 1244–1255. Springer, Heidelberg (2007)
16. Rozinat, A., van der Aalst, W.M.P.: Conformance Checking of Processes Based on Monitoring Real Behavior. Information Systems 33(1), 64–95 (2008)
17. Rozinat, A., de Jong, I.S.M., Günther, C.W., van der Aalst, W.M.P.: Process mining applied to the test process of wafer steppers in ASML. IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews (2009) (to appear)
18. Song, M., van der Aalst, W.M.P.: Supporting Process Mining by Showing Events at a Glance. In: Chari, K., Kumar, A. (eds.) Proceedings of 17th Annual Workshop on Information Technologies and Systems (WITS 2007), Montreal, Canada, pp. 139–145 (December 2007)

SeaFlows Toolset – Compliance Verification Made Easy for Process-Aware Information Systems*

Linh Thao Ly[1], David Knuplesch[1], Stefanie Rinderle-Ma[2], Kevin Göser[3], Holger Pfeifer[4], Manfred Reichert[1], and Peter Dadam[1]

[1] Institute of Databases and Information Systems
Ulm University, Germany
{thao.ly,david.knuplesch,manfred.reichert,peter.dadam}@uni-ulm.de
[2] Institute of Artificial Intelligence
Ulm University, Germany
holger.pfeifer@uni-ulm.de
[3] Faculty of Computer Science
University of Vienna, Austria
stefanie.rinderle-ma@univie.ac.at
[4] AristaFlow GmbH, Germany
kevin.goeser@aristaflow.com

Abstract. In the light of an increasing demand on business process compliance, the verification of process models against compliance rules has become essential in enterprise computing. The SeaFlows Toolset featured in this paper extends process-aware information systems with compliance checking functionality. It provides a user-friendly environment for modeling compliance rules using a graph-based formalism and for enriching process models with these rules. To address a multitude of verification settings, we provide two complementary compliance checking approaches: The *structural compliance checking* approach derives structural criteria from compliance rules and applies them to detect incompliance. The *data-aware behavioral compliance checking* approach addresses the state explosion problem that can occur when the data dimension is explored during compliance checking. It performs context-sensitive automatic abstraction to derive an abstract process model which is more compact with regard to the data dimension enabling more efficient compliance checking. Altogether, SeaFlows Toolset constitutes a comprehensive and extensible framework for compliance checking of process models.

Keywords: Compliance rules, Data-aware compliance checking, Process verification.

1 Introduction

In the light of an increasing demand for business process compliance [1], the verification of process models within process-aware information systems against

* This work was done in the research project SeaFlows which is partially funded by the German Research Foundation (DFG).

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 76–91, 2010.
© Springer-Verlag Berlin Heidelberg 2010

compliance rules has become essential in enterprise computing. To ensure compliance with imposed rules and policies, compliance audits for process models become necessary. Due to the increasing complexity of process models [2], manual compliance verification is hardly feasible. Tool support is particularly needed in order to deal with changes at different levels. On the one hand, changes of regulatories and policies occur which then necessitate leading to changes of implemented compliance rules. On the other hand, changes to business processes may take place, which result in changes of implemented process models. This further necessitates tool support for (semi-)automatic compliance verification.

In this paper, we introduce SeaFlows Toolset, a tool framework for business process compliance verification, and underlying concepts. These resulted from our research in the SeaFlows project. In this project, we aim at providing techniques to enable compliance with imposed regulatories throughout the process lifecycle. This includes compliance checking of business process models at buildtime, but also requires compliance monitoring for process instances at runtime [3]. With the implementation of SeaFlows Toolset, so far, we have realized concepts addressing compliance checking of process models at buildtime. In particular, the toolset enables to visually model compliance rules by means of so-called *compliance rule graphs*. In addition, it supports the verification of process models against imposed compliance rules. To support a variety of verification scenarios and to exploit their specific properties, we introduce two complementary verification approaches. First, we discuss a *structural compliance checking* approach based on node relations, which enables efficient compliance verification for block-structured process models. In addition, we provide a general *behavioral compliance checking* approach that realizes data-awareness as well.

Example. Throughout this paper an exemplary order-to-delivery scenario will be used to illustrate basic concepts and SeaFlows Toolset: Fig. 1 depicts a process model P for a simplified order-to-delivery process. For brevity, we abstain from modeling the complete data flows of P. The order-to-delivery process, in turn, may be subject to the compliance rules collected in Table 1. They constrain the execution and ordering of activities and events within a process model.

The remainder of this paper is structured as follows. In the following, we first introduce the concepts behind SeaFlows Toolset before presenting it in more detail. Compliance rule graphs, the visual modeling formalism, are introduced in Sect. 2. The structural as well as the behavioral compliance checking approach are discussed in Sect. 3. The particular components of SeaFlows Toolset are introduced in Sect. 4. Related work is discussed in Sect. 5 before we close the paper with an outlook on future developments in Sect. 6.

2 Compliance Rule Graphs – A Visual Modeling Formalism

Prerequisite to automatic compliance checking is that compliance rules are modeled using a suitable formalism. On the one hand, the formalism has to be sufficiently expressive. On the other hand, the its complexity should not impede

Table 1. Examples of compliance rules for order-to-delivery processes

c_1	A received order has to be either confirmed or declined.
c_2	Outsourced production shall be followed by a quality test.
c_3	After confirming an order and previous to confirming shipping, shipping has to be prepared and eventually shipped.
c_4	Premium customer status shall only be offered after a prior solvency check.
c_5	For orders of a non-premium customer with a piece number beyond `80,000`, a solvency check becomes necessary before assessing the order.
c_6	After confirming an order of a non-premium customer with piece number of at least `125,000`, premium status should be offered to the customer
c_7	Shipping orders with piece number below `80,000` does not require shipping insurance.
c_8	Orders with piece number beyond 40,000 shall only be confirmed after prior approval.

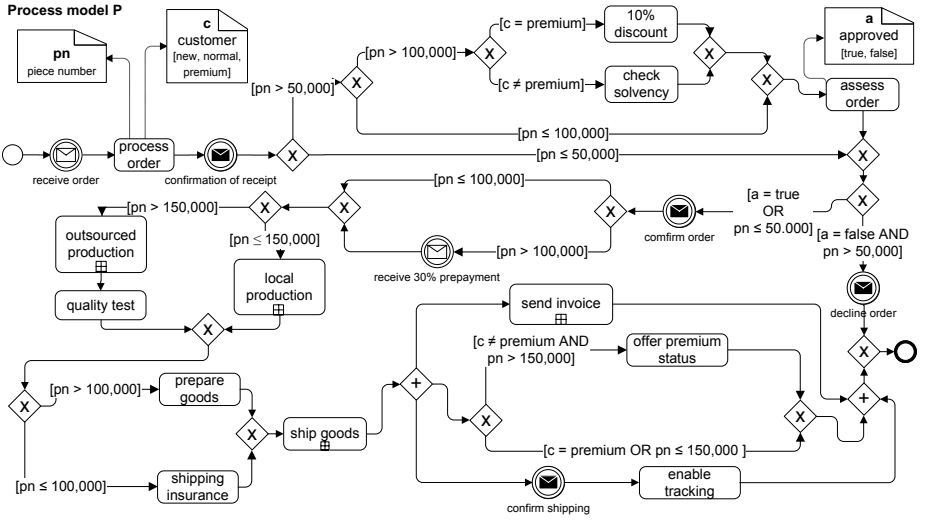

Fig. 1. An order-to-delivery process (modeled with BPMN)

its application. Apparently, logic formalisms, such as linear temporal logic, are powerful. However, their complexity can become a barrier to their application to compliance rule modeling by domain experts in practice. To address this issue, pattern-based approaches [4,5,6] have been suggested recently (cf. Sect. 5). Visual patterns are provided which represent placeholders for logic formulas. However, pattern-based approaches provide only limited support for modeling more complex compliance rules. Hence, we opted for a compositional graph-based

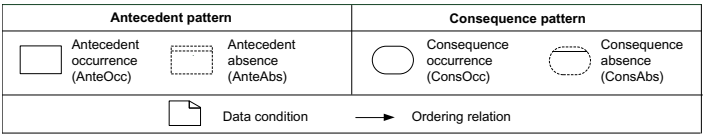

Fig. 2. "Ingredients" for modeling compliance rule graphs

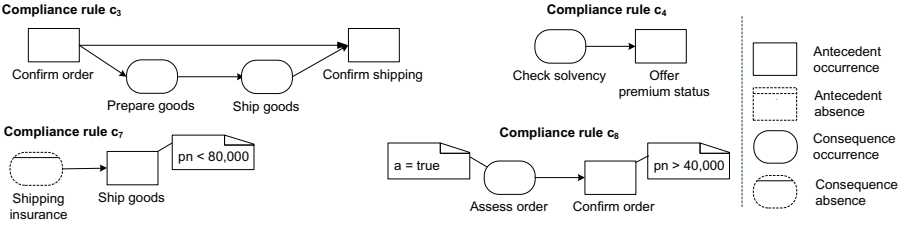

Fig. 3. Compliance rule graphs

approach that provides both a visual formalism and formal semantics. The basic idea is to enable compliance rule modeling in a way similar to process modeling, namely by means of graphs representing fragments of process executions.

In our case studies (e.g., in the clinical domain) we identified the typical structure of compliance rules. Roughly described, it states that if some event patterns occur then some other event patterns must also occur. Each event pattern, in turn, can express the occurrence or absence of particular events, or be structured in a complex manner (e.g., particular relations between events). Based on these observations, we developed our compliance rule graph (CRG) approach that enables the modeling of compliance rules by means of directed graphs. The basic "ingredients" for composing a CRG are depicted in Fig. 2. They comprise four different node types indicating occurrence and absence of activities of an associated type. AnteOcc and AnteAbs are dedicated to modeling the antecedent pattern activating a compliance rule whereas ConsOcc and ConsAbs are dedicated to modeling consequence patterns that have to occur upon activation of a compliance rule. CRG nodes can be related to each other using ordering relations. In addition, a CRG can be enriched with data conditions to be able to capture data-aware compliance rules. Fig. 3 shows how these "ingredients" are applied to model some of the compliance rules from Table 1. CRG c_3, for example, states that if the antecedent pattern consisting of the sequence `confirm order` and `confirm shipping` occurs in a process execution, the sequence consisting of `prepare goods` and `ship goods` must occur in between. CRG c_8, in turn, shows how data conditions are applied. It expresses that activity `assess order` with data condition `a = true` is required prior to confirming an order with data condition `pn` $>$ `40,000`.

CRGs go beyond a pure visual notation. We also equipped them with formal semantics that enable formal compliance verification. Due to space limitations,

we omit the definition of CRG formal semantics and refer to [7], where CRG formulas and their interpretation over execution traces are discussed.

3 Structural and Behavioral Compliance Checking

Constraining process behavior, compliance rules typically are specified at a behavioral level whereas a process model is a structure describing possible process behavior. To verify process models against such behavioral compliance rules, we basically have two options:

Behavioral Compliance Checking is conducted by verifying the process behaviour against imposed compliance rules. This can be achieved by exploring possible process behavior with regard to compliance.

Structural Compliance Checking derives *structural properties* from behavioral compliance rules. Such structural properties can be applied to check the process model for compliance with imposed rules. Depending on the process meta model employed, structural compliance checking can be conducted in a more efficient manner than behavioral compliance checking.

We can draw parallels to Petri Nets research. Since reachability analysis is costly, researchers also developed strategies to structurally analyze Petri Nets which promise a more efficient checking of certain Petri Net properties [8].

While behavioral compliance checking is a general approach and thus is broadly applicable, efficient structural compliance checking relies on certain conditions. For example, structural compliance checking of data-aware compliance rules is rather not feasible, since the data dimension has to be explored. To provide support for a multitude of compliance verification settings, we integrated a structural as well as a behavioral compliance checking approach into SeaFlows Toolset. The structural compliance checking approach, introduced in Sect. 3.1, conducts efficient compliance checking for a subset of CRGs. The behavioral approach, discussed in Sect. 3.2, addresses data-aware compliance checking and provides strategies to avoid state explosion.

3.1 Structural Compliance Checking Based on Node Relations

The basic idea of our structural compliance checking approach is to automatically derive *structural criteria* on the process model from behavioral compliance rules. We first introduced structural compliance checking in [9], however, so far we only addressed basic exclusion and dependency constraints. We have further extended our approach to provide support for a broader range of compliance rules. Based on the assumption of unique labels (i.e., unique occurrences of activities within a process model), we have developed an efficient structural compliance checking approach for a subset of CRGs. This approach is designed to support loop-free process models and abstracts from data conditions.

Our structural compliance checking approach is conducted in three steps as illustrated in Fig. 4. In Step 1, for each CRG, a set of structural criteria to be

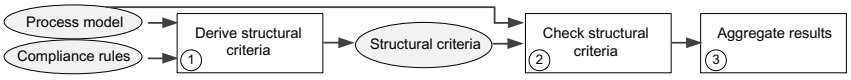

Fig. 4. The process of structural compliance checking

checked over the process model is determined automatically. These criteria can be considered queries on the relations of nodes within the process model (i.e., *node relations*) that are relevant to the compliance rule. We define five structural criteria consisting of containment, occurrence, and precedence relations:

$\oplus$ **A (contains A)** is a unary structural containment relation that applies if A is contained in the process model.

A $\otimes$ B (A excludes B) is a structural occurrence relation that applies if A and B are located on different branches of an exclusive gateway. For example, `10% discount` $\otimes$ `check solvency` applies to process model P from Fig. 1.

A $\triangleright$ B (A implies B) is a non-directed structural occurrence relation. In order for it to evaluate to `true`, B must not be located on a branch of an exclusive gateway, on which A is not located as well, provided that A and B are both present in the process model. For example, `assess order` $\triangleright$ `check solvency` will be evaluated to `false` over the order-to-delivery process from Fig. 1, since `check solvency` is located on a branch of an exclusive gateway (i.e., branch with condition `c=premium`) while `assess order` is not located on that branch. This means, that `check solvency` will be executed optionally to `assess order`. However, the node relation `check solvency` $\triangleright$ `assess order` will be evaluated to `true`, since `assess order` is located on the exclusive branch with data condition `pn>50,000` and `check solvency` is also located on this branch.

A $\triangleright$ $B_1|B_2|\ldots|B_n$ (A implies B_1,B_2,..., B_n) is a non-directed structural occurrence relation. It applies if A is always executed together with B_1,B_2,..., or B_n. At the structural level, this criterion is checked by adopting strategies from data flow analysis.

A $\gg$ B (A precedes B) is a structural precedence relation that applies if there is a directed path in the process model leading from A to B. For example, `prepare goods` $\gg$ `ship goods` will be evaluated to `true` over the order-to-delivery process from Fig. 1.

In Step 2, the process model is checked for compliance with the derived structural criteria. Thus, we can precisely identify those structural criteria causing incompliance. In case a compliance violation is detected, these structural criteria will be collected in Step 3 and will be used for error diagnosis and for the generation of intelligible feedback. By showing the process designer, which structural criteria are not satisfied by the process model, the system can help to resolve incompliance. By exploiting properties of the process meta model properties such as block-structuring, the structural criteria can be evaluated efficiently. Adopting the paradigm of dynamic programming, we cache node relations already queried to enable faster evaluation if the same relations are queried a second time.

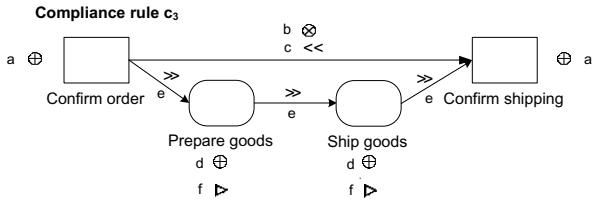

Fig. 5. A CRG annotated with structural criteria to be applied for conducting structural compliance checking

In the following, we will sketch how structural criteria are derived and checked for compliance rule c_3 from Table 1 and the corresponding CRG from Fig. 3.

Example. In Step 1, structural criteria to be queried in order to verify compliance with c_3 are derived based on c_3's structure. Fig. 5 depicts CRG c_3 annotated with types of structural criteria that will be applied during structural compliance checking and the order of application. Exploiting the antecedent-consequence structure of CRGs, it is first checked whether the process behavior modeled by the CRG's antecedent pattern can be produced by the process model. For CRG c_3 the antecedent pattern consists of `confirm order`, `confirm shipping`, and the ordering relation between them. If the antecedent can be activated (i.e., the pattern can be produced by the process model), structural compliance checking proceeds with checking compliance with the consequence pattern.

a. Premise to the activation of an antecedent pattern by a process model is that relevant antecedent activities are contained in the model. Hence, as the first step, it is checked wether the activities `confirm order` and `confirm shipping` assigned to the AnteOcc nodes are both contained in the process model using the $\oplus$ relation. If one of them is not contained in the process model, the process behavior modeled by the antecedent CRG can never be produced and further checking would not be necessary.

b. The antecedent pattern can only be produced by a process model if the AnteOcc activities `confirm order` and `confirm shipping` are located in the model such that they can be executed together. Whether this applies can be determined by checking whether these activities exclude each other by querying `confirm order` $\otimes$ `confirm shipping`. If this is the case, the antecedent pattern can never occur, which, in turn, makes further compliance checking superfluous.

c. The AnteOcc activities `confirm order` and `confirm shipping` are assigned an ordering relation in c_3. To check whether ordering relations apply, the $\gg$ relation is employed. Provided that a and b have not led to an interruption of the checking, `confirm order` can only be followed by `comfirm shipping` if precedence relation `confirm shipping` $\gg$ `confirm order` does not apply (i.e., the activities are ordered the other way around or in parallel branches).

d. In order to check compliance with the consequence pattern, it is first queried whether the relevant activities `prepare goods` and `ship goods` are contained in the process model. This is done by using the $\oplus$ relation. If one of these acitivites

are not contained in P, the consequence of the compliance rule apparently will never be satisfied and compliance checking can be ceased.

e. Ordering relations modeled in a CRG's consequence pattern also have to be verified. In our example, the correct ordering of `prepare goods` and `ship goods` with regard to each other as well as with regard to the AnteOcc activities `confirm order`and `confirm shipping` is checked by applying the $\gg$ relation.

f. According to c_3, `prepare goods` and `ship goods` always have to be executed when `confirm order` and `confirm shipping` are executed. This can be queried using the $\triangleright$ relation. Activity `prepare goods` for example will always be executed when `confirm order` and `confirm shipping` are executed, if it is structurally implied by one of the latter activities. This can be checked by apply the following query: (`confirm order` $\triangleright$ `ship goods`) $\vee$ (`confirm shipping` $\triangleright$ `ship goods`). If the query applies, the execution of `ship goods` together with `confirm order` and `confirm shipping` will be ensured. The same has to be checked for `prepare goods`.

Fig. 6 shows the results of querying the derived structural criteria over the order-to-delivery process P from Fig. 1 as obtained in Step 2. The criteria are arranged along a decision tree. Particular query results are printed in boldface. As Fig. 6 shows, the antecedent pattern can be produced by P. In addition, the required activities of the consequence pattern (i.e., `prepare goods` and `ship goods`) are both contained in P and fulfill the required precedence relations. However, while occurrence relation query OR3 is evaluated to `true`, OR2 does not

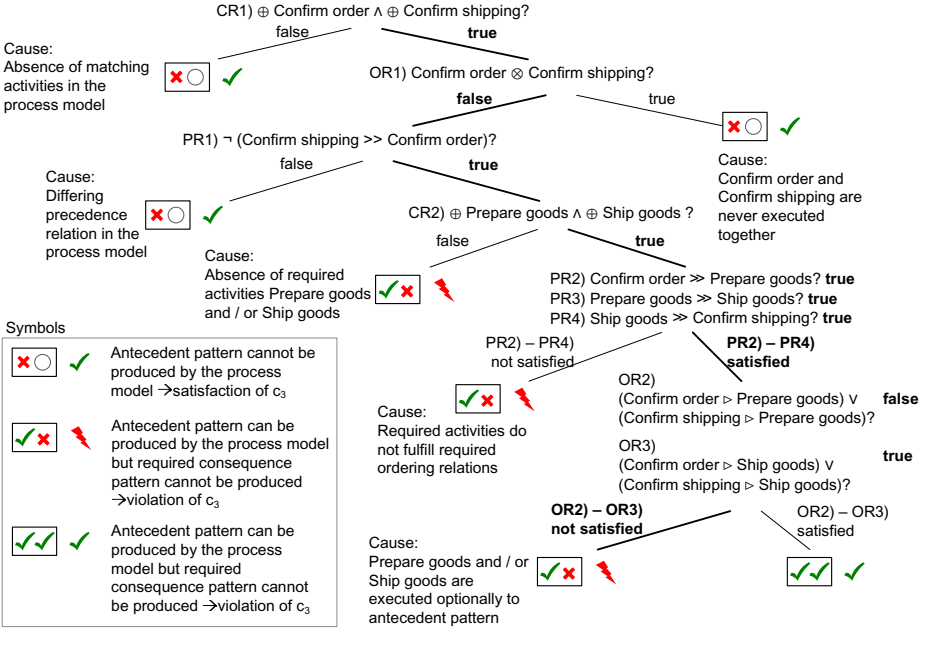

Fig. 6. Structural compliance checking of c_3 against the order-to-delivery process

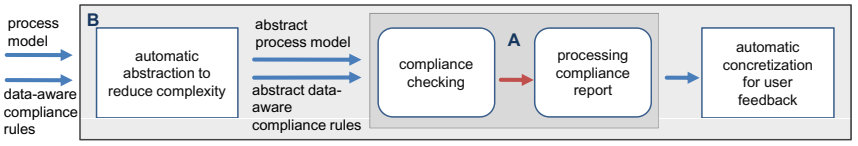

Fig. 7. Abstraction and concretization as pre- and postprocessing steps to the actual data-aware compliance checking

apply. This means that `prepare goods` is executed optionally to both `confirm order` and `confirm shipping`. Thus, at the behavioral level, executing P can lead to execution traces containing both `confirm order` and `confirm shipping` but not `prepare goods`. Hence, P does not comply with c_3.

In Step 3, the results obtained from querying the structural criteria can be aggregated and used to generate error diagnosis. In our example, we can precisely identify the violation of OR2 as the cause of incompliance. In Sect. 4.2, we will show how the *SeaFlows Structural Compliance Checker* uses the query results to generate compliance reports.

3.2 Data-Aware Behavioral Compliance Checking

As previously discussed, behavioral compliance checking is a general approach, and thus, broadly applicable. Its particular advantage over structural compliance checking becomes manifest, when it comes to data-aware compliance checking. Data-awareness is vital to support a variety of compliance verification settings. A closer look at compliance rules and scenarios reveals two major scenarios:

a) Compliance rules without data conditions, but the process model contains data-based gateways that can affect the verification outcome.
b) Compliance rules containing data conditions.

An example of case a) is given by compliance rule c_4 from Table 1. While c_4 does not contain any data conditions, abstracting from the data perspective when checking compliance of the order-to-delivery process P from Fig. 1 with c_4 would lead to an erroneous compliance report. In particular, abstracting from the data-based gateways contained in process model P would lead to the conclusion that a solvency check is not sure to be executed prior to offering premium customer status. However, due to the correlation of data-based gateways within P, it is ensured that premium customer status is only offered after a prior solvency check. Note that `check solvency` is carried out for non-premium customers and `pn>100,000` while `offer premium status` is only carried out for non-premium customers and `pn>150,000`. Hence, the order-to-delivery process, in fact, complies with c_4. For examples of case b), consider compliance rules c_7 and c_8 from Table 1 and corresponding CRGs in Fig. 3, each containing data conditions.

As these examples show, in both cases, data-awareness is necessary to provide correct compliance reports. The challenge with data-aware compliance checking

is that the exploration of the data dimension during compliance checking can lead to state explosion and thus, to intractable complexity. To tackle this, we introduced abstraction strategies to reduce the complexity of data-aware compliance checking [10]. By analyzing the data conditions contained in the compliance rule and in the process model, our approach reduces the state space of the data dimension to be explored during verification. This is achieved by abstracting from concrete states of data objects to abstract states. Based on the compliance rules to be checked our approach *automatically* derives an *abstract process model* and corresponding *abstract compliance rules* (cf. Fig. 7 B). The abstract process model is more compact than the original process model with regard to the data dimension. Thus, it enables more efficient exploration of the data dimension when being used as input to the actual compliance checking (cf. Fig. 7 A).

For conducting behavioral compliance checking for the abstract process model, we apply model checking techniques. In case of violation, the counterexample obtained from the model checker is concretized to yield not only the incompliant execution but also its data conditions. To address both scenarios a) and b), the *SeaFlows Data-aware Compliance Checker* is able to deal with data-aware compliance rules (i.e., compliance rules containing data conditions) and data conditions in process control flow (i.e., data-based gateways).

4 SeaFlows Toolset

We implemented the concepts introduced here in SeaFlows Toolset. It extends process-aware information systems (PAIS) by compliance checking functionality. Fig. 8 depicts the interplay between existing infrastructure stemming from PAIS (e.g., activity repository, process modeling tool, and process model repository) and components introduced by SeaFlows Toolset[1].

The *SeaFlows Graphical Compliance Rule Editor* (cf. Fig. 8) allows to model CRGs over process artifacts (cf. Sect. 2). By interacting with the activity repository managing process artifacts relevant within a business domain, the Graphical Compliance Rule Editor enables compliance rule modeling over exactly the process artifacts available in the domain. Thus, we can enrich process models by compliance rules that are imposed on the corresponding business process. This can be done at an early stage, when the process is modeled to enable *compliance by design*. Compliance rules may be also assigned to a completed or released process model to conduct compliance audits.

SeaFlows Toolset currently comprises two compliance checking components to verify process models (cf. Fig. 8). The *Structural Compliance Checker* implements the structural compliance checking approach while the *Data-aware Compliance Checker* employs a behavioral approach and addresses data-awareness. By interacting with the process modeling tool of PAIS, the SeaFlows compliance checkers enable process designers to verify process models already during process

[1] The Rule Graph Execution Engine for executing CRGs is currently under implementation.

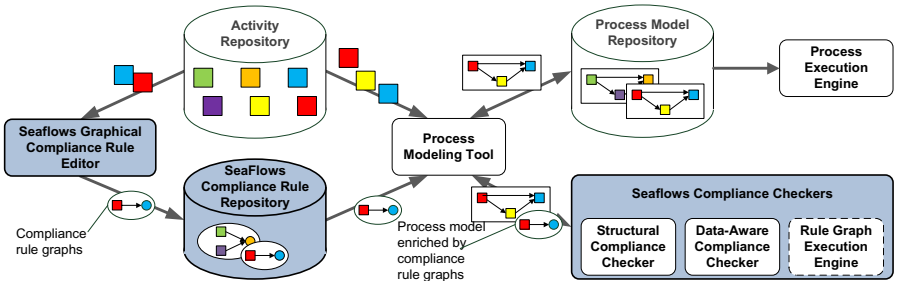

Fig. 8. Overall infrastructure around the SeaFlows Toolset

design. Meaningful compliance reports help process designers to identify incompliant process behavior. Based on them, process designers may further modify the process model until incompliance is resolved.

To transfer our concepts into a comprehensive prototype, we opted to base our implementation on AristaFlow BPM Suite which originated from research activities in the ADEPT project [11]. AristaFlow BPM Suite provides a powerful application programming interface (API) which enables us to extend existing PAIS functionality by compliance checking mechanisms in an elegant manner. Thus, SeaFlows compliance checking components are smoothly integrated into the process modeling environment of AristaFlow BPM Suite. In the following, the components of SeaFlows Toolset are discussed in more detail.

4.1 Graphical Compliance Rule Editor and Compliance Rule Repository

The *Graphical Compliance Rule Editor* provides a user-friendly environment for modeling CRGs. Fig. 9 shows c_3, c_4, c_7, and c_8 as modeled using the compliance rule editor. Nodes of CRGs are assigned to activity types available in the activity repository (cf. Fig. 8). Modeled CRGs are exported as separate XML-files which enables their organization within rule sets in the Compliance Rule Repository. In addition, versioning and collaborative modeling of compliance rules are also supported by the repository. Being implemented based on the Eclipse Modeling Framework, modeled CRGs are based on a defined data object model which facilitates their import and processing in compliance checking tools.

4.2 Structural Compliance Checker

Based on the results of checking the structural criteria, the *Structural Compliance Checker* is able to provide detailed diagnosis helpful to locate incompliance. Fig. 10 shows the compliance report for checking compliance rule c_3 against the order-to-delivery process P. Corresponding to Fig. 6, the Structural Compliance Checker identifies that while there is no problem with required activity `ship goods`, the other required activity `prepare goods` is optional to both

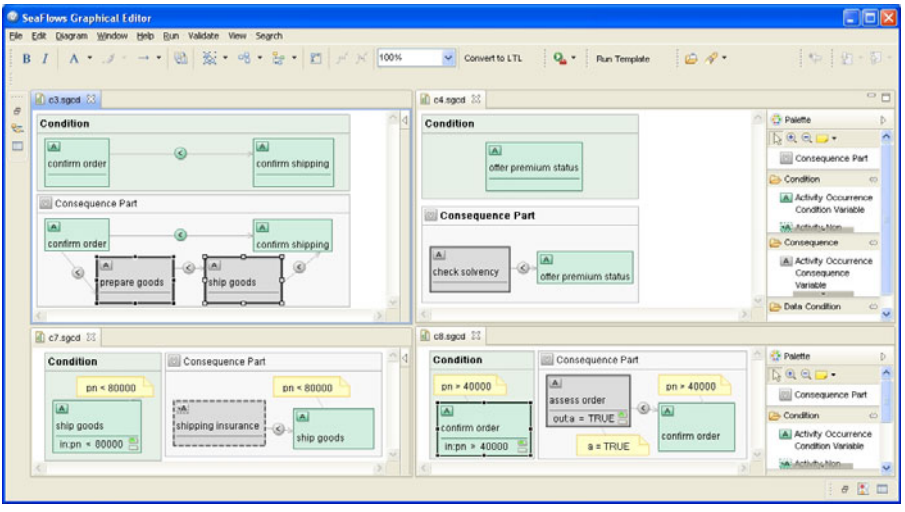

Fig. 9. The SeaFlows Graphical Compliance Rule Editor

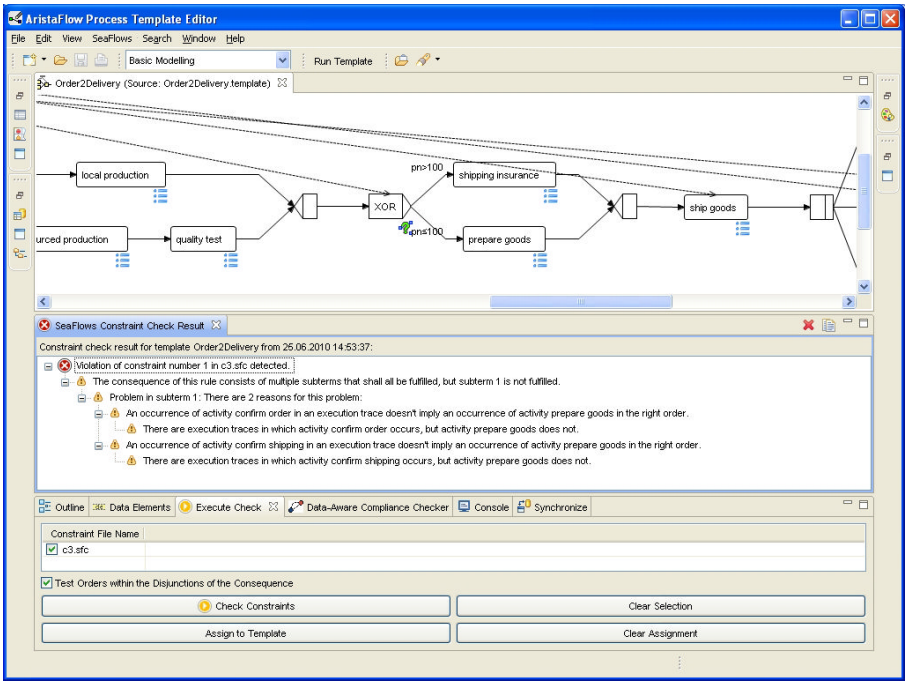

Fig. 10. The SeaFlows Structural Compliance Checker integrated into AristaFlow Process Template Editor

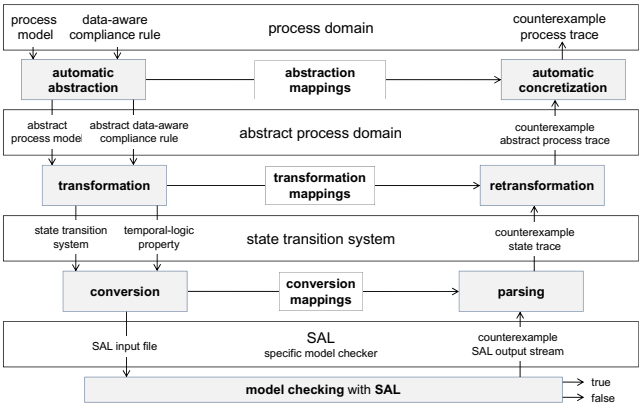

Fig. 11. Data-aware compliance checking and generation of counterexample

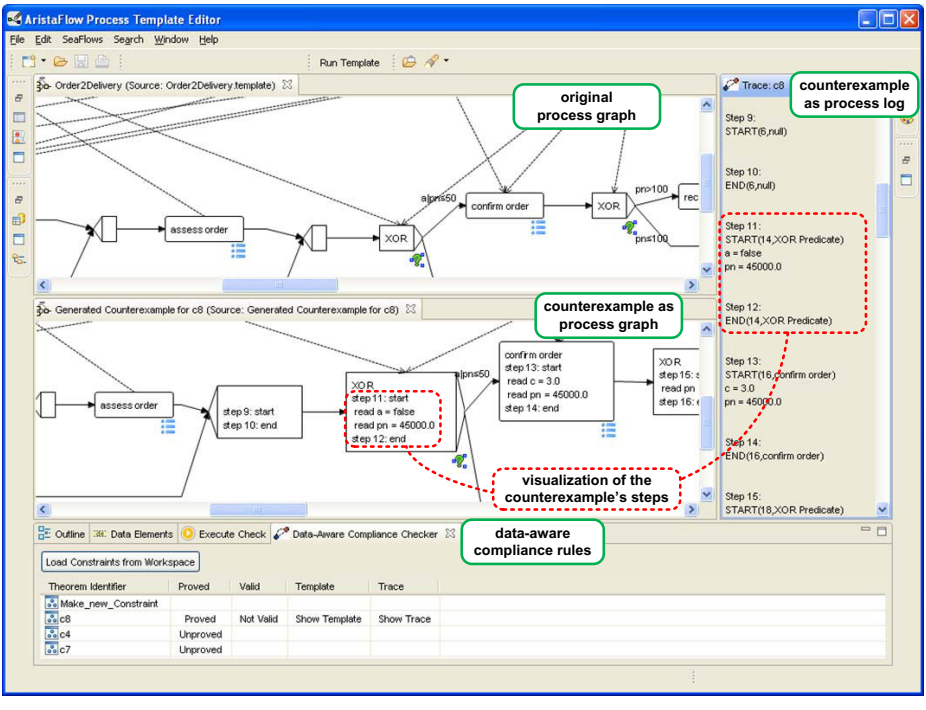

Fig. 12. The Data-aware Compliance Checker visualizes the counterexample as execution trace and process graph

`confirm order` and `confirm shipping` in P. This detailed diagnosis can further be applied to resolve incompliance. For example, the present incompliance can be resolved by repositioning activity `prepare goods` in the P.

The *Structural Compliance Checker* is implemented as an Eclipse-plug-in for the AristaFlow Process Template Editor and thus, is smoothly integrated into the process modeling environment. Therefore, "on the fly" compliance checks during process modeling can be carried out to support compliance by design.

4.3 Data-Aware Compliance Checker

The *Data-aware Compliance Checker* first performs automatic abstraction (cf. Fig. 7). Then, it transforms the abstract process model into a state space representation to apply behavioral compliance checking. For the actual compliance checking, we employed the model checker SAL [12], which, in turn, performs automatic exploration of the state space model and checks for conformance to the compliance rule (cf. Fig. 11). In case incompliance is detected, the *Data-aware Compliance Checker* retransforms the counterexample output of the model checker and visualizes it as an execution trace and within the originial process model. In addition to the violating execution the data condition under which the violation occurred is also illustrated. Fig. 12 shows the output of the checker for the verification of compliance rule c_8 against the order-to-delivery process.

The *Data-aware Compliance Checker* is integrated into the process modeling environment. The class hierarchy comprising about 70 interfaces and 210 classes indicates its complexity. Automatic abstraction is supported for domains of numbers and for large domains of object references.

5 Related Work

Three major challenges arise from compliance verification of process models: compliance rule modeling, verification techniques, and feedback generation. The concepts implemented in SeaFlows Toolset address all three issues. Existing approaches for modeling compliance rules range from rather informal annotations of process models with compliance rules, over formal languages [13], to visual patterns and languages [14, 4, 15]. With the CRGs, we opted for a compositional graph-based modeling formalism that supports the typical antecedent-consequence-structure of rules.

For compliance verification, model checking is often applied in literature [15,14, 13]. As advantage we obtain an approach that is not specific to a particular process meta model or process modeling notation. One challenge of model checking, however, is the generation of meaningful feedback from the report provided by the model checker (e.g., counterexample). The *Data-aware Compliance Checker* addresses this challenge and enables visualization of the counterexample within the process model. The major drawback of model checking is the complexity caused by the necessary model transformations and particularly the exploration of the state space representation. Our structural compliance checking approach exploits specific meta model properties, such as block-structuring, to enable more efficient compliance verification. Some approaches address the verification of data-aware compliance rules [4, 15]. However, the state explosion arising from exploration of the data dimension is not addressed by these approaches. In SeaFlows Toolset we

implemented an abstraction approach that serves as preprocessing step to the actual data-aware compliance checking to limit state explosion.

In [16], Awad et al. address visualization of incompliance by querying the process model for anti-patterns defined for each compliance rule pattern. In our structural approach, structural criteria are automatically derived from CRGs. Checking the structural criteria allows for identifying precisely the structural cause of incompliance.

Similar to DECLARE [17] SeaFlows enables to model graphical compliance rules. In DECLARE constraints are mapped onto formulas in temporal logic and then to finite automata in order to execute constraint-based workflows. In contrast, SeaFlows CRGs are used to verify process models.

SeaFlows Toolset can be further complemented by other process analysis tools, such as the process mining framework ProM [18] to provide comprehensive support of compliance checking a priori as well as a posteriori.

6 Summary and Outlook

In this paper, we introduced concepts underlying SeaFlows Toolset. We showed how compliance rules can be modeled as CRGs to provide a visual process-like and yet still formal representation. In addition, we discussed complementary compliance checking strategies, namely structural compliance checking and behavioral compliance checking. We introduced our structural compliance checking approach based on node relations that enables efficient verification of process models against imposed compliance rules. To address further scenarios, we introduced a behavioral compliance checking approach that addresses data-awareness. SeaFlows Toolset extends process-aware information system by compliance checking functionality. It enables modeling CRGs independently from specific process models by making use of an activity repository. Process models can be enriched by CRGs for documentation purposes and for compliance verification. Two compliance checkers, the *Structural Compliance Checker* and the *Data-aware Compliance Checker*, addressing specific compliance verification scenarios complement each other and thus, ensure broad applicability.

In our future work, we will further extend SeaFlows Toolset to provide support for compliance checking during process execution (cf. the SeaFlows Rule Graph Execution Engine in Fig. 8). In addition, SeaFlows Toolset will be extended by a visualization and explanation component to provide advanced visual user feedback. Finally, case studies can be conducted to validate the concepts implemented in SeaFlows Toolset.

References

1. Sadiq, W., Governatori, G., Namiri, K.: Modeling control objectives for business process compliance. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 149–164. Springer, Heidelberg (2007)
2. Bobrik, R., Reichert, M., Bauer, T.: View-based process visualization. In: Alonso, G., Dadam, P., Rosemann, M. (eds.) BPM 2007. LNCS, vol. 4714, pp. 88–95. Springer, Heidelberg (2007)

3. Ly, L.T., Rinderle-Ma, S., Dadam, P.: On enabling integrated process compliance with semantic constraints in process management systems. Information Systems Frontiers, 1–25 (2009)
4. Awad, A., Weidlich, M., Weske, M.: Specification, verification and explanation of violation for data aware compliance rules. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) ICSOC-ServiceWave 2009. LNCS, vol. 5900, pp. 500–515. Springer, Heidelberg (2009)
5. Yu, J., Manh, T., Han, J., Jin, Y., Han, Y., Wang, J.: Pattern based property specification and verification for service composition. In: Aberer, K., Peng, Z., Rundensteiner, E.A., Zhang, Y., Li, X. (eds.) WISE 2006. LNCS, vol. 4255, pp. 156–168. Springer, Heidelberg (2006)
6. Namiri, K., Stojanovic, N.: A formal approach for internal controls compliance in business processes. In: 8th Workshop on Business Process Modeling, Development, and Support, BPMDS 2007 (2007)
7. Ly, L.T., Rinderle-Ma, S., Dadam, P.: Design and verification of instantiable compliance rule graphs in process-aware information systems. In: Pernici, B. (ed.) CAiSE 2010. LNCS, vol. 6051, pp. 9–23. Springer, Heidelberg (2010)
8. Barkaoui, K., Ben Ayed, R., Sbaï, Z.: Workflow soundness verification based on structure theory of petri nets. International Journal of Computing & Information Sciences 5(1), 51–61 (2007)
9. Ly, L.T., Rinderle, S., Dadam, P.: Semantic correctness in adaptive process management systems. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 193–208. Springer, Heidelberg (2006)
10. Knuplesch, D., Ly, L.T., Rinderle-Ma, S., Pfeifer, H., Dadam, P.: On enabling data-aware compliance checking of business process models. In: Parsons, J., Saeki, M., Shoval, P., Woo, C., Wand, Y. (eds.) ER 2010. LNCS, vol. 6412, pp. 332–346. Springer, Heidelberg (2010)
11. Dadam, P., Reichert, M.: The ADEPT project: a decade of research and development for robust and flexible process support. Computer Science - Research and Development 23(2), 81–97 (2009)
12. Bensalem, S., et al.: An overview of SAL. In: Proc. of the Fifth NASA Langley Formal Methods Workshop, NASA Langley Research Center, pp. 187–196 (2000)
13. Ghose, A., Koliadis, G.: Auditing business process compliance. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749, pp. 169–180. Springer, Heidelberg (2007)
14. Förster, A., et al.: Verification of business process quality constraints based on visual process patterns. In: Proc. 1st Joint IEEE/IFIP Symposium on Theoretical Aspects of Sofware Engineering, pp. 197–208. IEEE Computer Society, Los Alamitos (2007)
15. Liu, Y., Müller, S., Xu, K.: A static compliance-checking framework for business process models. IBM Systems Journal 46(2), 335–361 (2007)
16. Awad, A., Weske, M.: Visualization of compliance violation in business process models. In: Business Process Management Workshops. LNBIP, vol. 43, pp. 182–193. Springer, Heidelberg (2010)
17. Pesic, M., Schonenberg, M., Sidorova, N., van der Aalst, W.: Constraint-based workflow models: Change made easy. In: Meersman, R., Tari, Z. (eds.) OTM 2007, Part I. LNCS, vol. 4803, pp. 77–94. Springer, Heidelberg (2007)
18. van der Aalst, W.M.P., van Dongen, B.F., Günther, C.W., Mans, R.S., de Medeiros, A.K.A., Rozinat, A., Rubin, V., Song, M., Verbeek, H.M.W(E.), Weijters, A.J.M.M.T.: ProM 4.0: Comprehensive support for *real* process analysis. In: Kleijn, J., Yakovlev, A. (eds.) ICATPN 2007. LNCS, vol. 4546, pp. 484–494. Springer, Heidelberg (2007)

Decisions and Decision Requirements for Data Warehouse Systems

Naveen Prakash[1], Deepika Prakash[2], and Daya Gupta[2]

[1] MRCE, Sector-43,
Faridabad, India
praknav@hotmail.com
[2] DTU,
Delhi, India
dpka.prakash@gmail.com, daya_gupta@yahoo.co.in

Abstract. We develop the notion of a decision requirement as the pair <decision, information> where 'information' is that required by the decision maker to assess if the 'decision' is to be taken or not. It is shown that there are two kinds of decisions, imperative and managerial. The former are decisions about which transactional service out of a choice of transactional services is to be provided. Managerial decisions determine what infrastructure out of a set of possibilities is to be put in place. It is shown that a decision is the reason why a functionality of an information system is invoked. The notion of decision requirement is clarified through a decisional requirement meta model. This is supported by a decision and information meta model. The example of a health scheme is taken to illustrate the different kinds of decisions and decision requirements.

Keywords: Decision, Information, Data Warehouse.

1 Introduction

Goal oriented requirements engineering techniques [1-5] have been developed in the area of information systems/software engineering. These techniques aim to discover the functions of the system To-Be and lay the basis for system design.

The role of Requirements engineering in developing Data Warehouses has been investigated only in the last decade or so [6-13]. It has been argued [13] that Requirements engineering for DW systems focuses on **discovery of data** whereas for information/software systems, it focuses on **system functionality**. Early Data Warehouse development did not emphasize Requirements engineering. Indeed, Inmon [20] argued that requirements of data warehouses are usually the last thing to be discovered. Interest was in using existing databases for development of data warehouse systems. This was augmented by ER driven approaches where ER diagrams could be converted to star schemata. Now, there is a body of opinion that uses goal oriented techniques [10, 11, 13, 15, 16] for determining data warehouse structure.

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 92–107, 2010.
© Springer-Verlag Berlin Heidelberg 2010

One goal-oriented approach [10, 11, 13] is based on the notion of the Goal-Decision-Information diagram. This approach postulates that the decision making capacity is determined by organizational goals. Additionally, it associates the information that has a bearing on a decision with the decision itself. In this paper, we represent this association as a pair, <decision, information> and refer to it as a **decision requirement**. Thus, in order to represent data warehouse contents, the set of decision requirements must be explicitly modeled. A given set of decision requirements evolves over time. The evolution of data warehouse is outside the scope of this paper. Further, this paper does not directly address the issue of conversion of decision requirements to data warehouse structures. We rely on the proposal of [13] to do this conversion.

Evidently there is a close relationship between the information systems and data warehouse of an organization. The former are used to populate the latter through the ETL process. In the opposite direction, the decision taken by using the data warehouse has the effect of changing information system contents. This means that information systems operate in a **decisional environment**. We consider this environment in the next section and show that there are two kinds of decisions, imperative and managerial. In the subsequent section we develop a meta model for decision requirements. Here we also model the notion of a decision and information from the data warehouse perspective. In section 4, we present a case study of a health scheme that uses the notion of a decision requirement. In section 5 we discuss our proposals with other related work.

2 The Decisional Environment

The decisional environment provides the context in which an information system (IS) operates. This is shown in Fig. 1. The interface between the decisional environment and an information system is the stimulus sent from the former to the latter. When the information system is sent a stimulus from the decisional environment then the functionality that responds to this stimulus is invoked.

This implies that (see Fig.1) the decisional environment provides the rationale for the stimulus and a decision is taken here to send the appropriate stimulus out a set of possible alternative stimuli. For example, the information system may provide for transfer of an employee as well as for recruitment of a new employee. In the decisional environment, it is to be decided which of the two possible stimuli should be sent: depending upon salary budgets and whether another department can spare an employee or not, either a transfer stimulus or a recruitment stimulus will be sent. We can therefore say that the decision is the cause of the invoked functionality.

There are two different kinds of actors for an information system, IS administrators and IS operators. It is the job of the former to ‘initialize’ the IS. The latter then work within the initialized IS to operate the system. For example, consider a railway reservation system. IS administrators shall initialize the trains, their start stations, destinations and stop over stations, fares etc. that are used in operating the reservation system. They also handle changes: additions or deletions of trains, stations, changes in fare structure and so on. IS operators, in contrast invoke functionality to make reservations on given trains using the fare table and other information set up by the IS administrator.

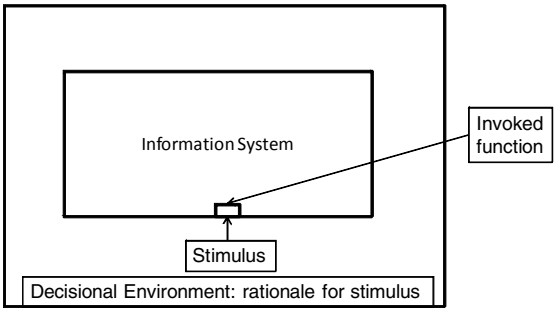

Fig. 1. Embedded IS in a Decisional Environment

We see that there are two kinds of stimuli that the decisional environment can send: those decided by IS administrators and operators respectively. This leads to two kinds of decisions, that we call managerial and imperative decisions respectively.

2.1 Imperative Decisions

Let there be a manager who has to perform extra work and needs to allot it to an employee. He can find the employee by either transferring one from another department to this one, hire an altogether new employee or overload an existing employee with more responsibility. Thus, the choice set is {Transfer employee, Hire employee, Overload employee}. The manager needs information to decide which alternative to pick and, also which individual employee shall be transferred, hired or overloaded respectively. Thus, there are two decision making problems here, (a) to select from the choice set of functions as above, which we call tactical decisions, and (b) to identify the individual from the choice set of individuals that we refer to as operational decisions.

Tactical decisions supply the rationale for operational ones. In tactical decision making, the state of the organization is browsed through and analyzed to select fromt eh choice set. For example, Transfer employee may be selected if a department is seen to be overstaffed, or an ill fitting employee has been posted there etc. Similarly, a new employee may be recruited because completely new skills are required.

The best alternative selected is the rationale behind an operational decision. It calls for formulation of the stimulus that shall be sent to the information system. Operational decision making is the process of selection of the appropriate individual from a choice set. For example, given the tactical decision Transfer employee, the operational decision maker needs to identify which one to transfer. The choice set consists of employees, and operational decision making leads to selection of one of these. Data about employees, their skills performance, past record, etc. is picked up from the data warehouse, analyzed and the most relevant employee data is sent as stimulus to the IS function that shall carry out the transfer.

Consider Fig. 2 which shows the interplay of tactical and operational decisions. The tactical decision making environment contains the operational decision making environment. Both environments refer to the data warehouse as shown by the dashed lines in the figure. Once a tactical decision is made, it is sent to the operational

decision making environment which sends a stimulus to the information system. Fig. 2 shows that the tactical decision to Transfer an employee enters the operational decision making environment where the employee is identified and the stimulus to be sent to the information system is completely formulated. The information system performs the desired function and this information is now available to be sent to the DW at refresh time.

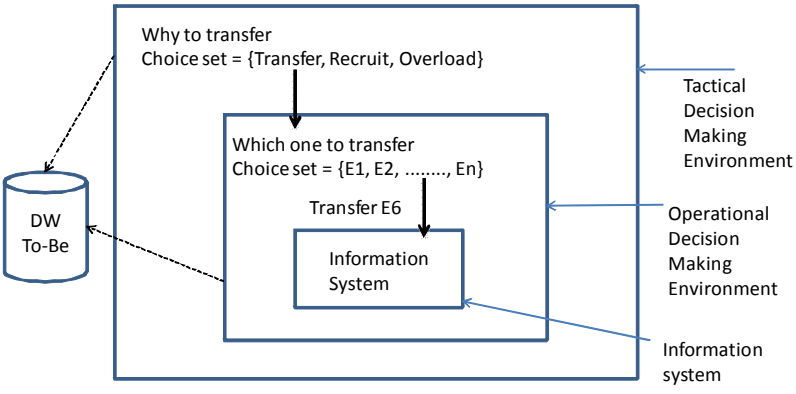

Fig. 2. Imperative Decisions

Since, the data warehouse is consulted in both forms of imperative decision making, full information about decisions and related information must be kept in it.

Looking from the information system outside, the decision making layers surrounding it formulate the stimulus to which the IS responds. This stimulus must identify IS functionality and the data. The former is done in the tactical environment whereas the latter is done in the operational decision making environment.

2.2 Managerial Decisions

The class of managerial decisions is not concerned with operational functionality/ transactional capability but with performing the administrative tasks to set up the system so that imperative decisions can be taken. Recall the reservation example cited earlier where it is necessary to initialize trains, fare tables etc.

We postulate that there are two kinds of managerial decisions, those that follow a business policy, enforce it or create exceptions to it, and those that formulate the policy. We refer to the former as **administrative decisions**, since they are concerned with administering the system and to the latter as **policy decisions**. These two kinds of managerial decisions are related to one another. The latter provides the context for the former.

Let us be given a policy that the number of first class bogeys in a train should be one third of the train length. The rest are second class bogeys. In other words, the ratio first class: second class :: 1:2. This norm is usually to be followed. However, there could be exceptional situations like the train is making losses or it is overloaded.

In such situations this norm could be violated, and an administrative decision is needed to choose from the choice set, {Modify number of first class bogies, Modify number of second class bogies}. This administrative decision creates an exception to the norm.

Policy decisions are of two kinds. They define the norms and standards that are used by administrative decisions and also define business rules. They address the question of where a norm or standard comes from. Norms and standards may be defined externally, for example, by regulatory bodies, or internally within the organization. Assuming that external norms and standards are imposed on an organization, policy decision making is concerned with formulating internal ones. Consider the internal norm first: second:: 1:2 of the railway system considered earlier. Defining this through a policy decision requires knowledge of the state of the organization, patterns of bookings made, revenue targets, revenue receipts etc. Out of the many choices available to fix the ratio, the policy decision maker uses this knowledge to fix the desired one.

The second type of policy decisions formulates business rules. Let it be that the number of days in advance a booking can be made is to be decided. Again, reference to existing information like pricing and booking pattern is made and the manager decides it to be 30 days.

The choice set available to the policy decision maker is {Define, Modify, Delete}.

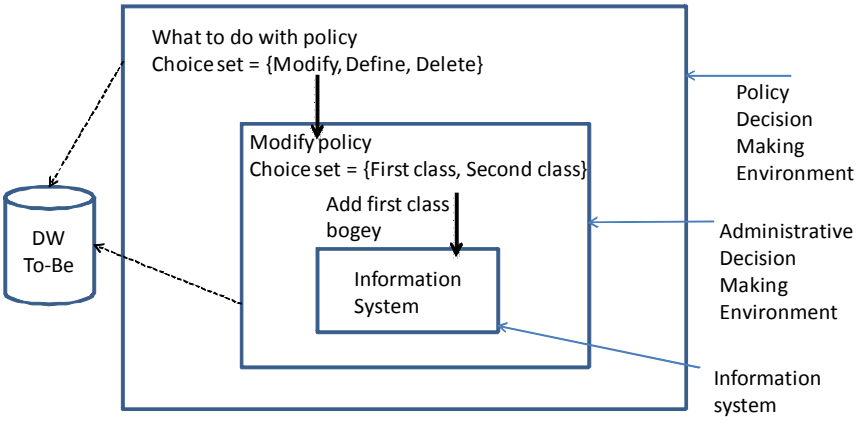

Fig. 3. Managerial Decisions

The interplay of policy and administrative decisions is shown in Fig. 3. Both the policy and administrative decision making environments refer to the data warehouse with the latter contained in the former. Once a policy decision is made, it is sent to the administrative decision making environment for enforcement. The information system is sent a stimulus from the administrative decision making environment to which it responds. Fig. 3 shows the policy decision to modify the ratio of first to second class bogeys in a train. The administrative decision to add a first class bogey is taken, and the information system is stimulated to reflect the change. This information is now available for train reservation purposes and is also available to be sent to the DW.

Notice that even if a policy is held constant, an administrative decision may still be made. Thus, for example, even when the first class to second class ratio is not changed, an administrative decision to increase/decrease this ratio may be taken. This happens when there are sudden surges of demand for a particular kind of bogey.

As for imperative decisions, it is possible to support decision making of both kinds of managerial decisions. The data warehouse needs to be populated with information associated with these.

3 Decision Requirement

We have seen that in order to pick an alternative from the choice set, reference to the information in the data warehouse needs to be made. This information is a specification of the data to be kept in the data warehouse To-Be. We represent this association as a pair <decision, information> and refer to it as a decision requirement.

In this section we elaborate on the notion of decision requirement. First we present the decision requirement meta model and then develop meta models for the notion of a decision and information.

3.1 The Decision Requirement Meta-model

The Decision Requirement, DR, meta-model is shown in Fig. 4. As shown it is modeled as an aggregate of information and decision. It can be expressed as a pair,

$$DR = <D, I>$$

where D is a decision and I is the information required for it.

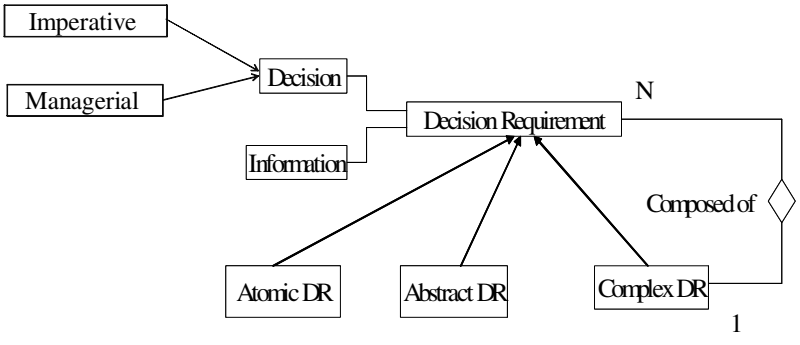

Fig. 4. Decision Requirements Meta-model

Fig. 4 shows that there are three kinds of decision requirements, atomic, abstract and complex. An atomic DR is the smallest decision requirement. It cannot be decomposed into its parts.

An abstract DR is a decision requirement that is arrived using generalization/ specialization principles. This gives rise to ISA relationships between decision requirements. Finally, a complex DR is composed of other simpler decision

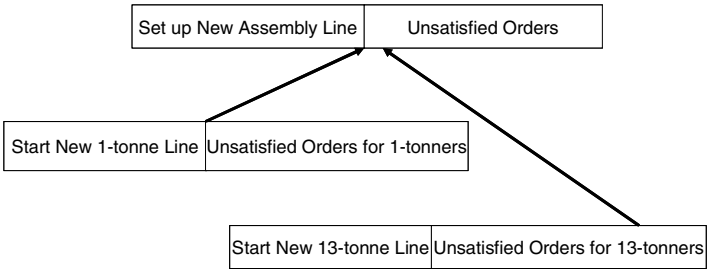

Fig. 5. An Abstract Decision Requirement ISA Hierarchy

requirements. Complex decision requirements form an AND/OR hierarchy in a manner analogous to AND/OR goal hierarchies of goal-oriented Requirements Engineering techniques.

To illustrate an abstract DR, consider an automobile plant that makes 1-tonne and 13-tonne trucks (Fig. 5). Let the decision of interest be *Set up New Assembly Line* and the required information be *Unsatisfied Orders.* The DR is shown in Fig. 5. This DR can be specialized into two DRs, with decisions *Start New 1-tonne Line* and *Start New 13-tonne Line* respectively and required information, *Unsatisfied Orders for 1-tonners* and *Unsatisfied Orders for 13-tonners*. Each of these is an ISA relationship with *Set up New Assembly Line*.

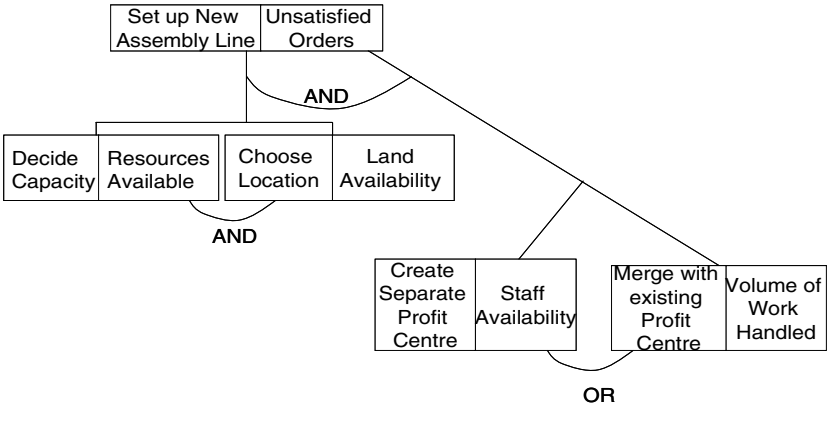

Fig. 6. Composition of Decision Requirements

Now let us illustrate a complex DR. The Decision Requirement <*Set up New Assembly Line, Unsatisfied Orders*> is a complex one having two component decision requirements, <*Decide Capacity, Resources Available*> and <*Choose Location, Land Availability*>. An AND link connects these two components so as to define the complex decision requirement, <*Set up New Assembly Line, Unsatisfied Orders*> (see Fig. 6).

Similar to AND composition, we can have OR composition. For example, when setting up a new assembly line, the question of whether or not it shall be part of an

existing profit centre has to be addressed. This gives us the decision requirements (Fig. 6), <*Create Separate Profit Centre, Staff Availability*> and <*Merge with existing Profit Centre, Volume of Work Handled*>. These two are in an OR relationship with one another. Notice that we have arrived at this through decision composition as well.

Aside from showing AND and OR decomposition of decisions, the foregoing also shows that a DR can be decomposed to reflect the decomposition of its decision component. It is also possible to do DR decomposition through information decomposition. In this case, the decision part is held constant whereas information components are elaborated. The Choose Location decision of Fig. 6 is shown as associated with the information, Land Availability. Land availability can be decomposed into two pieces of information, Land site and Land size

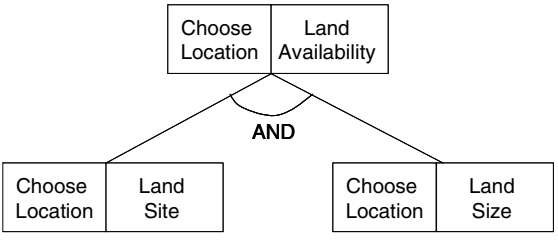

Fig. 7. Information Decomposition for DR Decomposition

Then the complex DR <Choose location, Land availability> can be decomposed into <Choose Location, Land site> and <Choose Location, Land size> respectively as shown in Fig.7.

3.2 Meta-model of Decisions

The key concept underlying the decision meta model of Fig. 8 is that of a decision parameter. Decision parameters denote different aspects of a decision. They reveal the factors that must be taken into consideration before a decision can be selected by the decision maker.

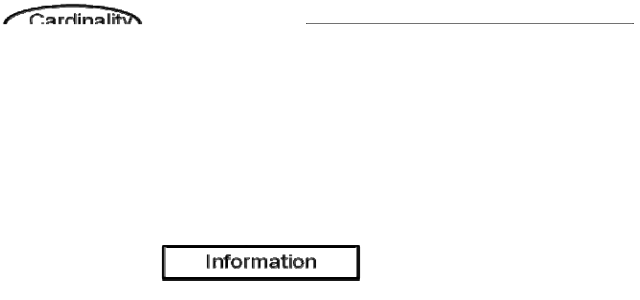

Fig. 8. Decision Meta Model

The decision to decision parameter relationship is M:N. There can be many decision parameters for a decision and a given decision parameter can be associated with more than one decision. A decision must have at least one decision parameter associated with it. This is because the absence of a decision parameter implies that there is no factor having a bearing to the decision. Therefore there is nothing to consider in taking the decision and it can be taken spontaneously. Such irrational decisions do not contribute to our objective of discovering the information to be maintained in the Data Warehouse and are therefore, out of our scope. Fig. 8 also shows that a decision parameter must be associated with at least one decision. This is to avoid irrelevant decision parameters from cluttering up the requirements engineering process.

It is possible to classify decision parameters in two different ways. According to one classification, a decision parameter may be dependent or independent. According to the second, a decision parameter may be a decision or information. We consider each of these in turn.

Dependent parameters depend on other, basic parameters for their existence. Those parameters that amplify the properties of other parameters are said to be dependent parameters. If the basic parameter becomes relevant or irrelevant, then parameters dependent on it also become relevant or irrelevant respectively. In contrast, independent parameters are orthogonal to one another. Each such parameter determines a completely new aspect of a decision. Independent parameters may have dependent parameters but are themselves not dependent on any other decision parameter for their existence.

Consider the decision Set_Up_New_Assembly_Line(Product Type, Location, Line Capacity). Here, the parameters, Product Type and Location are independent of one another. In contrast, Line capacity is dependent on Product Type since it is determined by the type of the product built by the line.

Now, consider the second typology of a decision parameter. A decision parameter which is a decision represents the decomposition property of decisions that is used for DR decomposition. It shows the decomposition structure of the decision. On the other hand a decision parameter that is information behaves as a specification of the data to be kept in the data warehouse To-Be.

In this paper we have given a graphical notation for the decision requirements and their structure. In [17] we have introduced a dependency graph to represent decision parameters and dependencies between them. We used this graph there to distinguish between those parameters that are decision and information (details of information are given in the next section) respectively.

3.3 Modeling Information

We have developed in Fig 9, our model of information from the point of view of Data Warehouse support for decision making. We are interested in three kinds of information, that which exists in the most detailed form, summarized information or aggregates, and historical information. Aggregate information is obtained as a summary by computing from simpler information. This is shown in Fig. 9, by the specialization of information into Simple and Aggregate as well as by the 'Is computed from' relationship between Aggregate and Information.

Historical information is represented by the relationship 'History of' between Information and Temporal unit. The cardinality of this relationship shows that it is possible for information to have no temporal unit associated with it. In such a case, only current information is to be maintained. However, when a temporal unit is associated with information then we must also know the number of years of history to be maintained. This is captured, as shown in the figure, in the attribute Period.

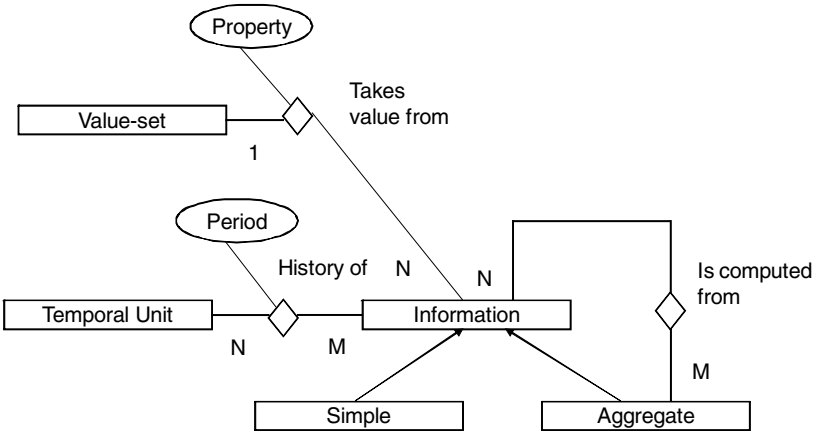

Fig. 9. Model of Information in Data Warehouses

Information is also associated with a value-set and takes on values from it. In Fig. 9 this association is called "Takes value from".

4 Case Study: A Health Scheme

The ideas presented here were applied to a health scheme operating in India. There are a number of dispensaries for providing medical services to members of the health scheme. There are well defined criteria for determining who can and who cannot be members. All eligible people may exercise the choice of obtaining membership of the health scheme and paying a fixed monthly charge. Once membership is granted, the beneficiary may go to any dispensary under the scheme and register for obtaining services there. The registrant may change dispensaries or discontinue membership in the scheme at will.

A dispensary is staffed with doctors who may refer registrants to other hospitals and specialists. A dispensary has in-house pharmacy with staff to disburse medicine. If a prescribed medicine is not available with this pharmacy, then a registrant may buy it from a notified list of external pharmacies and the costs shall be reimbursed. It is possible to obtain services of laboratories, radiology etc.

The information system provides a number of functions. For example, there are actions to consult a doctor, issue medicine, refer to a hospital, undergo tests etc. These functions are directly aimed at providing services to registrants. Additionally, there

are functions to induct a new doctor, add an external pharmacy to the notified list, empanel specialists and hospitals etc.

Imperative decisions related to health care can be made at the dispensary. For example, consider the tactical decision from the choice set {stock medicine in pharmacy of dispensary, obtain from central stores on need, allow medicine from external pharmacies}. A study of patterns of occurrence of health problems, fast moving/slow moving stock, and existing storage facilities leads to deciding what is to be stocked, what is to be obtained from the central store and what is to be allowed from external pharmacies. Once tactical decisions are made then operational decisions are made about orders for individual medicines and quantities. The information system is then invoked for indents, purchase and deliveries to be posted.

As another example, consider providing specialized advice. The dispensary makes an analysis of occurrence of health problems, ability to service them, patient convenience and other such factors. As a result, it may identify the need for some specialist medical advice not conveniently available in the dispensary. This constitutes an imperative decision of the operational kind. However, since the dispensary does not have the authority to do this, it makes a recommendation to headquarters about the need for the new specialization. Headquarters makes an administrative decision of the managerial kind, identifies the facility, and communicates it to the dispensary which, in turn, updates its information system.

Now let us consider managerial decisions. In the health scheme, these are essentially taken by headquarters. In doing so, requirements of dispensaries are taken into account as brought out in the example above.

A major managerial decision is about empanelment of medical facilities. Headquarters makes periodic calls for applications from hospitals for empanelment. The decision requirements are as shown in Fig. 10. There can be three kinds of hospitals, general (Gen.), super specialized (SS), and AYUSH. The last are those providing Indian systems of medicine (Ayurveda, Yoga, Naturopathy and Unani). For deciding which to empanel, headquarters needs information about available facilities. Thus, we get the DRs as shown in Fig. 10. There is an abstract DR <empanel hospital, facilities> which is specialized into the three DRs for the different kinds of hospitals.

Facilities comprise three aspects, space, equipment, and staff. Thus, <Empanel hospital, facilities> is an aggregate of the three DRs shown in Fig. 11, namely,

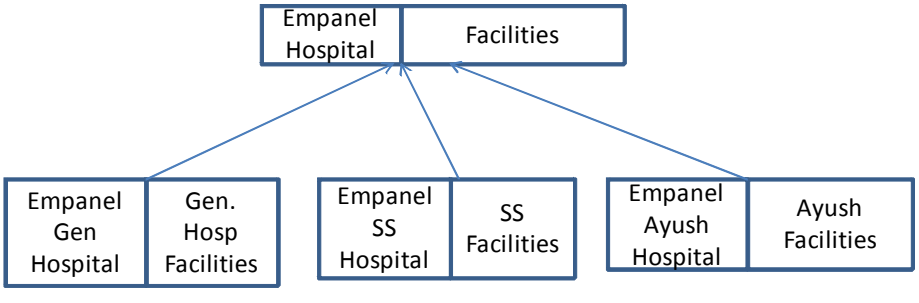

Fig. 10. The abstract DR < hospital, facilities>

<empanel hospital, space>, <empanel hospital, equipment> and <empanel hospital, staff>. Space information consists of sizes of wards, sizes of rooms, pharmacy stores and other Ayush spaces [14] and occupancy rates; equipment is the laboratory equipment, operating theatre equipment, number of beds and their occupancy ratio etc. and their use; staff information consists of doctors, paramedics, nurses and others [14] with their qualifications, the number and nature of cases/patients handled by them and other personal information [14].

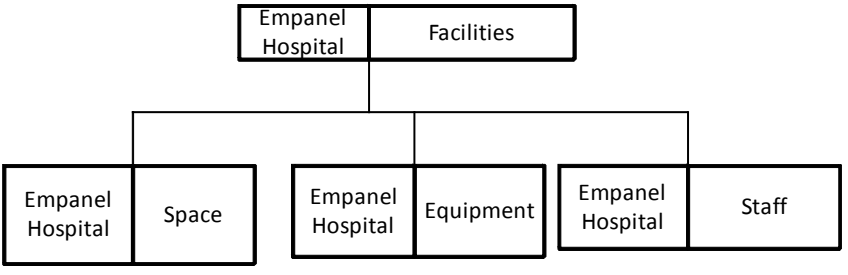

Fig. 11. A complex DR

Now, headquarters defines norms and standards that must be met by all empanelled hospitals. These take the form of specified equipment, minimum requirements, and various norms. We give below examples of these for the Ayush class of hospitals. The full details of the case can be found in [14].

- Snehan Room 100 Sq. ft.
- Sirodhara Yantra & appliances 2 Nos.
- Vasti Yantra
- Droni/Bath tub
- O.P.D. 300 Sq. ft.
- Pharmacy/Store 300 Sq. ft
- Resident Medical Officer 1
- Masseurs/Panchakarma Attendents 4
- Staff Nurses 4 (Round the clock)
- Pharmacist 2
- Bed occupancy rate (Norm 50%)
- Bed:Nurse ratio for general bed 8:1

Whereas administrative decisions are taken after considering the extent to which given norms and standards are met, policy decisions define these. The elicitation of these norms is an unaddressed issue.

5 Related Works

First, let us consider the notion of goal in goal oriented requirements engineering approaches and compare it with our proposal. On the left hand side of Fig 12 we show

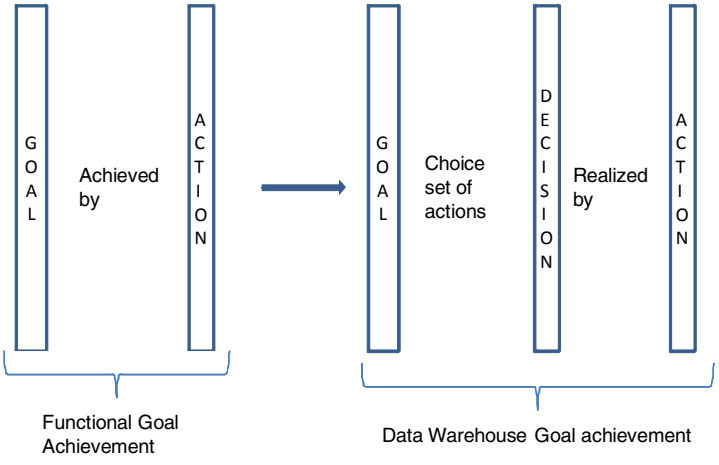

Fig. 12. Functional and Data Warehouse Goal orientation

the former. The notion here is of a goal being achieved by an action, a function. This helps in specifying system functionality. No support is provided to the user in determining which of the many actions is to be performed at a given moment. In our proposals, however, the focus is on determining the most suitable alternative, by examining the state of the organization as reflected in the data warehouse. Once this is done, then the action corresponding to the decision is invoked.

Now consider our approach with respect to database and ER driven approaches compared in Table 1. It can be seen that the decision requirement driven approach does not attempt to directly reach facts and dimensions like the others. Additionally, unlike the data base and ER approaches, the decision requirement approach can identify aggregate and historical information to be maintained. Finally, from a conceptual point of view, there is an explicit notion of a decision and decision requirements modeling which lays the basis for data warehouse development. Evidently, our approach lies upstream of the others.

Now, let us compare our proposals with goal oriented approaches in data warehouse development: Böhnlein and Ulbrich-vom Ende [6,7], Golfarelli and Rizzi [15] and Bonifati et al [16]. The first of these determines measures and dimensions from an initial study of goals and services to be provided by the organization. The second extends Tropos to requirements engineering of Data Warehouses. In the organizational perspective (a) an actor diagram is built (b) goals of actors are identified for example, ‘manage a/c transactions’ and (c) facts are associated to goals. Goals are decomposed using goal decomposition techniques. Thereafter, in the decisional perspective, the actor diagram is analyzed and each fact is related to its dimensions and a set of measures are associated with facts. The last proposal builds data marts by determining goals using the Goal-Question-Metric approach. The result is aggregated and refined and ideal star schemata are extracted. Then it uses ER schemata of operational data bases and extracts candidate star schemata. The ideal star schemata are matched with these and candidate star schemata are ranked according to the metrics for selection.

Table 1. A Comparison

	Database driven	**ER driven**	**Decision Requirement driven**
Facts	Yes, manually	Yes, manually	No
Dimensions	Yes, manually	Yes, algorithmically	No
Star schemas	Yes, manually	Yes, manually	No
Aggregates	Manually. No guidance	Manually. No guidance	Yes as Information property
History	Manually. No guidance	Manually. No guidance	Yes, as information property
Decision Capacity	Not articulated	Not articulated	Yes

Again, notice that [6,7] and [16] reach data warehouse contents directly from goals without an explicit decisional stage. In this respect the approach of [15] goes forward and recognizes the need to do further analysis from the decisional point of view. In our case, this movement has been taken to its logical conclusion. We explicitly model the full decision making capability and associated information requirements. In earlier work, we have shown how information associated with decisions can be elicited [17]. This information is obtained at a high level and is relatively unstructured. It is made more structured through the information scenario technique of [13] which produces contents of the data warehouse To-Be in the form of an ER diagram. Thereafter, the ER diagram is converted into star schemata using available algorithms. Thus, it can be seen that our proposal is to fully investigate the decision-information association before proceeding to development of star schemata.

GRAI grid [18] gives an overview of "decision making procedures and informational flows within a production control system". It provides an architecture of decision making system. Decisional processes occur at the lowest level and are associated with time. GRAI models repetitive and 'period-driven' decisions. In our case, the decision making procedure per se is not interesting. Rather, it is the decision-information association that is of interest because it helps us to define data warehouse contents and thereby makes available information required for decision making. The procedure of decision making and its effect on the organization is outside our scope.

Finally, our decisional environment is similar to the work system proposed in [19]. However, it addresses decision making, not operational information systems

6 Conclusion

The notion of decision making is fundamental to data warehouses. It implies the existence of a choice set from which the best alternative, one that meets organizational

goals, is selected. We have shown that these alternatives can be (a) managerial, for setting up the environment for providing service and (b) imperative, for providing the right service. We have placed emphasis on modeling the set of decisions and associated information in an organization. It is only thereafter that one can proceed to subsequent stages of star schema design.

Future work is centered round elicitation of imperative and managerial decisions.

References

1. Mylopoulos, J., Chung, L., Nixon, B.: Representing and Using Nonfunctional requirements: A Process-Oriented Approach. IEEE Trans. on Software Engineering 18(6), 483–497 (1992)
2. van Lamsweerde, A.: Goal-Oriented Requirements Engineering: A Guided Tour. Invited Paper for RE 2001 - 5th IEEE International Symposium on Requirements Engineering, Toronto, pp. 249–263 (August 2001)
3. Sutcliffe, A., Maiden, N.: Bridging the Requirements Gap: Policies, Goals and Domains. In: Proc. IWSSD-7 - 7th Intl. Workshop on Software Specification and Design. IEEE, Los Alamitos (1993)
4. Yu, E.S.K.: Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering. In: Proc. Third IEEE Symposium on Requirements Engineering, pp. 226–235 (1997)
5. Giunchiglia, F., Mylopoulos, J., Perini, A.: The Tropos Software Development Methodology: Process, Models and Diagrams. Autonomous Agents and Multi-Agent Systems 8(3), 203–236 (2004)
6. Boehnlein, M., Ulbrich vom Ende, A.: Deriving initial Data Warehouse Structures from the Conceptual Data Models of the Underlying Operational Information Systems. In: Proc. Of Workshop on Data Warehousing and OLAP (DOPLAP), USA, pp. 15–21 (1999)
7. Boehnlein, M., Ulbrich vom Ende, A.: Business Process Oriented Development of Data Warehouse Structures. In: Proceedings of Data Warehousing 2000. Physica Verlag, Heidelberg (2000)
8. Rilson, F., Paim, S., Castro, J.F.B.: DWARF: An Approach for Requirements Definition and Management of Data Warehouse Systems. In: Proceeding of the 11th IEEE International Requirements Engineering Conference 1090-9, September 08-12 (2003)
9. Frendi, M., Salinesi, C.: Requirements Engineering for Data Warehousing. In: Proceedings of Workshop on REFSQ, pp. 75–82 (2003)
10. Prakash, N., Gosain, A.: Requirements Driven Data Warehouse Development. In: Eder, J., Missikoff, M. (eds.) CAiSE 2003. LNCS, vol. 2681, pp. 13–17. Springer, Heidelberg (2003)
11. Prakash, N., Singh, Y., Gosain, A.: Informational Scenarios for Data Warehouse Requirements Elicitation. In: Atzeni, P., Chu, W., Lu, H., Zhou, S., Ling, T.-W. (eds.) ER 2004. LNCS, vol. 3288, pp. 205–216. Springer, Heidelberg (2004)
12. Winter, R., Strauch, B.: Information Requirements Engineering for Data Warehouse Systems. In: ACM Symposium on Applied Computing, Cyprus, pp. 1359–1365 (2004)
13. Prakash, N., Gosain, A.: An Approach to Engineering the Requirements of Data Warehouses. Requirements Engineering Journal 13(1), 49–72 (2008)
14. Government of India, Department of Ayush, Ministry of Health and Family Welfare, No.Z.20018/4/2000-ISM (Tech)/HD Cell, National Competitive Bidding

15. Golfarelli, M., Rizzi, S.: Designing the Data Warehouse: Key Steps and Crucial Issues. Journal of Computer Science and Information Management 2(3) (1999)
16. Bonifati, A., Cattaneo, F., Ceri, S., Fuggetta, A., Paraboschi, S.: Designing Data Marts for Data Warehouses. ACM Trans. Softw. Eng. Methodol. 10(4), 452–483 (2001)
17. Prakash, N., Prakash, D., Sharma, Y.K.: Towards Better Fitting Data Warehouse Systems. In: Persson, A., Stirna, J. (eds.) PoEM 2009. LNBIP, vol. 39, pp. 130–144 (2009)
18. Carrie, A.S., Macintosh, R.: An Assessment of GRAI Grids and their use in the Strathclyde Integration Method, production Planning and Control. vol. 8(2), pp: 106–113 (1997)
19. Alter, S.: A General yet Useful Theory of Information Systems. Comm. of Association for Information Systems 1(13), 1–68 (1999)
20. Inmon, W.H.: Building the Data Warehouse. John Wiley and Sons, Chichester (1996)

A Tool for Enterprise Architecture Analysis Using the PRM Formalism

Markus Buschle, Johan Ullberg, Ulrik Franke,
Robert Lagerström, and Teodor Sommestad

Industrial Information and Control Systems, KTH Royal Institute of Technology,
Osquldas v. 12, SE-10044 Stockholm, Sweden
{markusb,johanu,ulrikf,robertl,teodors}@ics.kth.se

Abstract. Enterprise architecture advocates for model-based decision-making on enterprise-wide information system issues. In order to provide decision-making support, enterprise architecture models should not only be descriptive but also enable analysis. This paper presents a software tool, currently under development, for the evaluation of enterprise architecture models. In particular, the paper focuses on how to encode scientific theories so that they can be used for model-based analysis and reasoning under uncertainty. The tool architecture is described, and a case study shows how the tool supports the process of enterprise architecture analysis.

Keywords: Enterprise Architecture, Probabilistic relational Models, Software tool, Security Analysis.

1 Introduction

Over the last two decades, enterprise architecture has grown into an established approach for holistic management of information systems in organizations [1,2]. A number of enterprise architecture initiatives have been proposed, such as The Open Group Architecture Framework (TOGAF) [3], the Zachman Framework [4], and military architectural frameworks such as DoDAF [5] and NAF [6]. The core concept of the enterprise architecture approach is the employment of models, in terms of diagrammatic descriptions of information systems and their environment. Diagrammatic descriptions of IT systems and their environment are heavily used. However, enterprise architecture models are not limited to descriptive use only, but can also be employed to predict the behavior and effects of decisions. Rather than modifying enterprise information systems using trial and error, models allow predictions about the behavior of future architectures.

One prominent challenge to rational decision making is uncertainty. Therefore, a good enterprise architecture model should be able to capture uncertainties about assessment theory, system configuration or data quality, thus providing better decision support and risk management.

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 108–121, 2010.
© Springer-Verlag Berlin Heidelberg 2010

What constitutes a "good" enterprise architecture model is dependent on its purpose, i.e. the type of analysis it is intended to support [7]. For instance in the case of analyzing cyber security, the property of whether it is possible to reconfigure a firewall is of interest. This property however, is irrelevant for a number of other analyses, such as performance evaluation or data quality analysis.

Several enterprise architecture software tools are available on the market, including Metis [8], System Architect [9] and Aris [10]. These tools generally focus on the modeling of an architecture whereas the analysis functionality is generally limited to performing an inventory or to sum costs over the modeled architecture. None of the mentioned tools has significant capabilities for system quality analysis based on an elaborated theory. Furthermore, these tools do not support the consideration of uncertainty as described above.

In this paper an enterprise architecture software tool is presented. This tool does not only provide functionality to model enterprise architectures, but also supports the analysis of them. In order to support enterprise architecture analysis as it has been outlined in [7] the tool consists of two main components. In the first component the theory relevant to analyze a certain system quality, such as data quality or modifiability, is modeled. One can consider this as the definition of a language tailored to describe a certain aspect, e.g. cyber security. The second component supports the application of the theory to evaluate a specific enterprise architecture. This is done by modeling the "as-is" or "to-be" architecture of the enterprise. Based on the created models it is possible to determine how the architecture fulfills the requirements as they have been defined in the theory. The two-component architecture encourages the reuse of the developed theory as it is possible to use the same language to describe several architecture instances. The presented tool makes use of the Probabilistic Relation Models (PRM) formalism as it has been presented in [11] and can thereby manage the uncertainty aspects discussed above.

2 Enterprise Architecture Analysis

Enterprise architecture models serve several purposes. Kurpjuweit and Winter [12] identify three distinct modeling purposes with regard to information systems, viz. (i) documentation and communication, (ii) analysis and explanation and (iii) design. The present article focuses on the analysis and explanation (which is not to denigrate the usefulness of the others). The reason is that analysis and explanation are closely related to the notion of proper *goals* for enterprise architecture efforts. For example, a business goal of decreasing downtime costs immediately leads to an analysis interest in availability. This, in turn, defines the modeling needs, e.g. the need to collect data on mean times to failure and repair. In this sense, analysis is at the core of making rational decisions about information systems [7] [13]. An analysis-centric process of enterprise architecture is illustrated in Fig. 1. In the first step, assessment scoping, the problem is described in terms of one or a set of potential future scenarios of the enterprise

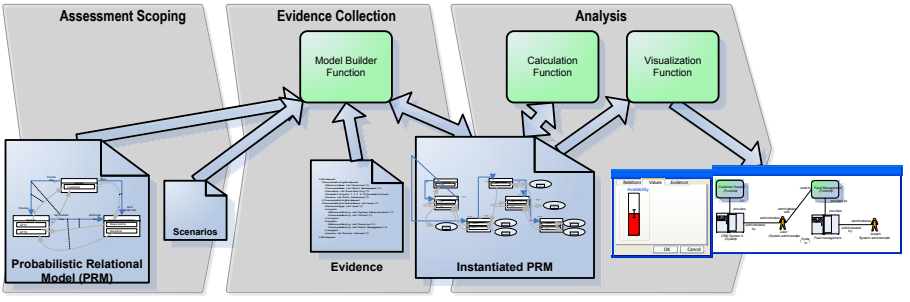

Fig. 1. The process of enterprise architecture analysis with three main activities: (i) setting the goal, (ii) collecting evidence and (iii) performing the analysis

and in terms of the assessment criteria with its theory (the PRM in the figure) to be used for scenario evaluation. In the second step, the scenarios are detailed by a process of evidence collection, resulting in a model (instantiated PRM, in the figure) for each scenario. In the final step, analysis, quantitative values of the models' quality attributes are calculated and the results are then visualized in the form of e.g. enterprise architecture diagrams.

More concretely, assume that a decision maker in an electric utility is contemplating changes related to the configuration of a substation. The modification of a new access control policy would reduce the probability that someone installs malware on a system and thereby reduce the risk that this type of unwanted software is executed. The question for the decision maker is whether this change is feasible or not.

As mentioned in the first step *assessment scoping* the decision maker identifies the available decision alternatives, i.e. the enterprise information system scenarios. In this step, the decision maker also needs to determine how the scenario should be evaluated, i.e. the goal of the assessment. One such goal could be to assess the security [14] of an information system. Other goals could be to assess the availability [15], interoperability [16] or data quality [17] [18] of the proposed to-be architecture. Often several quality attributes are desirable goals. In this paper, without loss of generality, we simplify the problem to the assessment of security of an electric powerstation.

Information about the involved systems and their organizational context is required for a good understanding of their data quality. For instance, it is reasonable to believe that a firewall would increase the probability that the system is secure. The availability of the firewall is thus one factor that can affect the security and should therefore be recorded in the scenario model. The decision maker needs to understand what information to gather and also ensure that this information is indeed collected and modeled. Overall, the effort aims to understand which attributes causally influence the selected goal, viz. data quality. It might happen that the attributes identified do not directly influence the goal. If so, an iterative approach can be employed to identify further attributes causally affecting the attributes found in the previous iteration. This iterative process

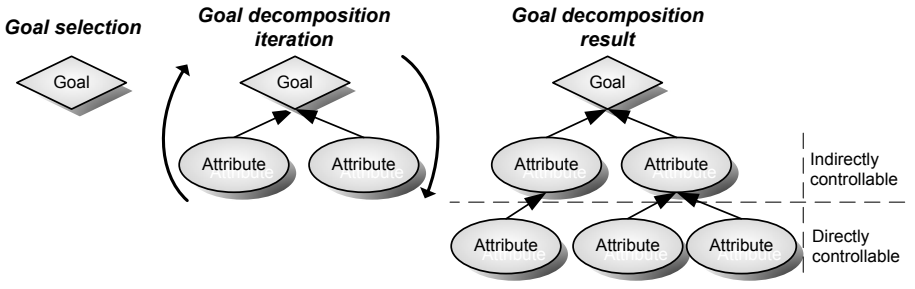

Fig. 2. Goal decomposition method, from [19]

continues until all paths of attributes and causal relations between them, have been broken down into attributes that are directly controllable for the decision maker [19] (cf. Fig. 2).

In the second step *collecting evidence* the scenarios need to be detailed with actual information to facilitate their analysis of them. Thus, once the appropriate attributes have been set, the corresponding data is collected throughout the organization. In particular, it should be noted here that the collected data will not be perfect. Rather, it risks being incomplete and uncertain. The tool handles this by allowing the user to enter the credibility of the evidence depending on how large the deviations from the true value are judged to be.

In the third and final step, *performing the analysis*, the decision alternatives are analyzed with respect to the goal set e.g. security. The mathematical formalism to be presented in section 2.1 plays a vital role in this analysis. Using conditional probabilities and Bayes' rule, it is possible to infer the values of the variables in the goal decomposition under different architecture scenarios [20]. By using the PRM formalism, the architecture analysis accounts for two kinds of potential uncertainties: that of the attribute values as well as that of the causal relations as such. Using this analysis framework, the pros and cons of the scenarios can be weighted against each other in order to determine which alternative ought to be preferred.

2.1 Probabilistic Relational Models

A *probabilistic relational model* (PRM) [21] specifies a template for a probability distribution over an architecture model. The template describes the metamodel for the architecture model, and the probabilistic dependencies between attributes of the architecture objects. A PRM, together with an instantiated architecture model of specific objects and relations, defines a probability distribution over the attributes of the objects. The probability distribution can be used to infer the values of unknown attributes, given evidence of the values of a set of known attributes. PRMs are related to Bayesian Networks. As it is succinctly put in [11], PRMs "are to Bayesian networks as relational logic is to propositional logic".

A PRM model may be instantiated as a *relational skeleton*, σ_r, containing objects, object relationships, and attributes. Furthermore, a *qualitative dependency*

structure $\mathcal{S}$ defines the details of the attribute relationships, i.e. the sets of probabilistic parents influencing each attribute. Finally, the PRM is completed by the set of *parameters* $\theta_{\mathcal{S}}$ specifying the full conditional probabilistic dependencies between attributes in the form of numbers in Conditional Probability Matrices (CPM). The following expression thus defines the conditional probability of an instance $\mathcal{I}$, given σ_r, $\mathcal{S}$, and $\theta_{\mathcal{S}}$:

$$P(\mathcal{I}|\sigma_r, \mathcal{S}, \theta_{\mathcal{S}}) = \prod_{x \in \sigma_r} \prod_{A \in \mathcal{A}(x)} P(\mathcal{I}_{x.A}|\mathcal{I}_{Pa(x.A)})$$

$$= \prod_{X_i} \prod_{A \in \mathcal{A}(X_i)} \prod_{x \in \sigma_r(X_i)} P(\mathcal{I}_{x.A}|\mathcal{I}_{Pa(x.A)})$$

Compared to the standard chain rule for Bayesian networks, this equation is different in three ways: (i) the random variables are the attributes of a set of objects, (ii) the parents of a random variable depend on the model context of the object, and (iii) the parameters are shared between the attributes of objects in the same class. In other words, the variables in the dependency structure are the properties of the objects in the instantiated information model, and their causal relations are expressed by the CPM [11].

A PRM thus constitutes a formal machinery for calculating the probabilities of various architecture instantiations. This allows us to infer the probability that a certain attribute assumes a specific value, given some (possibly incomplete) evidence of the rest of the architecture instantiation. In addition to expressing and inferring uncertainty about attribute values as specified above, PRMs also provide support for specifying uncertainty about the structure of the instantiations.

PRMs additionally allow specializing classes through inheritance relationships. Classes can be related to each other using the subclass relation $\prec$, and each class X is associated with a finite set of subclasses C[X]. So if $Z, Y \in C[X]$, both Z and Y are subclasses of X. If $Z \prec Y$ then Z is a subclass of Y, and vice versa Y is a superclass of Z. A subclass Z always contains all dependencies and attributes of its superclass Y. PRMs also allow the dependencies and conditional probability distributions of inherited attributes to be specialized in subclasses.

3 Architecture of the Tool

The presented tool is implemented in Java and is based on a Model-View-Controller architecture [22] [23]. Where the model represents the knowledge and considered data that in this case are PRM and instantiated PRM respectively. The implementation contains a mapping of the PRM structure to JAVA-classes. Thereby each part of a PRM (e.g. PRM-classes, their attributes, and their relations) can be used and combined as regular JAVA-objects. The view is in charge of the visualization of the model for the tool user. It presents the considered data in an understandable way. The view is updated as soon as the model changes. Finally the controller links the user to the application making the program react on the users input. Changes of the model are perforemd via the controller that acts as an interface in this case. Thereby a decoupling of data access, program logic, data presentation, and user interaction can be ensured. This facilitates the

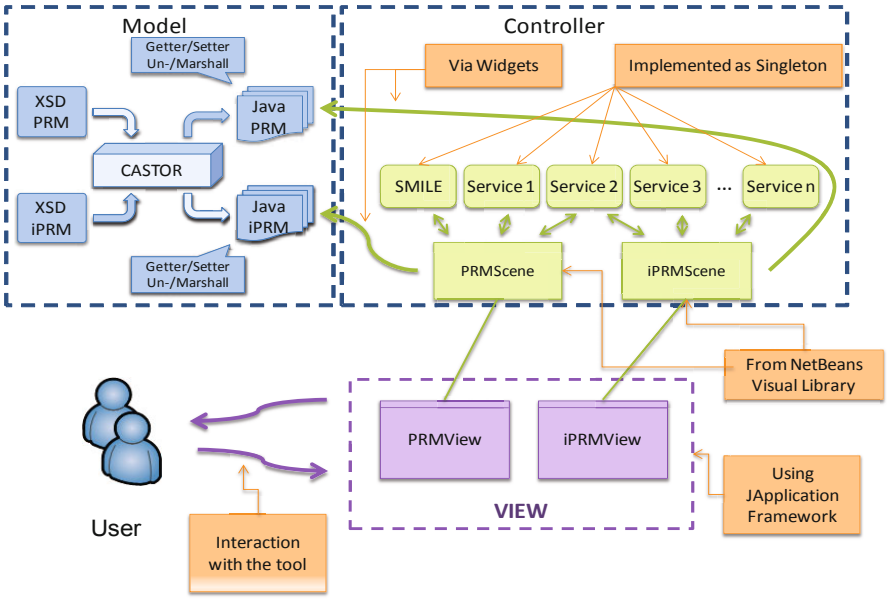

Fig. 3. High level architecture of the tool illustrating the main components. “iPRM" is shorthand notation for instantiated PRM.

extension of the implementation and eliminates potential error sources as the program code is structured and functionality grouped according to its purpose.

The data model for PRMs and instantiated PRMs, is specified in XSD and stored in XML files [24]. This is done through an application of the Castor library [25]. As XML is a widespread format created models can be imported into other applications and data does not need to be captured a second time. The user interface is built upon the NetBeans Visual Library [26] with usage of the JApplication framework. This library provides a set of reusable components, called widgets, and can be applied to create visualization. The widgets can be aggregated and related to each other thereby reflecting the creation of models intuitively. These models are drawn on a special canvas that in the NetBeans terminology is called scene. Besides the modeling capabilities the user interface provides the one applying the tool with support functionalities such as filtering, tagging, and exporting. These well-defined tasks are performed by corresponding tailored services that are implemented following the singleton pattern to ensure data consistency. The architecture described is depicted in Fig. 3, whereas how the user interface (and thereby the actions performed by the user) is linked to the architecture us show in Fig. 4.

The tool is separated into two units, one supporting the modeling of the PRM, the other one makes the tool user able to instantiate and analyze this defined structure. These parts have to be used sequentially, reflecting the method that has been described in section 2, starting with the modeling of classes and their attributes as well as the relationships and dependencies between them. Thereby

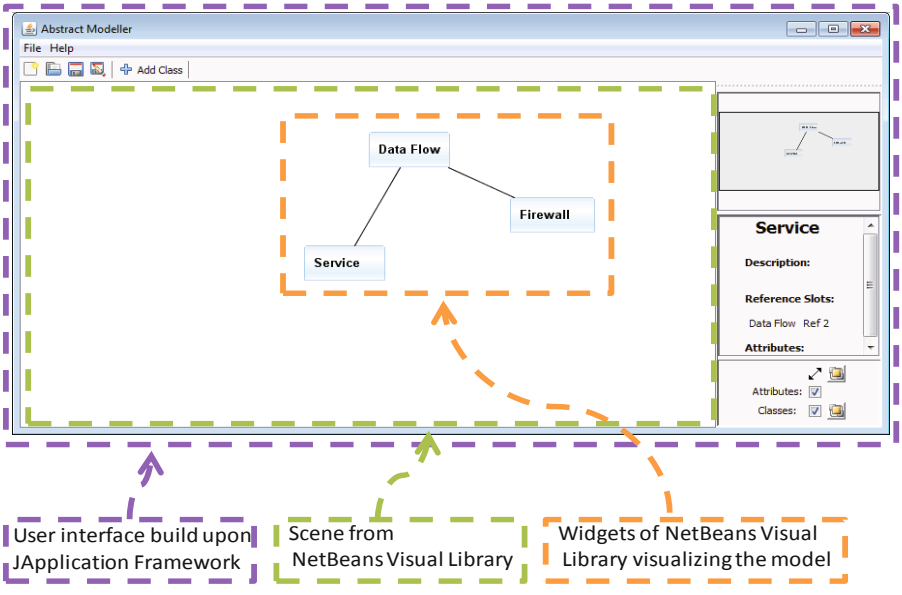

Fig. 4. User interface of the tool illustrating how the user interacts with the tool. Colors of boxes match with concepts described in Fig. 3.

the focus of the analysis is set, as the defined classes reflect the domain of interest. The second component of the Enterprise Architecture Analysis Tool (EAT) allows the instantiation of the PRM. Thereby one or several scenarios of interests are modeled according to constraints defined in the dependency structure for the PRM. Afterwards the analysis is performed. Therefore the instantiated PRM is translated into a Bayesian network that is understandable by the Smile library [27]. This library performs the evaluation of the network. Finally the calculated values are written back to the instantiated PRM and visualized for the tool user.

The person applying the tool can then compare the modeled scenarios and their contained classes by considering the probabilities that attributes of the classes are in a certain state; thereby the identification of the configuration that qualifies best is made possible.

4 Example Tool Application

This section will illustrate how the tool can be applied in practice. The metamodel and instance model presented here are drawn from a case study performed at a Swedish power utility company in November 2009. In this case study the cyber security of an electric substation was the concern. The metamodel is thus intended to support cyber security analysis and the instance model represents one of the utility's substations. The qualitative structure of the metamodel (or PRM) was created based on a literature review; the quantitative part was in this

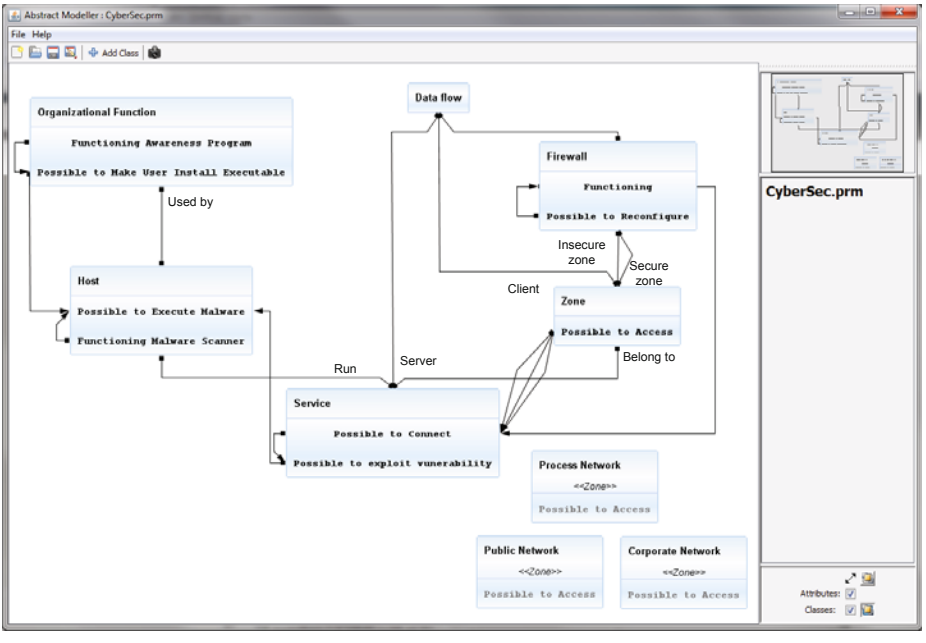

Fig. 5. The PRM for cyber security analysis showing classes and attributes relevant for the analysis

case assessed (subjectively) by a security researcher. Interviews with a system administrator and investigations of system documentation were used to create the instance model.

4.1 Probabilistic Relational Model over Security

The PRM used in this case study is depicted in Fig. 5. The PRM covers a number of concepts that are of relevance to the cyber security computer networks including firewalls, data flows, software services network zones and organizational functions. The qualitative structure of this PRM is described below together with some examples of conditional probabilities defined in the PRM.

The primary purpose of firewalls is to control access to network addresses. They do so by blocking unwanted data flows from adjacent zones, and by allowing those that are wanted. With a protection scheme following the principle 'deny by default', a *Firewall* will allow a number of *DataFlows* to pass into the secure *Zone* from other *Zones*. In this ConcretePRM a *Firewall* holds the reference slots *Allow* with range *DataFlow* which point to data flows that are allowed. A Firewall also has the reference slots *SecureZone* and *InsecureZone* with range *Zone* which refers to the zones that are directly separated by the firewall.

The *Firewall* has the attribute *PossibleToReconfigure* which indicates if it possible for a threat agent to reconfigure the firewall or not. This attribute influences the attribute *Firewall.Functioning* which indicates whether the firewall functions

as it should. If the firewall is working it will prevent unauthorized connections from insecure zones to secure zones. The attribute *Service.PossibleToConnect* states whether the threat agent can connect to a *Service*. The threat agent can also connect to the service if it has access to the service's zone or if data flows are allowed from a zone where the threat agent have access. If the threat agent has access or not is expressed through the attribute *Zone.PossibleToAccess* which has a different value in the subclasses *PublicNetwork*, *CorportateNetwork* and *ProcessNetwork*.

If it is possible to connect to the service (i.e. Service.PossibleToConnect=True) it might be possible to exploit a vulnerability in the service. The attribute *Service.PossibleToExploitVulnerability* expresses whether this is possible or not. The reference slot *Service.Host* points to the *Host* that executes the service. The possibility to exploit vulnerabilities in the service influences if it possible to execute malware on the service's host. The possibility to execute malware on the host is also influenced by the existence of a functioning malware scanner in the host. The attribute *Host.FunctioningMalwareScanner* indicates whether this is the case or not. Another way to influence the possibility to execute malware is through users in the *Organizational Function* that use the host and that install executables, i.e. the attribute *OrganizationalFunction.MakeUserInstallExecutable*.

4.2 Instance Model

The classes and reference slots in the PRM were used to model one of the utility's substations (cf. Fig. 6 for a screenshot of the tool). Four network zones where found in this study.

The *OfficeNetwork* is the insecure side of the *CorporateFirewall*. The *CorporateFirewall* allows two data flows: *RemoteDesktop* to pass through from the *OfficeNetwork* to the service *TerminalServices* and the data flow *SubstationCommunication* from the zone *ControlCenterNetwork* to *ControlSystemServer*. The *GatewayFirewall* is connected to the Internet and allows data to pass from both the *OfficeNetwork* and the *SuppliersLAN*.

Within the substation there are two instances of the *ProcessNetwork*. The *SubstationLAN* is where the services *ControlSystemServer* and *TerminalServices* belong; the service *VNCInterface* belongs to the *ModemLAN*. The *ControlSystemServer* is within the host *StationController*. The *ControlSystemServer* is also the server side of the data flow *SubstationCommunication*.

The service *TerminalServices* is associated with the host *ServiceGateway* and acts as the server side of the data flow *RemoteDesktop*. The *ServiceGateway* is known to have a malware scanner that is functioning and evidence to support this fact is stored within the model. The service *VNCInterface* belongs to the *ModemLAN* and is executed by the host *EmbeddedController*.

Three organizational functions use hosts within the substation: *Supplier*, *FieldEngineers* and *SubstationEngineers*. The *EmbeddedController* is used by two organizational functions: *Supplier* and *FieldEngineers*. The *ServiceGateway* is used solely by the *SubstationEngineers* and the *StationController* is used by both *SubstationEngineers* and *FieldEngineers*. None of the organizational functions

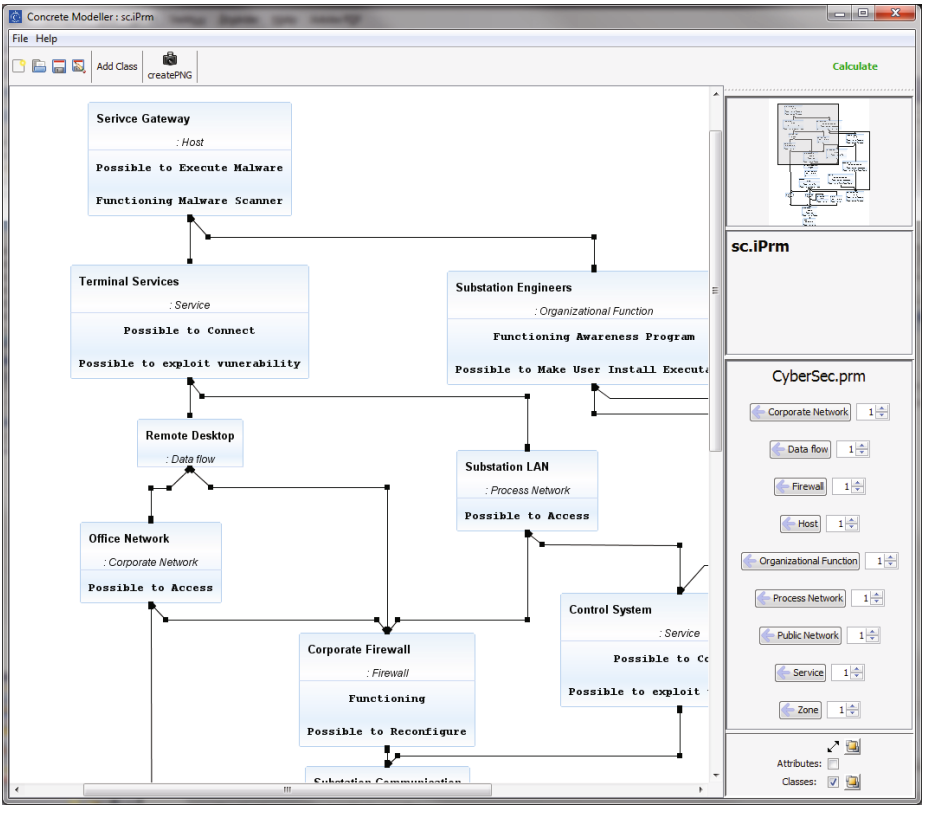

Fig. 6. Screen shot illustrating part of the instantiated PRM for cyber security analysis. For readability only the classes and reference slots are shown in the picture, attribute relationships are hidden.

are covered by a functioning awareness program, i.e. *FunctioningAwarnessProgram=false* for all instances of *OrganizationalFunction*. The complete PRM is shown in Fig. 7

4.3 Scenario Analysis

The instance model shown in Fig. 6 represents the architecture that existed at the time of the assessment. With this architecture as a starting point, different alternative scenarios were assessed.

One such scenario was to introduce an awareness program for substation engineers, i.e. to change *SubstationEngineers.FunctioningAwarnessProgram* to *True*. The impact of this is calculated to change *SubstationEngineers.MakeUserInstall Executable* from 20 % to 10 %, which in turn influences the *PossibleToExecuteMalware*-attribute in the hosts *StationController*, *EmbeddedController* and *ServiceGateway*.

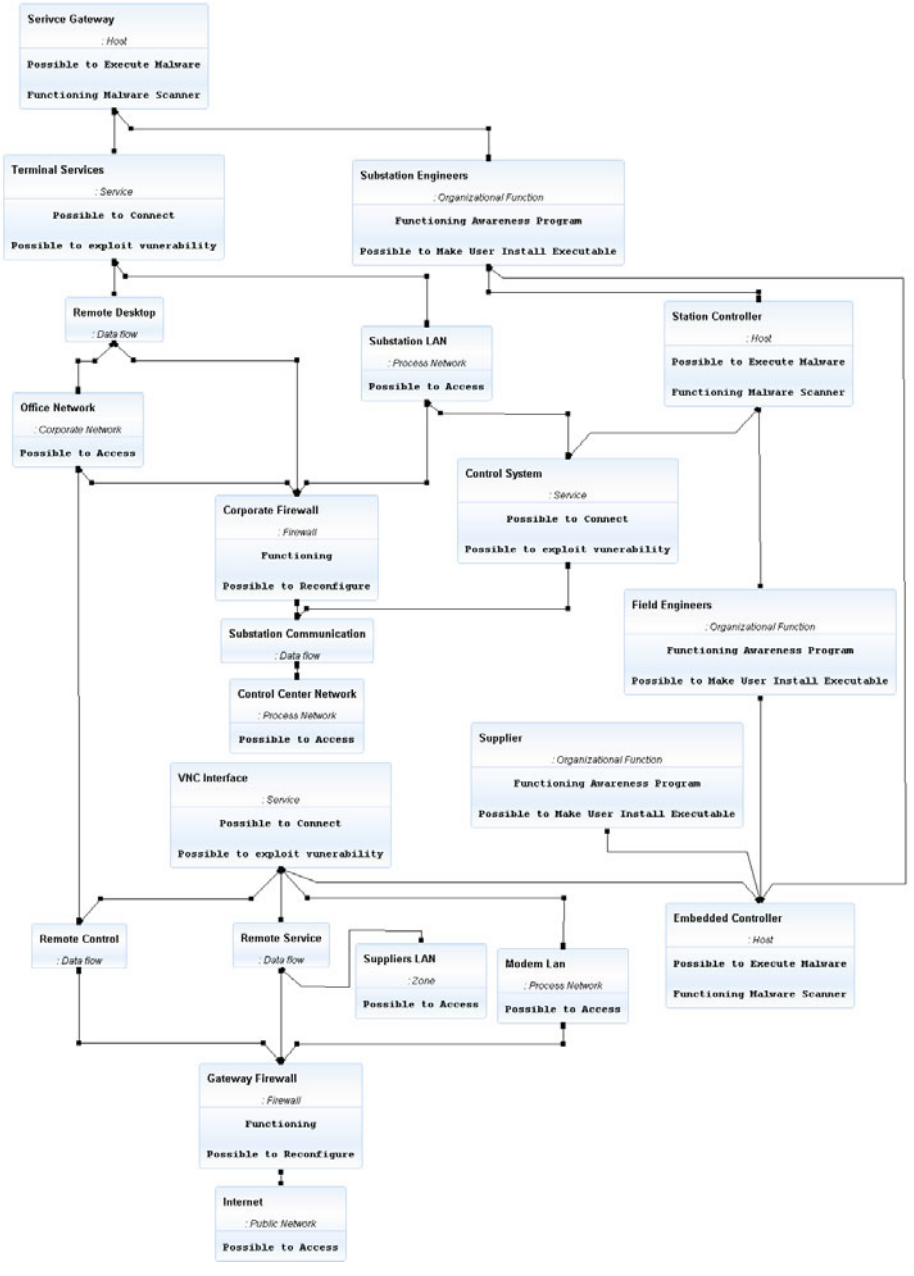

Fig. 7. Instantiated PRM for cyber security analysis. For readability only the classes and reference slots are shown in the picture, attribute relationships are hidden.

Another scenario that was investigated was the impact of a new access control policy. This modification would ensure that *SubstationEngineers* do not use the *EmbeddedController*. A change like this would remove the dependency relationship between *SubstationEngineers.MakeUserInstallExecutable* and *EmbeddedController.PossibleToExecuteMalware*. For all cases other than *SubstationEngineers.MakeUserInstallExecutable=False* this would reduce the probability of *EmbeddedController.PossibleToExecuteMalware* being *true*. Making the change in the tool with the present PRM changes *P(EmbeddedController. PossibleToExecuteMalware)* from 50 % to 37 %.

5 Discussion and Future Works

This paper presents a tool which supports enterprise architecture analysis with the use of the PRM formalism. While providing a powerful mechanism for the use of discrete variables in an analysis, the PRM formalism in its initial form has a few weaknesses that deserve further studies. Several system qualities are typically analyzed through the usage of continuous variables e.g. in [28] continuous variables are used for performance analysis. In order to perform those evaluations with support of the presented tool it is necessary to discretize all continuous variables. At the moment we are investigating how the PRM formalism can be extended so that it can be used with combinations of continuous and discrete variables, so called hybrid networks [29], as well as a corresponding tool implementation.

Another weakness of the PRM formalism is that it does not provide any means to query the models for structural information such as "given an information system, how many elements does the set of related data objects contain?" The Object Constraint Language (OCL) [30] is a formal language developed to describe constraints on UML models. OCL provides a means to specify such constraints and perform queries on the models in a formal language. OCL in its original form is side effect free, but currently an imperative version of OCL is being added to the tool. Thereby the analysis functionality can be extended to consider the structure of the PRM instantiation more comprehensively.

Besides the two mentioned shortcomings of the formalism used there are some improvements with respect to usability. Regarding the user interface of the tool, we are planning to make the models more intuitive and the information provided easier to understand. Enterprise architecture models are more understandable if they only depict the interesting parts of the model (in a goal-sense). Therefore, the tool should be extended to support views and viewpoints, e.g. as presented in [28]. Additionally we plan the support of iconic visualization of typical enterprise architecture elements, such as applications or data objects, to present the models in an easily understandable way. Finally we are planning to integrate the support of predefined model components. As models based on the same metamodel are likely to have common parts, the modeling process can be sped up if common building blocks are offered by the metamodel provider and used by the person that creates a certain model.

6 Conclusion

In this paper a tool and method for analysis of enterprise architecture scenarios was presented. To fulfill this purpose the tool consists of two separate parts, one for defining analysis theory and another for enterprise architecture modeling, and makes use of the PRM formalism for specifying theory. Applying this formalism allows for the consideration of uncertainty, an aspect that so far is uncommon in the field of enterprise architecture analysis. The paper describes the PRM formalism as well as the underlying architecture of the tool briefly.

In the paper an example of security assessment was outlined, but the tool supports the analysis of various quality attributes such as maintainability, data quality, and interoperability. The tool supports information system decision making as it allows the comparison of several scenarios with regard to a system quality. Thereby the "as-is" as well as several "to-be" architecture of an enterprise can be compared quantitatively in order to find the one that best satisfies decision maker requirements.

References

1. Ross, J.W., Weill, P., Robertson, D.: Enterprise Architecture As Strategy: Creating a Foundation for Business Execution. Harvard Business School Press, Boston (August 2006)
2. Winter, R., Fischer, R.: Essential layers, artifacts, and dependencies of enterprise architecture. Journal of Enterprise Architecture 3(2), 7–18 (2007)
3. The Open Group: TOGAF 2007 edition. Van Haren Publishing, Zaltbommel, Netherlands (2008)
4. Zachman, J.A.: A framework for information systems architecture. IBM Syst. J. 26(3), 276–292 (1987)
5. Department of Defense Architecture Framework Working Group: DoD Architecture Framework, version 1.5. Technical report, Department of Defense, USA (2007)
6. NAF: NATO C3 Technical Architecture (2005)
7. Johnson, P., Ekstedt, M.: Enterprise Architecture – Models and Analyses for Information Systems Decision Making, Studentlitteratur, Sweden (2007)
8. Troux Technologies: Metis (March 2010), `http://www.troux.com/products/`
9. IBM: System Architect (March 2010), `http://www-01.ibm.com/software/awdtools/systemarchitect/productline/`
10. Scheer, A.: Business process engineering: Reference models for industrial enterprises. Springer, New York (1994)
11. Getoor, L., Friedman, N., Koller, D., Pfeffer, A., Taskar, B.: Probabilistic relational models. In: Getoor, L., Taskar, B. (eds.) An Introduction to Statistical Relational Learning. MIT Press, Cambridge (2007)
12. Kurpjuweit, S., Winter, R.: Viewpoint-based meta model engineering. In: Enterprise Modelling and Information Systems Architectures, EMISA 2007 (2007)
13. Iacob, M., Jonkers, H.: Quantitative analysis of enterprise architectures. Interoperability of Enterprise Software and Applications, 239–252 (2006)
14. Sommestad, T., Ekstedt, M., Johnson, P.: A probabilistic relational model for security risk analysis. Computers & Security (February 2010) (accepted)

15. Franke, U., Johnson, P., König, J., Marcks von Würtemberg, L.: Availability of enterprise IT systems – an expert-based bayesian model. In: Proc. Fourth International Workshop on Software Quality and Maintainability (WSQM 2010), Madrid (March 2010)
16. Ullberg, J., Lagerström, R., Johnson, P.: A framework for service interoperability analysis using enterprise architecture models. In: IEEE International Conference on Services Computing (July 2008)
17. Redman, T.: Data quality for the information age. Artech House, Inc., Norwood (1997)
18. Redman, T.: Data quality: the field guide. Digital Pr. (2001)
19. Lagerström, R., Saat, J., Franke, U., Aier, S., Ekstedt, M.: Enterprise meta modeling methods – combining a stakeholder-oriented and a causality-based approach. In: Enterprise, Business-Process and Information Systems Modeling. LNBIP, vol. 29, pp. 381–393. Springer, Heidelberg (2009)
20. Jensen, F.V.: Bayesian Networks and Decision Graphs. Springer, New York (2001)
21. Friedman, N., Getoor, L., Koller, D., Pfeffer, A.: Learning probabilistic relational models. In: Proc. of the 16th International Joint Conference on Artificial Intelligence, pp. 1300–1309. Morgan Kaufmann, San Francisco (1999)
22. Reenskaug, T.: Models-views-controllers. Technical note, Xerox PARC (1979)
23. Sun Microsystems: Design Pattern: Model-View-Controller (2002), `http://java.sun.com/blueprints/patterns/MVC.html`
24. Bray, T., Paoli, J., Sperberg-McQueen, C., Maler, E., Yergeau, F.: Extensible markup language (XML) 1.0. W3C recommendation 6 (2000)
25. ExoLab Group: The Castor Project (March 2010), `http://www.castor.org/`
26. NetBeans: NetBeans Visual Library (March 2010), `http://graph.netbeans.org`
27. Decision Systems Laboratory of the University of Pittsburgh: SMILE (March 2010), `http://genie.sis.pitt.edu/`
28. Lankhorst, M.: Enterprise architecture at work: modelling, communication, and analysis. Springer, Heidelberg (2005)
29. Lauritzen, S.: Propagation of probabilities, means, and variances in mixed graphical association models. Journal of the American Statistical Association 87(420), 1098–1108 (1992)
30. Object Management Group: Object Constraint Language specification, version 2.0 formal/06-05-01. Technical report (2006)

Programming Electronic Institutions with Utopia

Pierre Schmitt, Cédric Bonhomme, Jocelyn Aubert, and Benjamin Gâteau

Centre de Recherche Public Henri Tudor
Service Science and Innovation Dpt.
Luxembourg, G.D. of Luxembourg
firstname.lastname@tudor.lu
http://www.tudor.lu

Abstract. In Multi-Agent Systems, Organizations are means to structure cooperation and collaboration between agents. MoiseInst is a normative Organization model giving the possibility to constraint agents behaviour according to four dimensions (structural, functional, contextual and normative). Mabeli as Electronic Institution model allows the supervision of MoiseInst Organizations compliance through an arbitration system. The difficulty is to easily instantiate such Organizations to obtain a dynamic entity in which agents can evolve. In this paper we introduce Utopia, our Institution-oriented and Institution-based programming framework. Utopia permits to easily and automatically set up a MAS thanks to a XML MoiseInst Specification file. The framework convert this file into an innovative mathematical structure namely a recursive graph, and solve several optimization problems in order to compute the most efficient role distribution. We show a concrete application of the prototype through RED, an EUREKA/CELTIC European project use-case.

Keywords: Normative Organisation, Electronic Institution, Multi-Agent System, Recursive Graphe.

1 Introduction

In human societies, Institutions define rules [1] that enclose all kinds of formal or informal constraints used by human beings to interact. In Multi-Agent System domain, Electronic Institutions have been introduced to model rules with normative systems [2]. That is why we define Electronic Institutions as a set of agents which behave according to Norms and by taking into account their possible violation (and sanction).

These last years Electronic Institution platforms have been improved thanks to new services making them able to express cooperation schemes defined by the user with an Organization Modelling Language such as for instance $\mathcal{M}$oise$^+$ [3], Islander [4], OMNI [5]. The aim of these services is to constraint and supervise agent's actions and interactions in order for them to achieve some global Goals. We call those explicit cooperation schemes *Orgazination Specification* (OS).

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 122–135, 2010.
© Springer-Verlag Berlin Heidelberg 2010

The model used to specify the organization of an Electronic Institution is $\mathcal{M}$OISE^{Inst} [6]. In this context, the functioning of the agents is supervised and controled with a set of *Institution services* regrouped in a specific "normative middleware" called $\mathcal{S}$YNAI on which the agents execute themselves.

This paper aims at presenting how it is possible to easily implement an Electronic Institution specified with $\mathcal{M}$OISE^{Inst}, supervised with $\mathcal{S}$YNAI and in which standard agents provided with the platform evolve and achieve their Goals. For that, three steps have been needed:

1. Define the structure of data in which the OS will be stored.
2. Develop a set of agents working in and able to supervise an Organisation Entity (OE) instantiating the OS defined by an user.
3. Develop a template of JADE based agents able to evolve in the OE (i.e. able to play Roles and achieve Goals) by loading specific behaviours provided by the user in order to execute actions achieving the Goals defined in the OS.

The paper is built as follows: in Section 2 we present rapidly $\mathcal{M}$OISE^{Inst} and $\mathcal{S}$YNAI multi-agents models and in Section 3 we present the recursive graphe mathematical structure on which the implementation of agent models will be done. Those models compose the foundations of our work. Section 4 deals with the implementation of the framework (named $\mathcal{U}$TOPIA) allowing the implementation of Electronic Institution with help of a set of manager agents. At last, before conclude, the Section 5 illustrates the use of $\mathcal{U}$TOPIA through an application of security policies deployment developped in the context of European RED project.

2 Normative Organization Modelling

$\mathcal{M}$OISE^{Inst} [6] is founded on the $\mathcal{M}$OISE^{+} organizational model [3]. It is composed of the following components that are used to specify an Organisation of agents in terms of structure, functioning, evolution and Norms (see Figure 1):

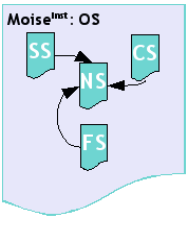

Fig. 1. $\mathcal{M}$OISE^{Inst}, a normative Organization Specification model

- A *Structural Specification* (SS) defines: (i) the Roles that agents will play in the Organization, (ii) the *relations* between these Roles in terms of authority, communication or accointance, (iii) the *Groups*, additional structural primitives used to define and organize sets of Roles;
- A *Functional Specification* (FS) defines global *business processes* that can be executed by the different agents participating to the Organization according to their Roles and Groups;
- A *Contextual Specification* (CS) specifies, *a priori*, the possible evolution of the Organization in terms of a *state/transition graph*;
- A *Normative Specification* (NS) defines the deontic relations gluing the three independant Specification (SS, FS, CS). This NS clearly states rights and duties of each Roles/Groups of SS on sets of Goals (Missions) of FS, within specific states of CS.

These four Specifications form the Organizational Specification (OS). The Organizational Entity (OE) is then built by instantiating the OS through the Agent playing roles, achieving goals and respecting active norms in valid contexts. The $\mathcal{S}$YNAI [7] middleware manages and controls the functioning of this OE . As depicted on Figure 2, $\mathcal{S}$YNAI is composed by a set of manager agents "Mng" supervising the actions of agents ("Agent1", "Agent2" and "Agent3") on the OE.

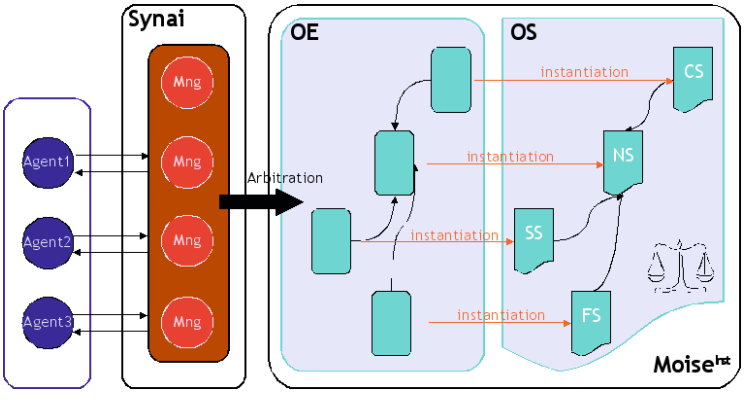

Fig. 2. Supervision by $\mathcal{S}$YNAI of an OE

This layer is in charge of: (i) managing the life cycle of SS as entering/exiting of agents within the Organization, or requesting/leaving of Roles or Groups by the agents, (ii) coordination of the concurrent execution of FS as commitment to Missions or achievement of Goals, etc, (iii) dynamic and evolution of the Organization state through the CS, (iv) the monitoring and supervision of Norms of NS activated/deactivated by the evolution of the Organization.

While agents evolve inside the organization, agents of $\mathcal{S}$YNAI have to interpret and "understand" the OS (in order to respect it or to control it). For that, we need to structure the data of the organization and to this end, we chose recursive graph.

3 Recursive Graph

Recursive graphs are mathematical structures [8] widely used to have a very generic representation of data. They could be very useful to represent any recursive structure such as UML diagrams [9] or to solve complex problems such as optimization of distributed processing [10].

3.1 Formal Definition

Simple graphs could be defined as follows:

$$G(V,E)\begin{cases} V = \{v_1, v_2, .., v_n\} \\ E = \{\{v_i, v_j\}/v_i \in V, v_j \in V\} \end{cases}$$

Where V is a vertex set and E is an edge set of G.

Hypergraphs, introduced by Claude Berge[11] allow to generalize a simple graph:

$$H(V,E)\begin{cases} V = \{v_1, v_2, .., v_n\} \\ E = \{E_1, E_2, .., E_i, .., E_m\} \end{cases}$$

Where E is a set of non-empty subsets E_i off V called hyperedges.

To define a recursive graph we are using some kind of hypergraphs to introduce typed edges. Each edge can have several type t, in order to optimize the memory.

$$H(V,E,T)\begin{cases} V = \{v_1, v_2, .., v_n\} \\ E = \{(t, \{v_i, v_j\})/t \in T, v_i \in V, v_j \in V\} \\ T = \{t_1, t_2, ..., t_p\} \end{cases}$$

Finally a recursive graph $R(V,E,T)$ is a graph composed by a set V of other recursive graphs, a set E of edges between those graphs and a set T of p types.

$$R(V,E,T)\begin{cases} V = \{R_1, R_2, .., R_N\} \\ E = \{(t, \{R_i, R_j\})/t \in T, R_i \in V, R_j \in V\} \\ T = \{t_1, t_2, ..., t_p\} \end{cases}$$

We can represent any simple graph by using *atomic recursive graphs*

$$R_a(\{\emptyset\}, \{\emptyset\}, \{\emptyset\})$$

3.2 Instantiation

In our work-case, recursive graph are helpful to store heterogeneous and recursive data of the Organization Specification. Indeed, $\mathcal{M}$OISE^{Inst} is mostly recursive: in the Structural Specification, Groups can include others Groups; in the Functional Specification, Missions can include others Missions, etc...

In order to represent this kind of data with Object-Oriented Programming we have to design recursive classes for the four specifications. For example, in Figure 3, we show a simple way to represent the Roles and Groups of the Structural Specification. To avoid repeating this design for every specification, we use recursive graphs to obtain a properly factored code (See Figure 4).

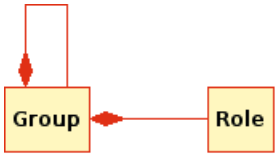

Fig. 3. Simple UML representation of the SS

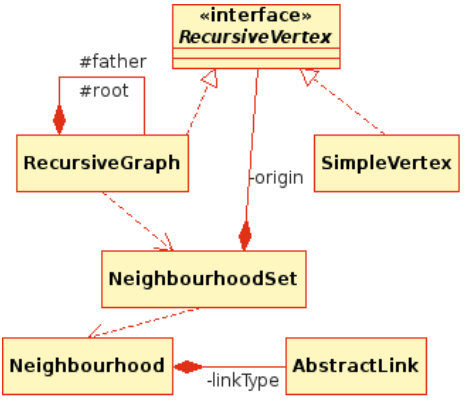

Fig. 4. Simple UML representation of a recursive graph

RecursiveGraph is an object using a *NeighbourhoodSet* composed by a set of *Neighbourhood*: one for every vertex of the graph (the origin of the relation). The class *Neighbourhood* is a set of *RecusiveVertex*, an interface allowing us to store heterogeneous data, that is to say both *SimpleVertex* and *RecursiveGraph*. Through the *Neighbourhood* class we can store all the edges between the origin and others vertexes using the *AbstractLink* class in order to manage several types of edges with the same graph. For the collections of data we use hash-maps (not represented on Figure 4).

Thanks to this structure, we can represent the whole OS. For the exemple above, *Role* implements *SimpleVertex*, *Group* implements *RecursiveGraph* and we have defined every type of relations between roles or groups (communication, authority...) with a class *RoleLink* which implements *AbstractLink*.

To conclude about our data representation, using recursive graphs allow us to:

- Have a factored an coherent code;
- Use the same type of objects for everything which makes the data sharing more easier and that is an obvious pro in a distributed software like $\mathcal{U}$TOPIA[1];
- See the whole problem of the implementation of an Electronic Institution as graph problems;

4 Implementation of Utopia

4.1 Motivations

Our research objectives are to design and implement a flexible and intelligent MAS-factory ($\mathcal{U}$TOPIA) using the Electronic Institution paradigm. Our challenge

[1] $\mathcal{U}$TOPIA is an insTitution Oriented ProgrammIng frAmework.

is to automatically obtain a MAS as close as possible to a MAS build from scratch for a specific use-case.

Indeed, for each Electronic Institution to implement, there is redundant work to do if you do not reuse previously written code. We can briefly reduce the work to:

1. Manage the data of the Organization Specification (OS)
2. Create agents for every Role
3. Manage communication possibilities between agents
4. Allow appropriate data transfers from OS to agents
5. Manage contexts and transitions in order to let the Institution evolve
6. Synchronize agents
7. Find what goals an agent have to realize
8. Implement arbitration layer ($\mathcal{S}$YNAI)
9. Implement what actions have to be done for every specified Goals

As seen before, we use recursive graphs for the data representation and its sharing between agents (1 & 4). Thanks to Jade, a framework dedicated to the agent oriented programming, we handle problems such as communication and synchronization (2, 3 & 6). To fit the others requirements of an Electronic Institution we have implemented several managers, a set of particular agents:

- ContextManager related to the Contextual Specification (5);
- RoleManager for the Structural Specification (2);
- GoalManager dedicated to the Functional Specification (7);
- Supervisor an agent which create and supervise all the managers above and implements $\mathcal{S}$YNAI . (8)

Finally, $\mathcal{U}$TOPIA and its architecture using an Electronic Institution paradigm make the essential problematics of setting up a Multi-Agent System easier. Indeed only two steps are needed:

1. Define the OS in a XML file (an authoring tool to specify the OS will be developped later).
2. Develop specific behaviours (in java classes) that the generic agents will load in order to execute actions achieving the goals defined in the OS.

To do that $\mathcal{U}$TOPIA is based on a set of agents managing each part of organisations entities namely the current contexts, the role played, the goal achieved and the active norms.

4.2 ContextManager

It aims at managing all Contexts (states) in which the system can be found according to the Contextual Specification It can be seen as a finite-state parallel machine running a thread for each Scene. A Scene is a set of contexts, transitions and possibly other sub-scenes.

The Scene's thread starts with the initial vertex and goes to the next state if the received message is a proper transition. The commands run by the ContextManager are mainly transitions and queries for all current states.

Finally, when each thread has finished its execution (when the final state of the CS is reached), ContextManager informs all the agents that the instantiation of the Specification is complete. ContextManager is thus the actor that allows termination of $\mathcal{U}$TOPIA.

4.3 RoleManager

It allows the management of the Roles which involves to:

- Find a distribution of the Roles (what Roles will be played by which agents) and to return the number of Agents needed to instantiate the Structural Specification.
- Provide all the Roles an agent must play or provide the same list again if a given agent asks several times.
- Indicate if all agents play an appropriate Role as needed.

4.4 GoalManager

GoalManager mainly performs a mapping between the Functional Specification Goal set and the concrete instances of classes implementing the actions associated with each Goals When this mapping is done, it can provide classes instances after each agent's queries.

4.5 Supervisor

It is at the top of the system hierarchy and its goals are:

- Creating a recursive graph based on the XML file of the Organization Specification (OS).
- Creating the 3 managers that we have seen: ContextManager, RoleManager and GoalManager.
- Sending OS' subgraphs to the corresponding managers namely CS for the ContextManager, SS for the RoleManager and the FS to GoalManager.
- Caring for the Normative Specification (NS) as the entire OS is required to manage it.

4.6 UtopiaAgent

A first step towards intelligent behavior from agents is to develop generic agents able to specialize independently in order to behave differently. To this end, the agent will retrieve a Role and will act (by running appropriates Goals thanks to the GoalManager) according to the Contexts of the Institution and Norms that apply to it. The main idea is to obtain a self-organization of agents.

5 Demonstration Scenario

Our use-case is part of a demonstrator set up in the context of the RED project [12] which defines and designs solutions to enhance the detection/reaction process, improves the overall resilience of IP networks to attacks by embedding means to enrich the alert with better characterized information, and additional information about the origin and the impact of the security incident.

To provide the detection and reaction functionalities, RED proposes an architecture containing a set of elements, depicted in Figure 5:

- ACE (Alert Correlation Engine): this element is in charge to receive alerts from network nodes, and enhances the detection of attacks by combining several diagnosis combinations.
- PIE (Policy Instantiation Engine): this element receives the information about attacks from the ACE and instantiates new security policies to react to the attack in a high level reaction loop. This paper is focused on this element.
- PDP (Policy Decision Point): this element receives the new security policies defined by the PIE and deploies them in the enforcement points.
- RDP (Reaction Decision Point): this element receives the information about attacks from the ACE and decides of how to act in a mid level reaction loop.
- PEP/REP (Policy Enforcement Point/Reaction Enforcement Point): This component, outside the RED node, enforces the security policies provided by the PDP and the reaction provided by the RDP. It also performs an immediate low level reaction.

RED proposes three different types of reaction based on level of diagnosis required to apply them:

- Immediate reaction, which is an automatic response with a diagnosis based on the capabilities embedded in the device and decided by the PEP/REP,
- Short term reaction, where the diagnosis is done with a limited and local vision of the monitored information system, decided by the RDP based on the information provided by the ACE and which does not instantiate new security policies,
- Long term reaction, where the diagnosis is done with a global vision of the monitored information system, decided by the PIE and which generate new security policies based on the ACE alerts which are sent to the PDP to deploy them in PEP.

A multi-agent system is used to represent RED nodes. Each component is represented by an agent playing a Role (ACE, RDP, PIE, PDP, REP, PEP) of the node which is represented as a $\mathcal{M}\text{OISE}^{Inst}$ Organization. In the following, we will describe the Goals that agents have to achieve in a context of a black-hole attack.

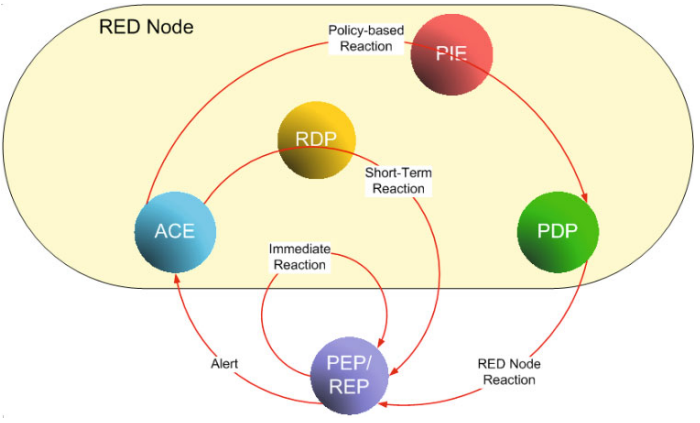

Fig. 5. RED architecture

5.1 Black-Hole Attack and Countermeasures

In our scenario Alice and Bob are communicating with help of a VoIP service provided by a SIP server. A Malicious node executes an attack structured in two successive steps. First, the Malicious node changes the ARP tables of Alice, Bob and the SIP Server (ARP poisoning) in order to have all the trafic routed by itself. Then, it carries out a black-hole attack by dropping (not retransmitting) the packets. As a result, the conversation between Alice and Bob cannot progress.

Once the attack succeeded, an intrusions detection tool detects the attack and sends alerts to the PIE and the RDP through the ACE. The agent playing the Role of RDP have to apply a short term reaction by asking PEP to delete their ARP entries corresponding to the MAC address of the malicious node. The agent playing the Role of PIE aims at implementing new policies forbidding the input and the forward of trafic coming from the malicious node (via its MAC address) and adding static ARP entries binding the real IP addresses and MAC addresses. Then the PIE agent sends these new policies to PDP which transform them into script and/or executable command regarding to PEP's specifications (type, host, OS, etc.). At last, agents playing PEP Role have to execute command and/or scripts on the device they interface. We will see more precisely in the next section how an Organization is implemented with $\mathcal{U}$TOPIA in order to represent a RED node as an Electronic Institution.

5.2 Implementation with Utopia

$\mathcal{U}$TOPIA make possible to easily deploy a MAS where agents play the appropriate Roles, namely ACE, PIE, PDP and PEP from a simple Structural Specification. Thanks to cardinalities, the MAS composition can respect the RED architecture: ACE, PIE and PDP are played by only one agent and PEP are distributed over the network devices.

We can handle the agent behaviour after an attack with a simple Functional Specification: four Missions (one for each agent) composed by two Goals run in parallel, one dedicated to messages reception, the other to message sending.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="xml/os.xsl" type="text/xsl" ?>
<!DOCTYPE OrganizationalSpecification SYSTEM "../xml/os.dtd">
<OrganizationalSpecification id="Red">
    <StructuralSpecification>
        [...]
    </StructuralSpecification>

    <FunctionalSpecification>
        [...]
    </FunctionalSpecification>

    <ContextualSpecification>
        [...]
    </ContextualSpecification>

    <NormativeSpecification>
        [...]
    </NormativeSpecification>

    <DomainKnowledgeSpecification>
        [...]
    </DomainKnowledgeSpecification>
</OrganizationalSpecification>
```

The Normative Specification only force the four agents playing the Roles of ACE, PIE, PDP and PEP to do their associated Missions, that is to say, to run two Java Goal implementations. Obviously, each Goal implementation allow the specialization of the agents, and thanks to $\mathcal{U}$TOPIA's primitive functions, it is very easy to send or receive messages and XML alerts.

– *Structural Specification*: we can easily deploy a MAS where agents play the appropriate Roles, namely ACE, PIE, PDP and PEP. Thanks to cardinalities, the MAS composition can respect the RED architecture: ACE, PIE and PDP are played by only one agent and PEP are distributed over the network devices;

```
<StructuralSpecification>
    <Group id="RED" min="5" max="5">
        <Role id="rPIE" min="1" max="1"/>
        <Role id="rACE" min="1" max="1"/>
        <Role id="rPDP" min="1" max="1"/>
        <Role id="rPEP" min="2" max="2"/>
        <Role id="rPEP-10.13.1.63" min="1" max="1"></Role>
        <Role id="rPEP-10.13.1.18" min="1" max="1"></Role>
[...]
        <Link source="rPIE" destination="rPDP" type="communication" [...] />
[...]
    </Group>
</StructuralSpecification>
```

– *Functional Specification*: we can handle the agent behavior after an attack with several Missions composed by Goals dedicated to messages reception or message sending;

```
<FunctionalSpecification>
    <GoalId>gPIESend</GoalId>
    <GoalId>gACESend</GoalId>
```

```
    <GoalId>gPDPListen</GoalId>
    <GoalId>gPEPListen</GoalId>
    <GoalId>gPEPIPListen</GoalId>
</FunctionalSpecification>
```

– *Contextual Specification*: here there is no reasons to use specific Contexts, we only distinguishe two Contexts (life and dead) and a transition "finish" in order to terminate RED;

```
<ContextualSpecification>
    <Scene id="sRED" initialCtxt="life" finalCtxt="dead">
        <ContextDesc id="dead"/>
        <ContextDesc id="life"/>
        <Transition id="t1"  source="life"  target="dead"  eventId="finish"/>
    </Scene>
</ContextualSpecification>
```

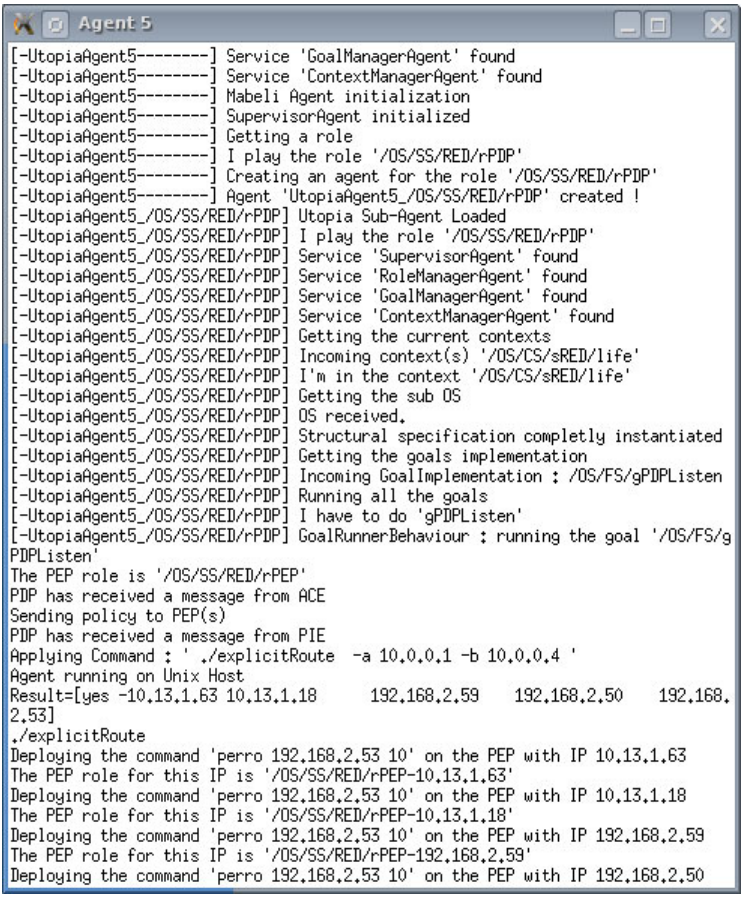

Fig. 6. $\mathcal{U}$TOPIA running the PDP's Role

– *Normative Specification*; force the agents playing the Roles of ACE, PIE, PDP and PEP to do their associated Missions, that is to say, to run appropriate Java Goal implementations..

```
<NormativeSpecification>
    <Norm id="N1"  weight="1"  operator="O"  bearer="rPIE"  issuer="rPIE"  context="life"
      actionT="mission"  action="gPIESend"/>
[...]
</NormativeSpecification>
```

Obviously, all the Goals specified implie concrete actions. For that, $\mathcal{U}$TOPIA permits a mapping between declared Goals and java classes. To make them runnable, they must implement the *GoalImplementation* class and more particulary the "run()" virtual method. Thanks to this virtual class we offer some interesting primitives to the final user so it is very easy to send or receive messages and XML alerts.

The following shows Domain Knowledge Specification of the goals binding them to their corresponding java classes that the user have to provide

```
<DomainKnowledgeSpecification>
    <Goal id="gPIESend" class="red.pie.GPIESend"></Goal>
    <Goal id="gACESend" class="red.ace.GACESend"></Goal>
    <Goal id="gPDPListen" class="red.pdp.GPDPListen"></Goal>
    <Goal id="gPEPListen" class="red.pep.GPEPListen"></Goal>
    <Goal id="gPEPIPListen" class="red.pep.GPEPIPListen"></Goal>
[...]
</DomainKnowledgeSpecification>
```

As results we can see $\mathcal{U}$TOPIA running agents **automatically** specialized in PDP (Figure 6) and PEP (Figure 7)) thanks to its supervisor and managers.

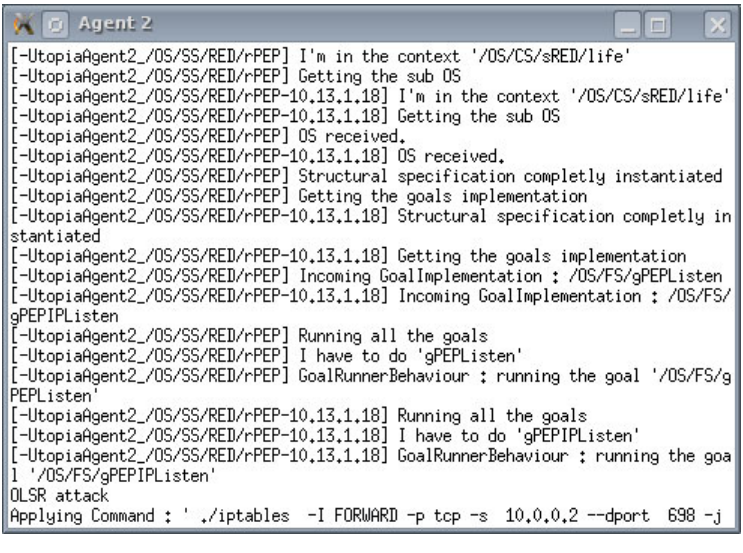

Fig. 7. $\mathcal{U}$TOPIA running the PEP's Role

6 Conclusion

In this paper we described an Electronic Institution programming framework named $\mathcal{U}$TOPIA based on $\mathcal{M}$OISE^{Inst} for the Organization Specification and on recursive graph for the Organization representation. Thanks to a recursive graph, all the homogeneous data are stored in an unique recursive structure, allowing us to easily distribute the shared information between agents of $\mathcal{U}$TOPIA using concepts such as sub-recursive graphs.

The main recursive graph is created by the Supervisor which also run the three managers: RoleManager, ContextManager and GoalManager in order to delegate some computation as we are in a distributed architecture. To this end, Supervisor sends the associated sub-specifications to its managers. This supervision layer allows an Agent playing a particular Role in specifics contexts of the Organization, to get the accurate sub-specification. That is to say its own "Organization view". With this sub-Organization Specification every Agent knows its relationships with other agents (with its sub-Structural Specification) what it may, must or must not do (with its sub-Normative Specification) and finally how to act properly (with sub-Functional Specification).

A set of supervisor and manager agents has been developped in order to help generic agents to specialize themselves autonomously (and therefore smartly) to achieve Organization's objectives by knowing their relationships with other agents (RoleManager), by knowing what they may, must or must not do (Supervisor) and finally by knowing how to act properly (GoalManager). To this end, one difficult question was to determine a way to distribute the Roles between the agents to meet the constraints of the Structural Specification. We reduced this problem to a clique-coloring graph problem that we solved with a clique coloring heuristic hybridized with a classical algorithm for k-coloration into stable sub-sets.

With the RED use-case we showed how easily the essential problematics of setting up a Multi-Agent System could be solved with $\mathcal{U}$TOPIA and its powerful architecture using an Electronic Institution paradigm. Actually $\mathcal{U}$TOPIA allows to simply deploy a MAS without any need of network programming (as Socket coding or thread management). Furthermore, with this kind of network abstraction, the implementation of RED is completely reusable: we can run the system on many different networks. Moreover, it is far easier to brings into the MAS development many security specialists, as Electronic Institution permits to clearly separate the different system Goals and thus, the different security problematics.

Despite the easiness of implementing a working Electronic Institution that $\mathcal{U}$TOPIA brings, as demonstrated in a real use-case, some improvements can be considered. Actually, the way of managers and supervisor to control the functionning of the organization is basically a centralized arbitration system. However the multi-agent system principles advocate decentralization. As a consequence, a first evolution could be done in order to obtain an Electronic Institution allowing the distribution of the OE and $\mathcal{S}$YNAI without putting the optimization of the role distribution aside. Moreover, the agents' decision taking mechanisms could be improved to exhibit a smarter behaviour in order to choose the right Goals to achieve at the right time more efficiently.

Acknowledgment

This work has been financed by TITAN project (C08/IS/21) funded by National Research Fund of Luxembourg (FNR-CORE programme).

References

1. North, D.C.: Institutions, Institutional Change and Economic Performance. Political Economy of Institutions and Decisions. Cambridge University Press, Cambridge (October 26, 1990)
2. Jones, A., Carmo, J.: Deontic logic and contrary-to-duties. In: Handbook of Philosophical Logic, pp. 203–279. Kluwer, Dordrecht (2001)
3. Hübner, J.F., Sichman, J.S., Boissier, O.: A model for the structural, functional, and deontic specification of organizations in multiagent systems. In: Bittencourt, G., Ramalho, G.L. (eds.) SBIA 2002. LNCS (LNAI), vol. 2507, pp. 118–128. Springer, Heidelberg (2002)
4. Esteva, M., Rosell, B., Rodriguez-Aguilar, J.A., Arcos, J.L.: Ameli: An agentbased middleware for electronic institutions. In: AAMAS 2004, pp. 236–243. ACM Press, New York City (July 19-23, 2004)
5. Dignum, V., Vázquez-Salceda, J., Dignum, F.P.M.: OMNI: Introducing social structure, norms and ontologies into agent organizations. In: Bordini, R.H., Dastani, M.M., Dix, J., El Fallah Seghrouchni, A. (eds.) PROMAS 2004. LNCS (LNAI), vol. 3346, pp. 181–198. Springer, Heidelberg (2005)
6. Gâteau, B., Boissier, O., Khadraoui, D., Martinez, F.H.: Controlling an interactive game with a multi-agent based normative organisational model. In: Noriega, P., Vázquez-Salceda, J., Boella, G., Boissier, O., Dignum, V., Fornara, N., Matson, E. (eds.) COIN 2006. LNCS (LNAI), vol. 4386, pp. 86–100. Springer, Heidelberg (2007)
7. Gâteau, B.: Modélisation et supervision d'institution multi-agent. PhD thesis, ENS Mines Saint-Etienne (2007)
8. Harel, D.: Towards a theory of recursive structures. In: Brim, L., Gruska, J., Zlatuška, J. (eds.) MFCS 1998. LNCS, vol. 1450, pp. 36–53. Springer, Heidelberg (1998)
9. Caseau, Y., Krob, D., Peyronnet, S.: Complexité des systèmes d'information: une famille de mesures de la complexité scalaire d'un schéma d'architecture. In: Génie Logiciel, pp. 23–30 (2007)
10. Simon, H.D.: Partitioning of unstructured problems for parallel processing. Computing Systems in Engineering 2, 135–148 (1991)
11. Berge, C.: Hypergraphes. Combinatoires des ensembles finis. Gauthier-Villars (1987) ISBN 2-04-016906-7
12. Feltus, C., Khadraoui, D., de Remont, B., Rifaut, A.: Business governance based policy regulation for security incident response. In: Crisis 2007, Marrakech, Morocco (July 2-5, 2007)

A Lightweight Approach to Enterprise Architecture Modeling and Documentation

Sabine Buckl, Florian Matthes,
Christian Neubert, and Christian M. Schweda

Technische Universität München, Institute for Informatics,
Boltzmannstr. 3, 85748 Garching, Germany
{sabine.buckl,matthes,neubert,schweda}@in.tum.de
http://www.systemcartography.info

Abstract. Not quite a few enterprise architecture (EA) management endeavors start with the design of an information model covering the EA-related interests of the various stakeholders. In the design of this model, the enterprise architects resort to prominent frameworks, but often create what would be called an "ivory tower" model. This model at best case misses if not ignores the knowledge of the people that are responsible for business processes, applications, services etc. In this paper, we describe how the wisdom of the crowds can be used to develop information models. Making use of Web 2.0 techniques, wikis, and an open templating mechanism, our approach ties together the EA relevant information in a way, which is accessible to both humans and applications. We demonstrate how the ivory tower syndrome can be cured, typical pitfalls can be avoided, and employees can be empowered to contribute their expert knowledge to EA modeling and documentation.

Keywords: enterprise architecture management, wikis, wisdom of the crowds, information model, collaborative modeling.

1 Introduction

Trends as globalization, rapid economic change, and the necessity to foster an organization's sustainable competitive advantage have considerably increased the importance of knowledge on the internal structure of an organization. While this knowledge is typically already existing in different data sources, e.g. CMDBs, process descriptions, or as implicit expert knowledge, linking these data sets is a challenge. Enterprise architecture (EA) management is an instrument to address this challenge by providing methods and means to enable a holistic view on the organization to foster the alignment of business and IT (cf. [1,2]).

Providing a holistic view on the EA yields two main challenges. First, to link the different data sources a common *information model* describing the EA, i.e. "the fundamental organization of [the enterprise] embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution" [3] has to be developed. This development more

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 136–149, 2010.
© Springer-Verlag Berlin Heidelberg 2010

often than not suffers from the 'ivory-tower syndrome', i.e. leads to the creation of a wish list in which each stakeholder raises concerns that he is interested in. This results in an unmaintainable giant information model, as documented in the EAM anti-pattern of the same name described by Buckl et al. in [4] . Second, documenting the EA is an advanced topic, as the EA typically includes different organizational units, a few hundreds up to thousands application systems and their interconnections, etc. The documentation process accordingly must be conducted in a way, which is on the one hand feasible for stakeholders with various backgrounds as e.g. process or application owners, and which on the other hand shows the benefits of their time spent on sharing knowledge.

To address challenges of the latter type, a new generation of systems has emerged in the last years facilitating team collaboration and knowledge exchange. These systems of which for example Wikipedia[1] is a prominent example utilize so-called *Web 2.0* technologies (cf. Oreilly in [5]). Due to the success of these tools in the Internet, the techniques are currently adopted and transfered to the enterprise context under the name *Enterprise 2.0* (cf. McAfee in [6,7]). Today's Enterprise 2.0 tools provide many distinct services, such as collaborative authoring, tagging, bookmarking, awareness, commenting, rating, linking, search, social networking, access control, as well as a multitude of different content types, such as wikis, blogs, or files [8].

In this article, we demonstrate how the "wisdom of the crowd" [9] may be used to enable EA documentation and modeling. By making use of Web 2.0 techniques, wikis, and an open templating mechanism, we show how the ivory tower syndrome can be cured, typical pitfalls can be avoided, and employees are empowered to contribute their expert knowledge. In Section 2, we introduce selected Web 2.0 and Enterprise 2.0 techniques. Utilizing these techniques, we present the hybrid wiki approach in Section 3. We further illustrate how it contributes to a lightweight approach to EA modeling as well as documentation and discuss use cases in the EA management context in Section 4. Complementing the illustration of the approach, Section 5 describes a prototypic implementation based on an existing enterprise wiki system. The paper concludes with a critical reflection of the achieved results and sketches areas of future research in Section 6.

2 Enterprise 2.0 Techniques

In [9] Surowiecki writes about collective intelligence and its impact on enterprises. He explains that collective intelligence of a group mostly surpasses the knowledge of particular experts and how companies can benefit from this fact, which he refers to as "the wisdom of crowds". Especially in context of managing socio-technical systems, such as enterprises, where a multitude of stakeholders with different backgrounds should be involved, Enterprise 2.0 techniques can be applied beneficially. Considering for instance the design of a suitable information model to model the architecture of an enterprise yields challenges that

[1] `www.wikipedia.org`

can be faced by utilizing the wisdom of crowds. More often than not, the ivory tower syndrome can be observed when an information model is developed, where a group of enterprise architects creates an 'invented' information model. Such model does on the one hand not adequately address the concerns of the corresponding stakeholders and on the other hand lacks acceptance as it is based on terms and definitions of a dedicated user group and therefore might be difficult to understand for other stakeholders. Enterprise 2.0 techniques can be used to cure this syndrome by enabling employees of all departments to make explicit and contribute their expert knowledge for parts of the EA they are responsible for (e.g. business processes, applications, IT infrastructure) thereby resulting in a 'vivid' information model.

2.1 Collaboration, Content Sharing, and Discretionary Access Control

Enterprise 2.0 is characterized as the usage of social software within enterprises [6,7]. A major objective of social software is to empower users to collaborate via the Internet. The users can create different kinds of content objects, such as wikis, blogs, files, which they can share with other users within certain groups. For instance, a group can be a part of the user's personal social network, or a group could have been created for a certain purpose of collaboration (cf. "functional groups" in [8]). Especially in enterprises the sharing of content requires mechanisms to protect sensitive content (objects) from unauthorized access. In [8] this mechanisms are described as "access control" services for integrated Enterprise 2.0 platforms.

Since the architecture of an enterprise can be very complex, consisting e.g. of hundreds or even thousands of business applications, the documentation and modeling of an EA requires the involvement of many different stakeholders with various backgrounds (cf. [10,11,12]). Collaboration, synchronization, and content sharing is most likely to be an appropriate approach to gather EA-related information and document constituents of an enterprise, in order to provide a holistic view on the EA. Thereby functional groups can be used to grant access to particular content objects, e.g. customers could be involved in the documentation process for some parts of the EA[2]. Furthermore, social networks may help to find persons which are experts for a certain part of the EA, e.g. a specific business process, to get in contact with for in-depth questions and discussions.

2.2 Notifications and Recommendations

Todays integrated Enterprise 2.0 platforms provide mechanisms to get users notified in case of certain system activities, which is referred to by Büchner et al. in [8] as "awareness". Users can register to get notifications to a) keep track of other users activities (e.g. a user creating a new wikipage) and b) to follow activities on certain content objects (e.g. a wikipage being modified by a user).

[2] A customer could be interested in providing information on a shared application interface which is critical for his business.

These activities are mirrored in a user's stream of notifications made available via a personal dashboard within the platform and typically complemented by an RSS feed.

Since the documentation of an EA has a strongly collaborative character, as discussed before, group-based notifications on modifications and communication of ongoing documentation initiatives is often regarded to be crucial for following reasons:

- A user responsible for certain parts of the EA is getting notified if corresponding parts of the documentation have changed, e.g. to perform quality assurance.
- The head of a group of users employed with documenting parts of the EA can keep track on the progress made in the documentation endeavor.

The notification mechanisms, as implemented in todays Enterprise 2.0 platforms, may be appropriate to address the demands as sketched above. Nevertheless, further demands may exist, e.g. a user might want to get notified on objects, which have not been changed for a certain period of time. These notification might give indications on parts of the EA documentation becoming outdated. Even though this service is not supported by any of the currently available platforms out-of-the-box (cf. Büchner et al. in [8]), such notifications can be realized based on the "last modification date" of a content object.

In *recommender systems* a user profile and user activities are analyzed in order to find information (e.g. content objects, other users) that are most likely to be interesting for the user under consideration. For instance, if a user visited wikipages that are almost all tagged with a certain keyword, it is most probably that he is interested in other wikipages which are tagged just the same. In this case the system could send him a recommendation message to visit these pages. Even though recommender systems are rather rare supported by Enterprise 2.0 platforms, this can be valuable for EA documentation to

- facilitate knowledge exchange, especially in large size companies employees often do not know the competences of their colleagues,
- to identify experts for certain parts of the EA for in-depth questions and discussions, and
- to discover already documented parts of the EA, which can be useful to avoid duplicated documentation efforts.

2.3 Structured Content in (Enterprise) Wikis

Wikis are the most widely supported content type of today's integrated Enterprise 2.0 platforms. A wikipage as part of a wiki may be characterized as a website, that can be directly edited in a web browser. Wikipages enable users to publish and share their knowledge in a way, which is easily accessible and thereby provides an intuitive way to collaboratively create and modify content. The most prominent wiki system in the internet is the encyclopedia Wikipedia. In Wikipedia each page represents a particular concept, e.g. a city, which is mainly

described in the content of the wikipage. Beside this description a wikipage additionally contains *metadata* of the represented concept, e.g. population and foundation date, which are shown as key-value-pairs in a tabular view. Since it would be laborious to *re-define* these metadata keys for each particular city to be describe in future, Wikipedia provides a reuse mechanism, called *templates*. Templates are prototypes for a certain concept with several attributes. Considering the table as explained above, these attributes are the keys of the metadata-table. Templates as used by the Wikipedia are rather unusual in Enterprise 2.0 platforms, since users have to learn a specific markup-language.

Since wiki templates enable to abstract, reuse and unify information, they might be a suitable instrument for EA documentation activities. An enterprise architect for example could predefine a set of templates in order to give suggestions how to document a certain part of the EA. As already mentioned before, an information model invented by a single person is likely to be unsuitable for EA management, where many different stakeholders should be involved. In Section 3 we describe how templates can be used to create an information model in a bottom-up fashion.

2.4 Linking and Bookmarking

In [13,14] the importance of relating EA-related information from different layers[3] like business processes, application systems, and infrastructure elements and providing a navigation mechanism between distinct information objects is discussed. This fact is also reflected in almost every Enterprise 2.0 platform. In order to connect any kind of content objects, these platforms supply several services for the management of internal and external links [8]. Considering internal objects, *backlinks* are further supported. Backlinks reference objects the currently considered content object is referenced by. The number of backlinks can be regarded as an indication for the popularity of a content object. In order to save and share links to favorite wikipages, the feature of *social bookmarking* is provided, which further supports tagging of pages.

3 Hybrid Wikis for EA Management

While utilizing the wisdom of the crowds requests for a way to store unstructured information, an EA management initiative builds on a common understanding of the constituents that make up the EA, i.e. structured information. Therefore, we present an approach building on the capability to combine structured and unstructured information in a wiki system. While an open-templating mechanism is used to store the structured parts of a wikipage, the content of the page contains the unstructured information. A wiki supporting to store structured and unstructured information is hence called a *hybrid wiki*. Our hybrid wiki approach is demonstrated alongside an open source web collaboration system

[3] See Winter and Fischer in [15] for typical layers in the context of EA management.

called Tricia [16], which can be characterized as an Enterprise 2.0 tool according to Büchner et al. in [8]. Tricia provides several content types, such as wikis, files, and blogs.

A template may be considered a simple table, providing multiple attribute-value-pairs and a global name. The name of the template is used to specify the concept which is described in the wikipage's content, e.g. in the context of EA management this could be the concept of an "Application System". Continuing this example, the wikipage could describe an actual application system "SAP Business Suite" in the page's content. "SAP Business Suite" obviously represents the name of the application system and therefore can be added as an attribute-value-pair (name: SAP Business Suite) to the template. In the content further attributes (and corresponding values) can be identified and also be added to the template, e.g. version, availability, criticality etc. By doing a meta-structure is continuously developed on the wikipages and the corresponding templates. Beside literals hyperlinks can be used as values of template attributes, referencing other wikipages also having a certain template assigned. Thereby, an information model is created bottom-up via the templates – corresponding to concepts in the information model – with the attribute and links – corresponding to the association in the information model.

Templates are connected with a corresponding meta-type, which is initialized when a new template is created on a wikipage "template definition". A template definition has a name and provides all attributes of the corresponding templates. Considering the previous example, the template definition has the name "Application System" and provides the following attributes: name, version, availability, and criticality. That means, all templates are instances of a certain template definition and furthermore, existing template definitions can be assigned to wikipages as templates. To manage the template definitions, e.g. create new definitions, rename attributes, type attributes, a second way of more restrictive modeling is enabled, which we refer to as top-down information modeling.

Figure 1 summarizes an application example of out approach. The enterpris hybrid wiki is therein used by several technical departments in the company for documenting relevant information. By using templates, they collaboratively elicit an information model which is further "gardened" by an enterprise architect. This means that the enterpris architect uses reporsts on the used templates as well as their associated attributes to decide on the attributes to keep, to set mandatory or to abandon. Further, the architect decides on "refactorings" of the attributes, i.e. on name changes, etc. The employees of the department start creating wikipages for each element to be documented. They assign templates to the wikipages in order define the concept described in the page, structure the content, and link to other pages the modeled concept is related to. Since all employees of the department are in the same functional group, i.e. are members of Technical Department I, all users of this group are notified by any system changes via RSS feeds, e.g. when a template of a wikipage changes or a template attribute is renamed. Since the users are aware of new content and changes, they can use other Enterprise 2.0 services to improve the quality of the content, e.g. by means of discussing, commenting, and rating services.

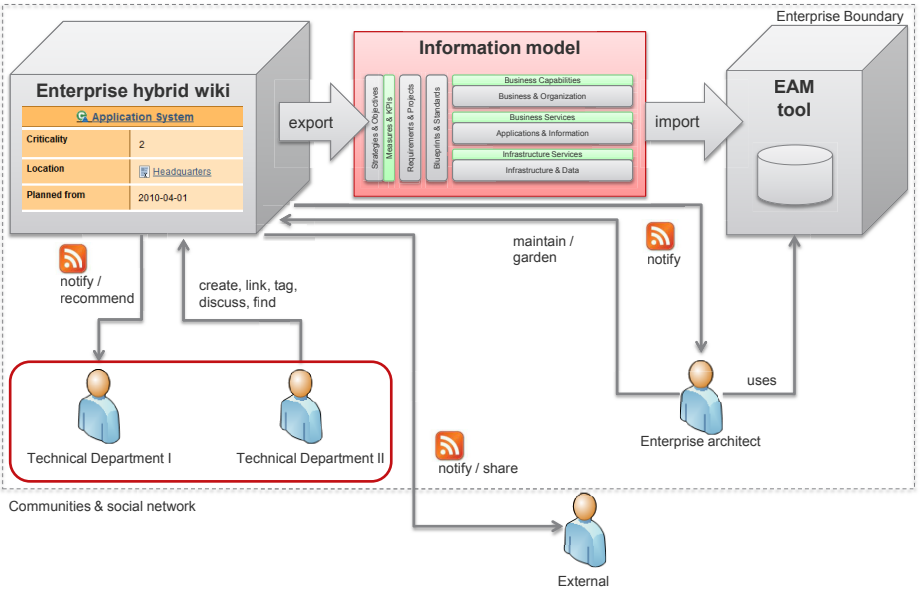

Fig. 1. Enterprise Hybrid Wiki for EA management

The system may also recommend wikipages, which might be useful for particular users within the group, by analyzing the content of the pages and user behavior. In the example, an *external user* is allowed to follow the system changes by RSS notifications. For instance, it could be useful to keep customers informed about parts of the EA which are relevant for their business. One could also imagine to grant write access to external users, e.g. allowing customers to comment on wikipages, or take part in discussions.

4 Use Cases

In the following we describe the use cases a hybrid wiki as discussed in the above section needs to correspond to. These use cases mirror the information model related use cases put forward by Matthes et al. in [14] and form the foundation for the prototypic implementation described in Section 5. We distinguish between use cases for the open templates, which can be assigned to wikipages and for the management of their definitions.

Uses cases for the open templating mechanism

New templates can be defined on a wikipage and existing templates can be assigned to a wikipage. Each wikipage can only have one template assigned (UC10). After the assignment of the template all attributes available for this template are shown with empty values (UC20). A new attribute can be added to a template which is assigned to a wikipage, i.e. the new attribute is added to the

template definition. In response on all other wikipages having the same template assigned an empty value is shown for the new attribute (UC30). Values can be set for attributes of a wikipage template. A value can either be literal or a link (UC40). An attribute of a wikipage template can be deleted. If another wikipage has the same template assigned and a non-empty value for this attribute, the attribute remains in the template definition. Otherwise the attribute is deleted in the template definition (UC50). Templates can be removed from a wikipage. If the template is not assigned to another wikipage, the corresponding template definition is deleted (UC60). An attribute can have multiple values. In this case a mixture of both text and link values is allowed (UC70). A template has the same access rights as its owning wikipage (UC80).

Use cases for managing templates

A new template definition can be created independent of wikipages. A template definition has a unique name (UC90). A new attribute can be added to an existing template definition. The attribute has a unique name (UC100). For each template definition the following statistics of its templates are provided (UC110):

- A numeric indication is given how many templates belong to a template definition, i.e. total number of instances.
- A numeric indication for each attribute is shown illustrating how many templates have not empty values, i.e. non-empty attributes.

For a template definition the corresponding templates with their attribute values are shown in an tabular view (UC120). A template definition attribute can be renamed, i.e. all corresponding template attributes are renamed too (UC130). A template definition attribute can be deleted, i.e. all corresponding template attributes are deleted, too (UC140). A template definition can be deleted, i.e. all corresponding templates are deleted, too (UC150). By default all in the system registered users may read a template definition and create a new one. The access rights can be defined more restrictive (UC160).

5 Prototype Implementation

According the use cases specified in Section 4 we introduce a prototypic implementation of our hybrid wiki approach. The implementation is based on the web collaboration and knowledge management tool Tricia [16]. Tricia is an open source web framework which enables a straightforward development of custom applications by providing extension points and a plugin mechanism, following a data model-driven approach. Since Tricia belongs to the category of Enterprise 2.0 tools [8], it provides many of the services as introduced in Section 2, e.g. role-based access control, search, notification, etc. and additionally provides a wiki plugin out-of-the-box. Subsequently, we illustrate the concepts of the prototype's data model, present exemplary user interfaces for templates and their definitions, and show how an information model can be extracted from these templates.

Data model

The data model of the hybrid wiki application is shown in Figure 2. A `TemplateDefintion` provides a unique name, the name of the concept, e.g. business application. Each `TemplateDefinition` has a rich text field to enable users to add a detailed description of the concept via a hypertext editor and may have multiple features (`TemplateDefinitionFeatures`), also providing a unique name attribute, e.g. availability.

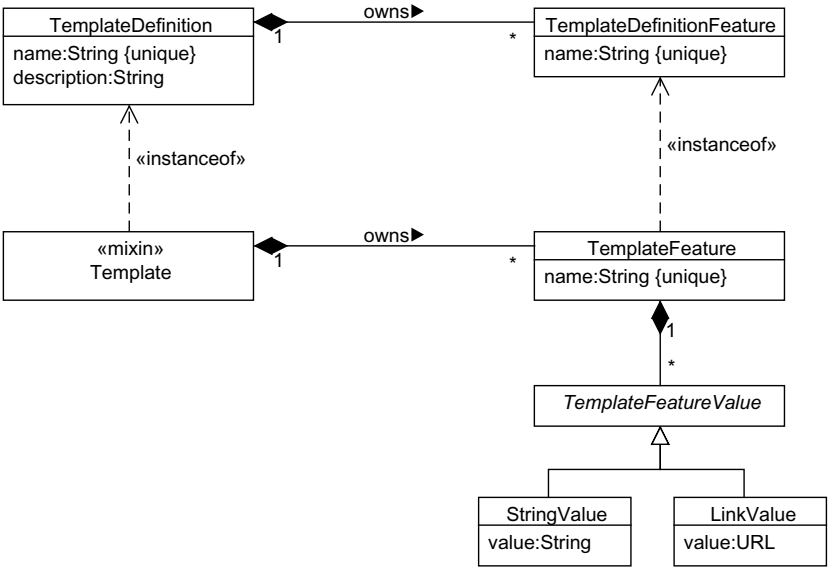

Fig. 2. Data model of the hybrid wiki approach

Templates are realized by using the mechanism of OptionalMixins of the Tricia framework [16], which allow dynamically attaching a `Template` to a Tricia wikipage at runtime. Each `Template` belongs to exactly one `TemplateDefinition` and may have multiple features (`TemplateFeature`). The `TemplateFeature`s of a `Template` comply with the `TemplateDefinitionFeature`s of its `TemplateDefinition` (i.e. their names are matching), with the exception, that a `TemplateFeature` is only stored in the database if at least one value (literal or link) is assigned. Each `TemplateFeature` may have multiple values (`TemplateFeatureValue`), latter can either be a literal (`StringValue`) or a hyperlink (`LinkValue`). Since Tricia treats links as first class objects, the link attribute of `LinkValue` only contains the primary key of the linked object.

Management of templates

In the example of Figure 3 a `TemplateDefinition` of the concept "Application System" is given. The application system provides a detailed hypertext description of its purpose and several attributes (`TemplateDefinitionFeatures`), e.g.

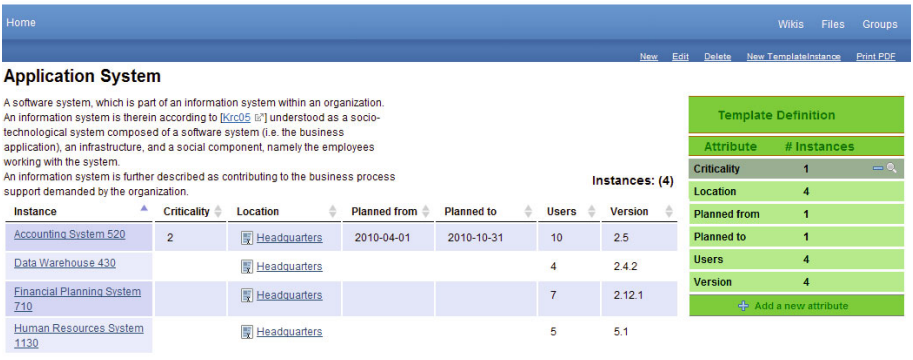

Fig. 3. Definition for the template application system

criticality, location, etc. Attributes can be added, edited and removed, and both kind of template statistics are provided (c.f. Section 4). Furthermore the corresponding templates with their attribute values are shown in an tabular view. The *New* action shown in the header of the page enables user to create new template definitions independent of wikipages.

Templates as part of wikipages

Figure 5 shows a wikipage having an application system template assigned. Since the site is a wikipage, the application system named *Accounting System 520* can be described in detail assisted by a hypertext editor (WYSIWYG editor). In the example the *Location* attribute contains a hyperlink to an organizational unit *Headquarters*, where the *Application System 520* is located at, i.e. Headquarters is a wikipage, which has an *Organizational Unit* template assigned (cf. Figure 4). The remaining attributes of *Accounting System 520* are literal values (`StringValue`). The attribute *Users* is provided by the template definition *Application System*, but not inscribed in the template, i.e. not stored as a `TemplateFeature`.

The template can be removed from the wikipage, attributes can be added (+), modified (input field), and removed (-). In the division *Referenced by* at the bottom right all templates are listed having an attribute (`TemplateFeature`) with a hyperlink (`LinkValue`) to *Accounting System 520* grouped by template definitions, i.e. in the example *Accounting System 520* is referenced by two *Organizational Unit* wikipages, namely *Headquarters* and *Finance*, from the attribute *Application* (c.f. Figure 4). The *Application* attribute from the *Headquarters* template in Figure 4 uses multiple hyperlinks (`LinkValue`) to list all deployed applications.

Information model

Considering the sample data from above, an information model, which is shown as an UML class diagram in Figure 6, can be extracted from the template

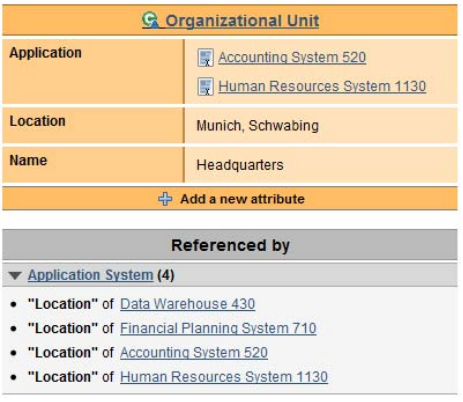

Fig. 4. Template *organizational unit* of a wikipage *Headquarters*

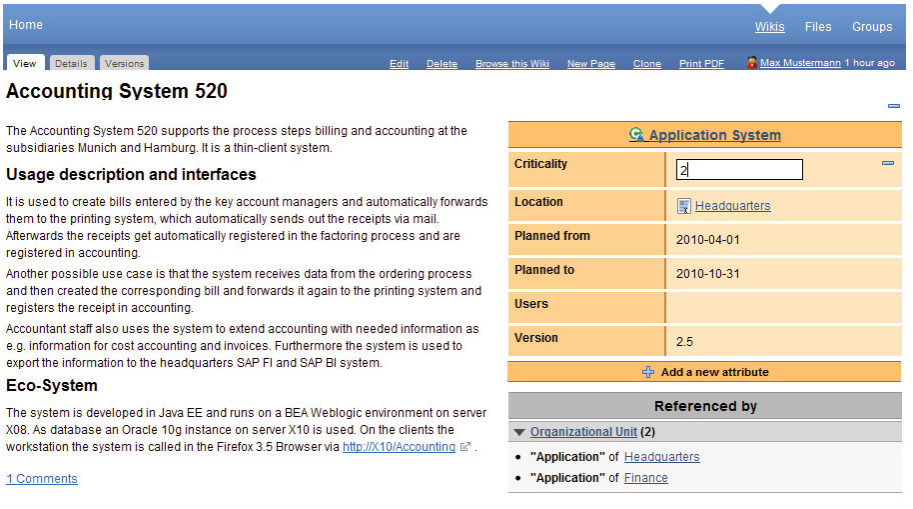

Fig. 5. Wiki page with template *Application System*

definitions and their template instances. Thereby each template definition with its literal attributes (`StringValue`) is mapped to an UML class with corresponding string attributes. Each hyperlink value (`LinkValue`) of a template attribute referencing a wikipage also having a template assigned is mapped to an UML association between the classes representing both, the referenced and the referencing template. Thereby a role name which complies with the name of the template attribute is created for the association. In the example two associations emerge – location and application. The multiplicity of *application* may be assumed as arbitrary (*) since the application attribute of *Headquarters* (c.f. Figure 4) has multiple link values to *Application System* templates assigned. The

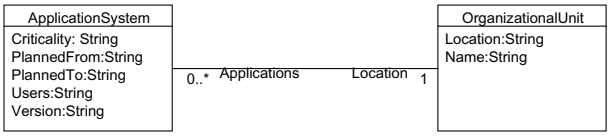

Fig. 6. Information model extracted from templates

multiplicity of *location* is assumed to be exactly one (1) since all *Application System* templates reference exactly one *Organizational Unit*, namely *Headquarters* (cf. Figure 3).

6 Summary and Future Work

In this paper, we demonstrated how Web 2.0 techniques, wikis, and an open templating mechanism can be used to support EA management. We described how the wisdom of the crowds can be utilized for the documentation of EA-related concepts and their instances collaboratively in wikipages. Furthermore, we introduced an open templating mechanism for wikipages to collaboratively evolve an information model in the context of EA management. Thereby both ways of lightweight EA modeling can be applied, bottom up by evolving the model by means of the templates and top down by using template definitions. The theoretic discussion was complemented by the description of a prototypic implementation of the hybrid wiki approach based on the open source Enterprise 2.0 tool Tricia (cf. [16]). Other Tricia services may be useful in the context of EA management, e.g. change feeds, to enable users getting notified when the information model or the documentation changes. Therefore, future areas of research encompass the

- export of the information model based on the Eclipse Modeling Framework [17],
- integration of generated visualizations of the information model in wikipages for different stakeholders [18],
- beside literals and links, support of other data types for template attributes, e.g. Date, Integer,
- validation rules and constraints for template attributes, e.g. mandatory attributes,
- semantic annotations within the wiki page's content,
- template-based recommender system with respect to the groups a user belongs to, and
- support of inheritance for template definitions.

Further aspects of interest are e.g. the utilization of rating functions on wikipages to allow users to provide feedback in a structured and uniform way. Concerning these functions a multitude of implications, e.g. on user motivation would have to be considered. Additionally, these ratings could provide valuable input

to the analysis of documentation templates, although a clear correlation between a low rating and the corresponding template used is not likely to be easily determinable. The Hybrid Wiki approach is currently being evaluated in practice.

References

1. The Open Group: TOGAF "Enterprise Edition" Version 9 (2009), http://www.togaf.org (cited 2010-02-25)
2. Zachman Institute for Framework Advancement: The zachman enterprise framework (2009), http://www.zachmaninternational.com/index.php/the-zachman-framework (cited 2010-02-25)
3. International Organization for Standardization: ISO/IEC 42010:2007 Systems and software engineering – Recommended practice for architectural description of software-intensive systems (2007)
4. Buckl, S., Ernst, A.M., Matthes, F., Schweda, C.M.: How to make your enterprise architecture management endeavor fail? In: Pattern Languages of Programs 2009 (PLoP 2009), Chicago (2009)
5. Oreilly, T.: What is web 2.0: Design patterns and business models for the next generation of software. Social Science Research Network Working Paper Series (August 2007)
6. McAfee, A.: Enterprise 2.0: The dawn of emergent collaboration. IEEE Engineering Management Review 34(3), 38 (2006)
7. Bughin, J.: The rise of enterprise 2.0. Journal of Direct, Data and Digital Marketing Practice 9(3), 251 (2008)
8. Büchner, T., Matthes, F., Neubert, C.: A concept and service based analysis of commercial and open source enterprise 2.0 tools. In: International Conference on Knowledge Management and Information Sharing, Madeira, Portugal, pp. 37–45 (2009)
9. Surowiecki, J.: The Wisdom of Crowds. Random House, New York (2004)
10. Fischer, R., Aier, S., Winter, R.: A federated approach to enterprise architecture model maintenance. In: Enterprise Modelling and Information Systems Architectures – Concepts and Applications, Proceedings of the 2nd International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2007), St. Goar, Germany, October 8-9, pp. 9–22 (2007)
11. Kurpjuweit, S., Winter, R.: Viewpoint-based meta model engineering. In: Reichert, M., Strecker, S., Turowski, K. (eds.) Enterprise Modelling and Information Systems Architectures – Concepts and Applications, Proceedings of the 2nd International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA 2007), St. Goar, Germany, Bonn, October 8-9, LNI, pp. 143–161. Gesellschaft für Informatik (2007)
12. Moser, C., Junginger, S., Brückmann, M., Schöne, K.M.: Some process patterns for enterprise architecture management. In: Software Engineering 2009 – Workshopband, Bonn, Germany. Lecture Notes in Informatics (LNI), pp. 19–30 (2009)
13. Kurpjuweit, S., Aier, S.: Ein allgemeiner Ansatz zur Ableitung von Abhängigkeitsanalysen auf Unternehmensarchitekturmodellen. In: 9. Internationale Tagung Wirtschaftsinformatik (WI 2007), Wien, Austria, pp. 129–138. Österreichische Computer Gesellschaft (2007)

14. Matthes, F., Buckl, S., Leitel, J., Schweda, C.M.: Enterprise Architecture Management Tool Survey 2008. Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany (2008)
15. Winter, R., Fischer, R.: Essential layers, artifacts, and dependencies of enterprise architecture. Journal of Enterprise Architecture 3(2), 7–18 (2007)
16. InfoAsset: Tricia – open source web collaboration and knowledge management software (2010)
17. Moore, B., Dean, D., Gerber, A., Wagenknecht, G., Vanderheyden, P.: Eclipse development using the graphical editing framework and the eclipse modeling framework (2004), http://www.redbooks.ibm.com/redbooks/pdfs/sg246302.pdf (cited 2010-02-25)
18. Buckl, S., Ernst, A.M., Lankes, J., Matthes, F., Schweda, C., Wittenburg, A.: Generating visualizations of enterprise architectures using model transformation (extended version). Enterprise Modelling and Information Systems Architectures – An International Journal 2(2), 3–13 (2007)

Towards Flexible Process Support on Mobile Devices

Rüdiger Pryss, Julian Tiedeken, Ulrich Kreher, and Manfred Reichert

Institute for Databases and Information Systems, Ulm University, Germany
{ruediger.pryss,julian.tiedeken,ulrich.kreher,manfred.reichert}@uni-ulm.de

Abstract. Ubiquitous computing is considered as enabler for linking everyday life with information and communication technology. However, developing pervasive and mobile applications that provide personalized user assistance still constitutes a challenge. Mobile application scenarios are diverse and encompass domains like healthcare, logistics, and sales. For their support two fundamental technologies with increasing maturity are emerging: *development frameworks for mobile devices* and *light-weight process engines*. Their integrated use, however, is in a rather premature state. Generally, the use of a process engine for supporting mobile collaboration raises many challenging issues. This paper picks up some of these challenges and shows how we have coped with them in the MARPLE project. MARPLE targets at a tight integration of process management technology with mobile computing frameworks in order to enable mobile process support in advanced application scenarios. We give insights into the MARPLE architecture and its components. In particular, we introduce the MARPLE process engine, which enables light-weight as well as flexible process support on mobile devices. This will be key for mobile user assistance in advanced application scenarios.

Keywords: Mobile Process, Distributed Process, Process Management.

1 Introduction

Mobile assistance in daily life as empowered by information and communication technology is a much discussed topic [1,2]. To better understand relating challenges, we analyzed real-world process scenarios from different domains in which mobile user assistance is urgently needed. This includes process scenarios from healthcare, logistics, and sales. Altogether, our studies revealed the fundamental role of process support in the context of mobile and personalized user assistance. In this paper we pick up a realistic healthcare scenario in which chronically ill patients shall be assisted by mobile devices. The latter shall give recommendations about therapies (e.g., medications), collect data from the patient, or help general practioners to plan encounters with their patients. Recommendations may be made remotely by healthcare professionals and often depend on previously gathered patient data (e.g., blood pressure or blood sugar). Despite its high potential, so far, there exists no comprehensive mobile user assistance for realizing such scenarios in a flexible way and in a large scale. One major task in

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 150–165, 2010.
© Springer-Verlag Berlin Heidelberg 2010

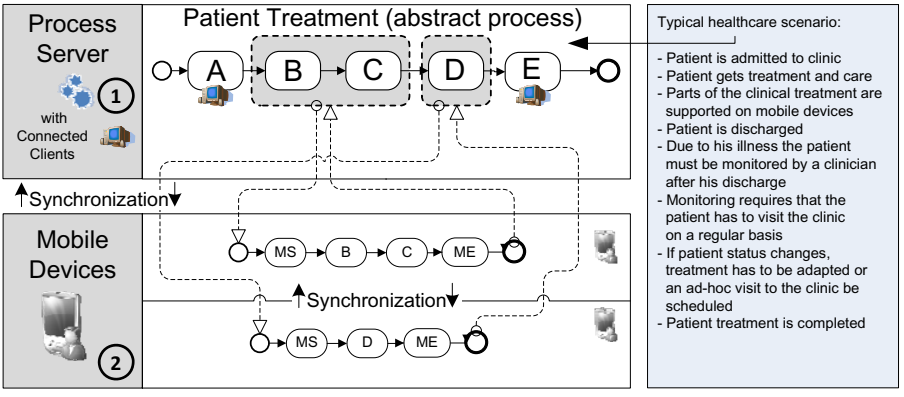

Fig. 1. Abstract Healthcare Scenario (see Fig. 2 for a connect scenario)

this context is to decide which parts of a global process (i.e. *process fragments*) shall run on mobile devices (e.g., a patient's mobile) and which ones shall be controlled by process servers (e.g., running on the physician's workstation or any other backend server). In the following we refer to such a healthcare scenario to discuss fundamental challenges, and to show the benefits coming with mobile user assistance. Fig. 1 illustrates both a traditional realization of this scenario (①) and its implementation based on mobile user assistance (①+②). After discharging patients the usual way to monitor their health status is to schedule regular visits for them in the clinic. In certain cases, however, this can lead to delayed adaptations of treatment plans, e.g., if patient status changes while they stay at home. To improve this situation and to enable remote monitoring of patients (cf. Fig. 1 Steps B,C), mobile data collection and mobile user assistance ② would be highly welcome by all parties; i.e., patients should be assisted by a mobile device which gathers medical data from them and informs clinicians about important status changes. Further, it should enable clinicians to remotely adapt their treatment plan or to schedule meetings with patients if required. In addition, it might be desirable to run parts of the treatment procedure on mobile devices (cf. Fig. 1 Step D). To realize such scenario and to provide personalized treatment support, patient-specific application logic needs to be provided on the mobile device. As a consequence, the overall treatment process is partitioned ② and the resulting process fragments are deployed to both process servers and mobile devices. In particular, process fragments running on mobile devices need to be quickly configurable to the specific patient and be dynamically adapted if patient's status changes. Hard-coded process implementations are therefore not adequate in this context. Instead flexible support of processes with mobile assistance is needed.

To enable mobile process assistance we developed a light-weight process engine called *MARPLE* [1] that runs on mobile devices and is able to interact with backend processes if required. In addition, we provide advanced tools for defining,

[1] **MA**naging **R**obust mobile **P**rocesses in a comp**LE**x world.

configuring, verifying and deploying process fragments in such an environment. In the following, we focus on fundamental challenges on the one hand and on the core architecture and components of the *MARPLE* mobile process engine on the other hand. Conceptual issues related to the partitioning of processes as well as to the synchronization of the resulting process fragments are outside the scope of this paper. When developing the *MARPLE* engine we had one shining example in mind - the ADEPT process management system, which we had developed during the last decade [3]. In particular, we adopted basic design principles from ADEPT (e.g., its correctness-by-construction principle and its dynamic process change capabilities), but also aligned the *MARPLE* architecture with specific needs relating to mobile process support.

The remainder of this paper is organized as follows: Section 2 introduces a concrete application scenario. In Section 3 we describe requirements derived from case studies as basis for the *MARPLE architecture*. Section 4 gives insights into the MARPLE architecture, while Section 5 shows how the described application scenario can be supported. In Section 6 we discuss selected *MARPLE* features in the context of advanced application scenarios. Section 7 discusses related work and Section 8 concludes with a summary and outlook.

2 Application Scenario

Fig. 2 shows a typical healthcare process (modeled in terms of BPMN) involving three parties: clinic, patient, and homecare. The first swim lane shows activities as performed in the clinic, which also starts the process. When completing Step ①, the execution of the process fragment on the mobile device of the patient is triggered (patient swimlane). This mobile process then collects medical data from the patient and coordinates required actions; e.g., to measure blood pressure or to gather ECG data.

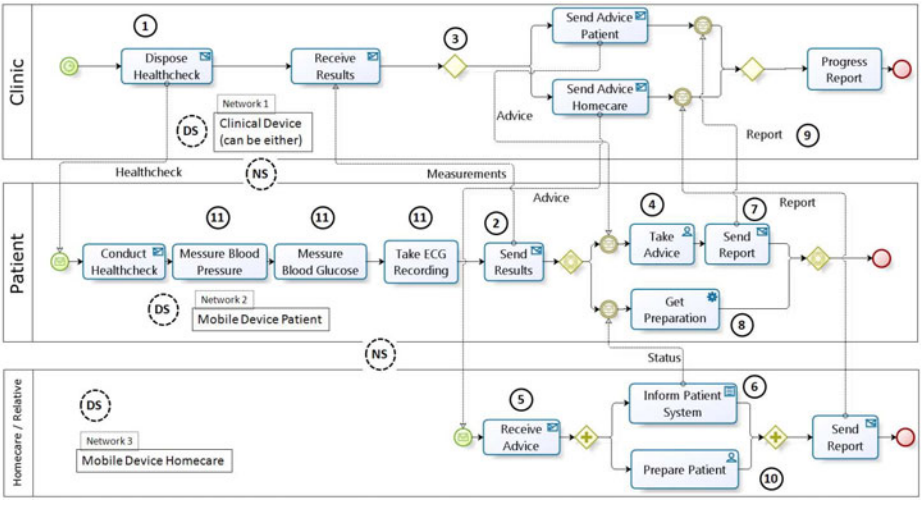

Fig. 2. Healthcare Process with Mobile Patient Assistance

Let us consider the patient-specific process fragment in more detail: While the patient stays at home, he gets a message from the clinic through his mobile device. Then he measures and collects the requested data being assisted by the process fragment running on his mobile device. Following this, results are sent back to the clinic ②, which then decides about next steps ③. Ideally, everything is OK and no special actions concerning the patient are required. In this case a message is sent back to the mobile device containing information about the patient's medication ④. In case of problems, the clinic might send an alternative message with information about further or special treatment to homecare ⑤ (either provided by a professional service or by relatives of the patient). In the latter case, an additional process fragment is started on the mobile device of the person being responsible for homecare. Its execution then has to be synchronized with the one running on the patient's mobile device. Finally, either the process running on the mobile device of homecare or the one of the patient sends back a report ⑦ to the clinic, before the process is finished.

3 Requirements

Table 1 gives an overview of characteristic requirements raised when running processes and process fragments on mobile devices. We elicitated these requirements when analyzing different process scenarios from domains like healthcare, sales, logistics, and emergency management.

Table 1. Requirements Overview

Requirements for Enabling Mobile Processes	
Category I: Process Implementation	**Category III: Runtime**
R_1 ***(Partitioning):*** It must be possible to partition a global process model and to allocate the resulting fragments on mobile devices and (distributed) process servers.	R_7 ***(Synchronization):*** When running fragments on distributed machines and mobile devices respectively, their execution must be correctly synchronized. Further, messages must be exchanged in a reliable and secure way. A mechanism for queuing messages is needed.
R_2 ***(Soundness):*** Soundness (i.e., correct execution behavior) of both the global process (i.e., the process choreography) and the process fragments (i.e., process orchestrations) needs to be ensured.	R_8 ***(Adaptations):*** Both the global process model and its constituting fragments might have to be adapted during runtime (e.g., to deal with exceptions). Such adaptations must not lead to inconsistencies or errors.
R_3 ***(Lifecycle):*** Full lifecycle support for the global process as well as its constituting process fragments is needed.	R_9 ***(Hand-Over):*** Handing over running process fragments between mobile devices must be possible; i.e., to move a running process fragment from the current device to a new one and to restart its execution from a safe point.
Category II: Supporting Infrastructure	R_{10} ***(Robustness/Self-Healing):*** Mobile processes need to be executed in a robust manner. Self-healing mechanisms for reacting on unforeseen events (e.g., device failures) are required.
R_4 ***(Configuration Management):*** The infrastructure must provide user-friendly mechanisms for configuring process and service deployment, device characteristics, and application components.	
R_5 ***(Connection Management):*** Physical problems like broken connections or mal-functioning devices need to be handled by the supporting infrastructure, but without burdening users.	R_{11} ***(Real-Time Data):*** A mobile process must be able to combine gathered real-time data from sensors (e.g., blood pressure of a particular patient). Furthermore, failure data (e.g., a sensor does not deliver data) and QoS parameters (e.g., execution time or cost) need to be considered.
R_6 ***(User Management):*** The infrastructure must enable user changes on the mobile device without necessitating users to be aware of the location and characteristics of the mobile device (or surrounding sensors). Additionally, the infrastructure must provide context-aware mechanisms to decide which user change is possible and suitable. If a user must skip his activity (e.g., due to an emergency call), the infrastructure must cope with these unforeseen situations as well. Therefore a categorization of user changes must be provided by the infrastructure.	R_{12} ***(Events):*** During process execution events have to be gathered, filtered, aggregated, and processed. In addition, a categorization of events is needed for enabling better configuration support for devices and processes.
	R_{13} ***(Actor Assignments):*** Actor assignments for process steps and process fragments respectively must be resolvable at any point in time during process execution. Furthermore, actor assignments of upcoming process steps should be resolved in advance in order to pursue an optimal migration strategy.

Reconsider our example from Section 2. In this simple scenario three process fragments exist whose execution needs to be synchronized among the three parties. In this context, the overall system architecture must be able to cope with communication problems, device failures, and so forth. In Fig. 2 the pictograms with label DS and NS indicate potential network and device switches within the overall process choreography. For example, assume that the mobile device of the patient or its connection with the clinic fail. Consequently, the clinic has no information about the status of the patient, but only knows that the network connection is broken. Such failure scenarios must be adequately covered by the architecture.

4 MARPLE - Overview and Architecture

We first describe the *MARPLE architecture* (cf. Fig. 3) whose two core components are the *MARPLE Mobile Engine* and the *MARPLE Mediation Center*. We focus on those parts of the *MARPLE architecture* that are fully implemented and relevant in the context of our application scenario. Other components of MARPLE are only sketched and will be subject of future papers.

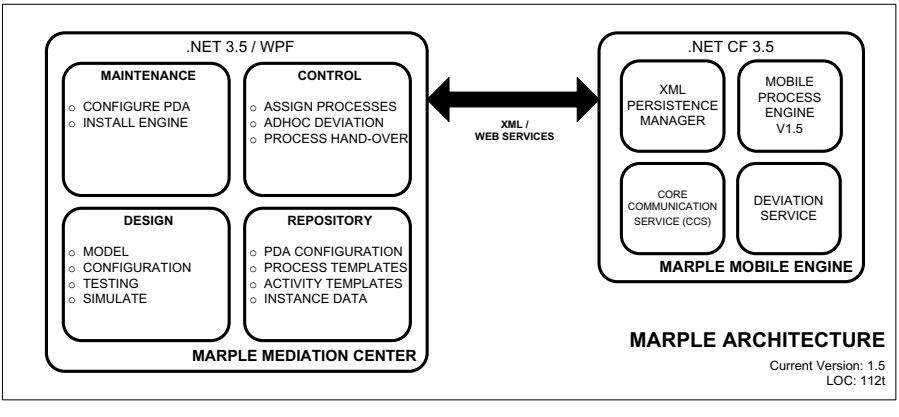

Fig. 3. MARPLE Architecture

4.1 MARPLE Mediation Center

The *MARPLE Mediation Center* (cf. Fig. 3) consists of four major parts. First, its *Maintenance* component enables configuration of mobile devices such that they can be used for mobile process support. Second, the *Control Unit* enables users to assign executable process fragments to mobile devices, to enact them on the mobile device, to invoke user forms, web services and other kinds of activity components during process fragment execution, and to define ad-hoc deviations from the prescribed process logic if required. Another fundamental feature of the MARPLE *Control Unit* is its ability to hand-over running process fragment instances from one mobile device to another; e.g., if a patient wants to switch his

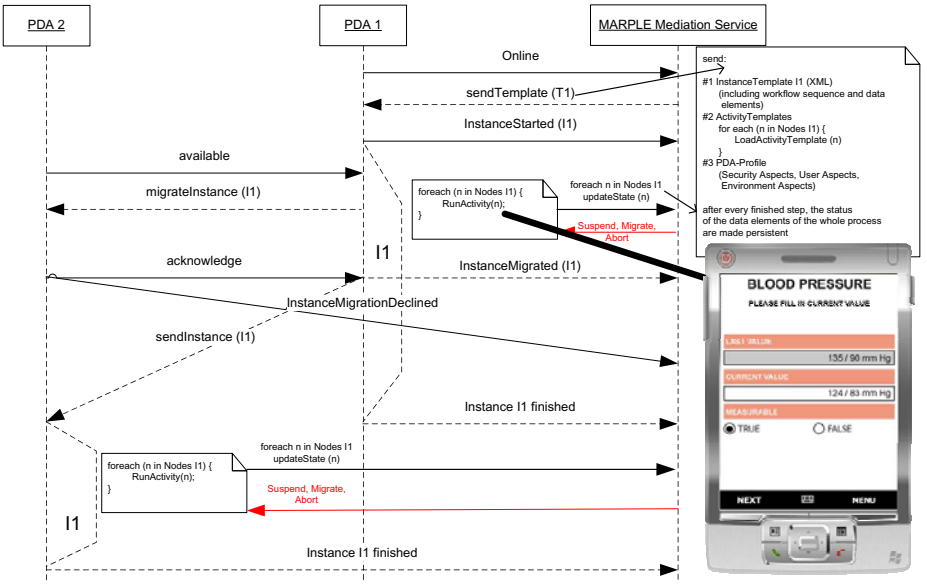

Fig. 4. MARPLE: Interaction Sequence

device or a homecare person wants to hand over his process fragment to someone else. Like in the case of ad-hoc changes, such hand-over can be initiated locally by the owner of the mobile device as well as remotely by authorized users via the *Control Unit*.

Fig. 4 exemplarily illustrates possible interactions between the *MARPLE Mediation Service* and two mobile devices. Initially, only one mobile device is involved in the interaction. Then a second device is added. Following this, the process instance running on the first mobile device is handed over to the newly introduced one, e.g., due to connection problems with the first device, better technical features of the new one, or needed hand-over of tasks. As mentioned this handing over can be triggered either by the *MARPLE Mediation Center* or by the device owners. During process execution, the *Control Unit* may suspend, resume, abort, and monitor running processes. Finally, *MARPLE Mobile Process Engine V1.5* logs process events using the *Persistence Manager* (cf. Fig. 3). Another important aspect that emerges when integrating end-users and process management technology on mobile devices concerns usability. We provide advanced support for deploying the *MARPLE* process engine to mobile devices. The *Mediation Center* makes the engine available through web services, which enables users to deploy it to their mobile devices.

4.2 MARPLE Modeler

Another important component of the *MARPLE Mediation Center* is its *Modeler*, which adopts the basic correctness principles and verification procedures

we developed in ADEPT [4]. Additionally, it provides features for partitioning global processes into several process fragments, for allocating the resulting process fragments to different machines (e.g., mobiles devices), and for linking process activities with activity components (cf. Sect. 4.4). Consider again our example from Section 2. Using *MARPLE Modeler*, for instance, the process fragment, coordinating data collection steps and running on the mobile device of the patient, can be defined (see Fig. 5). The *MARPLE Modeler* is subdivided into three areas: The toolbar on the left side ⓐ depicts available modeling elements: `Activity Nodes` and `Connectors`, `Data Elements`, and `Activity Templates`. ⓑ depicts the main area of the *Modeler*: Basically, the process model can be created through drag & drop operations, which copy elements from the toolbar to the design area. Finally, the top toolbar ⓒ provides useful modeling functions (e.g. for layouting the models). The *MARPLE Modeler* is based on the *Windows Presentation Foundation* as part of the *Microsoft .NET Framework*. The process elements depicted within area ⓑ are defined using XML. The editor stores modeled process fragments via XSLT-transformation according to the internal format required by the *MARPLE Repository*.

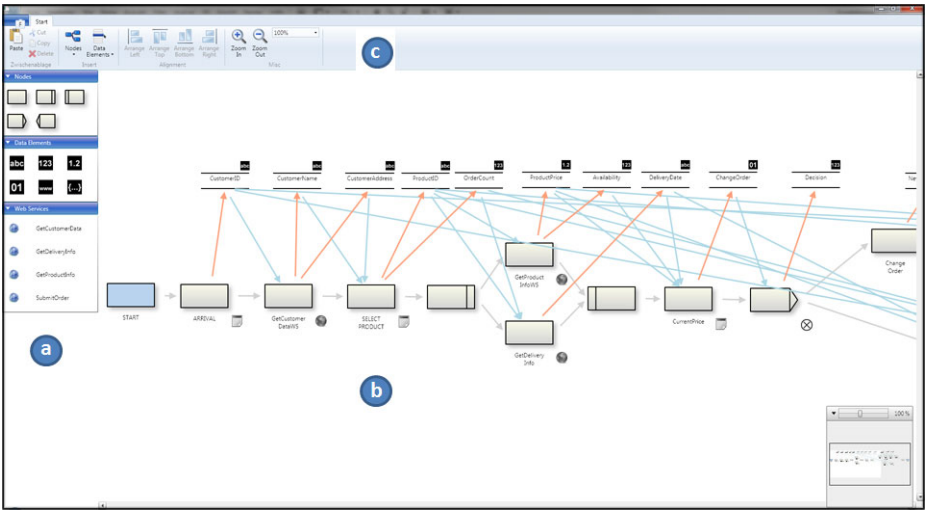

Fig. 5. MARPLE Modeler

Configurable user forms: User-friendliness is one major aspect for mobile applications. As we learned in our healthcare case studies, physicians and nurses rapidly become discouraged when being confronted with inappropriate user interfaces. We therefore integrated a form editor with the *MARPLE Modeler*. It comprises features to cope with the problem of limited screen size on mobile devices (e.g., if too many data elements shall be displayed in one form, this can be splitted into two forms). In addition, to meet context demands (like in emergency situations), we added specific features for enabling more sophisticated

user interactions. For example, instead of entering data manually, a physician or nurse may enter it only through pre-defined controls.

4.3 MARPLE Mobile Engine

When developing the *MARPLE Mobile Process Engine*, we re-used basic concepts and design principles of the ADEPT process management technology, which we had developed during the last decade [3]. In particular, we adopted the ADEPT process meta model, applied its fundamental correctness notions as well as verification procedures, and support flexible and adaptive process enactment on the mobile device as well. The latter enables dynamic structural adaptations of process fragment instances running on the mobile device; e.g., to cope with contextual changes in the environment or exceptional situations. Basic to the support of such ad-hoc changes is the *MARPLE Mobile Deviation Service*. Despite these commonalities with ADEPT it is noteworthy that we provide a complete new implementation of the kernel of the *MARPLE Mobile Process Engine* in order to meet performance requirements of mobile scenarios and to cope with issues specific to mobile processes (e.g., broken connections and limited GUIs). In particular, the implementation framework on which MARPLE is based differs from the one used in ADEPT - ADEPT relies on JAVA, while MARPLE is based on *.NET Compact Framework*. In the following we describe selected services of the *MARPLE Mobile Process Engine* in more detail.

Configuring mobile devices: When a mobile device is added to the *MARPLE* environment, it first needs to be equipped with basic software services. Amongst others, this includes *Core Communication Services (CCS)* as basic pillar of the *MARPLE Mobile Process Engine V1.5*. Thereby, we apply a light-weight approach; i.e., services initially not needed are not uploaded to the device. Following this, the mobile device can connect to the *MARPLE Mediation Center*. When starting the *MARPLE* configuration procedure on the mobile device using the *MARPLE Mediation Center*, CCS dynamically loads the *MARPLE Mobile Process Engine*, the *XML Persistence Manager*, and selected *process* as well as *activity templates* (see also Section 4.4) to this device. The *XML Persistence Manager* fulfills two functions: First, it allows (de)serializing process models in order to exchange them between the *MARPLE Mediation Center* and the mobile devices. Second, it enables the storage of process models on mobile devices.

Enactment of processes on mobile devices: MARPLE allows to deploy process models and fragments, respectively, to mobile devices and to create corresponding instances. In the first version of the *MARPLE Mobile Process Engine* we only considered the execution patterns *sequence* and *conditional* routing. Later we added support for the *parallel execution* of process activities on mobile devices, which is required for performing "background"

Fig. 6. Parallel Execution

activities, e.g., database operations or web service calls (cf. Fig. 6). Parallel processing of activities on mobile devices can be also utilized to reduce throughput times and error probabilities as well as to enable better monitoring. Due to limited screen size of mobile devices, however, concurrent execution of activities on mobile devices should be more restrictive than on normal process servers. Displaying multi-windows simultaneously, for example, is not meaningful. For process fragments running on mobile devices, *MARPLE* therefore disallows the concurrent processing of forms referring to different parallel branches. As example consider Step ⑥ of our healthcare scenario from Section 2. Here messages are sent to the patient device in parallel to the form-based process step *Prepare Patient* (as performed by homecare). Parallel processing is very helpful in the given context to foster monitoring. Let us consider that the process instance of the patient shall be monitored by a clinical doctor and homecare is involved. The homecare process fragment sends (cf. Fig. 2) in parallel information back to the activity *Prepare Patient*. This enables the clinician to already monitor the activity *Prepare Patient* during execution without waiting of its finishing. Especially, if the activity has a long duration, this information could be very important.

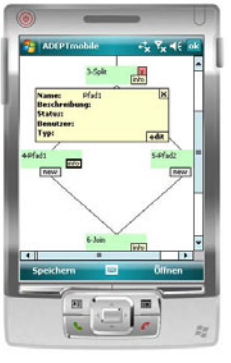

Fig. 7. Ad-hoc Deviations

Ad-hoc deviations during mobile process enactment: The ability to deviate from pre-modeled process templates during runtime is crucial for mobile process support. Both changes in the local environment (e.g., blood pressure of a patient increases) and changes in the infrastructure or global environment (e.g., a particular sensor is not measuring data) might require ad-hoc changes of a mobile process. For example, if results are missing during the examination of a patient, but are urgently needed, medical tests or procedures may have to be dynamically added. To be able to adjust a particular instance of the mobile process correctly and quickly during runtime, we implemented a sub-component of the *MARPLE Mobile Process Engine* that enables users to locally adapt processes running on the mobile device. Thereby, *MARPLE* displays the process to the user and offers a number of context-enabled changes (cf. Fig. 7). The offered set is based on context information provided locally through the *MARPLE Mobile Process Engine* (e.g., regarding the current status of the executed fragment and the role of the respective user). Change operations implemented so far include the insertion and deletion of single activities. In the current implementation only human tasks (i.e., form-based activities) can be added or deleted locally. In our future research, remote adaptation of process fragments running on mobile devices will become a topic of interest as well.

Integration with calendar systems: On the one hand many companies use collaboration tools like e-mail clients and calendar systems with built-in schedule capabilities. On the other hand, mobile scenarios like the one from Section 2 are

usually linked with many related time events, of which some can be related to calendar items. Regarding our healthcare scenario, one important time event at the clinic concerns the triggering of the whole process. Assume that doctors have to trigger such events manually. Then it would be highly welcomed by them if the activity had been linked to their personal calendar. Amongst others this would reduce omission errors. Furthermore, if the activity has to be postponed, the infrastructure can explicitly cope with this situation. We therefore interlinked the *MARPLE Architecture* with *Microsoft Outlook*. With the help of the *MARPLE Mediation Center* the interaction between the two tools can be configured. The *MARPLE Mobile Process Engine* then starts process instances on the mobile device accordingly to pre-defined calendar items.

4.4 Activity Templates

In *MARPLE*, single process steps can be implemented based on reusable *Activity Templates* (cf. Fig. 8). These encapsulate pre-manufactured application components.

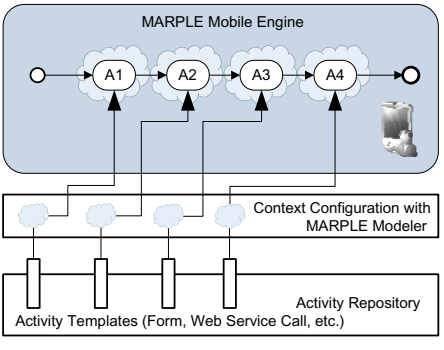

Fig. 8. Activity Templates

Their implementation must be a *dll* component of the *.NET Compact Framework*. We store available *activity templates* in the *MARPLE Repository*. Based on the latter *activity templates* are linked to process steps at design time using the *MARPLE Modeler*. Thereby, they have to be configured to the given context. *Activity Templates* further must be provided with a number of configure parameters depending on the given type (e.g., concrete parameters for a web service call like the message format or URI). During process execution, the *MARPLE Mobile Process Engine* loads the *activity templates* incrementally when activating process steps (cf. Fig. 8).

As standard activity templates MARPLE provides support for invoking user forms and web services as well as for evaluating transition (xor) conditions. Based on respective core activity templates, simple mobile scenarios like the one from Section 2 can be implemented. As example consider activity ⑪ in our scenario, where the user interacts with a form to enter his blood pressure value.

From other mobile process scenarios we implemented with *MARPLE*, we revealed the need for additional kinds of activity templates covering locations awareness, database connectivity, and barcode management.

Database activity template: When investigating mobile scenarios from the sales domain we started considering database functions for mobile process

management. We talked to sales representatives from the insurance domain and collected characteristic requirements. Most prominently, they claimed that data management in respect to their mobile device and applications is not satisfactory. An appropriate data management component needs to cover three aspects. First, required customer data should be automatically transferred to the mobile device in advance (e.g., by utilizing information about customer appointments). Second, for a process running on the mobile device and its interlinked processes in the backend, data consistency must be ensured. Third, if no connection can be made during the meeting with a customer, correct re-transfer of data into the backend system should be provided. All these steps require a database connection in order to perform respective queries. To meet this requirement, we implemented a database activity template. Since parallel processing of activities on the mobile device is possible in *MARPLE* any activity based on the database template can frontload data for later process activities. For example, while the sales manager runs form-based activities for entering customer data, certain data management steps can be executed in parallel. Additionally, data collected via the mobile device can be transferred back to the backend system at the earliest point in time. Consequently, this data can be used for recovery purposes, e.g., when an abnormal situation (e.g., broken device) occurs.

Fig. 9. Barcode Reading

Activity templates enabling sensor awareness: One major goal of *MARPLE* is to provide personalized and situation-aware mobile processes. For reaching this objective the integration of real-time data as provided by sensors (e.g., blood pressure of a particular patient can be gathered via *Bluetooth*) with mobile processes is needed. Our healthcare scenarios indicated that often patient data can be gathered via sensors (e.g., measuring pulse, respiration, blood pressure, and body temperature). The gathered data then has to be processed, aggregated, and interpreted considering the given context. Due to lack of process-awareness, effective processing of this data is usually limited. We further learned from our clinical scenarios that sensor-gathered data is usually only visualized based on specialized applications provided by the sensor vendor. In order to utilize such context data (e.g., a medical parameter has exceeded its threshold multiple times) we are developing *Activity Templates* that enable sensor-awareness. We have already implemented a *Barcode Activity Template*, which enables users to scan data collectable from packages (e.g., drug packages) (cf. Fig. 9). As example consider the clinical admission of patients. The barcode of medications brought along by the patients to the clinic could be scanned and archived in the patient record. In the context of another project in the field of airline catering, we are currently working on a *RFID Activity Template*. This template shall enable users to read RFID tags and utilize gathered data within the mobile processes respectively.

Map Activity Template: Mobility implies a larger extent of flexibility on the one hand, and a lot more sources of problems (e.g. no network connection, limited battery life) on the other hand. This must be considered in the context of mobile process execution as well. Location awareness is one feature we should add when compared to classical workflows. *MARPLE* therefore supports a *location activity template*, which can display either the current user location, if a GPS module exists, or any location based on given coordinates (cf. Fig. 10). Regarding our sample process this activity template can be used to realize Step ⑤ in order to guide the homecare service when visiting the patient. Furthermore, context information about location can be used to provide context-specific assistance for users (e.g., by only displaying those instances to them which are relevant in the given context). Additionally, coordinates of the mobile devices can be periodically submitted to the *Mediation Center*. This way resources (e.g. workload, errands) can be managed in a more efficient way.

Fig. 10. Street Map Integration

4.5 Summary of the MARPLE Features

Table 2 summarizes core features of MARPLE components:

Table 2. Marple Architecture Feature Overview

Component	Services	Description
Mediation Center	Design	Modeling processes via “drag&drop"-operations and customization facilities.
	Control	Allocate processes on mobile devices and remotely monitor running instances. Perform instance migrations to other devices or ad-hoc deviations if needed.
	Repository	Save templates, process instances, and configurations.
	Maintenance	Manage mobile devices and configurations.
Mobile Engine	Process Enactment	Execute processes on a mobile device. Supported activity templates are: forms, web services, database access, barcode scanning & processing, and location services.
	Process Deviations and Visualization	User interface for inserting new activities and for skipping existing ones.
	Persistency	Persists current instance data on mobile device and additionally stores the process template as well as activity templates on the mobile device.
	Communication	Built-in server for handling the overall communication with the *Mediation Center*.
Integration with Calendar System (Outlook)	Scheduling	Assign appointments to process steps, i.e., link process instances with calendar items.

5 Realizing the Healthcare Scenario with MARPLE

We revisit our scenario from Section 2 and show how it can be realized using MARPLE. Fig. 11 depicts the user interface of the *MARPLE Mediation Center*

and shows a concrete process instance running on a mobile device as it can be monitored with the *MARPLE Mediation Center*. Note that this perspective displays both the current status of the mobile process and the data values collected during process execution (see ⑦). Obviously, this is exactly the information a medical professional needs when remotely monitoring the patient process.

How does the patient process fragment look like in *MARPLE*: Fig. 11 depicts a part of this model together with instance-specific markings. Activity ② constitutes a *receive* activity which is waiting for an incoming message requesting a health check. The following three activities constitute data collection steps, which are either implemented as user forms or sensing activities. The blood pressure, in turn, is gathered via a bluetooth activity template which is linked with the blood pressure measurement machine. Blood glucose and ECG recordings are entered by users via form-based activities; i.e., the user of the mobile device gets respective requests in his worklist and then has to fill in the two forms (e.g. see the PDA display in Fig. 4).

Following data collection, activity ③ is automatically executed. It invokes a web service at the clinic to report about measured results (e.g., to add them to the electronic patient record). The subsequent activity ④ then waits until a message is received either from the clinic or from homecare. The administrator toolbar on the left side of Fig. 11 ⑧ displays available functions for managing process templates, users, mobile devices, and mobile device settings. Further, ⑥ displays the list of currently released process templates, which can be assigned to registered mobile devices.

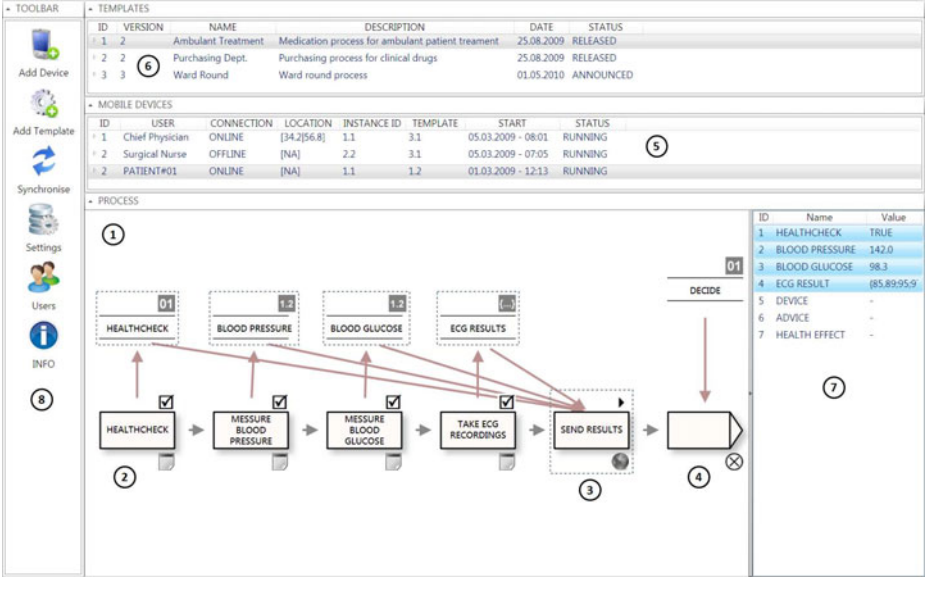

Fig. 11. MARPLE: Mediation Center

6 Evaluation of Mobile Process Support with MARPLE

We applied MARPLE to process scenarios from healthcare, logistics and sales, which we implemented using the *MARPLE Architecture*. We learned that the provision of a light-weight process engine for mobile devices is essential in order to realize human-centric processes. Furthermore, our prototypical implementations revealed additional demands we picked up in this paper. Table 3 gives an overview of the considered scenarios and relates them to *MARPLE* features.

Table 3. Marple Features Usage

MARPLE Features / Scenarios		Scenario 1 *Medical Telecare*	Scenario 2 *Clinical Ward Round*	Scenario 3 *Ambulance Service*	Scenario 4 *Insurance Sales (CRM)*	Scenario 5 *Airline Catering*
Activity Templates	Barcode	o	x	o	-	x
	Database	o	x	-	f	x
	Map	o	-	x	x	f
	User Form	x	x	x	x	f
	Web Service	o	x	o	f	x
Calendar Integration		f	o	-	o	f
Configurable User Forms		f	f	f	o	f
Local Process Deviations		o	f	o	o	f
Parallel Activities		o	x	f	f	x
Process Migration		o	f	o	o	o
Process Monitoring		x	o	o	o	x

(-) : not needed **(o)** : rarely needed **(f)** : frequently needed **(x)** : heavily needed

7 Related Work

In literature, we can find several approaches which focus on logical models for mobile processes on the one hand and approaches addressing architectural and implementation issues of light-weight process engines on the other hand. Regarding the first category, for example, several approaches exist for the partitioning of BPEL processes [5,6,7,8]. A similar approach has been suggested in the context of $ADEPT_{distribution}$ [9,10]. However, none of the two approaches has provided an architecture for mobile process support as MARPLE does. Taking network dynamics as core demand for mobile process engines, several approaches deal with failures and exceptions like broken connections or lack of communication facilities [11,12,13,14]. Respective tools usually apply web service standards and base process execution on BPEL or more specific execution models derived from it. We consider the use of BPEL as process execution language as too low level, particularly if it shall be possible to dynamically evolve or adapt mobile processes during runtime. Instead we provide a process model at a higher level of abstraction that can be adapted by both remote users and users of the mobile device. We further believe that self-healing techniques and migration management will be crucial for mobile processes in the large scale. However, existing approaches dealing with these aspects in process-aware information systems, again focus on

BPEL as execution language [6,15]. The same applies in respect to mobile processes [16]. As described, combining real-time data and domain knowledge raises additional challenges for mobile process management. The MobiHealth project [17], for example, supports context-aware services which allow to integrate sensor data. However, context-awareness is restricted in the sense that execution can only be switched between pre-configured process variants when the context is changing. Generally, approaches allowing to switch between pre-configured process variants can be found in other projects as well [18,16]. However, the fusion of real-time data and domain knowledge with mobile processes has not been considered in a suitable manner yet. Finally, there are approaches focusing on process-aware systems using web services and running on mobile devices [19,7,20]. They enable simple web service flows on mobile devices based on a mobile engine. Approaches only focusing on web service calls, however, restrict the potential of mobile processes in several respects.

8 Summary and Outlook

We introduced the MARPLE approach and described how its core components enable the execution and monitoring of processes (fragments) on mobile devices. Our overall vision is to provide sophisticated mobile process support; i.e., to realize generic process management features including support for process instance changes, process instance migrations, sensor data integration, etc. To foster this vision we base our presented work on core design principles and fundamental concepts we developed in our ADEPT project as well as our $ADEPT_{distribution}$ project [10,21]. In future work we will extend the *MARPLE Modeler* such that it provides sophisticated methods for modeling complex process choreographies including numerous process servers, devices, and actors possibly distributed over many departments. This will include, for example, a methodology for correctly partitioning processes models, for allocating resulting fragments on different machines and devices based on more intelligent allocation techniques, and for synchronizing different process fragments at runtime. In particular, we will adopt and extend concepts from autonomic computing and self-healing systems [22] to cope with the many failure scenarios in connection with distributed and mobile applications. This will be crucial when realizing mobile processes in the large scale.

References

1. Smith, H., Fingar, P.: Business Process Management: The Third Wave. Meghan-Kiffer Press (2003)
2. Bonnici, E., Welch, P.H.: Mobile processes, mobile channels and dynamic systems. In: IEEE Congress on Evolutionary Computation (CEC 2009), pp. 232–239 (2009)
3. Dadam, P., Reichert, M.: The ADEPT project: a decade of research and development for robust and flexible process support - challenges and achievements. Computer Science - Research and Development 23, 81–97 (2009)
4. Reichert, M., Rinderle-Ma, S., Dadam, P.: Flexibility in process-aware information systems. In: Jensen, K., van der Aalst, W.M.P. (eds.) Transactions on Petri Nets and Other Models of Concurrency II. LNCS, vol. 5460, pp. 115–135. Springer, Heidelberg (2009)

5. Baresi, L., Maurino, A., Modafferi, S.: Workflow partitioning in mobile information systems. In: MOBIS 2004, pp. 93–106 (2004)
6. Baresi, L., Guinea, S.: Dynamo and self-healing BPEL compositions. In: Proc. 29th Int. Conf. on Software Engineering, pp. 69–70 (2007)
7. Hahn, K., Schweppe, H.: Exploring transactional service properties for mobile service composition. In: MMS 2009, pp. 39–52 (2009)
8. Schmidt, H., Hauck, F.J.: SAMPROC: middleware for self-adaptive mobile processes in heterogeneous ubiquitous environments. In: Proc. 4th Middleware Doctoral Symposium, New York, NY, USA, pp. 1–6 (2007)
9. Bauer, T., Reichert, M., Dadam, P.: Intra-subnet load balancing in distributed workflow management systems. Int'l Journal of Cooperative Information Systems 12, 205–323 (2003)
10. Reichert, M., Bauer, T.: Supporting ad-hoc changes in distributed workflow management systems. In: Meersman, R., Tari, Z. (eds.) CoopIS 2007. LNCS, vol. 4803, pp. 150–168. Springer, Heidelberg (2007)
11. Kunze, C.P.: DEMAC: A distributed environment for mobility-aware computing. In: Gellersen, H.-W., Want, R., Schmidt, A. (eds.) PERVASIVE 2005. LNCS, vol. 3468, pp. 115–121. Springer, Heidelberg (2005)
12. Hackmann, G., Haitjema, M., Gill, C.: Sliver: A BPEL workflow process execution engine for mobile devices. In: Dan, A., Lamersdorf, W. (eds.) ICSOC 2006. LNCS, vol. 4294, pp. 503–508. Springer, Heidelberg (2006)
13. Schmidt, H., Kapitza, R., Hauck, F.J.: Mobile-process-based ubiquitous computing platform: a blueprint. In: Proc. 1st Workshop on Middleware-Application Interaction, pp. 25–30 (2007)
14. Gerhard, S., Jürgen, M., Erich, S.: Building a modular service oriented workflow engine. In: Int. Conf. on Service-Oriented Comp. and Applications (2009)
15. Baresi, L., Guinea, S., Pasquale, L.: Self-healing BPEL processes with DYNAMO and the JBoss rule engine. In: Int. Workshop on Engineering of Software Services for Pervasive Environments, pp. 11–20 (2007)
16. Zaplata, S., Kottke, K., Meiners, M., Lamersdorf, W.: Towards runtime migration of WS-BPEL processes. In: Dan, A., Gittler, F., Toumani, F. (eds.) ICSOC/ServiceWave 2009. LNCS, vol. 6275, pp. 477–487. Springer, Heidelberg (2010)
17. Jones, V.M., van Halteren, A.T., Dokovski, N.T., Koprinkov, G.T., Peuscher, J., Bults, R.G.A., Konstantas, D., Widya, I.A., Herzog, R.: Mobihealth: mobile services for health professionals. Technical Report TR-CTIT-06-38, Enschede (2006)
18. Hallerbach, A., Bauer, T., Reichert, M.: Capturing Variability in Business Process Models: The Provop Approach. Journal of Software Maintenance and Evolution: Research and Practice 22(6-7), 519–546 (2010)
19. Battista, D., de Leoni, M., De Gaetanis, A., Mecella, M., Pezzullo, A., Russo, A., Saponaro, C.: ROME4EU: A web service-based process-aware system for smart devices. In: Bouguettaya, A., Krueger, I., Margaria, T. (eds.) ICSOC 2008. LNCS, vol. 5364, pp. 726–727. Springer, Heidelberg (2008)
20. Zaplata, S., Dreiling, V., Lamersdorf, W.: Realizing mobile web services for dynamic applications. In: Godart, C., Norbert Gronau, S.S.G.C. (eds.) I3E 2009, pp. 240–254. Springer, Heidelberg (2009)
21. Reichert, M., Bauer, T., Dadam, P.: Flexibility for distributed workflows. In: Handbook of Research on Complex Dynamic Process Management: Techniques for Adaptability in Turbulent Environments, pp. 137–171. IGI Global, Hershey (2009)
22. Gui, C., Mohapatra, P.: SHORT: Self-healing and optimizing routing techniques for mobile ad hoc networks. In: Proc. 4th ACM Int. Symposium on Mobile Ad-hoc Networking & Computing, pp. 279–290 (2003)

A Mashup Tool for Collaborative Engineering of Service-Oriented Enterprise Documents

Nelly Schuster[1], Raffael Stein[2], and Christian Zirpins[2]

[1] FZI Forschungszentrum Informatik, Haid-und-Neu-Straße 10-14,
76131 Karlsruhe Germany
{nschust}@fzi.de
[2] Karlsruhe Institute of Technology (KIT),
76131 Karlsruhe Germany
{raffael.stein,christian.zirpins}@kit.edu

Abstract. Enterprise documents combine the representation of organizational processes and rules with knowledge and data to support human communication in a visual appealing and possibly interactive way. These characteristics are not only beneficial for the support of highly structured and optimized business processes but can be also leveraged to drive ad hoc collaborative innovation processes. However, collaborative authoring requires content consolidation and coordination mechanisms. While IT-supported document engineering for structured and recurring collaboration processes is well established, engineering of documents that emerge and evolve instantaneously still lacks appropriate support. Subsequently we present the MoSaiC tool for collaborative document engineering: MoSaiC supports a new type of service-oriented enterprise documents that are represented as mashups of content creation, transformation and publication services over the Web. This 'living' format supports teams to rapidly regulate and control collaborative activities by a) mapping them to services provided by team members or enterprise systems and b) mashing them to document structure and processing rules in an interactive, intuitive and dynamic way.

Keywords: service-oriented enterprise documents, collaborative document engineering, situational document collaboration.

1 Introduction

Enterprise documents provide a unique way to communicate information in a purpose-optimized (structured, annotated, graphically appealing, legally binding) form of representation in order to collect and share data between human recipients in the course of collaborative work. Sophisticated enterprise documents encapsulate business rules and serve as input to or output of business processes, thus enabling their regulation and enforcement in a highly efficient manner; e.g. utilizing customer relationship, and workflow management systems (WfMS). On the other hand, documents often evolve during ad hoc collaborations between members of a possibly virtual team. Such documents are subject to ad hoc changes as new contents from different sources – humans and software systems – come in. Dependencies between different

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 166–173, 2010.
© Springer-Verlag Berlin Heidelberg 2010

activities of team members and corresponding parts of a document require ad hoc coordination mechanisms for the authors of the document. Examples can be found for instance in IT service management or software engineering, where collaborators have to react on unexpected incidents and collaboratively develop solutions which are captured in interrelated documents [1]. Another example is collaborative research resulting in a joint publication [2].

Several enterprise document engineering approaches [3,4] and various technologies exist to support coordination of structured and recurring business processes (e.g. BPM technologies like WfMS). Other technologies and tools support collaborative creation and evolution of unstructured documents (e.g. CSCW technologies, wikis or other Web-based collaborative writing applications). However, they largely fail to provide effective support for collaborative document engineering that involves ad hoc coordination processes.

In order to address this need, we have developed the MoSaiC approach that firstly realizes a service-oriented way of enterprise document engineering. We model enterprise documents as situational compositions (so called mashups) of human- or software-based content provision, transformation or publication services provided by human collaborators or Web-based software services [5]. Lightweight coordination of collaborating authors and auxiliary Web services is facilitated by interaction rules that are defined as part of the document in a declarative way. Thus, a document is 'alive', reacting to changes of the document itself and its underlying services.

In this paper, we present a Web-based mashup tool and platform that implement our unique collaborative document engineering approach. In Chapter 2 we will sketch the concept of collaborative document mashups. Chapter 3 outlines the architecture of our mashup tool and platform. An exemplary use case is given in Chapter 4. We discuss related work in Chapter 5. Chapter 6 concludes with a summary and outlook.

2 The Concept of Collaborative Document Mashups

Collaborative document engineering requires coordination of team members that jointly work on and evolve ad hoc documents as well as consolidation of content coming from a variety of sources including team members but also Web-based data sources. Content consolidation in document mashups is achieved through representing manual as well as automated activities through software services that are composed into the document structure as so called *contributions*. Document mashups build on a hierarchical structure that represents the logical decomposition of a collaboration goal into tasks that add contributions to a document representing the collaboration result. This structure is defined by one or more coordinating team members (*coordinators*). Subsequently, tasks are bound to document services representing either collaborative activities of team members (*collaborators*) or automated functions of software systems (*robots*) that result in creation (e.g., writing text, drawing diagrams, accessing databases) or transformation (e.g., proof-reading, translating, layout) of document contents of any kind. As soon as a service is bound to a document mashup the service provider becomes a participant (either co-author or robot) of the document collaboration responsible for the associated task.

The coordination of participants in an evolving document mashup builds on *events* and *rules*: each service exposes events indicating state transitions of its underlying resources (i.e. task results) which are consumed and reacted upon by document mashups. An important part of document engineering is to define rules that are triggered by events and lead to corresponding actions thus expressing causal dependencies or temporal constraints between service instances; e.g., a rule may specify that a proof-read service is called by the document mashup as soon as a text-based content service emits an update event. Another example might be the notification of all document services that failed to deliver results at a certain deadline. Coordinators are free to specify any rules that fit the collaboration at hand.

We characterize document mashups as 'living' because they are constantly evolving in terms of structure and contents towards achieving a common goal. To a large extend, this dynamicity is driven by manual interaction of team members with the document: Document engineers evolve and control the document structure and rules. Collaborators create or transform document contents. Additionally, certain aspects of document evolution are controlled by rules that react on events and trigger activities of team members and data sources automatically. As we do not separate phases of design and enforcement, service calls will be made as soon as services are added to a document mashup and each change of the document structure immediately affects the control of the collaboration.

3 The MoSaiC Tool and Platform Architecture

In order to realize our concept of document mashups we have developed a Web-based platform and RIA to support collaborative document engineering. Fig. 1 gives an overview of the architecture. Users access the document collaboration environment through their Web browser. The frontend integrates various graphical drag-and-drop user interface components supporting mashup authoring, service provisioning and participant management use cases:

- Fundamentally, the *user management interface* allows logging into the system or registering as a new user. Subsequently, the *participant management interface* enables users to specify their role in one or more document collaborations. This might include the responsibility of a coordinator to participate in the engineering of the document itself. This might also include the responsibility of a collaborator to carry out certain activities within a collaboration process and provide associated document services manually. As a variant, users might delegate responsibility to a robot that provides contributions by means of automated Web services (e.g. to integrate contents from Flickr).
- Using the *document mashup editor*, coordinators carry out the main document engineering tasks including design and management of document mashups. Management includes administrating a repository of living documents as well as state control of individual mashups which happens in sync with an ongoing design process. To this end the editor represents and visualizes document mashups from different perspectives of structure, responsibilities and document content. In the structure perspective, engineers design the logical document structure on a canvas, e.g. a research paper split into parts of the research methodology as well as result

tables and diagrams. The editor offers lists of content, transformation and publication service types which can be dragged and dropped on the canvas in order to append instances of these service types to the mashup. The coordinator can specify roles linking to participants and assign these roles to the services in the mashup. Furthermore, the editor allows definition and management of rules expressing causal and temporal constraints as well as manual control of actions. The responsibilities perspective shows a list of all activities which are involved and allows the coordinator to keep track of them. The document content perspective displays the current content of the document.

– In order to support collaborators of a document mashup in providing services, the platform offers a *human service provision editor.* This interface allows users to receive service calls coming from a document mashup management system (the same or any other), edit and persist appropriate content, and communicate it back to the document mashup manager. However, there are also other potential ways for collaborators to connect like an email bridge or word processor plug-ins that enable them to handle document service calls.

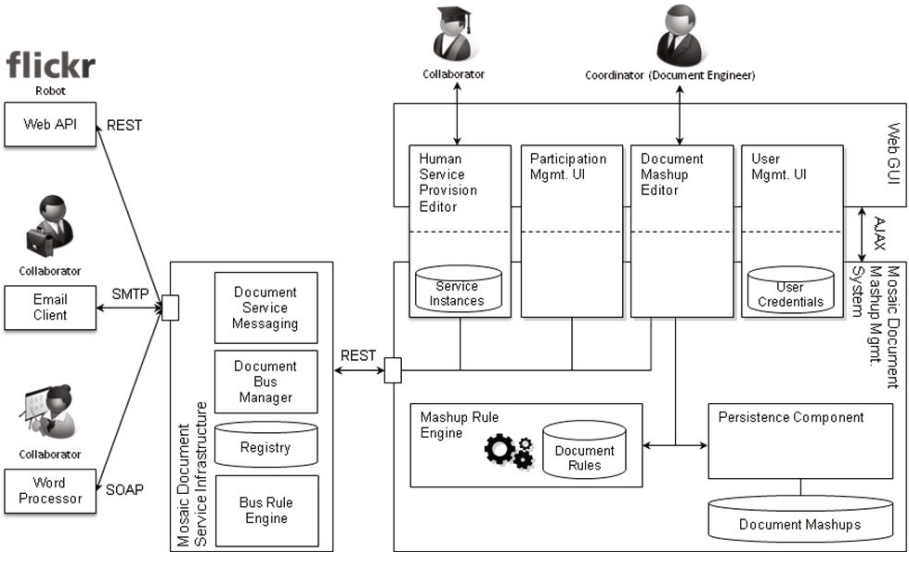

Fig. 1. Architectural Overview of the MoSaiC Mashup Tool and Platform

Mashup logic is enforced by the MoSaiC *document mashup management system* which might run on an enterprise application server or in the cloud. The frontend uses AJAX calls to communicate with the application and persistence logic components. The *persistence component* stores all mashups including structure, tasks, rules, services, participants, roles, contents and control state. It keeps track of mashup and service updates and stores them as separate versions. A *mashup rule engine* drives automatic enforcement of mashup logic. It is able to process complex events and is pre-configured with base rules, which are essential for the enforcement of mashups. Rules follow the event-condition-action (ECA) principle: the engine listens for events

indicating changes of the mashup or its services, checks conditions and takes actions like calling another service. E.g. one of the base rules declares that if a user gets assigned as collaborator, a 'create' call is sent out to its document service. In addition, coordinators specify rules that are mashup-specific. E.g. the above base rule might be changed to express a causal dependency.

The document mashup management system uses a *document service infrastructure* for event-based interaction between services and document mashups. This infrastructure might be deployed independently and is accessed through RESTful service interfaces. Fundamentally, the infrastructure maintains a *registry,* which stores information about existing mashups, document services and providers. The optional *document service messaging component* offers capabilities to route and queue service messages. Service calls generally conform to a uniform RESTful interface and an asynchronous interaction protocol that allows requesting the creation of a new service as well as getting, updating or deleting the content of an existing one. A dedicated *bus rule engine* allows for a specific type of rules that directly affect the interaction between services. Enforcing rules in the bus promises more agile interactions since messages skip the document mashup management system if possible. An example is the automatic routing of certain text content through a translation service before delivering it to the document mashup management system.

4 An Application Scenario of Scientific Publication

We will now illustrate some concepts of document collaboration with MoSaiC by means of a familiar application scenario: collaborative research leading to a joint publication. In research, collaborative work enables researchers to address more complex problems and benefit from mutual inspiration as well as synergy effects. The results are in most cases captured by means of jointly written research paper, where the individuals provide different parts like texts, pictures and references of a specific section describing a partly solution of the complex problem. Fundamentally the collaborators need to coordinate in order to discuss the concepts, structure the document, provide interrelated contents and proofread the results thereby checking for completeness, correctness and readability. Fig. 2 shows the experimental GUI of our tool used to exemplary engineer a scientific publication.

In our scenario, a researcher, who plays a coordinator role within the research team and therefore also in the document mashup, has already started creating a research paper by defining the structure of the document in the mashup editor structure perspective (right hand side of Fig. 2). The coordinator drags several document elements for different parts, texts and figures from a list to the mashup canvas. Furthermore, he adds elements for an abstract and bibliography. Whenever he drops an element on the canvas, the change in the mashup structure is immediately persisted.

In the document service list, the coordinator identifies a layout service that is able to format mashups into an appropriate format and a submission service that uploads a formatted document to the conference server. For both services there is a robot that he can add to the mashup. Furthermore, he defines a rule that the mashup is formatted by the layout service when all content elements are in final state. Afterwards a final

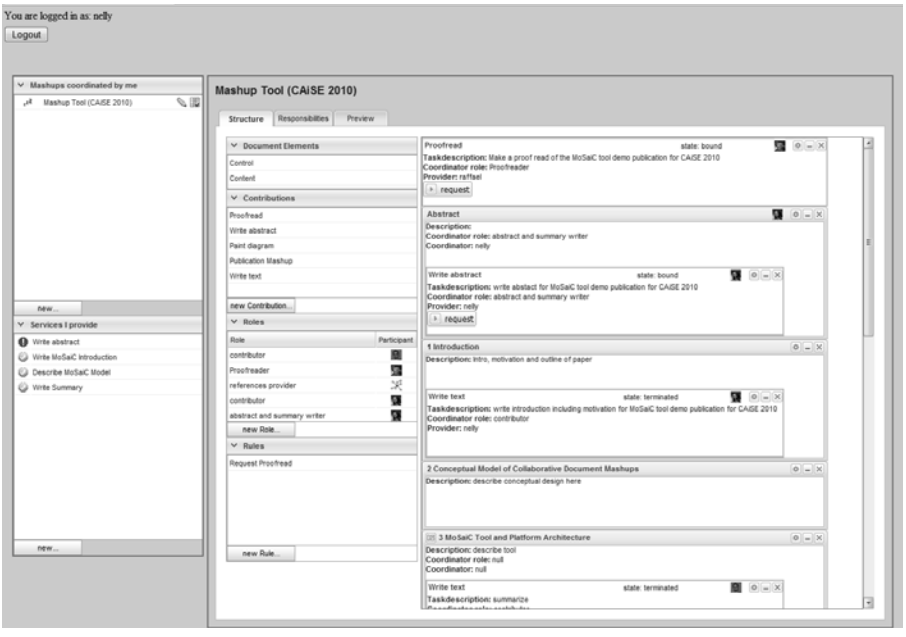

Fig. 2. Screenshot of the Mashup Editor

proof-read by the first author is required and, in case the proof-read is ok, the submission service uploads the document to the conference server. All rules are stored within the mashup and within the rule engine.

Having specified the rules, the coordinator defines roles like proofreader or contributor and uses the provider list to find his collaborators. He discovers Carol, who is a member of the collaborative team. The coordinator assigns her to the role responsible for providing the introduction and assigns this role to the introduction service.

Now the mashup comes to life. Carol receives a notification that she has been assigned to write this paragraph. The task is added to her personal to-do list, which is presented to her in the human service provision editor after logging into the mashup tool (bottom left of Fig. 2). In the following weeks, the team does several changes to the initial mashup to evolve the paper in terms of structure, tasks and, of course, contents. At one point in time, Carol logs into the system, pastes the introduction text into the human service provision editor, marks its state as final and submits it. The rule engine observes the state change and triggers a rule, which declares that as soon as all section elements are in final state, the mashup is to be sent to the layout service. After the layout service returned a formatted document, the first author is requested to proof-read. A reply indicating success triggers the final action: the formatted document is sent out by the submission service.

5 Related Work

MoSaiC adopts concepts of collaborative document engineering and writing based on service mashups for ad hoc composition of mostly human-based software services.

Collaborative document engineering has a long research history, which resulted in various prototypes and products. A prominent example is Google Docs (http://docs.google.com/) that allows collaborative creation of rich text documents. However, these tools do not consider documents as re-active to events. Furthermore, they do not support assignment of activities to providers or integration of automated services. Thus, lightweight ad hoc processes cannot be specified or enforced. Another related technology coming closer to our approach is Google Wave (http://wave.google.com/). A wave is a collaboration of participants based on XML documents consisting of wavelets. This is similar to the composition of document services in a mashup. Waves might include automated robots that are comparable to our automated document services. However, there is no way to define interaction rules based on events and to re-use wavelets in other waves.

A study of tools for collaborative writing, including early Wiki software, is presented in [6]. The study shows, that the idea of splitting a document into fragments for different authors and propagating updates of these fragments to other authors is not new. However, we did not find a tool which supports coordination of the authors or fragments through rules or the integration of different sources from the Web.

The term document engineering as used in [3] describes the analysis and design of documents and rules which are input to business processes or serve as their interfaces. This has similarities to our approach but focuses on structured, recurring processes. There is no concept of document composition like in document mashups that would support collaborative or active documents and ad hoc processes. Interactive Web documents [6] define a REST protocol and format for Web based documents that include data and behavior. However, although they mention a prototype, they do not show how these documents can be collaboratively authored and coordinated.

Mashup technologies have recently gained broad attention from industry [8] and are increasingly addressed by academic research. The mashup paradigm emphasizes user-driven composition of situational apps from Web-based content and services. Several mashup research tools and products exist. An example is IBM Mashup Center (http://www-01.ibm.com/software/info/mashup-center/) that lets users compose widgets which reference services. Based on the description of which events a widget might expose, a mashup developer can add rules that route data between widgets. However, the mashup is still an application and cannot be used as an active document.

6 Summary and Outlook

Shaping and driving ad hoc collaboration processes by means of enterprise document engineering promises to enhance productivity and foster innovation but also poses substantial requirements on flexibility of document structure and dynamicity of contents. Respective documents need to reflect progressing states of multiple collaborators and render their visual representation from various sources. Simultaneously, collaborators need reliable tools allowing them to evolve the structure of collaborative documents in highly interactive yet predictable ways. Our approach of document mashups establishes a basis to meet these. Document mashups support interactive specification of the structure and collaborative regulation of a shared document in a flexible declarative way on the fly. Document services facilitate the integration of dynamic content from various collaborators.

In this paper, we have presented a mashup tool and platform for collaborative engineering of service-oriented enterprise documents. We have briefly outlined our approach of collaborative document mashups and demonstrated how to realize it by means of contemporary service-oriented infrastructure technologies including RESTful Web services, complex event processing and rich Web 2.0 Internet applications. Aiming for a practical perspective to present our work, we have illustrated our general approach and the operation of our tool by means of an application scenario coming from the familiar area of scientific collaboration.

For future work we plan to underpin and evolve the current basis of collaborative document mashups in various directions. At the moment we are thoroughly evaluating our prototype platform by means of case study experiments. Simultaneously we are extending the mashup engineering methodology by formal verification of rule declarations and service value network analysis. We further plan to extend document mashups with versioning information that enable traceability of changes and lead to more transparency. Finally, we aim to study document interaction patterns for common control structures (like, e.g., the 'four-eye principle') for the specific case of situational document collaboration. We intend to provide our findings as reusable design patterns for ad hoc development of document mashups.

References

1. Schuster, N., Zirpins, C., Tai, S., Battle, S., Heuer, N.: A service-oriented approach to document-centric situational collaboration processes. In: Reddy, S. (ed.) WETICE, pp. 221–226. IEEE Comp. Society, Los Alamitos (2009)
2. Marchese, M., Giunchiglia, F., Casati, F.: Liquid publications: Scientific publications meet the web. Technical report DIT-07-073, University of Trento, Department of Information Engineering and Computer Science (2007)
3. Glushko, R.J., McGrath, T.: Document engineering: analyzing and designing documents for business informatics & web services. The MIT Press, Cambridge (2008)
4. Hull, R.: Artifact-Centric Business Process Models: Brief Survey of Research Results and Challenges. In: Meersman, R., Tari, Z. (eds.) OTM 2008, Part II. LNCS, vol. 5332, pp. 1152–1163. Springer, Heidelberg (2008)
5. Schuster, N., Zirpins, C., Schwuchow, M., Battle, S., Tai, S.: The MoSaiC Model and Architecture for Service-Oriented Enterprise Document Mashups. In: 3rd International Workshop on Web APIs and Services Mashups (Mashups 2009), OOPSLA. ACM, New York (2009)
6. Noël, S., Robert, J.-M.: How the Web is used to support collaborative writing. Behaviour & IT 22(4), 245–262 (2003)
7. Boyer, J.M., Wiecha, C., Akolkar, R.P.: A REST protocol and composite format for interactive web documents. In: ACM Symposium on Document Engineering, pp. 139–148. ACM, New York (2009)
8. Hoyer, V., Fischer, M.: Market overview of enterprise mashup tools. In: Bouguettaya, A., Krüger, I., Margaria, T. (eds.) ICSOC 2008. LNCS, vol. 5364, pp. 708–721. Springer, Heidelberg (2008)

Robust and Flexible Error Handling in the AristaFlow BPM Suite

Andreas Lanz, Manfred Reichert, and Peter Dadam

Institute of Databases and Information Systems, University of Ulm, Germany
{Andreas.Lanz,Manfred.Reichert,Peter.Dadam}@uni-ulm.de

Abstract. Process-aware information systems will be not accepted by users if rigidity or idleness due to failures comes with them. When implementing business processes based on process management technology one fundamental goal is to ensure robustness of the resulting process-aware information system. Meeting this goal becomes extremely complicated if high flexibility demands need to be fulfilled. This paper shows how the AristaFlow BPM Suite assists process participants in coping with errors and exceptional situations in a flexible and robust way. In particular, we focus on novel error handling procedures and capabilities using the flexibility provided by ad-hoc changes not shown in other context so far.

Keywords: Process-aware Information System, Adaptive Process, Error Handling.

1 Introduction

During the last decade we developed the next generation process management system ADEPT2 [1]. Due to the high interest of companies in this technology, we transferred it into an industrial-strength process management system called *AristaFlow BPM Suite* [2]. One of the fundamental goals of AristaFlow is to enable robust and flexible *process-aware information systems* (PAISs) in the large scale. In particular, we want to ensure error-safe and robust process execution even at the occurrence of exceptions or dynamic process changes. Recently, ADEPT2 and AristaFlow, respectively, were applied to a variety of challenging applications in domains like healthcare [3,4], disaster management [5], logistics [6], and software engineering [7]. The overall goal was to learn more about process flexibility issues in advanced applications and to study how adaptive process management technology can be applied to deal with errors and exceptions.

This paper complements our previous work on ADEPT2 [1,2,8,9] and focuses on a fundamental pillar of any robust process implementation: *error handling*. One important dimension in this context concerns *error prevention*. We achieve the latter by applying a "correctness-by-construction" principle during process composition and by guaranteeing correctness and robustness in connection with (dynamic) process changes as well. This was probably the most influential challenge for our whole research. It also had significant impact on the development of

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 174–189, 2010.
© Springer-Verlag Berlin Heidelberg 2010

the AristaFlow BPM Suite. In particular we try to detect as many errors as possible (e.g. incomplete data flow specifications or deadlocks) already at buildtime in order to obviate their occurrence during runtime. In general errors cannot be always foreseen and thus be prevented. Therefore, another important dimension of PAIS robustness concerns *exception handling*. We will show that AristaFlow provides an easy, but yet powerful tool to handle exceptions during runtime. As we will show, in respect to flexible exception handling ad-hoc process changes have proven to be extremely useful. By utilizing them it even becomes possible to cope with severe process failures and to continue repaired processes in a correct way.

In Section 2 we introduce a simple application scenario which we use as illustrating example throughout the paper. Section 3 gives backgrounds on the AristaFlow BPM Suite. In Section 4 we show how this process management system copes with errors that might occur during process execution, and how to do this in a flexible and robust way. Section 5 summarizes real-world cases to which AristaFlow was applied. Section 6 discusses related work and Section 7 concludes with a summary and outlook.

2 Illustrating Application Scenario

We use a simple example to illustrate how different kinds of errors within PAISs can be handled when using AristaFlow. Consider Fig. 1. It shows a simple process of an online book store. In the first step, a customer request is entered and required data is collected. Next the bookseller requests pricing offers from his suppliers. In this example he will request an offer from Amazon using a web service and another offer from a second vendor using e-mail. After having received the pricing offers from both suppliers, the bookseller checks whether or not he can find a special offer for the requested books in the Internet. Finally, he makes an offer to his customer for the requested books.

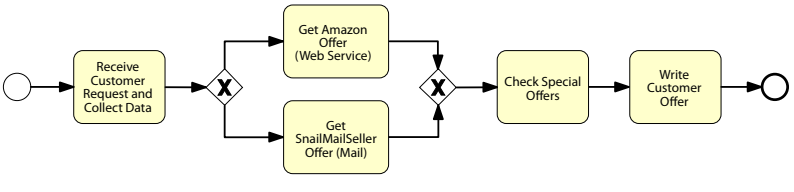

Fig. 1. Scenario: A simple process calling a web service (in BPMN notation)

As we will show, this scenario contains several sources of potential errors. Some of them can be detected and prevented at buildtime while others cannot. Assume, for example, that the process implementer does not foresee a way to enter the offer from *SnailMailSeller* into the system. In this case, activity *Write Customer Offer* might fail or produce an invalid output since its input parameters are not provided as expected. Another source of error is the Amazon web service,

Table 1. Typical errors in a PAIS

Buildtime	**Runtime**
– structural errors, e.g. • deadlocks • isolated activities – dataflow errors, e.g. • missing parameter values	– activity failure, e.g. • broken database connection • invalid input – context failure, e.g. • mismatch between real-world and process running in PAIS

which it might be not available when making the request and therefore the activity *Get Amazon Offer* might fail during runtime. Respective errors can be foreseen and hence be considered at buildtime. However, non-expected errors might occur as well; e.g., activity *Check Special Offers* might fail due to troubles with the Internet connection. Table 1 lists some typical errors in PAISs. For a more complete discussion we refer interested readers to [10].

In summary the following requirements for error-safe and robust process execution exist:

- Errors should be obviated at buildtime if possible.
- Users should be enabled to effectively deal with both expected and unexpected errors during runtime.
- Error and exception handling must not counteract formal process properties (e.g., proper termination) as guaranteed at buildtime by applying the aforementioned "correctness-by-construction" principle.

3 Background

As aforementioned the AristaFlow BPM Suite[1] is based on research results we obtained in the ADEPT1 and ADEPT2 projects. With ADEPT1 a first powerful prototype of the ADEPT technology became operational [11]. Based on hands-on experiences we gathered in several projects in the healthcare domain its most interesting feature was certainly the support of ad-hoc deviations [12]. Later ADEPT1 served as implementation platform for numerous projects (e.g., [4,6,13]). From this, we learned that a better integration of the offered features as well as an open API was needed. In 2004 we therefore started the development of the ADEPT2 project in which we targeted at a process management technology which enables ease of use for all user groups involved in the specification and execution of processes. Furthermore, the realization of robust process implementations and flexible support of dynamic process changes were fundamental project goals. Ensuring both robustness and ease of use for *process implementers* and *application developers* are indispensable in this context. This challenge was

[1] The AristaFlow BPM Suite is provided free of charge to universities for research and educational purposes. Please visit www.AristaFlow-Forum.de for more information on this topic. For commercial usage please visit www.AristaFlow.com.

probably the most influential one for the whole project [1,2]. It had significant impact on the development of the used process meta model as well as on our work on process flexibility and adaptivity. It meant, in essence, the following:

1. We have to hide the inherent complexity of process-orientation (especially in conjunction with flexibility) as far as possible from *system supervisors* and *application developers*; i.e., we have to perform all complex things "beneath the surface" in the process management system.
2. We have to provide powerful, high-level interfaces to *application developers*, based on which they can implement easily usable end user interfaces.

To achieve this, we realized a "correctness-by-construction" principle and guarantee correctness in the context of ad-hoc changes at the process instance level. "Correctness-by-construction" is realized by providing a theory [11] which precisely defines correctness criteria for the ADEPT meta model (e.g., absence of deadlocks, no isolated activities). This theory also defines a comprehensive set of change operations with pre-/post-conditions which ensure that, if the desired change satisfies the preconditions, the resulting process schema will be correct again, i.e., change operations allow to transform a structurally correct process schema into another structurally correct one [11]. Additionally, all change operations obey that data flow correctness is not violated. This is achieved by utilizing the block-structure of the underlying process meta model [11]. Particularly, a process schema can only be deployed to the process engine if it satisfies the above mentioned correctness criteria [2]. Finally, the change operations allow to safely deviate from the predefined process schema during runtime.

Another important aspect in the context of robustness is error handling. Any PAIS will not be accepted by users if rigidity comes with it or if its use in error situations is more expensive than just handling the error by calling the right people by phone. Therefore, users may deviate from the pre-modeled process by structurally adapting it, but without violating correctness properties.

4 Supporting the Described Application Scenario

In the following we re-consider the scenario from Section 2 from the perspectives of the *process implementer*, the *system*, the *end user*, the *system supervisor*, and the *process reengineer*. We demonstrate how each of these parties can be involved in handling errors. This is by no means a complete list of all parties which participate in the business process life cycle. We focus on those user groups involved in the specification and execution of processes, but exclude other user groups (e.g. business analysts or business process owners) who are mainly engaged in the monitoring and analysis of business processes.

4.1 Process Implementer Perspective

We first consider the *process implementer*. He is responsible for correctly modeling processes as well as for linking their activities to application services.

Process Modeling. Fig. 2 shows a part of the process from Fig. 1 as it can be modeled using the *AristaFlow Process Template Editor*. Additionally, it depicts parts of the data flow between the process activities (e.g., data element *Customer Name* is read by activity *Write Customer Offer (JAVA)*). For implementing processes, we pursue the idea of process composition in a *"plug & play"* style which is additionally supported by on-the-fly correctness checks if needed. More precisely, AristaFlow provides an intuitive graphical editor and composition tool to *process implementers* (cf. Fig. 2), and it applies the *correctness-by-construction* principle by providing at any time only those operations to the user which allow to transform one structurally sound process schema into another one; i.e., change operations are enabled or disabled according to the region which is marked in the process graph for applying an operation [2]. Deficiencies not prohibited by this approach (e.g., concerning data flow correctness) are checked on-the-fly and are reported continuously in the problem window of the *Process Template Editor*. As a prerequisite, for example, implicit data flow dependencies among the application services (implementing the process activities) have to be made known to the process engine. An example is depicted in Fig. 2, where AristaFlow detects that data element *Customer Price per Unit* is read by activity *Write Customer Offer* although it is not written by any preceding activity. Such deficiencies are highlighted to the *process implementer* who then can correct the model as required.

Generally, we should not require from *process implementers* that they have detailed knowledge about the internals of the application services they can assign to process activities. However, this should not be achieved by undermining the *correctness-by-construction* principle.

Activity Implementation and Configuration. In AristaFlow, all kinds of executables (e.g., user forms, web services, database components, file operations,

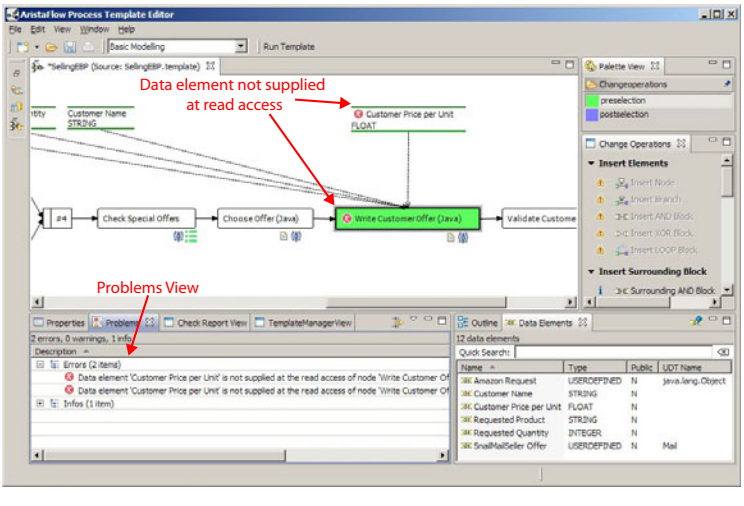

Fig. 2. AristaFlow Process Template Editor

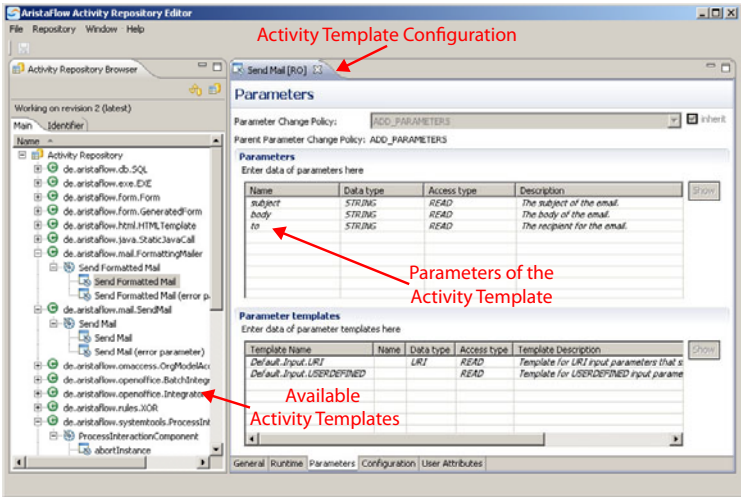

Fig. 3. Activity template configuration with AristaFlow Activity Repository Editor

etc.) that may be associated with process activities first need to be registered in the *Activity Repository* as activity templates (cf. Fig 3). An activity template, in turn, provides required information to the *Process Template Editor*; e.g., about mandatory and optional input/output parameters or about data dependencies to other activity templates. An *application developer* who wants to introduce a new application service to the PAIS must first implement a corresponding activity template and then add it to the *AristaFlow Activity Repository*. This way it becomes available and accessible within the *Process Template Editor* during process composition.

To ease the implementation of activity templates, AristaFlow provides several levels of abstraction. This includes the execution environment at the lowest one. An execution environment defines the set of methods (e.g., initialization and execution of the activity) needed to interact with the AristaFlow runtime system (i.e., the process engine) as well as to implement the operations and properties (e.g., shall the activity be suspendable, abortable, etc.) to be provided by the activity template. However, the implementation of such execution environment requires knowledge about AristaFlow internals and, therefore, will typically not be the task of an ordinary *application developer*, but be performed by *system implementers*. Next, activity templates provide predefined configuration sets for execution environments. Depending on the intended purpose of usage, an activity template can be very specific or rather generic, where a more specific activity template can be derived from a generic one. An activity template for a web service call, for example, may be completely pre-configured; e.g., the input/output parameters and all configuration and connection settings are fixed. In this case, the only remaining task for the *process implementer* is to check whether or not the proposed mapping of input/output parameters to process data elements (i.e., the process variables used within the respective process to exchange data

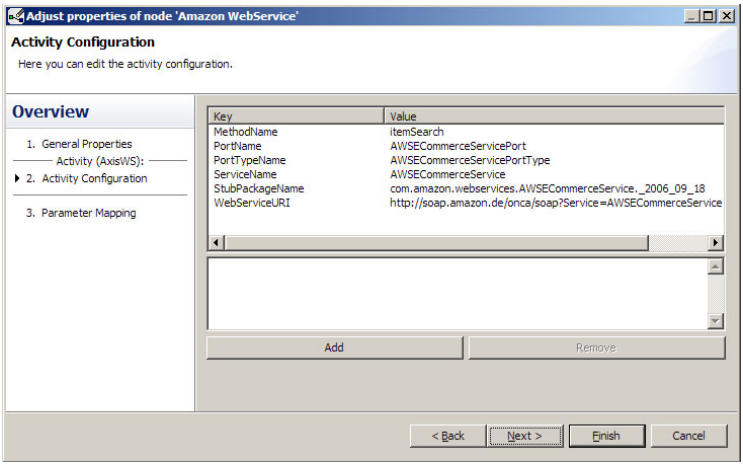

Fig. 4. Activity configuration of a generic web service activity template

among activities) is correct. A more generic web service activity template, in turn, may allow the *process implementer* to specify connection details of the web service (as illustrated in Fig. 4) or even allow to configure the number and types of input/output parameters.

In general, the AristaFlow runtime environment requires certain information about the runtime behavior of the activities; e.g., whether or not they may be aborted, suspended or undone. The implementer of an activity template has to inform the AristaFlow runtime environment which of these facilities are supported by the activity. For this case, he must also provide the implementation of this functionality in the respective execution environment [8]. Such provision of activity templates at different levels of abstractions constitutes a powerful and flexible means for process composition in a "plug & play"-like fashion.

Given the *AristaFlow Process Template Editor* and the *AristaFlow Activity Repository*, the *process implementer* just drags and drops the activity templates from the *Activity Repository Browser* window of the *Process Template Editor* onto the desired location in the process graph. Configuration efforts in respect to the chosen activity template are then reduced to the provision of the remaining configuration values in the configuration wizard of the respective activity template (cf. Fig. 4). For example, if a web service activity template shall be implemented one page of the wizard will fix the required input parameters and the output parameters for the attribute values of the web service, a second one the settings of binding information for the web service (e.g. the web service URL) (cf. Fig. 4), and a third one the mapping of activity input/output parameters to process data elements (cf. Fig. 5).

One major advantage of this approach is that common errors, e.g. missing data bindings, can be completely prevented at buildtime. Therefore the time needed for testing and debugging can be significantly reduced; i.e., AristaFlow

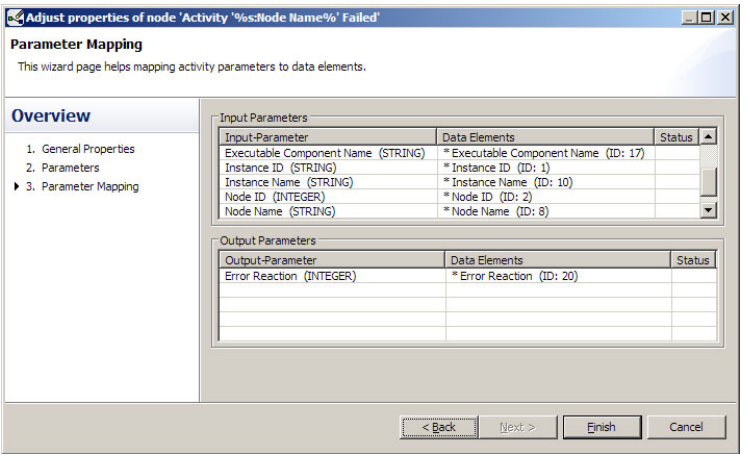

Fig. 5. Mapping activity parameters to data elements

guarantees that released process implementations are sound and complete with respect to the data dependencies of the used activity templates.

Nevertheless, *correctness-by-construction* and automated checks can only ensure correct execution of processes on a syntactical level. By contrast, semantic errors cannot be detected by automated checks. Therefore, AristaFlow provides a sophisticated *Test Environment* which allows *process implementers* to test processes prior their release in the production system. Using the *AristaFlow Test Client* it is even possible to execute only partially specified processes, i.e., not all activities need to have associated activity templates or staff assignment rules may be still undefined. In case an activity has no assigned activity template, in this test mode the user will be supported by automatically generated forms, which allow him to review the input parameters of the activity and to set its output parameters. In particular, this enables *process implementers* to get rapid feedback from future users during process development. Consequently, semantic errors and misinterpretations can be partially detected at a very early stage of the process implementation phase.

4.2 System Perspective

In principle, the approach described in Section 4.1 ensures that released process models are executable by the system in an error-safe way. As always, this might not hold in practice. Again, consider the scenario from Fig. 1. The web service associated with activity *Get Amazon Offer* might not be available during process execution, leading to an exception that needs to be handled. Such errors can neither be detected in advance nor be prevented by buildtime checks.

However, failures of the Amazon web service might be anticipated by the *process implementer*. Thus he can assign specific error handling procedures to the respective activity. Following a strict process paradigm, AristaFlow runs specific processes to handle exceptions; i.e., we provide a reflective approach

in which error handling itself can be accomplished base on a (normal) process instance running in the PAIS. A simple error handling process is shown in Fig. 6. Depending on whether or not the failure of the process activity was triggered by the user (e.g. through an abort button) either the *system supervisor* is notified about the failure or the process terminates silently. Generally, error handling processes can be arbitrarily complex and long running processes (e.g., comprising compensation tasks). It is noteworthy that AristaFlow treats error handling processes the same way as any other process. Thus they may refer to any activity registered in the repository. In particular, this enables error handling at a higher semantic level, as well as the involvement of users if required. We further can assign error handling processes to the activities of another error handling process if desired. This way it becomes possible to implement several layers of error-handling on top of each other.

If an activity fails the error handling process assigned to it will be initiated and be provided with all data necessary to identify, classify, and handle the error; e.g. the ID of the failed activity instance, the agents (i.e., user or automated agent) responsible for the activity, and the cause of the error (cf. Fig. 6).

After an error handling process has been created and deployed to the AristaFlow Server, it can be assigned to an activity by simply selecting it from a list of available processes. It is further possible to assign an error handling process to the whole process instead of single activities. This general error handling process will then be used if a failed activity has no directly associated error handling process. In case there is no error handling process being assigned to either the activity or the process a default error handling process will be used.

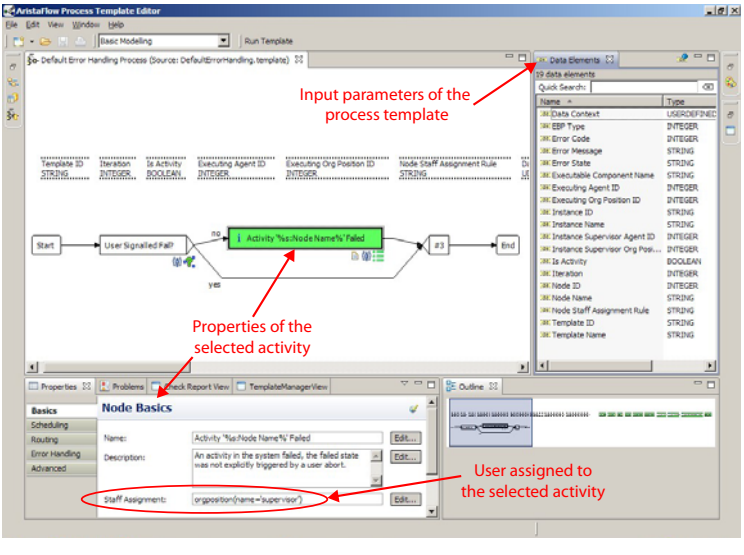

Fig. 6. A simple error handling process

Another advantage of user-defined processes for error handling is the possibility to use standard process modeling tools and techniques for designing error handling strategies. Therefore *process implementers* do not need to learn any new concept to enable error handling. Another important advantage is that error handling at a higher semantic level can be easily achieved. For example, it is also possible to use more complex error handling strategies like compensation or to apply ad-hoc changes to replace parts of the failed process.

4.3 End User Perspective

In certain cases simple error handling processes like the one depicted in Fig. 6 might be not appropriate since they increase the workload of the *system supervisor*. Most standard errors can also be handled in a (semi-)automatic way by the agent executing the activity. Upon failure of the respective activity the agent responsible for its execution could be provided with a set of possible error handling strategies he can choose from. An example for a more complex error handling process is depicted in Fig. 7. Here the agent can choose between several ways to handle the occurred error: retrying the failed process step, aborting the whole process instance, or applying predefined changes to fix or compensate the error. Additional error handling strategies may, for example, include the escalation of the respective case to a responsible supervisor or an enquiry with a more advanced user on how to handle the respective error.

Generally the concrete error handling strategies suggested to particular users may depend on their capabilities and position as captured in the organizational model. Consequently the assignment of error handling processes and respective activities can be done dynamically and user-dependent in AristaFlow.

To flexibly cope with errors and exceptions, end users are not only allowed to dynamically adapt process instances, but are also assisted in retrieving and reusing knowledge about previously performed process changes applied in similar problem context. Basic to this change reuse is the integration of the adaptive process management system with concepts and methods provided by case-based reasoning technology. This allows for expressing the semantics of process changes, for memorizing these adaptations, and for reusing them in similar context later. We implemented such an integrated approach in the ProCycle project [14].

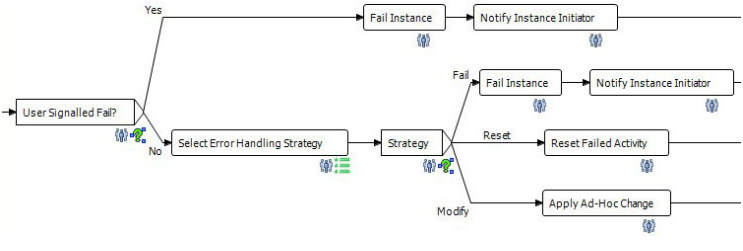

Fig. 7. A more complex error handling process involving the user

The described semi-automatic, user-centered approach offers many advantages. Since for each process activity a predefined set of possible error handling strategies can be provided to users, they do not need to have detailed knowledge about the process to handle errors appropriately, but are guided by the system in error situations instead. This particularly fosters the reduction of waiting times in the context of failed activity instances since users can handle errors immediately by their own and do not have to wait for their busy help desk to do this for them.

4.4 System Supervisor Perspective

Certain errors cannot be handled by the user. This applies, for example, to errors that might not have been foreseen at buildtime, i.e., no appropriate error handling process exists. In other cases it might be simply not possible to handle errors in an easy and generic way. At that point a *system supervisor* should be notified about the error. For example, this can either be done through an automatic notification by the error handling process (e.g., by adding a respective item to his worklist) or by a user calling the help desk by phone. The *system supervisor* then can use the *AristaFlow Process Monitor* shown in Fig. 8 to identify the process in trouble. This can be done by either using the process identifier provided by the error process or by applying different sets of filters to the list of process instances currently known by the system (cf. Fig. 8). Process instances can, for example, be identified by searching for active instances with failed activities, searching for modified instances, or searching for instances by name. Next, the *system supervisor* can take a look at the process instance in trouble, analyze its execution log, and decide for appropriate error handling measures. Additionally, the *system supervisor* can use the above described filters of the *AristaFlow Process Monitor* to keep track of failed instances; e.g., he can intervene if a web service becomes unavailable for a longer period of time.

Consider again our bookseller example from Fig. 1. Assume that a process instance wants to issue a request for a book using Amazon's web service facilities, but then fails in doing so. The *system supervisor* detects that the process is in trouble and uses the *AristaFlow Process Monitor* to take a look at this process instance (cf. Fig. 8). Analyzing the execution log of the failed activity he detects that its execution failed because the connection to Amazon could not be established. Let us assume that he considers this as a temporary problem and just resets the activity so that it can be repeated once again. Being a friendly guy, he takes a short look at the process instance and its data dependencies, and realizes that the results of this and the subsequent activity are only needed when executing the *Choose Offer* activity. Therefore, he moves these two activities after activity *Check Special Offers*; i.e., the user can continue to work on this process instance before the PAIS tries to re-connect to Amazon (cf. Fig. 9). To accomplish this change he would switch to the *Instance Change Perspective* of the *Process Monitor* which provides the same set of change operations as the *Process Template Editor* (for a general overview on process change patterns see [15]). In fact, the *Instance Change Perspective* is the *Process Template Ed-*

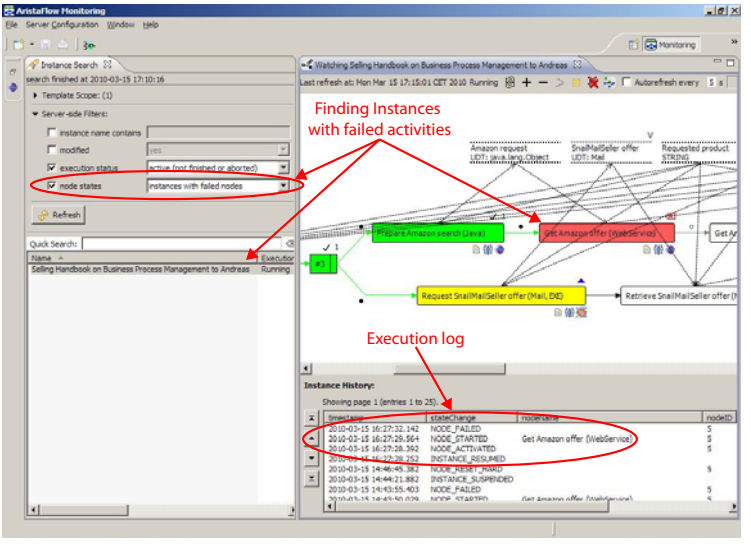

Fig. 8. Process Monitor: Monitoring Perspective

itor, but it is aware that a process instance has been loaded and, therefore, all instance-related state information is taken additionally into account when enabling/disabling change operations and applying correctness checks.[2] The system administrator would now move the two nodes to their new position by using the respective standard change operation. The resulting process is depicted in Fig. 9.

Assume now that the web service problem lasts longer than expected and, therefore, the user wants to call Amazon by phone to get the price that way. In this case he would ask the *system supervisor* to delete the activities in trouble and to replace them with a form-based activity (or any other suitable activity) which allows to enter the price manually. Note that structural ad-hoc changes provide the means to realize such advance exception handling policies.

4.5 Process Reengineer Perspective

As discussed, in AristaFlow certain errors can be handled by dynamically adapting the corresponding process instance. When considering a larger instance collection, respective model adaptations often result in a large number of model variants derived from the same process model, but slightly differing in structure. We foster learning from process instance adaptations in order to discover an improved process model that can serve as reference for future process instances of the respective type. An algorithm enabling such learning and model improvement is presented in [16]. Finally, after identifying potential changes the process (re-)engineer can perform a schema evolution. In this context he may also migrate running instances to the new process model version if desired [9].

[2] Whether or not a particular change can be applied in the current process state is decided based on well defined correctness criteria as suggested by ADEPT2 [9].

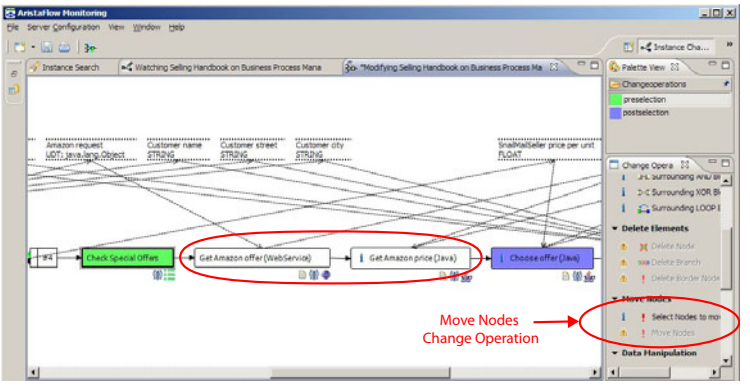

Fig. 9. Process Monitor: Instance Change Perspective

5 Applying AristaFlow in Practice

Recently, several projects applied the AristaFlow process management technology in challenging domains like healthcare, logistics, disaster management, and software engineering. In each project sophisticated PAISs were realized which make use of the AristaFlow error handling features. For us, one important goal of these projects was to understand whether or not the designed features are applicable in practice.

Applying AristaFlow to Software Engineering Processes. The Q-Advice project [7] tries to assist overburdened software engineers by providing orientation and guidance through automated workflows. Yet, since there are so many different kinds of issues with ambiguous and subjective delineation, it is difficult and burdensome to universally and correctly model them in advance. This also leads to workflows of considerable size and complexity. The Q-Advice project tries to alleviate this by starting with a basic and simple workflow for each case and then, utilizing context information, dynamically extends it with activities matching the current situation. Overall, Q-Advice provides situational and tailored support and guidance for software engineers. In particular the workflows resulting from the Q-Advice approach are much simpler than pre-modeled workflows would be. Respective adaptation features directly make use of AristaFlow's change facilities. In this context, the AristaFlow error handling processes and the provided change support features have proven to be especially useful. For example, if a bug is detected during the final steps of a release phase it needs to be decided whether or not this bug shall to be fixed prior to the release. If the bug turns out to be a show stopper, in turn, it may become necessary to change great parts of the release process.

Applying AristaFlow in Healthcare and Logistics. Healthcare and logistics are both characterized by high flexibility demands. Additionally, both require tools that are easy to use since domain specialists have no IT knowledge. By supporting domain-specific views on processes (e.g., clinical pathways) and services, the *SPOT* project [3] (*Service-based technologies for orchestrating*

PrOcesses in logisTics and healthcare) enables end-users to actively shape the different phases of the process life cycle. In both domains exceptional situations are part of the daily business and need to be handled quickly and in a way easy to use by end-users. Another requirement fully met by AristaFlow concerned application integration, i.e., to integrate heterogeneous, autonomous applications in a process-oriented way.

Applying AristaFlow in Disaster Management. The project on *process-aware, cooperative emergency management of water infrastructures* [5] aimed at improving and supporting emergency management for flood responses through new IT methods. During the project, procedures and courses of actions were analyzed, and results were mapped to formal process models. On the basis of an organizational model, the activities of the process models were assigned to responsible parties, thus enabling the involved organizations to act faster and in a more coordinated way. AristaFlow was used to manage and control the procedures and tasks during flood events as well as the corresponding information flow. Thus, it supported responders in planning and executing flood response operations in a coordinated, but flexible way. As emergency situations are not predictable, one important aspect was to provide the necessary flexibility, while ensuring robustness and error-safety of the PAIS.

Overall, in all these projects, adaptive process management technology for dynamically defining, composing and adapting process instances as well as for flexibly handling errors was indispensable. AristaFlow was one of the few systems that offers adaptation interfaces as well as adaptation features for this. All mentioned projects have proven that the previously described concepts are highly necessary in practice. At buildtime they enable us to detect and obviate design and runtime errors. This allows for easier and faster development of processes as the time needed for testing can be significantly reduced. At runtime the provided adaptation features enable PAIS utilizing the AristaFlow system to rapidly and dynamically react to exceptional situations like errors or changes in a process execution context. Additionally, the flexibility provided by the AristaFlow system allows processes to be at first only partially specified and then be further developed as they are executed.

6 Related Work

Besides ADEPT, YAWL [17] has been one of the first process engines to support some sort of "correctness-by-construction" as well as correctness checks at buildtime. jBPM [18] rudimentarily supports ad-hoc deviations of running process instances, but without any correctness assurance as provided by the AristaFlow BPM Suite. Furthermore, only simple adaptation patterns are provided [15]. Most BPEL-based workflow engines like WebSphere Process Server [19] support error handling processes using fault handlers, but without the possibility to structurally change process instances during runtime.

Declarative workflow systems like Declare [20] and Alaska Simulator [21] allow for a great degree of flexibility during process execution, but mostly lack means for ensuring robust process execution and for enabling application integration.

Additionally, they neither support the specification of data dependencies between activities nor correctness checks of the respective data flow.

Exception handling is an important topic in PAISs. In [10] the authors propose a classification framework for workflow exception handling in terms of workflow exception patterns. [22] incorporates language primitives for error handling into workflow systems and, like AristaFlow, allows error handling strategies to be modeled in the same notation as used for workflow processes. [23] uses a case-base reasoning approach to match errors with suitable error handling strategies. In [24] the authors propose several algorithms for mining process execution logs regarding exception handling. Based on the way past exceptions were handled, proposals are made on how to cope with the current one.

7 Summary and Outlook

Most existing PAISs are ill-equipped to meet the requirements of complex, real-world processes especially in the context of exceptional situations. Although a lot of papers claim that respective problems are solved in principle, we strongly believe that up to now they are even not completely understood. As shown, our tool provides an integrated solution for the easy and flexible handling of a variety of errors in exceptional situations. Most of the discussed projects would not have been possible without flexible process support as provided by the AristaFlow BPM Suite. Due to its *correctness-by-construction* principle and its comprehensive support of ad-hoc changes during runtime, as well as the possibility to define arbitrary error handling processes, AristaFlow is well suited to enable robust process implementations while preserving the possibility to flexibly react to exceptional situations during runtime.

The projects also showed that the provided flexibility is difficult to control by end-users, i.e., there is need for further research. Especially more sophisticated user-interfaces and user assistance are required for daily work. In this context, other perspectives like temporal constraints [25] become increasingly important.

References

1. Reichert, M., Rinderle-Ma, S., Dadam, P.: Flexibility in process-aware information systems. LNCS Transactions on Petri Nets and other Models of Concurrency (ToPNoC) 2, 115–135 (2009)
2. Dadam, P., Reichert, M.: The ADEPT project: A decade of research and development for robust and flexible process support - challenges and achievements. Computer Science - Research and Development 22, 81–97 (2009)
3. Frauenhofer ISST: SPOT Project (2010), `http://www.spot.fraunhofer.de/` (accessed 07.09.2010)
4. Müller, R., Rahm, E.: Dealing with logical failures for collaborating workflows. In: Scheuermann, P., Etzion, O. (eds.) CoopIS 2000. LNCS, vol. 1901, pp. 210–223. Springer, Heidelberg (2000)
5. Wagenknecht, A., Rüppel, U.: Improving resource management in flood response with process models and web GIS. In: TIEMS 2009, pp. 141–151 (2009)
6. Bassil, S., Keller, R., Kropf, P.: A workflow-oriented system architecture for the management of container transportation. In: Desel, J., Pernici, B., Weske, M. (eds.) BPM 2004. LNCS, vol. 3080, pp. 116–131. Springer, Heidelberg (2004)

7. Grambow, G., Oberhauser, R., Reichert, M.: Semantic workflow adaption in support of workflow diversity. In: 4th International Conference on Advances in Semantic Processing, SEMAPRO 2010 (2010)
8. Reichert, M., Dadam, P., Jurisch, M., Kreher, U., Göser, K., Lauer, M.: Architectural design of flexible process management technology. In: Proc. PRIMIUM Subconference at MKWI 2008, pp. 415–422 (2008)
9. Rinderle, S., Reichert, M., Dadam, P.: Flexible support of team processes by adaptive workflow systems. Distributed and Parallel Databases 16, 91–116 (2004)
10. Russell, N., van der Aalst, W., ter Hofstede, A.: Workflow exception patterns. In: Martinez, F.H., Pohl, K. (eds.) CAiSE 2006. LNCS, vol. 4001, pp. 288–302. Springer, Heidelberg (2006)
11. Reichert, M., Dadam, P.: ADEPTflex - supporting dynamic changes of workflows without losing control. Journal of Intelligent Information Systems 10, 93–129 (1998)
12. Reichert, M., Hensinger, C., Dadam, P.: Supporting adaptive workflows in advanced application environments. In: Proc. EDBT Workshop on Workflow Management Systems, pp. 100–109 (1998)
13. Bassil, S., Benyoucef, M., Keller, R., Kropf, P.: Addressing dynamism in e-negotiations by workflow management systems. In: Proc. 13th International Workshop on Database and Expert Systems Applications (DEXA 2002), pp. 655–659 (2002)
14. Weber, B., Reichert, M., Wild, W., Rinderle-Ma, S.: Providing integrated life cycle support in process-aware information systems. Int. Journal on Cooperative Information Systems 18, 115–165 (2009)
15. Weber, B., Reichert, M., Rinderle-Ma, S.: Change patterns and change support features - enhancing flexibility in process-aware information systems. Data and Knowledge Engineering 66, 438–466 (2008)
16. Li, C., Reichert, M., Wombacher, A.: Discovering reference models by mining process variants using a heuristic approach. In: Dayal, U., Eder, J., Koehler, J., Reijers, H.A. (eds.) BPM 2009. LNCS, vol. 5701, pp. 344–362. Springer, Heidelberg (2009)
17. Russell, N., ter Hofstede, A.: Surmounting BPM challenges: the YAWL story. Computer Science - Research and Development 23, 67–79 (2009)
18. Koenig, J.: JBoss jBPM (whitepaper) (2004)
19. Kloppmann, M., König, D., Leymann, F., Pfau, G., Roller, D.: Business process choreography in WebSphere: Combining the power of BPEL and J2EE. IBM Systems Journal 43, 270–296 (2004)
20. van der Aalst, W., Pesic, M., Schonenberg, H.: Declarative workflows: Balancing between flexibility and support. Computer Science - Research and Development 23, 99–113 (2009)
21. Weber, B., Reijers, H.A., Zugal, S., Wild, W.: The declarative approach to business process execution: An empirical test. In: van Eck, P., Gordijn, J., Wieringa, R. (eds.) CAiSE 2009. LNCS, vol. 5565, pp. 470–485. Springer, Heidelberg (2009)
22. Hagen, C., Alonso, G.: Exception handling in workflow management systems. IEEE Transactions on Software Engineering 26, 943–958 (2000)
23. Luo, Z., Sheth, A., Kochut, K., Miller, J.: Exception handling in workflow systems. Applied Intelligence 13, 125–147 (2000)
24. Hwang, S., Ho, S., Tang, J.: Mining exception instances to facilitate workflow exception handling. In: Proc. Database Systems for Advanced Applications, pp. 45–52 (1999)
25. Lanz, A., Weber, B., Reichert, M.: Workflow time patterns for process-aware information systems. In: Nurcan, S., Ukor, R. (eds.) BPMDS 2010 and EMMSAD 2010. LNBIP, vol. 50, pp. 94–107. Springer, Heidelberg (2010)

The NORMA Software Tool for ORM 2

Matthew Curland[1] and Terry Halpin[2]

[1] LogicBlox, USA
[2] LogicBlox, Australia and INTI International University, Malaysia
{matthew.curland,terry.halpin}@logicblox.com

Abstract. Second generation Object-Role Modeling (ORM 2) is a prime exemplar of fact-orientation, an approach that models the underlying facts of interest in an attribute-free way, using natural sentences to identify objects and the roles they play in relationships. ORM 2 provides languages and procedures for modeling and querying information systems at a conceptual level as well as mapping procedures for transforming between ORM structures and other structures, such as Entity Relationship (ER) models, class models in the Unified Modeling Language (UML), relational database models, extensible markup language schemas (XSD), and datalog. This paper provides an overview of Natural ORM Architect (NORMA), an ORM 2 tool under development that is implemented as a plug-in to Microsoft Visual Studio. For data modeling purposes, ORM typically provides greater expressive power and semantic stability than provided by UML or industrial versions of ER. NORMA's support for automated verbalization and sample populations facilitates validation with subject matter experts, and its live error-checking provides efficient feedback to modelers.

1 Introduction

Fact-oriented modeling is a conceptual approach (including languages and procedures) for modeling, transforming, and querying information, that specifies the fact structures of interest as well as the applicable business rules in terms of concepts that are intelligible to the business users. Unlike Entity-relationship modeling (ER) [4] and class diagramming in the Unified Modeling Language (UML) [18], fact-orientation makes no use of attributes as a way to express facts, instead representing all ground assertions of interest as atomic (non-decomposable) facts that are either existential facts or elementary facts. An existential fact simply asserts the existence of an entity (e.g. There is a country named 'Australia'). An elementary fact predicates over individuals (objects that are either entities or values). For example, "The Country named 'Australia' is large" expresses a unary fact about an entity, and "The Country named 'Australia' has the Nickname 'Down Under'" expresses a binary fact that relates an entity to a value.

Elementary facts are expressed using mixfix predicates, and are instances of *fact types*. For example, the UML attributes Person.isSmoker and Person.birthdate are modeled instead as Person smokes (unary fact type) and Person was born on Date (binary fact type). Higher arity fact types are allowed, for example Person played Sport for Country (a ternary)

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 190–204, 2010.
© Springer-Verlag Berlin Heidelberg 2010

and Product in Year in Region sold in Quantity (a quaternary). This attribute-free approach facilitates natural verbalization and population of models (important for validating models with nontechnical domain experts), and promotes semantic stability (e.g. one never needs to remodel an attribute and associated access paths if one later wants to talk about an attribute).

Fact-oriented modeling approaches include *Object-Role Modeling* (*ORM*) [8], Cognition-enhanced Natural Information Analysis Method (CogNIAM) [16], the Predicator Set Model (PSM) [14], and Fully-Communication Oriented Information Modeling (FCO-IM) [1]. The Semantics of Business Vocabulary and Business Rules (SBVR) initiative is fact-based in its use of attribute-free constructs [19]. For an overview of fact-oriented modeling approaches, including history and research directions, see [7].

Since the 1970s, various *software tools* have supported fact-orientation. Early tools based on NIAM include IAST and RIDL* (based on the RIDL language [15]). CogNIAM is currently supported by Doctool. FCO-IM is supported by the Case Talk tool. Related ontology tools include DOGMA Studio and Collibra. ORM tools began with InfoDesigner, which later evolved into InfoModeler, VisioModeler, and the ORM Source Model solution in Microsoft Visio for Enterprise Architects. ActiveQuery [3] is an ORM conceptual query tool released as a companion to InfoModeler.

More recently, a number of tools have supported second generation ORM (*ORM 2*) [6]. These include *Natural ORM Architect (NORMA)*, ActiveFacts [13], and ORM-Lite. For data modeling purposes, the ORM 2 graphical notation is far more expressive than UML's graphical notation for class diagrams, and is also much richer than industrial ER notations [**9**].

Business rules are modeled in ORM 2 as constraints or derivation rules that apply to the relevant business domain. *Alethic constraints* restrict the possible states or state transitions of fact populations (e.g. **No** Person is a parent of **itself**). *Deontic constraints* are obligations that restrict the permitted states or state transitions of fact populations (e.g. **It is obligatory that each** Doctor is licensed to practice). *Derivation rules* enable some facts or objects to be derived from others. For example, grandparenthood facts may be derived from parenthood facts using the rule Person_1 is a grandparent of Person_2 **if** Person_1 is a parent of **some** Person_3 **who** is a parent of Person_2. Similarly, instances of the subtype Father may be derived from parenthood and gender facts using the rule **Each** Father **is a** Person **who** is male **and** is a parent of **some** Person_2.

A detailed summary of the ORM 2 graphical notation is accessible at http://www.orm.net/pdf/ORM2GraphicalNotation.pdf. A thorough treatment of the theory and practice of ORM 2 may be found in [12].

The rest of this paper provides an overview of the NORMA tool, and is structured as follows. Section 2 summarizes the main components of NORMA. Section 3 illustrates some important capabilities of NORMA. Section 4 provides details of the implementation architecture. Section 5 summarizes the main contributions and outlines future research directions.

2 Overview of NORMA

NORMA is implemented as a plug-in to Microsoft Visual Studio. Most of NORMA is open-source, and a public domain version is freely downloadable [17]. A professional version of NORMA is also under development. Fig. 1 summarizes the main components of the tool. Users may declare ORM object types and fact types textually using the Fact Editor, and these are then displayed graphically using auto-layout. Alternatively, object types and fact types may be entered directly on the current diagram by dragging elements from the toolbox and supplying appropriate names. Large ORM schemas are typically displayed using many pages of ORM diagrams.

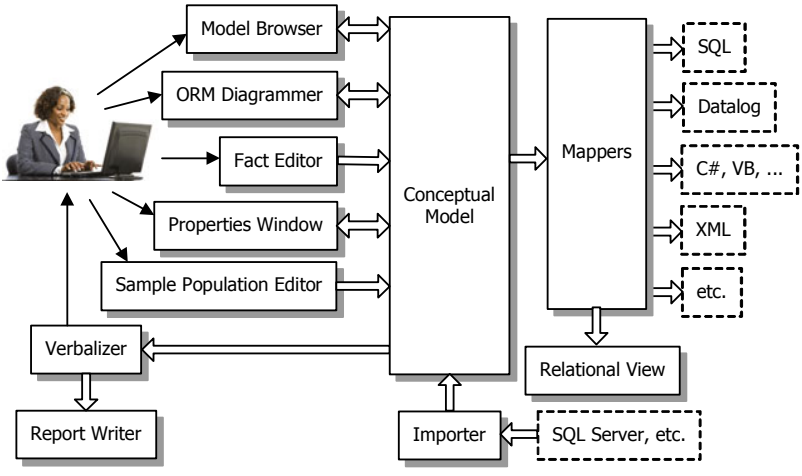

Fig. 1. Main components of NORMA

Currently, most ORM constraints must be entered in the ORM diagrammer or the Properties Window. These constraints are automatically verbalized in FORML (Formal ORM Language), a controlled natural language that is understandable even by nontechnical people. We plan to allow complete ORM models, including constraints and derivation rules, to be entered textually in FORML. An initial prototype to provide this support has been developed as an extension to the Fact Editor [11].

The Model Browser tool window provides a hierarchical view of all model components, and allows model elements to be dragged onto diagram pages. Sample object and fact instances may be entered in tabular format in the Sample Population Editor, and automatically verbalized. Explanatory comments and informal descriptions may be added to model elements. These comments are included in the output displayed in the Verbalization Browser. Reports in HTML format may be automatically generated including complete verbalizations of ORM models with embedded hyperlinks for easy navigation. These reports are especially valuable for non-technical domain experts to validate and sign off on the model specification.

The VisualBlox team at Logicblox recently extended the Model Browser to enable derivation rules for both fact types and subtypes to be formally captured and stored in

a rules component of the conceptual model based on the role calculus [5]. These derivation rules are also automatically verbalized.

Using mappers, ORM schemas may be automatically transformed into various implementation targets, including relational database schemas for popular database management systems (SQL Server, Oracle, DB2, MySQL, PostgreSQL), datalog [10], .NET languages (C#, VB, etc.), and XML schemas. A Relational View extension displays the relational schemas in diagram form. The semantics underlying relational columns can be exposed by selecting them and automatically verbalizing the ORM fact types from which they were generated. An import facility can import ORM models created in some other ORM tools, and reverse engineer relational schemas in SQL Server into ORM schemas. Import from further sources is planned.

Other components facilitate navigation and abstraction. For example, the Diagram Spy window facilitates views onto various pages as well as the copying of diagram selections from one page to another. Hyperlinks in the Verbalization Browser allow rapid navigation through a model. The Context Window allows one to focus on a given model element and progressively expand outwards to view its neighborhood in the global schema.

3 Some NORMA Highlights

Feedback from industrial practitioners indicates that *automated verbalization* support is one of the most useful features of NORMA. Fig. 2 shows a screen shot from NORMA illustrating verbalization of a join subset constraint.

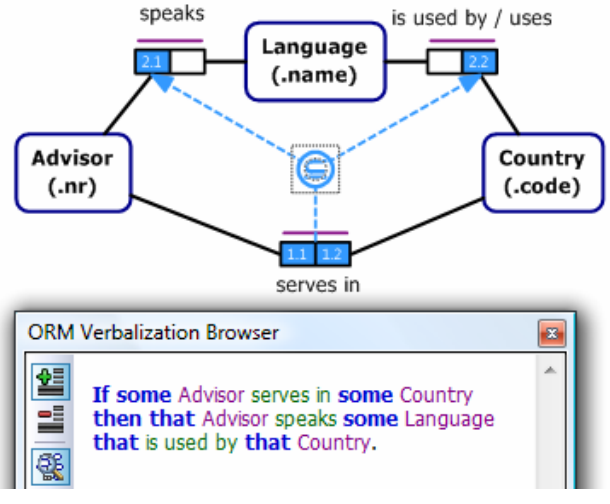

Fig. 2. NORMA screenshot showing verbalization of a join-subset constraint

Here we have three binary fact types: Advisor serves in Country; Advisor speaks Language; Language is used by Country. Entity types are shown as named, soft rectangles with their reference mode in parenthesis. In ORM, the term "role" means a part played in a

relationship (of arity one or higher). Logical predicates are depicted as a named sequence of role boxes connected to the object types whose instances play those roles. The bar over each predicate depicts a spanning uniqueness constraint, indicating that the fact types are *m:n*, and can be populated with sets of fact instances, but not bags.

The circled "⊆"connected by dashed lines to role pairs depicts a subset constraint. When the constraint shape is selected, NORMA displays role numbers to highlight the role sequence arguments to the constraint. In this example, the set of advisor-country instances of the role-pair (1.1, 1.2) are constrained to be a subset of the set of advisor-country instances populating the role-pair (2.1, 2.2) projected from the role path from Advisor through Language to Country. In passing through Language, a conceptual inner join is performed on its entry and exit roles, so this is an example of a join-subset constraint. The meaning of the constraint is clarified by the verbalization shown at the bottom of Fig. 2. Because every aspect of an ORM model can be automatically verbalized in such a high level language, non-technical domain experts can easily validate the rules without even having to see or understand the diagram notation.

A feature of NORMA that is especially useful to modelers is its *live error checking* capability. Modelers are notified immediately of errors that violate a metarule that has been implemented in the underlying ORM metamodel. Fig. 3 shows an example where the subset constraint is marked with red fill because it is inconsistent with other constraints present. In this case, the committee role of being chaired is declared to be mandatory (as shown by the solid dot on the role connection), while the committee role of including a member is declared to be optional. But the subset constraint implies that if a committee has a chair then it must have that person as a member. So it is impossible for the two fact types to be populated in this situation. NORMA not only detects the error but suggests three possible ways to fix the problem.

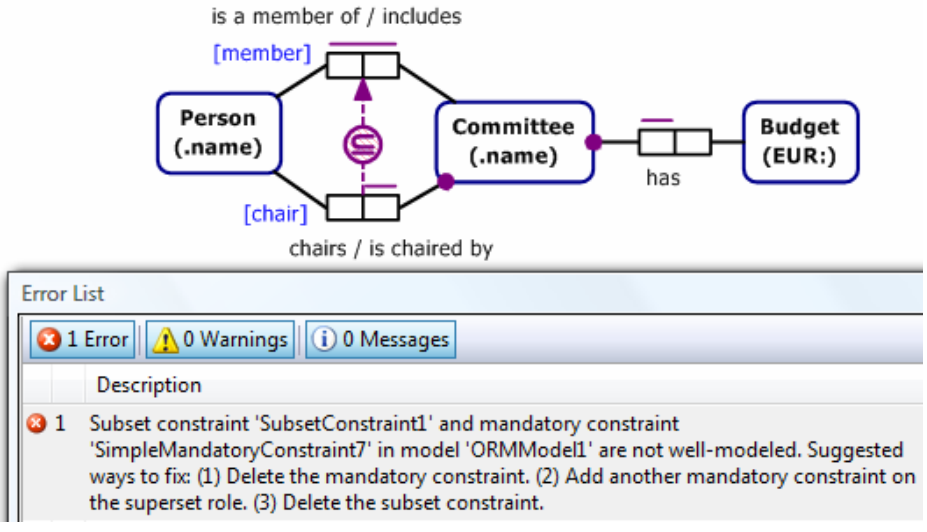

Fig. 3. Example of live error detection involving a problematic constraint pattern

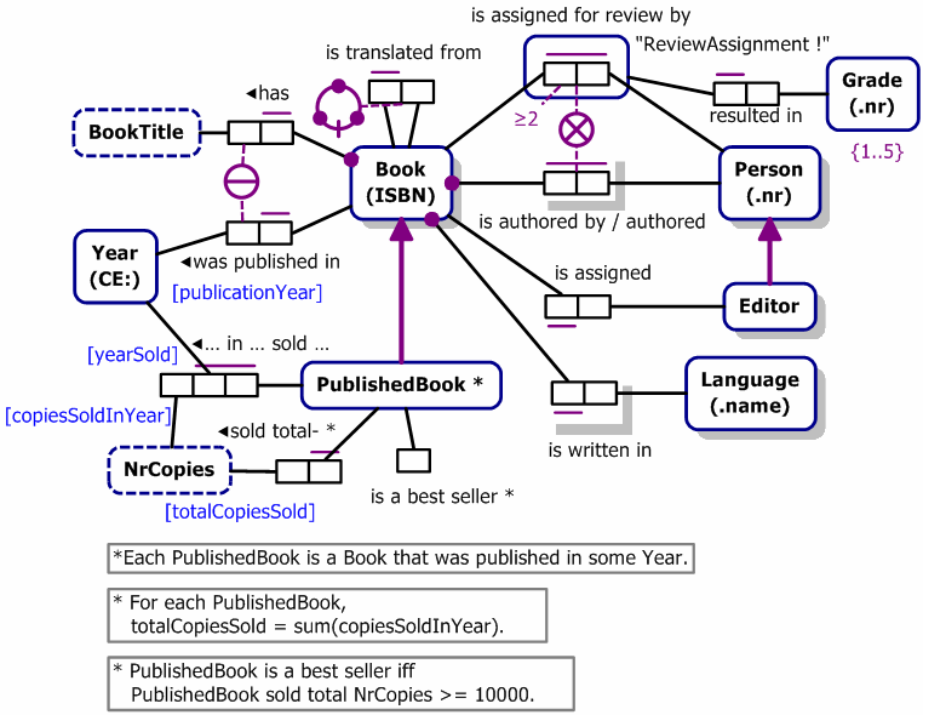

Fig. 4. NORMA screenshot illustrating rich support for graphic constraints

Fig. 4 shows a NORMA screenshot of one page from a larger ORM model concerning book publishing. This gives some idea of ORM's richly *expressive graphic constraint notation* for data modeling compared with UML and industrial ER. All fact types must have at least one reading, but may be given as many as desired. Here the authorship fact type has forward and inverse predicate readings, separated by a slash.

PublishedBook and Editor are subtypes, connected to their supertype by an arrow. PublishedBook is a derived subtype, as shown by its trailing asterisk, and the derivation rule supplied for it: **Each** PublishedBook **is a** Book **that** was published in **some** Year. This rule is expression in relational-style, using predicate readings. Optionally, roles may be assigned names, displayed in square brackets next to the role box. In this example, four roles are named. One reason to add role names in ORM is to enable derivation rules to be formulated in attribute-style (using role names for "attributes"). For example, the derived fact type PublishedBook sold total NrCopies is defined using the following attribute-style derivation rule: **For each** Publishedbook, totalCopiesSold = **sum**(copiesSoldInYear). Editor is an asserted subtype, so has no derivation rule. NORMA also supports semiderived types (some instances may be asserted and some derived).

The hyphen after "total" in the predicate "sold total" invokes hyphen-binding to bind the adjective "total" to its noun for automated verbalization. This causes the uniqueness constraint on this fact type to verbalize as "**Each** PublishedBook sold **at most one**

total NrCopies". Without the hyphen, the "**at most one**" quantifier would appear after "total" instead of before it. NORMA supports both forward and reverse hyphen binding.

The circled bar connecting the BookTitle and publicationYear roles depicts an external uniqueness constraint (each book title, publication year combination applies to at most one Book). The annotated circle next to the book translation fact type depicts an acyclic ring constraint, constraining the translation relationship to exclude any cycles. ORM includes a large list of such ring constraints (irreflexive, asymmetric, antisymmetric, intransitive, etc.). The "≥ 2" on the review assignment role is a frequency constraint (each book that is assigned for review is assigned to at least two reviewers). ORM's mandatory role and frequency constraints work properly with fact types of any arity, unlike UML's multiplicity constraints which fail to cater for some constraint patterns on *n*-ary associations (e.g. see pp. 361-362 of [12]).

The soft rectangle around the "is assigned for review by" predicate objectifies the relationship as ReviewAssignment. The exclamation mark "!" after the name of this objectification type indicates that it is independent (instances of it may exist independently of playing in other facts). The {1..5} annotation on Grade depicts a value constraint, restricting its grade numbers to be in the range 1 to 5 inclusive. The circled "X" between the review assignment and authorship fact types is a pair-exclusion constraint indicating that no book can be reviewed by one of its authors.

The shadows around the object types Book, Person, Editor and Language indicate that these types also appear elsewhere in the global schema. In the NORMA tool, if you right-click on such a duplicated shape and choose "Select on Diagram" you will be presented with a list of all the other pages on which that shape appears; and selecting the desired page will then take you to that page with the shape highlighted.

Similarly, the shadows around the "is authored by" and "is written in" predicates indicate that these fact types appear elsewhere in the global schema. Fig. 5 shows a fragment from another page in the global schema, where these two facts types appear. The circled subset constraint runs from the (language, person) pairs projected from the join path Person authored Book that is written in Language to the (language, person) pairs in the fact type Person is fluent in Language. The constraint is *deontic*, as shown by the small "o" (for "obligatory") on the constraint symbol, as well as its blue color (in contrast to alethic constraints, which are colored violet). The automated verbalization shown below indicates the deontic nature by starting with "**It is obligatory that**".

It is obligatory that
if some Person authored **some** Book **that** is written in **some** Language
then that Person is fluent in **that** Language.

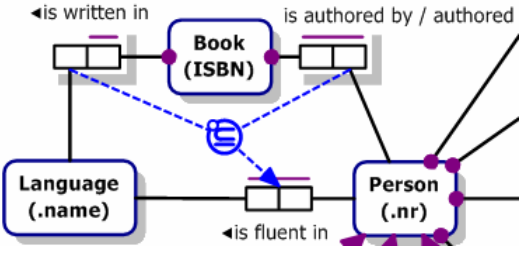

Fig. 5. NORMA screenshot fragment illustrating a deontic constraint

While Fig. 4 and Fig. 5 show many of the constraint varieties in ORM, they by no means cover all of them. A complete list of ORM constraints, with examples, may be accessed online at http://www.orm.net/pdf/ORM2GraphicalNotation.pdf. Further explanation of the first page of the book publisher example, as well as comparative schemas in UML and ER, may be found in [9].

Fig. 6 illustrates a very simple example of *mapping* from ORM to a relational schema. NORMA provides several mechanisms for controlling how the names of tables and columns are generated (e.g. here the use of the role name "birthCountry" ensures that the column name will be the same). The relational view displays columns that are not nullable in bold, and marks primary key columns and foreign key columns by "PK" and "FK" respectively. The SQL code for creating the relational schema is automatically generated for the main industrial DBMSs.

For traceability and validation purposes, the ORM verbalization is accessible directly from the relational columns, so touching any column automatically provides the underlying semantics. For example, Fig. 7 is a NORMA screenshot showing the verbalization displayed when the birthCountry column is selected. This feature is extremely useful for validating relational models with clients, and can be used even if they are not familiar with the ORM schema from which the relational schema was generated. This example also includes some sample data entered in the ORM model.

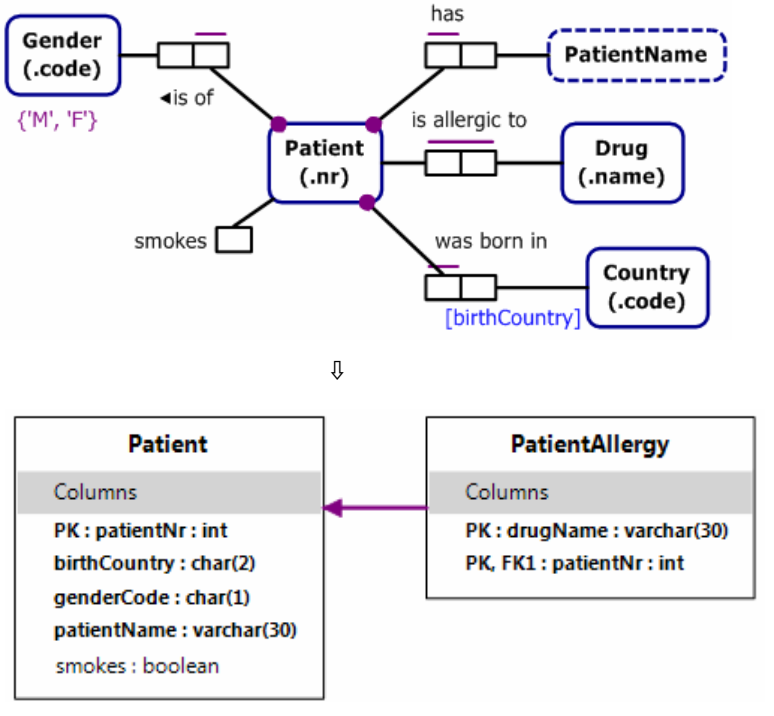

Fig. 6. Simple example of mapping an ORM schema to a relational schema

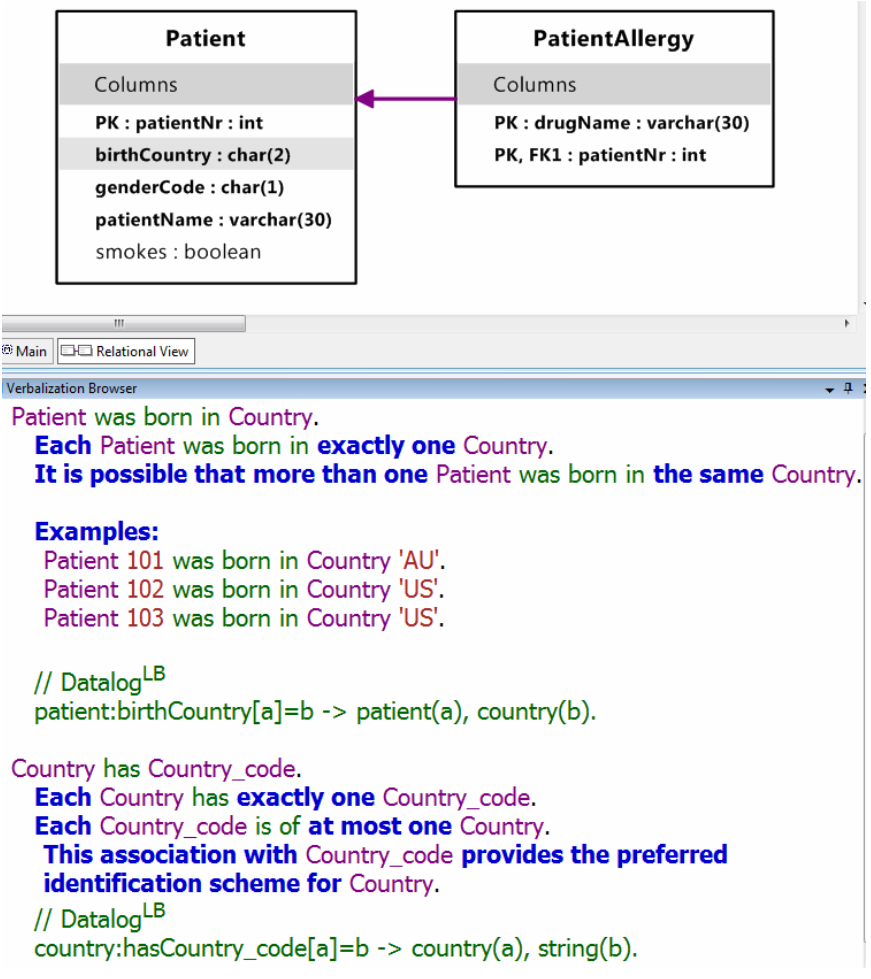

Fig. 7. Verbalization for the selected birthCountry column in the relational schema

This screenshot also shows some of the Datalog^{LB} code that can be generated if the predicate mapping extension is enabled (this extension is not currently available in the open source version of NORMA). Datalog^{LB} is an advanced form of typed datalog that provides rich support for complex rules as well as high performance. Current research at LogicBlox is extending NORMA to generate complete Datalog^{LB} code for ORM models, including advanced constraints and derivation rules [10]. For this trivial model, the Datalog^{LB} code may be set out as shown. For a more complex example, see [9].

```
Patient(x), patient:nr(x:y) -> uint[32](y).
Gender(x), gender:code(x:y) -> string(y).
Drug(x), drug:name(x:y) -> string(y).
Country(x), country:code(x:y) -> string(y).
```

```
patient:smokes(x) -> Patient(x).
patient:gender[x]=y -> Patient(x), Gender(y).
patient:name[x]=y -> Patient(x), string(y).
patient:drug:isAllergicTo(x,y) -> Patient(x), Drug(y).
patient:birthCountry[x]=y -> Patient(x), Country(y).
Patient(x) -> patient:gender[x]=_.
Patient(x) -> patient:name[x]=_.
gender:code(_:y) -> y="M"; y="F".
```

4 Implementation

The NORMA designer is built primarily on the Domain Specific Language (DSL) Toolkit from the Visual Studio SDK, as well as general Visual Studio extension points. The implementation of NORMA adds multiple framework services to DSL to enable a highly modularized and extensible system. Extension points are available for file importers, additional DSL models and designers that interact with the core NORMA models, and extension points for artifact generation. All core NORMA components have exactly the same architecture as the extension models, except that the core models cannot be removed from the list of current extensions.

DSL was chosen because it was a model-driven system, providing for a large percentage of the required code to be generated. The generated code defines a transacted object model with standard notifications that enable responsive secondary model changes in response to atomic changes in the object model. A particularly important feature of DSL is the built-in support for *delete closures*, which provide notifications for elements that are pending deletion but have not yet been detached from the model. Delete closures enable NORMA to minimize the parts of the model that require revalidation in response to a model change, which in turn enables extremely responsive incremental validation irrespective of model size.

The NORMA implementation relies on a number of enhancements to the DSL tooling and runtime components. Most extensions are necessary because many of the DSL supporting SDK components assume that the complete metamodel is known at all times, whereas NORMA makes the opposite assumption—the complete metamodel and associated rules are not known until a model file is opened and the set of extension models are read from the root XML element in the model file. Allowing multiple models also required significantly more flexibility for serialization of NORMA models than for a standard DSL model. The NORMA modeling framework includes multiple extensions to the DSL rules engine to enable compartmentalized model validation that minimizes incremental processing within a model and isolates dependent models from other models that they do not even know are loaded.

Extensions to NORMA may be classified into the following areas:

1. *Importers* allow XML data sources with schemas not supported first-hand by the NORMA designers (including older NORMA file formats) to be transformed automatically on load. Most importers are XSLT transformations, although additional wizards can be registered with Visual Studio to first translate a non-XML data source into XML suitable for import. An example

of a non-XML based importer is the database import wizard, which uses several steps to translate the schema of an existing database into an ORM model.

a. The import wizard identifies the type of DBMS target and uses DBMS-specific code to generate a temporary XML file with a .orm extension. This XML file uses the same database structure representation as the relational artifact generator mentioned below.
b. Visual Studio opens the file with the NORMA designer, which is associated with the .orm file extension.
c. The NORMA designer—which recognizes exactly one root format plus one format per extension—examines the root element in the XML file and determines whether the root file format or one or more of the extension formats are not native or do not correspond to the current recognized extension versions.
d. The designer looks for a registered import transform that recognizes the XML format and then executes the transform, producing a new XML file. Steps c and d are repeated as often as needed to produce a file format that can be directly read by the designer. For the database import case, three transforms are needed. Two of the transform steps prompt the user for transform parameters, which is another import feature supported directly by NORMA.
e. This import mechanism is also used for file format changes that are occasionally required as NORMA evolves. To protect against data loss, opening an existing file with an outdated format will result in a user notification offering to block the file save operation (SaveAs remains enabled).

2. *Primary Domain Models* (domain model is a DSL term for a metamodel) are extensions that provide model elements and validation rules for runtime execution inside the designer. An extension model provides an XML schema for serialized elements, a mapping between the in-memory model and the XML elements, a load fixup mechanism to reach an internally consistent state at load completion, and rules to maintain consistency for user-initiated changes after the model is loaded. Provided primary domain models include the core ORM metamodel (*FactType*, *ObjectType*, etc) and the relational extension metamodel, with elements such as *Table*, *Column*, and *ReferenceConstraint*.

The ability to dynamically validate model state enables a scalable platform by not requiring a full model analysis for incremental changes. Although NORMA validation errors are conceptually deontic constraints, each error is modeled as a normal element in the domain model, just like *ObjectType* or *FactType*. The transacted object model framework provided by DSL supports fine-grained tracking of atomic model changes, allowing rules to record elements related to a change, and then verify the error state for affected elements later in the transaction.

One of the challenges in incremental error validation is knowing when all changes directly caused by user input have been entered into the model. Validating affected elements before all external changes are known results

in duplication of work and expensive analysis with an incomplete state. The NORMA team has developed a system known as *delayed validation* that enables model validation to register an element/validator pair for delayed processing. The validator function is called once for each registered element after normal input has been completed. Validators run inside a transaction, so they can modify the contents of the model. Placing error validation objects in the models also means that validation is not required during undo/redo operations.

3. *Presentation Models* have the same characteristics as primary domain models and are modeled similarly. However, presentation elements are treated as views on the underlying model, not the model itself. The *ORM Diagram* and *Relational View*, along with their contained shapes, are defined in presentation models. A single element from a primary domain model may be associated with multiple presentation shapes. Presentation model changes are performed near the end of a transaction to ensure that all underlying model changes are complete.

4. *Bridge Models* are also domain models with the special function of relating two primary domain models. Bridge models consist of relationships between two other models, plus generation settings that control the current relationships between the models. Primary domain models are designed to be standalone so that transformations can be performed in either direction. This allows, for example, an importer to be written targeting the XML schema of the in-memory relational model that can be fully defined without corresponding ORM elements in the file. However, this design means that elements in a primary relational model cannot directly reference elements in the underlying ORM model. Bridge models store these relationships between primary models and allow changes in one standalone model to be applied to another standalone model generated from it.

   It is very important during bridge analysis used to generate one standalone model from another to have the complete set of changes from the source model. To facilitate this, the NORMA delayed validation mechanism prioritizes validator execution based on both model dependencies and explicit prioritization directives in the code. Model dependency analysis is sufficient in many cases. For example, a presentation model depends on the referenced primary model, so presentation validation is automatically performed after the primary model validation is complete. For bridge models, the default behavior is not correct because the bridge validators will naturally run after (not in-between) validation of the two models it is bridging.

   For cases where the default validator ordering is insufficient, validation routines use .NET attributes to explicitly run their validators *before*, *with*, or *after* another model. By running bridge validation after native validators for another model, we ensure a consistent and complete state of the source model. Careful placement of the bridge validators allows the bridge models to trigger changes based on both primary elements and derived error state

changes. For example, since the generated relational model requires a *FactType* to be completely specified (all role players specified with identification schemes with known data types plus an allowed uniqueness constraint pattern), the bridge model can watch for addition and deletion of validation errors like *RolePlayerRequiredError* and *FactTypeRequiresInternalUniquenessConstraintError* instead of reanalyzing underlying relationships.

5. *Shell Components* are views and editors targeting specific parts of an in-memory ORM model, such as the Model Browser and Fact Editor tool windows discussed earlier. Shell components interact with domain models by initiating transactions that modify the underlying model in response to user input, then responding to *model events* to update the shell display. Model events are a playback of a transaction log after the transaction is initially committed or repeated with an undo and redo operations. ORM model files can be loaded independently of any Visual Studio hosted shell components, allowing external tools to load an ORM model without Visual Studio running.

6. *Artifact Generators* produce non-ORM outputs such as DDL, class models, and other implementations mapped from an ORM model. In general, artifact generators are much easier to create than DSL models because generators deal with a static artifact—namely a snapshot of the ORM model in XML form—and do not have to worry about the change management that is the bulk of the implementation cost for domain models. NORMA's generation system supports a dependent hierarchy of generated files based on output format. A single generator can request inputs of both the standard ORM format (with required extensions specified by the generator) and any other formats produced by other registered generators.

   The hierarchical generation process allows analysis to be performed once during generation, and then reused for multiple other generators. NORMA provides XML schemas for all intermediate formats, allowing extension generators to understand and leverage existing work. Some examples of intermediate formats we use are DCIL (relational data constructs), DDIL (XML representation of data-definition constructs, created from the DCIL format), and PLiX (an XML representation of object-oriented and procedural code constructs). These intermediate XML formats are transformed into target-specific text artifacts. DDIL is directly transformed to either SQL Server or Oracle specific DDL formulations, and PLiX generates C#, VB, PHP, and other languages—all without changing the intermediate file formats.

   Another advantage of the use of well-defined intermediate structures is the ability to modify these structures during artifact generation. For example, attempting to decorate an ORM model with auditing constructs is extremely invasive at the conceptual model level. However, adding auditing columns to each table is a simple transform from DCIL to DCIL with the modified file conforming to the same schema but containing additional columns for each table. After all intra-format transforms are complete, the modified file is passed to the next stage in the generation pipeline.

5 Conclusion

This paper provided a brief overview of the NORMA tool and its support for ORM 2, a conceptual data modeling approach with greater semantic stability and richer graphical constraint expressibility than UML class modeling and industrial ER modeling. Key features of NORMA's support for ORM 2 were highlighted, including live error checking, and validation by automated verbalization and sample populations.

Major, recent work not reported on here because of space restrictions includes deep support for entry of formal derivation rules and their automated verbalization, as well as a prototype implementation of FORML 2 as an input language. Details on the latter may be found in [11]. Research is also under way to extend NORMA with support for dynamic rules [2].

References

1. Bakema, G., Zwart, J., van der Lek, H.: Fully Communication Oriented Information Modelling. Ten Hagen Stam (2000)
2. Cao, V.-T., Halpin, T.: Formal Semantics of Dynamic Rules in ORM. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM-WS 2008. LNCS, vol. 5333, pp. 699–708. Springer, Heidelberg (2008)
3. Bloesch, A., Halpin, T.: Conceptual queries using ConQuer-II. In: Embley, D.W. (ed.) ER 1997. LNCS, vol. 1331, pp. 113–126. Springer, Heidelberg (1997)
4. Chen, P.P.: The entity-relationship model—towards a unified view of data. ACM Transactions on Database Systems 1(1), 9–36 (1976)
5. Curland, M., Halpin, T., Stirewalt, K.: A Role Calculus for ORM. In: Meersman, R., Herrero, P., Dillon, T. (eds.) OTM 2009 Workshops. LNCS, vol. 5872, pp. 692–703. Springer, Heidelberg (2009)
6. Halpin, T.: ORM 2. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM-WS 2005. LNCS, vol. 3762, pp. 676–687. Springer, Heidelberg (2005)
7. Halpin, T.: Fact-Oriented Modeling: Past, Present and Future. In: Krogstie, J., Opdahl, A., Brinkkemper, S. (eds.) Conceptual Modelling in Information Systems Engineering, pp. 19–38. Springer, Berlin (2007)
8. Halpin, T.: Object-Role Modeling. In: Liu, L., Tamer Ozsu, M. (eds.) Encyclopedia of Database Systems. Springer, Berlin (2009)
9. Halpin, T.: Object-Role Modeling: Principles and Benefits. International Journal of Information Systems Modeling and Design 1(1), 32–54 (2010)
10. Halpin, T., Curland, M., Stirewalt, K., Viswanath, N., McGill, M., Beck, S.: Mapping ORM to Datalog: An Overview. In: Meersman, R., Dillon, T., Herrero, P. (eds.) On the Move to Meaningful Internet Systems 2010: OTM 2010 Workshops. LNCS. Springer, Heidelberg (2010) (to appear)
11. Halpin, T., Wijbenga, J.P.: FORML 2. In: Nurcan, S., Ukor, R. (eds.) BPMDS 2010 and EMMSAD 2010. LNBIP, vol. 50, pp. 247–260. Springer, Heidelberg (2010)
12. Halpin, T., Morgan, T.: Information Modeling and Relational Databases, 2nd edn. Morgan Kaufmann, San Francisco (2008)
13. Heath, C.: ActiveFacts (2009), http://dataconstellation.com/ActiveFacts/

14. ter Hofstede, A., Proper, H., van der Weide, T.: Formal definition of a conceptual language for the description and manipulation of information models. Information Systems 18(7), 489–523 (1993)
15. Meersman, R.: The RIDL Conceptual Language. In: Int. Centre for Information Analysis Services, Control Data Belgium, Brussels (1982)
16. Nijssen, M., Lemmens, I.M.C.: Verbalization for Business Rules and Two Flavors of Verbalization for Fact Examples. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM-WS 2008. LNCS, vol. 5333, pp. 760–769. Springer, Heidelberg (2008)
17. NORMA (Natural ORM Architect) tool download site for public-domain version, http://www.ormfoundation.org/files/folders/norma_the_software/default.aspx
18. Object Management Group 2009, UML 2.2 Specifications, http://www.omg.org/uml
19. Object Management Group 2008, Semantics of Business Vocabulary and Business Rules (SBVR), http://www.omg.org/spec/SBVR/1.0/

Alaska Simulator Toolset for Conducting Controlled Experiments on Process Flexibility

Barbara Weber[1], Jakob Pinggera[1], Stefan Zugal[1], and Werner Wild[2]

[1] Quality Engineering Research Group, University of Innsbruck, Austria
{Barbara.Weber,Jakob.Pinggera,Stefan.Zugal}@uibk.ac.at
[2] Evolution Consulting, Innsbruck, Austria
werner.wild@evolution.at

Abstract. Alaska Simulator Toolset (AST) is a software suite developed at the University of Innsbruck for exploring different approaches to business process flexibility by using a familiar metaphor, i.e., travel planning and execution. For this, AST provides integrated support of several decision deferral based approaches to process flexibility and enables their systematic comparison. Moreover, AST facilitates the design and execution of controlled experiments through experimental workflow support. This paper introduces AST, explains its underlying meta-model, discusses factors which can be investigated using AST and shows how it can be used for designing and conducting experiments.

1 Introduction

Alaska Simulator Toolset (AST) has been developed to address the investigation of strengths and weaknesses of different approaches to flexibility in Process-aware information systems (PAIS). In particular, AST allows a systematic comparison of these approaches through the execution of controlled experiments.

A critical success factor in applying a PAIS is the option to flexibly deal with process changes [3,4] (e.g., to react to changes, like shifts in customers' attitudes or the introduction of new laws [1]). To address the need for flexible PAISs, competing paradigms enabling process changes and process flexibility have been introduced, e.g., adaptive processes [5], case handling [6], declarative processes [7], and late binding and modeling [8] – for an overview see [9]. While adaptive processes provide flexibility by supporting structural *process adaptations*, flexibility can also be achieved through loosely specified process models which enable *deferring decisions* about the exact process structure from modeling time to run-time [9], when more information is available. By deferring decisions to run-time the strict separation of build-time (i.e., modeling or planning) and run-time (i.e., execution), which is typical for traditional workflow management systems, can be relaxed and planning and execution become more closely interwoven. Unfortunately, however, little research exists so far on the suitability of respective approaches depending on the characteristics of business processes to be supported. In particular, there is a lack of empirical insights on the selection of an appropriate approach.

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 205–221, 2010.
© Springer-Verlag Berlin Heidelberg 2010

To pick up this need this paper introduces AST, which is used for the systematic comparison of different approaches to decision deferral. Section 2 elaborates on the support for different approaches for deferring decisions and describes the underlying meta-model. Section 3 discusses several factors which have a potential impact on the suitability for these approaches. Section 4 then describes how AST can be used for designing and executing controlled experiments. Related work is listed in Section 5, Section 6 concludes the paper with a summary and an outlook.

2 Alaska Simulator Toolset

To foster the comparison of different approaches for process flexibility Alaska Simulator Toolset[1] has been implemented, which takes a *journey* as metaphor for a *business process*. Section 2.1 summarizes the different approaches for process flexibility and decision deferral supported by Alaska Simulator, while Section 2.2 introduces its meta-model and Section 2.3 discusses the journey metaphor.

2.1 Support for Decision Deferral Patterns in Alaska Simulator

To foster the comparison of different approaches to process flexibility, Alaska Simulator provides integrated support for the decision deferral patterns described in [9] (cf. Fig. 1).

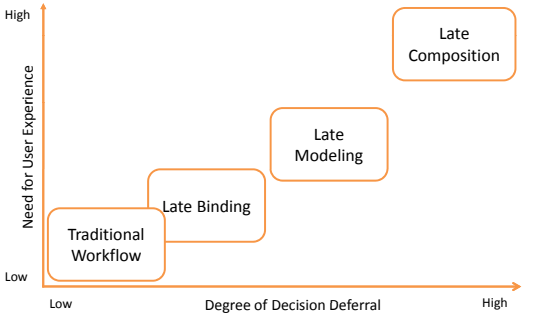

Fig. 1. Different Planning Approaches

Traditional workflow management systems offer little flexibility and require to completely predefine a business process in advance, thus employing a strictly plan-driven approach separating modeling and execution entirely (cf. Fig. 2A). *Plan-driven approaches* try to overcome uncertainty by developing a very detailed plan up-front, which is then regarded as the schema for execution. However, as plans tend to be more imprecise the earlier they are elaborated [18],

[1] Developed at the University of Innsbruck, `http://www.alaskasimulator.org`

concentrating all planning efforts in the planning phase imposes the risk of rendering plans useless due to changing conditions (e.g., changing requirements) [1].

To address this risk both *Late Binding* and *Late Modeling* offer slightly more flexibility by allowing for placeholder activities which can be refined during run-time (cf. Fig. 2B). Using *Late Binding* the selection of the actual content for the placeholder activity (from a set of predefined fragments) can be deferred to run-time. The *Late Modeling* pattern goes one step beyond this and allows for the modeling (not only selection) of the placeholder content at run-time. Using *Late Binding* or *Late Modeling* the strict separation of modeling and execution is slightly relaxed allowing for modeling activities during run-time, but restricted to pre-specified parts in the process, i.e., to the placeholder activities (cf. Fig. 2B).

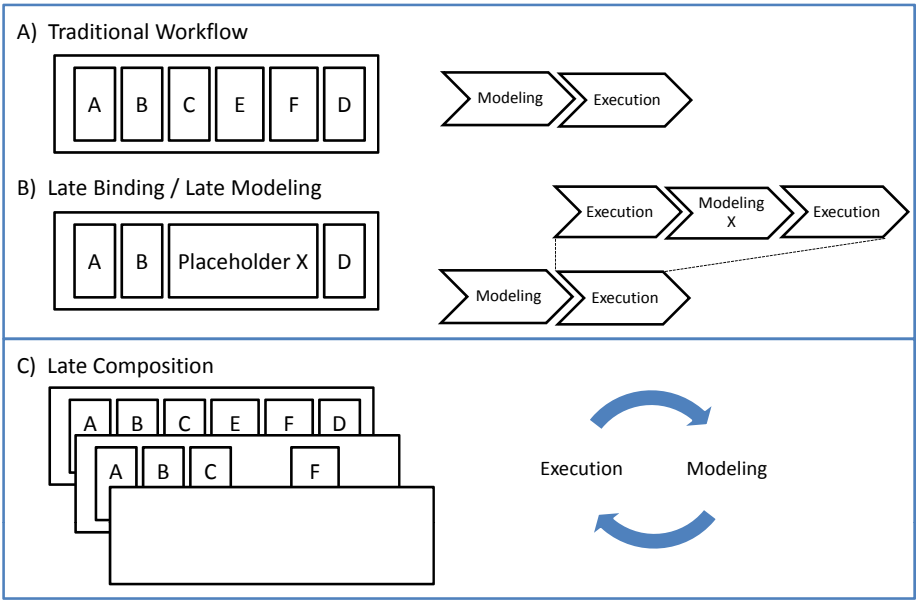

Fig. 2. Planning and Execution Modes

Most flexibility is offered by the *Late Composition* pattern, which allows users to iteratively compose a business process (or a journey respectively) by selecting activities from a repository, while respecting all existing constraints. By employing iterative and continuous planning *Late Composition* mitigates the risk of rendering plans useless due to changing conditions [1,19,20] (cf. Fig. 2C). In contrast to the plan-driven approach (as employed by traditional workflow management systems) no detailed up-front plan is created, the initial plan remains more coarse grained [19]. Over time the initial plan is iteratively refined, allowing for the integration of newly gained knowledge. In contrast to plan-driven approaches, which focus on the plan as an artifact by itself, the emphasis is put on planning as an ongoing activity. Distributing planning effort over the entire

duration enables deferring decisions to the last responsible moment [1] and to create opportunities for learning. When this last responsible moment actually will be, highly depends on the characteristics of a business process. Figure 3 classifies activities according to their availability and reliability and shows strategies on how to best deal with particular classes of activities.

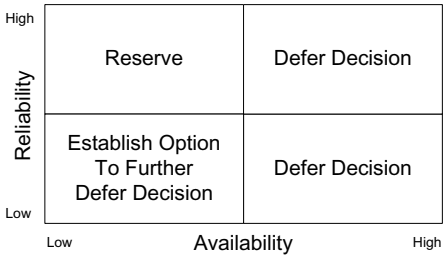

Fig. 3. Deciding at the Last Responsible Moment

Fig. 3 shows that committing to highly available activities (i.e., by making a resource reservation) should be avoided. Reserving respective activities introduces unnecessary early commitment without gaining any benefits. In contrast, activities with high reliability in combination with low availability should be reserved before the reservation deadline to guarantee resource availability (i.e., the last responsible moment would be just before the reservation deadline). Activities with low availability and low reliability are most difficult to handle. Depending on the business value that can be gained and the cost of the respective activity, it might be worth to commit to the activity and to schedule a second more reliable activity in parallel. Therefore, instead of choosing between one of the activities right away an option is build allowing to defer the last responsible moment for deciding between the alternatives.

How much pre-planning is "enough" highly depends on the characteristics of the business process. In the most extreme cases a fully pre-specified plan like in traditional workflow management systems (i.e., if all activities have a low availability and a high reliability) or an empty initial plan (i.e., if the availability is high) might be most appropriate.

2.2 Alaska Simulator Meta-model

This section describes the meta-model underlying Alaska Simulator (cf. Fig. 4).

Goal. When conducting a journey with AST, the goal is to maximize the travel experience (i.e., the overall "business value" of the journey). For optimizing the execution of a journey, information about the benefits (i.e., business value), costs and duration of actions, but also about their availability and reliability is essential.

Journey, Journey Plan, Locations. Alaska Simulator supports users in composing *journeys* based on a pre-specified *journey configuration* (i.e., instance of the meta-model) on a *journey plan.* A journey plan is represented by a calendar

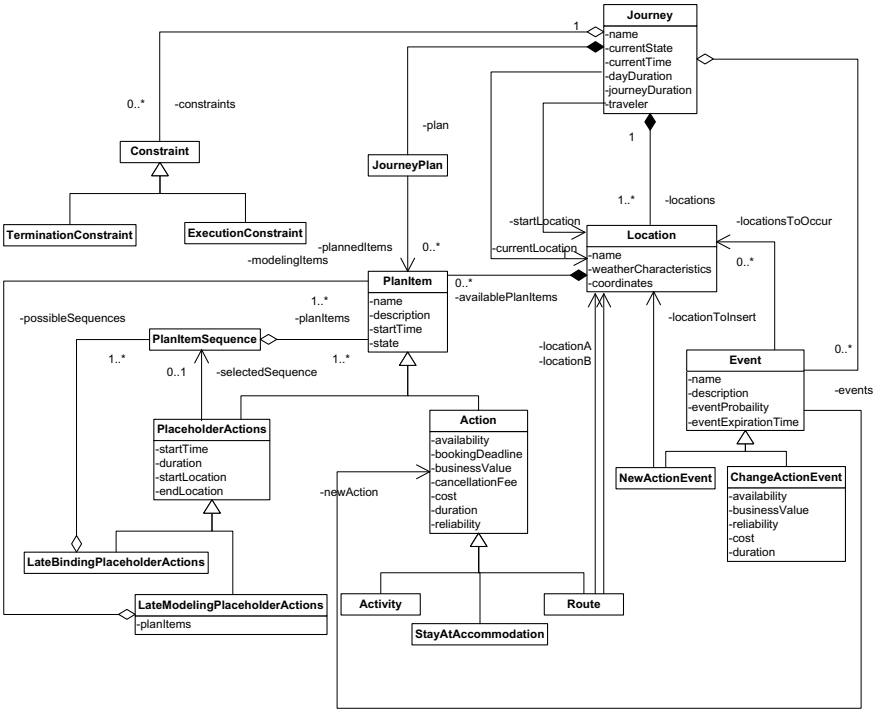

Fig. 4. Alaska Simulator Meta-Model

for scheduling *plan items* (i.e., activities, routes or accommodations). During a journey different *locations* can be visited, each providing distinct *plan items* to select from (e.g., activity `Visit Denali National Park` is only available a location `Denali`).

Plan Items. Plan Items can be *atomic actions* or *complex placeholder actions*. *Atomic actions* can either be *activities*, *routes* or *stays at accommodations*. For each atomic action its "business value", costs, duration, availability and reliability are specified. Moreover, a booking deadline as well as cancellation fees are known. *Complex placeholder actions*, in turn, provide placeholders which can be refined during the execution of a journey. *Late Binding Placeholder Actions* provide a set of *Plan Item Sequences* from which one can be selected during run-time. *Late Modeling Placeholder Actions*, in turn, allow users to compose a *Plan Item Sequence* during run-time from a set of predefined plan items respecting existing constraints. *Plan items* in general, in addition to their name and a description, have a start time (which is determined by their position in the journey plan) and a particular state (e.g., scheduled, booked, started, executed). A state transition diagram depicting the lifecycle of an action is illustrated in Fig. 5.

Constraints. When composing a journey different constraints have to be considered. Constraints can be classified as execution and termination constraints.

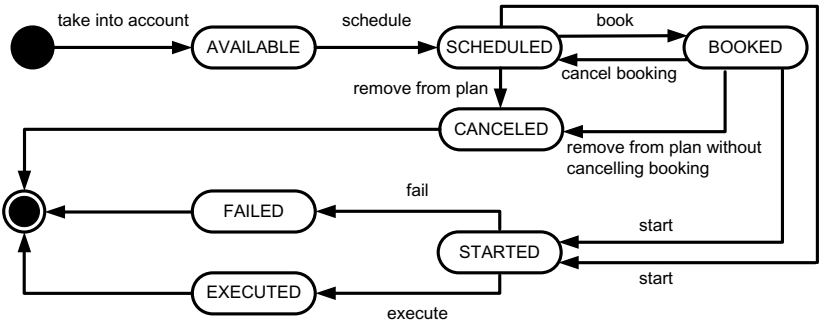

Fig. 5. Action Lifecycle

Execution constraints restrict the execution of activities (e.g., an activity can be executed at most once, or two activities are mutually exclusive). *Termination constraints*, in turn, affect the termination of process instances and specify when process termination is possible. For instance, an activity must be executed at least once before the process can be terminated or a particular end location must be reached for terminating the journey. Alaska Simulator provides support for all control-flow constraints described in [13] (e.g., a *response constraint* requiring that activity `A` is eventually followed by `B` or a *mutual exclusion constraint* prohibiting that activity `A` and `B` co-occur). In addition, several resource and time related constraints are supported (e.g., a *budget constraint* restricting the available travel budget, an *end location constraint* requiring that the journey ends at a particular location, a *completion day constraint* restricting the journey to a particular duration, an *execution time constraint* restricting the time when a particular activity can be executed, and a *time lag constraint* requiring a minimum time-lag between executing two activities).

Uncertainty. Journeys are characterized by incomplete information prior to execution. The outcome of an action within a journey plan is not predefined and varies with the *weather conditions* encountered. The degree of variation is defined by the action's *reliability*, i.e., low reliability indicates that the outcome of an action is highly weather dependent. The overall business value of a journey (i.e., a numeric value representing the travel experience) is calculated as the sum of business values of all performed actions. Prior to performing the journey only the *expected business value* for each action as well as its reliability are known. The expected business value which is calculated based on the action's reliability and the average weather characteristics of the location where the action is available. During the journey the action's *actual business value* is calculated based on the actual weather conditions encountered.

Resource Scarcity. When planning a journey potential resource scarcity has to be considered. By firmly *booking* an action, its *availability* can be guaranteed, but the cost of the action must be paid immediately. If the booking is canceled during the journey, a *cancellation penalty* might apply, thus making too early

commitments costly. Furthermore, booking is only possible up to a certain time before executing the action, as specified by the *booking deadline*.

Events. In addition to changing weather conditions, *unforeseen events* (e.g., a traffic jam resulting in delays) create uncertainty in a journey. On the one hand, a *ChangeActionEvent* changes values of existing actions during the journey. For example, a particular event might increase or decrease the business value, the availability, the costs or the duration of an action. On the other hand, a *NewActionEvent* allows to introduce new actions at a particular location during run-time. Moreover, events occur with a predefined probability during run-time, may have an expiration time and can be attached to a particular location (i.e., the event only occurs when the traveler is at a particular location).

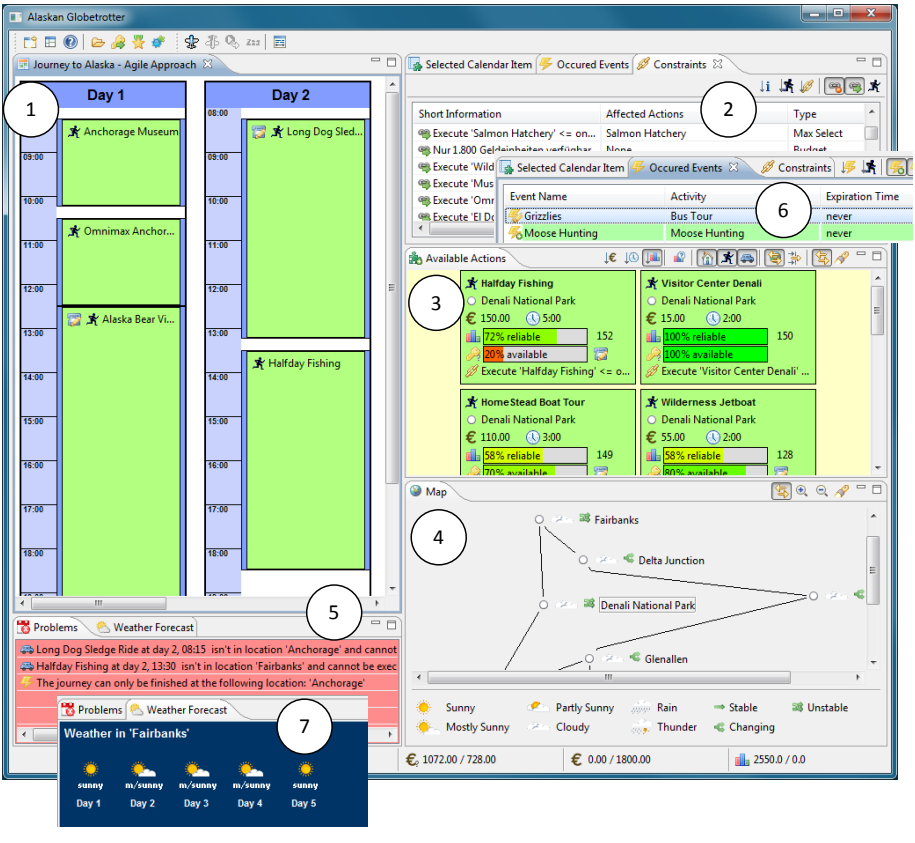

Fig. 6. Screenshot of Alaska Simulator

Fig. 6 depicts the graphical user interface of the Alaska Simulator. Users can compose their individual journey plan by dragging available actions from the Available Actions View (3) onto the travel itinerary (i.e., journey plan) (1). Actions are usually only available at a particular location on the map (4).

Existing constraints are displayed in the Constraint View (2) and have to be considered when composing a concrete journey plan. After each user action, the journey plan is validated and the user is informed about any constraint violations and plan inconsistencies (5). Unforeseen events are shown in the Event View (6). Prior to executing the journey users have to rely on weather statistics derived from the weather characteristics of each location, which can be selected on a map (4). During the journey a weather forecast is provided with the current weather conditions at each location (7).

2.3 The Journey Metaphor

This section discusses the journey metaphor and shows how the concepts described in the previous section relate to business processes.

While the *goal* of a journey is to maximize "travel experience", typical goals for business processes are minimizing costs, cycle times or the optimization of quality or customer satisfaction [14].

Journeys are composed based on *journey configurations*, likewise *process instances* representing concrete business cases based on a predefined *process schema*. Both in journeys and business processes certain activities are only available at certain `locations` (e.g., in a medical process surgeries can only be performed in an operating room). While journeys are composed from activities, routes and accommodations, business processes comprise a set of activities. In addition to *atomic activities*, *complex placeholder activities* (allowing for Late Binding and Late Modeling) can also be found in business processes.

When composing a journey, or a business process respectively, existing *contraints* need to be considered. Thereby, support provided by Alaska Simulator is comparable to existing declarative process management systems like DECLARE [13].

Like journeys, highly flexible business processes are characterized by incomplete information prior to execution. In a journey, *uncertainty* is, for example, caused by unforeseen weather conditions, while uncertainty in business processes refers to the difficulty to predict the exact activities and resources that are required to perform a particular process (e.g., due to changing requirements caused by dynamic changes of the business environment) [15].

Both, when planning a journey or executing a business process, potential *resource scarcity* has to be considered (e.g., appointments in medical processes to deal with the limited availability of doctors) [16].

Unforeseen events have to be handled when conducting journeys (e.g., a traffic jam resulting in delays), or executing a business process (e.g., changing requirements or an allergic reaction during a medical treatment) [17].

Motivation. The journey metaphor has been chosen since the domain of travel planning is familiar to the majority of experimental subjects. Furthermore, journey planning is an attractive context for many people to become engaged in, which significantly improves their willingness to use the system for experimental purposes and as a consequence increases internal validity of data.

Limitations. Using a metaphor, however, imposes also several limitations. While the concepts underlying Alaska simulator are mostly comparable to the ones in PAISs, the domain under investigation is fixed to travel planning. Generalizability to other domains (which are less familiar to the experimental subjects) has to be investigated. Another limitation relates to the fact that Alaska Simulator, even though it allows scheduling of actions in parallel, currently does not support the parallel execution of actions (since this does not naturally fit the travel metaphor). As a consequence, the findings obtained with Alaska Simulator, should be complemented with field studies for further validation.

3 Factors for Investigating Different Flexibility Approaches

In the previous sections we have introduced the concepts underlying Alaska Simulator (cf. Section 2.2), explained the journey metaphor (cf. Section 2.3) and illustrated how it supports the different decision deferral patterns (cf. Section 2.1). This section discusses different factors having an impact on the suitability of a particular approach for decision deferral, which can be investigated using Alaska Simulator. These factors can be grouped into two categories: configuration characteristics and personal characteristics.

3.1 Configuration Characteristics

Alaska Simulator allows researchers to investigate the impact of different configuration characteristics on the suitability of a particular approach for decision deferral. Furthermore, Alaska Simulator can be used to compare different approaches for particular configuration characteristics.

- **Actions.** The different attributes of the actions in the journey configuration are a potential factor influencing the suitability of a particular approach. For example, planning and executing a business process or a journey respectively and in particular resolving exceptional situations might be easier if the configuration comprises many small actions in contrast to a few very large actions.
- **Constraints.** The suitability of a particular approach for planning and executing a business process or a journey respectively might be affected by the number and type of constraints. Moreover, the presence of global constraints might have a potential negative impact on the applicability of *Late Binding* or *Late Modeling* respectively.
- **Events.** The probability of unforeseen events that can occur during run-time might be another factor affecting suitability of the different approaches.
- **Uncertainty.** The selection of a suitable approach might also depend on the *reliability* of the different activities. With increasing uncertainty decision deferral might become more important.
- **Resource Scarcity.** Suitability might also be affected by the *availability* of activities. While availability of resources is facilitating decision deferral, resource scarcity is creating significant challenges.

3.2 Personal Characteristics

Even though the advantages of applying decision deferral techniques are apparent, especially for highly uncertain environments – they foster both *learning* (i.e., the ability to resolve uncertainty) and *flexibility* (i.e., the ability to exploit learning outcomes) [21], their successful application also requires user experience. In particular, as illustrated in Fig. 1, with increasing flexibility the need for user support increases. To fully exploit the benefits, adopters must not only have the skills to make use of decision deferral techniques, to be able to determine the last responsible moment and to know how and when to build options (i.e., to create opportunities for learning), but also be able to leverage learning results. Literature suggests that talent and skills are among the critical people-factors for agile methods [22,23], which tend to necessitate a richer mix of higher-skilled people than plan-driven approaches [24]. This raises the question of how well inexperienced users can employ respective planning techniques and whether these techniques can outperform plan-driven approaches for a lower level of expertise.

Novices and experts might not only differ in their ability to defer decision to run-time, but also in handling unforeseen events or obeying complex constraints. AST allows, for example, to investigate to what extent novice users and experts differ and how the novice user's modeling approach changes when uncertainty increases.

4 Designing and Executing Experiments with Alaska Simulator Toolset

This section describes details about the functionality AST provides for supporting researchers in designing and conducting experiments. AST consists of three major components: *Alaska Configurator*, *Alaska Simulator* and *Alaska Analyzer*.

- **Alaska Configurator** allows researchers to design journey configuration (including locations, actions, events and constraints as well as the degree of uncertainty and resource scarcity) executable through Alaska Simulator. In addition, Alaska Configurator supports researchers in designing experimental workflows guiding subjects during the experiment.
- **Alaska Simulator** allows to plan and execute journeys making use of different decision deferral approaches. Thereby, each step performed while planning and executing a journey is logged in an MXML log file for later analysis (e.g., using the process mining tool ProM [25]).
- **Alaska Analyzer** allows detailed manual analysis of planning behavior by supporting replay of journeys step by step.

In the following we describe main functionalities of AST and how design and execution of controlled experiments is supported, using the experiment described in [26] as our example (cf. Fig. 7 for the experimental design).

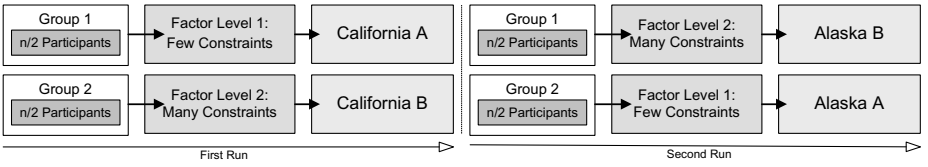

Fig. 7. Experimental Design

Alaska Configurator was used to design two journey configurations (California and Alaska) including locations, actions, events, constraints as well as variability of weather conditions (cf. Fig. 8 and Fig. 9). For each journey configuration two variants were created, one for each factor level (i.e., few and many constraints). To gather the participants' demographic information Alaska Configurator was used for designing a survey (cf. Fig. 10). The journey configurations and the survey were then assembled to an experimental workflow (cf. Fig. 11).

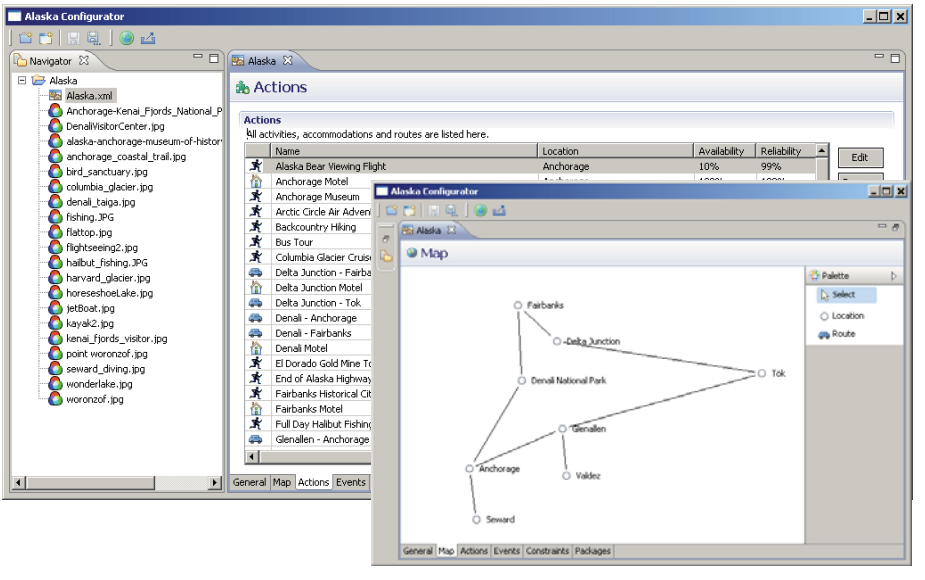

Fig. 8. Defining Actions and Locations with Alaska Configurator

During experiment execution participants were guided by the experimental workflow. After presenting them with a survey (cf. Fig. 12), half the students obtained configuration California with few constraints, while the other half obtained the same configuration with many constraints. The students then planned and executed a journey to California. Each step that was performed while planning and executing was logged for later investigation and detailed analysis. Having completed their California journeys, subjects planned and executed a journey to Alaska.

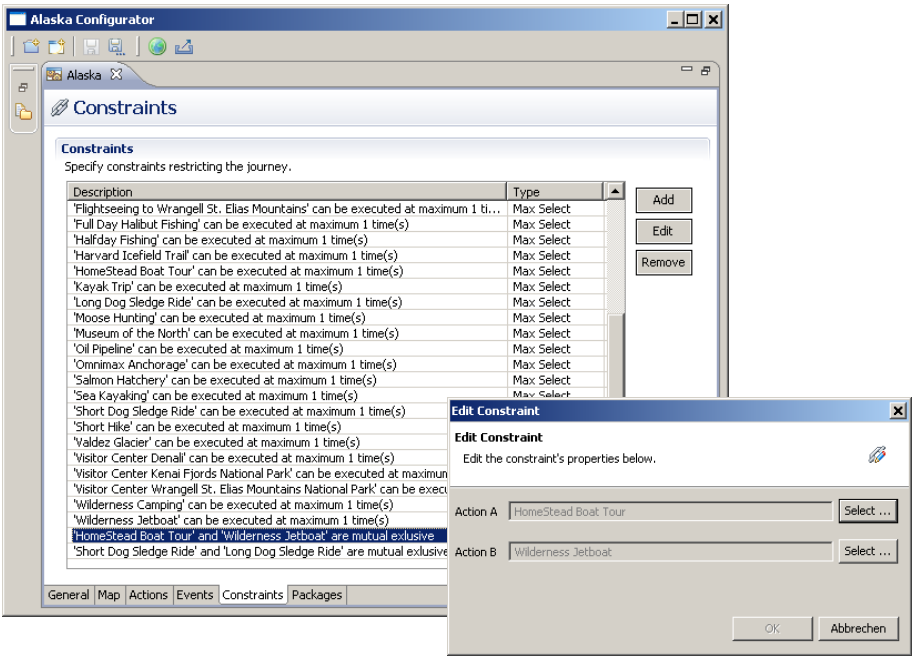

Fig. 9. Defining Constraints with Alaska Configurator

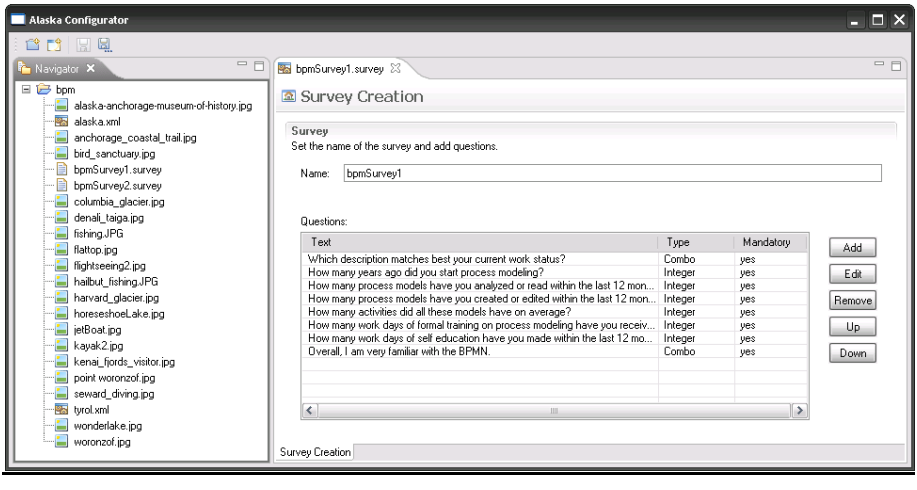

Fig. 10. Questionnaire Builder

After the planning session researchers were supported in analyzing the journeys by enabling them to replay each journey step by step, using Alaska Analyzer (cf. Fig. 13).

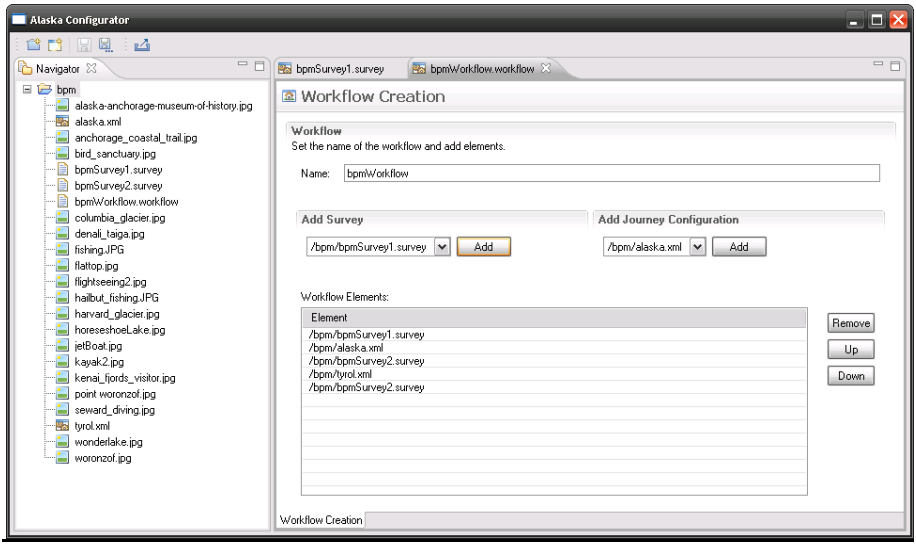

Fig. 11. Designer for Experimental Workflows

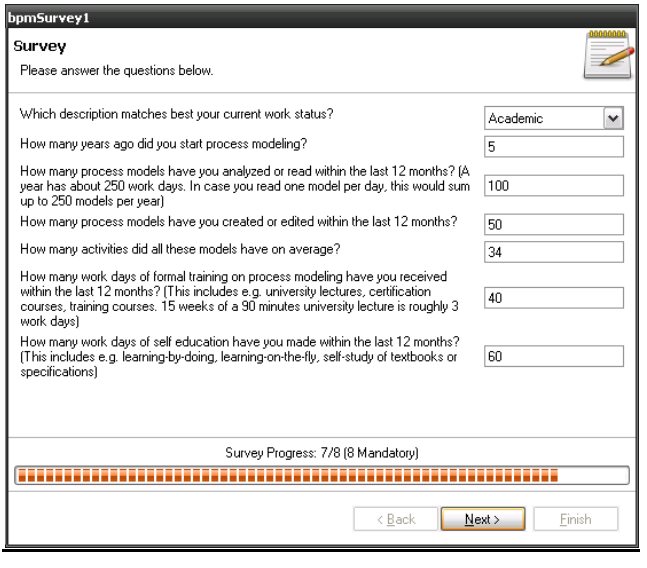

Fig. 12. Survey Generated Using Questionnaire Builder

Alaska Simulator, including a test configuration, extensive documentation and screencasts can be downloaded from http://www.alaskasimulator.org. For detailed information on the results of controlled experiments conducted using Alaska Simulator we refer to [26,27].

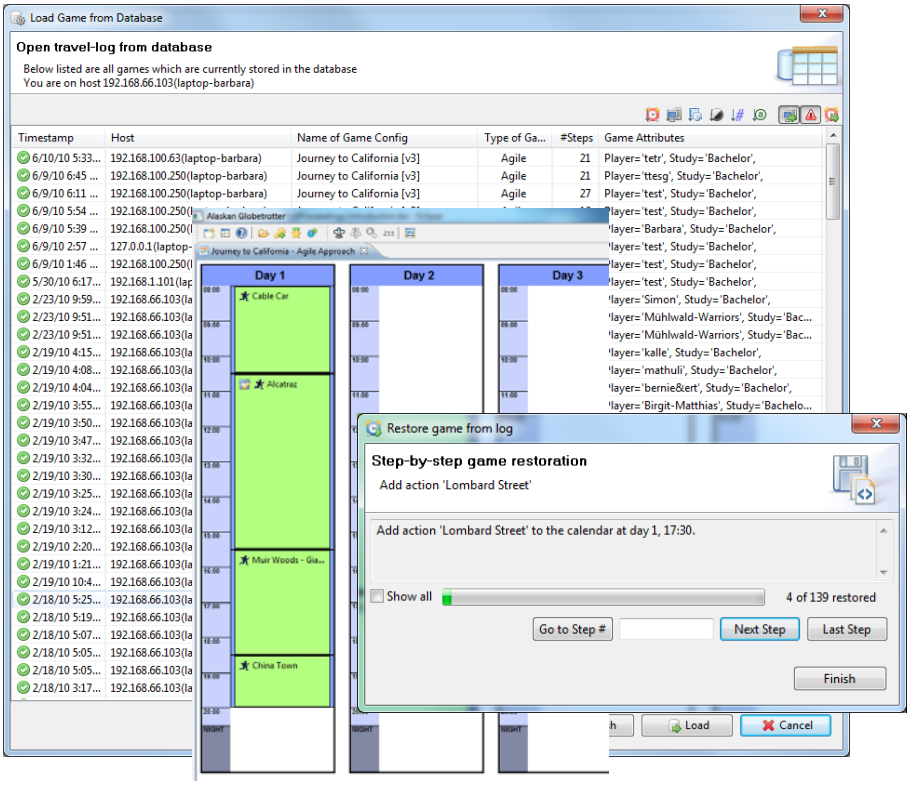

Fig. 13. Replaying Journeys with Alaska Analyzer

5 Related Work

Most existing work about flexibly dealing with exceptions, changes, and uncertainty in the context of PAISs and related technologies is strongly *design-centered*, i.e., aiming at the development of tools, techniques, and methodologies. For overviews and discussions of these approaches, see [9,28,29].

Only few empirical investigations exist that aim to establish the suitability of the various proposed artifacts. Closely related to this paper is our previous work, which provides empirical insights into the use of declarative processes [26]. In particular, it investigates how well users can cope with the gained flexibility provided by declarative approaches, especially when processes become rather complex in terms of constraints. The effect of events on how end users can cope with declarative approaches, in turn, is investigated in [27]. A theoretical discussion on declarative versus imperative approaches is provided in [30]. In [10], the results of a controlled experiment comparing a traditional workflow management system and case-handling are described. The systems are compared with respect to their associated implementation and maintenance efforts. In turn,

the impact of workflow technology on PAIS development and PAIS maintenance is investigated in [31]. Other empirical works with respect to PAISs mainly deal with establishing their contribution to business performance improvement, e.g. [32,33], and the way end-users appreciate such technologies, e.g., [34,35].

Worth mentioning here is a stream of research that relates to so-called *change patterns* [9]. It provides a framework for the qualitative comparison of existing flexibility approaches.

6 Conclusion

Alaska Simulator Toolset supports researchers in systematically comparing different approaches for decision deferral by means of controlled experiments. This paper describes the concepts underlying Alaska Simulator and shows how AST supports the different patterns for decision deferral. In addition, the paper discusses factors having a potential impact on the suitability of a particular approach for decision deferral, which can be investigated using AST. Moreover, the paper demonstrates how AST supports researchers in conducting controlled experiments. In particular, *Alaska Configurator* allows researchers to design journey configuration executable with Alaska Simulator and supports them in designing experimental workflows guiding users during the experiment. *Alaska Simulator* allows to plan and execute journeys making use of different approaches for decision deferral. *Alaska Analyzer*, in turn, supports detailed analysis of data and replaying journeys step by step.

Future work includes the planning and execution of additional controlled experiment with AST. On the tool side, the development of a dashboard simplifying the supervision of experiments is planned.

Acknowledgements. We thank Felix Schöpf for his work on the experimental workflow support, and Michael Schier on the Late Binding and Late Modeling Support in Alaska Simulator.

References

1. Poppendieck, M., Poppendieck, T.: Implementing Lean Software Development. Addison Wesley Longman, Amsterdam (2006)
2. Weske, M.: Business Process Management: Concepts, Methods, Technology. Springer, Heidelberg (2007)
3. Van der Aalst, W., Jablonski, S.: Dealing with workflow change: Identification of issues an solutions. Int'l Journal of Comp. Systems, Science and Engineering 15, 267–276 (2000)
4. Mutschler, B., Reichert, M., Bumiller, J.: Unleashing the effectiveness of process-oriented information systems: Problem analysis, critical success factors and implications. IEEE Trans. on Systems, Man, and Cybernetics 38, 280–291 (2008)
5. Reichert, M., Dadam, P.: ADEPT$_{flex}$ – Supporting Dynamic Changes of Workflows Without Losing Control. JIIS 10, 93–129 (1998)

6. Van der Aalst, W., Weske, M., Grünbauer, D.: Case handling: A new paradigm for business process support. Data and Knowledge Engineering 53, 129–162 (2005)
7. Pesic, M., Schonenberg, M., Sidorova, N., van der Aalst, W.: Constraint-Based Workflow Models: Change Made Easy. In: Proc. CoopIS 2007, pp. 77–94 (2007)
8. Sadiq, S., Sadiq, W., Orlowska, M.: A Framework for Constraint Specification and Validation in Flexible Workflows. Information Systems 30, 349–378 (2005)
9. Weber, B., Reichert, M., Rinderle-Ma, S.: Change patterns and change support features -enhancing flexibility in process-aware information systems. In: Data and Knoweldge Engineering, pp. 438–466 (2008)
10. Weber, B., Mutschler, B., Reichert, M.: Investigating the Effort of Using Business Process Management Technology: Results from a Controlled Experiment. Science of Computer Programming 75, 292–310 (2009)
11. Myers, G.J.: A Controlled Experiment in Program Testing and Code Walkthroughs/Inspections. ACM Commun. 21, 760–768 (1978)
12. Lott, C.M., Rombach, H.D.: Repeatable Software Engineering Experiments for Comparing Defect-Detection Techniques. Emp. Software Eng. 1, 241–277 (1996)
13. van der Aalst, W.M.P., Pesic, M.: DecSerFlow: Towards a truly declarative service flow language. In: Bravetti, M., Núñez, M., Tennenholtz, M. (eds.) WS-FM 2006. LNCS, vol. 4184, pp. 1–23. Springer, Heidelberg (2006)
14. Reijers, H.: Design and Control of Workflow Processes. LNCS, vol. 2617. Springer, Heidelberg (2003)
15. Gebauer, J., Schober, F.: Information system flexibility and the cost efficiency of business processes. J. of the Assoc. for Information Systems 7, 122–147 (2006)
16. Mans, R.S., Russell, N.C., van der Aalst, W.M.P., Bakker, P.: Schedule-aware workflow management systems. In: Proc. PNSE 2009, pp. 81–96 (2009)
17. Strong, D.M., Miller, S.M.: Exceptions and exception handling in computerized information processes. ACM Trans. Inf. Syst. 13, 206–233 (1995)
18. Boehm, B.W.: Software Engineering Economics. Prentice Hall, Englewood Cliffs (1981)
19. Cohn, M.: Agile Estimating and Planning. Prentice Hall Professional, Englewood Cliffs (2006)
20. Beck, K.: eXtreme Programming Explained. Addison-Wesley Longman, Amsterdam (2000)
21. Erdogmus, H.: The Economic Impact of Learning and Flexibility on Process Decisions. IEEE Software 22, 76–83 (2005)
22. Highsmith, J., Cockburn, A.: Agile Software Development: The Business of Innovation. Computer 34, 120–122 (2001)
23. Lindvall, M.: Empirical Findings in Agile Methods. In: Proc. XP/Agile Universe 2002, pp. 197–207 (2002)
24. Boehm, B.W., Turner, R.: Balancing Agility and Discipline: A Guide for the Perplexed. Addison Wesley Professional, Reading (2003)
25. van der Aalst, W.M.P., Reijers, H.A., Weijters, A.J.M.M., van Dongen, B.F., de Medeiros, A.K.A., Song, M., Verbeek, H.M.W.E.: Business process mining: An industrial application. Inf. Syst. 32, 713–732 (2007)
26. Weber, B., Reijers, H.A., Zugal, S., Wild, W.: The declarative approach to business process execution: An empirical test. In: van Eck, P., Gordijn, J., Wieringa, R. (eds.) CAiSE 2009. LNCS, vol. 5565, pp. 470–485. Springer, Heidelberg (2009)
27. Weber, B., Pinggera, J., Zugal, S., Wild, W.: Events during business process execution: An empirical test. In: Proc. ER-POIS 2010 (2010)
28. Kammer, P., Bolcer, G., Taylor, R., Hitomi, A., Bergman, M.: Techniques for Supporting Dynamic and Adaptive Workflow. CSCW 9, 269–292 (2000)

29. Reijers, H., Rigter, J., van der Aalst, W.: The case handling case. Int'l J. of Cooperative Information Systems 12, 365–391 (2003)
30. Fahland, D., Lübke, D., Mendling, J., Reijers, H.A., Weber, B., Weidlich, M., Zugal, S.: Declarative versus imperative process modeling languages: The issue of understandability. In: Proc. BPMDS 2009 and EMMSAD 2009. LNBIP, vol. 29, pp. 353–366 (2009)
31. Kleiner, N.: Supporting usage-centered workflow design: Why and how? In: Desel, J., Pernici, B., Weske, M. (eds.) BPM 2004. LNCS, vol. 3080, pp. 227–243. Springer, Heidelberg (2004)
32. Oba, M., Onoda, S., Komoda, N.: Evaluating the quantitative effects of workflow systems based on real case. In: Proc. HICSS 2000 (2000)
33. Reijers, H., van der Aalst, W.: The effectiveness of workflow management systems: Predictions and lessons learned. Int'l J. of Inf. Management 25, 458–472 (2005)
34. Bowers, J., Button, G., Sharrock, W.: Workflow from within and without: technology and cooperative work on the print industry shopfloor. In: Proc. CSCW 1995, pp. 51–66 (1995)
35. Poelmans, S.: Workarounds and distributed viscosity in a workflow system: a case study. ACM SIGGROUP Bulletin 20, 11–12 (1999)

Facing the Challenges of Genome Information Systems: A Variation Analysis Prototype

Ana M. Martínez, Ainoha Martín, Maria José Villanueva,
Francisco Valverde, Ana M. Levin, and Oscar Pastor

Centro de Investigación en Métodos de Producción de Software
Universidad Politécnica de Valencia
Camino de Vera S/N 46022, Valencia, Spain
{amartinez,amartin,mvillanueva,fvalverde,alevin,opastor}@pros.upv.es
http://www.pros.upv.es

Abstract. In Bioinformatics there is a lack of software tools that fit with the requirements demanded by biologists. For instance, when a DNA sample is sequenced, a lot of work have to be performed manually and several tools are used. The application of Information Systems (IS) principles into the development of bioinformatics tools opens a new interesting research path. One of the most promising approaches is the use of conceptual models in order to precisely define how genomic data is represented into an IS. This work introduces how to build a Genome Information System (GIS) using these principles. As a first step to achieve this goal, a conceptual model to formally describe genomic mutations is presented. In addition, as a proof of concept of this approach, a variation analysis prototype has been implemented using this conceptual model as a development core.

1 Introduction

In 1953, James D. Watson and Francis Crick described the DNA structure as a "double helix"[1]. In 1990, the Human Genome Project [2] officially begins and twenty-three years later, in 2003, all the effort is rewarded and the whole sequence appears [3,4]. The availability of this new information opens the door to the creation of new ways of diagnosis, new types of medicines, new therapy strategies, new studies, etc.

Thanks to this breakthrough and the advances in DNA sequencing, a big amount of genetic data is being produced by researchers every day. Most of these experiments are focused on the understanding of the relationship between genotype (gene configuration and combination of a particular individual) and its phenotype (expression of the genes in a specific human feature). As a consequence, the creation of biological databases and tools to exploit the data produced has grown drastically. However, these tools and databases have usually been defined to support a specific research area or experiment. Therefore, when biologists want to use them for a particular essay, it is very unlikely that they support their specific requirements.This leads to a situation where the researcher

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 222–237, 2010.
© Springer-Verlag Berlin Heidelberg 2010

has to spend a lot of time and effort to perform a simple analysis. Since these bioinformatics tools are not developed using IS principles, they are not aligned with the user requirements. The main consequences of this issue are:

- Some biological databases are only human readable, thus cannot be processed properly in an automatic way.
- The extraction of relevant data is difficult because it is spread around different databases.
- Since several tools are required to analyze the data, the tooling workflow specification and integration is far from trivial.
- Inclusion of new studies and bibliography into the available tools turns into a hard task.

To solve these questions, some researchers have proposed [5] the development of Genomic Information Systems (GIS), specifically designed IS capable to handle a big amount of genomic data. In this work, a new approach to develop GIS is proposed: the use of conceptual models to define and organize the genomic data in a formal way.

Thanks to the close collaboration with biologists in the context of this work, the gap between the disciplines of software engineering and genetics can be solved. The result of this interdisciplinary collaboration is the design of a conceptual model that guides the alignment of concepts among both fields. Therefore the design and implementation of the software artifacts that made up a GIS becomes an easier process.

Following that idea, this paper presents a GIS prototype that analyzes DNA sequences and compares them to the reference in order to find variations for a specific gene. Once all variations are located in the sequence, the prototype splits them into two groups: one group contains harmless variations and the other one contains variations that produce a change in gene or protein function. For those in the last group, their specific phenotype is reported as it has been described in the literature.

This information is bibliographically referenced and gathered in a report that helps the researcher to understand the genetic meaning of the variation and why it produces a certain phenotype. This is very useful because it can speed up the diagnosis of a specific disease. Furthermore, it is widely accepted that an early detection of a disease might be determinant.

The rest of the paper is organized as follows. In section 2 a review of DNA variation analysis tools is presented. Section 3 details a conceptual model to describe genomic mutations. Section 4 describes how the variation analysis prototype has been developed. Finally, in section 5 conclusions and future work are stated.

2 Related Work

There are two approaches to perform a DNA sequence analysis: genotyping, which analyzes small DNA fragments, and sequencing, which analyzes the whole

genome. In this section, some tools that use these approaches are analyzed. The majority of the studied tools use genotyping, which means that in the majority of the cases relevant information located at unexplored regions is ignored. To solve this problem, the presented prototype uses the sequencing approach to detect all the variations produced in one gene.

In recent years, several commercial tools have been developed to provide genomic analysis. These tools perform tests to estimate the probability of the customer to suffer certain diseases. Navigenics [6],23andMe [7], deCODEme [8], DNADirect [9] and Knome [10] are the most relevant tools in this field. The differences between them are briefly summarized in Table 1. 23andMe, Navigenics, DNADirect and deCODEme are tools that help their clients to make decisions about their health providing information such as: their probability of suffering certain disease, the mutations that could affect their family future and its own or the possibility of finding what pharmacotherapy is best suited for their organism. These mentioned services use genotyping to analyze the sequence. This approach does not analyze the whole sequence, only small fragments, as for example SNPs (Single Nucleotide Polymorphism -variation that occurred at least in 0,5 or 1% on the population-), which are interpreted to perform the diagnostic.

As a result of using genotyping, the analysis provided may miss some relevant data set in these unexplored areas. And in general, the diagnosis obtained from these tools should be always supervised by a doctor.

Other drawback of these tools is that the only variations reported are SNPs. The prototype improves the quality of this analysis detecting other complex variations, such as insertions or deletions. Furthermore, the diseases detected by these commercial tools are limited to the number of supported genes. The prototype overcomes this constraint since the conceptual model supports the addition of new genes and their discovered variations.

Table 1. Comparison of DNA analysis tools

	Navigenics	23andMe	deCODEme	Knome	DNADirect
Analysis Type	Genotyping	Genotyping	Genotyping	Sequencing	Sequencing
Platform	Affymetrix [11]	Illumina [12]	Illumina [12]	SOLID3	Illumina[12]
Variations (10^6)	1 (only SNP)	0.5 (only SNP)	1.2 (only SNP)	unlimited	-
Detected diseases	28	51	49	+1000	-

On the other hand, Knome and DNADirect are equivalent tools that offer a revolutionary approach using sequencing instead of genotyping. Sequencing reduces the limitations of genotyping approaches, and enables the detection of other variants that cannot be identified with the above commented tools. However, these tools are far from been accesible to the people due to its price.

The complete understanding of the genomic field can only be achieved through the integration of biological information and the different analysis tools and applications available [13]. The proliferation of these tools has thus increased the importance of workflow systems.

In the commercial field it is possible to find workflow systems such as Pipeline Pilot [14], the first workflow system in life sciences. Pipeline Pilot is widely applied in drug discovery and high-throughput screening (HTS), but it also encompasses Decision Trees, Squene Analysis, BioMining, Text Analytics and Integration Collection, which are flexible mechanisms to link external applications and databases. Another commercial workflow system is the one of InforSense KDE [15] which has got specialized extensions such as BioSense, ChemSense and TextSense. The first mentioned extension covers high performance bioinformatics solutions ranging from sequence analysis to microarrays informatics and remote database annotation.

Both Pipeline Pilot and InforSense KDE help researchers in the life sciences domain in a fast and efficient way. Nevertheless this help is diminished because they cost a lot of money. Therefore these tools are still not accessible to academics and small labs that cannot invest so much money for that kind of solutions.

On the other hand, there are a lot of open-source workflow systems, some of them began as small-scale projects. The major part of the workflow systems in the public domain differ in their architecture and in other features such as workflow language, primary data types, mechanism to add additional resources and available domain resources.

The open-source workflow systems represent an important aid for the scientific domain not only because they are free, but also because they are founded on community development models, in which people from different backgrounds have contributed to the application in an active way. It is worth mention that there are many commercial products that make use of open-source and public available programs. An example of an open-source workflow system is Pegasys [16], developed by the university of British Columbia. This specialized workflow management gives high-throughput sequence data analysis and annotation. Pegasys also includes numerous tools for pair-wise and multiple sequence alignment, gene prediction, RNA gene detection and masking repetitive sequences in genomic DNA, and it also permits the incorporation of new tools into existing frameworks due to its flexible architecture. ^{my}Grid project is considered the most powerful workflow system in the public domain. Fields like Genome and Proteome Annotation (e-Protein), Integrative Systems Biology (myIB) or Integration of Biological Data (PlaNet, EMBRACE) are covered by different tools based on ^{my}Grid. But the most important workflow system rooted on ^{my}Grid is Taverna. The use of booth tools has been extensively employed to execute different *in silico* experiments like simple sequence manipulation or genotype-phenotype correlations. The Genome Analysis and Database Update (GADU) system uses Pegasus to perform high-throughput analysis and annotation of genomic information. GADU workflows are being run across the Open Science Grid and TeraGrid, and to enrich the warehouse they are applying tools such as BLAST or BLOCKS. Nevertheless, the disadvantage of Pegasus is that it does not support a validation checking for workflow.

On the other hand, due to the information explosion in biology, the number of ontologies grows directly proportional to the biological data. The classical

example of an ontology created to provide a list of controlled terms to describe biological entities is the Gene Ontology [17]. The GO Consortium was established to create standard terms to describe what the gene products do, where they act and how they perform these activities. Another example is the one of BioPAX [18] whose purpose is to provide a standard language that enables integration, exchange, visualization and analysis of biological pathways data. TAMBIS project [19] includes an ontology, the TAMBIS ontology, that aims to provide transparent information retrieval and filtering from biological information services by building a homogenizing layer on top of the different sources.

Nevertheless, the purpose of an ontology is to give a description of the terminology used in a specific domain rather than to provide a conceptual representation of the structures used to store data [20]. Contrary to conceptual schemas, ontologies do not allow showing the data structure and relations. Thus the comprehension of the domain is easier in conceptual schemas. The way in which data can be described is made explicit by separating the information models from the system description, which is an advantage for developers of future GIS. For instance, the information technology developers can quickly and easily adopt a model to implement a new feature in a domain without having to repeatedly tackle the same complex modeling challenges. The developed system will be interoperable with other projects that use the same conceptual model. Therefore, tasks such as data integration from different repositories will be simplified. There are some projects that are developing this idea; one is the Phenotye and Genotype Experiment Object Model [21] which has been approved as a standard by OMG. The PaGE-OM was formulated by an international consortium of 20 groups involved in genotype-phenotype projects. The goals of PaGE-OM specification are to reach a balance between being too generic and too specific and to enable the structured capture of at least the minimum amount of the information required to properly report most generic experiments related to genotype and phenotype information. [20] provides conceptual models that describe eukaryotic genome sequence data and genome organization, transcription data and results from gene deletions. Other object-oriented models of biological data have been implemented, such as the one presented in [22], which includes models for representing genomic sequence data.

3 A Conceptual Model to Describe Variations

The main objective of the conceptual model presented in this paper is to establish a connection between the genomic field and the IS development domain. One of the main characteristics of the genomic field is heterogeneity. The unification of relevant concepts is a difficult task, since genomic concepts are not precisely defined. Moreover the field of knowledge is still developing and concepts are constantly evolving, wich complicates the organization of all the genetic data available.

Genetic databases are affected by this heterogeneity problem. In this fielda, each database captures the concepts according to the interpretation and terminology of a biologist. However, there are different definitions for the same

concept; for example, a variation in the DNA sequence is referred under the terms: variation, mutation, polymorphism or SNP [23]. Even though all of them represent more or less the same concept, there are slight differences among them.The problem of heterogeneous data representation can be solved with the use of conceptual models, as some works propose [24]. The development of a conceptual model to represent the human genome is a useful approach to understand this complex domain since precise concepts are defined and related among them. If new concepts, relations or changes are discovered, they can be easily incorporated on the model.

The conceptual model presented here claims to be precise with genetic concepts and IS principles because it has been developed by software engineers and biologists specialized in the genomic field. The model presented in this section is focus on the description of genomic variations. However, it is an excerpt of a widest one [25], whose main goal is the specification of the required human genome concepts for developing GIS.

Figure 1 shows the proposed conceptual model. At the top of the picture (1) the *Gene* and the *Allele* classes are defined. *Gene* entity models the generic concept of gene whereas *Allele* entity represents the individual instances of a gene. The *Allele* entity has two specializations: *Allelic Reference Type* and *Allelic Variant*. *Allelic Reference Type* models the reference sequence that defines an "universal" gen to be used for comparison purposes. These reference sequences are extracted from trusted data sources as RefSeqGene database [26]. *Allelic Variant* represents a DNA sequence of an individual which has several variations from the allelic reference.

Each variation discovered by means of the comparison process performed over a sequence, is modeled by the *Variation* entity (2). The *Variation* entity stores all the variations documented in the genetic literature that are associated to some disease or to normal changes because of the intrinsic nature of an individual. This entity has two different specializations groups. The first one corresponds to the variation description and it is made up of two specializations: *Precise* variations, which define variations that are completely located and *Imprecise* variations, whose location details are not specified. *Precise* variations are also categorized in four entities according to the change performed in the sequence: *Insertion*, *Deletion*, *Indel* (insertion/deletion) and *Inversion*. An indel can be categorized as *SNP* as well when it occurs at least in 1% of the population. The second group consists of three specializations: *Mutant* variations, which represents those variations that are related to some disease, *Unknown consequence*, categorizes the specializations whose consequence has not been discovered yet, and *Neutral polymorphism* variations, the ones that do not have an associated disease.

A variation that is specified in the model is always related to its phenotype, which is modeled by the *Phenotype* entity (4). The *Certainty* entity specifies the probability that a phenotype could show up because of a concrete variation on the genotype. In case a genotype-phenotype association is identified, it is essential to know information about the bibliographic reference and the original database where the discovery was stated. This data is defined by the *Bibliography* Reference

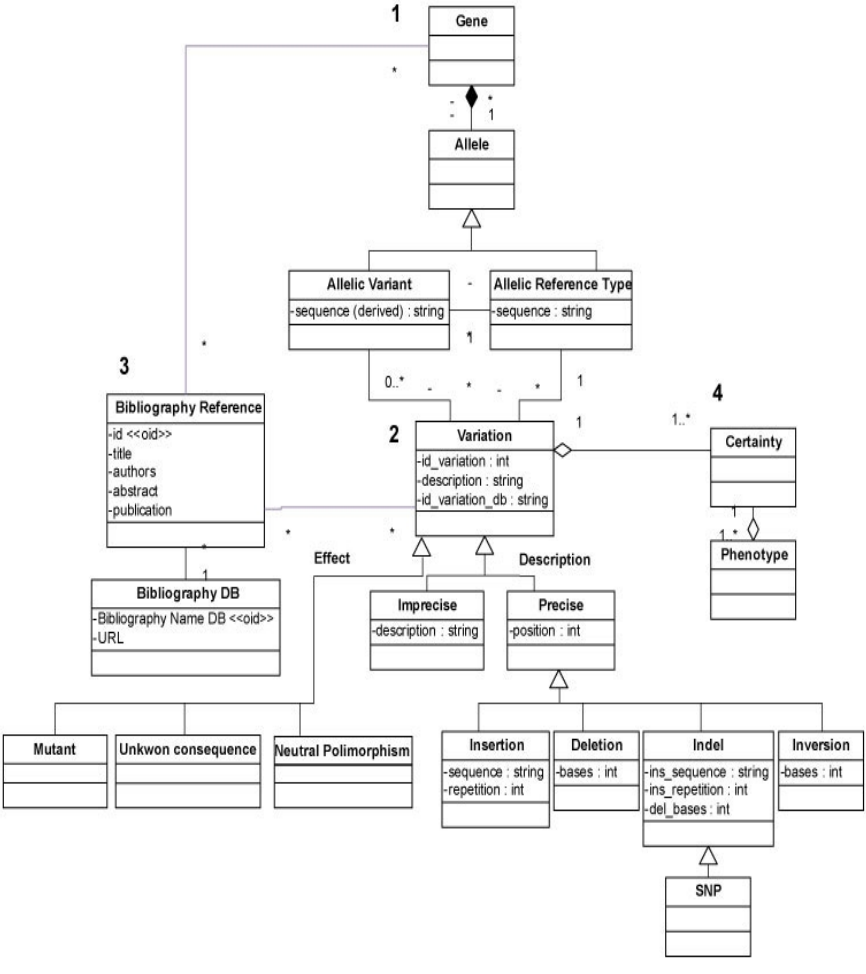

Fig. 1. Conceptual Model for describing variations

and *BibliographyDB* entities (3) respectively. As a first result of this conceptual model, a genetic database (GDB) has been created to store the variation information that is used by the presented GIS prototype.

4 A GIS Proof of Concept : A Variation Analysis Tool

The main goal of the prototype is to show how conceptual models can be useful to define a GIS. One of the most common tasks in the genomic area is the analysis of genomic sequences [27]. Researchers perform the analysis comparising a certain DNA sample from a specific gene and its reference sequence. The comparison is done using an alignment tool that shows a list of differences among them. After that, an experienced researcher has to decide which variations are relevant and which not. Afterwards, they have to dive into the vast and

non-structured amount of information that is scattered across the Web and search for the bibliography that justifies each relevant variation. Performing this work manually is a tedious and time consuming task.

The proposed prototype reduces this time by automating the major part of the manual work. This automation can be done thanks to the conceptualization of the domain by the presented conceptual model. Data such as genes, variations, phenotypes and bibliographic references are now represented as perfectly defined conceptual entities. Thanks to this conceptualization, heterogeneity and data dispersion problems are solved, avoiding the manual preprocess of some non-computer legible data and ensuring the quality of the data stored.

The most widely accepted and used free implementation of BLAST [28] is NCBI BLAST, but more sequence alignment algorithms exist. One of them is BLAT [28], which is an extremely fast alternative but slightly less accurate than BLAST to compare huge nucleotide sequences. This is the chosen algorithm for the implementation of the prototype proposed in this paper.This choice is based on its speed to scan for relatively short matches and also its extension into high-scoring pairs. Differing from BLAST [29], BLAT stitches each area of homology between two sequences into a larger alignment. The prototype explained in this section has been implemented as a web tool developed using ASP.NET and C#.

The service offered by the presented GIS prototype is to receive a DNA sample from a patient and provide a report that helps the doctor to diagnose a certain disease. The experts only have to introduce the sample in the suitable format and review the provided results, forgetting everything about manual treatment and endless searches across the bibliography.

The analysis process performed by the prototype is summarized in Figure 2. Some conceptual model entities that are used in the different steps are depicted in white rectangular boxes. The process is divided into five main steps:

1. Input data: The biologist selects a gene from the set supported by the prototype, for instance the BRCA1 gene, and introduces the DNA sample to be analyzed. The sample input can be performed manually or by uploading a file in FASTA format.
2. Alignment report: According to the selected gene, the prototype locates the suitable reference using the allelic reference entity. After that, an alignment process between the sample and the reference is carried out for finding variations. This alignment is performed using the BLAST algorithm, however importing results from DNA sequencing tools as Sequencher [13] will be supported in next versions. Using the defined conceptual model, each discovered difference is formalized as an instance of the variation entity. This formalization, which is not present at the moment in other tools or databases, is independent of the output from any alignment tool and provides a suitable way for exchanging variations. A report that summarizes all the changes is generated using these variation entities.
3. Variation knowledge: Thanks to the report generated in the previous phase the classification problem is simplified. Variations are located according to a well-know reference sequence and their positions matched to the genomic

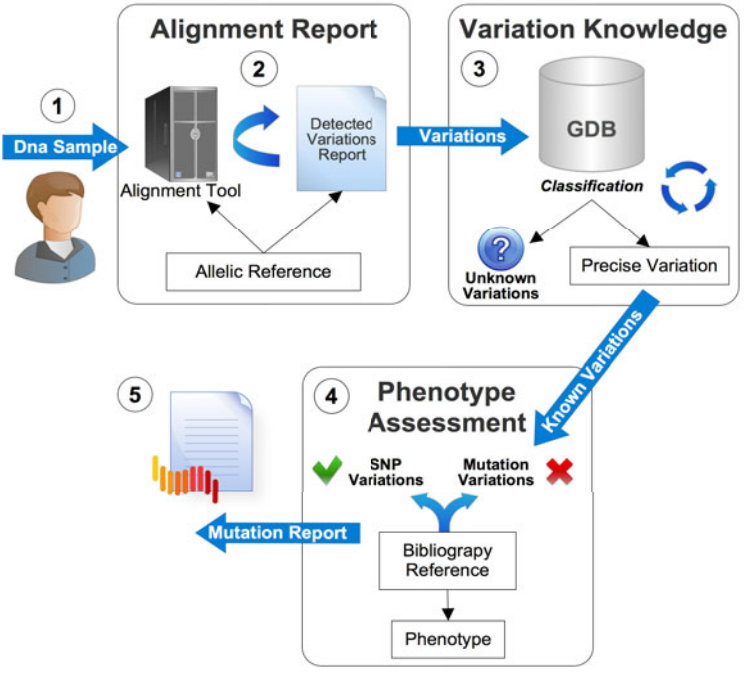

Fig. 2. Mutational analysis tool based on the conceptual model

data stored in GBD. Then, each variation is queried into the GDB to determine if it has been defined as a precise variation. If a variation cannot be found in our GDB is classified as unknown. At this point, known variations are classified into a specific type of sequence change. Unknown variations are classified as non-silent if the variation produces a change, in other words, an effect in the expected gene product (protein).

4. Phenotype Assessment: Variations classified as known may have some phenotype associated. In order to asses if the phenotype is related to a specific disease, a research publication is required to provide trustful evidence. For those cases, the conceptual model describes the bibliographical reference that supports the phenotype for a specific variation. In the context of this work, variations with pathogenic phenotype are classified as mutations whereas they are classified as SNPs if no negative phenotype is described.
5. Report creation: All the obtained information is gathered in a report. This report contains information about the variations found: mutations, variations whose phenotype is not a disease and unknown variations. Each variation is provided with the following information: the location where it was found in the sequence, its type (Insertion, Deletion, Indel or Inversion) and the number of nucleotides inserted or deleted. For the mutations found in the GDB their associated phenotype and its bibliography is added as well. Finally, the report file can be saved as a text document.

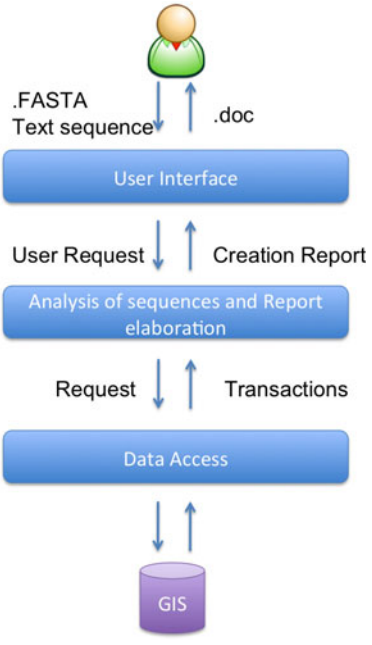

Fig. 3. Architecture of the variation analysis tool

Several key points have been discussed during the implementation of this prototype. The first one was the choice of the architecture to be implemented. A multi-layer architecture is the one that suits best to carry out the implementation of the prototype proposed here. This type of architecture allows a logical separation of the processes related to the presentation, the application processing, and the data management. N-tier application architecture provides a model for developers to create a flexible and reusable application.

By breaking up an application into layers, developers only have to modify or add a specific tier, rather than rewrite the entire application over. Having into account that in the biological domain, and specially in the genomic field, a lot of investigations are being carried out and thus new information is being discovered, it is important to facilitate the way in which the developed system can be modified and maintained. The three-layer architecture implemented to this application is showed in Figure 3.

Another issue is to define the classes that are going to be part of the application. Figure 4 shows a brief summary of the most important implemented classes of the variation analysis tool and also some of their attributes and services.

It's important to keep in mind that the goal of this prototype is to obtain all the mutations that belong to a certain DNA sequence. Some of the differences between this class diagram and the model represented in Figure 1 are a direct consequence of this constraint.

Variation is the principal class. It represents the variations that are found after the DNA sequence analysis. Due to the comparison between DNA sequences,

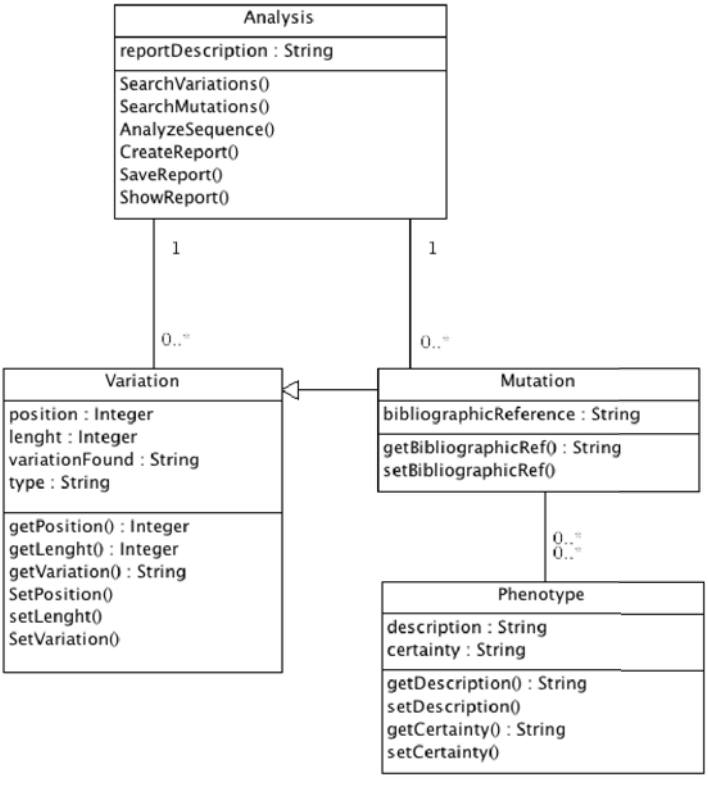

Fig. 4. Diagram Class

only specific positions are obtained. Thus, only precise variations are analyzed and imprecise variations remain outside of the application scope at this moment. Hence, *Variation* class and *Precise* class are joined in one unique class. The specializations of *Precise* class are also joined to this class becoming the type attribute. These classes define the difference between the types of variations, and their attributes can be unified in the same way as they did. Therefore, these new attributes are added to Variation class to facilitate the implementation of the tool by unifying the way in which the properties of the variation type are represented. The services of this class are more related to access and manage their own properties.

Mutation class represents the variations that are related to a disease. *Mutations* class has got the same main properties as *Variation* so, it is implemented as a specialization. Nevertheless, there are some attributes of the class *Mutation* but not of *Variation*, for instance the *BibliographicReference* attribute. As mentioned before, the implemented tool is focused on the mutation search and that is the reason why only this kind of variation needs a bibliographic reference. Instead of using a new class to represent this property, the bibliographic reference is implemented as a *Mutation* class attribute. This choice is made based on the

Fig. 5. GIS prototype interface

assumption that this kind of information is going to be read only as a property of a variation and not managed. In fact, the method *setBibliographicRef* is private and it is only used to provide the value of this attribute through the data base extraction method.

The *Phenotype* class represents the way a variation in a DNA sequence is expressed. However, taking into account the constraint commented at the beginning, this class is only linked to *Mutation* class. *Phenotype* has, apart from its description, an attribute certainty that express the reliability of the phenotype.

Finally the *Analysis* class is the main class of the application and it is responsible for the report creation. The visibility of the *SearchVariations*, *SearchMutations*, *AnalyzeSequence* and *ShowReport* methods is private. The user can only make use of the *CreateReport* and *SaveReport* methods and these services will be responsible for calling the private ones internally.

Referring to the visual aspect Figure 5 shows the interface of the presented GIS prototype. This interface is divided into three parts:

– Input data: At the top there is a list of the currently supported genes. The user can choose which gene is the object of study. There is a text area below to choose the reference sequence. Additionally, the researcher is provided

with two ways to insert the patient's DNA sequence. The first one is a text area where the researcher can introduce the DNA sequence directly. The second way is by uploading a FASTA file.
- Functionality: The actions provided are three: 1) showing the report, this action takes place after the data entry and works in background without the researcher being aware of it 2) cleaning the work area, clean button allows to clear the fields where the researcher can enter data 3) and saving the information obtained.
- Medical report: this is the area in which all the information obtained by the tool is displayed to the user. The different features that were commented in previous sections: location, position, type of variation, etc., are listed here.

5 Conclusions and Future Work

In the genetics domain, the occurrence of a new database is a constant fact. There are a lot of genomic databases containing different kinds of data (variations of DNA sequences, gene specific disease information, ARN or protein data, pathways, microarrays, etc.). Despite the distinct nature of their content, all of them were built under a common goal: structuring data in order to achieve a better comprehension of their discoveries. However, most of them are really useful for the daily work of researchers; but when dififcult questions arise, the problem of interrelate different data located in several databases appear.

This work proposes an IS engineering solution in order to solve the heterogeneity problems on the genomic domain. A conceptual model is presented which describes and defines formally the concepts related to genomic variations. As a proof of concept, a GIS prototype with this conceptual model as background has been implemented. This prototype analyzes human DNA samples searching for variations and identifies the phenotypes that each variation could provoke on the individual.

One of the advantages of using the presented GIS prototype is that the variation analysis can be performed using only one tool, avoiding the data workflow. In addition, using a conceptual model to guide the development simplifies the acquisition of the genetic data and is precisely referenced to the bibliography.

However, the study of the prototype efficiency working with real DNA samples must be analyzed. In order to perform this task, further studies of the sequencing algorithms will be carried out.

Moreover, heterogeneity is not a completely novel research area because some tools to organize the genomic data have also been proposed before [17,20,30]. The main contribution of the work presented here is a conceptual model specifically designed to guide the development of software artifacts using a model-driven approach.

As further work it is planned to extend the GIS prototype in order to achieve a higher accuracy and to facilitate the sequence input. As a final goal, the GIS prototype will be tested in a real environment by means of a collaboration with IMEGEN, a genomic medicine institute, and a couple of local hospitals.

References

1. Watson, J., Crick, F.: A structure for deoxyribose nucleic acid. Nature 171, 737–738 (1953)
2. Jordan, E.: The American Journal of Human Genetics 51, 1–6 (1992)
3. Craig, J., Venter, J.C., Adams, M.D., Myers, E., Li, P.W., Mural, R.J., Sutton, G.G., Smith, H.O., Yandell, M., Evans, C.A., Holt, R.A., Gocayne, J.D., Amanatides, P., Ballew, R.M., Huson, D.H., Wortman, J.R., Zhang, Q., Kodira, C.D., Zheng, X.H., Chen, L., Skupski, M., Subramanian, G., Thomas, P.D., Zhang, J., Gabor Miklos, G.L., Nelson, C., Broder, S., Clark, A.G., Nadeau, J., McKusick, V.A., Zinder, N., Levine, A.J., Roberts, R.J., Simon, M., Slayman, C., Hunkapiller, M., Bolanos, R., Delcher, A., Dew, I., Fasulo, D., Flanigan, M., Florea, L., Halpern, A., Hannenhalli, S., Kravitz, S., Levy, S., Mobarry, C., Reinert, K., Remington, K., Abu-Threideh, J., Beasley, E., Biddick, K., Bonazzi, V., Brandon, R., Cargill, M., Chandramouliswaran, I., Charlab, R., Chaturvedi, K., Deng, Z., Di Francesco, V., Dunn, P., Eilbeck, K., Evangelista, C., Gabrielian, A.E., Gan, W., Ge, W., Gong, F., Gu, Z., Guan, P., Heiman, T.J., Higgins, M.E., Ji, R.R., Ke, Z., Ketchum, K.A., Lai, Z., Lei, Y., Li, Z., Li, J., Liang, Y., Lin, X., Lu, F., Merkulov, G.V., Milshina, N., Moore, H.M., Naik, A.K., Narayan, V.A., Neelam, B., Nusskern, D., Rusch, D.B., Salzberg, S., Shao, W., Shue, B., Sun, J., Wang, Z., Wang, A., Wang, X., Wang, J., Wei, M., Wides, R., Xiao, C., Yao, A., Ye, J., Zhan, M., Zhang, W., Zhang, H., Zhao, Q., Zheng, L., Zhong, F., Zhong, W., Zhu, S., Zhao, S., Gilbert, D., Baumhueter, S., Spier, G., Carter, C., Cravchik, A., Woodage, T., Ali, F., An, H., Awe, A., Baldwin, D., Baden, H., Barnstead, M., Barrow, I., Beeson, K., Busam, D., Carver, A., Center, A., Cheng, M.L., Curry, L., Danaher, S., Davenport, L., Desilets, R., Dietz, S., Dodson, K., Doup, L., Ferriera, S., Garg, N., Gluecksmann, A., Hart, B., Haynes, J., Haynes, C., Heiner, C., Hladun, S., Hostin, D., Houck, J., Howland, T., Ibegwam, C., Johnson, J., Kalush, F., Kline, L., Koduru, S., Love, A., Mann, F., May, D., McCawley, S., McIntosh, T., McMullen, I., Moy, M., Moy, L., Murphy, B., Nelson, K., Pfannkoch, C., Pratts, E., Puri, V., Qureshi, H., Reardon, M., Rodriguez, R., Rogers, Y.H., Romblad, D., Ruhfel, B., Scott, R., Sitter, C., Smallwood, M., Stewart, E., Strong, R., Suh, E., Thomas, R., Tint, N.N., Tse, S., Vech, C., Wang, G., Wetter, J., Williams, S., Williams, M., Windsor, S., Winn-Deen, E., Wolfe, K., Zaveri, J., Zaveri, K., Abril, J.F., Guigó, R., Campbell, M.J., Sjolander, K.V., Karlak, B., Kejariwal, A., Mi, H., Lazareva, B., Hatton, T., Narechania, A., Diemer, K., Muruganujan, A., Guo, N., Sato, S., Bafna, V., Istrail, S., Lippert, R., Schwartz, R., Walenz, B., Yooseph, S., Allen, D., Basu, A., Baxendale, J., Blick, L., Caminha, M., Carnes-Stine, J., Caulk, P., Chiang, Y.H., Coyne, M., Dahlke, C., Mays, A., Dombroski, M., Donnelly, M., Ely, D., Esparham, S., Fosler, C., Gire, H., Glanowski, S., Glasser, K., Glodek, A., Gorokhov, M., Graham, K., Gropman, B., Harris, M., Heil, J., Henderson, S., Hoover, J., Jennings, D., Jordan, C., Jordan, J., Kasha, J., Kagan, L., Kraft, C., Levitsky, A., Lewis, M., Liu, X., Lopez, J., Ma, D., Majoros, W., McDaniel, J., Murphy, S., Newman, M., Nguyen, T., Nguyen, N., Nodell, M., Pan, S., Peck, J., Peterson, M., Rowe, W., Sanders, R., Scott, J., Simpson, M., Smith, T., Sprague, A., Stockwell, T., Turner, R., Venter, E., Wang, M., Wen, M., Wu, D., Wu, M., Xia, A., Zandieh, A., Zhu, X.: The Sequence of the Human Genome Science 291, 1304–1351 (2001)
4. Collins, F.S., Green, E.D., Guttmacher, A.E., Guyer, M.S.: A vision for the future of genomics research Nature 422, 835–847 (2003)

5. Gilbert, D.G.: Eugenes: a eukaryote genome information system. Nucleic Acids Research 30, 145–148 (2002)
6. Navigenics (2010), http://www.navigenics.com
7. 23andme (2010), https://www.23andme.com
8. Decodeme (2010), http://www.decodeme.com
9. Medco acquires leading genetics healthcare company, DNA Direct (2005), http://www.dnadirect.com/web/
10. Knome (2010), http://www.knome.com
11. Irizarry, R.A., Bolstad, B.M., Collin, F., Cope, L.M., Hobbs, B., Speed, T.P.: Summaries of affymetrix genechip probe level data. Nucleic Acids Research 31, e15 (2003)
12. Klein, R.: Power analysis for genome-wide association studies. BMC Genetics 8, 58 (2007)
13. Tiwari, A., Sekhar, A.K.: Workflow based framework for life science informatics. Computational Biology and Chemistry 31, 305–319 (2007)
14. Hassan, M., Brown, R.D., Varma-O'brien, S., Rogers, D.: Cheminformatics analysis and learning in a data pipelining environment. Molecular Diversity 10, 283–299 (2006)
15. Watson, C., Guo, Y., Sheldonb, J.: Yike Guo and Jonathan Sheldon of InforSense discuss the impact of workflow technology on drug discovery. Drug Discovery Today 10, 1211–1212 (2005)
16. Shah, S., He, D., Sawkins, J., Druce, J., Quon, G., Lett, D., Zheng, G., Xu, T., Ouellette, B.: Pegasys: software for executing and integrating analyses of biological sequences. BMC Bioinformatics 5 (2004)
17. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Gene ontology: tool for the unification of biology. Nature Genetics 25, 25–29 (2000)
18. BioPax-Consortium: Biological pathways exchange (2005), http://www.biopax.org/
19. Stevens, R., Baker, P., Bechhofer, S., Ng, G., Jacoby, A., Paton, N.W., Goble, C.A., Brass, A.: Tambis: Transparent access to multiple bioinformatics information sources. Bioinformatics 16, 184–186 (2000)
20. Paton, N.W., Khan, S.A., Hayes, A., Moussouni, F., Brass, A., Eilbeck, K., Goble, C.A., Hubbard, S.J., Oliver, S.G.: Conceptual modelling of genomic information. Bioinformatics 16, 548–557 (2000)
21. Brookes, A., Lehvaslaiho, H., Muilu, J., Shigemoto, Y., Oroguchi, T., Tomiki, T., Mukaiyama, A., Konagaya, A., Kojima, T., Inoue, I., Kuroda, M., Mizushima, H., Thorisson, G., Dash, D., Rajeevan, H., Darlison, M.W., Woon, M., Fredman, D., Smith, A.V., Senger, M., Naito, K., Sugawara, H.: The phenotype and genotype experiment object model (PaGE-OM): a robust data structure for information related to DNA variation. Human Mutation 30, 968–977 (2009)
22. Medigue, C., Rechenmann, F., Danchin, A., Viari, A.: Imagene: an integrated computer environment for sequence annotation and analysis. Bioinformatics 15, 2–15 (1999)
23. den Dunnen, J.T., Antonarakis, E.: Nomenclature for the description of human sequence variations. Human Genetics 109, 121–124 (2001)
24. Richesson, R., Turley, J.P.: Conceptual models: Definitions, construction, and applications in public health surveillance. Journal of Urban Health 80, i128 (2006)

25. Pastor, O., Levin, A., Casamayor, J., Celma, M., Virueta, A., Eraso, L., Pérez-Alonso, M.: Enforcing conceptual modeling to improve the understanding of human genome. In: Procs. of the IVth Int. Conference on Research Challenges in Information Science (2010)
26. NCBI: The refseqgene project (2010), http://www.ncbi.nlm.nih.gov/RefSeq/RSG
27. Stevens, R., Goble, C., Baker, P., Brass, A.: A classification of tasks in bioinformatics. Bioinformatics 17, 180–188 (2001)
28. Kent, W.J.: Blat, the blast-like alignment tool. Genome Research 12, 656–664 (2002)
29. Altschul, S., Gish, W., Miller, W., Myers, E.W., Lipman, D.: Basic local alignment search tool. Journal of Molecular Biology 215, 403–410 (1990)
30. Ram, S.: Toward Semantic Interoperability of Heterogeneous Biological Data Sources. In: Pastor, Ó., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 32–32. Springer, Heidelberg (2005)

GAMES: Green Active Management of Energy in IT Service Centres

Massimo Bertoncini[1], Barbara Pernici[2], Ioan Salomie[3], and Stefan Wesner[4]

[1] Engineering Ingegneria Informatica, Italy
[2] Politecnico di Milano, Italy
[3] Technical University of Cluj-Napoca
[4] High Performance Computing Centre Stuttgart

Abstract. The vision of the recently started GAMES European Research project is a new generation of energy efficient IT Service Centres, designed taking into account both the characteristics of the applications running in the centre and context-aware adaptivity features that can be enabled both at the application level and within the IT and utility infrastructure. Adaptivity at the application level is based on the service-oriented paradigm, which allows a dynamic composition and recomposition of services to guarantee Quality of Service levels that have been established with the users. At the infrastructure level, adaptivity is being sought with the capacity of switching on and off dynamically the systems components, based on the state of the service centre. However, these two perspectives are usually considered separately, managing at different levels applications and infrastructure. In addition, while performance and cost are usually the main parameters being considered both during design and at run time, energy efficiency of the service centre is normally not an issue. However, given that the impact of service centres is becoming more and more important in the global energy consumption, and that energy resources, in particular in peak periods, are more and more constrained, an efficient use of energy in service centres has become an important goal. In the GAMES project, energy efficiency improvement goals are tackled based on exploiting adaptivity, on building a knowledge base for evaluating the impact of the applications on the service centre energy consumption, and exploiting the application characteristics for an improved use of resources.

1 Introduction

Over the last years, with the increasing digitalization of the business processes in many application domains, like online banking, e-commerce, digital entertainment, and e-health, the data centre industry has seen a great expansion due to increased need for computing capacity to support business growth. As a consequence, management of IT Processes, Systems and Data Centres has dramatically emerged as one of the most critical environmental challenges to be dealt with and new research directions are being taken towards an energy-efficient management of data centres. An estimation is reported in [15] that the

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 238–252, 2010.
© Springer-Verlag Berlin Heidelberg 2010

US servers and data centres consumed about 61 billion kilowatt-hours (kWh) in 2006 (1.5 percent of total U.S. electricity consumption). This estimated level of electricity consumption has been evaluated as equal to the amount of electricity consumed by approximately 5.8 million average U.S. households.

In the last years, large IT systems and data centres are moving towards the adoption of a Service-based Model, in which the available computing resources are shared by several different users or companies. In such systems, the software is accessed as-a-service and computational capacity is provided on demand to many customers who share a pool of IT resources. The Software-As-A-Service model can provide significant economies of scale, affecting to some extent the energy efficiency of data centres. The service-based approach is becoming the most common way to provide services to users, compared to traditional applications developments. Services and their composition, both at the providers' side (to provide new value-added services), and at the users' side (with mash-ups of services composed by the users themselves), are becoming more and more widespread in a variety of application domains. Hence, since the service-oriented approach is steadily increasing for many application domains, its impact on data and service centres will become more and more significant. A very similar model is applied to the provision of services in the High Performance Computing domain where users are allocated to these precious resources in a shared way by using complex scheduling mechanisms.

The EPA report [15] contains a forecast of doubling the energy consumption estimated in 2006 within five years. In this report it is indicated that there is a potential of reducing energy consumption with existing technologies and design strategies by 25 percent or more. However, there is yet a potential for improving energy efficiency in several aspects of a data centre. In fact, despite the big effort that has been put for assessing energy efficiency of IT service centres aiming at the reduction of energy costs [5], the most of these actions have been concerned with solutions in which energy efficiency leverages only on single, yet not interrelated factors, such as the identification of good practices for energy savings based on: (i) the dynamic management of servers according to workload and servers consolidation and virtualization; (ii) the development of low power techniques at IT component level; and (iii) the design of energy-effective facility environments for data centres through reuse of heat or air conditioning. The analysis of the characteristics of the software applications run in data centres are just starting to be considered, such as for instance in the EU best practices for data centres [6].

Mostly, these policies have been implemented in an isolated and fragmented way, not taking into account all the interrelations between the different decision-making layers and were unable to evaluate simultaneous trade-off between power, workload and performance and users' requirements. In particular, the applications running in the service centre are usually only analyzed based on their general characteristics, such as frequency of execution and requests for resources. The analysis of applications at the design level, however, could provide useful information to better manage the resources in the infrastructure. For instance,

the structure of the application can be a basis for predicting the resources (e.g., data) that will be necessary for its execution. Such an information can in turn be useful for an internal management of storage resources. On the other hand, information about IT resources can also be used to design energy efficient applications. In fact, while there has been a focus on optimization and negotiation of Quality of Service and performances in the past [11,9], very little attention has been paid to the issues of energy consumption and development of energy efficient services. A first proposal has been presented in [7], where energy consumption and energy efficiency have been considered in composed services at the same level of other quality of service parameters. This allows designing applications that can dynamically adjust to the IT infrastructure state in order to reach energy-efficiency goals, while keeping the agreed quality of service levels.

The vision of the GAMES (Green Active Management of Energy in IT Service centres) project (2010-2012) is for a Green, Real-Time and Energy-aware IT Service Centre. The central innovation sustaining the GAMES vision is that for the first time, to our knowledge, the energy efficiency of the IT Service Centres will be considered simultaneously at different levels, trading-off (1) user and functional requirements and Quality of Services versus energy costs at business/application level, (2) performance, expressed as physical resources workload and Service Level Agreement, against energy costs at IT infrastructure level, (3) HVAC (Heating, Ventilating and Air Conditioning) and lighting versus the power required by the IT infrastructure and the business processes and application, as received by upper levels, at Facility level.

At design time, the assessment and benchmarking of the energy consumption and efficiency of all the different building blocks composing the GAMES IT Service Centres energy efficiency (HVAC, lighting at the facility level, servers, storage, network and processors at IT infrastructure level, services, applications, QoS) will be made for each of the sub-optimal configurations. With this regard, what-if simulation analysis will be carried out in order to determine at design time the best energy-effective distributions of services on the virtualized machines, what will be the best resource and workload configurations with less energy costs, and the impact of these configurations on the energy and carbon emissions balance of the IT Service Centre facility. Historical and required power information and the energy usage profile, combined with Business Intelligence, Data Mining and Information Extraction technologies as well as simulation technologies will be matched with users' business, functional and applications requirements to align energy demand with availability (energy contracted with the utility operator) to design energy efficient applications on an energy efficient infrastructure, able to exploit adaptivity during execution.

The optimized configurations, which will be the output of the GAMES system at design time, will be continuously monitored and adaptively controlled at runtime, through a suitable sensing and monitoring technology infrastructure able to measure temperature, power consumption and humidity of each single IT device (servers, storage, network). The GAMES co-design methodology will aim

at co-designing business level applications and services and the IT infrastructure, to support a global energy-aware adaptive approach.

In Section 2 the chosen drivers for the design and the validation of the GAMES approach are described in an abstract way and requirements for the overall approach presented in the following Section 3 are provided. In Sections 4 and 5 we discuss the co-design approach and the adaptive run-time environment respectively.

2 Application Scenarios

The GAMES approach is neither targeting for a specific application sector nor bound to a specific solution for the realisation of the service layer or the applications hosted by the service provider. In order to drive this generic approach with requirements and to allow their validation, two major scenarios are targeted:

- High Performance Computing Service Provision addressing the need for large scale simulations demanding for the co-allocation of a very large number of computing resources for a single task;
- Cloud based computing service provision where the elasticity and dynamism of requested amount of the resources is high.

2.1 Specific Challenges of the HPC Scenario

In this scenario users such as computer engineers submit a job request via a Grid Middleware or directly using an interactive shell environment. Typically such job requests are expressed using the Job Service Description Language (JSDL) [2], an XML based schema allowing to express the requirements such as memory demand, number and speed of CPUs, etc. Such job requests are then either given to a meta-scheduler per provider that aims to find the appropriate resource or directly to a batch oriented queueing system for a single cluster system. In order to allow a differentiation between different consumer types, such requests are typically further detailed with negotiated Service Level Agreements expressing non technical requirements and corresponding guarantees from the provider side as well as penalties in case of violations of the consumer or provider obligations.

From the GAMES perspective, several potentially conflicting constraints need to be actively managed. On one hand, the Service Level Agreements from different customers with different level of importance (in the sense of how important the customer is from a business viewpoint) and different penalties in case of violations need to be aligned with policies like “prioritize large jobs compared to small ones for this resource”, “maximise utilization of the resource” with Green Performance Indicators such as “reduce workload during days where free cooling is impossible due to high outside temperatures”.

As a further illustration consider the simplified example of a computing resource having 20 nodes. Currently 3 jobs requiring 15 computing nodes each and a couple of smaller jobs with only 8 nodes each are waiting in the job queue.

The policy of preferring large jobs would prioritize the two 15 node jobs ahead of the 8 node jobs. The policy for reducing the workload if the outside temperature does not allow free cooling (and therefore higher costs) would switch off 12 nodes during the hot time of the day pushing the 8 node jobs and would move to the 15 node jobs during the evening/night when free cooling is possible again. The job mix is typically much more varying and the number of nodes on high end production systems is several thousands.

An additional problem related to this scenario is that many applications hosted by the provider or provided by the consumer at job submission time could be even only available in binary form (e.g. provided by an Independent Software Vendor) and cannot be easily made GAMES enabled to react on commands that are control attempts from the GAMES framework. Consequently, the GAMES framework can only intervene with the supporting infrastructure such as the queuing systems and job scheduling frameworks and the SLA assessment and validation infrastructure.

Nevertheless, by influencing on one hand the underlying provisioning middleware as well as influencing the queuing infrastructure, a plethora of control possibilities do exist. For example one can limit access to certain queues allowing prioritised access to the resources for certain users or jobs, as well as limit the maximum amount of time one can use a resource demanding a restart capability of the application developer to continue a simulation after it got canceled at the last stable state. Additionally, different customer profiles provide additional control possibilities. Such customer profiles could range from cost optimized best-effort computing, over time boxed simulations (e.g. in a car design process) up to urgent computing cases where simulation results are expected to be used as input to a decision support system for a medical treatment.

2.2 Specific Challenges of the Cloud Scenario

In contrast to the scenario above, the applications hosted in this scenario have no pre-defined maximum wall clock time or a fixed amount of resources but demand a dynamically changing amount of resources (commonly referred to as elasticity) from the provider and do not have the requirement to use several thousands of computing nodes exclusively at the same time but can operate on top on a virtualized infrastructure potentially sharing one physical computing node with several other customers running their own virtual machines. Additionally, properties like reliability and robustness are of much higher importance as in contrast to the simulation jobs of the HPC scenario it is not easily possible to perform a re-start of the simulation at the last checkpoint some hours later without affecting the Quality of Experience (QoE) of the consumer.

Considering that beyond the Infrastructure as a Service (IaaS) paradigm, also Platform as a Service (PaaS) and Software as Service (SaaS) are becoming more and more important in the cloud provisioning model (see also [10]), one can also assume that in this case the applications that are supposed to be provided as services can be fully GAMES enabled and provide a direct interface from the cloud operation level over the platform up to the software level.

However, similar to the HPC use case, the differentiation of the Service Level for different customers, business policies to prefer specific kinds of applications (e.g., applications with a high demand or low demand for elasticity) are similarly conflicting with the demand to save energy.

2.3 Major Requirements Summary

Summarising the major challenges of the two distinct scenarios above, one can say that the following aspects are currently missing in existing data and computing centre infrastructures:

- A mitigation framework allowing to derive clear policies and actions for the underlying infrastructure based on the potentially contradictory and conflicting optimisation goals of the different aspects (e.g., SLAs, Green IT aspects, Differentiated customer support, and so on).
- A rich sensing and monitoring framework allowing the collection of the necessary information on all levels from the facility and environment, over physical resources over platform services up to the applications themselves.
- Data Mining and Reasoning elements analysing the historical data collected aiming for a prediction and derived counter measures to bring the overall infrastructure back to the desired operational point.
- Actors and control elements on the different levels allowing the counter measures to be applied on the appropriate level in the necessary short time.

Orthogonal to this more operational oriented requirements, one can clearly realise the need to plan for the monitoring at the design phase of a building, hardware and the corresponding software services. Similarly, one needs to think about which elements can be controlled and which hooks need to be designed into this overall setting. Beside a Runtime Environment aiming for the optimisation of the operation of the application services, similarly the right design of the overall system from facility and energy provision model up to the service composition and application layer is of equal importance.

3 The GAMES Approach

In the GAMES approach, we consider a joint management of the applications and the IT infrastructure on which they are running. Both at the application level and at the infrastructure level, we assume that the system is adaptive: the applications can change their modes of operation at run time, and the IT infrastructure can be dynamically reconfigured in the service centre. Adaptation is performed according to a number of adaptation strategies that are encapsulated in adaptation rules that are evaluated at run time. The adaptation strategies and their rules are designed for the service centre taking into consideration both the IT infrastructure and its capability of providing an autonomic behavior and the characteristics of the applications being executed on the system and their requirements. We assume that the service centre is exploiting a virtualization of

the IT infrastructure, so that virtualized IT resources can be associated to each application, thus application management and IT management can be decoupled. The adaptivity of the system is performed considering its general context of execution, which includes physical environment parameters such as the external temperature and humidity, the parameters of the service centre facility infrastructure, which include cooling and heating and servers and storage parameters, and application parameters, mainly in terms of Quality of Service requirements and utilization rates of infrastructure components assigned to the application. We propose an adaptive SBA (Service Based Architecture) as the basis of the energy-aware design and management of service-based information systems and their IT infrastructure. At the application level, we consider applications as being executed by invoking services or composite services, which can be possibly dynamically modified. To each service, Service Level Agreements are associated, covering Quality of Service requirements, which can be dynamically renegotiated. Similarly, platform services and infrastructure are all considered as services in the system, creating a complex environment of service compositions, where each service is associated to a number of possible adaptation actions.

The goal of the GAMES approach is to realise a self-adaptive data and service centre architecture across all kinds of offered resources ranging from IT infrastructure data and facility over computing up to the service layers. The conceptual architecture in Figure 1 shows the components needed for run-time management to continously balance the agreed service contracts and to derive the necessary measures needed based on the monitored values (energy consumption, load situation, risk to fail on an SLA, etc.) as well as the interaction with the design environment.

In the following, we briefly introduce the main components of the GAMES architecture:

- the Energy Sensing and Monitoring Infrastructure (ESMI)
- the Run-Time Environment (RTE)
- the Design-Time Environment (DTE)

The **Energy Sensing and Monitoring Infrastructure** (ESMI) provides services to interact with the energy grid, with the environment monitoring infrastructure and with the data centre resources, for energy consumption and other physical measures. The ESMI has an energy service layer providing basic monitoring, messaging, event derivation features, and mining services for analysing historical data targeting the generation of useful adaptation patterns and knowledge. The ESMI will be partially based on the energy service layer being developed in BeAware [8]. The sensing infrastructure will be interfaced with monitoring services, which will in addition gather relevant information from the IT infrastructure and SBA layer, generating relevant events from the sensor information. A context management support module will manage context information.

The **Run-Time Environment** (RTE) provides an energy-aware and self-* adaptivity controller. It includes functionalities for event analysis, based on

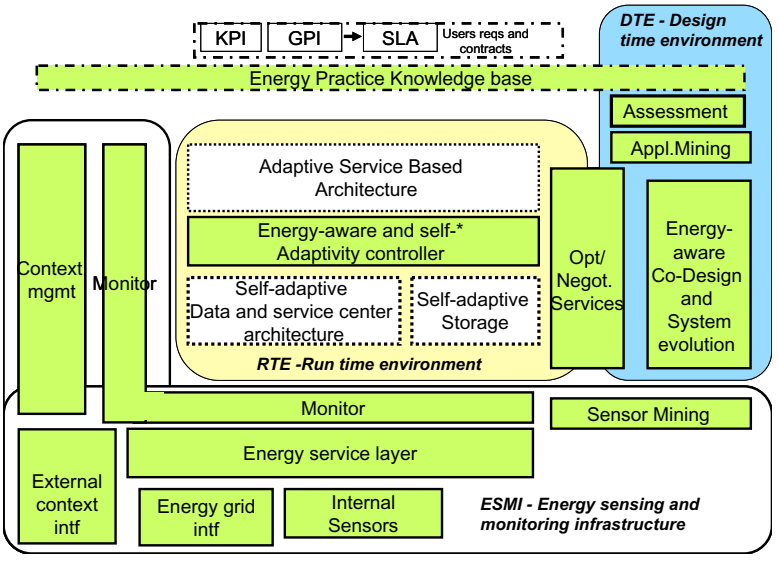

Fig. 1. GAMES architecture

the general knowledge of the environment and energy characteristics of services, controlling the adaptivity under a global perspective of a service and an architectural level. Control will be also based on a general optimiser and negotiator, which, starting from static tools for architecture optimisation and SLA templates, will be enhanced with dynamic and energy-aware functionalities, exploiting also the Energy Practice Knowledge Base. The self-adaptive data centre architecture module comprises an adaptation of the architectural part and of the storage-part through strategies and decisions on data placement and storage quality of service based on access patterns and mapping of application services to data storage level.

The **Design Time Environment** (DTE) will support an energy-aware co-design of service-based information systems and IT architecture in the data and service centre. All design choices are driven by users demands expressed as a set of Key Performance Indicators (KPI) and Green Performance Indicators (GPI) that are part of the negotiated Service Level Agreements (SLA). Starting from a static evaluation of existing configurations, optimisation and negotiation techniques for design time, choices will be developed to devise the optimal functioning points to be exploited for run-time adaptivity. The design will include also the identification of the observable needs for optimal and efficient run-time event detection. Users involvement will be considered through test cases and user experience models. An assessment tool will provide an initial analysis of the users requirements, service and data characteristics and IT infrastructure and facility from which the energy-aware adaptive service and infrastructure design will start.

4 Designing an Energy-Efficient Service Centre

Energy-aware service-based information systems design will be tackled based on a three-fold perspective: a) strategic-level decisions in developing green IT service centres (e.g., identifying Green Key Performance Indicators (GPI) and analysing the impact of QoS business process levels on energy costs); b) developing control strategies to evaluate, optimise, and control services and data at run-time on multiple time scales and adapt them at run time; c) realizing technological mechanisms and tools to reduce the energy consumption of IT service centres based on self-adaptive services and architectures. Energy savings can be obtained by exploiting the characteristics of existing adaptive platforms both at the business/application level, where adaptive service compositions can be executed, and at the architectural level, based on adaptation of IT architectures and components. The problem to be solved is how to combine the existing approaches in a layered architecture, considering a large number of information systems using the same services and sharing the same data centre(s). We propose a combined design-time and run-time approach. At design time, co-design is proposed to create adaptive service-based information systems and self-adapting architectures based on the requirements. At run-time, we propose an event-based adaptation process that takes into consideration the run-time context information (energy consumption) and design-time context information (user and business contexts).

We will focus on the design of energy-aware information systems, in which the information system functionalities and the IT system architecture are co-designed to get improved energy efficiency. The energy dimension is currently not considered in information systems design, where functionality and quality of service considerations are driving design choices. Based on some research experiments and simulation [1,3], we advocate that considering the energy consumption dimension, different and more efficient design choices could be performed.

Examples of energy-aware co-design include several aspects at different levels: strategies to minimize the number or similar/redundant services, e.g. by using virtualisation technologies or a balanced number of servers performing supporting services operations (e.g., having only a minimal number of authentication servers); an evaluation of the impact on needed cooling capacities based on different load scenarios of servers; a focus on business process analysis of core activities-services-data as shown in [14], annotating business processes with meta-information useful for performing an energy-efficient management of the application at run time.

We will develop a cost-based approach to design the system globally and to select the adaptation strategies that are recommended at run time at the application (process/service composition) and at the IT level and to identify the variables and components which need to be monitored in order to ensure a correct control of the system. Business processes will be analyzed considering their quality of service requirements and their needs for IT processing infrastructure. Process meta-information will include data requirements and task dependencies, the ability to use alternative services in service compositions, and their

context-awareness, in order to be able to enhance the adaptive capability of the application itself, both at application management and IT management level.

5 Energy Efficiency at Run Time

GAMES defines a new approach for a run-time, energy-aware adaptive mechanism. The basic concept is to consider and use the system context situation enhanced with energy/performance information for controlling / adjusting / enforcing the run-time energy efficiency goals. We approach the problem of minimizing the energy consumption in a service centre by using Dynamic Power Management (DPM) and consolidation techniques. In a classical data centre, the computing resources are over provisioned to handle the peak value. We propose to minimize the energy consumption by monitoring the service centre servers to determine the over provisioned resources with the goal of putting them in low power states.

Layered feedback architecture will be considered for run-time controlling of systems performance/energy ratio, by combining autonomic and context aware computing methodologies, techniques, algorithms and tools with methods and tools specific to the systems and control theory.

We propose the development of two types of control loops that will be used to adjust and adapt the system execution to the energy efficiency goals established in the co-design phase: a set of local control loops associated to IT Infrastructure servers and one global control loop associated to the whole system.

5.1 Local Control Loop (Server Level Controller)

The local control loops are used to locally optimize the energy consumption at the level of each server. The controller is developed by using a set of server specific energy optimization rules, predefined at design time, which can be executed on a very fine time grain, without affecting the system overall performance. The main guiding idea of server level energy efficiency is to pro-actively identify and take appropriate actions to reduce servers resource over-provisioning so that it matches the application requirements. This way, energy can be saved by transitioning the over-provisioned resources to low power states while maintaining performance by satisfying the application requirements. For the local loop controller design we have used server specific DPM and workload allocation techniques, considering the CPU and external storage as the main controlled resources. The local loop controller takes DPM actions based on the current server state and on the workload received from the global control loop (see Fig. 2). The local loop controller uses the context model instance, a knowledge-base and a prediction engine for inferring the most appropriate DPM actions.

*C*PU management. For the energy-aware CPU management, two workload allocation strategies and an improved Dynamic Voltage Scaling (DVS) technique is proposed. The first workload allocation technique distribute as many tasks per core (allowing the unused cores to be switched into low power states) and per

CPU (allowing the unused CPUs to stay in idle states as much as possible) while the second strategy distribute evenly the tasks and obtaining as a result a lower overall utilization, thus enabling the transition of the entire core / socket system to lower frequencies. Starting from the observation that service requests are unpredictable over time and peak loads are orders of magnitude larger than those in steady states, we have decided to use a fuzzy-logic based control algorithm, by adapting the work presented in [4] for implementing the DVS techniques aiming at processor dynamic voltage scaling.

*S*erver storage management. The local control loop also manages the server HDDs aiming at maximizing energy efficiency. Unlike the CPU which is very flexible in terms of power management, the state transitions of HDDs are more rigid, involving significant performance degradation when executing a wrong power management decision. For HDD management, the local control loop uses an advanced adaptive algorithm, based on adaptive learning trees [12], aiming at identifying the access patterns and deciding about the possible spin-down of the drives. Moreover, the resource management component of the local control loop will attempt to distribute the incoming workload associated storage requirements in such a way to maximize the idleness periods of some drives while preserving the SLA indicators.

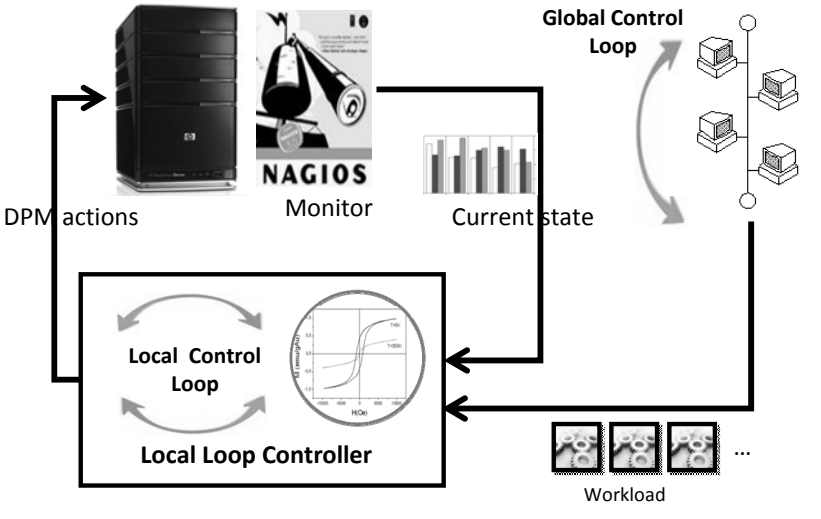

Fig. 2. Local control loop

Local control loop architecture for a virtualized server is presented in Fig. 3. Virtualization allows for better control of energy savings by (i) the fine grain resource allocation in a per virtual machine policy and by (ii) facilitating server consolidation by virtual machine migration, leading to the complete shutdown of unused servers. Local control loop consists of three main modules: the resource allocator, the monitoring subsystem and the Dynamic Power Management (DPM) module. The resource allocator module assigns resources to the virtual machines

so that it delivers the expected performances while the CPU and HDDs make low power states transitions as often as possible. The monitoring and analyzing module analyze the state of the server and the state of the virtual machines and provides the monitoring data to the resource allocator module and to the DPM Controller. The DPM module controller enforces the power states transitions determined by the CPU and HDD prediction engines.

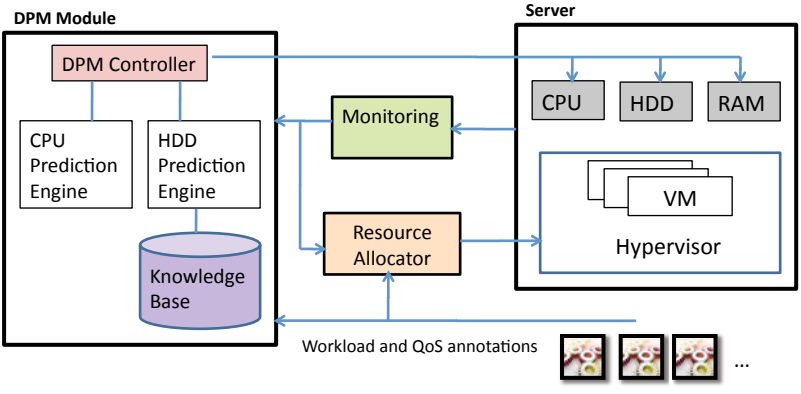

Fig. 3. Local loop controller architecture

5.2 Global Control Loop (Service Centre Level Controller)

The objective of the global control loop (see Fig. 4) is to evaluate the energy enhanced service center context situation (represented by the context model instance) and determine the most appropriate adaptation decisions to enforce and realize the Key Performance Indicators (KPI) and GPI (Green Performance Indicators) defined in the co-design phase. The context model instance records current workload and energy data from all service centre individual servers, environmental data and business process features. The context model used in the GAMES project is based on the RAP model [13] which represents the context as three sets together with their relationships: a set of resources, a set of actions and a set of policies. Context resources are context data generators or consumers and can be classified as passive or active context resources. Passive context resources (such as physical or virtual sensors) aim at capturing and storing context data while active context resources (such as actuators or facilities) can interact directly with the context and modify the context state. Context policies are used to represent the set of conditions used to guide and control all interactions within the context. The set of adaptation actions are specified in design time and are executed in run-time to enforce the context policies.

The global control loop objectives are implemented as an energy-aware and self-* controller by using a MAPE cycle paradigm which involves the Monitoring, Analyzing, Planning and Execution phases. During the monitoring phase, the context instance is updated with workload and energy related data collected from service centre computing resources and facility, as well as with QoS and SLA

related data of the incoming business process requiring execution. In the analysis phase the context instance is evaluated to determine if the predefined context policies are fulfilled. If there are no broken policies, the system is considered as being in an energy-efficient state and no adaptation action is necessary. If one or more policies are broken, an adaptation plan consisting of a set of healing actions is generated in the planning phase. The execution phase is running the adaptation plan actions, aiming at system transition into an energy-efficient state.

At the global control loop level we have defined three types of adaptation actions: computing resource oriented adaptation actions, facility resource oriented adaptation actions, and application oriented adaptation actions. Computing resource oriented adaptation actions are executed to enforce the set of predefined GPI and KPI indicators on the service centre computing resources. We defined two types of such actions: consolidation actions and DPM actions. Consolidation actions aim at identifying computing resources featuring inefficient workload distribution and taking balancing actions to distribute it in an energy efficient manner. Consolidation actions are implemented as activity migration or activity deployment actions. DPM actions aim at determining the over provisioned resources with the goal of putting them into low power states for minimizing energy consumption. Facility resource oriented adaptation actions such as adjust the room temperature or start the CRAC, can be enforced through the service centre facility active resources. Application oriented adaptation actions are executed upon the GAMES-enabled applications targeting their redesign for energy efficiency.

The global control loop decisions may also include other energy-aware context-based adaptivity actions such as minimizing the necessity of calling a remote service when one local similar service is available (minimize data/service transfer), minimizing the substitution of services during maintenance, optimizing the number of necessary backup operations or favoring the use of services that require low energy.

To determine the best adaptation actions for a given context situation, the service centre level controller uses reasoning / learning and data mining tools, what-if analysis tools and a knowledge base. To derive knowledge about the service centre and its energy efficiency, the GAMES framework integrates information models that uniformly represent the system historical energy consumption related data. The general approach is based on extracting domain knowledge base from large amounts of historical data by using data mining techniques. The historical energy consumption related data will be also used together with a traceability model to understand the impact of changes in the provisioning infrastructure on energy efficiency and service quality, in order to allow both operators and consumers to select the appropriate mix as needed. With the GAMES framework it will be possible to align business requirements such as optimize for low power demand providing response time up to 200ms versus optimize response time based on historical data and the currently monitored status. By combining at design and run time the historical, predictive, context

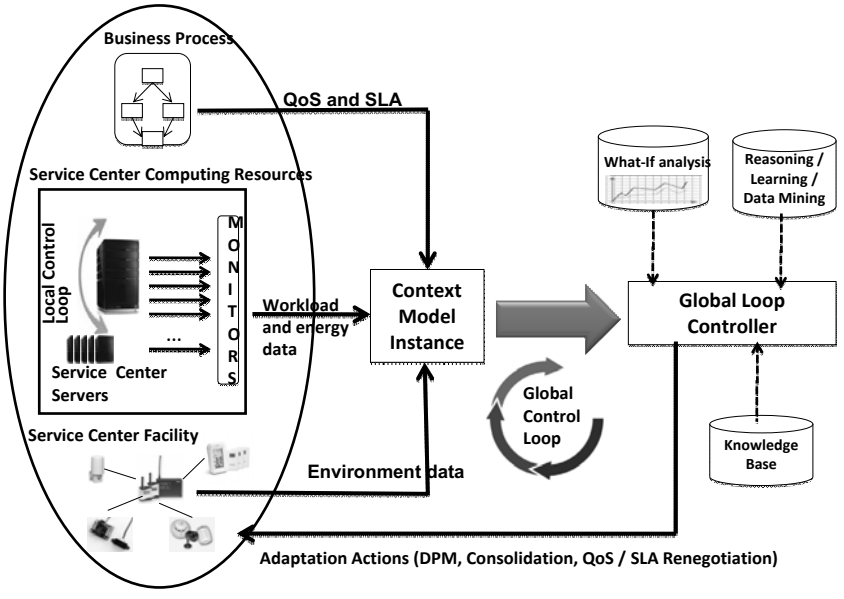

Fig. 4. Global control loop

and the externally available information with the GAMES knowledge base will allow the selection of the most adequate adaptation patterns and profiles.

6 Conclusions

This paper has presented the GAMES approach to design and manage energy-efficient service centres. For implementing in a successful way the GAMES concept of energy efficiency, new overall energy efficiency metrics are needed, which will be able to assess the energy efficiency and carbon emissions in an integrated way, combining the facility with the business/process and IT architecture levels, while the most popular ones nowadays (PUE and DCiE, defined by the GreenGrid consortium [5]), are dealing only with the facility level. With this regard, the GAMES project will define and introduce new energy efficiency and emissions metrics, the GAMES Green Performance Indicators.

The general approach of co-design and adaptivity both at service and at infrastructure layer need validation, both from a theoretical point of view and from experimentation. Models and tools to be developed must be sufficiently performant and the monitoring light enough not to overload the running system. Validation in the project is planned within two large data centres, on experimental settings.

Acknowledgments. This work has been partially supported by the GAMES project (http://www.green-datacenters.eu/) and has been partly funded by the

European Commission's IST activity of the 7th Framework Program under contract number ICT-248514. This work expresses the opinions of the authors and not necessarily those of the European Commission. The European Commission is not liable for any use that may be made of the information contained in this work.

References

1. Almeida, J., Almeida, V., Ardagna, D., Francalanci, C., Trubian, M.: Managing energy and server resources in hosting centers. In: Proc. ICAC (2006)
2. Anjomshoaa, A., Brisard, F., Drescher, M., Fellows, D., Ly, A., McGough, S., Pulsipher, D., Savva, A.: TheJob Submission Description Language (JSDL) Specification, Version 1.0. Technical report, Open Grid Forum (2005)
3. Ardagna, D., Cappiello, C., Lovera, M., Pernici, B., Tanelli, M.: Active energy-aware management of business-process based applications. In: Mähönen, P., Pohl, K., Priol, T. (eds.) ServiceWave 2008. LNCS, vol. 5377, pp. 183–195. Springer, Heidelberg (2008)
4. Chen, Z., Zhu, Y., Yuan, M.Y.: Towards self-optimization in utility computing using fuzzy logic controller (2005)
5. Belady, C. (ed.): Green grid data center power efficiency metrics: PUE and DCiE (2008)
6. EU Stand-by Initiative. 2010 Best Practices for the EU Code of Conduct on Data Centres. Technical report (December 2009)
7. Ferreira, A.M., Kritikos, K., Pernici, B.: Energy-aware design of service-based applications. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) ICSOC-ServiceWave 2009. LNCS, vol. 5900, pp. 99–114. Springer, Heidelberg (2009)
8. Gamberini, L., Jacucci, L.G., Spagnolli, A., Bjorkskog, C., Kerrigan, D., Chalambalakis, A., Zamboni, L., Valentina, G., Corradi, N., Zappaterra, P., Bosetti, G.: Technologies to improve energy conservation in households: The users' perspective, Maastricht. In: First European Conf. Energy Efficiency and Behaviour (October 2009)
9. Hasselmeyer, P., Koller, B., Schubert, L., Wieder, P.: Towards SLA-Supported Resource Management. In: Gerndt, M., Kranzlmüller, D. (eds.) HPCC 2006. LNCS, vol. 4208, pp. 743–752. Springer, Heidelberg (2006)
10. Jefferey, K., Neidecker-Lutz, B.: The Future of the Cloud (January 2010)
11. Koller, B., Schubert, L.: Towards autonomous SLA management using a proxy-like approach. International Journal of Multiagent and Grid Systems 3, 313–325 (2007)
12. Lu, Y., Chung, E., Simunic, T., Benini, L., Micheli, G.D.: Quantitative comparison of power management algorithms. In: Proceedings of the Conference on Design, Automation and Test in Europe (2000)
13. Salomie, I., Cioara, T., Anghel, I., Dinsoreanu, M.: RAP - A basic context awareness model. In: Proceedings of 4th IEEE International Conference on Intelligent Computer Communication and Processing, Cluj-Napoca, Romania, pp. 315–318 (2008)
14. Schmidt, N.-H., Erek, K., Kolbe, L.M., Zarnekow, R.: Towards a Procedural Model for Sustainable Information Systems Management. In: Proceedings of the 42nd Hawaii International Conference on System Sciences, HICSS 2009, Hawaii, USA, pp. 1–10. IEEE Computer Society, Los Alamitos (2009)
15. U.S. Environmental Protection Agency. Report to congress on server and data center energy efficiency public law 109-431. Technical report, ENERGY STAR Program (August 2007)

Modeling Deployment of Enterprise Applications

Susanne Patig

University of Bern, IWI, Engehaldenstrasse 8, CH-3012 Bern
susanne.patig@iwi.unibe.ch

Abstract. Deployment comprises installing, activating and updating applications. The applications to be deployed usually require certain conditions that can refer to hardware capabilities, other software (dependencies), physical artifacts or configuration. Deployment planning aims at satisfying the applications' prerequisites without violating the hardware's capabilities. This paper presents the domain-specific language *ADeL (Application Deployment Language)* that was designed to describe and validate deployment plans. The ADeL metamodel was implemented within the Eclipse Modeling Framework (EMF) and contains a set of OCL constraints (implemented with the tool *Topcased*) to enable the automatic validation of deployment plans.

1 Introduction

As a result of mergers, acquisitions and evolving business needs, the application software (*applications*) and the *IT infrastructure* (hardware, system software and network) of a company change. Typical *enterprise architecture (EA)* approaches do not trace in detail the applications to the used IT infrastructure [2]. Such tracing information, however, is needed for IT consolidation, dependency analysis and the management of application portfolios [2].

This paper tries to close the gap between applications and IT infrastructure by dealing with deployment planning in data centers. *Deployment* comprises all activities that make some released software ready for use, namely installation, activation and updating [5]. Especially installation plays a key role as it has to obey software dependencies and sets up the initial relationship between applications and IT infrastructure. Installation presupposes the selection of the appropriate hardware, which does not only depend on the software's requirements, but also on deployment *goals* (e.g., performance or failure safety) and deployment *restrictions* (such as the current IT resource allocation or co-location constraints). Thus, deployment provides the basis for IT consolidation from a data center point of view.

Modeling deployment has several *advantages*: First, all information about the given hardware and its capabilities as well as about the applications and their requirements are *described.* Secondly, model validation can check the modeled deployment scenarios for realizability *before* installation starts and, thus, prevent installation breakdowns or even trigger the provisioning of missing hardware or software. Finally, software configuration [20], installation guidelines as well as virtualization layers [21] can be *generated* from the models.

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 253–266, 2010.
© Springer-Verlag Berlin Heidelberg 2010

As a first step towards putting the advantages listed above into practice, this paper concentrates on modeling and validating deployment scenarios. In Section 2, the requirements of deployment planning are sketched; they are derived from real-world cases of deploying enterprise applications. Section 3 summarizes existing approaches for deployment modeling. As none of these approaches meets the requirements gathered in Section 2, Section 4 proposes a new domain-specific language called ADeL (*A*pplication *De*ployment *L*anguage) and applies it to a real-world case. Section 5 reflects on the language design process, and Section 6 contains the overall conclusions and an outlook.

2 Requirements of Application Deployment

The requirements of planning the deployment of complex applications in data centers are derived from two real-world cases, namely the installation of SAP SCM in the SAP UCC in Magdeburg and the installation of the content management system openCMS (http://www.opencms.org/) in the VLBA Lab Magdeburg[1]. During requirements elicitation, particular system's instances as well as documents related to installation were analyzed, and the staff involved in the installation process was interviewed. The complete description of the cases can be found in [17]; only the elicited requirements are listed in the following.

In detail, an approach that supports the deployment of complex applications in data centers must be capable to express (*expressiveness requirements*):

[Rq1] The available *hardware* and its technical characteristics (*capabilities*). The most important technical characteristics are CPU type (restricting the operating system) and CPU count as well as the sizes of random access memory (RAM) and hard disk (HD).

[Rq2] All that is to be deployed and has certain prerequisites, i.e., application components, system software or installation media. The prerequisites can refer to *hardware* capabilities, other *software* (i.e., dependencies), physical *artifacts* (e.g., executables, configuration files) or *configuration* activities (defining ports, IP addresses etc.). The objects to be deployed are called *requirement units*.

[Rq3] The direct or indirect *assignment* of requirement units to hardware; indirect assignments are realized transitively via intermediate requirement units.

[Rq4] Deployment *constraints*, e.g., whether or not some software units are allowed to run on the same server.

[Rq5] *Choices* in realizing some functionality (e.g., 'database functionality') by distinct software products (e.g., Oracle, DB2, MaxDB).

Interviews with the staff involved in the installation of complex applications made it clear that the expressive power reflected by the requirements [Rq1] to [Rq5] should be realized by a *modeling language* [Rq6] that is *SI*mple, *E*xtensible and *G*eneral; I call this the *SIEG principle. Simplicity* [Rq7] means that only a small set of well separated

[1] All names of products are trademarks, service marks or registered trademarks of the respective companies.

concepts should be used as the cognitive capacity of humans is limited [3]. *Extensibility* [Rq8] enables the adaptation of the new approach to specific deployment situations (by adding metamodel elements) and unanticipated usage scenarios (by adding (meta-) model transformations in model-driven development). As real-world application landscapes and hardware are heterogeneous, the new approach should be *general* [Rq9], i.e., independent of particular hardware, software and software architecture. Finally, the interviewed staff appreciated checking modeled deployment plans for their *validity* [Rq10] prior to installation as an important benefit of modeling.

The next section analyses whether or not the existing approaches in the field of deployment modeling satisfy the elicited requirements.

3 Existing Approaches in the Field of Deployment

Software deployment has been largely neglected in academic discussions. The earliest papers, e.g. [5], coin the key terms and classify existing technologies such as installers or package mergers. At that time, the *aim* was to *automate* deployment by *tools*, and modeling was just a prerequisite to achieve this aim. In contrast, the primary purpose of more recent approaches - such as ArchiMate [12] or UML Deployment Diagrams [15] - is to *describe* deployment scenario by *models*. These models are mainly used to document deployment or to discuss deployment scenarios with stakeholders, but not for automation.

Table 1 groups the existing approaches according to their (primary) aim. The term 'approach' is used as a generic name for tools, modeling languages and standards. Within each group, the approaches are ordered chronologically. The columns reflect the expressiveness requirements [Rq1] to [Rq4] of Section 2; requirement [Rq5] is omitted as it is not satisfied by any of the existing approaches. Concerning *software*, Table 1 distinguishes between conceptual *units,* such as components and packages and their relationships, and *physical artifacts*, e.g., executables or configuration files.

In the following, it is analyzed how the existing approaches meet the expressiveness requirements (see Section 2). Afterwards, the requirements related to the SIEG principle are investigated. Proprietary tools that automate the deployment of particular applications are disregarded here as they are not general [Rq9] and do not rely on modeling [Rq6].

The *Software Dock framework* [9] supports the deployment of software components via the Internet and assumes that software producer and consumer negotiate during deployment. The software to be deployed ('software families'[2]) is described by the *Deployable Software Description (DSD)*, a language with textual concrete syntax that produces hierarchical schemas. The tool ORYA [13] allows the description and execution of a general deployment process (install, activate, deploy). Here, the focus is on automating the deployment workflow (involving legacy tools), but not on matching requirements of the software to the existing IT infrastructure. The tool *ADAGE* (a research prototype) can deploy components on grids [11], given that the components follow the CORBA component model, a distinct programming model. Only ADAGE satisfies requirement [Rq1] as it provides constructs to describe hardware; see Table 1.

[2] The term is not used in the sense of software product lines.

All automation-oriented approaches naturally have an *IT view* on deployment, i.e., they solely consider software, IT infrastructure and the corresponding assignments. In contrast, approaches that *describe deployment* can adopt an IT or a business view. The *business view* on deployment starts from business goals and business processes and links them to types of applications; it is typical for approaches stemming from the field of enterprise architecture (ArchiMate [12]) or go even beyond (MEMO ITML [8]). Both approaches provide distinct layers and diagrams; here, only the ones relevant to deployment are considered; see Table 1. In contrast, UML deployment diagrams [15], IBM topologies [14], [17] and the Common Information model[3] (CIM) [6], which is implemented in configuration management databases (CMDB), represent the *IT view* on deployment. Some approaches support both views, see Table 1.

Table 1 illustrates that all approaches that aim at descriptions of deployment scenarios use distinct types of units to express the software to be deployed (e.g., UML [15]: specification, entity, process, service) and the IT infrastructure that is the target of deployment (e.g., UML [15]: execution environment, application server, client workstation, mobile/embedded device; ArchiMate [12]: device, system software; IBM topologies [14]: application server, WINDOWS/UNIX/LINUX/General OS etc.). Often these types are rather vaguely defined. Moreover, the description-oriented approaches share the usage of distinct link types (see Table 1), e.g., *dependencies* between software components, *realization links* that connect some conceptual deployment entity to another unit that will replace it (after deployment) and *assignment links* showing the IT infrastructure that will run (host) the deployed software unit. Often the semantics of the link types overlaps [14], [17]. In the resulting diagrams, the same units can be connected by several links, which affects understandability (see [17] for an example).

Altogether, all description-oriented approaches satisfy the requirements [Rq1] to [Rq3] as they can express hardware (IT infrastructure), software and the corresponding assignments. Minor restraints refer to the representation of artifacts: Explicitly representing artifacts is only possible with the UML by stereotypes such as ‘script’, ‘source’ and ‘executable’ [15]. In the IBM topologies, artifacts correspond to either attributes or additional constructs. Gaps exists for the other expressiveness requirements: The UML relies on the OCL [16] to specify any kind of *deployment constraint* [Rq4], whereas the IBM topologies support a limited set of constraint types by particular constructs [14]. Deployment *choices* [Rq5] are not covered by the existing approaches, except for indirect support within the IBM topologies (see [17]).

The postulated SIEG principle ([Rq7] to [Rq9]) is mostly not obeyed: Though the number of model elements in the approaches that automate deployment is small [Rq7], only the ORYA approach is general [Rq9]. Extending [Rq8] the automation-oriented approaches for other deployment situations or usage scenarios is not possible.

In general, the large number of distinct unit and link types of the description-oriented approaches interferes with simplicity: In its common part, the CIM consists of 408 classes and 201 associations and is, thus, not simple [6]. However, because of being a standard, the CIM is general [Rq7] and adaptable to any deployment situation [Rq9]. The situation for the UML is analogous - the language is not simple, but both

[3] The CIM Specification V2.0 has been available sind June, 1999 [7].

Table 1. Existing approaches related to modeling deployment

Approach	Domain	Focus	Elements					Links				Constraints
			Software			Infrastructure		Deployment			General	
			Units	Relationship	Artifacts	Units	Relationship	Depend	Realize	Assign	Group	
Automation												
DSD (Software Dock) [9]	SW	IT	SW family with properties	(Properties)	X	—	—	X	—	—	Nesting	Assertions
ORYA [13]	APP	IT	APP	See Links	Resource	—	—	X	—	—	—	Hardware, Software
ADAGE [11]	CORBA APP	IT	SW Entities (processes)	Connection	Codes to load	(Compute Node)	—	—	—	—	—	Attributes of SW entities
Description												
UML [15]	SW	IT	Component Diagram			Deployment Diagram						
			Component (stereotypes)	Connector	X	Node (stereotypes)	Communication path	X (deploy)	Dependency	(Nesting)	X, Agg., Comp.	(OCL)
CIM [6]	ITM	IT/B	CIM_Managed-Element (sterotypes)	Dependency	Setting-Data	ComputerSystem, Logical-/Physical- (stereotypes),	Device_ Connection	X	—	(X)	Collection	—
ArchiMate [12]	EA	IT/B	Application Layer			Dependency Layer		Used by	X	X	X, Agg., Comp	—
			APP Component (Collaboration)	APP Interaction	—	Node (Device, system SW)	Communication link					—
IBM Topologies [14]	J2EE APP	IT	Component Deployment Unit	See Links	Implicit	Node (stereotypes)	See Links	X	X	Hosting	Membership	X (Links)
MEMO-ITML [8]	ITM	B/IT	Software (roles)	(Requires; Specialization)	—	Hardware (roles), Network/-Device	(Comprises; Specialization)	(X)	(X)	X	(Comprise)	OCL

Abbreviations:
Agg: Aggregation, APP: Application, B: Business View, Comp: Composition, Config: Configuration, EA: Enterprise Architecture, ITM: IT Management, SW: Software

general and extensible (because of stereotyping and the available tools for model-driven development). ArchiMate is not really simple and not really general (as it focuses on 'service-orientation' [12]), yet extensible. At the aggregated level, the number of constructs of the IBM topologies is comparatively small (units, components, deployment units, nodes, links), but they are broken down into a vast number of fine-grained, predefined constructs that mostly refer to IBM products. Thus, the IBM topologies are not general [14], [17], but they can be extended by user-specific constructs.

Validity checks [Rq10] are inherent in the automation-oriented approaches, but restricted to the particular tools and technologies these approaches support. Among the description-oriented approaches, only the IBM topologies offer some form of model validation by checking the modeled constraints [14], [17].

To sum it up, the main deficiencies of the existing approaches are missing simplicity [Rq7] as well as lack of support for deployment choices [Rq5], constraints [Rq4] and model validation [Rq10]. The domain-specific language ADeL proposed in the next section was designed to overcome these deficiencies.

4 ADeL – The Application Deployment Language

4.1 ADeL Metamodel

The ADeL metamodel consists of the abstract syntax depicted in Fig. 1 as well as a set of constraints that are expressed by OCL invariants.

The core ADeL metamodel elements are `units`; each `unit` can be linked (`isLinked`) to an arbitrary number of other `units`. As an abstract super class, a `unit` defines the common properties of both `RUnits` (*requirement units*, see [Rq2] in Section 2) and `hardware`: the `name`, an identifier `id` (if units cannot be recognized from their names), the type of CPU (`CPU_type`), the total number of CPU cores (`CPU_count`), the sizes of hard disk (`HD`) and random access memory (`RAM`). All properties except for the name are optional.

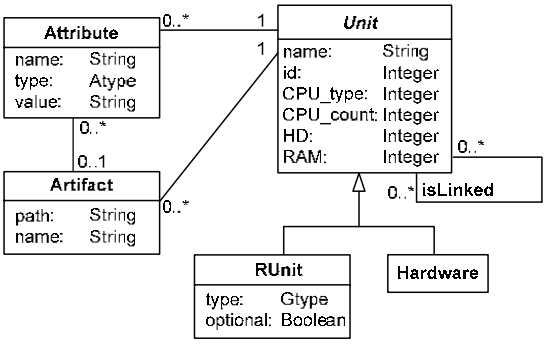

Fig. 1. Abstract syntax of the ADeL metamodel

Units of the subtype `hardware` represent physical capabilities [R1] to host some `RUnit`(s). Basically, the prerequisites for `RUnits` can refer to hardware, software, physical artifacts or configuration (see [Rq2] Section 2). Hardware prerequisites are expressed by the properties listed above and paths to hardware [Rq3], whereas software prerequisites (dependencies) correspond to links (`isLinked`) between `RUnits`.

Units of the subtype `hardware` represent physical capabilities [R1] to host some `RUnit`(s). Basically, the prerequisites for `RUnits` can refer to hardware, software, physical artifacts or configuration (see [Rq2] Section 2). Hardware prerequisites are expressed by the properties listed above and paths to hardware [Rq3], whereas software prerequisites (dependencies) correspond to links (`isLinked`) between `RUnits`.

The predefined properties of `units` express standard deployment needs. Unforeseen prerequisites or capabilities can be modeled by `attributes` [Rq8]. A `unit` may be associated with an arbitrary number of `attributes`.

`RUnits` have the additional properties `type` and `optional`. The property `type` indicates whether a `RUnit` is elementary (`GType = E`), which is the default, or groups other `RUnits`. Groups of `RUnits` are either conjunctive (`GType = A`, i.e., all grouped `RUnits` must be deployed), disjunctive (`GType = O`, i.e., at least one of the contained `RUnits` must be deployed) or exclusive (`GType = X`, i.e., one and only one `RUnit` of the group is to be deployed). Often such groups are conceptual, i.e., serve the purposes of structuring ADeL models or preparing deployment choices [Rq5]. The property `optional` describes whether or not some `RUnit` must be deployed at all.

Physical artifacts are needed for deployment execution, IT operations or result from configuration activities (e.g., configuration files, start profiles). They can be represented by the metamodel element `artifact`. A `unit` can be linked to any number of `artifacts`. The location of an `artifact` must always be given (property `path`), whereas the property `name` as well as associations to `attributes` are optional.

The ADeL metamodel is supplement by two OCL invariants that are independent of deployment, namely: (1) A node of the type `RUnit` must exist that is not the target of any association *isLinked*. This node is called the *root node*. (2) Identical `hardware` units agree in the values of their capabilities (CPU type and count as well as the sizes of RAM and HD).

The ADeL metamodel was implemented within the Eclipse Modeling Framework EMF 2.4.2 [4] and Eclipse 3.4 Ganymed. The current concrete ADeL syntax consists in to the graph provided by the EMF.Edit framework [4]; see Fig. 2 in Section 4.3.

4.2 Deployment Constraints

An instance of the ADeL metamodel, i.e., an ADeL model, corresponds to a *deployment plan* that successively assigns the `RUnit` of the root node (which is to be deployed) to hardware (leaf nodes); an example is given in Section 4.3. Only valid deployment plans can be effectuated. To be *valid*, a deployment plan (ADeL model) must satisfy

all the RUnits' prerequisites (*deployment constraints*) without interfering with the hardware's capabilities (*hardware constraints*). Both groups of constraints are specified as OCL invariants [16] and explained in the following.

Deployment and hardware constraints rely on deployment paths, which exploit the association isLinked between units: A *deployment path* always starts at a RUnit and terminates at a unit of the types hardware or RUnit, respectively. In the first case, the start node is said to be *deployed* and *undeployed* otherwise.

Deployment constraints comprise the invariants [deployed] and [choice]. The invariant [deployed] requires that all RUnits that are not optional must be either linked to another unit (the child, which can be hardware) or belong to a non-elementary RUnit:

```
context RUnit
inv deployed :
    if self.optional<>true then
        (self.isLinked->size()>=1 or
        RUnit.allInstances()->exists(r|r.isLinked->includes(self)))
    else true endif
```

The deployment of non-optional, non-elementary RUnits is guarded by the invariant [choice]: If the group type (GType) of a non-elementary RUnit is A/O/X, then for all/at least one/exactly one non-optional member(s) of the group a deployment path ending at a unit of the type hardware unit must exist:

```
context RUnit
inv choice :
if self.optional<>true then
  if type=GType::A then  self.isLinked->forAll(u|u.oclAsType(RUnit).optional<>true implies u.path()) else
    (if type=GType::X then self.isLinked->one(u|u.oclAsType(RUnit).optional<>true implies u.path()) else
      (if type=GType::O then self.isLinked->exists(u|u.oclAsType(RUnit).optional<>true implies u.path())
        else true endif)endif)
    endif
else true endif
```

The recursive help function path()returns true when the last node of a deployment path is of the type hardware and false otherwise:

```
context Unit
def: path() : Boolean = (self.isLinked->iterate(
        u: Unit; deployed:Boolean=false|
        if u<>null then
            if u.oclIsTypeOf(Hardware) then true else u.path() endif
        else false
        endif))
```

Hardware constraints, which are specified by the invariants [HD], [RAM], [CPU_count] and [CPU_type], guarantee that the aggregation of requirements along *all* deployment paths that target at the *same* hardware unit observe the hardware's capabilities. Consequently, these invariants must be specified in the context hardware, and navigation occurs along the reverse deployment path, i.e., from the leafs to the root of an ADeL model. Reverting the deployment path is achieved by iterating over all instances of the type RUnit and selecting parent RUnits that are linked with the corresponding (child) RUnit; see, e.g., the invariant [HD] :

```
context Hardware
inv HD:  RUnit.allInstances()->select(r|r.isLinked->exists(m|m.name=self.name))->
    collect(u|u.aggrHD(u.HD))->sum()<=self.HD
```

All invariants of hardware constraints use help functions for specific *aggregations* along the deployment path, i.e., (1) to sum up the required HD size (help function `aggrHD()`, see below), (2) to find the maximum required RAM size or CPU count or (3) to check the equality of the required CPU type. The invariants `RAM`, `CPU_count`, `CPU_type` and their help functions are specified analogously to the invariant `HD`.

```
context RUnit
def: aggrHD(hd: Integer) : Integer=(
        let pv: RUnit =
        RUnit.allInstances()->select(isLinked->includes(self))->asOrderedSet()->first() in
        if pv=null then hd else pv.aggrHD(hd+pv.HD) endif)
```

All OCL constraints for ADeL models are implemented with the Eclipse-based toolkit *Topcased* [22]. Topcased is integrated with the EMF modeling framework, includes a comfortable OCL editor and can evaluate both descriptive and statistic OCL constraints. The implemented ADeL OCL constraints are tied to the ADeL metamodel only and can be easily attached to all ADeL models. Moreover, standard OCL can be used to define additional constraints if necessary. Fig. 3 in the next section shows the result of evaluating the ADeL OCL constraints for the ADeL model of Fig. 2.

4.3 Application Example

Fig. 2 depicts the ADeL model for the deployment of SAP SCM (a real-world case investigated in [17]). On the right hand side of Fig. 2, examples for the properties of each metamodel element are given.

The SAP SCM deployment case has the following characteristics: The application software to be deployed (SAP SCM) consists of several `RUnits` (SAPKernel, DBSID, LID, OptID); an additional `RUnit` 'Install' expresses the installation requirements (installation media, JRE). The `RUnit` 'OptID' is optional. The 'SAPKernel' is a conjunctive `RUnit` (`GType = A`) since both the global instance 'SAPSID' and the central instance 'DVEBMG' as well as the 'C++ Runtime environment' must be installed. In contrast, the particular software products to be deployed for the `RUnits` 'DBSID', 'LID' and 'OptID' must be chosen from a set [Rq5]; thus, these `RUnits` are exclusive groups (`GType = X`).

The graph structure of ADeL models supports the distribution of some `RUnit` over several `hardware` units, though this did not happen in the real-world case. The particular deployment of a `RUnit` is visible from the deployment path to a `hardware` unit. For example, the `RUnit` 'DBSID' is realized by the `RUnit` 'Oracle 10.2' (operating system 'HP-UX 11.23') and installed on the `hardware` unit 'HP Integrity rx8620'. The same `hardware` unit also hosts the `RUnit` 'LID' that requires the `RUnit` 'HP-UX 11.23' as operating system. In general, a `hardware` unit can host more than one `RUnit`.

The SAP SCM deployment case in Fig. 2 uses `attributes` and `artifacts`. For example, the `attribute` 'SWAP', which is associated with the `RUnit` 'SAPKernel', expresses the additional requirement of 20 Gigabyte (GB) of SWAP space. (All sizes are specified in GB in this paper). Moreover, the `RUnit` 'SAPSID' is associated with `artifacts` that specify the locations (`path`) of the directory `/sapmnt` and of the start `profiles`; the names of the `artifacts` help in distinguishing them from each other.

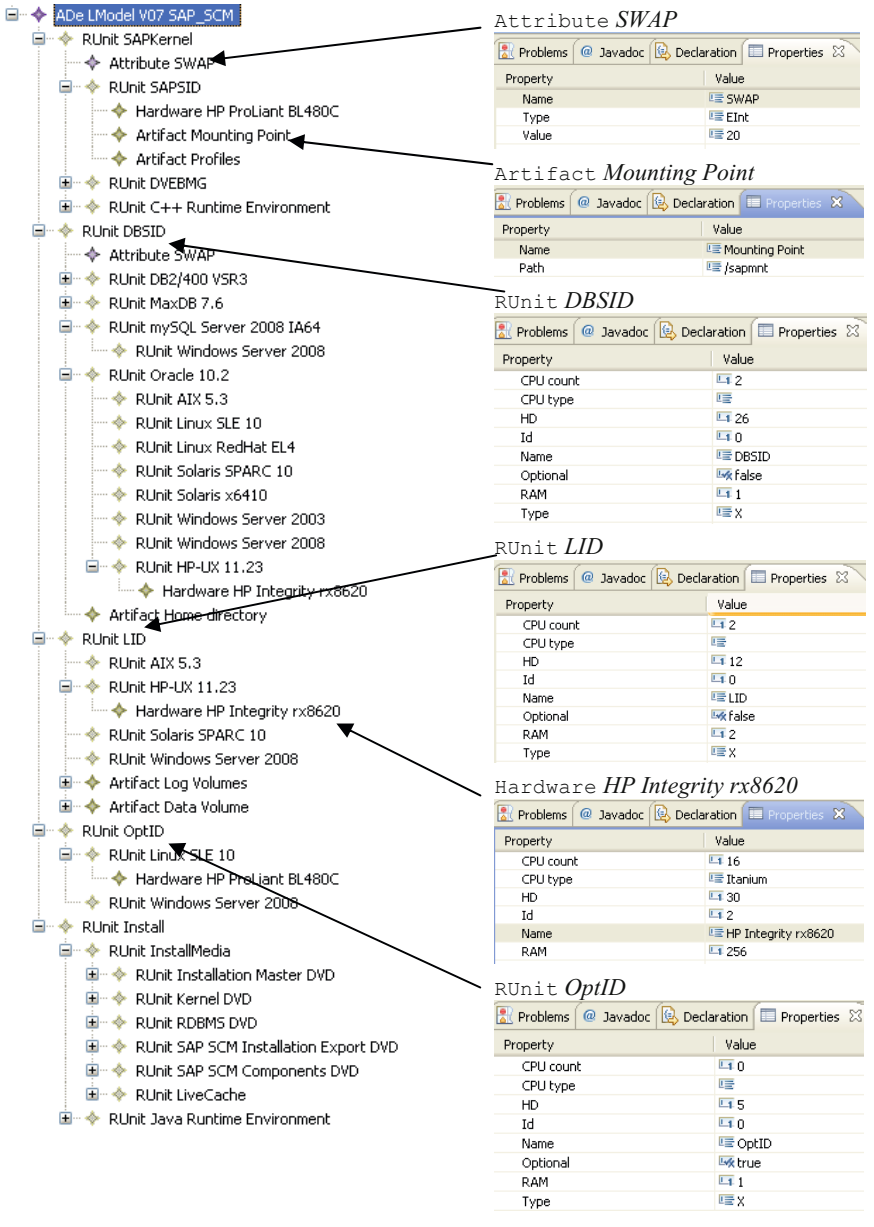

Fig. 2. Concrete syntax of the ADeL metamodel for the example of SAP SCM (extract)

Note that it is not necessary to detail all properties of `units` in an ADeL model as the ADeL OCL statements filter out null values. Thus, ADeL models can concentrate on the information essential for a deployment scenario.

The results of evaluating the ADeL OCL constraints for the ADeL model of Fig. 3 with the tool *Topcased* are shown in Fig. 3: For the purpose of illustration we assume

that the `hardware` unit 'HP Integrity rx8620' already hosts other applications so that only 30 GB HD are available. The `hardware` unit 'HP Integrity rx8620' is the target of two deployment paths (from the `RUnits` 'DBSID' and 'LID', respectively; see Fig. 2). Along each path, the required hard disk space `HD` is smaller than the available `HD` of the `hardware` unit 'HP Integrity rx8620', namely 26 GB and 12 GB for the paths starting from the `RUnits` 'DBSID' and 'LID', respectively. However, as both `RUnits` are to be installed on the same hardware, the aggregated HD requirement amounts to 38 GB, which exceeds the HD capacity of the `hardware` 'HP Integrity rx8620'. For that reason the evaluation of the invariant `[HD]` fails in Fig. 3. All other ADeL OCL constraints are observed by the deployment plan of Fig. 2.

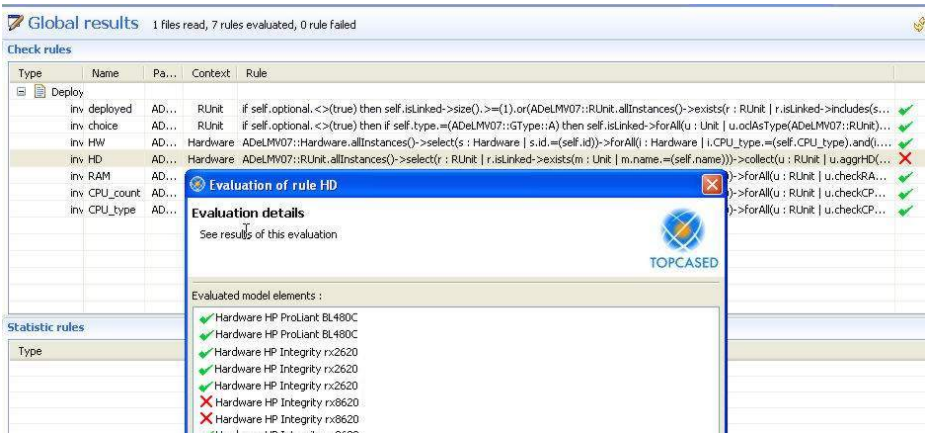

Fig. 3. Evaluation of the ADeL OCL constraints for the sample ADeL model

5 Reflections on the ADeL Design Process

The current version of the ADeL is the result of several major revisions. In this section I reflect on them as they possibly hint at generic issues in the design of domain-specific languages.

The advantages of deployment modeling listed in Section 1 call for methods and tools from model-driven software development. For that reason, the first version of the ADeL was a UML profile. However, the depth of inheritance within the UML infrastructure [23] turned out to be burdensome. To get rid of all the unnecessary, inherited semantics, a distinct, 'pure' ADeL metamodel was built, which can be seen as a UML profile at the level of `Infrastructure::Core::Abstractions::Classifier` [23].

Probably inspired by the existing deployment modeling approaches (see Table 1 in Section 3), the first versions of the ADeL metamodel were rich in specific types of units and links. For example, the metamodel element `unit` was specialized in `abstract` and `concrete unit`. `Concrete unit` contained the subtypes `hardware` and `software`, whereas `abstract unit` consisted of the subtypes `concept` (elementary unit) and `set` – to structure deployment plans. Moreover, it

was distinguished between the link types `membership`, `requires` and `realizes`. Though facilitating modeling by the specific semantics, the extensive type vocabulary became heavy in writing OCL constraints: Basically, casting was possible between types of units, but made the OCL statements rather complicated. Even more severe were the problems in navigating along deployment paths that rely on distinct link types. As a result, the final version of the ADeL does not differentiate between link types and has only one layer of inheritance. Note that no information is lost by using only the generic association `isLinked` - as the `units` that are connected define the semantics.

Though the OCL is a natural choice to express constraints [Rq4] in the field of model-driven development, it is probably not the best one for reverse navigation in graphs: The OCL statements defining these navigations require the usage of the predefined operation `allInstances()`, which is deprecated because it makes navigation difficult to understand [19] and increases the worst case complexity of OCL evaluation [1]. However, the latter argument can be mitigated by the fact that the number of instances of each type within ADeL models is small, even in real-world deployment scenarios.

Altogether, the recursive functions to navigate in the graphs of the ADeL models are procedural rather than declarative (see, e.g., the help function `path()` in Section 4.2). Eventually this observation indicates that the OCL is not the appropriate means to validate deployment plans.

6 Criticism and Future Research

Probably the main objection to the ADeL approach is over simplification, as the abstract syntax is a graph of linked units. However, recent research on enterprise architecture has shown that linked units are the basis to generate any kind of EA visualization and to exchange EA models between tools [10].

By using the ADeL language, it is very easy to assign the evolving set of applications in a data center to the existing IT infrastructure and, at the same time, to assure the validity of these assignments. In detail, the domain-specific language ADeL satisfies all the requirements listed in Section 2: The assignment [Rq3] of software or other units [Rq2] to some hardware [Rq1] as well as deployment choices [Rq5] can be described and validated [Rq10]. To specify other deployment-related constraints [Rq4], the OCL [22] can be used. Because of relying on just three mandatory, non-abstract metamodel elements (`RUnit`, `hardware`, `isLinked`), the modeling language ADeL [Rq6] is simple [Req7], yet extensible [Req8] – by the metamodel elements `attribute` and `artifact`. Moreover, all metamodel elements are independent of any real-world entities, tools and modeling languages and, thus, generic [Req9]. Finally, because of staying within the frameworks of model-driven software development, the ADeL language can be extended to a full-fledged approach of model-driven deployment that generates, e.g., installation guidelines and virtualization layers. Such an extension is a goal for future research.

The application of the ADeL to model and validate real-world deployment plans was shown. Currently, a more sophisticated editor is prepared to make modeling with the ADeL more convenient and appealing.

In contrast to ordinary (i.e., not extended) feature diagrams, which are validated based on propositional logic, the validation of ADeL deployment plans by the OCL introduces the expressiveness of predicate logic. Because of the promising results for extended feature diagrams [18], we intend to achieve an optimization of deployment plans by a transformation to constraint satisfaction problems.

References

[1] Altenhofen, M., Hettel, T., Kusterer, S.: OCL support in an industrial environment. In: Auletta, V. (ed.) MoDELS 2006. LNCS, vol. 4364, pp. 169–178. Springer, Heidelberg (2007)

[2] Aier, S., Riege, C., Winter, R.: Unternehmensarchitektur-Literaturüberblick und Stand der Praxis. WIRTSCHAFTSINFORMATIK 40, 292–304 (2008)

[3] Anderson, J.R.: Cognitive Psychology and its Implications, 5th edn. Worth, New York (2000)

[4] Budinsky, F., Steinberg, D., Merks, E., Ellersick, R., Grose, T.J.: Eclipse Modeling Framework: A Developer's Guide. Addison-Wesley, Boston (2004)

[5] Carzaniga, A., et al.: A Characterization Framework for Software Deployment Technologies. Techn. Report CU-CS-857-98, Dept. of Computer Science, University of Colorado (1998)

[6] Distributed Management Task Force (DMTF): Common Information Model (CIM) Standards. CIM Schema Version 2.24.0, `http://www.dmtf.org/standards/cim/cim_schema_v2240/` (2010)

[7] Distributed Management Task Force (DMTF): CIM FAQ (2010), `http://dmtf.org/about/faq/cim_faq#C4`

[8] Frank, U., et al.: ITML: A domain-specific modeling language for supporting business-driven IT Management. In: Proc. 9th OOPSLA Workshop on Domain-Specific Modeling. DSM Forum (2009), `http://www.dsmforum.org/events/DSM09/Papers/Heise.pdf`

[9] Hall, R.S., Heimbigner, D., Wolf, A.L.: A Cooperative Approach to Support Software Deployment Using the Software Dock. In: Proc. 21st Int. Conf. on Software Engineering (ICSE 1999). ACM Press, New York (1999)

[10] Kruse, S., et al.: Decoupling Models and Visualisations for Practical EA Tooling. In: Dan, A., Gittler, F., Toumani, F. (eds.) ICSOC/ServiceWave 2009 Workshops. LNCS, vol. 6275, pp. 62–71. Springer, Heidelberg (2010)

[11] Lacour, S., Pérez, C., Priol, T.: Generic Application Description Model: Toward Automatic Deployment of Applications on Computational Grids. Rapport de Recherche No. 5733, INRIA, Rennes (2005)

[12] Lankhorst, M., et al.: Enterprise Architecture at Work: Modeling, Communication, Analysis. Springer, Berlin (2005)

[13] Lestideau, V., Belkhatir, N.: Providing highly automated and generic means for software deployment process. In: Oquendo, F. (ed.) EWSPT 2003. LNCS, vol. 2786, pp. 128–142. Springer, Heidelberg (2003)

[14] Makin, N.: Deployment modeling in Rational Software Architect Version 7.5, Part I & II (2008), `http://www.ibm.com/developerworks/rational/library/08/{1202|1230}_makin/`

[15] Object Management Group (OMG): Unified Modeling Language: Superstructure, Version 2.2, formal/2009-02-02 (2009), `http://www.omg.org/`

[16] OMG: UML 2.0 OCL Specification, formal/2006-05-01 (2006), http://www.omg.org/
[17] Patig, S., Herden, S., Zwanziger, A.: Modeling Deployment of Enterprise Applications Cases and Conclusions. Preprint No. 224, University of Bern, IWI (2009)
[18] Benavides, D., Trinidad, P., Ruis-Cortés, A.: Automated Reasoning on Feature Models. In: Pastor, Ó., Falcão e Cunha, J. (eds.) CAiSE 2005. LNCS, vol. 3520, pp. 491–503. Springer, Heidelberg (2005)
[19] Warner, J., Kleppe, A.: Object Constraint Language 2.0. Bonn: mitp (2004)
[20] Giese, S., Seibel, A., Vogel, T.: A Model-Driven Configuration Management System for Advanced IT Service Management. In: Proc. 4th Int. Workshop on Models@run.time at the 12th IEEE/ACM Int. Conf. on Model Driven Engineering Languages and Systems (MoDELS 2009). CEUR Workshop Proceedings, vol. 509, pp. 61–70 (2009)
[21] Galan, F., Lopez de Vergara, J., Fernandez, D., Munoz, R.: Scenario-based Configuration Management for Flexible Experimentation Infrastructures. In: Proc. 5th Int. IEEE Conf. on Testbeds and Research Infrastructures for the Development of Networks & Communities. IEEE Press, Los Alamitos (2009), http://doi.ieeecomputersociety.org/10.1109/TRIDENTCOM.2009.4976228
[22] Project Group: Topcased, Version 2.6.0, http://www.topcased.org
[23] Object Management Group (OMG): Unified Modeling Language: Infrastructure, Version 2.2, formal/2009-02-04 (2009), http://www.omg.org/

Closing the User-Centric Service Coordination Cycle by Means of Coordination Services

Hans Weigand[1], Paul Johannesson[2], Birger Andersson[2],
Maria Bergholtz[2], and Jeewanie Jayasinghe Arachchige[1]

[1] Tilburg University, P.O. Box 90153,
5000 LE Tilburg, The Netherlands
{H.Weigand,J.JayasingheArachchig}@uvt.nl
[2] Royal Institute of Technology
Department of Computer and Systems Sciences, Sweden
{pajo,ba,maria}@dsv.su.se

Abstract. In the future vision of an Internet of Services, users take an active role in service selection and composition. In this context, web services are mostly interfaces to real services and can be classified as coordination services with respect to the latter. To enable users to perform service composition, the effect of the coordination services must be described in such a way that users are not only able to discover services but also to detect and prevent possible conflicts in their composition. To meet these requirements, a service description language for coordination services is proposed based on the REA business ontology.

Keywords: Internet of Services, service design, REA, IOPE.

1 Introduction

In spite of considerable progress that has been made in the area of Service Oriented Computing, the impact on society has still been limited. There is not yet such a thing as an Internet of Services that would allow users to integrate the services they want to use easily and seamlessly. It has been acknowledged that users must play a more active role in service composition, if only because of the long tail of specific and heterogeneous services around [AC06] that simply cannot be handled all by the IT departments. Enterprise mashups may provide an instrument to realize this service co-creation effort of users and developers [HS09]. In this paradigm, software resources such as *web services* are embedded in *widgets* that provide simple user interaction mechanisms to these resources; these (visual) widgets are combined by the user himself to create *mashups*.

However, users are not interested in composing web services as such. To them, these are merely interfaces to "real" services such as traveling, meeting support, child care, entertainment or car maintenance. Users have a need to *plan* and *coordinate* the services they use (cf. [Be04]).

Fig. 1 depicts the envisioned user-centric service coordination cycle: users compose mashups and interact with the widgets in them to access web services. The

P. Soffer and E. Proper (Eds.): CAiSE Forum 2010, LNBIP 72, pp. 267–282, 2010.
© Springer-Verlag Berlin Heidelberg 2010

web service typically supports the coordination with a service provider who offers a real-world service as part of a service bundle. The service affects a resource that concerns the user (the resource could be the user himself, for instance in the case of a hotel reservation). That web services themselves may be composite software entities is left out of this figure as being less relevant to the user, but is of course relevant to the software developer.

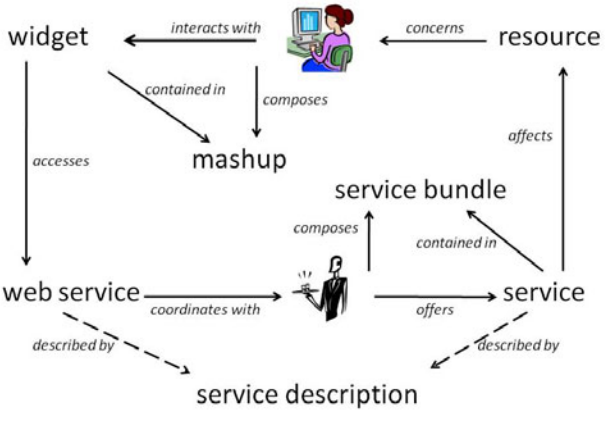

Fig. 1. User-centric service coordination cycle

Both web services and services need a description, but what should be in this description? In composing web services, a major challenge is to reconcile incompatible data representations. In composing services in the real world, a major challenge is to meet the constraints imposed by the fact that resources are scarce, can only be in one place at a time and often cannot be shared. For that reason, [PT09] argues convincingly that "asset-driven" service modeling will be a central concern in developing an Internet of Services and claims that "novel methodologies and tools are needed to support the modeling of the key assets of services". In our view, this modeling should support at least conflict prevention and conflict detection.

In order to make conflict prevention and conflict detection possible at all, web services must provide more information than input and output requirements such as we find in a WSDL document. What we need is a generic language to describe services, the resources they use as well as planned and actual events on the type level, Web services can use this language to represent the preconditions and effects of the real services they connect to as well as their own semantics. A mashup environment can collect and combine this information, integrate it with other sources such as the user's agenda (that should be represented in the same format) in order to provide the user with means for conflict prevention and conflict detection. On the basis of the service description and after instantiating the formulae with actual data, the user immediately knows the effect of a successful service invocation.

In this paper, we propose to ground the service description language in the REA ontology [Mc82] where we concentrate on coordination services as being of most interest to the user. An advantage of REA is that it has a very small set of basic

concepts, and therefore is relatively easy to understand. For conflict detection and prevention we propose a small set of coordination services and supporting concepts. By conflict detection and prevention we mean the following. Let *s* be a service that a user U intends to consume and let M be the set of resources and actors involved in the execution of *s*. Each *m* in M has a time-based context A(E,C) where E is a set of events planned for *m* and C a set of constraints on E. The goal of *conflict prevention* is to ensure that when *s* is added to the planning of U, all context constraints are still met, for all *m* in M. Typical events that stem from the planning of *s* are the start of the service execution and its ending. The goal of *conflict detection* is to check context constraints when an event *e* is added. Typical events are contingencies such as a flight being delayed. We can assume that in a future Internet of Services and Internet of Things, most of these events are generated without active user involvement. If *s* is a composite service, then the check should be done on all the services involved individually and jointly.

To arrive at rigorous and relevant research results, we use Peffers' design science phases [PT08]. The *problem identification and motivation* has been stated. Our *solution objective* is to develop a coordination service description language based on REA. In section 2, we position coordination services as the "glue" between software services and real services. In section 3 we work how to represent services and the coordination of services in REA. We identify a couple of very common coordination patterns (*design*). On the basis of that we show in section 4 how service descriptions can be developed that enable the required conflict detection (*design and development*). This is applied to the well-known hotel reservation case (*demonstration*); we adopted this case to facilitate comparison with other solutions.

2 Coordination as a Service

According to the OASIS reference architecture foundation for SOA, it is essential that participants can use a SOA-based system to realize actual effects in the world [OA09]. However, when talking about the real world, OASIS makes a sharp distinction between the social world and the physical world (in line with the Language/Action Perspective tradition and the communicative theory of Habermas [Di06]). Many, if not most, effects that are desired in the use of SOA-based systems are actually social effects rather than physical ones. For example, opening a bank account is primarily about the relationship between a customer and a bank – the effect of the opened account is a change in the relationship between the customer and the bank. For that reason, OASIS talks about social actions that result in social facts. "A social fact is an element of the state of a social structure that is sanctioned by that social structure". Social facts include policies and commitments where "a commitment is a social fact about the future: in the future some fact will be true and a participant has the current responsibility of ensuring that that fact will indeed be true". A completed business transaction establishes a set of social facts relating to the exchange; typically to the changes of ownerships of the resources being exchanged.

What remains a bit out of the OASIS picture is that social facts refer to physical world events, such as the delivery of a product. For a full account of service effects, this relationship between social facts and the real world must be made explicit (Fig.2).

It is widely recognized that input and output descriptions of web services, or its operations, are not sufficient for capturing the semantics that users need. Precondition and Effect descriptions have been added. Although WSDL-S provides a mechanism to include these attributes, it does not give guidance on how to do specify their contents. The OASIS reference model views web services as coordination mechanisms and emphasizes the social effects. How these are to be represented, and how these social facts relate to real-world business events is still to be worked out. In the following, we address this research gap by proposing the REA ontology for coordination service description.

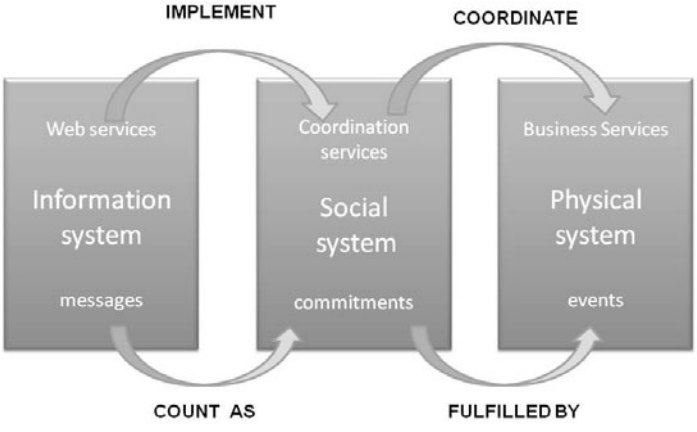

Fig. 2. Coordination services are the glue between web services and business services

3 REA - Background and Modifications

3.1 The REA Business Ontology

The Resource-Event-Agent (REA) ontology was first formulated in [Mc82] and has been developed further, e.g. in [UM03, GM99, Hr06]. The following is a short overview of the core concepts of the REA ontology based on [WJAB09].

A *resource* is any object that is under the control of an agent and regarded as valuable by some agent. This includes goods and services. The value can be monetary or of an intangible nature, such as status, health state, and security. Resources are modified or exchanged in processes. A *conversion process* uses some input resources to produce new or modify existing resources, like in manufacturing. An *exchange process* occurs as two agents exchange (provide, receive) resources. To acquire a resource an agent has to give up some other resource. An *agent* is an individual or organization capable of having control over economic resources, and transferring or receiving the control to or from other agents [GLP08].

The constituents of processes are called *economic events*. An economic event is carried out by an agent and affects a resource. The notion of stockflow is used to specify in what way an economic event affects a resource. REA identifies five

stockflows: produce, use, consume, provide and receive, where the first three occur in conversion processes and the latter two in exchange processes. REA recognizes two kinds of duality between events: conversion duality and exchange duality.

Events can be assigned to a *location*. Sometimes the acronym REAL is used for REA plus location [OL99] (Fig. 3).

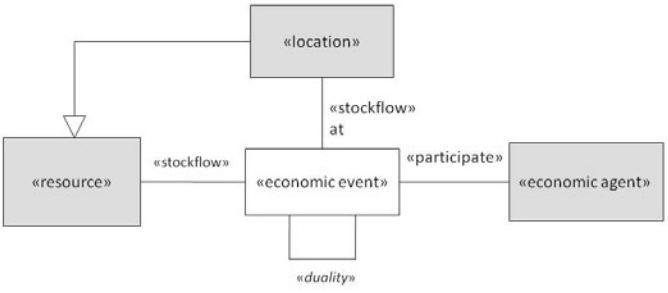

Fig. 3. REA basic categories including location. Events are rendered in white, the other objects in grey.

3.2 Commitments in REA

Commitments were added to the REA ontology in [GM99] as "important economic phenomena", and modeled as the pair-wise connection of requited commitments. The pair-wise connection is similar to the *duality* relationship between actual exchanges or conversions but as it is not between events, REA calls it a *reciprocal* relationship. In the following, we refine and extend the commitment concept of REA by adding explicit commitment events and by rethinking the "reserve" relationship. Starting point is that we consider a commitment as a special type of resource, so that it can be handled in the same way, that is, be manipulated and used in exchange and conversion events using stockflow relationships.

A commitment is a promise regarding the future. Commitments are formalized as clauses in contracts and those commitments are subsequently fulfilled through economic events. A distinction can be made between increment commitments (assets in the agent's perspective) and decrement commitments (liabilities in the agent's perspective) [Hr06]. Figure 4 illustrates this: in an economic event, a provider gives a decrement commitment that is received by a customer in the same event. He has given a promise to e.g. deliver a service in the future. Depending on the commitment type (decrement vs. increment) the relationship to the commitment is characterized as a give or take stockflow. A customer can, in a *decommit* event, take a d-commitment that is received by a provider in the same event. This represents an absolving of a commitment. The provider did commit to deliver something in the future and this promise is now considered void by the customer.

A structure involving increment commitments can be constructed as well (not illustrated here) for the customer's part of the contract, but still from the provider's point of view. In a commit event a provider becomes the receiver of an i-commitment (increment) through a take stockflow. The customer owes the provider. The provider

can, in a decommit event, give the customer an i-commitment back, thereby cancelling the debt. Note that the customer cannot cancel this debt himself, but he can request for it. The exchange reciprocity between commitments reflects an exchange duality between commitment events.

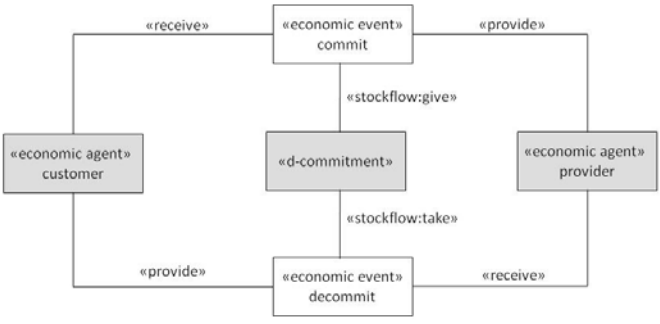

Fig. 4. REA commitment pattern for decrement commitment

Thus contract formation can be thought of as giving and taking corresponding d-commitments and i-commitments (Fig 4).

Commitments can also be returned in a *decommit* event. Two main types of decommit can be distinguished that maintain the duality axioms of REA. In the case of *canceling*, the commitment is returned in exchange with the reciprocal commitment being returned. For instance, a purchase order is cancelled and the payment is cancelled at the same time (of course, the contract may specify a penalty for the one who requests cancelation or even forbid cancelations altogether). In the case of *fulfillment*, the commitment is returned in exchange with some other economic event being provided, being the content of the commitment. For instance, when a delivery is made, the purchase order commitment is returned. In the following, we will adhere to the standard REA fulfill relationship as an abbreviation for this duality.

Committing is modeled as an economic event, rather than as a system event as in standard REA. An economic event is defined as a change in the value of economic resources of the enterprise. We claim that this also applies to taking or giving a commitment, e.g. the commitment to a future payment.

Commitments are most often about resource *types* (e.g. a non-smoking hotel room, or a certain book title to be delivered), whereas the business transaction itself is about a resource instance, that is, a specific hotel room or a specific copy of the book. In some cases, the commitment is about the resource instance itself, e.g. in the sales contract of a house. In order to handle both commitments in a unified way, we apply the notion of resource group [GM06]. Let the object of the commitment be a resource group of a certain resource type. Cardinality of the set/quantity of the resource is the most important attribute of resource group, and additional constraints can be specified. The relationship between resource group and resource type is a *policy* relationship [GeMc06]. It specifies the type of resources that may go into the resource group. Fig. 5 presents our refinement of the REA commitment ontology using the grouping concept. It works as follows. When a commitment is created, the commitment always refers to a resource group that is created at that time. However,

the *grouping* need not be filled in. In the case that a particular resource is to be reserved (e.g. a specific house), the *grouping* relationship is made at commitment time. In all other cases, it is specified later when the purchase contract is being executed. For example, a commitment to reserve two hotel rooms is formalized as a clause in a contract. The object of the commitment is a group. The policy for this group says that it must contain "double non-smoking rooms", and 2 of these. In the economic event that fulfills the commitment two double rooms (resources of the specified type) are allocated to this group.

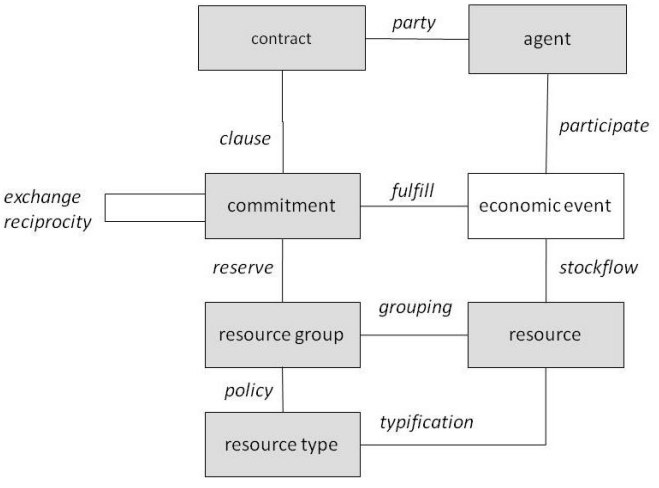

Fig. 5. REA commitment "reserve" revisited

3.3 Capacity Planning

Using the REA model, we can define the notions of *capacity* and *availability*. We take the perspective of the resource manager *a* (e.g. hotel manager) who has received or reserved certain resources from another agent *x* (e.g. hotel owner). He can commit resources of a certain resource type to another agent *x* for a certain date. In that case, there is a specify relationship between the reservation and the resource type. The commitment/reservation has a cardinality indicating the number of resources reserved. The actual allocation of resources (instances) to a certain reservation is usually done later. If we assume the Capacity is stable over time, the following definitions suffice:

$$\text{Capacity}(a,t) = \text{card}(R)$$
$$R = \{r\text{: resource} \mid \text{typification}(r,t) \wedge (\exists x\text{:agent receive}(a,x,r) \vee \exists s\text{:reservation (provide}(x,s) \wedge \text{receive}(a,s) \wedge \text{specify}(s,t) \wedge \text{reserve}(s,r))\}$$
$$\text{Reserved}(a,t,d) = \sum \text{card}(s), \quad s \in RS(a,t,d) \text{ where}$$
$$RS(a,t,d) = \{s\text{: reservation} \mid (\exists x,a\text{:agent provide}(a,s) \wedge \text{receive}(x,s) \wedge \text{specify}(s,t) \wedge \text{date}(s,d)\}$$
$$\text{Available}(a,t,d) = \text{Capacity}(a,t) - \text{Reserved}(a,t,d)$$

The capacity for a resource type *t* is what the agent has received or that is made available to him (and that is of the resource type *t*). To calculate the availability at some date/time *d*, we first sum up the commitments, and detract this number from the capacity.

3.4 Services in REA

In REA, a service is a kind of resource as it is viewed as valuable by some agent and can be transferred between agents [WJAB09]. As such, it inherits all features of resources, in particular that it can be exchanged between agents, that it is governed by a contract and that it is part of a conversion process chain. As depicted in Fig. 6, the service is exchanged between agents in return for money (top right cluster). All the coordination services that can be used within an exchange process apply to service exchanges as well; we will use this feature below. At the same time, the service is a resource produced in a conversion process by the provider (top left cluster), and consumed in a conversion process by the customer (bottom cluster). REA usually renders only one agent perspective, but for the understanding of the service interfacing between the provider and customer, we have included both perspectives (indicated by dotted rectangle) in one figure. Note that Fig. 6 is simplified in order to reduce clutter. In particular, the usual agent boxes are not present in top left or bottom.

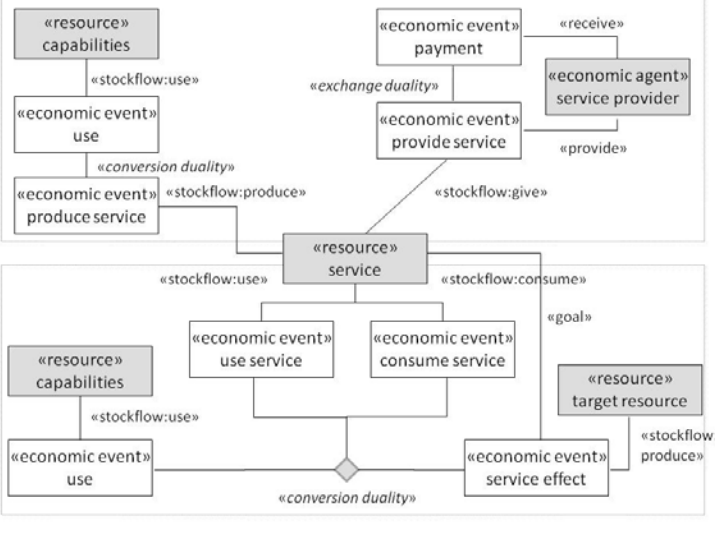

Fig. 6. REA application pattern for Service Exchange. It is assumed that the «goal» stereotype is defined in the REA meta-model

The economic increment event for creating the service stands in conversion duality to one or more resource use events. For example, a hotel service is realized by *using* the hotel room resources. At the customer side, we distinguish between service use and service consumption. Both can add value (production event) to some target

resource, typically in combination of some effort of the customer himself – that is why we also include a resource use event here. However, in the case of service consumption, the service is no longer available after the event, whereas service use draws on the existence of the service without changing its status. The service consumption may be conceived of as an atomic event or as a process over time. The latter is especially the case when the service is offered for a certain period. Then for economic purposes, the amount of service consumption is typically linear on the time having passed by.

The difference between service consumption and service use can be illustrated by the example of [FG09] of a fire brigade. This could be a service hired by a municipality for a certain period. Service consumption is here a matter of time: at the end of the period, it is completely consumed. During the period, the fire brigade may become active in the case of an emergency, as stipulated in the service contract. This is service use. The effect of service use is a particular house (resource) being rescued, whereas the effect of the service consumption is the increased security of all houses in town (resource).

For the user-centric description of a service, the "*goal"* is important [WJAB09]. A service aims to produce an effect on resources of the customer in such a way that the value increases. If the effect is not reached, this may cause the transaction to fail. Formally, the goal relationship can be seen as an extension of the REA meta-model. However, as it can also be seen as a derived relationship, since it is defined as "the production events at the customer' side that stands in conversion duality with the service use and consumption". When also the *consumption* events at the customer and provider side are relevant, we could add a "source" relationship, analogous to the "goal" relationship. Together, source and goal provide a reference to all resources affected by the service execution. As the description of all kinds of failures and exceptions is never exhaustive, we refrain from including that in the effect. It can be specified in the contract.

For web services and similar software artifacts to deserve the label "service", the service model elements should be clear. What is the goal of the web service, that is, what resources does it create or affect that have value to the client? Who are the actors involved in the exchange process? In the next section, we will consider coordination services as one important subclass of web services.

3.5 Coordination Services and Coordination Objects

Coordination services are defined in [WJAB09] as services supporting an exchange process (a set of events) for a good or a service. Processes like identification, negotiation, order execution and after-sales take place in a good exchange as well as a service exchange. We introduce the notion of coordination *object* for the object of these processes: what *is* negotiated and executed? The central coordination object is the purchase order that is first negotiated, then created, and then fulfilled by the exchange event. In complex business processes there are more coordination objects. The following two also reoccur often, especially when services are concerned: *reservation* and *appointment*. The rationale for the latter (appointment) is that the delivery of a service requiring resources from both the provider and customer to be present at the same time and place requires more coordination than the delivery of a good.

The rationale for reservation is the following. Note first that in standard REA, a reservation is a relationship between a commitment and a resource or resource group (section 3.2). In the context of coordination, we use the term "reservation" more specifically for a commitment that precedes the purchase order, which obliges a provider not to sell a resource to any other agent than the customer for whom the reservation is created. From an economic point of view, the main objective of this kind of reservations is to reduce uncertainty about the business transaction – to mitigate the risks involved, such as items being out of stock or functionality not available, and to reduce the need for slack [WH06]. So although the reservation has some costs in the form of less operational discretion, it increases the total value for both customer and provider.

Drawing on the REA ontology, coordination objects can be specified in terms of commitments. Therefore, another way of characterizing coordination services is to say that these services manipulate commitments: their goal is to provide, receive and fulfill commitments.

For a better understanding of the three coordination objects and their commitments, is it good to see how they are related. Although they are applied differently in different domains, there exist logical dependencies between them. The reservation always precedes the purchase order. The conversation around the reservation addresses questions such as: Is the resource available? If so, can I reserve it? How can I cancel the reservation, if needed? Do I have to pay for the reservation or for the cancellation? The seller commits himself to make sure that the necessary resources will be available. Usually, the customer has fewer commitments, but he may commit himself to the availability of necessary resources (in his case, financial resources) as well. This can be achieved in practice by providing credit card details, for instance.

The purchase order follows after the reservation logically and in time. A contract is being formed with terms and conditions. Now the commitments go much further: the seller promises to deliver the good or service at some moment or period in time, and the customer promises to pay an agreed upon price. These are firm commitments that can only be left unfulfilled in exceptional circumstances as specified in general law or in the contract itself.

We assume that for all coordination objects (so not only for the Purchase Order) there is an agreement process first followed by an execution and evaluation process, that is, the coordination process per coordination object takes the form of a "Conversation for Action" [Di04, Go06]. The message exchange in these conversations is not in the scope of this paper, but what is important is the effect of these conversations, since that is directly relevant for a user composing and using a certain mashup application.

The REA application model in Fig. 7 focuses on the coordination object purchase order but also specifies how it relates to the coordination object reservation. The reservation is a commitment that specifies a resource type and there is a "reserve" relationship with resource, being all resources involved in the fulfillment of the commitment and set apart for that purpose. In line with [Hr06] we distinguish between d-commitments (decrement) and i-commitments (increment), for commitments by or to the service provider, respectively. The fulfill relationship is one between commitment and economic event. The fulfillment of the reservation is the accept-order event by which the purchase order is created. The fulfillment of the

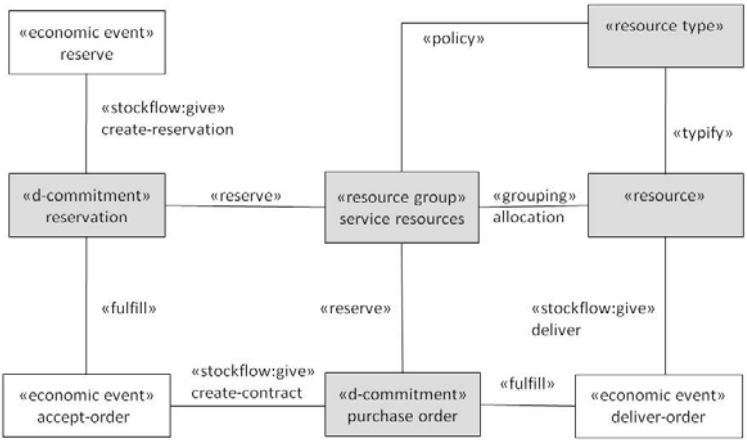

Fig. 7. REA Application Model for purchase order including relationship with reservation

purchase order is the service exchange event. The actual service exchange can be seen as the ultimate objective of the reservation as well. So in certain sense, this exchange fulfills the reservation, although indirectly. To capture this notion of fulfillment, we define a fulfill* relationship being the recursive closure of fulfill-relationships. So the actual exchange *fulfils* the purchase order, and *fulfils** the reservation.

It should be noted that although the meaning of reservation and purchase order is quite stable over different domains, these two coordination objects are not always applied in the same way. This can be confusing, for example when we talk about hotel reservations and flight reservations in one breath. In the case of a hotel, the purchase order is created when the customer checks in. At that moment, the reservation, if any, is fulfilled. In the case of a flight ticket, the purchase order is made when the ticket is sold, typically long before the check-in at the airport. What happens at the check-in is the allocation of a specific resource (a chair with a number). Sometimes, it is possible to take an option on a flight ticket for a few days before buying it. That option is a case of reservation, but the ticket itself is not.

The complete *reservation pattern* is represented in Fig.8. It shows the reciprocity relationships with other commitments that are grouped together in a contract.

An *appointment pattern* is used when two or more agents want to meet at a specific location. Appointments can be made for their own sake, but can also be part of a purchase contract, for example, when customer and provider have to agree on where to deliver the service or good.

Fig. 9 shows an application model for show-up appointments where the commitment is from the side of the customer (so it is an i-commitment), typically reciprocal to an appointment of the other party to be there as well. Since the appointment includes at least a resource (the customer himself, or some resource related to the customer; and the location) there are two “reserve” links. In accordance with our “reserve” ontology, these links point to groups that specify the reservation on an abstract level and that are populated at some time with specific instances.

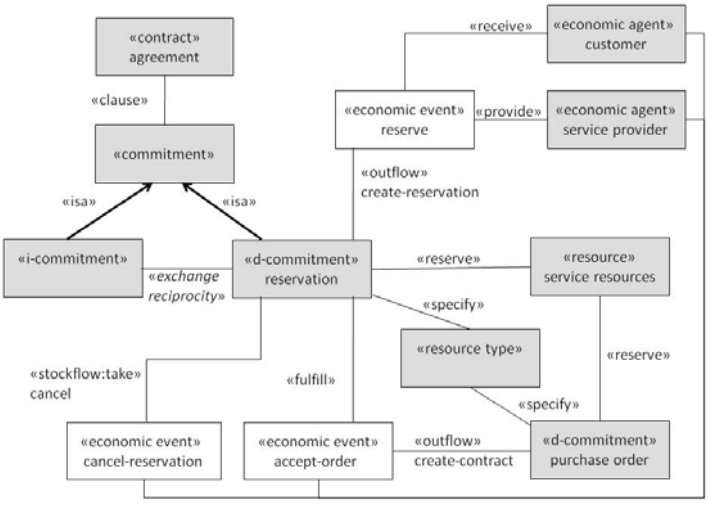

Fig. 8. REA Application Model for reservation

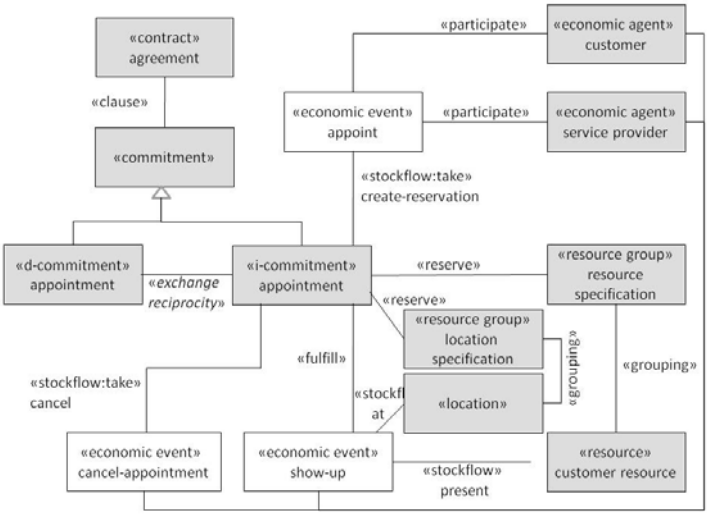

Fig. 9. REA Application Model for (show-up) Appointment

The appointment can be applied throughout the process (for instance, an appointment to negotiate the purchase order via Skype at some specific hour), but most prominently at the fulfillment of the purchase order when this fulfillment is not immediate, as noted above.

We have described three coordination objects corresponding to three coordination services – reservation, purchase contract, and appointment. On the basis of our

experience and considering standards such as [UM03] and the set of patterns identified by [Hr06], we claim that these are the most prominent ones, but the question whether more coordination objects exist should be addressed further by both formal and empirical research.

4 Service Description and Conflict Detection

4.1 Service Description Using REA

Using the REA ontology, service descriptions can be developed for coordination services either in the form of REA events or REA relations. Table 1 specifies the basic predicates.

Table 1. Basic REA predicates

RELATIONS	EVENTS	TERMS
At(Agent,Location)	*Commit(Id,Agent,Agent, e(Resource Type,Time))*	*contract*
Fulfill(Event,Commitment)	*Cancel(Id,Commitment)*	*commitment*
Clause(Commitment, Contract)	*Purchase(Id,Agent,Agent, Resource)*	
Available(Agent, ResourceType,Time): Number	*Pay(Id,Agent,Agent,Money)*	
Capacity(Agent , Resource Type):Number	*Move(Id,Agent,Location)*	
PlannedCapacity(Agent, Resource Type, Time):Number	*Move(Id,Agent,Resource, Location)*	

The relations and terms have a direct counterpart in REA or have been defined in section 2. We use some shorthand for the events. *Commit* stands for create commitment, *Cancel* for withdraw commitment. *Purchase* and *Pay* stand for the standard exchange events. *Move* stands for the event of changing the location of the agent or some resource. In both Commit and commitment we make use of an embedded functor *e(x,t)* where *e* is an Event Type, *x* can be a any object (and there may be more than one argument *x*) and *t* is a time reference. Expressions of this form are called *i-events* and are used in the same way as actions in the situation calculus [MH69], where they can be the object of a *do*-action.

Using these predicates, we define the following list of coordination services (table 2). Note that they are services in terms of [WJAB09]: their goal is an event that affects a relevant resource. Being coordination services, they manipulate commitments. Table 2 presents the IOPEs (Input/Output/Precondition/Effect) for hotel services but in a quite general way. As such it can be applied to a flight service or theater service as well. However, the way the coordination services are bundled in web services may differ. In the typical hotel case, the Create_Contract and Check_In are one transaction: at the moment the customer shows up, according to his reservation, a contract is set up and a specific resource is allocated. In the typical flight case, the Create_Contract is performed long time before the Check_In.

Table 2. Generic coordination services

Coordination Service	Input	Output	Precondition	Effect (Goal)
Check_Availability	ResrcType R Time T User U	Bool A	A= (Available(Self,R,T)>0)	*Not a social fact*
Create_Reservation	Customer C Time T ResrcType R	Id Res	Available(Self,R,T)>0 At(Self,L)	commit(i,Self,C, e(R,T)) and i=Res and commit(j,C,Self, move(C,L,T.start))
Cancel_Reservation	Customer C Time T ResrcType R Id Res	-	commitment(i, Self,C, e(R,T)) and i=Res and not exist p: fulfill(p,i)	cancel(j,i) and forall j: commitment(j,C,Self, move(C,L,T.start)) implies cancel(j)
Create_Contract	Customer C Time T Id Res	Id PO Amount F	commitment(i,Self,C, e(R,T)) and i=Res	commit(j,Self,C,e(Rs,T)) and j=PO and typeof(Rs,R) and exist contract(CT) and clause(PO,CT) and clause(Inv,CT) and commitment(Inv,C,Self, pay(F,T2)) and fulfill(PO.Res)
Check_In	Customer C Time T Id PO	Id Ri	commitment(j,C, Self, move(C,L,T.start) and j= LRes and at(C,L) and commitment(i, Self,C, e(Rs,T)) and i=PO	commit(i,Self,C,e(Ri,T)) and realize(Rs,Ri) and forall m: move(m,C,L) implies fulfill(m,LRes)
Check_Out	Customer C Id Ri	Id S	commitment(i,Self,C, e(Rs,T)) and i=PO and realize(Rs,Ri)	purchase(j,Self,C,Ri,T) and i=S and fulfill(S,PO)
Receive_Payment	Customer C Id PO	Id P	exist contract (CT) and clause(PO,C) and clause(Inv,C) and commitment(Inv,C,Self, pay(F,T2))	pay(j, C,Self, F) and j=P and fulfill(P,Inv)
Cancel_Contract	Customer C Time T Resource Rs Id PO	-	commitment(i, Self,C, e(Rs,T)) and i=PO and exist contract(C) and clause(PO,C)	cancel(j,i) and forall j: commitment(j,C,Self, Q,T') implies cancel(j)

4.2 Conflict Detection

As said in section 1, each resource or agent is assumed to have a time-based context A(E,C) where E is a set of events planned and C a set of constraints on E. To support conflict detection and conflict prevention, we should be able to check whether E meets the constraints C. This applies both to the service provider and the resources that he manages and to the service user and the resources that she manages. In the following, we take the perspective of the user and focus in particular on her agenda.

Let M be the set of resources relevant to agent *U*. To determine the contents of M, the set of E_u of committed events for *U* is calculated first as follows:

$E_u = \{e: \text{event} \mid \exists c: \text{commitment} \wedge \text{fulfill}^*(e,c) \wedge \text{participate}(e,U)\}$

Then

$M = \{m \mid \exists e \in E_u: \text{stockflow}(e,m) \vee \text{participate}(e,m)\}$ (resources involved and agents participating, as far as known)

For some $m \in M$, the context E_m contains the committed events that involve m. Note that $U \in M$ – by definition, the user herself is also involved and hence the commitment should be put in her agenda. However, not only the context of U, but the context of every $m \in M$ should not violate its constraints. The constraints in the context can be resource-specific, but a very fundamental constraint is that there can be no "agenda conflict":

$$\forall e_1, e_2 \in E_m \; e_1.\text{time} \cap e_2.\text{time} = \varnothing$$

Another general constraint is that a physical resource can be at only one place at a time, and needs time for moving.

$$\forall e_1, e_2 \in E_m : e_1.\text{time.end} = e_2.\text{time.start} \Rightarrow e_1.\text{location} = e_2.\text{location}$$
$$\forall e_1, e_2 \in E_m : \text{next}(e_1, e_2) \wedge e_1.\text{location} <> e_2.\text{location}$$
$$\Rightarrow (\exists e_i \in E_m : e_1 < e_i < e_2 \wedge e_i.\text{type} = \text{move}() \wedge e_1.\text{object} = m \wedge e_i.\text{destination} = e_2.\text{location})$$

where $\text{next}(e_1, e_2)$ means that e_2 is the first event after e_1.

To prevent conflicts when considering the use of a service s, the user first adds the commitments produced by s to his context (using the coordination service effect descriptions), and then executes the conflict detection process. If a conflict is detected, the user is informed and/or the coordination service in question is triggered to find an alternative.

5 Concluding Remarks

From an end-user perspective, coordination services form an important service class (cf. [Be04]), as these services allow the user to manage real services that matter to him/her. When using these services, he should be aware of the real-world effects, to detect and prevent possible conflicts with his own agenda (already existing commitments) or the agenda of other resources involved. In this paper, we have explored how REA can be used to describe the effect (IOPE) of a representative set of coordination services.

The focus of this paper has been on the description of coordination services which are the services that support an exchange process (a set of events) of a resource. Creating, executing and evaluating commitments is done in a combination of informational and material processes. The Language/Action Perspective ([Di06,Go06]) has explored a couple of standard micro-patterns on which the informational processes can be based. However, we have not focused on describing processes in this paper.

One line of future research concerns the interpretation of commitments in terms of rights. When an agent commits (d-commitment), he gives away some right on the resources involved, which assumes that he did hold that right before. REA posits a "control" relationship between agents and resources. If control is made more precise in terms of rights (e.g. ownership, access and custody), this can be used in the specification of commitment preconditions and effects.

References

[AC06] Anderson, C.: The Long Tail: How endless choice is creating unlimited demand. Random House Business Book, London (2006)

[Be04] Benatallah, B., Casati, F., Toumani, F.: Web Service Conversation Modeling: A Cornerstone for E-Business Automation. IEEE Internet Computing 8, 1 (2004)

[Di06] Dietz, J.: Enterprise Ontology - Theory and Methodology. Springer, Berlin (2006)

[FG09] Ferrario, R., Guarino, N., Fernandeze Barrera, M.: Towards an Ontological Foundation for Service Science: the Legal Perspective (2009)

[GLP08] Gailly, F., Laurier, W., Poels, G.: Positioning and Formalizing the REA enterprise ontology. Journal of Information Systems 22, 219–248 (2008)

[GM99] Geerts, G., McCarthy, W.E.: An Accounting Object Infrastructure For Knowledge-Based Enterprise Models. IEEE Int. Systems & Their Applications, 89–94 (1999)

[GeMc06] Geerts, G., McCarthy, W.: Policy-Level Specifications in REA Enterprise Information Systems. Journal of Information Systems 20(2), 37–63 (2006)

[Go06] Goldkuhl, G.: Action and media in interorganizational interaction. ACM Comm. 49(5), 53–57 (2006)

[HS09] Hoyer, V., Stanoevska-Slabeva, K.: Towards a reference model for grassroots enterprise mashup environments. In: Proc. ECIS 2009 (2009)

[Hr06] Hruby, P.: Model-Driven Design of Software Applications with Business Patterns. Springer, Heidelberg (2006)

[Mc82] McCarthy, W.E.: The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. The Accounting Review (1982)

[MH69] McCarthy, J., Hayes, P.J.: Some philosophical problems from the standpoint of artificial intelligence. Machine Intelligence 4, 463–502 (1969)

[OA09] OASIS Reference Architecture Foundation for Service Oriented Architecture 1.0 (2009), `http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-cd-02.pdf`

[OL99] O'Leary, D.: REAL-D: A Schema for Data Warehouses. Journal of Information Systems 13(1), 49–62 (1999)

[PT08] Peffers, K., Tuunanen, T., Rothenberger, M., Chatterjee, S.: A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems 24(3), 45–77 (2008)

[PT09] Pistore, M., Traverso, P., Paolucci, M., Wagner, M.: From Software Services to a Future Internet of Services. In: Tselentis, G., et al. (eds.) Towards the Future Internet. IOS Press, Amsterdam (2009)

[UM03] UN/CEFACT Modelling Methodology (UMM) User Guide (2003), `http://www.unece.org/cefact/umm/UMM_userguide_220606.pdf`

[WH06] Weigand, H., van den Heuvel, W.J.A.M.: A conceptual architecture for pragmatic web services. In: Schoop, M., de Moor, A., Dietz, J. (eds.) Proc. 1st Int. Conf. on the Pragmatic Web, pp. 53–66. Springer, Heidelberg (2006)

[WJAB09] Weigand, H., Johannesson, P., Andersson, B.: Bergholtz Value-based Service Modeling and Design: Toward a Unified View of Services. In: van Eck, P., Gordijn, J., Wieringa, R. (eds.) CAiSE 2009. LNCS, vol. 5565, pp. 410–424. Springer, Heidelberg (2009)

Author Index