Preface

The CAiSE conference has been a leading conference for the information systems engineering community for more than two decades, emphasizing different themes in each year. The special theme of CAiSE 2010 is "Evolving information systems", and it is devoted to the continuous evolution process of information systems and the methodologies and technologies supporting this process.

The leading position of the CAiSE conference causes a fierce competition among papers, accompanied by a low acceptance rate. This poses an obstacle that hinders the acceptance of papers devoted to innovative research projects, whose results are interesting to the community, but not yet final and mature enough to be acceptable to the conference. As it is beneficial to the CAiSE community to be exposed to these ideas and to discuss and communicate about them even at this phase, the CAiSE Forum was created

CAiSE Forum is a special event within the CAiSE conference. The Forum is intended to serve as an interactive platform for presenting and discussing new ideas related to information systems engineering. As such, it aims at the presentation of fresh ideas, new concepts, as well as demonstration of new and innovative systems, tools and applications.

The Forum taking place at CAiSE 2010 is the eighth Forum event. Over the years different formats have been used for this event, seeking to increase the interactive nature and to better facilitate communication and exchange of ideas among its participants.

The 24 high quality papers accepted for presentation in CAiSE Forum 2010 can roughly be categorized into two kinds of papers. One includes papers reporting innovative research projects, which were selected based on the novelty of the ideas they present. The second kind is papers devoted to innovative tools and prototypes that implement the results of research efforts. These tools and prototypes are presented as demos in the Forum event.

Forum co-chairs:
Erik Proper
Pnina Soffer

Exploiting Tag Clouds for Database Browsing and Querying

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 show how tag clouds can be used alongside more traditional query languages and data visualisation techniques as a means for browsing and querying databases. Our approach is based on a general, extensible framework that supports different modes of visualisation as well as different database systems. A number of demonstrator databases and interfaces will be used to show how tag clouds can be used to visualise and browse data or metadata and even a mix of both in object databases and relational databases. Further, we will demonstrate synchronised browsing based on tag clouds as well as ways in which tag clouds can be combined with other forms of querying and data visualisation.
Keywords: tag cloud, data visualisation, database interface

1 Introduction

Tag clouds are widely used in Web 2.0 applications for visualising user-generated tags and folksonomies of specific web sites such as Flickr[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 or some form of relationship between tags.

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 [1]. The use of a query language requires the user to master not only the query language but also the 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. Another approach is to focus on the data rather than the schema as supported in keyword search interfaces to databases. Tag clouds have been proposed as a means of summarising and refining the results of keyword searches as presented in [2, 3]. In this case, the term *data cloud* is used to refer to their particular adaptation of tag clouds for this purpose. An interesting

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

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 [4] 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.

Our goal was to investigate the extent to which tag clouds could be exploited to support more traditional forms of database browsing and querying, either 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 them. 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 and particularly liked the fact that it could be combined with query expressions.

Our contributions include:

- A data browser that allows any data source to be browsed and queried using tag clouds.
- Experimentation with text and position features of tags in a tag cloud to make clouds more informative.
- A tool that serves different purposes: Novice users are able to access structured data sources without knowing the query language and schema, while expert users can browse a data source in order to get to know the schema and thus be enable to express complex queries over the data source.
- An extensible and flexible platform for experimentation where new data sources and new visualisation techniques can be added.

In the following sections, we provide an overview of the data browser, the architecture and also the demonstration.

2 Data Browser

As highlighted in [5], tag clouds serve multiple purposes. They can be used for searching for specific information, browsing a data collection without a specific target, as a tool for impression formation and gisting, and to recognise what a data collection is about. In the Web, tags of a tag cloud are usually hyperlinks that lead to a collection of items that are associated with a 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 [5–7] have experimented with tag cloud features and positioning and their impact on users. According to [5, 6], 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 [7].

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. We note that concepts similar to those proposed for data clouds in [2, 3] could be adopted to return entities rather than tuples for specific applications. Similarly, it is possible to mix different attribute values in a single tag cloud or to form tag clouds from combined attribute values. In addition, we use these concepts to also browse metadata and have even experimented with a mix of metadata and data within tag clouds.

We now explain these concepts further by means of an example based on a database with information about contacts and their locations. Generally, 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.

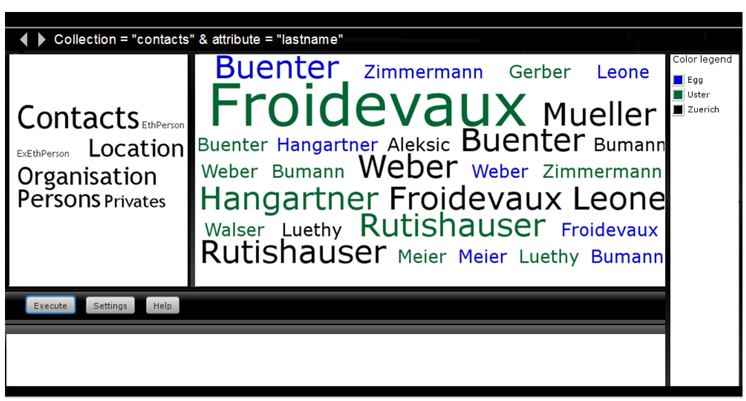


Fig. 1. Schema and Data Browsing

The metadata that defines the schema of a database can itself be represented by a tag cloud as shown in Figure 1. On the lefthand side of this figure, the tag cloud gives 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.

A user can start browsing a database either by entering a query expression in the window below the tag clouds or by selecting one or more of the tags in the schema tag cloud. Each collection can have a default attribute or set of attributes specified for its visualisation as a tag cloud. However, the user can also specify this by means of a simple selection of attributes through checkboxes. Alternatively, one can display the attributes themselves as a tag cloud in the lefthand window 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.

In the example of Figure 1, the attribute `lastname` is displayed in the tag cloud on the right as indicated by the navigation path shown on top of the cloud window. The size of the tags in this tag cloud represents how many data items have that attribute value. In this way, the tag cloud can be considered as a visualistaion 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.

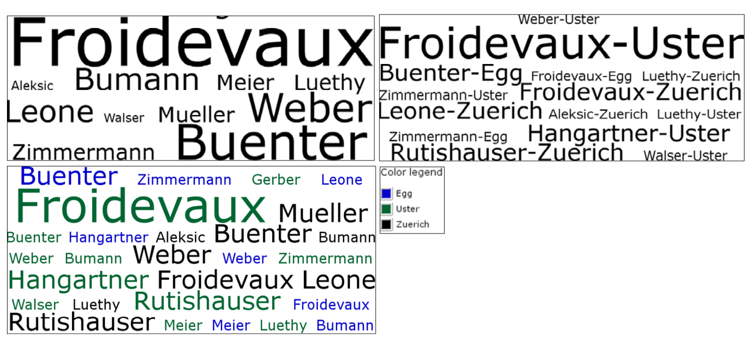


Fig. 2. Exploiting tag cloud features

We offer different modes for visualisation, as depicted in Figure 2. In the tag cloud in the upper-left corner, the contacts are displayed by lastnames. In the upper-right corner, two attributes are bound to the tag content feature, 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 lastname 'Froidvaux' who live in 'Zurich'. As one can see in this example, more people with the name 'Froidvaux' live in 'Uster', than in 'Zurich'. In the lower-left corner of Figure 2, 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 one can see from the index on the righthand side of the figure, each distinct attribute value of the city attribute is assigned a specific colour. We have experimented with these different tag features in a user study. Care has to be taken in choosing the right attributes to bind to the colour feature. It only makes sense, if the distinct set of values is not too large, since otherwise the index becomes very large and the tag colours are not very informative.

3 Architecture

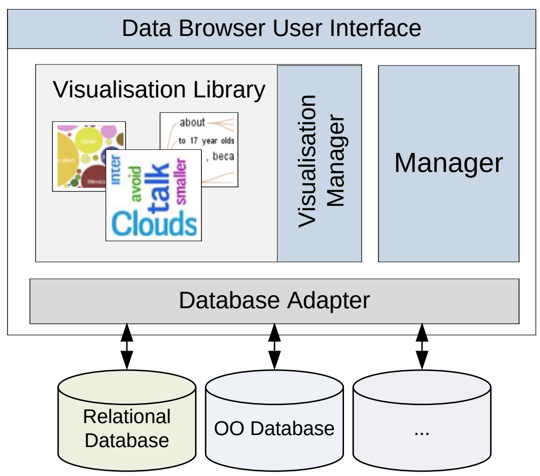


Fig. 3. System Architecture

Figure 3 gives an overview of the system architecture. The manager component is the heart of the system and responsible for handling requests from the user interface, 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. 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[2], OMS Avon[3] and OMSPro[4] as well as a MySQL implementation. Secondly, the visualisation manager can manage different kinds of visualisation techniques. Therefore, we provide a visualisation

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

interface which has to be implemented to add a new technique to the visualisation library. We currently provide a tag cloud visualisation, and are 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.

4 Demonstration

In our demonstration, we will show how users can browse both relational and object databases using our data browser. The demonstration will include showing tag clouds over data, metadata and a mix of data and metadata. We will provide a set of demonstrator databases including a contacts database and a publications database implemented using both relational and object databases. Visitors will be able to freely browse these databases, pose queries and exploratively get an impression of the schema and the data. We will also provide a list of query tasks from our user study so that users can experience how query results can be obtained using only the bowser, using only query expressions and using the browser in conjunction with query expressions.

References

1. Jagadish, H.V., Chapman, A., Elkiss, A., Jayapandian, M., Li, Y., Nandi, A., Yu, C.: Making Database Systems Usable. In: Proc. ACM SIGMOD'07. (2007)
2. Koutrika, G., Zadeh, Z.M., Garcia-Molina, H.: Data Clouds: Summarizing Keyword Search Results over Structured Data. In: Proc. EDBT'09. (2009)
3. Koutrika, G., Zadeh, Z.M., Garcia-Molina, H.: CourseCloud: Summarizing and Refining Keyword Searches over Structured Data. In: Demo Proc. EDBT'09. (2009)
4. Kuo, B.Y.L., Hentrich, T., Good, B.M.., Wilkinson, M.D.: Tag Clouds for Summarizing Web Search Results. In: Proc. WWW'07. (2007)
5. 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. CHI '07. (2007)
6. Bateman, S., Gutwin, C., Nacenta, M.: Seeing Things in the Clouds: The Effect of Visual Features on Tag Cloud Selections. In: Proc. 19th ACM Conf. on Hypertext and Hypermedia. (2008)
7. Schrammel, J., Leitner, M., Tscheligi, M.: Semantically Structured Tag Clouds: An Empirical Evaluation of Clustered Presentation Approaches. In: Proc. CHI'09. (2009)

Facing the Challenges of Genome Information Systems: a Variation Analysis Prototype.

Ana M. Martinez, Ainoha Martín, Maria José Villanueva, Francisco Valverde, Ana M. Levín, 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

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 must be performed manually and several tools are used. The application of Information Systems (IS) principles into the development of bioinformatic 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

Thanks to the breakthrough of the Human Genome Project and the advances in DNA sequencing, an enormous 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 phenotype (expression of the genes in a specific human feature). As a consequence, the creation of biological databases and tools to exploit the produced data have grown drastically. However, these tools and databases have usually been defined to support an specific research area or experiment. Therefore, when biologists want to use them for a particular assay, it is very unlikely that they support their specific requirements. This issue leads to a situation where the researcher 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 specification of the tooling workflow and integration is far from trivial.
- Inclusion of new studies and bibliography into the available tools turns into a hard task.

With the goal of facing these issues, some researchers have proposed [1] the development of Genomic information Systems (GIS), an IS specifically designed 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 organize genomic data and guide the development. Thanks to the close collaboration with biologists in the context of this research project, the gap between the disciplines of Software Engineering and Genetics is solved. The result of this interdisciplinary collaboration is 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 a DNA sequence in order to find documented variations for a specific gene. Once all variations are located in the sequence, the prototype splits them in 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 disease detection might be determinant. The main contribution of this work is that the GIS development is supported by a set of conceptual model entities that formalize the domain concepts related with genomic variations. As a consequence, the conceptual model plays an integration role to provide the genomic knowledge in an unambiguous way and independent from specific datasource details.

With that goal in mind, 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 for describing genomic variations. Section 4 describes how the variation analysis prototype has been developed. Finally, in section 5 conclusions and future work are stated.

2 Related Work

In recent years, several commercial tools have been developed to provide genomic analysis. These tools can perform tests in order to estimate the customer probability to suffer certain diseases. Navigenics [2], 23andMe [3] and deCODEme [4] are the most relevant tools in this field. The differences between them are briefly summarized in Table 1. However, the accuracy of these tools is far from ideal.

Results are not reported in an unambiguous way because biological concepts are not precisely defined. Without a conceptual model that guides the precise definition of the domain, further integration with external tools is complex to achieve.

Another drawback of these tools is that the only variations reported are SNPs (Single Nucleotide Polymorphism). The conceptual model improves the reports quality because other complex variations such as repetitive insertions or deletions are classified. Furthermore the diseases detected by these commercial tools are constrained to the number of supported genes. The use of a conceptual model overcomes this constraint because provides guidelines to support several gene sequence references and their new discovered variations.

Table 1. Comparison of DNA analysis tools.

	Navigenics	23andMe	deCODEme
Analysis Type	Genotyping	Genotyping	Genotyping
Platform	Affymetrix [5]	Illumina [6]	Illumina [6]
Variations (million)	1 (only SNP)	0.5 (only SNP)	1.2 (only SNP)
Detected diseases	28	51	49

3 A Conceptual Model for Describing Variations

The main objective of the conceptual model presented in this paper is to establish a connection point between the genomic field and the GIS development domain. One of the main characteristics of the genomic field is heterogeneity. The unification of the relevant concepts is a difficult task, since genomic concepts are not precisely defined. Moreover, the field knowledge is still developing and these concepts are constantly evolving, making the organization of all the genetic data available more difficult.

Genetic databases are thus affected by this heterogeneity problem. Each database reflects 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 [7]. Even though all of them represent more or less the same concept, there are slight differences among them. The problem of heterogeneous data can be solved with the use of conceptual models, as some works have proposed [8]. 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 into 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 [9], whose main goal is the specification of the required human genome concepts for developing GIS.

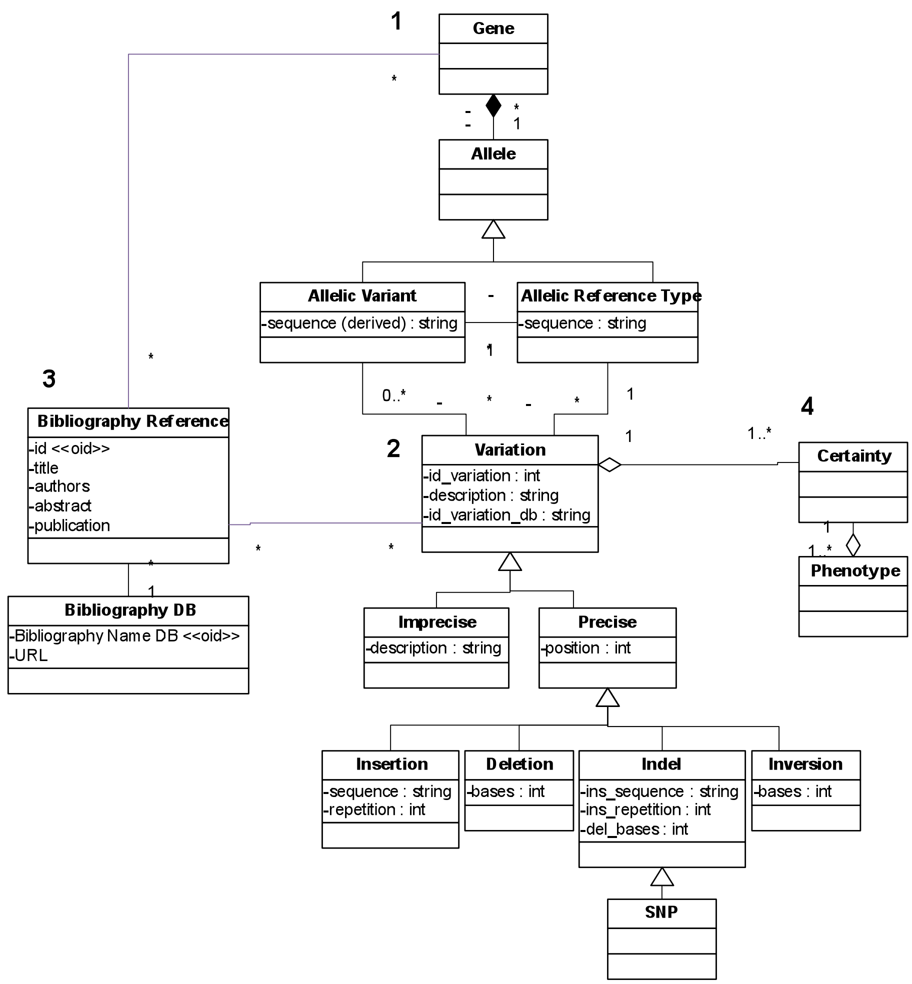


Fig. 1. Conceptual Model for describing variations

Figure 1 shows the proposed conceptual model. At the top of the picture (1) *Gene* and *Allele* modeling entities are defined. *Gene* entity models the generic concept of gene whereas *Allele* entitiy 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 definesa "universal" gene to be used for comparison purposes. These reference sequences are extracted for trusted data sources as RefSeqGene database [10]. *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* entitiy 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 specializations: *Precise* variations, which define a variation that is 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.

A variation that is specified in the model is always related to its phenotype, which is modeled by the *Phenotype* entity (3). The *Certainty* entity specifies the probability that a phenotype could show up because of a concrete variation on the genotype. In case is identified a genotype-phenotype association, 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* and *BibliographyDB* entity (4) 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 Prototype

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 the genomic sequences [11]. Researchers perform the analysis by doing a comparison between a certain DNA sample from a concrete 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. Then, they have to dive into the vast and non-structured amount of information that is scattered across the Web and search 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 is 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.

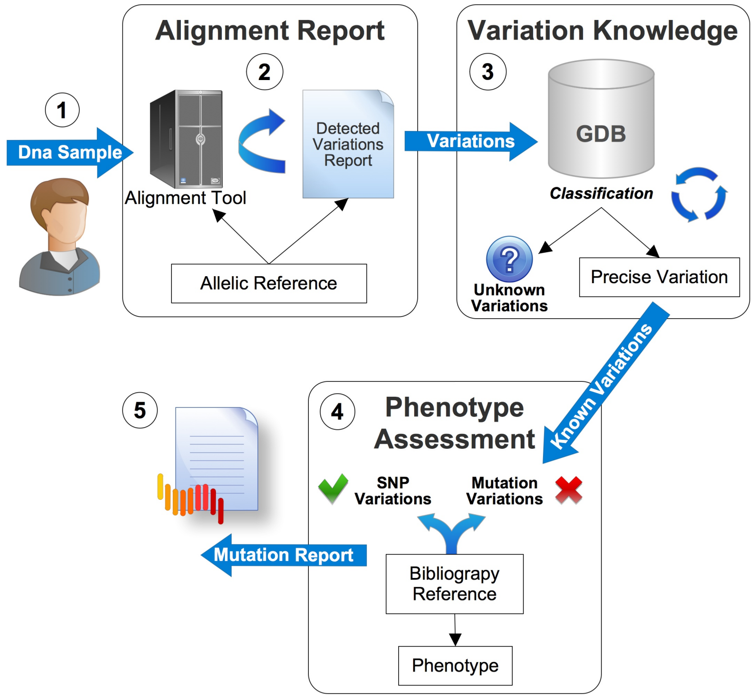


Fig. 2. Prototype Phases

The purpose of 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 input of the sample 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 [12], 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 formal-

ization, which it 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 match with the genomic 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 an 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 an specific disease, a research publication is required to provide a trustful evidence. For those cases, the conceptual model describes the bibliographical reference that supports the phenotype for an specific variation. In the context of this work, variations with a 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.

5 Conclusions and Future Work

This work proposes a GIS engineering solution in order to solve the problems of heterogeneity 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, which uses this conceptual model as background, has been implemented.

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 can be precisely linked to the bibliography.

However, the study of the prototype performance working with real DNA samples must be analyzed. In order to fulfill this task, further studies related with sequencing algorithms and tools will be carried out.

Conceptual modeling of genes is not a completely novel research area. Some works [14] [15] [16] to organize the genomic data have also been proposed before. The main contribution of the presented work is that the conceptual model proposed here is specifically designed to guide the implementation of software artifacts using a model-driven development approach.

As further work it is planned to extend the GIS prototype with the aim of achieving a higher accuracy and to facilitate the input of sequences. 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.

Acknowledgments

This research work has been developed with the Generalitat Valenciana support under the project ORCA (PROMETEO/2009/015)

References

1. Gilbert, D.G.: Eugenes: a eukaryote genome information system. Nucleic Acids Research **30** (2002) 145–148
2. Navigenics. `http://www.navigenics.com` (2010)
3. 23andMe. `https://www.23andme.com` (2010)
4. deCODEme. `http://www.decodeme.com` (2010)
5. 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** (2003) e15
6. Klein, R.: Power analysis for genome-wide association studies. BMC Genetics **8** (2007) 58
7. den Dunnen, J.T., Antonarakis, E.: Nomenclature for the description of human sequence variations. Human Genetics **109** (2001) 121–124
8. Richesson, R., Turley, J.P.: Conceptual models: Definitions, construction, and applications in public health surveillance. Journal of Urban Health **80** (2003) i128
9. Pastor, O., Levin, A.M., Casamayor, J.C., Celma, M., Villanueva, M.J., Eraso, L.E., Alonso, M.P. Enforcing conceptual modeling to improve the understanding of human genome. Research Challenges in Information Science (RCIS 2010)
10. NCBI: The RefSeqGene project. `http://www.ncbi.nlm.nih.gov/RefSeq/RSG` (2010)
11. Stevens, R., Goble, C., Baker, P., Brass, A.: A classification of tasks in bioinformatics. Bioinformatics **17** (2001) 180–188
12. Altschul, S., Gish, W., Miller, W., Myers, E.W., Lipman, D.: Basic local alignment search tool. Journal of Molecular Biology **215** (1990) 403–410
13. Gene Codes Corporation.: Sequencher. `http://www.genecodes.com` (2010)
14. Consortium, T.G.O.: Gene ontology: tool for the unification of biology. Nature genetics **25** (2000) 25–29
15. 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** (2000) 548–557
16. Ram, S.: Toward Semantic Interoperability of Heterogeneous Biological Data Sources. In: Advanced Information Systems Engineering. Springer Berlin / Heidelberg (2005) 32

Alaska Simulator Toolset for Conducting Controlled Experiments
– Tool Paper –

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 is an interactive software suite developed at the University of Innsbruck which allows to explore different approaches to process flexibility by using a familiar metaphor, i.e., travel planning and execution. In particular, Alaska Simulator Toolset is used for studying research questions in the context of business process management and other related fields. For this, Alaska Simulator Toolset provides integrated support of different approaches to process flexibility fostering their systematic comparison. Moreover, Alaska Simulator Toolset facilitates the design and execution of controlled experiments through experimental workflow support.

Providing effective IT support for business processes has become an essential activity of enterprises in order to stay competitive in today's market [1]. When assessing the usability of BPM approaches, however, enterprises have to rely on vendor promises or qualitative data rather than on empirical or experimental research [2]. This is rather surprising as these research methods have been successfully applied in similar research areas like software engineering (e.g., [3, 4]). Alaska Simulator Toolset (AST) has been developed to address this need and allows the investigation of strengths and weaknesses of different approaches for process flexibility through the execution of controlled experiments. Due to the many similarities between business processes modeling and execution and journey planning, AST uses a journey as a metaphor[1]. Furthermore, the used metaphor provides an attractive context to be engaged in, thus increasing the willingness of subjects to participate in experiments. AST consists of three major components: *Alaska Configurator*, *Alaska Simulator* and *Alaska Analyzer*. In the following we describe the main functionalities of AST and how design and execution of controlled experiments is supported using the experiment described in [5] as example (cf. Fig. 1 for the experimental design).

[1] For a detailed description of the journey metaphor visit the simulator's website: http://www.alaskasimulator.org

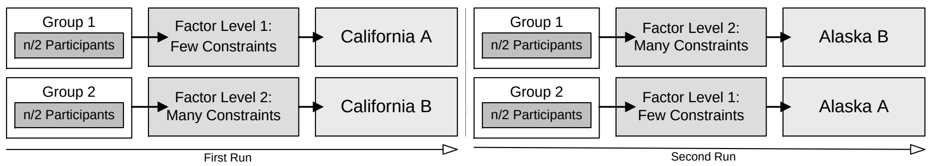


Fig. 1. Experimental Design

Alaska Configurator was used to design two journey configurations (California and Alaska) including locations, actions, events, constraints as well as variability of weather conditions. 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. The journey configurations and the survey were then assembled to an experimental workflow.

During experiment execution participants were guided by the experimental workflow. After presenting them with a survey, half of the students obtained configuration California with few constraints, while the second 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.

After the planning session researchers were supported in analyzing the journeys by enabling them to replay journeys step by step using Alaska Analyzer.

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 a controlled experiments which was conducted using Alaska Simulator we refer to [5].

Acknowledgements: We thank Felix Schöpf for his work on the experimental workflow support.

References

1. Antonucci, Y.L.: Using Workflow Technologies to Omprove Organizational Competitiveness. Int'l. J. of Management **14** (1997) 117–126
2. 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** (2010) 292–310
3. Myers, G.J.: A Controlled Experiment in Program Testing and Code Walkthroughs/Inspections. Commun. ACM **21** (1978) 760–768
4. Lott, C.M., Rombach, H.D.: Repeatable Software Engineering Experiments for Comparing Defect-Detection Techniques. Empirical Software Engineering **1** (1996) 241–277
5. Weber, B., Reijers, H., Zugal, S., Wild, W.: The Declarative Approach to Business Process Execution: An Empirical Test. In: Proc. CAiSE'09. (2009) 470–485

Making Business Process Implementations Flexible and Robust: Error Handling in the AristaFlow BPM Suite

Andreas Lanz, Manfred Reichert, and Peter Dadam

Institute of Databases and Information Systems, University of Ulm, Germany

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 realized process-aware information system. Meeting this goal becomes extremely complicated if high flexibility demands need to be fulfilled. This software demonstration shows how the AristaFlow BPM Suite assists process participants in coping with errors and exceptional situations. In particular, we focus on new error handling procedures and capabilities using the flexibility provided by ad-hoc changes not shown in other context so far.

1 Introduction

During the last decade we developed the ADEPT2 next generation process management technology [1]. Due to the high interest of companies, ADEPT2 has since 2008 been transformed into an industrial-strength process management system called *AristaFlow BPM Suite* [2]. One of our basic goals is to enable robust and flexible *process-aware information systems* (PAIS) in the large scale. In particular, we want to ensure error-safe and robust process execution even at the presence of exceptions or dynamic process changes. In 2009, AristaFlow BPM Suite was applied to a variety of challenging applications in domains like healthcare, disaster management, logistics, and software engineering.

Our tool demo complements previous demos of ADEPT2 and focuses on a fundamental pillar of robust process implementations: *error handling*. One important aspect in this context is *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. This was probably the most influential challenge for our research activities. It also had significant impact on the development of the AristaFlow BPM Suite. In particular we try to detect as many potential errors as possible during buildtime (e.g. incomplete data flow specifications or deadlocks) to prevent them from happening during runtime. As we will show, however, errors cannot always be prevented. Therefore another important aspect of PAIS robustness concerns is *exception handling*. Our demo will show that the AristaFlow BPM Suite provides an easy but yet powerful way to handle exceptions during runtime. In this context ad-hoc process changes are extremely helpful. By utilizing them it becomes possible to even cope with severe process failures and to continue and complete respective processes.

The paper is structured as follows: In Section 2 we introduce a simple application scenario which we use as running example. Section 3 introduces the AristaFlow BPM Suite. In Section 4 we demonstrate how AristaFlow copes with errors that are encountered during process life and how to do this in a flexible way. Section 5 discusses related work and Section 6 concludes with a summary and outlook.

2 Application Scenario

We will use a simple example to demonstrate how errors can be handled in the AristaFlow BPM Suite. Consider Fig. 1, which 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 company using e-mail. After he has received the pricing offers from both suppliers the bookseller checks whether 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.

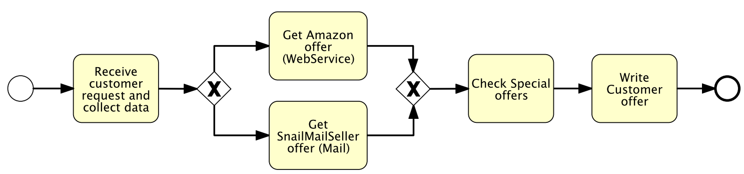


Fig. 1. Scenario: A simple process calling a web service (in BPMN notation)

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 the final activity might fail or produce an invalid output since its input parameters are not provided as expected. Another source of errors might be the Amazon web service; e.g. it might be not available when making the request and therefore the *Get Amazon offer* activity might fail during runtime. Respective errors can be foreseen and hence considered at buildtime. However non-expected errors might occur as well; e.g., activity *Check Special offers* might fail due to troubles with the user's Internet connection.

In summary the following requirements for error-safe and robust process execution exist: On the one hand errors should be avoided during buildtime, on the other hand PAIS must enable users to effectively deal with expected and unexpected errors during runtime. In the following we show how AristaFlow BPM Suite meets these requirements.

3 AristaFlow BPM Suite

As aforementioned AristaFlow BPM Suite[1] is based on the results of the ADEPT2 project. In this research project we targeted at a process management technology which enables ease of use for process implementers, application developers and end users. Furthermore, robustness of process implementations, and support of dynamic process changes were fundamental project goals. 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. 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. These challenges were probably the most influential one for the whole ADEPT2 project [1, 2].

4 Demonstration of the Application Scenario

In the following we consider the scenario from Section 2 from the perspectives of the process implementer, the system, the end user, and the system supervisor. We demonstrate how each of these parties can contribute to the handling of errors.

Process implementer perspective: Fig. 2 shows a part of the process from Fig. 1 as it can be modeled using the *AristaFlow Process Template Editor*. For implementing workflows, we pursue the idea of process composition in a *"plug & play"* style supported by correctness checks. The latter contributes to exclude certain errors during process execution. As prerequisite, for example, implicit data flow dependencies among application services have to be made explicit to the process engine. AristaFlow provides an intuitive graphical editor and composition tool to process implementers (cf. Fig. 2), and it applies a *correctness by construction* principle by providing at any time only those operations to the user which allow to transform a structurally sound process schema into another one; i.e., change operations are enabled or disabled according to which region in the process graph is marked for applying an operation. Deficiencies not prohibited by this approach (e.g., concerning data flow) are checked on-the-fly and are reported continuously in the problem window of the *Process Template Editor*. An example can be seen in Fig. 2, where AristaFlow detects that data element *Customer price per unit* is read by activity *Write Customer offer* but not written by any preceding activity.

Generally, we should not require from process implementers that they have detailed knowledge about the internals of the application functions they can assign to process activities. However, this should not be achieved by undermining

[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.

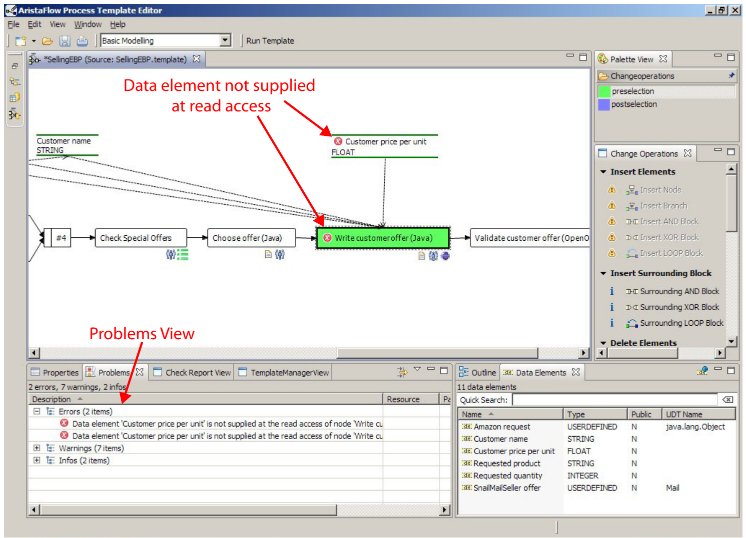


Fig. 2. AristaFlow Process Template Editor

the *correctness by construction* principle. In AristaFlow, all kinds of executables that may be associated with process activities are first registered in the *Activity Repository* as activity templates. An activity template provides all information to the *Process Template Editor*, e.g., about mandatory and optional input/output parameters or data dependencies to other activity templates. The process implementer just drags and drops an activity template from the *Activity Repository Browser* window of the *Process Template Editor* onto the desired location in the process graph.

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 guarantees that process models without any detected deficiencies are sound and complete with respect to the activity templates used.

System perspective: The approach described above ensures that in principle the process model is 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 involved by activity *Get Amazon offer* might not be available when the process is executed, leading to an exception during runtime. Such errors can neither be detected in advance nor completely be prevented by the system.

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 the workflow paradigm, AristaFlow uses processes to handle exceptions, i.e., AristaFlow provides a reflective approach in which error handling is accomplished using a workflow being executed by AristaFlow. A simple error handling process is shown in Fig. 3. 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 silently terminates. Generally, error handling processes can be arbitrarily complex, long running processes. It is important to note that AristaFlow treats error handling processes the same way as any other process. Thus they can contain any activity available in the repository. Note that this enables error handling of higher semantical level, involving users where required. If an activity fails, the respective error handling process is initiated and provided with all the data necessary to identify and handle the error, e.g. the ID of the failed activity instance, the agents responsible for the activity and the process, the cause of the error, etc. (cf. Fig. 3).

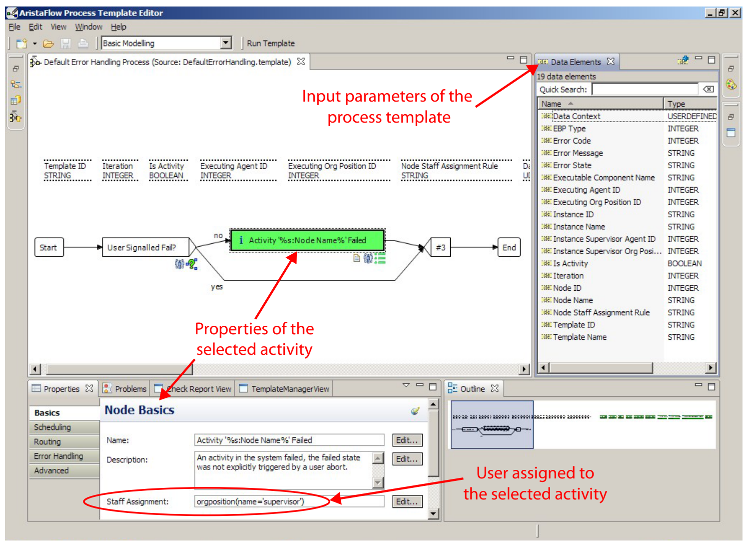


Fig. 3. A simple error handling process

After an error handling process has been created and deployed to the AristaFlow Server it can be assigned to an activity or process by simply selecting it from a list of processes. Whether or not a process is suitable as error handling process is decided based on its signature, i.e. the input and output parameters of the process.

It is also possible to assign an error handling process to a complete process instead of an activity. In this case this general error handling process will be used if no other error handling process is associated with the failing activity. In case there is no error handling process assign to either the activity or the process a system default error handling process will be used.

One of the advantages of using processes for error handling is that standard process modeling tools and techniques can be used for designing error handling strategies as well. Therefore process implementers do not need to learn any new concept to provide error handing. Another important advantage is that error handling at a higher semantical 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.

End user perspective: In certain cases the simple error handling process from Fig. 3 might be not appropriate since it increases 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 executing this activity could be provided with a set of possible error handling strategies he can choose from. An example for such more complex error handling process is shown in Fig. 4. Here the user can choose between a variety of ways to handle the respective error: retrying the failed process step, aborting the whole process instance or applying predefined ad-hoc changes to fix or compensate the error.

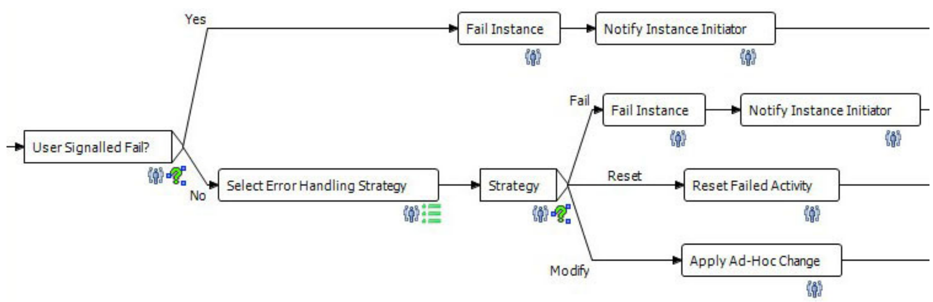


Fig. 4. A more complex error handling process involving the user

Additionally, suggested error handling strategies may depend on the background of the respective user, i.e. his knowledge and position in the organizational model and various other factors. Based on the selection of the user the respective strategy is then applied to handle the error.

Such a semi-automatic, user-centered approach offers many advantages. Since for each process activity a predefined set of possible strategies can be provided to users, they do not need to have deep insights into the process to handle errors appropriately. This allows to significantly reduce waiting times for failed instances since users can handle errors immediately by their own and do not have to wait for a probably busy helpdesk to handle errors for them. This in turn allows to relieve the helpdesk from the tedious task of handling simple process errors.

System supervisor perspective: Certain errors cannot be handled by the user. For example they might not have been foreseen at buildtime, i.e., no appropriate error handling process exists; or it might be simply not possible to handle errors in an easy and generic way. In such cases the system supervisor can use the *AristaFlow Process Monitor* shown in Fig. 5 to take a look at this process instance, to analyze its execution log, and decide for an appropriate error handling. Additionally the system supervisor can use the *AristaFlow Process Monitor* to keep track of failed instances; e.g., he may intervene if a web service becomes unavailable permanently.

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,

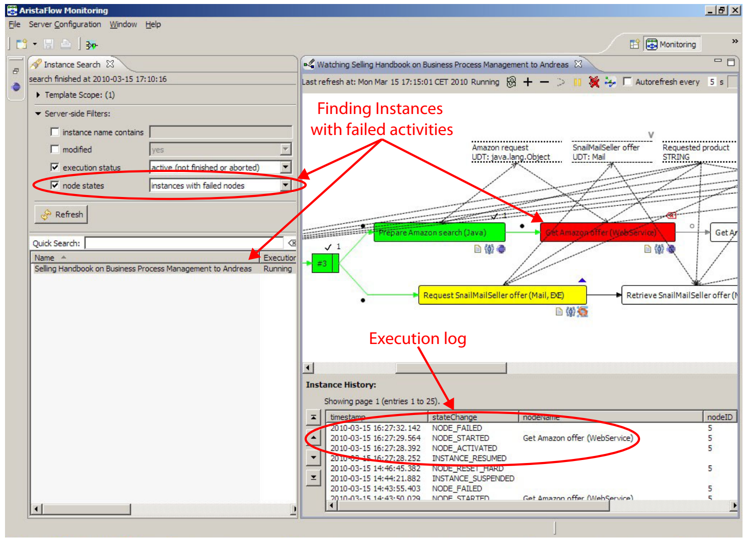


Fig. 5. Process Monitor: Monitoring Perspective

but then fails in doing so. The system administrator detects that the process is in trouble and uses the *AristaFlow Process Monitor* to take a look at this process instance (cf. Fig. 5). 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 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 sees that the result of this and the subsequent activity is 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. 6). 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*. In fact, it is the *Process Template Editor*, 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. 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. 6. 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 administrator to delete the activities in trouble and to replace them with a form-based activity which allows to enter the price manually.

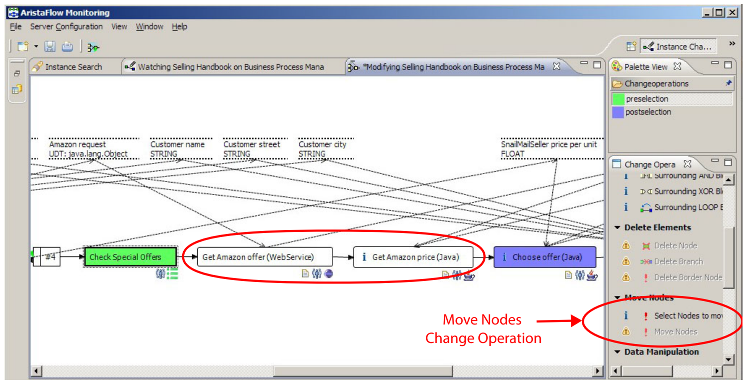


Fig. 6. Process Monitor: Instance Change Perspective

5 Related Work

Besides ADEPT, YAWL [3] has been one of the first workflows engines to support some sort of "correctness by construction" as well as correctness checks at buildtime. jBPM [4] rudimentarily supports ad-hoc deviations of running process instances, but without any correctness assurance as provided by the AristaFlow BPM Suite. Most BPEL-based workflow engines like Websphere Process Server [5] support error handling processes using fault handlers, but without the possibility to structurally change process instances during runtime.

6 Summary and Outlook

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. Currently, we investigate the handling of other kinds of errors (e.g. time related errors).

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** (2009) 115–135
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** (2009) 81–97
3. Russell, N., ter Hofstede, A.H.M.: Surmounting BPM challenges: the YAWL story. Computer Science - R&D **23** (2009) 67–79
4. Koenig, J.: JBoss jBPM (whitepaper) (2004)
5. Kloppmann, M., Konig, D., Leymann, F., Pfau, G., Roller, D.: Business process choreography in websphere: Combining the power of BPEL and J2EE. IBM Systems Journal **43** (2004) 270–296

SeaFlows Toolset – Compliance Verification Made Easy*

Linh Thao Ly[1], David Knuplesch[1], Stefanie Rinderle-Ma[2], Kevin Göser[3], 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] Faculty of Computer Science
University of Vienna, Austria
stefanie.rinderle-ma@univie.ac.at

[3] 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 tool demonstration extends process-aware information system by compliance checking functionality. It provides a user-friendly environment for modeling compliance rules using a graph-based formalism. Modeled compliance rules can be used to enrich process models. To address a multitude of verification settings, SeaFlows Toolset provides two compliance checking components: The *structural compliance checker* derives structural criteria from compliance rules and applies them to detect incompliance. The *data-aware compliance checker* 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.

Key words: Compliance rules, Process verification, Tool support, Data-awareness

1 Introduction

In the light of an increasing demand on 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).

compliance rules has become essential in enterprise computing. To ensure compliance with imposed rules and policies, compliance audits for process models are necessary. Due to 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 and evolution of regulatories and policies may occur, leading to changes in implemented compliance rules. On the other hand, changes to business processes may take place, resulting in changes of implemented process models. This further necessitates tool support for (semi-)automatic compliance verification.

The toolset featured in this tool demonstration 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 inludes compliance cheking of business process models at buildtime but also compliance monitoring of 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. The particular components shown in this tool demonstration enable modeling compliance rules as visual compliance rule graphs as well as verifying process models against imposed compliance rules [4]. To support a variety of verification scenarios and to exploit their specific properties, SeaFlows Toolset comprises several verification components: a *structural compliance checker*, enabling efficient compliance verification for block-structured process models and a *data-aware compliance checker*, enabling data-aware compliance checking using model checking techniques.

In the following, the particular components of SeaFlows Toolset are introduced. Related work is discussed in Sect. 3 before we close the paper with an outlook on future developments in Sect. 4

2 SeaFlows Toolset

SeaFlows Toolset extends process-aware information system (PAIS) by compliance checking functionality. Fig. 1 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[2].

The SeaFlows Graphical Compliance Rule Editor (cf. Fig. 1) allows to model compliance rules over process artifacts as *compliance rule graphs* [4] (cf. Sect. 2.1). By interacting with the activity repository responsible for organizing and 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 perform compliance audits.

[2] The Rule Graph Execution Engine for executing compliance rule graphs is currently under implementation

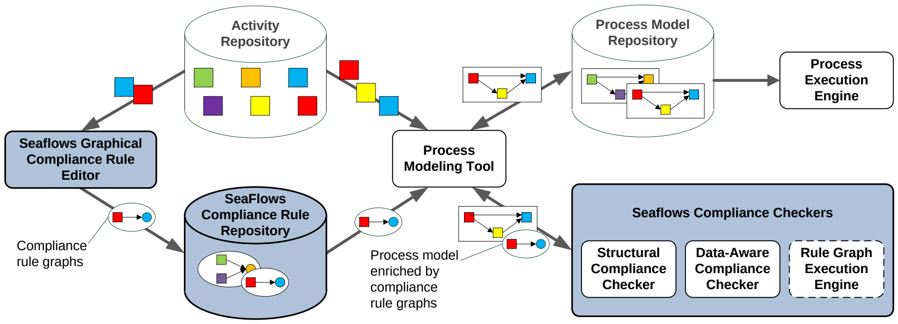


Fig. 1. Overall infrastructure around the SeaFlows Toolset

SeaFlows Toolset currently comprises two compliance checking components to verify process models (cf. Fig. 1), namely the Structural Compliance Checker and the Data-aware Compliance Checker. By interacting with the process modeling tool of PAIS, the SeaFlows compliance checkers enable the process designer to verify process models already during process design. Meaningful compliance reports help the process designer to identify incompliant process behaviour. Based on them, the process designer 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 the commercial process management system AristaFlow BPM Suite orginated from research activities in the ADEPT project [5]. AristaFlow BPM Suite provides a powerful 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 (cf. Fig. 1) and underlying concepts are discussed in more detail.

2.1 Graphical Compliance Rule Editor and Compliance Rule Repository

We developed a graph-based compliance rule specification language that enables modeling compliance rules in a manner similar to process modeling. Designed to support intuitive compliance rules modeling, *compliance rule graphs* are modeled by linking nodes representing absence and occurrence of activity executions of certain types [4]. In particular, (sub-)graphs are used to respresent an antecedent pattern that activates the compliance rule and corresponding required consequence patterns. This enables modeling frequent compliance rule patterns [6, 7] in a straightforward manner. Further, compliance rule graphs can be enriched with annotations of temporal constraints (e.g., minimal temporal distance) as well as data conditions.

The Graphical Compliance Rule Editor provides a user-friendly environment for modeling compliance rule graphs (cf. Fig. 2). Nodes of compliance rule graphs

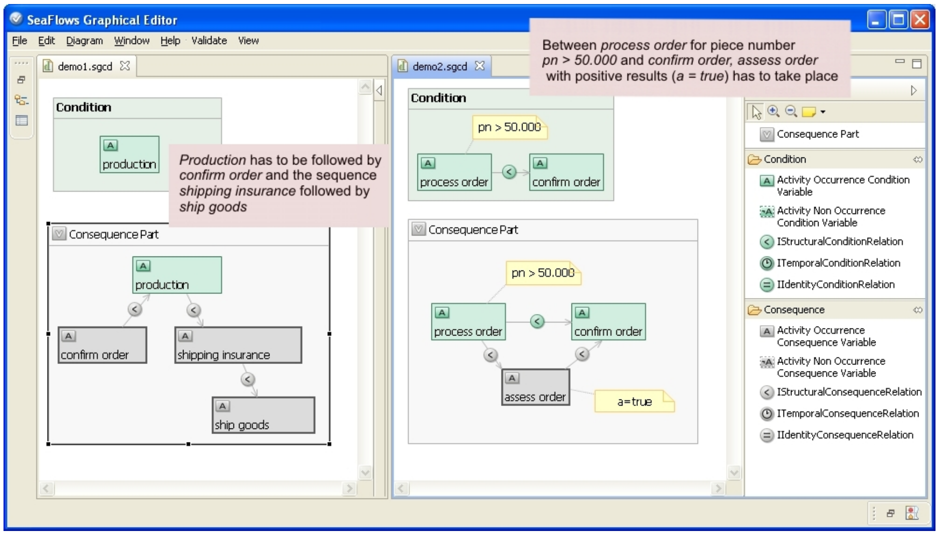


Fig. 2. The SeaFlows Graphical Compliance Rule Editor

are assigned to activity types available in the activity repository (cf. Fig. 1). Modeled compliance rule graphs are exported as separate XML-files which enables their organization within rule sets in the Compliance Rule Repository. In addition, versioning of compliance rules is also supported by the repository. Being implemented based on Eclipse Modeling Framework, modeled compliance rule graphs are based on a defined data object model that facilitates their import and processing in compliance checking tools.

2.2 Structural Compliance Checker

The basic idea underlying the Structural Compliance Checker is to efficiently verify process models by automatically deriving *criteria on the process structure* from compliance rules [8]. Following the dynamic programming paradigm, for each compliance rule a set of simple binary structural criteria (such as "A excludes B") whose satisfaction ensure compliance with the corresponding rule is derived. By checking the process model for compliance with these derived criteria, we can identify the criteria not fulfilled by the process model. This is useful information to generate intelligible textual feedback in case incompliance is detected. Based on the results of checking the structural criteria, the Structural Compliance Checker is able to provide detailed diagnosis that is helpful to locate incompliance (cf. Fig. 3). For example, the feedback Fig. 3 indicates that *shipping insurance* is optional to *production* in the process model. This detailed diagnosis can further be applied to resolve incompliance.

By making assumptions on the verification setting (e.g., unique label assumption) and exploiting the block-structure of process models the Structural Compliance Checker identifies incompliance in an efficient manner.

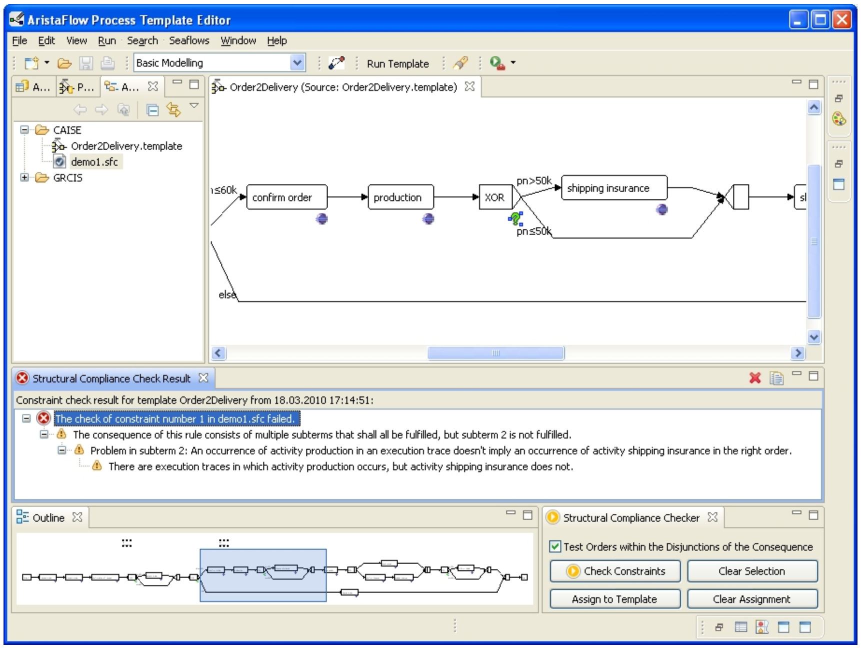


Fig. 3. The SeaFlows Structural Compliance Checker integrated into AristaFlow Process Template Editor

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

2.3 Data-aware Compliance Checker

The Data-aware Compliance Checker is able to deal with data-aware compliance rules and data conditions in process control flow. 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 developed a process-meta-model-independent approach for automatic context-sensitive (i.e., rule-specific) abstraction (cf. Fig. 4 B). By analyzing the data conditions contained in the compliance rule and in the process model, it reduces the state space of the data dimension to be explored during verification. The obtained *abstract process model* and *abstract compliance rule* are given as input to the actual compliance checking procedure (cf. Fig. 4 A). For compliance checking we used a model checker. In case of violation, the counterexample obtained from the model checker is conretized to yield not only the incompliant execution but also the its data conditions.

The Data-aware Compliance Checker first performs automatic abstraction, then transforms the abstract process model into a state space representation.

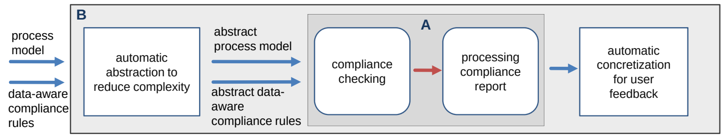


Fig. 4. Abstraction and concretization as pre- and postprocessing steps to the actual data-aware compliance checking

The latter is then passed to the model checker SAL [9], which carries out the exploration of the abstract process model. In case compliance violation is detected, the Data-aware Compliance Checker retransforms the counterexample output of the model checker and visualizes it as an execution trace and as process graph.

Similar to the Structural Compliance Checker, the Data-aware Compliance Checker is directly integrated into the process modeling environment. 17.000 lines of code and the class hierarchy comprising about 70 interfaces and 210 classes indicate its complexity. Automatic abstraction is supported for domains of numbers and for large domains of object references.

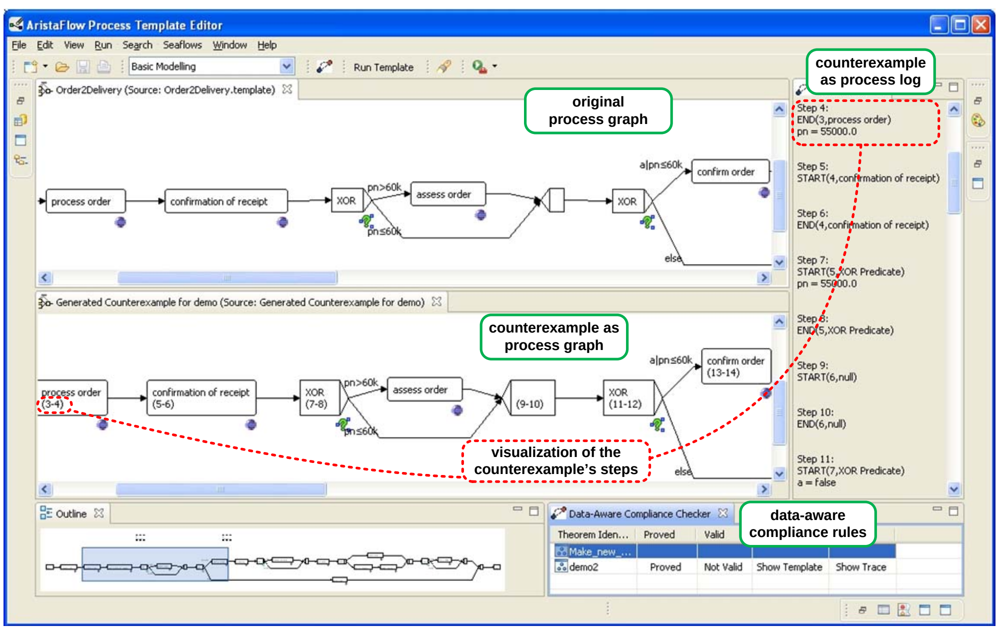


Fig. 5. The Data-aware Compliance Checker visualizes the counterexample as execution trace and process graph

3 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 [10], to visual patterns and languages [11, 12, 13]. With the compliance rule graphs, 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 [12, 13, 10]. 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 (e.g., counterexample) provided by the model checker. SeaFlows Toolset implements two compliance checking approaches, one based on model checking and another based on structural criteria, that complement each other.

Some approaches address the verification of data-aware compliance rules [11, 12]. However, the state explosion problem 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.

[7] addresses visualization of incompliance by querying the process model for anti-patterns that are defined for each compliance rule pattern. In our approach, structural criteria are automatically derived from the compliance rule by the Structural Compliance Checker. Checking the structural criteria allows for identifying precisely the structural reason for incompliance.

Similar to DECLARE [14], the declarative process management system, SeaFlows enables to model graphical compliance rules. In DECLARE constraints are mapped onto formula in temporal logic and then to finite automata in order to execute constraint-based workflows. In contrast, SeaFlows compliance rule graphs are used to verify process models.

SeaFlows Toolset can be further complemented by other process analysis tools, such as the process mining framework ProM [15] to provide comprehensive support of compliance checking a priori as well as a posteriori.

4 Summary and Outlook

SeaFlows Toolset featured in this tool demonstration extends process-aware information system by compliance checking functionality. It enables modeling compliance rule as graphs independently from specific process models by making use of an activity repository. Process models can be enriched by compliance rules 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 (e.g., data-awareness) 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. 1). In addition, SeaFlows Toolset will be extended by a visualization and explanation component to provide advanced user feedback.

References

1. Sadiq, S., Governatori, G., Naimiri, K.: Modeling control objectives for business process compliance. In: Proc. BPM '07. (2007) 149–164
2. Bobrik, R., Reichert, M., Bauer, T.: View-based process visualization. In: Proc. of the 5th Int'l Conf. on Business Process Management (BPM'07). Volume 4714 of LNCS. (2007) 88–95
3. Ly, L.T., Rinderle-Ma, S., Dadam, P.: On enabling integrated process compliance with semantic constraints in process management systems. Information Systems Frontiers , accepted for publication.
4. Ly, L.T., Rinderle-Ma, S., Dadam, P.: Design and verification of instantiable compliance rule graphs in process-aware information systems. In: The 22nd Int'l Conf. on Advanced Information Systems Engineering (CAiSE'10). (2010) accepted for publication.
5. 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) (2009) 81–97
6. Dwyer, M.B., Avrunin, G.S., Corbett, J.C.: Patterns in property specifications for finite-state verification. In: Proc. ICSE'99. (1999) 411 – 420
7. Awad, A., Weske, M.: Visualization of compliance violation in business process models. In: Proc. BPM Workshops. (2009)
8. Ly, L.T., Rinderle-Ma, S., Dadam, P.: Integration and verification of semantic constraints in adaptive process management systems. Data & Knowledge Engineering **64** (2007) 3–23
9. Bensalem, S., et al.: An overview of SAL. In: Proc. of the Fifth NASA Langley Formal Methods Workshop, NASA Langley Research Center (2000) 187–196
10. Ghose, A., Koliadis, G.: Auditing business process compliance. In: ICSOC '07. Volume 4749 of LNCS., Springer (2007) 169–180
11. Awad, A., Weidlich, M., Weske, M.: Specification, verification and explanation of violation for data aware compliance rules. In: Proc. ICSOC/ServiceWave'09. (2009) 500–515
12. Liu, Y., Müller, S., Xu, K.: A static compliance-checking framework for business process models. IBM Systems Journal **46**(2) (2007) 335–261
13. 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. (2007)
14. Pesic, M., Schonenberg, M., Sidorova, N., van der Aalst, W.: Constraint-based workflow models: Change made easy. In: OTM 2007, Part I. Number 4803 in LNCS, Springer-Verlag (2007) 77–94
15. van der Aalst, W., et al.: ProM 4.0: Comprehensive support for real process analysis. In: Proc. ICATPN'07. Volume 4546 of LNCS., Springer (2007) 484–494

The NORMA Tool for ORM 2

Matthew Curland[1] and Terry Halpin[2]

[1]LogicBlox, USA
[2]LogicBlox, Australia and INTI Education Group, Malaysia
e-mail: {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, NORMA typically provides greater expressive power and semantic stability than provided by tools based on ER or UML. 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) [17], fact-orientation makes no use of attributes as a way to encode facts, instead representing all ground assertions of interest as atomic (non-decomposable) facts that are either existential facts (e.g. There is a country named 'Australia') or elementary facts that predicate over first-order individuals (objects that are either entities or values).

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) 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).

Business rules are modeled 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), while 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.

Fact-oriented modeling approaches include *Object-Role Modeling* (*ORM*) [8], Cognition-enhanced Natural Information Analysis Method (CogNIAM) [15], the Predicator Set Model (PSM) [13], 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 [18]. For an overview of fact-oriented modeling approaches, including history and research directions, see [7].

Since the 1970s, various tools have been developed to support fact-orientation. Early tools based on NIAM include IAST and RIDL* (based on the RIDL language [14]). 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 been developed based on second generation ORM (*ORM 2*) [6]. These include *Natural ORM Architect (NORMA)*, ActiveFacts [12], 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. 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 [11].

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 [16]. 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, or drag new elements off the toolbox. New model components are added to the conceptual model and displayed with graphical shapes on one or more ORM diagrams The Model Browser tool window also provides a hierarchical view of all model components. Sample object and fact instances may be entered in tabular format in the Sample Population Editor.

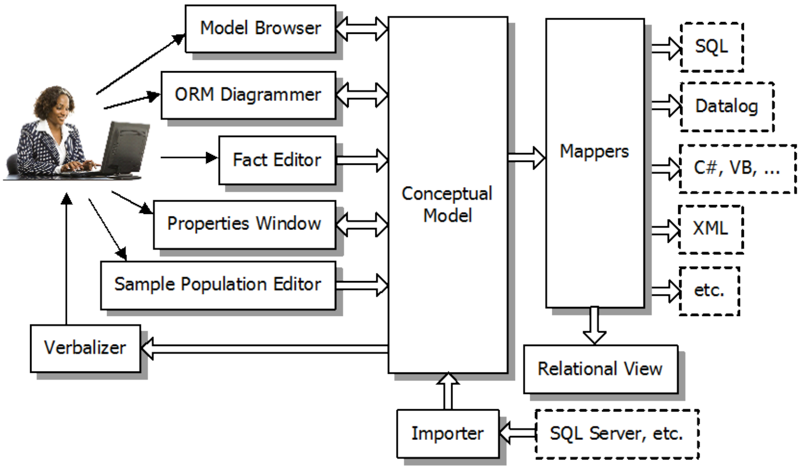


Fig. 1. Main components of NORMA

Currently, 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 non-technical people. Our modeling 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, .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 can reverse engineer relational schemas in SQL Server into ORM schemas. Import from further sources is planned.

Other components facilitate navigation and abstraction. For example, multiple concurrent windows viewing the same model allow shapes to be copied between diagrams, the ORM Diagram Spy and hyperlinks in the Verbalization Browser allow rapid navigation through a model, and the ORM Context Window automatically displays the global schema neighborhood of a selected ORM element.

3 Examples of Some NORMA Features

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.

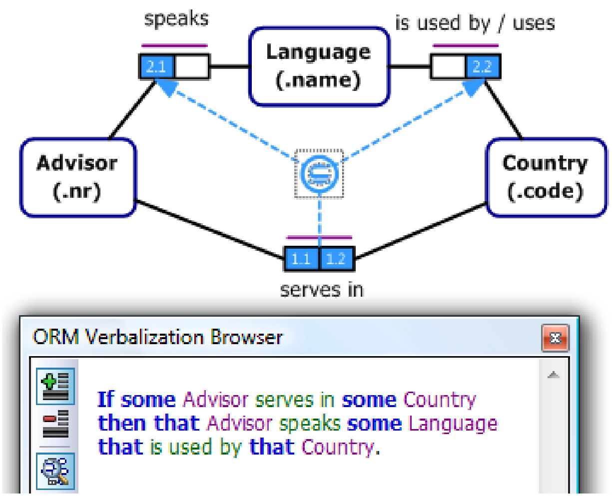


Fig. 2. NORMA screenshot showing verbalization of a join-subset constraint

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. 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.

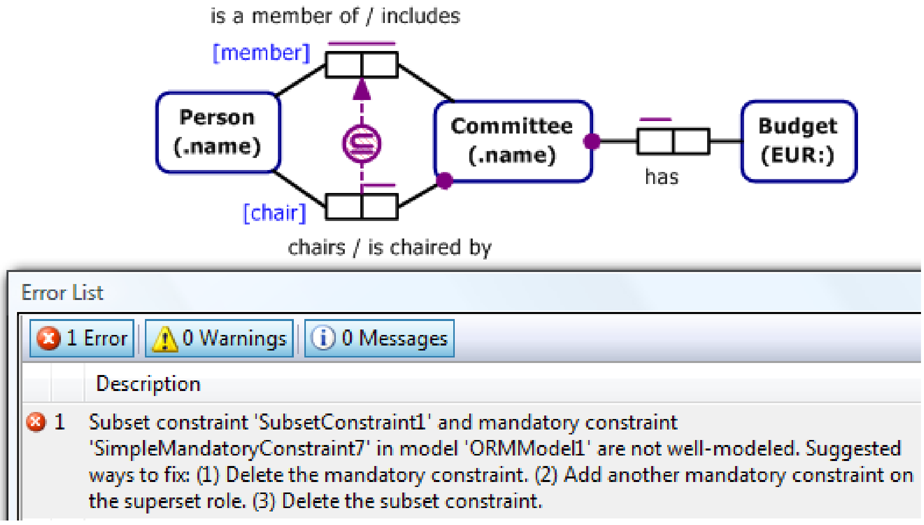
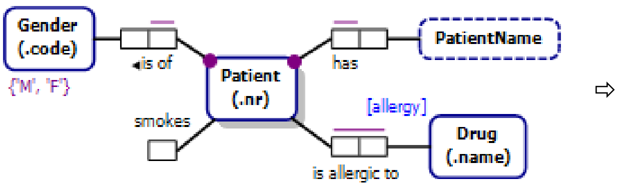
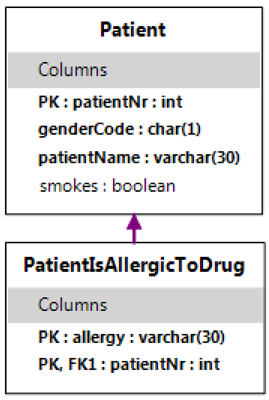


Fig. 3.

Fig. 4 illustrates a very simple example of *mapping* from ORM to a relational schema as well as a datalog schema. For a detailed discussion of a much more complex example, as well as comparisons with ER and UML, 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).
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(x) -> patient:gender[x]=_.
Patient(x) -> patient:name[x]=_.
gender:code(_:y) -> y="M"; y="F".
```

Fig. 4. Simple example of mapping an ORM schema to a relational schema and datalog

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 can be classified in 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.
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. Extension models provide schematized XML components and a mapping from the in-memory model to the XML elements, load fixup mechanisms to reach an internally consistent state at load completion, and rules to maintain consistency after the model is loaded. Provided extensions in this category include the core ORM metamodel (*FactType*, *ObjectType*, etc) and the relational metamodel, which contains elements such as *Table*, *Column*, and *ReferenceConstraint*.

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.
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 for the XML schema of the relational model that is generated directly from a database and has no ORM information. Bridge models enable changes in one primary model to be applied to another loaded standalone model while maintaining the relationships between the source and target.
5. *Shell Components* are views and editors targeting specific parts of an inmemory ORM model, such as the Model Browser and Fact Editor tool windows discussed earlier.
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 additional columns for each table, with the modified file continuing in the generation pipeline.

5 Conclusion

This paper provided a brief overview of the NORMA tool and its support for ORM 2. 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 [10]. Research is also under way to extend NORMA with support for dynamic rules [2].

References

1. Bakema, G., Zwart, J., & van der Lek, H. 2000, *Fully Communication Oriented Information Modelling*. Ten Hagen Stam.
2. Balsters, H. & Halpin, T. 2008, 'Formal Semantics of Dynamic Rules in ORM', *On the Move to Meaningful Internet Systems 2008: OTM 2008 Workshops*, eds. R. Meersman, Z. Tari, P. Herrero et al., Monterrey, Mexico, Springer LNCS 5333, pp. 699-708.
3. Bloesch, A. & Halpin, T. 1997, 'Conceptual queries using ConQuer-II', *Proc. ER'97: 16th Int. Conf. on conceptual modeling*, Springer LNCS, no. 1331, pp. 113-126.
4. Chen, P. P. 1976, 'The entity-relationship model—towards a unified view of data'. *ACM Transactions on Database Systems*, 1(1), pp. 9−36.
5. Curland, M., Halpin, T. & Stirewalt, K. 2009, 'A Role Calculus for ORM', *On the Move to Meaningful Internet Systems 2008: OTM 2009 Workshops*, eds. R. Meersman, P. Herrero et al., Vilamoura, Portugal, Springer LNCS 5872, pp. 692-703.
6. Halpin, T. 2005, 'ORM 2', *On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops*, eds R. Meersman, Z. Tari, et al., Cyprus. Springer LNCS 3762, pp 676-87.
7. Halpin, T. 2007, 'Fact-Oriented Modeling: Past, Present and Future', *Conceptual Modelling in Information Systems Engineering*, eds. J. Krogstie, A. Opdahl & S. Brinkkemper, Springer, Berlin, pp. 19-38.
8. Halpin, T. 2009, 'Object-Role Modeling', *Encyclopedia of Database Systems*, ed. L, Liu & M. Tamer Ozsu, Springer-Verlag, Berlin.
9. Halpin, T. 2010, 'Object-Role Modeling: Principles and Benefits', *International Journal of Information Systems Modeling and Design*, Vol. 1, No. 1, IGI Global, pp. 32-54.
10. Halpin, T. & Wijbenga, J. P. 2010, 'FORML 2', *Enterprise, Business-Process and Information Systems Modeling*, eds. I. Bider et al., Springer LNBIP 50, pp. 247–260.
11. Halpin, T. & Morgan, T. 2008, *Information Modeling and Relational Databases, Second Edition*, Morgan Kaufmann, San Francisco.
12. Heath, C. 2009. ActiveFacts Website: http://dataconstellation.com/ActiveFacts/.
13. ter Hofstede, A., Proper, H. & van der Weide, T. 1993, 'Formal definition of a conceptual language for the description and manipulation of information models', *Information Systems*, vol. 18, no. 7, pp. 489-523.
14. Meersman, R. 1982, *The RIDL Conceptual Language*, Int. Centre for Information Analysis Services, Control Data Belgium, Brussels.
15. Nijssen, M., & Lemmens, I. 2008, 'Verbalization for Business rules and Two Flavors of Verbalization for Fact Examples', *On the Move to Meaningful Internet Systems 2008: OTM 2008 Workshops*, eds. R. Meersman, Z. Tari, P. Herrero et al., Springer LNCS 5333, pp. 760-769.
16. NORMA (Natural ORM Architect) tool download site for public-domain version: http://www.ormfoundation.org/files/folders/norma_the_software/default.aspx.
17. Object Management Group 2009, *UML 2.2 Specifications*. Online at: www.omg.org/uml.
18. Object Management Group 2008, *Semantics of Business Vocabulary and Business Rules (SBVR)*. Online at: http://www.omg.org/spec/SBVR/1.0/.

XES Tools

H.M.W. Verbeek, J.C.A.M. Buijs, B.F. van Dongen, and W.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@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 - *XESMa* and *ProM 6* - and highlights the main innovations and the role of XES. *XESMa* 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 [1–4, 6, 15]. 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.

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 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 Mapper tool. 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 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

Fig. 1 shows the XES meta model, which is taken from [11]. In XES the log, trace and event objects only define the structure of the document: they do not contain

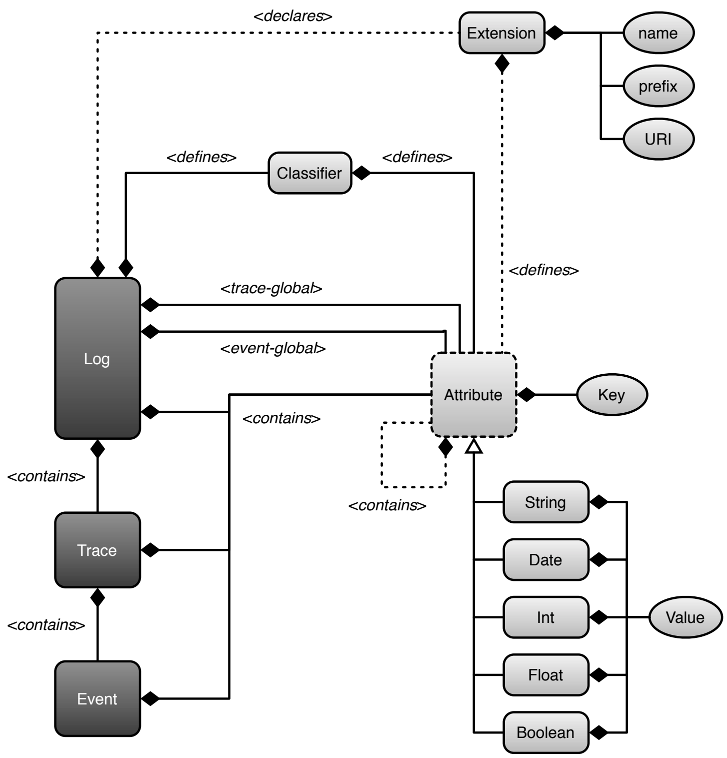


Fig. 1. XES Meta Model.

any information themselves. To store any information, attributes are used. Every attribute has a string based key and a value of some type. Possible value types are string, date, integer, float and boolean. Note that attributes can have attributes themselves which can be used to provide more specific information.

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. Note that the semantic extension is inspired by SA-MXML (Semantically Annotated MXML) [14].

Furthermore, *event classifiers* can be specified in the log object 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.

Table 1. List of XES extensions and the attribute keys they define.

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.

3 XES Mapper

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 an 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 the XES Mapper [5], or *XESMa* for short.

We use an example to explain XESMa. From some company, we received a database export in the form of thirteen CSV (Comma Separated Values) tables. From the thirteen tables, only two were required for the event log extraction.

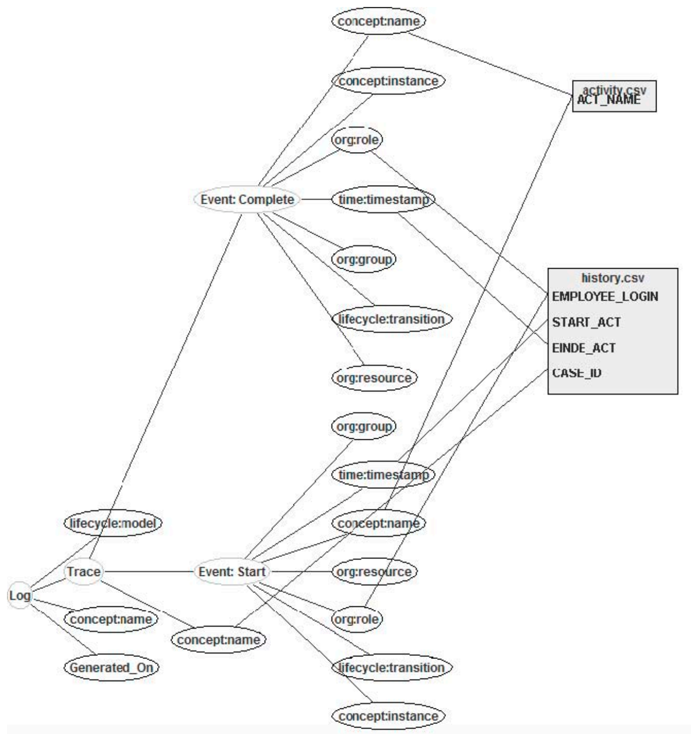


Fig. 2. Mapping visualization.

The first table (history.csv) contains 19,223,294 records, measures 2.14 GB and holds the history of all activities performed in the year 2008, while the second table (activity.csv) contains 811 records, measures 45 KB and holds additional information on the tasks defined in the system.

First, the domain expert needs to tell XESMa how the event log should be extracted from both tables. Fig. 2 shows the visual representation of this mapping. The left-hand side of Fig. 2 shows a log, a trace, two events, and their attributes, whereas the right-hand side shows both tables. The lines from the attributes to the tables indicate how the actual value for this attribute is extracted from the tables. As an example, the time:timestamp attribute of a Start event will be extracted from the START_ACT field of the history.csv table. Note that although we only have two events in the mapping, the resulting event log will contain almost 40 million events as for every record from the history.csv table both a Start event and a Complete event will be generated, and that although we only have a single trace, the resulting log will contain as many traces as the history.csv table contains different values for the CASE_ID field.

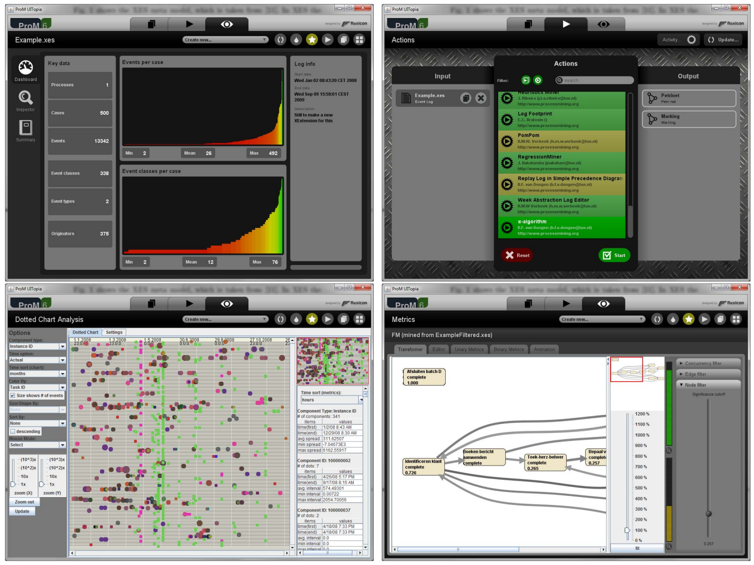


Fig. 3. ProM 6 results.

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

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.

We use a selection of the XES event log obtained from XESMa, as described in the previous section, to showcase ProM 6. Fig. 3 shows some results obtained. The left upper view shows some basic characteristics of the log, like the number of traces, number of events, and distribution of trace length. The right upper view shows the list of installed plug-ins with the α-miner selected. On the left-hand side of this view the necessary inputs for this plug-in are shows, while on the right-hand side the expected outputs are shown. Note that ProM is aware of these inputs and outputs, which allows us to chain series of plug-ins into workflows to conduct larger process mining experiments. The left bottom view shows a dotted chart [16] on a filtered part of the log, whereas the right bottom view shows the result of the fuzzy model [12] mined from this filtered log.

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, XESMa [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 doing the extraction. Instead, XESMa 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.

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. W.M.P. van der Aalst, H.A. Reijers, A.J.M.M. Weijters, B.F. van Dongen, A.K. Alves de Medeiros, M. Song, and H.M.W. Verbeek. Business Process Mining: An Industrial Application. *Information Systems*, 32(5):713–732, 2007.

2. W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. Workflow Mining: A Survey of Issues and Approaches. *Data and Knowledge Engineering*, 47(2):237–267, 2003.
3. W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
4. R. Agrawal, D. Gunopulos, and F. Leymann. Mining Process Models from Workflow Logs. In *Sixth International Conference on Extending Database Technology*, pages 469–483, 1998.
5. J.C.A.M. Buijs. Mapping Data Sources to XES in a Generic Way. Master's thesis, Eindhoven University of Technology, 2010.
6. A. Datta. Automating the Discovery of As-Is Business Process Models: Probabilistic and Algorithmic Approaches. *Information Systems Research*, 9(3):275–301, 1998.
7. B.F. van Dongen and W.M.P. van der Aalst. A Meta Model for Process Mining Data. In J. Casto and E. Teniente, editors, *Proceedings of the CAiSE'05 Workshops (EMOI-INTEROP Workshop)*, volume 2, pages 309–320. FEUP, Porto, Portugal, 2005.
8. B.F. van Dongen, A.K. Alves de Medeiros, H.M.W. Verbeek, A.J.M.M. Weijters, and W.M.P. van der Aalst. The ProM framework: A New Era in Process Mining Tool Support. In G. Ciardo and P. Darondeau, editors, *Application and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 444–454. Springer-Verlag, Berlin, 2005.
9. C. Günther and W.M.P. van der Aalst. A Generic Import Framework for Process Event Logs. In J. Eder and S. Dustdar, editors, *Business Process Management Workshops, Workshop on Business Process Intelligence (BPI 2006)*, volume 4103 of *Lecture Notes in Computer Science*, pages 81–92. Springer-Verlag, Berlin, 2006.
10. C. W. Günther. *Process Mining in Flexible Environments.* PhD thesis, Eindhoven University of Technology, Eindhoven, 2009.
11. C. W. Günther. XES Standard Definition. Fluxicon Process Laboratories, November 2009.
12. C.W. Günther and W.M.P. van der Aalst. Fuzzy Mining: Adaptive Process Simplification Based on Multi-perspective Metrics. In G. Alonso, P. Dadam, and M. Rosemann, editors, *International Conference on Business Process Management (BPM 2007)*, volume 4714 of *Lecture Notes in Computer Science*, pages 328–343. Springer-Verlag, Berlin, 2007.
13. IEEE Task Force on Process Mining. www.win.tue.nl/ieeetfpm.
14. A.K. Alves de Medeiros, C. Pedrinaci, W.M.P. van der Aalst, J. Domingue, M. Song, A. Rozinat, B. Norton, and L. Cabral. An Outlook on Semantic Business Process Mining and Monitoring. In R. Meersman, Z. Tari, and P. Herrero, editors, *Proceedings of the OTM Workshop on Semantic Web and Web Semantics (SWWS '07)*, volume 4806 of *Lecture Notes in Computer Science*, pages 1244–1255. Springer-Verlag, Berlin, 2007.
15. A. Rozinat and W.M.P. van der Aalst. Conformance Checking of Processes Based on Monitoring Real Behavior. *Information Systems*, 33(1):64–95, 2008.
16. M. Song and W.M.P. van der Aalst. Supporting Process Mining by Showing Events at a Glance. In K. Chari and A. Kumar, editors, *Proceedings of 17th Annual Workshop on Information Technologies and Systems (WITS 2007)*, pages 139–145, Montreal, Canada, December 2007.

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.

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).

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 composing the foundations of our work. Section 3 deals with the implementation of the framework (named Utopia) allowing the implementation of such Electronic Institution. At last, before conclude, the Section 4 illustrates the use of Utopia 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):

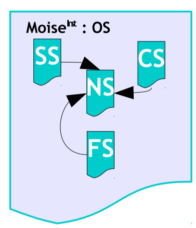


Fig. 1. $\mathcal{M}$OISE^{Inst}, a normative Organization Specification model

- 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 supervising the actions of agents "Agt" on the OE.

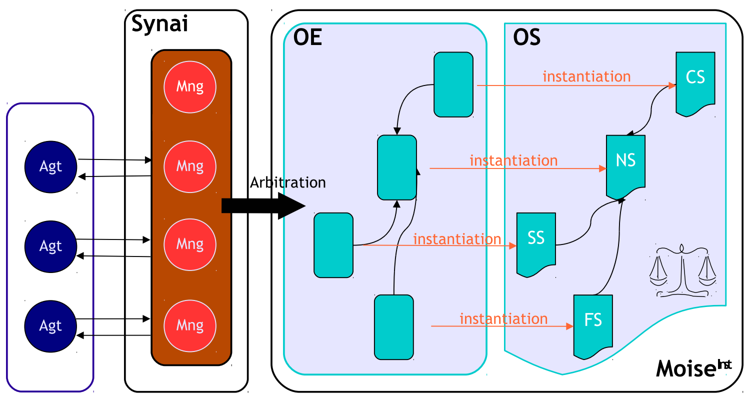


Fig. 2. Supervision by $\mathcal{S}$YNAI of an OE

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 Implementation of Utopia

Recursive graphs are innovative mathematical structures [8] widely used to have a very generic representation of data. In our case, a recursive graph particularly meets the underlying needs of $\mathcal{M}$OISE^{Inst} which is mostly recursive : Groups can include others Groups, Missions can include others Missions, etc... Moreover, the sub recursive graph extraction makes the data sharing more easier.

Utopia and its architecture using an Electronic Institution paradigm make the essential problematics of setting up a Multi-Agent System easier. Indeed 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.

4 Demonstration scenario

Our use-case is part of a demonstrator set up in the context of the RED project [9] 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 3:

– 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,

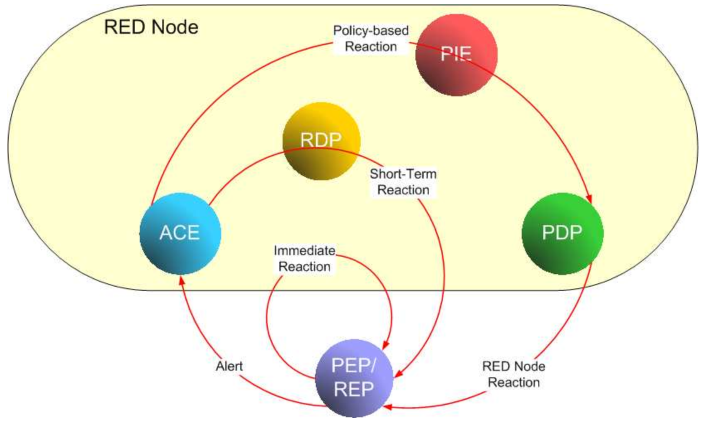


Fig. 3. RED architecture

– 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 $\mathcal{M}$OISE^{Inst} Organization. In the following, we will describe the Goals that agents have to achieve in a context of a black-hole attack.

4.1 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 Utopia in order to represent a RED node as an Electronic Institution.

4.2 Implementation with Utopia

Utopia 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. The following shows Domain Knowledge Specification of the goals binding them to their corresponding java classes that the user have to provide, and the FS coming from the OS XML file. There is no grouping of goals in missions, that's why the FS is so simple.

```
<?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">
    <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>
[...]
    <FunctionalSpecification>
      <GoalId>gPIESend</GoalId>
      <GoalId>gACESend</GoalId>
      <GoalId>gPDPListen</GoalId>
      <GoalId>gPEPListen</GoalId>
      <GoalId>gPEPIPListen</GoalId>
    </FunctionalSpecification>
</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 Utopia's primitive functions, it is very easy to send or receive messages and XML alerts.

5 Conclusion

In this paper we described an Electronic Institution programming framework named Utopia based on $\mathcal{M}\text{OISE}^{Inst}$ 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 Utopia using concepts such as sub-recursive graphs.

With the RED use-case we showed how easily the essential problematics of setting up a Multi-Agent System could be solved with Utopia and its powerful architecture using an Electronic Institution paradigm. Actually Utopia 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 Utopia 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 funded by Luxembourger FNR-CORE project TITAN (C08/IS/21).

References

1. North, D.C.: Institutions, Institutional Change and Economic Performance. 1st edition edn. Political Economy of Institutions and Decisions. Cambridge University Press (October 26 1990)
2. Jones, A., Carmo, J.: Deontic logic and contrary-to-duties. In: Handbook of Philosophical Logic. Kluwer (2001) 203–279
3. Hübner, J.F., Sichman, J.S., Boissier, O.: A model for the structural, functional, and deontic specification of organizations in multiagent systems. In: SBIA'02. Number 2507 in LNAI, Springer (2002) 118–128
4. Esteva, M., Rosell, B., Rodriguez-Aguilar, J.A., Arcos, J.L.: Ameli: An agent-based middleware for electronic institutions. In: AAMAS'2004, New York City, USA, ACM Press (19-23 July 2004) 236–243
5. Dignum, V., Vazquez-Salceda, J., Dignum, F.: Omni: Introducing social structure, norms and ontologies into agent organizations. In: ProMAS International Workshop 2004, New York, USA (2004)
6. Gâteau, B., Boissier, O., Khadraoui, D., Dubois, E.: Controlling an interactive game with a multi-agent based normative organizational model. In: COIN II. Volume 4386/2007 of LNCS., Springer (2007) 86–100
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 Springer, ed.: 23rd International Symposium on Mathematical Foundations of Computer Science. Volume 1450 of LNCS. (1998) 36–53
9. Feltus, C., Khadraoui, D., de Remont, B., Rifaut, A.: Business governance based policy regulation for security incident response. In: Crisis'07, Marrakech, Morocco (2-5 July 2007)

A Generative Approach for Creating Stakeholder-specific Enterprise Architecture Views

Sabine Buckl, Florian Matthes, and Christian M. Schweda

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

Abstract. Understanding the architectures of complex system, e.g. enterprises, is greatly facilitated by using graphical *views* thereof. These views result from the application of an underlying *viewpoint* to a comprehensive architectural description. The viewpoint thereby describes, which architectural concepts should be visualized in which way. The creation of views that consistently represent the enterprise architecture (EA) from a specific viewpoint, is a challenge of ongoing interest in EA management. In this paper, we present a technique that can be used to create views consistent to arbitrary architecture viewpoints and show, how this technique is realized in a prototypic tool. Central constituents of the tool are a model providing the graphical primitives for describing visualizations (called *visualization model*) and a model-to-model transformation reifying an architectural viewpoint.

Key words: model transformation, viewpoints, views, EA management

1 Motivation

A major task of enterprise architecture (EA) management is to provide transparency concerning the overall architecture, i.e. a common goal of EA management is to foster the communication between the different stakeholders with business and/or IT background [1]. Commonly, visualizations (views) are regarded as important means to facilitate communication between the stakeholders from business and IT. In order to be useful these visualizations should not be arbitrary "drawings" of the architecture, but should correspond to selected architecture *viewpoints*[1] on dedicated parts of the overall architecture (cf. [4, 9]). What up to this points reads quite similar to the challenge of architecture visualization in the context of software engineering, shows at the second look some subtle complexities. In contrast to the situation in software engineering, where widely-accepted conceptualizations of software systems, e.g. via "classes", "components", and "interfaces", exist, the field of EA management is lacking such well-defined and

[1] The term *viewpoint* is used in this context according to it's definition in [7].

grounded terminology. This might be ascribed to the novelty of the field, but quite a few researchers [1, 4] give a different explanation and challenge the idea that a "one-size-fits-it-all" conceptualization of EAs exists. These researchers in contrast expect the EA conceptualization to be organization-specific, such that every enterprise has its specific EA *information model* that incorporates only the information necessary for the dedicated EA management approach. Complementing the variety of information models, different organizations also employ different visualization standards, i.e. use viewpoints differing in respect to the used symbols as well as to the employed rules underlying the layout. Against the aforementioned background of largely differing information models and visualization principles, it becomes clear that the creation of a consistent EA view (for an example see Figure 1) is no simple task.

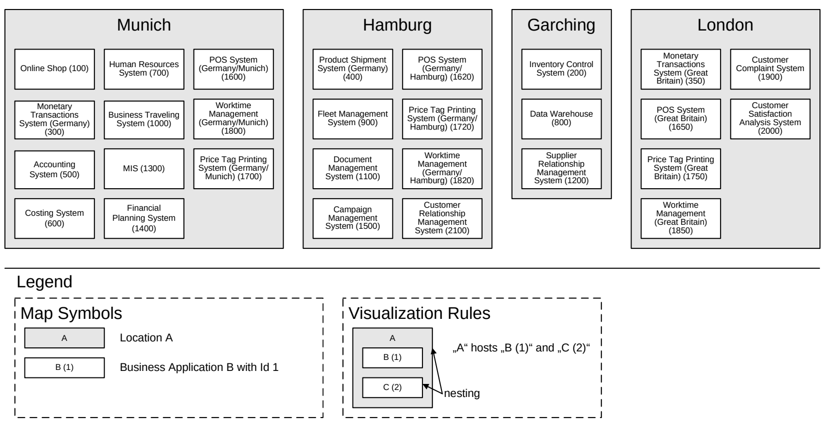


Fig. 1. Exemplary architectural view

A technique suitable for generating views consistent with underlying viewpoints based on arbitrary EA information models has to address different challenges. Matthes et al. give in [10] a detailed list of these challenges, of which we – due to reasons of brevity – only provide a summary:

Coupling of information and visualization meaning that a link should be maintained relating graphical elements in the visualization with the underlying architectural elements. The tool's mechanisms, thereby account for visualization correctness. If the graphical elements were in contrast decoupled from the underlying data, the views would degenerate to mere drawings, which need to be manually maintained if the underlying information changes.

Flexibility of information schema meaning that architectural models conforming to an arbitrary schema should be accessible to the tool. This reflects the fact that the information on the EA is gathered according to the specific demands of an organization.

Adaptability of viewpoint meaning that an enterprise architect should be able to adapt the viewpoint to its specific needs. For example, the architect should be able to adapt the colors according to the corporate identity of the enterprise. Adaptability thereby reflects the fact that visualizations, as mean of communication, heavily rely on stakeholder acceptance.

Composition and modularization of viewpoints meaning that an enterprise architect should be able to define *viewpoint modules*, i.e. reusable parts for creating visualizations. Thereby, especially the widely-used mechanism of *layering* visualizations is accounted for.

In Section 2 we outline a technique suitable for addressing the aforementioned requirements. The technique is based on the technology of model-to-model transformations, which is applied to connect information models on the one hand with a model of visual concepts, namely the *visualization model*, on the other hand. Complementing the description of the technique, we describe the prototypical realization thereof in a tool. Related tools as well as related techniques from nearby fields are presented in Section 3. Section 4 concludes the article with a critical reflection of the presented approach and an outlook on future work.

2 Generating visualizations via model transformation

Model transformation approaches in the context of generating visualizations can be used to maintain the strict separation between information and visualization, while also ensuring consistency between both concepts. The model transformation, called *syca transformation* in our approach, links different types of models, namely the model of the information to be visualized – the *semantic model* – and the model describing the visualization – the *symbolic model*. The transformation is thereby described relying on concepts of the models' respective meta models. These are the *information model* describing the concepts used for modeling information and the *visualization model* defining the graphical concepts for describing visualizations. Figure 2 illustrates the basic constituents of the approach and further introduces the concept of the *transformation meta model*, which provides the basis for specifying syca transformations, as well as the concept of a common meta model for both information and visualization model.

Different analyses of the information models used in EA management have been undertaken, e.g. by Buckl in [3], leading to the finding that the majority of currently used models follows the paradigm of object orientation. While the information models of many approaches do not explicitly account for the underlying meta-model, the analysis of Buckl further showed that mostly only core concepts of object-oriented modeling, e.g. *classes*, *properties*, and *associations* are used. This advocates for the utilization of a simplistic common meta-model with the OMG's Meta Object Facility (MOF) [11], more precisely the essential part thereof (EMOF) being a prominent candidate. The EMOF provides the meta-model of choice for the technique presented in this paper, especially as an implementation of the modeling facility is ready at hand as part of the *Eclipse Modeling Framework* (EMF). This framework was chosen, as its metamodel, the *ECore*-metamodel, can be considered to be very similar to the EMOF-metamodel[2]. Additionally, the EMF provides serialization and editing related functionalities, as well as an active user and developer community.

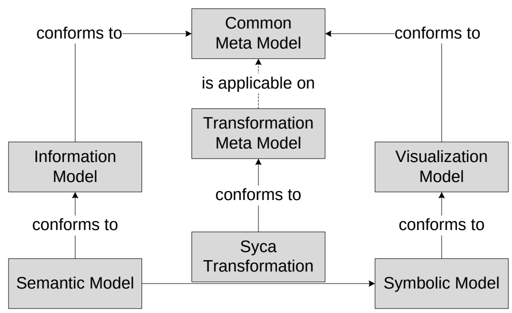


Fig. 2. Basic idea of the model transformation approach

2.1 Semantic model and information model – the *left* side

The information model sets up the language for describing the modeling subject, i.e. it introduces the core concepts, which are used to create a model of the subject's reality. In the context of EA management, the information model contains concepts like business processes, locations, etc., which are represented irrespective a visualization. Instance data documented in accordance to the information model is part of the semantic model, which contains so called *information objects*. In this sense, the information model (for an example see Figure 3) acts a meta-model for the semantic model, cf. Figure 4.

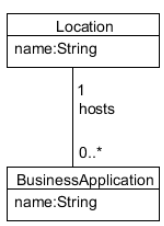


Fig. 3. Information model

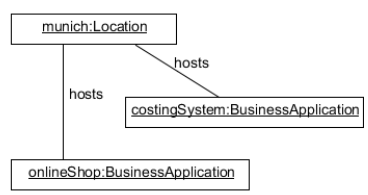


Fig. 4. Cutout of the semantic model

2.2 Symbolic model and visualization model – the *right* side

The visualization model contains elements, which represent graphical concepts, namely *map symbols*, e.g. "rectangle", or *visualization rules*, such as "nesting". These rules do not represent visible concepts, but they exert distinct demands on the positioning, size, or overall appearance of the symbol instances. For example, instances of the "nesting" rule demand that "inner" map symbol instances are grouped into the "outer" map symbol. Figure 5 introduces the map symbol

[2] Only minor differences concerning naming and the usage of references exist.

and visualization rule that are needed for the exemplary visualization. Figure 6 displays the symbolic model describing that rectangles representing business applications are nested in the rectangle representing the hosting location.

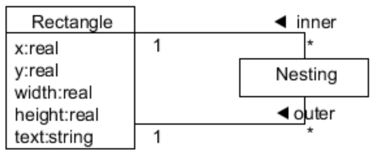


Fig. 5. Visualization model

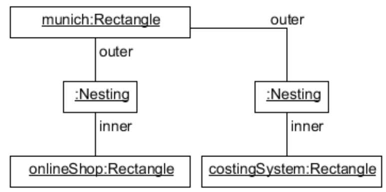


Fig. 6. Cutout of the symbolic model

The syca transformation creates a symbolic model based on the corresponding semantic model, while the map symbol instances in the symbolic model are not yet supplied with absolute positions. These positions are in a second step calculated by a layouter, which is capable to interpret the visualization rule instances and to compute appropriate positioning and sizing. Sketching the mathematical formalism incorporated in the layouter, we give examples of the layouting constraints that apply on the rectangle instance[3]:

```
munich.x - munich.width/2 < onlineShop.x - onlineShop.width/2
munich.x + munich.width/2 > onlineShop.x + onlineShop.width/2
munich.y - munich.height/2 < onlineShop.y - onlineShop.height/2
munich.y + munich.height/2 Y onlineShop.y + onlineShop.height/2
```

2.3 The syca transformation and its meta model – the *middle*

The syca transformation establishes the link between the data and its visualization and thereby enable the automatic generation of corresponding architectural views. Different types of model-to-model transformation languages may be used to implement the syca transformation.

The transformer component of the tool interprets the transformation rules and generates a symbolic model from the corresponding semantic model. Over the last years, we have successfully applied different model-to-model transformation languages for defining architectural viewpoints in an executable manner. Wiegelmann showed in [13] that the Atlas Transformation Language (ATL) [2] can be used to describe the necessary transformations. With ATL, architectural viewpoints can be defined in a highly declarative manner, although especially the appropriate instantiation of visualization rules becomes fairly complex, e.g. when matrix-like views should be created. Example ATL-like code generating the architectural visualization from Figure 1 is given below.

[3] According to the visualization model of Ernst et al. [6], the symbols' `x` and `y`-coordinates are anchored at the symbols' centers.

```
rule OrgUnit2Rectangle {
  from infoObject : Semantic.OrganizationalUnit
  to symbol : Symbolic.Rectangle (text = infoObject.name)
)

rule BusinessApp2Rectangle {
  from infoObject : Semantic.BusinessApplication
  to
    symbol : Symbolic.Rectangle (text = infoObject.name),
    rule : Symbolic.Nesting (
      inner = symbol,
      outer = transforming (infoObject.hostedAt)
    )
)
```

Complementing the model-to-model transformation languages, we further applied java to realize the syca transformations. Java-based transformations do – due to the maximum expressiveness – not run into the difficulties that the declarative model transformation languages have to deal with, but are even less intuitively to develop. Targeting an increased level of usability by rising the level of abstraction, Ramacher designed in [12] a java-based introspective framework consisting of highly-reusable transformation primitives that can be composed to syca transformations. First practical applications of this framework are still to be undertaken, but the first experiences are very promising.

3 Related approaches and tools

Domokos and Varro present in [5] an approach to ensure consistency between data and the corresponding visualization. The approach provides ”open visualization framework applicable to metamodel based modeling languages”, which is further developed towards executability. With no dedicated visualization model, the approach aims at transforming arbitrary information models to arbitrary visual languages, e.g. SVG, as far as both (information and visualization) can be described with *XML*. Consequently, the generation of a visualization in fact is an XSLT-based transformation between two XML-documents. In this respect, the approach does not reach a high level of abstraction, but calls for very basic transformation procedures. Further, the article does not encompass a visual language suitable for expressing the aspects of relative positioning, as the application concerns petri-nets and their representation as nodes-and-edges.

Kruse et al. present in [8] a component-oriented tool for supporting EA management. Central to their approach is a strong ’componentization’ in the way, that different types of viewpoints are implemented as different components of the tool. Each viewpoint in this respect brings along a dedicated information model, that describes the information necessary for creating the corresponding visualization. A company willing to use the corresponding tool for visualizing their EA or parts thereof, selects the appropriate visualization components and supplies a model-to-model transformation transforming parts of the organization-specific EA information model to the information model associated with a specific viewpoint. Put in other words, the approach of Kruse transforms and projects instances of one information model to instances of a different information model, which conversely is directly fed into a layout and visualization mechanism.

Multiple commercial tools provide support for EA management. Matthes et al. analyzed in [10] nine prominent of these tools, coming to the result that most of the tools fall for two types of visualization-related problems. On the one hand, the *process-* or *methodology-driven* tools bring along a fixed EA information model that underlies a set of predefined viewpoints. Visualizations corresponding to these viewpoints can mostly be generated automatically, and may use a rich visual language, including relative positioning of symbols to convey information. On the other hand, *metamodel-driven* tools can flexibly adapt their information model to the specific needs of the using organization, but are mostly limited to visualizations of the nodes-and-edges type. More complex visualizations, using e.g. relative positioning, have to be programmed in these tools utilizing scripting languages with no visualization- and layout-specific concepts.

4 Critical reflection and outlook

This article presented a technique to create visualizations (views) from arbitrary EA descriptions. The generated views thereby are consistent with a previously defined viewpoint and are created using a model-to-model transformation. Complementing the presented technique also a prototypic tool implementing the technique was described. This tool has been used in different practice cases in the past and showed the applicability of the technique on various different organization-specific EA information models. Different languages were utilized to realize the model-to-model transformation, namely the basic programming language java (cf. [12]) as well as the model transformation language ATL (cf. [13]). While both languages were sufficient to generate visualizations, they provide a rather low level of abstraction, when it comes to describing architectural viewpoints. More precisely, the complexity of the model transformations expressed in these languages is often beyond the level, that an enterprise architect can cope with.

Increasing the level of abstraction in defining viewpoints and thereby facilitating the creation of end-user defined viewpoints are the challenges to be addressed next. Two different strategies to achieve this can be pursued:

- Rise the level of abstraction in the visualization model, i.e. replace the fine grained visualization rules with more coarse ones. As an example, one could think of a *cluster*-rule for visualizations like the one in Figure 1.
- Rise the level of abstraction in the transformation language, i.e. provide domain specific concepts for specifying a syca transformation going beyond basic *query model*- and *transform model*-concepts.

While both ideas may be useful to achieve the goals, especially the latter one seems more appealing. In consequent continuation of the prefabrics of Ramacher (cf. [12]), a tailored transformation language could allow to stay with the basic visualization model concepts that have a clear and unambiguous semantics.

Acknowledgements

The development of the herein presented technique and prototypic tool is supported and partially funded by Siemens IT Solutions and Services (SIS).

References

1. S. Aier, S. Kurpjuweit, C. Riege, and J. Saat. Stakeholderorientierte dokumentation und analyse der unternehmensarchitektur. In H.-G. Hegering, A. Lehmann, H. J. Ohlbach, and C. Scheideler, editors, *GI Jahrestagung (2)*, volume 134 of *LNI*, pages 559–565, Bonn, Germany, 2008. Gesellschaft für Informatik.
2. ATLAS group at LINA & INRIA. Atl: Atlas transformation language, 2006.
3. S. Buckl. *Modell-basierte Transformationen von Informationsmodellen zum Management von Anwendungslandschaften.* Diploma thesis, Fakultät für Informatik, Technische Universität München, 2005.
4. S. Buckl, A. M. Ernst, J. Lankes, K. Schneider, and C. M. Schweda. A pattern based approach for constructing enterprise architecture management information models. In *Wirtschaftsinformatik 2007*, pages 145–162, Karlsruhe, Germany, 2007. Universitätsverlag Karlsruhe.
5. P. Domokos and Varró, Dániel. An open visualization framework for metamodel-based modeling languages. *Electronic Notes in Theoretical Computer Science*, 72(2), 2002.
6. A. M. Ernst, J. Lankes, C. M. Schweda, and A. Wittenburg. Using model transformation for generating visualizations from repository contents – an application to software cartography. Technical report, Technische Universität München, Chair for Informatics 19 (sebis), Munich, Germany, 2006.
7. International Organization for Standardization. Iso/iec 42010:2007 systems and software engineering – recommended practice for architectural description of software-intensive systems, 2007.
8. S. Kruse, J. S. Addicks, M. Postina, and U. Steffens. Decoupling models and visualisations for practical ea tooling. In *Pre-Proceedings of the 4^{th} Workshop on Trends in Enterprise Architecture Research, November 23^{rd}, 2009, Stockholm, Sweden*, pages 85–98), Stockholm, Sweden, 2009.
9. M. Lankhorst. *Enterprise Architecture at Work: Modelling, Communication and Analysis.* Springer, Berlin, Heidelberg, Germany, 2005.
10. F. Matthes, S. Buckl, J. Leitel, and C. M. Schweda. *Enterprise Architecture Management Tool Survey 2008.* Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany, 2008.
11. OMG. Meta object facility (mof) core specification, version 2.0 (formal/06-01-01), 2006.
12. R. Ramacher. *Entwurf und Realisierung einer Viewpoint Definition Language (VDL) für die Systemkartographie.* Diplomarbeit, Fakultät für Informatik, Technische Universität München, 2009.
13. J. Wiegelmann. *Analysis and Application of Model Transformation Languages for Generating Software Maps.* Bachelor thesis, Fakultät für Informatik, Technische Universität München, 2008.

Tool-based Support for Organization-specific Enterprise Architecture Management

Sabine Buckl, Thomas Dierl, Florian Matthes, and Christian M. Schweda

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

Enterprise architecture (EA) management is a commonly accepted instrument for modern organizations to deal with today's challenging environment. Effectively designed an organization-specific EA management function can improve the overall agility of an organization. The design of such management function is a challenging task, in which the different process steps, information flows, and roles that constitute such function have to be shaped and aligned. The complexity of the management subject EA and the high number of involved stakeholders further aggravate the creation of a consistent but organization-specific EA management, and call for tool support during the *design phase*. In this phase, the designer selects the goals to be pursued, the concerns to be addressed, and the roles to be involved. Based on this selection, the user is supplied with re-usable and practice-proven building blocks that can be integrated into a tailored EA management function for the using organization. Different sources for such building blocks exist, namely the EA management pattern catalog [1] or TOGAF [5], although the latter does not explicitly state such blocks.

After the organization-specific EA management function has been defined, it must be enacted in the organization. For the *conduction phase*, i.e. for managing the EA, the tool must provide support by initializing the corresponding processes and process steps, informing the relevant stakeholders, and ensuring that their information demands are fulfilled. For the latter, the tool must support the generation of EA views that correspond to well-defined viewpoints [1, 3]. These viewpoints prescribe which architectural information is conveyed to which stakeholder. Building on the prefabricates of [2], where a tool for flexibly visualizing EAs is presented, we subsequently outline the core idea behind an EA management design tool that further allows to enact the designed process, sketch realization ideas, and give an outlook on future developments.

Approach Central idea behind the design tool is the understanding that re-usable building blocks for an EA management function may be extracted from different EA management approaches in literature as well as observed in practical cases. These building blocks are then aligned to a common terminology and their underlying organizational contexts as well as their pursued management goals are elicited. Thereby, the different possibly competing building blocks are interlinked into a *nexus* (cf. Figure 1) that backs the design tool.

Implementation While EA management is a collaborative function, the design phase can be considered a single user task. The design tool should hence be implemented as standalone application, e.g. based on the Eclipse Rich Client Platform [4]. Aforementioned platform may especially be useful, as it supports the development of graphical modeling tools via the graphical modeling framework (GMF) and the eclipse modeling framework (EMF).

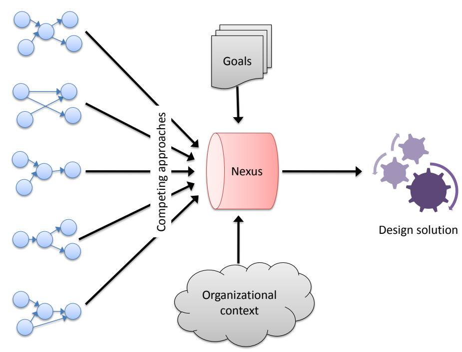


Fig. 1. Building block nexus backing the design tool

Outlook Building block-based modeling of EA management processes is in many ways different from typical process modeling. The above tool can therefore greatly benefit from using a modeling language appropriate for this purpose, i.e. a language especially supporting the composition of process building blocks. Further challenges arise from integrating the process with other building blocks, e.g. the viewpoints. In each integration, consistency must be verified and minor user-specific adaptations should be supported.

References

1. S. Buckl, A. M. Ernst, J. Lankes, and F. Matthes. Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008). Technical report, Chair for Informatics 19 (sebis), Technische Universität München, Munich, Germany, 2008.
2. S. Buckl, A. M. Ernst, J. Lankes, F. Matthes, C. Schweda, and A. Wittenburg. Generating visualizations of enterprise architectures using model transformation (extended version). *Enterprise Modelling and Information Systems Architectures – An International Journal*, 2(2):3–13, 2007.
3. S. Kurpjuweit and R. Winter. Viewpoint-based meta model engineering. In M. Reichert, S. Strecker, and K. Turowski, editors, *Enterprise Modelling and Information Systems Architectures – Concepts and Applications , Proceedings of the 2nd International Workshop on Enterprise Modelling and Information Systems Architectures (EMISA'07), St. Goar, Germany, October 8-9, 2007*, LNI, pages 143–161, Bonn, Germany, 2007. Gesellschaft für Informatik.
4. The Eclipse Foundation. Rich client platform. http://wiki.eclipse.org/index.php/Rich_ Client_ Platform (cited 2010-03-18), 2010.
5. The Open Group. TOGAF "Enterprise Edition" Version 9. `http://www.togaf.org` (cited 2010-02-25), 2009.

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 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, Data Quality

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. 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.

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 data quality, the property of whether the data objects are accurate with respect to the real world they describe is of interest. This property however, is irrelevant for a number of other analyses, such as performance evaluation.

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. data quality. 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 have 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 since this is necessary to make rational decisions about information systems [7]. 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 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 maintainance of the power grid. The introduction of an maintenance management system would improve the quality of the maintenance process and allow more cost efficient grid maintenance. The question for the decision maker is whether this change is feasible or not.

As mentioned, in the assessment scoping step, 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 data quality of an information system. Often several quality attributes are desirable goals.

During the next step, to identify the best alternative, the scenarios need to be detailed to facilitate analysis of them. 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 more precise data object attribute would increase the probability that the data quality of an information system is high. The impact of a certain data object attribute is thus one factor that can affect the data quality 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.

When the decision alternatives are detailed, they can be analyzed with respect to the desirable system quality. The pros and cons of the scenarios are then traded against each other in order to determine which alternative ought to be preferred.

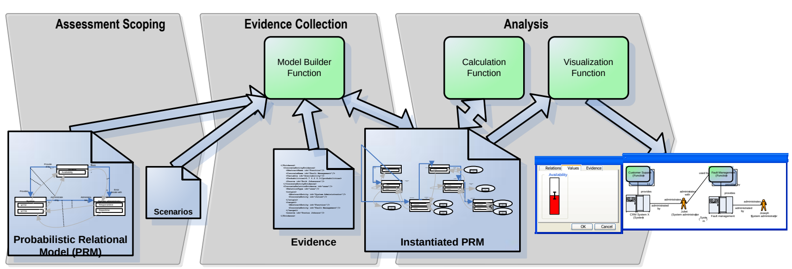


Fig. 1. The process of enterprise architecture analysis with three main activities: (i) setting the goal, (ii) collecting evidence and (iii) performing the analysis.

2.1 Probabilistic Relational Models

A *probabilistic relational model* (PRM) [13] 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 just 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 further 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 further 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 based on a Model-View-Controller architecture. The data model for PRMs and instantiated PRMs, is specified in XSD and stored in XML files. The user interface is build upon the NetBeans Visual Library [14] with usage of the JApplication framework. The tool is separated into two units, one supporting the modeling of the PRM, whereas 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 the focus of the analysis is set, as the defined classes reflect the domain of interest. The second component of the EAT allows the instantiation of the PRM. Thereby 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 gets translated into a Bayesian network understandable by the Smile library [15]. 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.

4 Example Tool Application

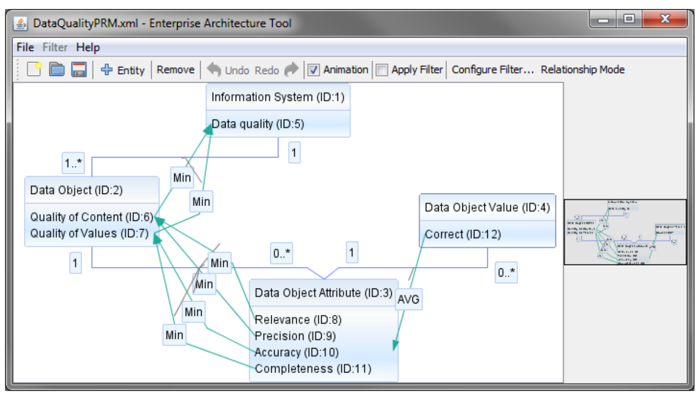


Fig. 2. The PRM for data quality analysis showing classes and attributes relevant for the analysis.

This section will illustrate the application of the tool through a case study performed at an electric grid operator in Sweden [16]. The case study focused on assessing data quality in a maintenance management system. In this paper a reduced version of the study is presented, and in the correspondingly reduced theory *data quality* is defined as the *quality of the content* and the *quality of the values* for the data objects used by the system. Quality of content is defined in terms of *relevance*, the degree to which the information objects have a purpose for the users, and *precision*, i.e. that the information objects are detailed enough for the application using them. Turning to *quality of values*, this is defined in terms of *accuracy* and *completeness* of the data objects, or more precisely the attributes contained within a data object. Accuracy is measured as the degree to which the values found in a system correspond to the actual values they represent, whereas completeness is measured as the amount of values stored in the system compared to the domain they represent.

Based on the theory outlined above the PRM for data quality analysis can be constructed. The PRM has four classes, firstly the *Information System* for which the assessment is performed. This system uses a set of *Data Objects* that constitute the abstract information model employed by the system. Each object contains one or more *Data Object Attributes* and in the operational system these attributes are instantiated to *Data Object Values*. Turning to the attributes of the PRM, each of the concepts defined in the theory, e.g. relevance and completeness, corresponds to one attribute. This can be seen in Fig. 2, for instance the attribute *Data quality* is associated to the class *Information System* and depends on the attributes *quality of values* and *quality of content* of the class *Data Object*. The conditional probabilistic dependencies between attributes were also defined, for details, see [16].

The data in this case study was collected primarily through interviews but for the acurracy of the data objects direct observations were made and compared with values in the system, i.e comparing the data object values with the actual items they represent to assess the accuracy of an attribute. The instantiated PRM is shown in Fig. 3 where the maintenance system, two data objects, three attributes and a set of values of these attributes are modeled. Based on (i) the model and (ii) the values of the descriptive attributes in the PRM (as found during the data collection), the assessment can be performed. The data quality of the maintenance system, measured in percent of complete fulfilment of data quality requirements, was found to be 62 percent.

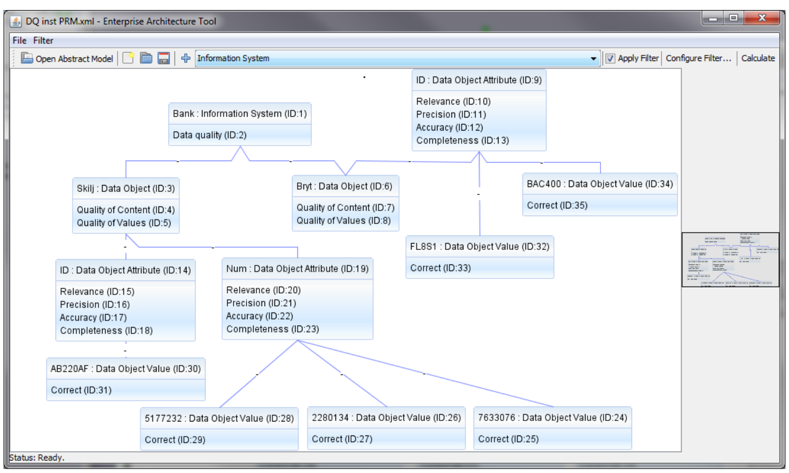


Fig. 3. Instantiated PRM for data quality analysis containing one system, two data objects and three attributes with values attached.

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 [17] 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 [18], 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) [19] 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 used formalism 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 graspable 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 [17]. 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 consist of two separate parts, one for defining analysis theory and one 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 data quality assessment was outlined, but the tool supports the analysis of various quality attributes, such as maintainability, information security, 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 (August 2006)
2. Winter, R., Fischer, R.: Essential layers, artifacts, and dependencies of enterprise architecture. Journal of Enterprise Architecture **Volume 3, Number 2** (2007) 7–18
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) (1987) 276–292
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. `http://www.troux.com/products/` (March 2010)
9. IBM: System Architect. `http://www-01.ibm.com/software/awdtools/systemarchitect/productline/` (March 2010)
10. Scheer, A.: Business process engineering: Reference models for industrial enterprises. Springer-Verlag New York, Inc. Secaucus, NJ, USA (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 (2007)
12. Kurpjuweit, S., Winter, R.: Viewpoint-based meta model engineering. In: Enterprise Modelling and Information Systems Architectures (EMISA 2007) (2007)
13. Friedman, N., Getoor, L., Koller, D., Pfeffer, A.: Learning probabilistic relational models. In: Proc. of the 16th International Joint Conference on Artificial Intelligence, Morgan Kaufman (1999) 1300–1309
14. NetBeans: NetBeans Visual Library. `http://graph.netbeans.org` (March 2010)
15. Decision Systems Laboratory of the University of Pittsburgh: SMILE. `http://genie.sis.pitt.edu/` (March 2010)
16. Magnusson, S., Udin, S.: Assessing data quality on the maintenance sytem at svenska kraftnät. Technical report, Industrial Information and Control Systems, KTH (2009)
17. Lankhorst, M.: Enterprise architecture at work : modelling, communication, and analysis. Springer (2005)
18. Lauritzen, S.: Propagation of probabilities, means, and variances in mixed graphical association models. Journal of the American Statistical Association **87**(420) (1992) 1098–1108
19. Object Management Group: Object Constraint Language specification, version 2.0 formal/06-05-01. Technical report (2006)

MEMOCenterNG – A full-featured modeling environment for organization modeling and model-driven software development

Jens Gulden and Prof. Dr. Ulrich Frank

University Duisburg-Essen,
Universitaetsstr. 9, 45141 Essen, Germany
{jens.gulden,ulrich.frank}@uni-duisburg-essen.de
http://www.wi-inf.uni-duisburg-essen.de/FGFrank/

Abstract. We present MEMOCenterNG[1], an integrated, full-featured modeling environment containing 11 built-in modeling languages together with meta-modeling support for creating new languages and corresponding diagram editors. Tools for analysis, model transformation and code generation are included.

Key words: Modeling, Model-driven software development, Multi-perspective modeling, Enterprise modeling

1 A comprehensive multi-language modeling tool

Large modeling projects typically use multiple modeling languages simultaneously to express a variety of aspects of a modeled system. This holds true in diverse types of modeling projects, especially in business process modeling, which unites conceptual business aspects with technical realizations. Model-driven software development benefits from multiple interrelated modeling perspectives to gain a coherent and as complete view as possible on the system to be developed.

To foster large modeling projects, an integrated modeling environment tool is desirable to reduce efforts in combining multiple model editors operating on shared model content. Semantic integrity is an indispensable requirement for multi-perspective modeling [6], that means, when different model editors reference the same concept (e.g., a person displayed both in an organization structure diagram and in a process diagram), the model editor components need to share common information about its unique identity. The need for semantic integrity requires model editors to work on top of a common set of meta-concepts [7] which ensures semantically valid relations among model data.

With MEMOCenterNG, we present a comprehensive modeling environment which integrates an extensible set of modeling languages on the basis of

[1] MEMOCenterNG is named after the MEMO enterprise modeling method [6]. "NG" stands for "Next Generation", MEMOCenterNG is the successor to an earlier MEMOCenter prototype application.

a consolidated common meta-model API. The provided modeling languages are suitable to express knowledge about a system on three different levels of abstraction from multiple interdependent perspectives.

The available levels of abstraction provided by built-in modeling languages in MEMOCenterNG are:

1. A meta-modeling layer which allows for creating new modeling languages and corresponding diagram editors, and also internally defines other built-in modeling lnaguages.
2. A set of built-in modeling languages for modeling the organizational environment of involved actors, their goals, behaviour and involved resources. This organizational abstraction layer is provided by domain-specific modeling languages of the MEMO modeling method [6].
3. An implementation model layer which offers general purpose modeling languages to express a software system's inner perspective in terms of, e.g., classes, attributes and relationships.

Figure 1 gives an overview on the component architecture of MEMOCenterNG, as it evolves from applying the MEMO Meta Modeling Language (MML) for language creation.

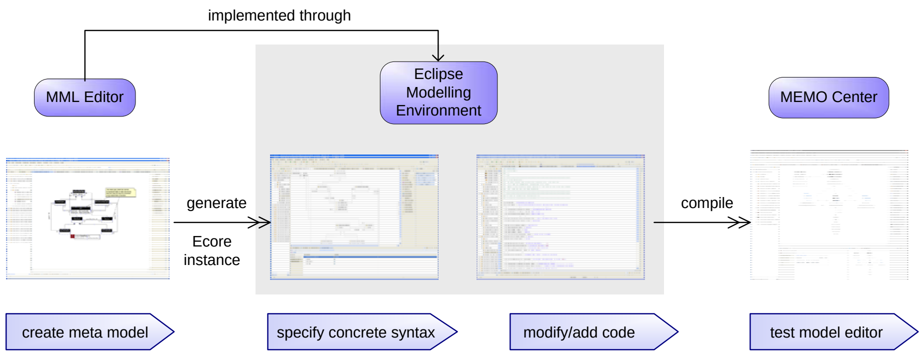


Fig. 1. MEMOCenterNG component architecture

MEMOCenterNG contains a total of 11 modeling languages on all three abstraction layers. This set is composed of 7 domain specific languages as proposed by the MEMO method [6], made up of *Activity Diagrams*, *Allocation Diagrams*, *Organization Diagrams*, *Process Control Flow Diagrams*, *Process Decomposition Diagrams*, *Resource Diagrams* and *Strategy Diagrams*. 3 implementation-level generic purpose modeling languages covering *Data Flow Diagrams (DFD)*, *Entity-Relationship-Models (ERM)* and class-diagrams of the *Unified Modeling Language (UML)*, and the *MEMO Meta-Modeling Language (MML)*. These initially available languages can be enhanced by any number of specific modeling languages created with the MML. Supplemented by third-party tooling components, models can be analyzed, model-transformations can be carried out, and code-generation mechanisms are available to generatively create software in a model-driven development process. Altogether, MEMOCenterNG forms a comprehensive environment for modeling on multiple layers of abstraction from multiple perspectives, store and manage interrelated models in a common environment, and further process model data inside the same platform. In model-driven software development projects, generated artefacts can furthermore be edited, compiled and packaged within the same tool. The platform is based on the Eclipse [4] environment which can additionally be enhanced by a multitude of third-party supplementary components for software development.

By means of the included MEMO Meta Modeling Language (MML, [7]), new modeling languages can efficiently be created, and appropriate diagram editors for using the languages are automatically created from MML meta-models.

Since all generated components run in the same environment as MEMOCenterNG, models and generated software components may even reflectively refer to MEMOCenterNG's models. This allows models to be integrated into software components at runtime as part of a self-referential information system architecture [8].

2 Organization modeling languages

To express the outer context of an incident to be modeled, organization modeling languages capture people's goals, their behavior, organizational structure and resources of the modeled context. Such types of models play an important role in enterprise modeling (EM, [6]), to express types of business processes that are performed in an organization. Besides business contexts in a narrow sense, any organizational setting and projects with shared goals among groups of people can generically be expressed with the semantic modeling concepts provided by these modeling languages.

MEMOCenterNG comes pre-packaged with interrelated modeling languages of the MEMO OrgML [6] that cover organization modeling. These are in the first place the *Organization Diagram* language for modeling organizational structure, and the *Process Control Flow Language* [6], which allow to express semantically rich process model descriptions of business processes and other methodical procedures in organizations. The Process Control Flow language is enhanced by the *Process Decomposition Language* which allows for specifying static decomposition relationships among process steps, i.e., express which process steps are made out of others. Finally, the *Strategy Diagram* and *Activity Diagram* languages for expressing strategy, goals and actions from a high-level strategic view are part of the environment.

To model physical and non-tangible resources in business contexts, the *ResML* [10] is included in the set of modeling languages, accompanied by the *Allocation Diagram* language which is responsible to express the mappings between process steps and resources.

Figure 2 shows an example Process Control Flow model edited in MEMO-CenterNG.

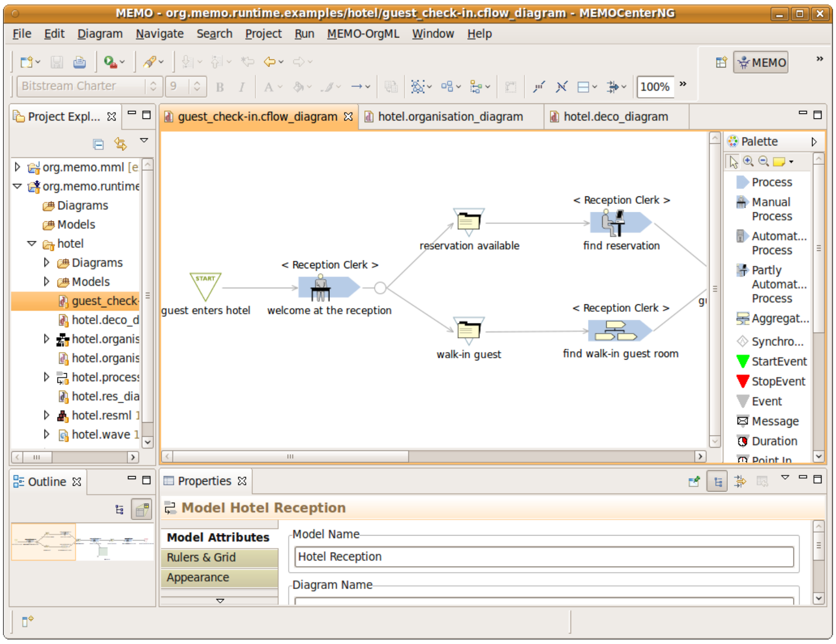


Fig. 2. Example *MEMO Process Control Flow* model

3 Included meta-modeling language

The *MEMO Meta Modeling Language* (MML, [7]) included in MEMOCenterNG has especially been designed to efficiently enhance existing languages, and to automatically generate deployable diagram editors from meta-model descriptions of modeling languages. The MML puts special focus on interlinking multiple modeling languages. Concepts from other, previously existing MML models, can be referenced. Every concept is classified by a unique graphical symbol which indicates to which language it belongs.

Full-featured diagram editors can automatically be generated by a single mouse-click from MML models within the user-interface of MEMOCenterNG's MML editor. These generated diagram editors are fully downwards-compatible to the Eclipse Modeling Framework (EMF, [13]) and Graphical Modeling Framework (GMF, [9]) components, which allows for applying any additional EMF / GMF technology component to MML-specified model editors and corresponding model instances, and also use model-instances for analysis, transformation and code-generation (see sect. 5).

Figure 3 gives an example of an MML model edited in MEMOCenterNG.

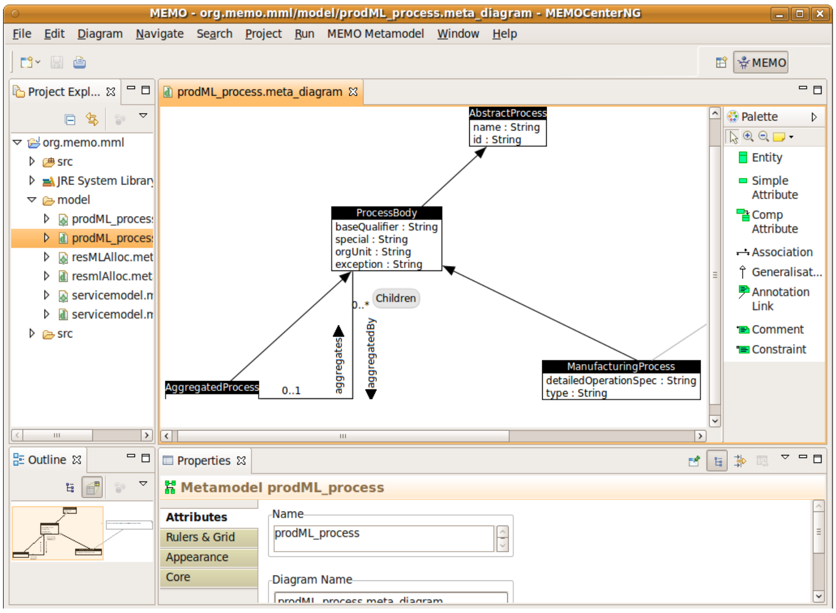


Fig. 3. Example of an MML model edited in MEMOCenterNG

4 Implementation-level modeling languages

In order to take a traditional modeling view on a low abstraction level, MEMOCenterNG contains three classical formal systems modeling languages, *Entity Relationship Models* (ERM, [2]), *Data Flow Diagrams* (DFD, [3]) and object oriented class diagrams from the *Unified Modeling Language* (UML, [1]). Selected concepts of these languages can be referenced from elements in organization models to trace implementation details from a high real-world abstraction level down to technical details.

Each of the implementation-level modeling languages comes with pre-packaged analysis and code-generation functionality. The ERM model editor allows to generate a relational database schema from ERM models as a sequence of executable SQL data declaration statements, which subsequently may be executed from inside the Eclipse platform to deploy the initial database. The DFD editor comes with basic analysis capabilities, and from UML class diagrams, the source code for Java classes can be generated.

Together with the software development features of the underlying Eclipse [4] platform, MEMOCenterNG forms a fully integrated model-driven software development environment. An example of integrating between an organization model and an implementation model is displayed in figure 4.

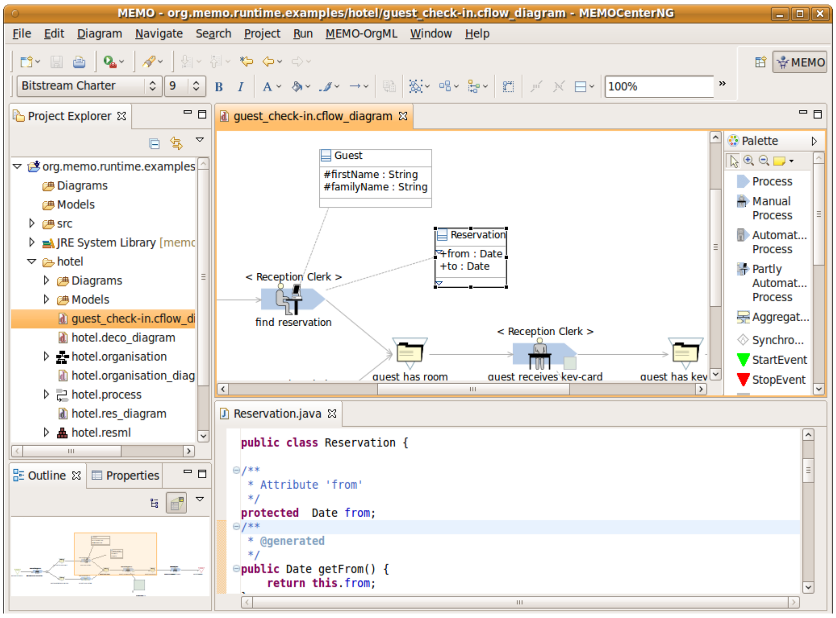


Fig. 4. Integrating between an organization model and an implementation model

5 Supplementary Tools for analysis, model-transformation and code-generation

All components of MEMOCenterNG are based on the Eclipse Modeling Project [9] components EMF and GMF, and make use of the Ecore language [5] through the MML. As a consequence, any supplementary component that exists for the Eclipse Modeling Project can be applied in MEMOCenterNG, too. This allows for seamlessly integrating supporting technology such as Xtend / Xpand [5], and other specific code generation languages such as Velocity [12], or the QVT [11] model transformation language. By default, the Eclipse components for Xtend / Xpand transformations are included in the MEMOCenterNG environment.

6 Availability

A beta-version of MEMOCenterNG is available for download at `http://www.wi-inf.uni-duisburg-essen.de/FGFrank/download/memo/`. Please request password information from the authors.

7 Conclusion and Outlook

We have presented a modeling tool that offers multiple modeling languages in an integrated environment, based on a meta-model supported language architecture and enriched by an easy-to-use meta-model editor for specific language enhancements.

A common language architecture ensures the semantic integration of concepts across multiple languages. By incorporating meta modeling and the creation of new modeling languages into the feature spectrum of the modeling tool, a new degree of flexibility and adaptability to future requirements part of the application. This makes MEMOCenterNG a comprehensive, full-featured integrated modeling environment for multiple types of modeling projects, including model-driven software development approaches. Currently, the tool is successfully used for teaching purposes and in medium-sized business projects carried out in co-operation with our research group.

In further research work, we intend to develop and integrate additional modeling languages, e.g., for designing indicator systems that are integrated with models of business processes and IT resources. We also plan to extend the framework with elaborate support for method engineering by integrating model editors with corresponding process editors.

References

1. Booch, G., Jacobson, I., Rumbaugh, J.: The Unified Modeling Language Reference Manual. Addison-Wesley, Reading (1999)
2. Chen, P.: The Entity-Relationship Model – Toward a Unified View of Data. In: ACM Transactions on Database Systems 1/1/1976 ACM-Press ISSN 0362-5915, pp. 9–36 (1976)
3. DeMarco, T.: Structured Analysis and System Specification. Prentice Hall, Upper Saddle River (1979)
4. Eclipse Foundation: Eclipse Platform. `http://www.eclipse.org/` (2010-04-25)
5. Efftinge, S., Friese, P., Haase, A. and others: openArchitectureWare User Guide. `http://www.openarchitectureware.org/pub/documentation/4.3.1/html/contents/index.html` (2008)
6. Frank, U., MEMO: A Tool Supported Methodology for Analyzing and (Re-)Designing Business Information Systems. In: Ege, R., Singh, M., Meyer, B. (eds.): Technology of Object-Oriented Languages and Systems, pp. 367–380 (1994)
7. Frank, U.: The MEMO Meta Modeling Language (MML) and Language Architecture. ICB-Research Report No. 24, Institute for Computer Science and Business (ICB), University Duisburg-Essen (2008)
8. Ulrich Frank, U., Strecker, S.: Beyond ERP Systems: An Outline of Self-Referential Enterprise Systems. ICB-Research Report No. 31, Institute for Computer Science and Business (ICB), University Duisburg-Essen (2009)
9. Gronback, R. C.: Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit. Addison-Wesley Longman, Amsterdam (2009)
10. Jung, J.: Entwurf einer Sprache für die Modellierung von Ressourcen im Kontext der Geschftsprozessmodellierung. Logos, Berlin (2007)

11. Object Management Group: Meta Object Facility Query / View / Transformations. `http://www.omg.org/spec/QVT/1.0` (2008)
12. Apache Software Foundation: The Apache Velocity Project. `http://velocity.apache.org/` (2010-04-25)
13. Steinberg, D., Budinsky, F., Paternostro, M., Merks, E.: EMF: Eclipse Modeling Framework (2nd edition). Addison-Wesley Longman, Amsterdam (2009)

Lightweight Approach For 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. Enterprise architecture (EA) management is a challenging task, modern enterprises have to face. This task is often addressed via organization-specific methodologies, which are implemented or derived from a respective EA management tool, or are at least partially aligned. Nevertheless, especially when starting an EA management endeavor, the documentation of the EA is often not likely to satisfy the level of formalization, which is needed to employ an EA management tool. This paper address the issue of starting EA management, more precise EA documentation and analysis, by utilizing a wiki-based approach.

Today's enterprise wiki-systems provide numerous of services such as collaborative authoring, tagging, bookmarking, awareness, commenting, rating, linking, search, social networking, versioning, access control [1]. Furthermore, recent efforts aim to allow annotating semantic data within the wiki pages' content in enterprise wiki-Systems. How these functions can be applied for collaborative enterprise architecture (EA) modeling and documentation is explained by means of the following scenario. We illustrate how to model and describe the concepts shown in Figure 1 and corresponding instances in a wiki-based manner.

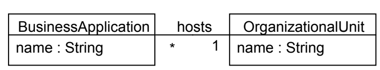


Fig. 1. UML diagram of the concepts business application and organizational unit

In order to model concepts and attributes (top-down) we use an open templating mechanism [2]. Thereby two wiki-pages describing the concepts business application (BA) (cf. Figure 1) and organizational unit (OU) with their name attributes are defined. To mark the wiki-page as *template* reserved tags are utilized, e.g. typedef. Since the templates are wiki pages, the concept BA and its attributes can be described textual in the wiki pages' content. By using semantic annotations [3] a "name" attribute can be defined by marking the literal "name" within the text. All attribute-definitions could be shown in a tabular view for purpose of exposing semi-structured parts of the wiki-page. In some cases it can be beneficial to define mandatory attributes for a template, e.g. all BAs must provide a name attribute. This could be achieved by specifying rules on template level. In contrast to templates, concepts can also evolve dynamically on "instantiation" level (bottom-up). Thereby, a wiki-page is created to describe a concrete instance of a BA (cf. Figure 1). To mark the page to be a BA the page can either be tagged "business application" on page level or the literal "business application" can be marked with semantic annotation within the text, e.g. "is-a". In the latter case a new template (providing all currently in the text defined attributes) could be created if no BA template exists so far. If a template is already defined, the assigned attributes (having no values) could be shown in a tabular view. Furthermore, additional attributes, which are frequently used in other BA wiki-pages could be recommended by showing them in a tabular view.

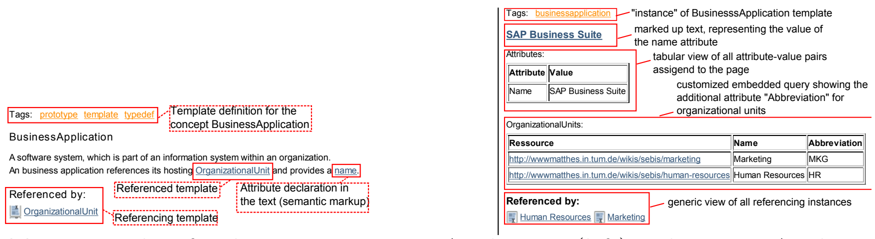


Table 1. Template for the concept BusinessApplication (left) and BusinessApplication instance "SAP Business" (right)

Association between concepts can be expressed via hyperlinks to other wiki-page instances representing an OU, i.e. which are tagged "organizational unit". The latter wiki-page could list all pages (as links) it is referenced by, by means of backlinks, i.e. links to all BAs the OU is using. This could either be achieved in a generic manner or by embedding custom queries in the page content. In both cases it might by useful to customize which attributes of the referencing concept to be shown. In contrast to define links on concrete (BA) instances (bottom-up), hyperlinks could also be used on template level to express directional relations between concepts (top-down). E.g., the literal "OrganizationalUnit" shown in Figure 1(left) could be a hyperlink to the OU-template. To express that exactly one OU is referenced by a link on instance level rules could be applied (cf. mandatory attributes).

Summary: Both ways of lightweight EA modeling can be applied, top down by using predefined wiki-templates as well as bottom up by evolving the model indirectly on the instance level. Wiki-pages can be utilized for documentation of EA concepts and their instances collaboratively. Besides that, other services of today's wiki-systems can be utilized, e.g lists and change feeds to get notified when the model or the documentation changes.

References

1. T. Büchner, F. Matthes, and C. Neubert. A concept and service based analysis of commercial and open source enterprise 2.0 tools. In *KMIS*, pages 37–45, 2009.
2. S. Buckl, F. Matthes, C. Neubert, and C. M. Schweda. A wiki-based approach to enterprise architecture documentation and analysis. In *The 17^{th} European Conference on Information Systems (ECIS) – Information Systems in a Globalizing World: Challenges, Ethics and Practices, 8. – 10. June 2009, Verona, Italy*, pages 2192–2204, Verona, Italy, 2009.
3. The Dublin Core Metadata Initiative. The dublin core metadata initiative. http://dublincore.org/ (cited 2010-03-18), 2010.

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

Nelly Schuster[1], Raffael Stein[2], 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 and 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. This 'living' format supports teams to rapidly regulate and control collaborative activities by mapping them to services provided by team members or enterprise systems and mashing them into document structure and 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). While several enterprise document engineering approaches exist for structured and recurring business processes [1,2], the engineering of documents which evolve during ad hoc collaboration between members of a possibly virtual team still lacks appropriate tool support. Such documents are subject to ad hoc changes as new contents from different sources – humans and software systems – come in. Dependencies between different parts of a document require coordination mechanisms and ad hoc process support 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 like bugs or are collaboratively developing solutions which are captured in interrelated documents [3]. Another example is the collaborative development of a research publication [4]. Various technologies and tools support either collaborative creation and evolution of documents (e.g. CSCW technologies, wikis or other Web-based collaborative writing applications [5]) or coordination of structured processes (e.g. BPM technologies like WfMS). However, they largely fail to provide effective support for collaborative document engineering which involves ad hoc processes.

In order to address this need, we have developed the MoSaiC approach for service-oriented 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 [6]. 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 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 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 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. Document mashups build on a hierarchical structure that represents the logical decomposition of a collaboration goal into tasks. 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 like chapters, paragraphs or figures. 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 service providers (i.e. collaborators) in an evolving document mashup builds on events and rules: each service exposes events indicating state transitions of its underlying resources (i.e. activity results) which are consumed and reacted upon by document mashups. An important part of document engineering is to define ECA 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 recall 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 service 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. Rules of a document mashup will immediately be executed when the event they listen for is emitted. This might lead to further activities like service calls.

During the course of the collaboration, document content and structure transform from a record of collaborative conversation to a meaningful result. 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 effects the control of a collaboration. Still, in order to enable the design of critical document parts, mashups can be set 'asleep'. This means, that the control of document services is idle until the mashup is 'woken up' again.

3 MoSaiC Tool and Platform Architecture

In order to realize our concept of document service mashups we have developed a Web-based platform and RIA to support collaborative document engineering. Figure 1 gives an overview of the architecture.

Users access the document collaboration environment through a graphical user interface that runs on their Web browser. The UI communicates with the application and persistence logic of the MoSaiC document mashup manager which might run on an enterprise application server or in the cloud. The GUI uses AJAX calls to communicate with the server components. The collaboration system uses a communication infrastructure for event-based interaction between services in document mashups. This infrastructure – the document service bus – might be deployed independently and is accessed through RESTful service interfaces.

The Web-based document collaboration tool integrates various graphical drag-and-drop user interface components supporting mashup authoring, service provisioning and participant management use cases:

- Fundamentally, the *user management UI* allows logging into the system or registering as a new user. Subsequently, the *participant UI* 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

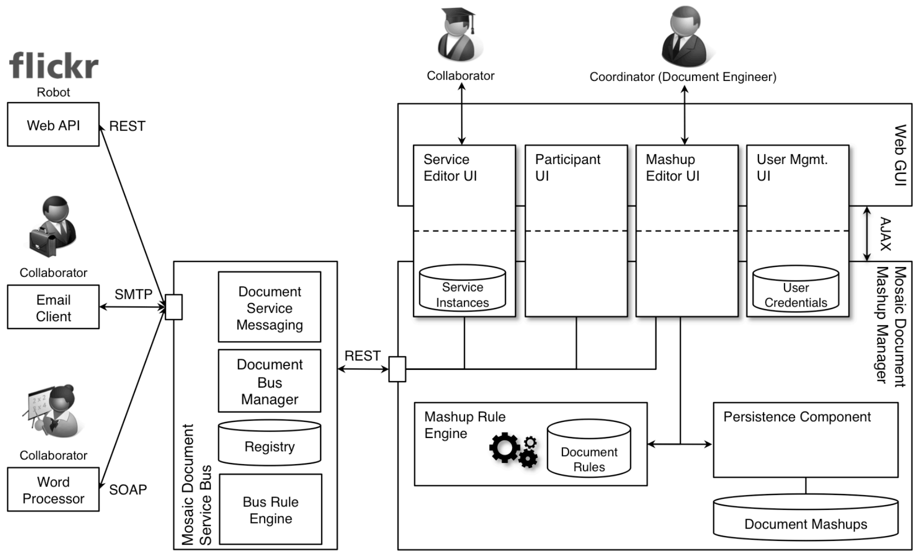


Figure 1: Architectural Overview of the MoSaiC Mashup Tool and Platform

also include the responsibility of a collaborator to carry out certain activities within a collaboration and provide associated document services manually. As a variant, users might delegate responsibility to a robot that provides document services by means of automated Web services (e.g. to integrate contents from Flickr).

- Using the *mashup editor UI*, coordinators carry out the main document engineering tasks including design and management of collaborative document mashups. Management includes administrating a repository of living documents. It also includes 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, collaboration and coordination. 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. In the collaboration perspective, the UI shows lists of content and transformation services as well as possible providers. The coordinator can drag and drop the services from this list on the canvas or assign providers to elements. The coordination perspective allows definition and management of rules expressing causal and temporal constraints as well as manual control of actions.
- In order to support collaborators of a document mashup in providing contributions through service calls, the platform offers a *service editor UI*. This UI allows users to receive service calls coming from a document mashup manager (the same or any other), edit and persist appropriate content, e.g. the introduction of a research paper, and communicate it back to the document mashup manager. However, there are also other ways for collaborators to connect like an email bridge or plug-ins for common word processors that enable them to handle document service calls.

Mashup logic is enforced by the mashup manager. Its *persistence component* stores all mashups including structure, tasks, rules, services, participants, roles, contents and control state. The persistence component listens to mashup and service update events and stores the information accordingly. The UI components learn about updates of a mashup through polling the persistence component.

A *mashup rule engine* drives automatic enforcement of mashup logic. It is able to process complex events and 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.

A *document service bus* enables service interaction. Fundamentally, the bus maintains a *registry,* which stores information about existing mashups, document services and providers. The *document service messaging* component routes and queues all 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. Furthermore, 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 mashup manager if possible. An example is the automatic routing of certain text content through a translation service before delivering it to the mashup manager.

4 Scenario and Use Case

Scientific publication involves collaboration of various researchers who provide different parts of a paper like texts, pictures and references. Furthermore the collaborators need to coordinate in order to discuss the concepts, structure the document, provide interrelated contents and proof-read the results thereby checking for completeness, correctness and readability.

In our scenario, Alice, a scientist, is creating a research paper mashup and starts with defining the structure of the document in the mashup editor. She drags several document elements for different parts, texts and figures from a list to the mashup canvas. Furthermore, she adds elements for an abstract and bibliography. For each element Alice provides a meaningful task description and whenever she drops an element on the canvas, the change in the mashup structure is immediately persisted.

Having established the first structural draft, Alice defines behavioral rules in the coordination perspective. She creates a rule declaring that all chapters need to be delivered three days before the conference deadline; i.e. each of the chapters has been updated and its state set to final. A timer is defined, which triggers a notification to all the authors who did not deliver content until the specified point of time. All rules are stored within the mashup and within the rule engine.

In the document service list, Alice 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 she can add to the mashup. She defines a rule that the mashup is formatted by the layout service when all content elements are in final state. Afterwards a final 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.

Having specified the rules, Alice uses the provider list in order to find collaborators that can assist with the tasks at hand. She discovers Bob, Carol and Ted, who are members of her research team and have already agreed to join the paper. Alice drags the icon for Bob on the introduction document part; the association between him and a document service representing the writing task is immediately stored. Bob uses the same tool to provide his content as a document service. Now Alice wakes up the mashup. Bob receives a notification that he has been assigned to write this paragraph. The task is added to his personal to-do list, which is presented to him in the service editor UI after logging into the mashup tool.

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. Three days before the deadline the rule engine triggers the "reminder" rule. It identifies the document elements which are not in final state and triggers updates of the associated services. Since Bob did not deliver yet, he receives another service call for his document service to provide the text. He logs into the system, pastes the introduction text into the service editor, refines the document element adding a picture element and marks its state as final.

The rule engine observes the state change and triggers another 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 research prototypes and products. A prominent example is Google Docs (http://docs.google.com/) that allows collaborative creation of rich text documents. However, it is not possible to define any dependencies between document elements or react to events. 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, Google Wave focuses more on communication than on collaborative evolution of an enterprise document. Also, 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 [5]. 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 [1] 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 [7] define a REST protocol and format for Web based documents which 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 which events a widget might expose, a mashup developer can add rules which route data between widgets. However, since these events are mainly caused by user clicks and not by updating any content and as they are exposed by widgets and not services or the mashup, it is not possible to specify complex rules involving several events or conditions. Collaboration support is also limited, since only the full mashup can be updated at a time. Also, the mashup is still an application and cannot be used as an active document.

Loomp [9] allows users to specify content fragments and enrich them semantically. These reusable fragments are readable by humans and machines alike and might be interlinked and combined into mashups. Distribution of fragments or data can be done through various channels, for instance as Wiki page, document or feed. However, there seems to be no support for notification on updates of fragments or process support for collaborative document engineering.

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 service mashups establishes a basis to meet these requirements that we would like to discuss and share with the community. 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 service 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 a case study coming from the familiar scenario of scientific collaboration.

For future work we plan to underpin and evolve the current basis of collaborative document service 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. Glushko, R. J. and McGrath, T.: Document engineering: analyzing and designing documents for business informatics & web services. The MIT Press (2008)
2. Hull, R.: Artifact-Centric Business Process Models: Brief Survey of Research Results and Challenges. In: OTM Conferences 2, Vol. 5332, pp. 1152-1163. Springer (2008)
3. Schuster, N., Zirpins, C., Tai, S., Battle, S., and Heuer, N.: A service-oriented approach to document-centric situational collaboration processes. In: 18th IEEE Int. Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, pp. 221–226, IEEE Comp. Society (2009)
4. Marchese, M., Giunchiglia, F., and 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)
5. Noël, S., and Robert, J.-M.: How the Web is used to support collaborative writing. In: Behaviour & IT, Vol. 22, Nr. 4, pp. 245-262 (2003)
6. Schuster, N., Zirpins, C., Schwuchow, M., Battle, S., and Tai, S.: The MoSaiC Model and Architecture for Service-Oriented Enterprise Document Mashups. In: 3rd International Workshop on Web APIs and Services Mashups (Mashups'09), OOPSLA, ACM (2009)
7. Boyer, J. M., Wiecha, C., and Akolkar, R. P.: A REST protocol and composite format for interactive web documents. In: ACM Symposium on Document Engineering ACM, pp. 139-148 (2009)
8. Hoyer, V. and Fischer, M.: Market overview of enterprise mashup tools. In: ICSOC '08: Proceedings of the 6th International Conference on Service-Oriented Computing Berlin, pp. 708-721. Springer, Heidelberg (2008)
9. Luczak-Roesch, M., and Heese, R.: Linked Data Authoring for Non-Experts. In: Proceedings of the WWW09, Workshop Linked Data on the Web LDOW2009 (2009)

Managing Processes on Mobile Devices: The MARPLE Approach

Rüdiger Pryss, Julian Tiedeken and Manfred Reichert

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

Abstract. Ubiquitous Computing is considered as enabler for linking everyday life with information and communication technology. However, developing pervasive applications that provide personalized user assistance is still a challenge. Relevant scenarios are diverse and encompass domains like healthcare, logistics, and business collaboration. Two of the technologies that show increasing maturity in respect to the demands of such applications are light-weight frameworks and process engines for mobile computing. Their fusion, however, is in a rather premature state. Generally, the support of mobile collaboration using a process engine raises challenging issues that need to be addressed. In the MARPLE project we target at a tight integration of process management technology with mobile computing frameworks in order to enable mobile process support in the aforementioned scenarios. In this demo paper 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.

1 Introduction

Mobile assistance in daily life as empowered by information and communication technology is a much discussed topic. To better understand the challenges emerging in this context, we analyzed real-world scenarios in which mobile user assistance is urgently needed and which stem from domains like healthcare, logistics and business collaboration. In particular, our analyses revealed the fundamental role of process support in respect to mobile and personalized user assistance.
This paper picks up a healthcare scenario in which chronically ill patients shall be assisted by mobile devices. Such mobile device gives recommendations in respect to medications. These recommendations, in turn, are made remotely by healthcare professionals and depend on the previously gathered patient data (e.g. blood pressure). Despite its high potential, so far there exists no comprehensive mobile assistance for such scenarios. One issue emerging in the given context is to decide which process parts shall run on mobile devices and which on stationary computers. In the following we refer to the described scenario to discuss fundamental challenges and to show the high potential of mobile assistance. Fig. 1 illustrates both traditional realization of this scenario ① and its realization based on mobile devices and mobile assistance respectively ②.

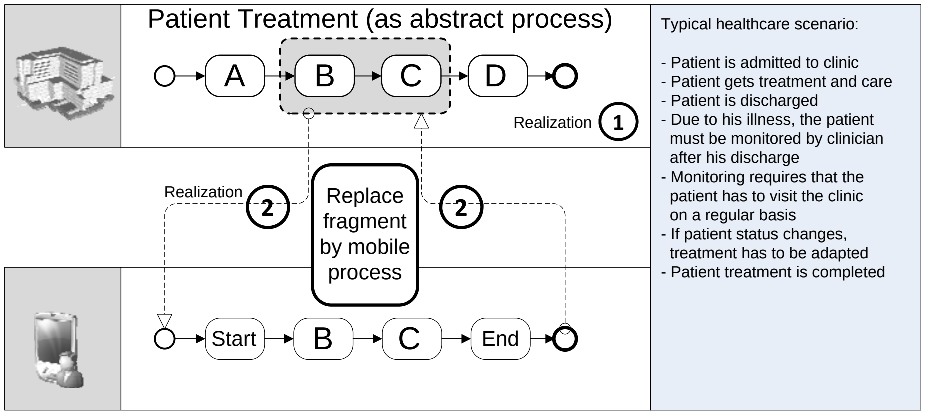


Fig. 1. Abstract Healthcare Scenario

After discharging a patient the usual way to monitor his health status is to schedule regular visits for him in the clinic. In certain cases, however, this can lead to delayed adaptations of his treatment plan in case his status has changed. To improve this situation and to enable real-time monitoring, mobile data collection and mobile assistance ② of the patient would be highly welcome by all parties; i.e., the patient needs to be assisted by a mobile device which gathers medical data from him and informs clinicians about status changes.

To realize the second scenario patient-specific application logic needs to be provided on the mobile device. Consequently, the overall treatment workflow is maybe partitioned ② and process fragments may run on stationary computers as well as on mobile devices. In particular, the process fragment running on the mobile device needs to be adapted to the specific patient and may evolve over time, i.e., hard-coded process implementations are not tolerable.

To enable mobile assistance we developed a light-weight process engine called MARPLE that runs on the mobile device and that is able to interact with back-end processes if required. In addition, we provide advanced tools for defining, configuring and verifying process fragments. In this paper, we focus on the core architecture and components of the MARPLE mobile process engine. Conceptual issues related to the partitioning of processes as well as to the synchronization of the resulting fragments are outside the scope of this paper.

When developing our MARPLE engine we had one shining example to follow - the ADEPT process management system we had developed during the last decade [1]. In particular, we adopt basic design principles from ADEPT (e.g., correctness-by-construction, dynamic process adaptability), but also align the MARPLE architecture with specific needs of mobile processes.

Section 2 introduces a concrete application scenario. In Section 3 we give insights into the MARPLE architecture, while Section 4 shows how the described scenario can be supported in MARPLE. Section 5 discusses related work and Section 6 concludes with a summary and outlook.

2 Application Scenario

Fig. 2 shows a typical example of a healthcare process which is modeled in terms of BPMN. Assume that for a particular patient this process is executed three times per day and involves three parties. The first swim lane ① shows activities conducted at the clinic, which starts the process. When executing step ②, the clinic triggers the process fragment running on the mobile device of the patient. This mobile process collects patient data and coordinates required actions (e.g., to measure blood pressure or to gather ECG data).

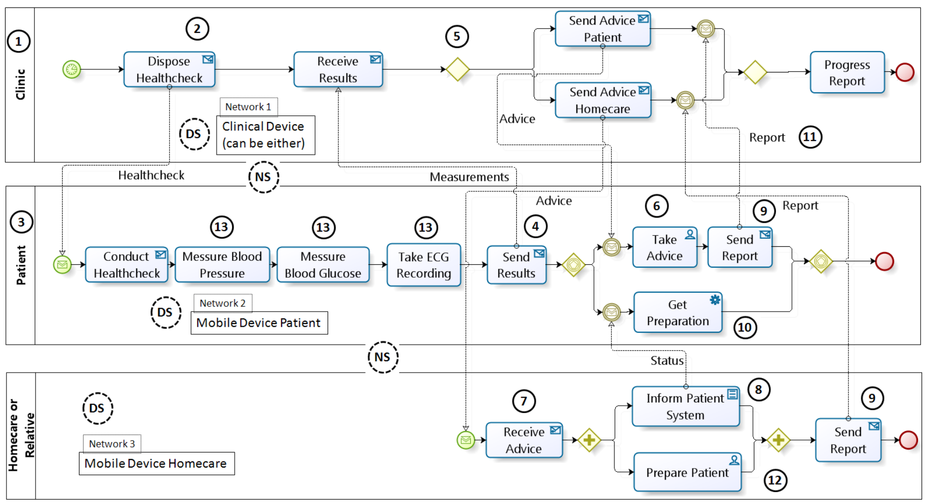


Fig. 2. Healthcare Process with Mobile Patient Support

Let us consider the process ③ to be run on the mobile device of the patient in more detail: While the patient stays at home, he first gets a message through his mobile device. Then he measures and collects the requested data being assisted by the process running on his mobile device. Following this, results ④ are sent back to the clinic which then decides ⑤ about next steps. Ideally no special actions are required. In this case a message is sent back to the patient's mobile device containing information about his medication ⑥. Alternatively, the clinic may send a message with information about further treatment or special treatment to home care ⑦ (either provided by a professional service or a relative of the patient). In the latter case, an additional process fragment is started on the mobile device of the person who is responsible for home care. This process has to be synchronized with the process with the one running on the patient's mobile device. Finally, either the process running on the mobile device of home care or the one running on the mobile device of the patient sends back a report ⑨ to the clinic. Then the process is finished.

Altogether the process fragments of three parties need to be synchronized. Thereby, the runtime infrastructure must be able to cope with communication problems, device failures and so forth. In Fig. 2 the pictograms with label DS and NS indicate a Network Switch or Device Switch within the overall process choreography. Assume that the mobile device of the patient or its connection with the clinic fail. Then the clinic has now knowledge about the status of the patient, but only has the information that the network connection has broken. Such failure scenarios must be covered by the architecture. In particular, the following requirements need to be met by a supporting infrastructure:

- It must be possible to partition a process model and to allocate the resulting fragments on mobile devices as well as stationary computers.
- Soundness of the process (i.e., the process choreography) needs to be ensured.
- The runtime infrastructure has to cope with physical problems like broken connections or malfunctioning devices.
- When running the fragments on distributed devices their execution must be synchronized and messages be exchanged in a reliable way.
- Both the overall process model as well as its fragments might have to be adapted during runtime, e.g. to deal with exceptional situations.
- A mobile process must be able to gather sensoric data during its execution.

3 MARPLE Architecture

In this section we give insights into the MARPLE architecture (cf. Fig. 3). Its two core components are the *MARPLE Mobile Process Engine V1.3* and the *MARPLE Mediation Center*. Here we focus on those parts of the MARPLE architecture that are relevant in the context of our application scenario. Other components and features of MARPLE are only mentioned shortly and will be subject of future publications.

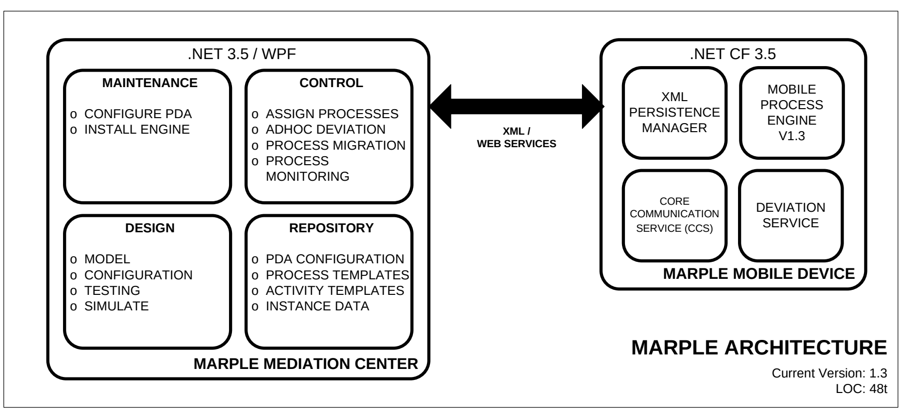


Fig. 3. MARPLE Architecture

If a mobile device shall be added to the MARPLE environment, it first needs to be equipped with the basic software services required in the MARPLE context. Amongst others this includes *Core Communication Services (CCS)* as part of the *MARPLE Mobile Process Engine V1.3*. Thereby, we follow a light-weight approach; i.e., services that are initially not needed are not loaded to the device. Following this, the mobile device can connect to the *MARPLE Mediation Center* and indicate that its configuration may start. When starting the MARPLE configuration procedure on the mobile device through the *MARPLE Mediation Center*, CCS dynamically loads the *MARPLE Mobile Process Engine V1.3*, the *MARPLE XML Persistence Manager*, and the relevant process as well as *Activity Templates* to this device. In this context, *Activity Templates* encapsulate pre-manufactured application components that implement the process steps. In MARPLE, for example, activity templates can be associated with a user forms or a (remote) web service call. Regarding our example from Section 2, in process step ⑦ a report is sent from the patient's mobile process to the clinic. When realizing the *MARPLE Mobile Process Engine V1.3*, we re-use fundamental concepts and design principles of the ADEPT process management technology, which we developed during the last decade [1]. In particular, we adopt the ADEPT process meta model, apply its fundamental correctness notions and correctness checks, and enable flexible process enactment on the mobile device. The latter includes dynamic adaptations of process instances running on the mobile device (e.g., to cope with contextual changes in the environment) and is realized by 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 V1.3* in order to meet performance requirements of mobile scenarios and to cope with their specific requirements (e.g., broken connections and limited GUIs). In particular, the implementation framework MARPLE is based on is not the same as the one used in the context of ADEPT. While ADEPT relies on JAVA, our MARPLE architecture is based on *.NET Compact Framework.* The *MARPLE Mediation Center* consists of four major parts. First, its *Maintenance* component allows us to configure mobile devices such that they can be used for mobile process support. Second, the *Control Unit* enables users to assign executable processes to mobile devices, to enact them on the mobile device, to invoke user forms or web services during process execution, and to apply ad-hoc deviations from the pre-modeled process logic.

Another fundamental feature of the MARPLE *Control Unit* is its ability to migrate running process instances from one mobile device to another (e.g., if a patient wants to switch his device). Like ad-hoc changes such process migration can be initiated by the owner of the mobile device as well as by authorized users via the *Control Unit.* Another important component of the *MARPLE Mediation Center* is its *Modeler*. This component adopts basic correctness principles we developed in ADEPT, but provides additional features for partitioning processes and for specifying conceptual models for mobile processes. Consider again our example from Section 2. Using *MARPLE Modeler*, the fragment representing the data collection process (see Lane ③) can be defined. The same holds for the process fragment relating to home care. All meta data (e.g., PDA configurations) needed by the different components of the MARPLE architecture are maintained in the Repository of the *MARPLE Mediation Center*. Fig. 4 exemplarily illustrates the interaction 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 migrated to the newly introduced one (e.g., due to connection problems with the first device or better technical features of the new one). Note that this migration can be triggered either by the *MARPLE Mediation Center* or by the owners of the two devices. During process executions, the *Control Unit* may suspend, resume, abort and monitor running processes. Further, the *MARPLE Mobile Process Engine V1.3* logs progress of the process using the *Persistence Manager*.

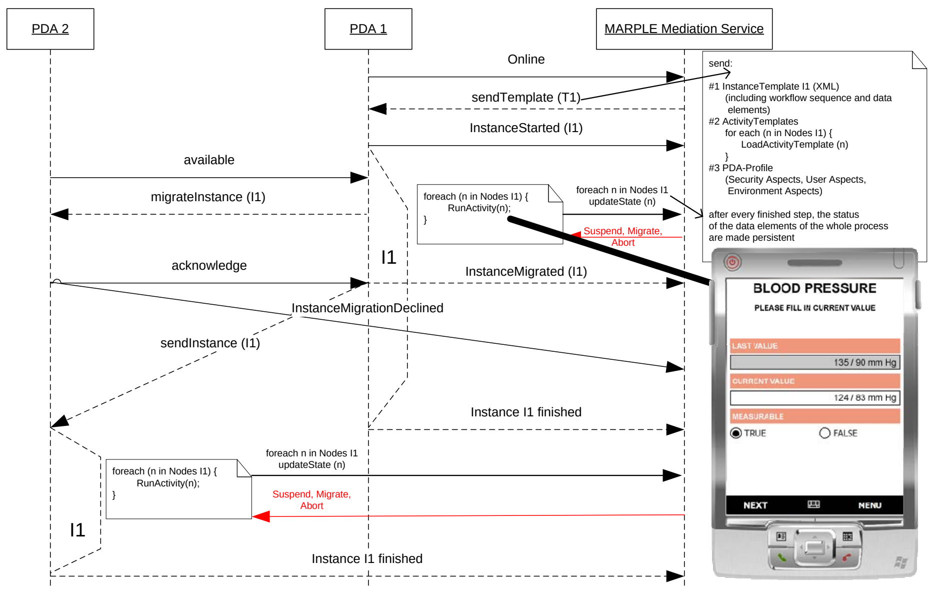


Fig. 4. MARPLE: Interaction Sequence

4 MARPLE-Demonstration

We revisit our scenario from Section 2 and show how it can be realized using MARPLE. Fig. 5 depicts the user interface of the *MARPLE Mediation Center*. With *MARPLE Modeler*, we can completely define the patient-centered data collection process from the middle lane in Fig. 2. Further, MARPLE, enables remote monitoring of process instances; e.g. Fig. 5 ① shows a concrete process instance running on a mobile device as it can be monitored using MARPLE. 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 would need when remotely monitoring patient processes. Let us shortly consider how the above mentioned process fragment is modeled in MARPLE. Fig. 5 shows a part of this model together with instance-specific markings. Activity ② is a *receive* activity which is waiting for an incoming message requesting a health check. The subsequent three activities constitute data collection steps which are either implemented as user forms or sensing activities; the blood pressure is gathered via a bluetooth activity template from the linked blood pressure system. Blood glucose and ECG recordings are entered 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 ④ 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). Subsequent activity ④ then waits until a message is received either from the clinic or from home care. The toolbar on the left of Fig. 5 (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. So far, we have focused on the implementation of the *MARPLE Mobile Process Engine V1.3* and on robustness issues emerging with mobile processes.

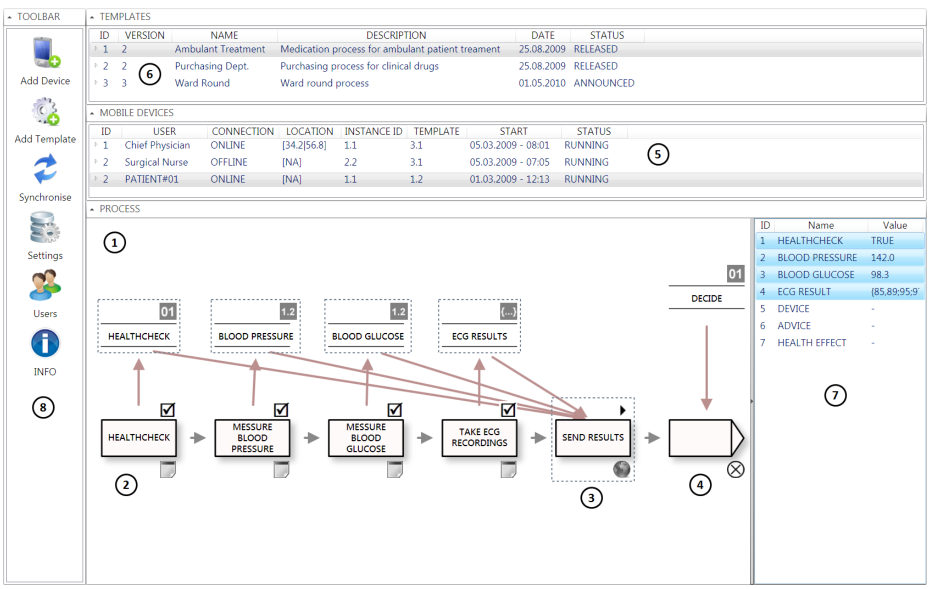


Fig. 5. MARPLE: Mediation Center

5 Related Work

In literature we can find 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, [2] deals with the partitioning of BPEL processes. A similar approach has been suggested in the context of ADEPT [3]. However, none of the two approaches has provided an architecture for mobile process support as suggested by MARPLE. Taking mobile network dynamics as core demand for mobile process engines, many approaches deal with failures and exceptions like broken connections or lack of communication facilities [4–7].

Respective tools usually apply web service standards and base process execution on BPEL or more specific execution models derived from BPEL. 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 high level process model that can be adapted by both remote users as well as users of the mobile device.

6 Summary and Outlook

We introduced our MARPLE architecture and described how its core components enable the execution and monitoring of processes 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, instance migrations, etc. To foster this vision we base our work on core design principles and fundamental concepts we developed in our ADEPT project. In future work we will extend the *MARPLE Modeler* such that it provides sophisticated methods for modeling complex process choreographies like the one from Fig. 2. This will include, for example, a methodology for correctly partitioning processes models, for allocating resulting fragments on different machines and devices, and for synchronizing them at runtime. In particular, we will adopt and extend concepts from autonomic computing and self-healing systems to cope with the many failure scenarios in connection with distributed and mobile applications.

References

1. 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** (2009) 81–97
2. Baresi, L., Maurino, A., Modafferi, S.: Workflow partitioning in mobile information systems. In: MOBIS'04. (2004) 93–106
3. Bauer, T., Reichert, M., Dadam, P.: Intra-subnet load balancing in distributed workflow management systems. Int'l Journal of Cooperative Information Systems **12** (2003) 205–323
4. Kunze, C.P.: Demac: A distributed environment for mobility-aware computing. In: Proc. 3rd Int. Conf. on Pervasive Computing. (2005) 115–121
5. Hackmann, G., Haitjema, M., Gill, C.: Sliver: A BPEL workflow process execution engine for mobile devices. In: Proc. 4th Int. Conf. on Service Oriented Computing. (2006) 503–508
6. Schmidt, H., Kapitza, R., Hauck, F.J.: Mobile-process-based ubiquitous computing platform: a blueprint. In: Proc. 1st Workshop on Middleware-application interaction. (2007) 25–30
7. Gerhard, S., Jürgen, M., Erich, S.: Building a modular service oriented workflow engine. In: IEEE Int. Conf. on Service-Oriented Computing and Applications. (2009)

SMART: Simple Monitoring enterprise Activities by RFID Tags

Fabrizio Angiulli[2], Elio Masciari[1]

[1] ICAR-CNR – Institute of Italian National Research Council
masciari@icar.cnr.it
[2] DEIS-UNICAL
Via P. Bucci, 87036 Rende (CS) Italy
fangiulli@deis.unical.it

Abstract. Datastreams are potentially infinite sources of data that flow continuously while monitoring a physical phenomenon, like temperature levels or other kind of human activities, such as clickstreams, telephone call records, and so on. RFID technology has lead in recent years the generation of huge streams of data. Moreover, RFID based systems allow the effective management of items tagged by RFID tags, especially for supply chain management or objects tracking. In this paper we introduce SMART (Simple Monitoring enterprise Activities by RFID Tags) a system based on outlier template definition for detecting anomalies in RFID streams. We describe SMART features and its application on a real life scenario that shows the effectiveness of the proposed method for effective enterprise management.

1 Introduction

In this paper we will focus on Radio Frequency Identification (RFID) data streams monitoring as RFID based systems are emerging as key components in systems devoted to perform complex activities such as objects tracking and supply chain management. Sometimes RFID tags are referred to as electronic bar codes. Indeed, RFID tags emit a signal that contains basic identification information about a product. Such tags can be used to track a product from manufacturing through distribution and then on to retailers. These features of RFID tags open new perspectives both for hardware and data management. In fact, RFID is going to create a lot of new data management needs. In more details, RFID applications will generate a lot of so called "thin" data, i.e. data pertaining to time and location. In addition to providing insight into shipment and other supply chain process efficiencies, such data provide valuable information for determining product seasonality and other trends resulting in key information for the companies management. Moreover, companies are exploring more advanced uses for RFID. For instance, tire manufacturers plan to embed RFID chips in tires to determine the tire deterioration. Many pharmaceutical companies are embedding RFID chips in drug containers to better track and avert the theft of highly controlled drugs. Airlines are considering RFID-enabling key onboard parts and supplies to optimize aircraft maintenance and airport gate preparation turnaround time.

Such a wide variety of systems for monitoring data streams could benefit of the definition of a suitable technique for detecting anomalies in the data flows being analyzed. As a motivating example you may think about a company that would like to monitor the mean time its goods stay on the aisles. Items are tagged by RFID tags so the reader continuously produces a readings that report the electronic product code of the item being scanned, its location and timestamp, this information can be used, as an example, for signaling that the item lays too much on the shelf since it is repeatedly scanned in the same position. It could be the case that the package is damaged and consequently customers tend to avoid the purchase. If an item exhibits such a feature it deserves further investigation. Such a problem is relevant to a so huge number of application scenario that it is impossible to define an absolute notion of anomalies (in the follow we refer to anomalies as outliers). In this paper we propose a framework for dealing with the outlier detection problem in massive datastreams generated in a network environment for objects tracking and management. The main idea is to provide users a simple but rather powerful framework for defining the notion of outlier for almost all the application scenarios at an higher level of abstraction, separating the specification of data being investigated from the specific outlier characterization.

2 Preliminaries

An RFID system consists of three components: the *tag*, the *reader* and the *application* which uses RFID data. Tags consist of an antenna and a silicon chip encapsulated in glass or plastic. RFID readers or receivers are composed of a radio frequency module, a control unit and an antenna to query electronic tags via radio frequency (RF) communication. They also include an interface that communicates with an application (e.g., the check-out counter in a store). Readers can be hand-held or mounted in specific locations in order to ensure they are able to read the tags as they pass through a *query zone* that is the area within which a reader can read the tag. The query zone are the locations that must be monitored for application purposes. In order to explain the typical features of an RFID application we consider the typical supply chain scenario.

The chain from the farm to the customer has many stages. At each stage goods are typically delivered to the next stage, but in some case a stage can be missing. The following three cases may occur: 1) the goods lifecycle begin at a given point (i.e. production stages, goods are tagged there and then move through the chain) and thus the reader in the zone register only departures of goods, we refer to this reader as *source reader*; 2) goods are scanned by the reader both when they arrive and they leave the aisle, in this case we refer to these reader as *intermediate reader*; 3) goods are scanned and the tag is killed, we refer to these readers as *destination reader*.

A RFID stream is (basically) composed of an ordered set of n sources (i.e., tag readers) located at different positions, denoted by $\{r_1, \ldots, r_n\}$ producing n independent streams of data, representing tag readings. Each RFID stream can be basically viewed as a sequence of triplets $\langle id_r, epc, \tau_s \rangle$, where: 1) $id_r \in \{1, .., n\}$ is the tag reader identifier (observe that it implicitly carries information about the spatial location of the reader) ; 2) *epc* is the product code read by the source identified by id_r and 3)τ_s is a *timestamp*, i.e., a value that indicates the time when the reading *epc* was produced by the source id_r.

An outlier is an observation that markedly differs from other observations as to lead to the suspect that it was generated by a different mechanism [4]. There exist several approaches to the identification of outliers, namely, statistical-based [2], distance-based [3], density-based [5] and MDEF-based [6]. The problem has been tackled from different viewpoint and in different scenarios such as static dataset, dynamic dataset and very large dataset[1]. In our application scenario we deal with massive datastreams that can be viewed as kind of a very large dynamic dataset. Based on the notion of RFID stream introduce so far, it is easy to see that each RFID reading generated by an RFID tag could be an outlier either because 1) the (product) features (obtained by the *epc* such as price, weight, height and so on) greatly differs from the others readings or 2) the latency time that the tagged item spent in a given location deviates significantly from an expected value.

In our system we will assume either distance based outlier function or statistical based outlier function to catch both source of anomaly and since we are interested in the problem formalization, we disregard here the actual outlier function implementation. More formally, given a set of objects S, a positive integer k, and a positive real number R. An object $o \in S$ is a $DB(k, R)$- outlier, or a distance-based outlier with respect to parameters k and R, if less than k objects in S lie within distance R from o. This kind of function will be exploited when searching for outliers based on their product features. To deal with deviation on time features we resort to statistical based outlier function. We point out that a formal analysis of the possible outlier detection methods is out of the scope of this paper, we mentioned here the main approaches used in literature since in our system implementation we allow any stream oriented implementation of outlier function to be used. The latter observation guarantees a high flexibility in our system for dealing with every possible application scenarios.

3 Statement of the Problem

In our model, *epc* is the identifier associated with a single unit being scanned (this may be a pallet or a single item, depending on the level of granularity chosen for tagging the goods being monitored).

This basic schema is simple enough to be used as a basic schema for a data stream environment, anyway since more information are needed about the outlier being detected we can access additional information by using some auxiliary tables maintained at a $Master$ site as shown in figure 2. More in detail, the

$Master$ maintains an intermediate local warehouse of RFID data that stores information about items, items' movements, product categories and locations and is exploited to provide details about RFID data upon user requests. The information about items' movements are stored in the relation $ItemMovement$ and the information about product categories and locations are stored in the relations *Product* and *Locations*, respectively. These relations represents, respectively, the *Product* and the *Location* hierarchy. Relation *EPCProducts* maintains the association between *epcs* and product category, that is, every *epc* is associated to a tuple at the most specific level of the *Product* hierarchy. Finally, RFID readers constitute the most specific level of the *Location* hierarchy.

$ItemMovements$ contains tuples of the form $\langle epc, DL \rangle$, where *epc* has the usual meaning, and DL is string built as follows: each time an *epc* is read for the first time at a node N_i a trigger fires and DL is updated appending the node identifier.

In the following we define a framework for integrating DSMS technologies and outlier detection framework in order to effectively manage outliers in RFID datastreams. In particular we will exploit the following features: *a)* The definition of a template for specifying outlier queries on datastreams that could be implemented on top of a DSMS by mapping the template in a suitable set of continuous queries expressed in a continuous query language language ESL-like[7]; *b)* The template need to be powerful enough to model all the interesting surveillance scenarios. In this respect, it should allow the definition of four components, namely: 1) the kind of objects (O) to be monitored (e.g. RFID data concerning dairy products),2) the reference population P (due to the infinite nature of datastream) depending on the application context (e.g. a subset of the items belonging to dairy products category), 3) the attributes (A) of the population used for signing out anomalies (e.g. time spent at a given location), 4) the outlier definition by means of a suitable function $\mathcal{F}(P, A, O) \rightarrow \{0, 1\}$ (e.g. deviation from the average time spent at a given location by an item); *c)* A mapping function that for a given template and DSMS schema, resolve the template in a set of outlier continuous queries to be issued on the datastream being monitored.

The basic intuition behind the template definition is that we want to run an aggregate function that is raises by the *Master* (that is a central node collecting the queries and the aggregate statistics along with the sample populations) and then instantiated on a subset of nodes in the network. An incoming stream is processed at each *node* where the template is activated by the $Master$ that issue the request for monitoring the stream. Once a possible outlier is detected, it is signaled to the $Master$. The master maintains management information about the network and some additional information about the items using two auxiliary tables $OutlierMovement$ and *NewTrend*. In the $OutlierMovement$ table it stores information about the outlying objects, in particular it stores their identifiers and the paths traveled so far as explained above for $ItemMovements$. The $NewTrend$ table stores information about objects that are not outliers but instead they represent a new phenomenon in the data. It contains tuples of the form $\langle epc, N, \tau_a, \tau_l, \rangle$, where N is a node, τ_a and τ_l are, respectively, the arrival time and the time interval spent at node N by the *epc*. The latter table is really important since it is intended to deal with the concept drift that could affect the data. Indeed, when items are marked as unusual but they are not an anomalies as in the case of varied selling rates they are recorded for later use in outlier definition. In particular, once the new trend has been consolidated, new statistics for the node where the objects appeared will be computed at $Master$ level and then forwarded to the pertaining node in order to update the parameters of its population.

As mentioned above candidate outliers are signaled at node level but they are managed by the master. More in detail, as a possible outlier is signaled by a given node the master stores it in the $OutlierMovement$ table along with its path if it is recognized as an anomaly or in the $NewTrend$ table if a signaled item could represent the symptom of a new trend in data. To summarize, given a signaled object o two cases may occur: 1) o is an outlier and then it is stored in the $Outlier$ table; 2) o represent a new trend in data distribution and then it should not be considered an outlier and we store it in the $NewTrend$ table. To better understand such a problem we define three possible scenarios on a toy example.

Example 1. Consider a container (whose *epc* is p_1) containing dangerous material that has to be delivered through check points c_1, c_2, c_3 in the given order and consider the following sequence of readings: $Seq_A = \{(p_1, c_1, 1), (p_1, c_1, 2), (p_1, c_2, 3), (p_1, c_2, 4), (p_1, c_2, 5), (p_1, c_2, 6), (p_1, c_2, 7), (p_1, c_2, 8), (p_1, c_2, 9), (p_1, c_2, 10), (p_1, c_2, 11), (p_1, c_2, 12)\}$. Sequence A correspond to the case in which the pallet tag is read repeatedly at the check point c_2. This sequence may occur because: *i*) the pallet (or the content) is damaged so it can no more be shipped until some recovery operation has been performed, *ii*) the shipment has been delayed. Depending on which one is the correct interpretation different recovery action need to be performed. To take into account this problem in our prototype implementation we maintain appropriate statistics on latency time at each node for signaling the possible outlier. Once the object has been forwarded to the master a second check is performed in order to store it either in $OutlierMovement$ or in $NewTrend$ table. In particular, it could happen that due to new shipping policy additional checks have to be performed on dangerous material, obviously this will cause a delay in shipping operations, thus the tuple has to be stored in the $NewTrend$ table.

Consider now a different sequence of readings: $Seq_B = \{(p_1, c_1, 1), (p_1, c_1, 2), (p_1, c_1, 3), (p_1, c_1, 4), (p_1, c_3, 5), (p_1, c_3, 6), (p_1, c_3, 7), (p_1, c_3, 8), (p_1, c_3, 9), (p_1, c_3, 10), (p_1, c_3, 11), (p_1, c_3, 12)\}$. Sequence B correspond to a more interesting scenario, in particular it is the case that the pallet tag is read at check point c_1, is not read at check point c_2 but is read at checkpoint c_3. Again two main explanation could be considered: *i*) the original routing has been changed for shipment improvement, *ii*) someone changed the route for fraudulent reason (e.g. in order to steal the content or to modify it). In this case suppose that the shipping plan has not been changed, this means that we are dealing with an outlier then we store it in the *OutlierMovement* table along with its path.

Finally, consider the following sequence of readings regarding products p_1, p_2, p_3 that are frozen foods, and product p_4 that is perishables, all readings generated at a freezer warehouse c: $Seq_C = \{(p_1, c, 1), (p_2, c, 2), (p_3, c, 3), (p_4, c, 4), (p_1, c, 5), (p_2, c, 6), (p_3, c, 7), (p_4, c, 8), (p_1, c, 9), (p_2, c, 10), (p_3, c, 11), (p_4, c, 12)\}$. Obviously, p_4 is an outlier for that node of the supply chain and this can be easily recognized using a distance based outlier function since its expiry date greatly deviates from the expiry dates of other goods.

The Template in a short In this section we will describe the functionalities and syntax of the *Template* introduced so far. A *Template* is an aggregate function that takes as input a stream. Since the physical stream could contain several attributes as explained in previous sections we allow *selection* and *projection* operation on the physical stream. As will be clear in next section we will use a syntax similar to *ESL* with some specific additional features pertaining to our application scenario. This filtering step is intended for feeding the reference population P. In particular, as an object is selected at a given node it is included in the reference population for that node using an *Initialize* operation, it persists in the reference population as a *Remove* operation is invoked (it can be seen as an *Initialize* operation on the tuples exiting the node being monitored).

We recall that a RFID tagged object is scanned multiple times at a given node N so when the reader no more detects the RFID tag no reading is generated. First time an object is read a *Validate* trigger fires and send the information to the *Master* that eventually updates the *ItemMovement* table. In response to a *Validate* trigger the *Master* performs a check on the item path, in particular it checks if shipping constraints are so far met. In particular, it checks the incoming reading for testing if the actual path so far traveled by current item is correct. This check can be performed by the following operations: 1) selection of the path for that kind of item stored in *ItemMovement*, 2) add the current node to the path, 3) check the actual path stored in an auxiliary table *DeliveryPlans* storing all the delivery plans (we refer to this check as *DELIVERY CHECK*). This step is crucial for signaling path anomalies since as explained in our toy examples that source of anomaly arise at this stage. If the item is not validated the *Master* stores the item information in order to solve the conflict, in particular it could be the case that delivery plans are changing (we refer to this check as *NEW PATH CHECK*) so information is stored in *NewTrend* table for future analysis , otherwise it is stored in the *OutlierMovement* table. To better understand this behavior consider the Seq_B in example 1. When the item is first time detected at node c_3 the *Validate* trigger fires, the path so far traveled for that object is retrieved obtaining $path = c_1$, the current node is added thus updating $path = c_1.c_3$ but when checked against the actual path stored in *DeliveryPlans* an anomaly is signaled since it was supposed to be $c_1.c_2.c_3$. In this case the item is stored in the *OutlierMovement* table and the *Master* signal for a recovery action. It works analogously for Seq_A as explained in example 1.

When an *epc* has been validated it is added to the reference population for that node (P_N) then it stays at the node and is continuously scanned. It may happen that during its stay at a given node an *epc* could not be read due to temporary field problem, we should distinguish this malfunction from the "normal" behavior that arise when an item is moved for shipping or (in case of destination nodes) because it has been sold. To deal with this feature we provide a trigger $Forget$ that fires when an object is not read for a (context depending) number of reading cycles (we refer in the following as *TIMESTAMP CHECK*. We point out that this operation is not lossy since we recall that at each node we maintain (updated) statistics on items. When $Forget$ runs, it removes the "old" item from the actual population and update the node statistics. Node statistics (we refer hereafter to them as $model_M$ where N is the node they refer to) we take into account for outlier detection are: number of items grouped by product category (*count*), average time spent at the node by items belonging to a given category (m), variance for items (v) belonging to a given category, maximum time spent at the current node by items belonging to a given category (max_t), minimum time spent at the current node by items belonging to a given category (min_t). By means of the reference population P_N and the node statistics $model_N$ the chosen outlier function checks for anomalies. In particular, we can search for two kind of anomalies: 1) *item based* anomalies, i.e. anomaly regarding the item features, in this case we will run a distance-based outlier detection function; 2) *time based* anomalies, i.e. anomaly regarding arrival time or latency time, in this case we will run a statistical based outlier detection function.

3.1 The RFID-$\mathcal{T}$ syntax

In this section we formalize the syntax for template definition. For basic stream operation we will refer to ESL-like syntax[7]. We point out that even if in this paper we focus on RFID data and outlier detection task, the framework is rather general and could be exploited in several application domains and for other task such as aggregate queries evaluation.

The first step is to create the stream related to nodes being monitored. Once the streams are created at each node the $Template$ definition has to be provided.

Aggregate function can be any SQL available function applied on the reference population as shown in Fig. 5, where *Return* and $Next$ have the same interpretation as in SQL and $< Type >$ can be any SQL aggregate function. An empty $TERMINATE$ clause refer to a non-blocking version of the aggregate.

As the template has been defined it must be instantiated on the nodes being monitored. In particular triggers $Validate$ and $Forget$ are activated at each node. As mentioned above they will continuously update the reference population and node and $Master$ statistics. The syntax of these triggers is shown in figure 6.

We point out again that $Validate$ trigger has the important side-effect of signaling *path* outliers. We point out that the above presented definition is completely flexible so if the user may need a different outlier definition she simply needs to add its definition as a plug-in in our system.

CREATE STREAM	$< name >$
ORDER BY	$< attribute >$
SOURCE	$< systemnode >$
DEFINE OUTLIER TEMPLATE	$< name >$
ON STREAM	$< streamname >$
REFERENCE POPULATION	$(< definepopulation >)$
MONITORING	$(< target >)$
USING	$< outlierfunction >$
$< definepopulation >$	INSERT INTO $< PopulationName >$ SELECT $< attributelist >$ FROM $< streamname >$ WHERE $< conditions >$
$< target >$	$< attributelist > \mid < aggegatefunction >$
$< outlierfunction >$	$< distancebased > \mid < statisticalbased >$

Fig. 1. Template Definition syntax

AGGREGATE	<Function Name> <Type>(Next Real) : Real
{ TABLE	<Table Name> (<attribute list>);
INITIALIZE:	{ INSERT INTO <Table Name> VALUES (Next, 1); }
ITERATE:	{ UPDATE <Population Name> SET <update condition>;
INSERT INTO RETURN	SELECT <output attribute> FROM <Table Name> }
TERMINATE :	{} }
CREATE TRIGGER	Validate
BEFORE	INSERT ON <Population Name>
REFERENCING NEW AS	NEW READING
IF PATH CHECK	INSERT INTO <Population Name> VALUES (NEW READING)
ELSE IF DELIVERY CHECK	INSERT INTO NewTrend VALUES (NEW READING)
ELSE	INSERT INTO OutlierMovement VALUES (NEW READING)
CREATE TRIGGER	Forget
AFTER	INSERT ON <Population Name>
REFERENCING OLD AS	OLD READING
IF TIMESTAMP CHECK	{DELETE FROM <Population Name> OLD READING UPDATE STATISTICS ON <Population Name> }

Fig. 2. Aggregate Function, Validate an Forget trigger syntax

References

1. F. Angiulli and F. Fassetti. DOLPHIN: An Efficient Algorithm for Mining Distance-Based Outliers in Very Large Datasets. *TKDD*, 8(1), 2009.
2. V. Barnett and T. Lewis. *Outliers in Statistical Data.* Wiley and Sons, 1994.
3. R. NG E. Knorr and V. Tucakov. Distance-based outlier: Algorithms and applications. *VLDB J.*, 8(3-4):237–253, 2000.
4. D. Hawkins. *Identification of Outliers.* Monographs on Applied Probability and Statistics. Chapman and Hall, 1980.
5. R. Ng J. Sander M.M. Breunig, H. Kriegel. Lof: Identifying density-based local outliers. In *In Proceedings of the International Conference on Managment of Data (SIGMOD00).*
6. P. Gibbons S. Papadimitriou, S. Kitagawa and C. Faloutsos. Loci: Fast outlier detection using the local correlation integral. In *In Proceedings of the International Conference on Data Enginnering (ICDE)*, pages 315–326, 2003.
7. H. Thakkar H. Wang Y. Bai, R.C. Luo and C. Zaniolo. An introduction to the Expressive Stream Language (ESL). *Tech. Report.*

Decisions and Decision Requirements for Data Warehouse Systems

Naveen Prakash[1], Deepika Prakash[2], Daya Gupta[3]

[1]MRCE, Sector-43, Faridabad, INDIA
praknav@hotmail.com
[2,3]Department of Computer Engineering, DTU, Delhi, INDIA
dpka.prakash@gmail.com, daya_gupta2005@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.

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]. Today, there is a body of opinion that uses goal oriented techniques [10, 11, 13, 15, 16] for determining data warehouse structure. 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.

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 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. When the information system is sent a stimulus from the decisional environment then the functionality that responds to this stimulus is invoked.

Stimuli can be sent by two different kinds of actors, IS administrators and IS operators. These stimuli correspond to two kinds of decisions, managerial and imperative. Managerial decisions are used to 'initialize' the IS where as the latter work within the initialized IS to operate the system. For example, in a railway reservation system IS administrators initialize train data whereas IS operators invoke functionality to make reservations and cancellations using information set up by the IS administrator.

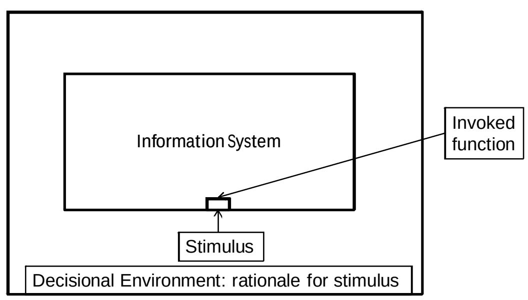


Fig. 1. Embedded IS in a Decisional Environment.

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 decide on the employee from the choice set {Transfer employee, Recruit employee, Overload employee}. The manager needs information to decide which alternative to pick and, also which individual employee shall be transferred, recruited, or overloaded respectively. There are two decision making problems here, to select from the choice set and to identify the individual, respectively. We shall use the notions of tactical decisions and operational decisions to classify these.

Fig. 2 shows the interplay of tactical and operational decisions. 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.

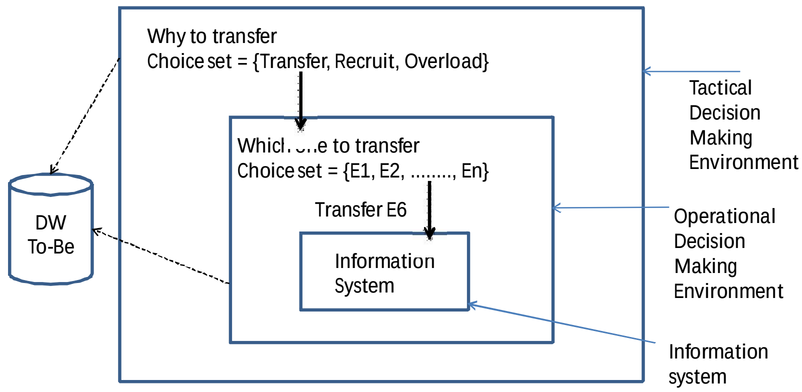


Fig. 2. Imperative Decisions and the interplay between tactical and operational decisions.

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

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**. The latter provide the context for the former.

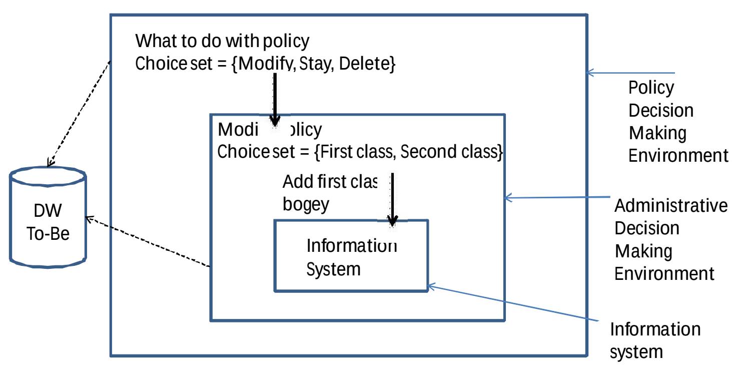


Fig. 3. Managerial Decisions

Let us be given a **policy decision** that the ratio of first class bogies in a train to second class bogies is 1:2. This policy is to be enforced as an **administrative decision**. **Policy decisions** may define the norms and standards that are used by administrative decisions or business rules used by imperative decisions. A policy decision requires knowledge of the state of the organization. For example deciding the 1:2 norm above requires the knowledge of 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.

Fig. 3 shows that the policy decision to modify the ratio of first to second class bogeys in a train leads to the administrative decision to add a first class bogey, 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.

3 Decision Requirement

We have seen that in order to make a decision reference to the information in the data warehouse needs to be made. We represent this as a pair <decision, information> and refer to it as a decision requirement. Here, we elaborate on the notion of decision requirement.

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.

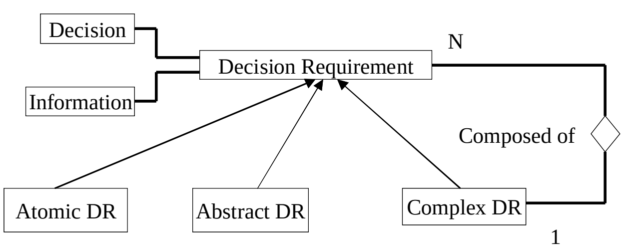


Fig. 4. Decision Requirements Meta-model.

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 requirements. Complex decision requirements form an AND/OR hierarchy.

To illustrate an abstract DR, consider an automobile plant that makes 1-tonne and 13-tonne trucks. Let the decision of interest be *Set up New Assembly Line* and the required information be *Unsatisfied Orders.* 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*.

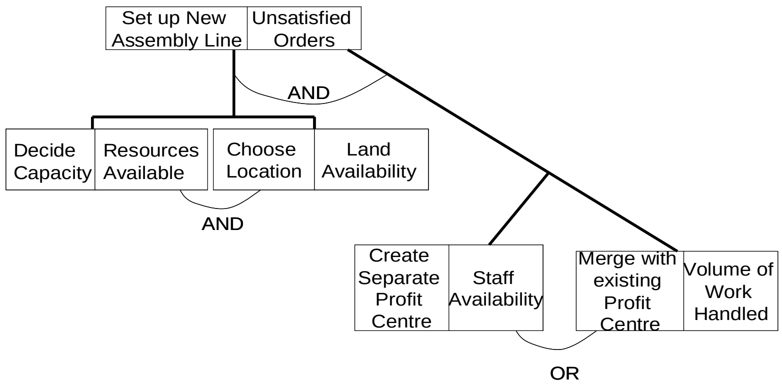


Fig. 5. Composition of Decision Requirements with AND and OR link

Now let us consider composition. 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. 5).

The foregoing 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. 5 is shown as associated with the information, Land Availability. Land availability can be decomposed into two pieces of information, Land site and Land size Then the complex DR <Choose location, Land availability> can be decomposed into <Choose Location, Land site> and <Choose Location, Land size> respectively.

3.2 Meta-Model of Decisions

The key concept underlying the decision meta model of Fig. 6 is that of a decision parameter. Decision parameters reveal the factors that must be taken into consideration before a decision can be selected by the decision maker.

The decision to decision parameter relationship is M:N. A decision parameter must be associated with at least one decision. Similarly a decision must be associated with at least one decision parameter. **Dependent decision parameters** depend on other parameters for their existence whereas **independent decision parameters** determine 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.

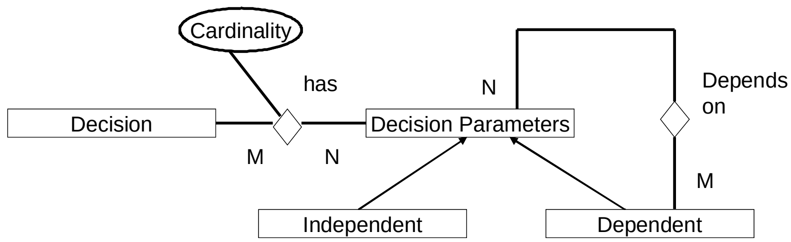


Fig. 6. Decision Meta Model

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.

3.2 Modeling Information

The information model in Fig. 7, shows three kinds of information, detailed, summarized or aggregates, and historical. Aggregate information is obtained as a summary by computing from simpler information. This is shown in Fig. 7, 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, by the attribute Period.

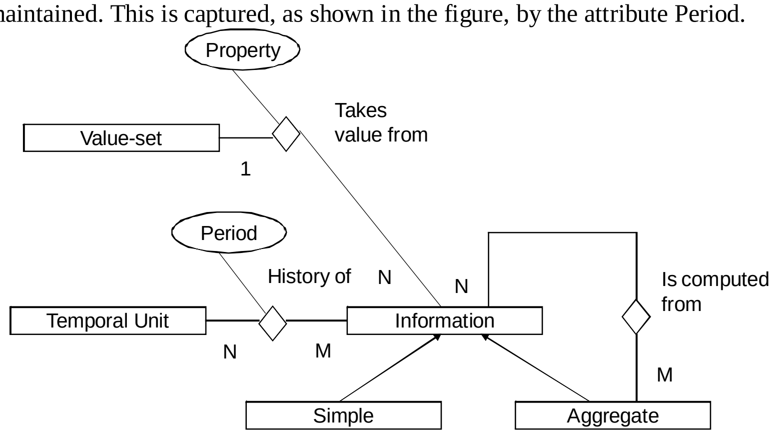


Fig. 7. Information Model in Data Warehouses showing three kinds of information.

Information is also associated with a value-set and takes on values from it. In Fig. 7 this association is called "Takes value from".

4 Comparison with Related Work

In traditional goal oriented requirements engineering, the aim is to specify system functionality. No support is provided in determining which of the many actions is to be performed. In our proposals, however, the focus is on the latter.

Our approach does not attempt to directly reach facts and dimensions unlike the database and ER driven approaches. Additionally, unlike these approaches, we can identify the required aggregate and historical information.

Goal oriented data warehouse development approaches of [6,7] and [16] reach data warehouse contents directly from goals without an explicit decisional stage. On the other hand, [15] recognizes the need to do further analysis from the decisional point of view. In contrast, we explicitly model the full decision making capability and associated information requirements.

Decision classification on the basis of time and planning horizon was proposed n GRAI grid [14]. The GRAI grid also provides an architecture of decisions of an organization. It provides a top level description of a system but does not aim to do requirements engineering for data warehousing.

Finally, our decisional environment is similar to the work system proposed in [17]. However, it addresses decision making, not operational information systems.

5 Conclusion

The notion of decision making implies the existence of a choice set from which the alternative that best meets organizational goals, is selected. These alternatives can be (a) managerial, for setting up the environment and (b) imperative, for providing the right service. Our emphasis is 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.

The ideas presented here have been tried out in a health scheme operating in India. Details can be obtained from the authors. Future work is centred 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, Vol. 18 No. 6, June 1992, pp. 483-497.
2. Lamsweerde, A. van, "Goal-Oriented Requirements Engineering: A Guided Tour" Invited Paper for RE'01 - 5th IEEE International Symposium on Requirements Engineering, Toronto, August, 2001, pp. 249-263

3. Sutcliffe, A., Maiden, N., "Bridging the Requirements Gap: Policies, Goals and Domains", Proc. IWSSD-7 - 7th Intl. Workshop on Software Specification and Design, IEEE, 1993.
4. Yu E. S. K., "Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering", Proc. Third IEEE Symposium on Requirements Engineering, 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, Proc. Of Workshop on Data Warehousing and OLAP (DOPLAP), USA,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 , 2000
8. Rilson F., Paim S., and Castro JFB, DWARF: An Approach for Requirements Definition and Management of Data Warehouse Systems, Proceeding of the 11^{th} IEEE International Requirements Engineering Conference 1090-9, September 08 - 12, 2003.
9. Frendi, M., Salinesi, C.: Requirements Engineering for Data Warehousing, Proceedings of Workshop on REFSQ, 75-82, 2003
10. Prakash N., Gosain A., Requirements Driven Data Warehouse Development, CAiSE Short Paper Proceedings, 13-17, 2003
11. Prakash N, Singh Y, Gosain A., Informational Scenarios for Data Warehouse Requirements Specification, Conceptual Modeling-ER'2004, Atzeni P., Chu W., Zhou S., Ling T K(eds.) LNCS 3288, Springer, 205-216, 2004
12. Winter R., Strauch B., Information Requirements Engineering for Data Warehouse Systems. ACM Symposium on Applied Computing, Cyprus, 1359-1365, 2004
13. Prakash N., and Gosain A., An Approach to Engineering the Requirements of Data Warehouses, Requirements Engineering Journal, Springer, 13 (1), 49-72, 2008
14. Carrie AS and Macintosh R, An Assessment of GRAI Grids and their use in the Strathclyde Integration Method, production Planning and Control, 8, 2, 106-113, 1997
15. Golfarelli, M., Rizzi, S., Designing the Data Warehouse: Key Steps and Crucial Issues" in Journal of Computer Science and Information Management, Vol. 2. No.3, 1999.
16. Bonifati A.. Cattaneo F.., CeriS., A. Fuggetta, and S. Paraboschi. Designing Data Marts for Data Warehouses. ACM Trans. Softw. Eng. Methodol., 10(4):452–483, 2001.
17. Alter S, A General yet Useful Theory of Information Systems, Comm.of Association for Information Systems, 1, 13, 1-68, 1999

Closing the User-Centric Service Coordination Cycle

Hans Weigand[1], Paul Johannesson[2], Birger Andersson[2], Maria Bergholtz[2], Jeewanie Jayasinghe Arachchige[1]

[1] Tilburg University, P.O.Box 90153,
5000 LE Tilburg, The Netherlands
H.Weigand@uvt.nl, J.JayasingheArachchig@uvt.nl
[2] Royal Institute of Technology
Department of Computer and Systems Sciences, Sweden
pajo,ba,maria@dsv.su.se

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 , service description

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 [1] 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 [7]. In this paradigm, software resources such as (REST or SOAP) *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. [2]).

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 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.

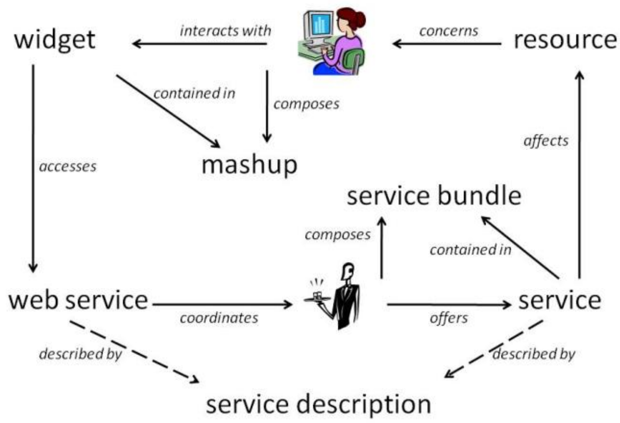


Fig. 1 User-centric service coordination cycle

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, [13] 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.

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.

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 the conflict prevention and conflict detection functionality described above. 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 [9] 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.

To arrive at rigorous and relevant research results, we use Peffers' design science phases [12]. The *problem identification and motivation* has been stated. Our *solution objective* is to develop a coordination service description language based on REA (without addressing a particular syntactic style, e.g. OCL or OWL). In section 2, we work out how REA represents services and the coordination of services. On the basis of that we show in section 3 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*).

2. Coordination Services in REA

2.1 REA and Capacity Planning

The Resource-Event-Agent (REA) ontology was first formulated in [9] and has been developed further, e.g. in [14,4,8]. The following is a short overview of the core concepts of the REA ontology based on [16].

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 [5]. Agents participate in events from inside (the primary perspective of the model) or outside.

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, give and take, 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 [11].

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:

$$\text{Capacity}(a,t) = \text{card}(R)$$
$$R = \{r\text{: resource} \mid \text{typify}(r,t) \wedge (\exists x\text{:agent received}(a,x,r) \vee \exists s\text{:reservation }(\text{give}(x,s) \wedge \text{take}(a,s) \wedge \text{specify}(s,t) \wedge \text{reserve}(s,r))\}$$
$$\text{Reserved}(a,t,d) = \textstyle\sum c\text{: card}(s,c), \quad s \in \text{RS}(a,t,d) \text{ where}$$
$$\text{RS}(a,t,d) = \{s\text{: reservation} \mid \exists x,a\text{:agent give}(a,s) \wedge \text{take}(x,s) \wedge \text{specify}(s,t) \wedge \text{date}(s,d)\}$$
$$\text{Available}(a,t,d) = \text{Capacity}(a,t) - \text{Reserved}(a,t,d)$$

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.

2.2 Coordination services

Coordination services are defined in [16] 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 both cases. 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 fulfilled by the exchange event, but in complex processes there are many more. The following two reoccur quite often, especially when services are concerned: appointment and reservation. The reason for that is simply 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.

Using REA 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 give, take and fulfill commitments.

We assume that for all coordination objects 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” [3,6]. 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.

In standard REA, a reservation is a relationship between a commitment and a resource. In the following, 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 [15]. So although the reservation has some costs in the form of less operational discretion, it increases the total value for both customer and provider.

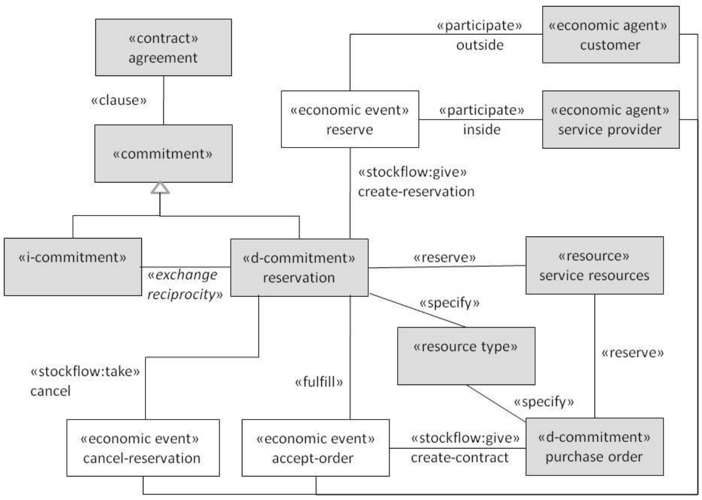


Fig. 2 REA Application Model for reservations

The REA application model in Fig. 2 contains and relates two coordination objects: reservation and purchase order. The reservation is 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. Quite often, the commitment specifies a resource type only and the allocation of the specific resource is done later. According to REA, there is exchange reciprocity between commitments. This reciprocity leads to dependencies between commitments that must be managed properly by the coordination services. The contract can be explicit or implicit. It may contain additional commitments, usually conditional ones (terms), such as a penalty for non show-up. In line with [8] 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 purchase order is the service exchange event. Since this could be seen as the ultimate objective of the reservation as well, we define a fulfill* relationship being the transitive closure of fulfill-relationships.

3. Service Description and Conflict Detection

3.1 Service Description Using REA

Using the REA ontology, service descriptions can be developed for coordination services either in the form of REA events 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*
Fulfil(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 shorthands 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 [10], 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 [16]: 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.

3.2 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 planned events 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.

Let *M* be the set of resources relevant to *U*. To determine the contents of M, the set E_u of committed events for *U* is calculated first as follows:

$E_u = \{e\text{: event} \mid \exists c\text{: commitment}(c) \wedge \text{fulfill*}(e,c) \wedge \text{participate}(e,U)\}$

Then

$M = \{m \mid \exists e \in E_u\text{: stockflow}(e,m) \vee \text{participate}(e,m)\}$ (resources and agents involved, as far as known)

For some $m \in M$, the context E_m contains the committed events that involve *m*. Note that $U \in M$. 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”:

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 typify(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)

$\forall e_1, e_2 \in E_m\ e_1.time \cap e_2.time = \emptyset$

Another general constraint is that the resource can be at only one place at a time, and needs time for moving:

$\forall\ e_1, e_2 \in E_m : e_1.time.end = e_2.time.start \Rightarrow e_1.location = e_2.location$

$\forall\ e_1, e_2 \in E_m : next(e_1, e_2) \wedge e_1.location <> e_2.location$

$\Rightarrow (\exists e_i \in E_m : e_1 < e_i < e_2\ \wedge e_i.type=$ «move» $\wedge\ e_1.object = m$

$\wedge\ e_i.destination = e_2.location)$

where $next(e_1, e_2)$ 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.

References

1. Anderson, C. The Long Tail: How endless choice is creating unlimited demand. Random House Business Book, London (2006).
2. Benatallah, B., Casati, F., and Toumani, F. Web Service Conversation Modeling: A Cornerstone for E-Business Automation. IEEE Internet Computing 8, 1(2004).
3. Dietz. J. Enterprise Ontology - Theory and Methodology. Springer, Berlin (2006).
4. Geerts, G., McCarthy, W.E. An Accounting Object Infrastructure For Knowledge-Based Enterprise Models. IEEE Int. Systems & Their Applications, pp. 89-94, (1999).
5. Gailly, F., Laurier, W., Poels, G. Positioning and Formalizing the REA enterprise ontology. Journal of Information Systems 22, 219-248 (2008).
6. Goldkuhl, G. Action and media in interorganizational interaction. Comm.. ACM 49, 5 pp.53-57 (2006).
7. Hoyer, V., Stanoevska-Slabeva, K. Towards a reference model for grassroots enterprise mashup environments. Proc. ECIS 2009 (2009).
8. Hruby, P. Model-Driven Design of Software Applications with Business Patterns. Springer Verlag (2006).
9. McCarthy W.E., The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment. The Accounting Review (1982).
10. McCarthy, J., Hayes, P.J. Some philosophical problems from the standpoint of artificial intelligence. Machine Intelligence, 4:463–502 (1969).
11. O'Leary, D., REAL-D: A Schema for Data Warehouses, Journal of Information Systems, Volume 13, Number 1, pp. 49-62 (1999).
12. Peffers, K., Tuunanen, T., Rothenberger, M., and Chatterjee, S. A Design Science Research Methodology for Information Systems Research. Journal of Management Information Systems, 24(3), 45-77 (2008)
13. Pistore, M., Traverso, P., Paolucci M., Wagner, M. From Software Services to a Future Internet of Services. In: G. Tselentis et al (eds), Towards the Future Internet. IOS Press, (2009).
14. UN/CEFACT Modelling Methodology (UMM) User Guide. Available at http://www.unece.org/cefact/umm/UMM_userguide_220606.pdf (2003)
15. Weigand, H., Heuvel, W.J.A.M. van den. A conceptual architecture for pragmatic web services. In M. Schoop, A. de Moor, & J. Dietz (Eds.), Proc. 1st Int. Conf. on the Pragmatic Web (pp. 53-66). Heidelberg: Springer-Verlag (2006).
16. Weigand H., Johannesson, P., Andersson, B., Bergholtz Value-based Service Modeling and Design: Toward a Unified View of Services. Proc. CAiSE'09, Springer, pp.410-424 (2009).

Towards a Reference Model for SOA Governance

C. Ott[1], A. Korthaus[2], T. Böhmann[3], M. Rosemann[2], H. 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 address this drawback by presenting selected elements of 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 an 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]. In their seminal work, [2] define IT governance as the process of "specifying the decision rights and accountability framework to encourage desirable behaviour in the use of IT." 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. To date, however, no widely accepted framework for SOA governance has emerged [3]. Given that the lack of a comprehensive governance approach has been cited as the most common reason for failures of post-pilot SOA projects [4], work in this area is highly relevant.

While definitions differ considerably, most authors agree on the basic elements a governance framework should address, namely the organisational structure, processes, policies and metrics [5], [6]. To provide a working definition for the rest of this paper, we build on [3] and [7] by specifying:

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 SOA is always 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 section 2, we point to related work and explicate our research approach. Section 3 outlines selected activities along the service lifecycle. Section 4 describes selected roles and the assignment of responsibilities. The paper concludes with summary and further research opportunities in section 5.

2 Related work and research approach

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 [2] and well-received frameworks such as Cobit [1] and ITIL [8] are the most prominent examples, the OECD Principles of Corporate Governance are among the most influential guidelines in the area of corporate governance [9]. For a detailed discussion of related work on SOA governance, such as the body of academic literature and approaches published by IT vendors and open standards organisations like OASIS, OMG and The Open Group, and how it relates to the approach presented here, please refer to our extended report in [10].

Starting from the existing knowledge base, we analysed the widely-used IT governance frameworks Cobit and ITIL and provided an initial evaluation of their utility in a case study in order to derive the core of the SOA governance framework. 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 there. Furthermore, the 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, focusing on the identification of main concepts, and conducted a series of interviews with 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 best known 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.

3 The service lifecycle

3.1 Overview

Cobit and ITIL are very detailed and widely used frameworks that propose a large number of best practices 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. 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. Due to space constraints, this paper covers only three of these layers: service portfolio-, service project- and service operation management.

While acknowledging that there is a broad variety of definitions, we agree with [12] 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 [13]. 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. 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.

A significant amount of research has been published regarding the lifecycle of a single service (cf. [14] for a comprehensive overview). 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, [14] mention a negotiation phase. The latter is primarily relevant if a service or part of its sub-services are provided or sourced externally.

3.2 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. 1 gives an overview and shows how management layers, lifecycle stages and activities are interrelated.

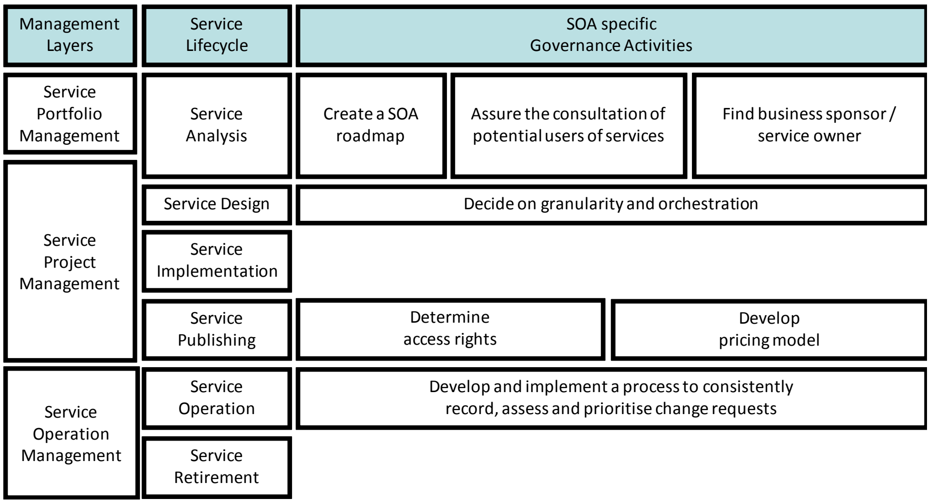


Fig. 1. Interrelationship of management layers, lifecycle stages and selected activities.

3.2.1 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 [15] 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. Please refer to [10], where we discuss how Cobit activities need to be adapted to a SOA environment using the examples "Create an SOA roadmap", "Assure consultation of potential users of services" and "Find business sponsor / service owner". As an example, we pick out the last activity here:

- *Find business sponsor / service owner:* An important step refers to the issue of funding [16]. Adapting services to the requirements of different users will be more expensive than developing them for the sole purpose of a single user [17]. 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 a centralised committee could ultimately own the service as well and should therefore be provided with a dedicated budget.

3.2.2 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. By focusing on the major differences compared to traditional software development, we identify and discuss in [10] the following particularly interesting activities: “Decide on granularity and orchestration”, “Determine access rights” and “Develop pricing model”. Other important aspects include issues regarding service contracts and business object governance. Let’s take an example from “service publishing”:

- *Develop pricing model:* Among traditional IT cost accounting methods (for an overview see [18]), activity-based costing is seen as one of the most effective representatives [19]. Under the SOA reuse paradigm, where services are shared among several business units or departments, new mechanisms like negotiation [18] 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.

3.2.3 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 as well as in the operation management phase. In [10], we discuss in more detail the activity of consistently recording, assessing and prioritising change requests.

4 Roles and assignment of responsibilities

We conducted a literature review and a comprehensive content analysis of more than 300 published job profiles at Seek.com (keyword: “SOA”), focusing on roles that are either not mentioned in the IT Governance frameworks or whose responsibilities change significantly under a SOA paradigm. Among these are the roles of Business Analyst, Enterprise/Business Architect, Project Manager, Service Owner and Service Librarian. Due to space constraints, we refer the reader to our discussion in [10] and briefly discuss the following two roles here as an example:

- *Service Owner* [11], [15]: 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* [20]: 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.

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.

5 Summary

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. 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) (2007). Cobit 4.1. Isaca, Rolling Meadows.
2. Weill, P. and J.W. Ross (2004). IT Governance: How Top Performers Manage IT Decision Rights for Superior Results. Harvard Business School Press, Boston, Mass.
3. Bernhardt, J. and D. Seese (2008). 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, Dec. 1.
4. Malinverno, P. (2006). Service-Oriented Architecture Craves Governance. Gartner Research, 20. Jan. 2006, 1-6.
5. Schelp, J. and M. Stutz (2007). SOA-Governance. HMD-Praxis der Wirtschaftsinformatik, 253, 66-73.
6. Bell, M. (2008). Service-Oriented Modeling: Service Analysis, Design, and Architecture. Wiley, Hoboken, NJ.
7. Dodani, M.H. (2006). Who Took the Cookie from the Cookie Jar? Journal of Object Technology 5(4), May-June, http://www.jot.fm/issues/issue_2006_05/column3/
8. ITIL (2007). IT Infrastructure Library v3. OGC, TSO, http://www.itil-officialsite.com/home/home.asp
9. OECD (2004). OECD Principles of Corporate Governance. Org. for Economic Co-Operation and Development. http://www.oecd.org/dataoecd/32/18/31557724.pdf
10. Ott, C., Korthaus, A., Böhmann, T., Rosemann, M. and Krcmar, H. (2010): Towards a Reference Model for SOA Governance (Extended Version). Working Paper. http://eprints.qut.edu.au/31057/1/Ott-Korthaus-Boehmann-Rosemann-Krcmar-SOA_Governance.pdf
11. Malinverno, P. (2006b). Sample Governance Mechanisms for a Service-Oriented Architecture. Gartner Research, 27. April 2006, 1-5.
12. Archer, N.P., and F. Ghasemzadeh (2004). Project Portfolio Selection and Management. In Morris, P.W.G. and J.K. Pinto (eds.). The Wiley Guide to Management Projects. John Wiley & Sons, NY.
13. Cooke-Davies, T. (2004). Project Success. In Morris, P.W.G. and J.K. Pinto (eds.). The Wiley Guide to Management Projects. John Wiley & Sons, New York.
14. Riedl, C., Boehmann, T., Rosemann, M. and H. Krcmar (2008). Quality Management in Service Ecosystems. Information Systems & E-Business Management, Springer, 1-23.
15. Deb, M., Helbig, J., Kroll, M. and A. Scherdin (2005). Bringing SOA to Life: The Art and Science of Service Discovery and Design. SOA Web Services Journal, 27, 42-47.
16. Schepers, T.G.J., Iacob, M.E. and P.A.T. Van Eck (2008). A Lifecycle Approach to SOA Governance. In Proc. of the 2008 ACM Symp. on Applied Computing (SAC ’08), Brazil, ACM, 1055-1061.
17. Ren, M. and K. Lyytinen (2008). Building Enterprise Architecture Agility and Sustenance with SOA. Comm. of the Association for Information Systems, 22 (March), 75-86.
18. Gerlach, J., Neumann, B., Moldauer, E., Aro, M. and D. Frisby (2002). Determining the Cost of IT Services. Comm. of the ACM, 45(9), Sept., 61-67.
19. Ross, J.W., Vitale, M.R. and C. M. Beath (1999). The Untapped Potential of IT Chargeback. MIS Quarterly, 23(2), 215-237.
20. Kajko-Mattsson, M., Lewis, G.A. and D.B. Smith (2007). 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 ’07), 7-12.

Modeling Deployment of Enterprise Applications

Susanne Patig[1]

[1] 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 these 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 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]. During deployment planning, applications must be assigned to a given IT infrastructure in such a way that the assignment is valid. The approach presented here is capable of modeling such assignments and checking their validity.

Section 2 summarized the requirements of deployment planning in data centers; Section 3 sketches the relevant existing approaches. In Section 4, a new domain-specific language called ADeL (*A*pplication *De*ployment *L*anguage) is proposed and applied to a real-world case. The last section contains critical reflections 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 people involved in the installation process were interviewed. The complete description of the cases can be found in [16]; for brevity, 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:

[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 RAM and 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 involve 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 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 concepts should be used because 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 (model-driven development by adding (meta-) model transformations). 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, checking modeled deployment plans for their validity [Rq10] prior to installation was rated as an important benefit of modeling.

The next section analyses whether or not the existing approaches in the field of deployment satisfy the elicited requirements.

3 Existing Approaches in the Field of Deployment

Software deployment has been largely neglected in academic discussion. Fig. 1 arranges the existing *approaches* (i.e., tools, modeling languages and standards) in a portfolio: The axes reflect the focus of the approaches and the kind of support they offer, respectively. The sizes of the bubbles illustrate the covering realized by each approach.

---

[1] All names of products are trademarks, service marks or registered trademarks of the respective companies.

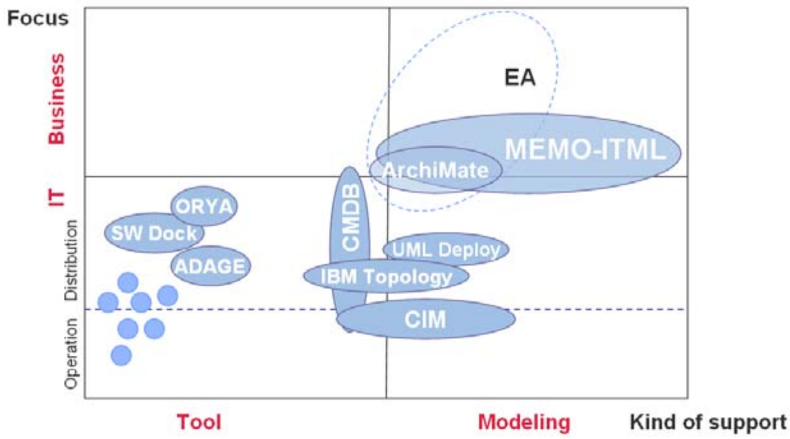


Fig. 1. Existing approaches in the field of deployment.

The lower left quadrant of the portfolio contains tools that automate all deployment activities (ORYA [12], Software dock [8]) or only installation (ADAGE [10]); the unlabeled bubbles represent proprietary tools. In this paper, these approaches are neglected as they do not satisfy the modeling requirement [Rq6]. Moreover, ADAGE and the proprietary tools are not general [Rq9].

The approaches that model deployment focus on business or IT. The business focus is typical for approaches stemming from the field of EA (ArchiMate [11]) or go even beyond (MEMO ITML [28]). Both approaches provide constructs to express applications and system software [Rq2] as well as hardware [Rq1] and the corresponding assignments [Rq3]. However, as enterprise architecture aims at aligning business and IT, the resulting models are hardly extensible for purposes beyond description [Rq8], constraints [Rq4] and choices [Rq5] cannot be represented, and the validity of the modeled deployment scenarios [Rq10] cannot be checked.

UML deployment diagrams [14], IBM topologies [13], [16] and the Common Information model (CIM) [6], which is implemented in configuration management databases (CMDB), represent the IT view on deployment. These approaches satisfy the requirements [Rq1] to [Rq3] related to expressive power (minor restraints refer to the representation of artifacts) as well as the requirement of extensibility [Rq8]. However, gaps exists for the other requirements: The UML relies on the OCL [11] to specify any kind of *deployment constraints* [Rq4], whereas IBM topologies support a limited set of constraint types by particular constructs [13]. Deployment choices [Rq5] are not covered by the existing approaches, except for an indirect modeling with IBM topologies (see [16]). Measured by the number of constructs, none of the approaches is simple [Rq7]. Because of being standards, UML deployment diagrams and CIM are general [Rq9], IBM topologies and CMDBs are not. Only IBM topologies include a (restricted) way to check the validity of deployment plans prior to installation [Rq10].

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 checking the validity of deployment models [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. 2 as well as a set of 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.

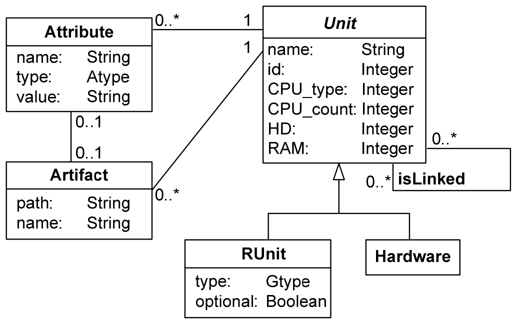


Fig. 2. Abstract syntax of the ADeL metamodel.

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`.

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`), disjunctive (`GType = O`) or exclusive (`GType = X`), i.e., all/at least one/one and only one of the grouped `RUnits` is to be deployed. Often such groups are conceptual, i.e., they structure ADeL models or prepare 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 OCL invariants of the ADeL metamodel are independent of deployment, namely: (1) Exactly one root node of the type `RUnit` must exit. (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 corresponds to the graph provided by the EMF.Edit framework [4]; see Fig. 4 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). 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 [11] and explained in the following. Due to space limitations, the OCL statements are not given here, but can be requested from the author of this paper.

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`. 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 hardware unit must exist.

Hardware constraints, which are specified by the invariants `[HD]`, `[RAM]`, `[CPU_count]` and `[CPU_type]`, guarantee that the aggregations of prerequisites 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 of `hardware`, and navigation occurs along the reverse deployment path, i.e., from the leafs of an ADeL model to its root. 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 rely on help functions for specific *aggregations* along the deployment path, i.e., (1) to sum up the required HD size (help function `aggrHD()`, (2) to find the maximum required RAM size or CPU count or (3) to check the equality of the required CPU type.

These predefined OCL invariants are implemented with the tool *Topcased* (http:// www.topcased.org) and must be evaluated for each ADeL model (see Fig. 3). Topcased can also be used to implement additional, deployment-specific OCL constraints.

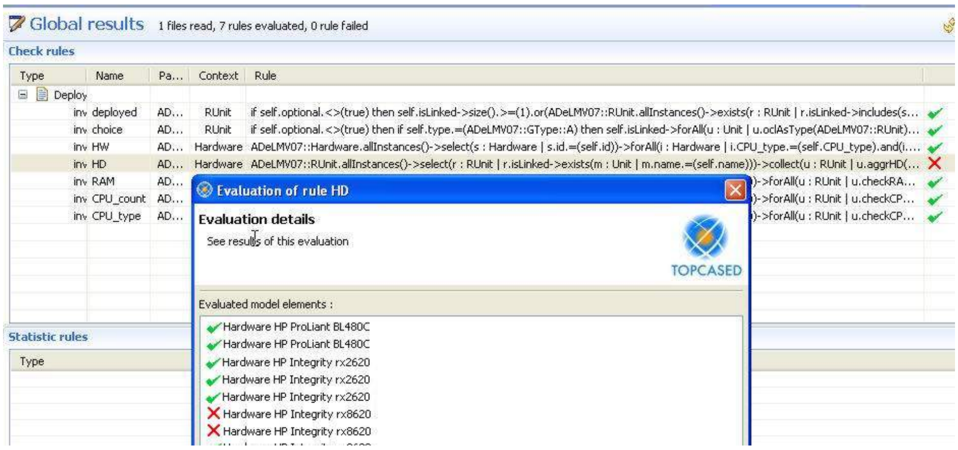


Fig. 3. Evaluation of the ADeL OCL constraints for the ADeL model of Fig. 4.

4.3 Application Example

Fig. 4 depicts the ADeL model for the deployment of SAP SCM (a real-world case investigated in [16]). SAP SCM, the root node, consists of several `RUnits` (the 'SAPKernel', a database instance 'DBSID', the LiveCache 'LID' and the optional optimizer 'OptID'). The `RUnit` 'Install' expresses installation prerequisites (installation media, JRE). The 'SAPKernel' is a conjunctive `RUnit` (`GType = A`) since both the global instance 'SAPSID' and the central instance 'DBSID' as well as the C++ Runtime environment must be installed. The software products that realize the `RUnits` 'DBSID', 'LID' and 'OptID' must be chosen from a set [Rq5]; thus, these `RUnits` are exclusive groups.

The 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 additional `attribute` 'SWAP' related to the `RUnit` 'SAPKernel' expresses that additional 20 Gigabyte (GB) of SWAP space are needed. (All sizes are specified in GB in this paper). Moreover, the `RUnit` 'SAPSID' is associated with an `artifact` that specifies the location (`path`) of the directory `/sapmnt`.

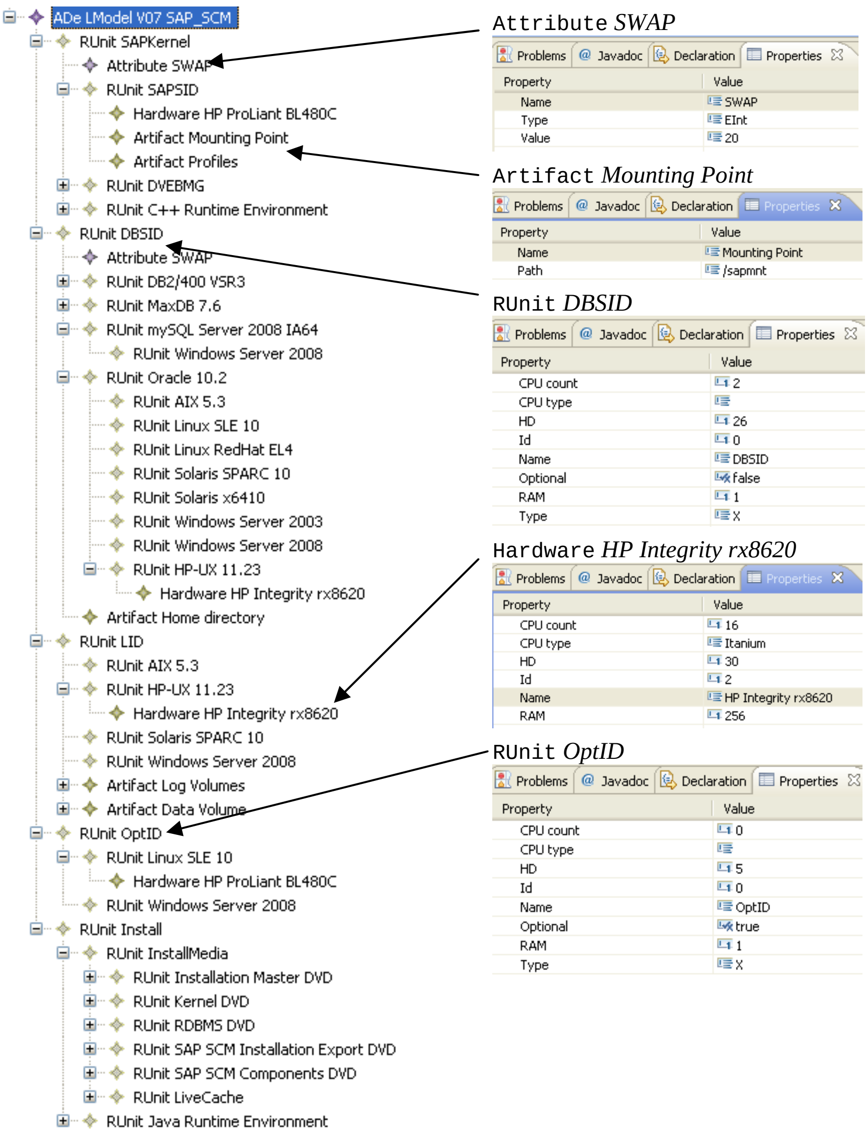


Fig. 4. Concrete syntax of the ADeL metamodel for the example of SAP SCM (extract).

5 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 [9]. Moreover, an extensive type vocabulary becomes burdensome when using the OCL: Deviating types of units must be casted;

recursive navigation along deployment paths is not possible if the link types are distinct. For that reason ADeL does not differentiate between link types.

Though the OCL is a natural choice to express constraints [Rq4] in the field of model-driven development, its appropriateness for the purpose can be doubted: First, as explained above, it affects the ADeL abstract syntax. Secondly, the ADeL hardware constraints require reverse navigation along deployment paths. This can only be achieved by the predefined operation `allInstances()`, which increases the worst case complexity of OCL evaluation [1]. 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. Nevertheless, a goal of my future research consists in replacing the OCL invariants by another formalism that is capable of handling constraints, e.g., constraint solving techniques. Other topics for future research are the implementation of a more sophisticated editor as well as the definition of transformations to generate installation guidelines and system configurations from ADeL models.

References

[1] Altenhofen, M., Hettel, T., Kusterer, S.: OCL support in an industrial environment. In: Proc. Workshops and Symposia at MoDELS 2006, pp. 169-178. Springer (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 ed., Worth, New York (2000)

[4] Budinsky, F., Steinberg, D., Merks, E., Ellersick, R., Grose, T.J.: Eclipse Modeling Framework: A Developer's Guide. Boston et al., Addison-Wesley (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] DMTF: Common Information Model (CIM) Standards. CIM Schema: Version 2.24.0 http://www.dmtf.org/standards/cim/cim_schema_v2240/

[7] Frank, U. et al.: ITML: A domain-specific modeling language for supporting business-driven IT Management. DSM Forum 2009 http://www.dsmforum.org/events/DSM09/Papers/Heise.pdf

[8] 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 (1999)

[9] Kruse, S. et al.: Decoupling Models and Visualisations for Practical EA Tooling. In: Proc. of the 4th Workshop on Trends in EA Research (TEAR 2009). Springer (2010)

[10] 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)

[11] Lankhorst, M. et al.: Enterprise Architecture at Work: Modeling, Communication, Analysis. Berlin et al.: Springer (2005)

[12] Lestideau, V., Belkhatir, N.: Providing highly automated and generic means for software deployment process. In: Proc. of the 9th European Workshop Software Process Technology (EWSPT 2003). Springer (2003)

[13] Makin, N.: Deployment modeling in Rational Software Architect Version 7.5, Part I & II. http://www.ibm.com/developer¬works/rational/library/08/{1202|1230}_makin/

[14] Object Management Group (OMG): Unified Modeling Language: Superstructure, Version 2.2, formal/2009-02-02, http://www.omg.org/ (2009)

[11] OMG: UML 2.0 OCL Specification, formal/2006-05-01, http://www.omg.org/ (2006)

[16] Patig, S., Herden, S., Zwanziger, A.: Modeling Deployment of Enterprise Applications - Cases and Conclusions. Preprint No. 224, University of Bern, IWI (2009)

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 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 center. 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 centers. An estimation is reported in [10] that the

US servers and data centers 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 similar 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 application 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 using complex scheduling mechanisms.

The report [10] contains a forecast of doubling the energy consumption estimated in 2006 within five years, and it indicates that there is a potential of reducing such consumption with existing technologies and design strategies by 25 percent or more. However, many current research directions have shown that such improvement can be significantly increased considering a number of potential improvements in several aspects of a data center. Despite the big effort that has been put for assessing energy efficiency of IT service centres aiming at the reduction of energy costs [3], 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 the dynamic management of servers according to workload and servers consolidation and virtualization; the development of low power techniques at IT component level; and 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 centers are just starting to be considered, such as for instance in the EU best practices for data centres [4].

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, also information about IT resources can 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 [8, 7], 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 [5], 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.

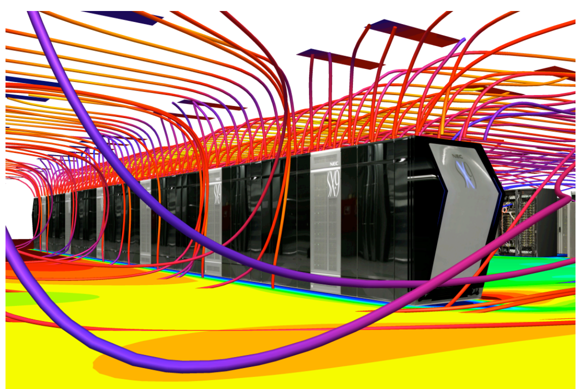


Fig. 1. Sample of simulation in the design phase

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 (e.g. Computational Fluid Dynamics simulations as shown in figure 1), 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 exploiting 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 run-time, 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 we illustrate the general approach to energy efficiency in GAMES, while in Sections 3 and 4 we discuss the co-design approach and the adaptive run-time environment respectively.

2 The GAMES approach

In the data and service centre, we envision the energy-aware design and management of service-based information systems and their IT infrastructure, supported by an adaptive SBA (Service Based Architecture), in which it is possible to dynamically modify service compositions driven by Service Level Agreements, covering Quality of Service. The goal is to realise a self-adaptive data and service centre architecture across all kind of offered resources ranging from data over computing up to the service layers. The run-time management continously balance the agreed service contracts and derive the necessary measures needed based on the monitored values (energy consumption, load situation, risk to fail on an SLA, etc.) as shown in the conceptual architecture in Figure 2.

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). In order to realise this architecture, three major building blocks have been identified.

The **Energy Sensing and Monitoring Infrastructure** (ESMI) provides services to interact with the energy grid, with the environment monitoring infrastructure and with the data center resources, for energy consumption and 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.

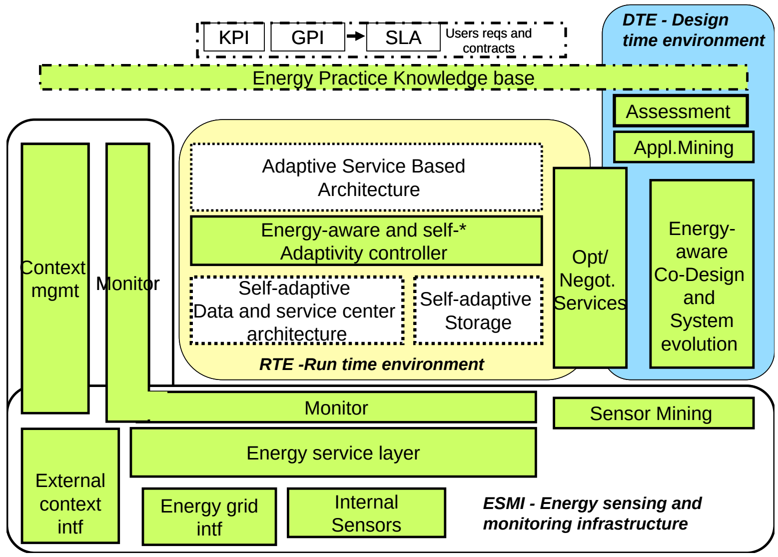


Fig. 2. GAMES architecture

The ESMI will be partially based on the energy service layer being developed in BeAware [6]. 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 including functionalities for event analysis, based on 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, and 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. 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.

3 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) control strategies to evaluate, optimise, and control services and data at run-time on multiple time scales and adapt them at run time; c) 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, 2], we advocate that considering the energy consumption dimension, different and more efficient design choices could be performed.

Examples of energy-aware co-design include not only minimized 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) or an evaluation of the impact on needed cooling capacities based on different load scenarios of servers, but also a focus on business process analysis of core activities-services-data as shown in [9].

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 processing requirements, data requirements and dependencies in their tasks, 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, but also that of the IT infrastructure, with an efficient use of the available resources as the main goal.

4 Energy efficiency at run time

A new approach for developing an energy-aware adaptive mechanism at run time will be defined and implemented. 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. A multi-layer feedback architecture will be considered for run-time controlling of system's 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 different 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. The local loop controllers are used to locally optimize the IT Infrastructure server specific energy consumption, without considering the whole system state. The local 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. Using the local control loops a optimal energy consumption will be obtained for each IT Infrastructure specific component. This optimum will be communicated together with the component specific data as events to the global system controller. The global controller receives the energy-related information from each specific local loop and from the environment monitoring infrastructure as well as the performance-related information from the system's service layer in order to take adaptation decisions to enforce and realize the Key Performance Indicators (KPI) and Green Performance Indicators (GPI) defined in the co-design phase. The global control loop decision may include the execution of the following examples of energy-aware context-based adaptivity actions: minimize the necessity of calling a remote service when one local similar service is available (minimize data/service transfer), minimize the substitution of services during maintenance, optimize the number of necessary backup operations, privilege the use of services that require low energy, etc.

To derive knowledge about the service center and its energy efficiency, the GAMES framework will integrate 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 e.g. "optimized 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 and the externally available information with the GAMES Knowledge Base will allow the selection of the most adequate adaptation patterns and profiles.

5 Conclusions

This paper has presented the GAMES approach to design and manage energy-efficient service centers. 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 [3]), 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 centers, on experimental settings.

Acknowledgments This work has been partially supported by the European Commission within the GAMES project funded under EU FP7.

References

1. J. Almeida, V. Almeida, D. Ardagna, C. Francalanci, and M. Trubian. Managing energy and server resources in hosting centers. In *Proc. ICAC*, 2006.
2. D. Ardagna, C. Cappiello, M. Lovera, B. Pernici, and M. Tanelli. Active energy-aware management of business-process based applications. In P. Mähönen, K. Pohl, and T. Priol, editors, *ServiceWave*, volume 5377 of *Lecture Notes in Computer Science*, pages 183–195. Springer, 2008.
3. Cristian Belady ed. Green grid data center power efficiency metrics: PUE and DCiE. 2008.
4. EU Stand-by Initiative. 2010 best practices for the eu code of conduct on data centres. Technical report, December 2009.
5. A. M. Ferreira, K. Kritikos, and B. Pernici. Energy-aware design of service-based applications. In *ICSOC 2009*, Nov. 2009.
6. L. Gamberini, L. G. Jacucci, A. Spagnolli, C. Bjorkskog, D. Kerrigan, A. Chalambalakis, L. Zamboni, G. Valentina, N. Corradi, P. Zappaterra, and G. Bosetti. Technologies to improve energy conservation in households: The users' perspective, Maastricht. In *First European Conf. Energy Efficiency and Behaviour*, Oct. 2009.

7. P. Hasselmeyer, B. Koller, L. Schubert, and P. Wieder. Towards SLA-Supported Resource Management. In *High Performance Computing and Communications - Second International Conference, HPCC 2006, Munich, Germany, September 13-15, 2006, Proceedings*, volume 4208 of *Lecture Notes in Computer Science*, pages 743–752. Springer, 2006.
8. B. Koller and L. Schubert. Towards autonomous SLA management using a proxy-like approach. *International Journal of Multiagent and Grid Systems*, 3:313–325, 2007.
9. N.-H. Schmidt, K. Erek, L. M. Kolbe, and R. Zarnekow. Towards a Procedural Model for Sustainable Information Systems Management. In *HICSS' 09: Proceedings of the 42nd Hawaii International Conference on System Sciences*, pages 1–10, Hawaii, USA, 2009. IEEE Computer Society.
10. 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.

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 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 and new user-defined applications can be 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, users are nowadays able to not only manage and share their own data, but also integrate a wide range of external data and services. At the same time, users 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. A Web-based pluggable interface architecture generates the interfaces automatically based on default or selected templates which can easily be customised.

Section 2 provides the background to our work and an overview of the approach is given in Sect. 3. Section 4 describes the development process, while implementation details are given in Sect. 5. Our approach is compared to related work in Sect. 6. Concluding remarks are given in Sect. 7.

2 Background

Personal information management (PIM) systems proposed in research tend to either use a predefined PIM domain model e.g. [1, 2], or work according to a "no-schema" or "schema-later" approach e.g. [3, 4]. On the interface level, they mostly offer a generic browser where a user can browse via associations. In Haystack [4], 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.

Web 2.0 has 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 applications offered by social networking sites such as Facebook. 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 control over their own data and also information fragmentation [5, 6].

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

3 Approach

Information components are intended to be the basic units of reuse in information systems at both the schema and data level. An information space consists of a set of information components where each component can contain both metadata and data. If an information component consists of only metadata, reuse is only at the schema level to support the design of an information space. Optionally, an information component may contain data as well as metadata which allows the reuse of data. Information components can be composed from other components as shown in Fig 1.

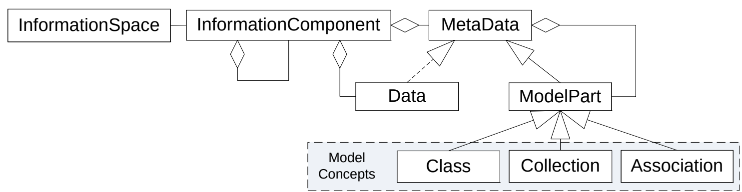


Fig. 1. Information components metamodel

Our prototype was developed using an object database based on the OM model [7] 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 components defined by other users based on a global component registry.

An application consists of an information component together with an associated user interface (UI). To create an 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.

4 Application Development Process

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. Figure 2 gives an overview of the steps involved as supported by the Application Manager integrated into our prototype system PIM 2.0.

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 modeling the application domain reusing existing component parts and/or specifying new domain concepts. Note that an information component can have arbitrarily complex models, but may also represent a single domain concept. Figure 3 shows a screenshot of component composition in the Application Manager.

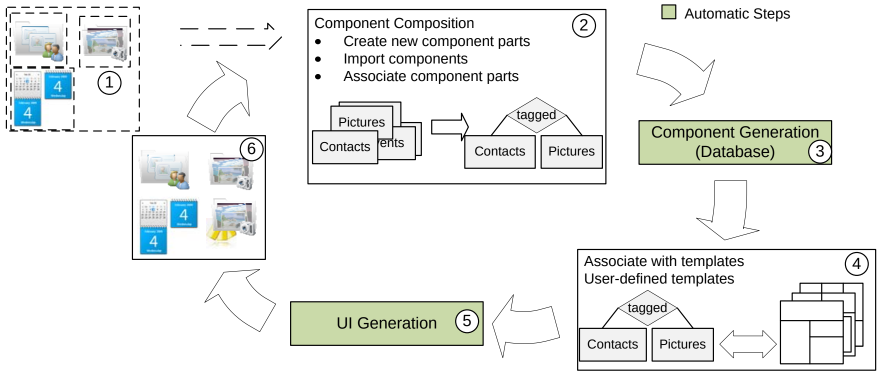


Fig. 2. Process of composing information components

Fig. 3. Application Manager

We assume the reuse of a *Contacts* component and a *Pictures* component. The following assumes a combined definition of collections and types.

```
Contacts{
    contacts(name:string, birthday:date, phone:set, address:set)
}
Pictures{
    pictures(picture:uri,  caption:string)
    albums(name:string)
    picturesInAlbums(picture:ref, album:ref)
}
```

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 by using the menu on the left. As shown in Fig. 3, a *PictureTagging* component can be created by reusing `contacts` from the *Contacts* component and `pictures` from the *Pictures* component and associating them with a `tagged` association. The textual representation of the new component is:

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

Collection and association names are qualified with their component names to ensure uniqueness. However, users can rename objects during composition in order to simplify the interface of the new component. Once defined, the component model is automatically created in the database ③.

To create the UI for the new component, it is associated with a structural template ④ which defines what should be displayed in terms of attributes and associated entities, the order in which these should be displayed and also the operations offered through context menus and buttons. To support the definition of the UI, there are standard templates for displaying collections and objects in read or write modes. Further, structural templates can be created automatically by 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 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 Implementation

The first step of implementing the PIM 2.0 system was to implement an object database that supported the concept of information components. 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 4 provides an overview of the PIM 2.0 architecture. 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 component manager is implemented as a metamodel extension module [8] of our object database Avon [9]. A metamodel extension module consists of three parts, namely metamodel concepts, CRUD classes to manipulate the metamodel concepts and an optional language extension. Through the metamodel extension mechanism, the information component concepts are represented as metadata in the database.

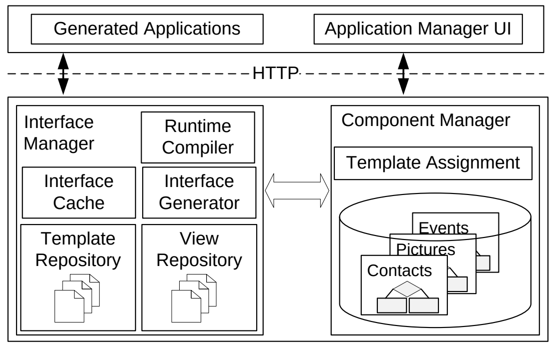


Fig. 4. Structure of PIM 2.0 system

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. The UI was implemented using the OpenLaszlo Web application framework. 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.

6 Discussion

Our approach combines the advantages of predefined, no-schema and schema-later approaches to PIM by offering users a set of 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.

While reuse in databases has been considered at the architectural level in terms of *c*omponent database systems [10] and also at the data level in terms of various forms of data integration services [11], 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 [12] 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.

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.

7 Conclusion

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 the more advanced Web users can create and share components with other users, while non-expert users can benefit from the expertise and experience of the community similar to collaboration evident in many Web 2.0 communities.

References

1. Gemmell, J., Bell, G., Lueder, R.: MyLifeBits: a Personal Database for Everything. Comm. ACM **49**(1) (2006)
2. Dong, X., Halevy, A.Y.: A Platform for Personal Information Management and Integration. In: Proc. CIDR 2005, Asilomar, CA, USA (2005)
3. 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)
4. Karger, D.R., Bakshi, K., Huynh, D., Quan, D., Sinha, V.: Haystack: A General-Purpose Information Management Tool for End Users Based on Semistructured Data. In: Proc. CIDR 2005, Asilomar, CA, USA (2005)
5. Leone, S., Grossniklaus, M., Norrie, M.C.: Architecture for Integrating Desktop and Web 2.0 Data Management. In: Proc. IWOOST 2008
6. Norrie, M.C.: PIM Meets Web 2.0. In: Proc. ER 2008, Barcelona, Spain (2008)
7. Norrie, M.C.: An Extended Entity-Relationship Approach to Data Management in Object-Oriented Systems. In: Proc. ER 1993, Arlington, Texas, USA (1993)
8. Grossniklaus, M., Leone, S., de Spindler, A., Norrie, M.C.: Dynamic Metamodel Extension Modules to Support Adaptive Data Management. In: Proc. CAISE 2010, Hammamet, Tunesia (2010)
9. Norrie, M.C., Grossniklaus, M., Decurtins, C., de Spindler, A., Vancea, A., Leone, S.: Semantic Data Management for db4o. In: Proc. ICOODB, Berlin, Germany (2009)
10. Dittrich, K.R., Geppert, A., eds.: Component Database Systems. Morgan Kaufmann (2001)
11. Halevy, A., Rajaraman, A., Ordille, J.: Data Integration: the Teenage Years. In: Proc. VLDB 2006, Seoul, Korea (2006)
12. Thalheim, B.: Component Development and Construction for Database Design. Data & Knowledge Engineering **54**(1) (2005)

A Goal-Oriented Requirements Engineering Method for Business Processes

Ken Decreus, Geert Poels

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

Abstract. The field of requirements engineering (RE) for business processes has grown during the last several years. As business processes are needed to fulfil organizational goals, the information captured in goal models provides a basis for designing business processes. Although research has started to explore how to transform goal models into business process models, current transformation methods need further research. This paper proposes a tool-supported method to model goals as part of the business requirements for business processes and to automatically generate business process design skeletons that respond to these business requirements.

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

1 Introduction

The information needs of an organisation set requirements for its information systems and the setting of goals, and the formulation of business strategies to achieve these goals, leads to business requirements for the business processes of the organisation. We define *business requirements* for business processes as the overall set of requirements that relate to business processes as given by the Business Motivation Model (BMM) [1] of OMG, such as vision, mission, goal, strategy, objective and tactic. More specifically, 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. 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. 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.

In a business process-centred organization, the architectural view on implementing business requirements through Business Process Management Systems (BPMS) [2] is given by Fig. 1. On the top layer, called Strategy Thinking Layer, artefacts such as vision, mission, goal, strategy, objective and tactic are positioned. The layer below is the Business Process Architecture Layer, where the business process models that document the business processes reside. The third layer is the Business Process Execution Layer, where BPMS-executable versions of the business process models on the layer above are managed in order to run the business (by means of automated or human activities). The bottom layer, called Business Process Infrastructure Layer, contains the IT infrastructural services (e.g. web services, service-oriented software applications) that are used to automate the non-human parts of business processes.

The importance of implementing requirements by means of BPMS software is illustrated by Gartner Research [3], which estimates that by 2015 30% of business applications will be developed by means of BPMS technology. As traditional software packages are expected to play a less important role, we foresee a growing need for RE techniques that are adapted to BPMS packages.

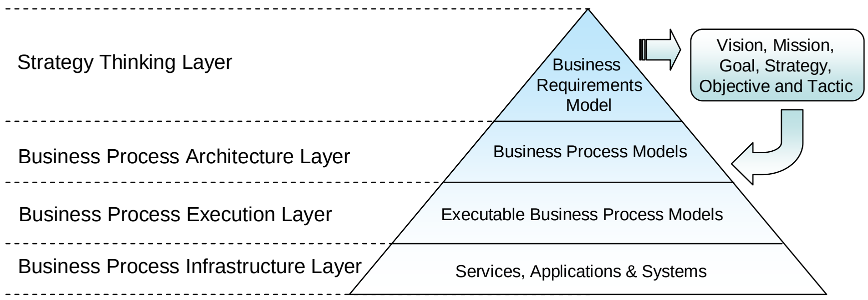


Fig. 1. Implementing business requirements through a Business Process Management System

Our research intends to contribute to the realization of the Business Process Architecture Layer. This paper presents an approach to model Strategy Thinking Layer goals as part of the business requirements for business processes and to automatically generate business process design skeletons (captured in models on the Business Process Architecture Layer) that respond to these business requirements. Our first contribution is taking an existing goal-oriented requirements engineering method, called the B-SCP (Business-Strategy Context Process) method [4], to create the Business Requirements Model of an organisation. The unique value proposition of B-SCP is combining the i* goal modelling language [5], Jackson's Problem Frames [6] and Role Activity Diagrams [7] into one overall top-down method. We extended the B-SCP method by developing a graphical editor for visually creating business requirement models and to generate B-SCP models based on the existing metamodels. Our second contribution is offering automatic transformation mappings to create business process design skeletons (in Business Process Modelling Notation - BPMN [8]) out of the B-SCP models. To this end, we reuse the work of Lapouchnian et al. [9] to support the generation of business process models.

The target user of our approach is a domain expert (called 'business user') who works in a business process-centred organisation and understands both high-level strategy concepts (such as business goals) and low-level operational details (such as the way business processes are organized). The development of this approach is based on the working hypothesis that it is useful and valuable to first model new or changed goals in a business requirement model and next to generate business process design skeletons out of this business requirement model in such a way that the changes to business process designs, needed to comply with the new/changed requirements, can be done more easily/effectively (compared to directly changing existing business process models).

2 Goal-Oriented Requirements Engineering for Business Processes

Our method provides the business user with an Eclipse-based Business Requirements Editor, which he/she employs to design a new Business Requirements Model (to initiate the Strategy Thinking Layer) or to adapt an existing Business Requirements Model (e.g. to add a new or modified business process to the Business Process Architecture Layer). In this paper, we will illustrate the first scenario that is detailed below (Step 1 to Step 6), and of which the first three steps are based on the B-SCP method of Bleistein et al. [4] and the fourth step relates to the work of Lapouchnian et al. [9]. The resulting Business Requirements Model is a hierarchical model of context diagrams, that have corresponding requirement diagrams. For instance, Fig. 2 shows the Business Requirements Model for the Seven-Eleven Japan (SEJ) case study [10], that describes the information-based strategies that have helped SEJ become a top performing retailer in Japan, selling high quality products through an industry-wide supply chain network of manufacturers, distributors, third-party logistics providers and franchise shops.

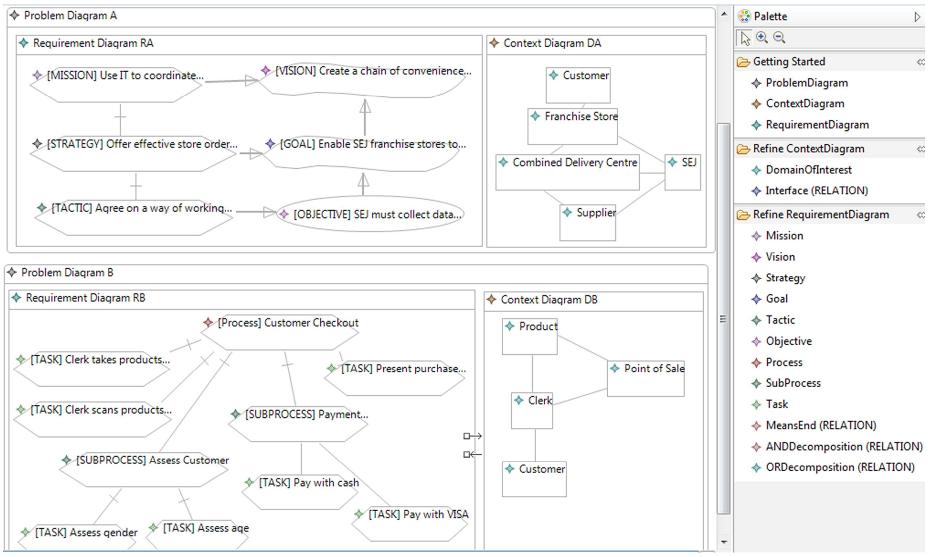


Fig. 2. Eclipse-based Visual Business Requirements Editor

Step 1. Identify the Business Model Participants and Their Relationships (B-SCP Original). The organisation of interest is determined, together with other business model participants (such as customers, allies and suppliers) and the flows of money, product and information between the participants are identified. For instance, SEJ is the organisation of interest, that relates to customers, franchise stores, combined delivery centres and suppliers (Fig. 2 – Context Diagram DA).

Step 2. Identify the Top-Level Business Requirements (B-SCP [4] Original). The VMOST (Vision, Mission, Objectives, Strategy, Tactics) method is used to deconstruct the motivational aspects of a company into business requirements, and the rules of OMG's Business Motivation Model (BMM) [1] are employed to relate the discovered business requirements. For instance, the vision 'create a chain of SEJ convenience stores where you can find a solution for any of your daily life problems at hours when needed' is supported by the mission 'Use IT to coordinate a supply chain of business partners' (Fig. 2 – Requirement Diagram RA).

Step 3. Identify Business Process Participants and Their Relationships (B-SCP [4] Original). For each business process that the business user wants to elaborate, determine the business process participants and the flow of products or information between the participants. The scope of the business process is defined by identifying the main business process participant, and by selecting other business process participants in function of the main one. For instance, the Point of Sale system of a franchise store relates to a product, the clerk and a customer (Fig. 2 – Context Diagram DB).

Step 4. Refine the Top-Level Business Requirements (Adapted from B-SCP [4]). Given a specific context of business process participants and their relationships, the top-level business requirements (such as vision, mission, goal, strategy, objective and tactic) should be refined into business requirements for business processes (such as the required business process itself, required subprocesses and required tasks). The original B-SCP framework considers all kinds of business requirements for business processes as *tactics*. In contrast, we consider a *business process* as something that realizes a *tactic* by means of an ordered collection of activities that takes one or more kinds of inputs and creates an output that is of value to the customer [11]. An activity in a business process could be a *subprocess*, which is a business process included in a higher level business process, or a *task*, which is an atomic activity performed by an end-user and/or an application. For instance, the business process 'Customer Checkout' has a subprocess 'Payment Process', that consists of a task 'Pay with cash' (Fig. 2 – Requirement Diagram RB).

Step 5. Add Control Flow Annotations (Based on Lapouchnian et al. [9]). In this paper, we want to support the basic control-flow patterns as published by the Workflow Patterns Initiative (WCP-1: Sequence, WCP-2: Parallel Split, WCP-3: Synchronisation, WCP-4: Exclusive Choice, WCP-5: Simple Merge) [12]. To this end, we let the business user annotate the links in the Business Requirements Model with control flow annotations. More specifically, the business user can choose means-end links, sequential AND decomposition links, parallel AND decomposition links, and OR decomposition links. Firstly, means-end links indicate a relationship between an end (i.e. i* soft/hard goal) and a means (i.e. i* task) for attaining this end (e.g. the mission is a means for attaining the vision). Secondly, a sequential AND decomposition link defines that the execution of an i* node depends on the left-to-right sequential execution of *all* nodes indicated by the link (e.g. the customer checkout process is achieved when the clerk starts by taking the products presented for purchase and ends by giving the receipt). Thirdly, a parallel AND decomposition link defines that the execution of an i* node depends on the execution of *all* nodes indicated by the link without imposing a particular order (e.g. to assess a customer, the clerk has both to assess customer age and gender, but the order in which this is done doesn't matter). Fourthly, an OR decomposition link defines that the execution of an i* node depends on the execution of *at least one* node indicated by the link (e.g. customers can pay the entire amount with cash or VISA, or partially cash and partially VISA).

Step 6. Transform Selected Problem Diagram into Business Process Model. The business user selects a problem diagram (e.g. Fig. 2 – Problem Diagram B consisting of RB and DB) from the Business Requirements Model, and activates the automated transformation mappings via the Eclipse environment. The automated mappings are implemented by means of the Atlas Transformation Language (ATL) project [13]. The basic requirements to run an ATL project are having a source metamodel, a target metamodel and an instance of the source metamodel. Based on the defined ATL mappings, an instance of the target metamodel will be generated from the source instance.

An extract of our ATL mappings that we defined is shown in Fig. 3. Firstly, sequential routing of business process activities, that are related to a domain of interest, map to workflow control-flow pattern WCP-1 (Fig. 3 – rule 4). Secondly, parallel routing of business process activities, that are related to a domain of interest, map to workflow control-flow pattern WCP-2 and WCP-3 (Fig. 3 –rule 5 and 6). Thirdly, conditional routing of business process activities, that are related to a domain of interest, map to workflow control-flow pattern WCP-4 and WCP-5 (Fig. 3 –rule 7 and 8). The output of the transformation mappings is a serialized business process model following the BPMN notation, which can be manually refined in an Eclipse-based BPMN editor (Fig. 4).

```
(1) rule ProblemDiagram{
     from a : BRM!ProblemDiagram
     to b : BPMN!BpmnDiagram(name <- a.name)}

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

(3) rule Task{
     from a : BRM!Task
     to b : BPMN!Activity(activityType <- 'Task', name <- a.name)}

(4) rule ANDDecomposition_Sequence{ --Implementation of WCP-1
     from a : BRM!ANDDecomposition(self.type = #SequentialOrder)
     to b : BPMN!SequenceEdge(iD <- 'Sequence Edge')}

(5) rule ANDDecomposition_Parallel_FirstOccurrence{ --Implementation of WCP-2 and WCP-3
     from a : BRM!ANDDecomposition(self.type = #ParallelOrder and
                                     BRM!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')}

(6) rule ANDDecomposition_Parallel_OtherOccurrences{ --Implementation of WCP-2 and WCP-3
     from a : BRM!ANDDecomposition(self.type = #ParallelOrder and not
                                   BRM!ANDDecomposition.allInstances()->first())
```

```
     to b : BPMN!SequenceEdge(iD <- 'Left Parallel Edge'),
        c : BPMN!SequenceEdge(iD <- 'Right Parallel Edge')}

(7) rule ORDecomposition_FirstOccurrence{ --Implementation of WCP-4 and WCP-5
     from a : BRM!ORDecomposition(BRM!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')}

(8) rule ORDecomposition_OtherOccurrences{ --Implementation of WCP-4 and WCP-5
     from a : BRM!ORDecomposition(not BRM!ORDecomposition.allInstances()->first())
     to b : BPMN!SequenceEdge(iD <- 'Left Conditional Edge'),
        c : BPMN!SequenceEdge(iD <- 'Right Conditional Edge')}
```

Fig. 3. ATL extract

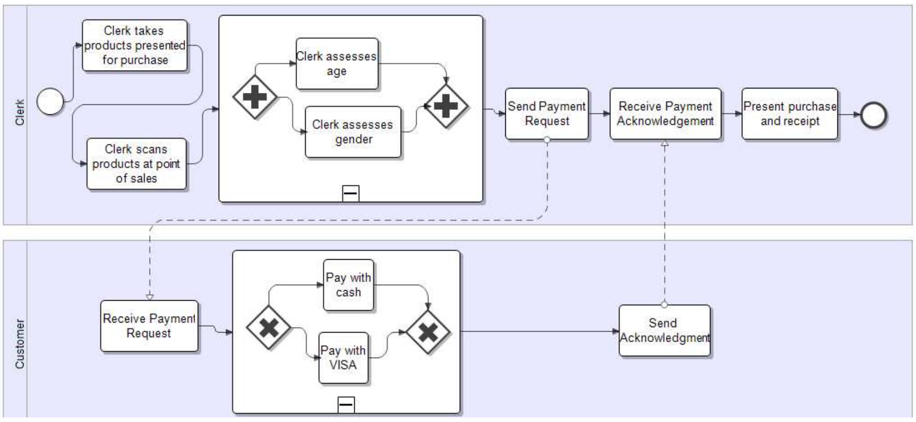


Fig. 4. Generated Business Process Model

3 Discussion

The overall goal of our research is to reduce the existing gap between RE research and industrial RE practice [14]. When considering the current research on goal-oriented requirements engineering, a lot of research is done related to the i* goal language. Nevertheless, 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 [15]. 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.

The differentiation between a business 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 [16], we believe it is easier to represent goals and business processes using different languages, and to provide automatic 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 Business 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 the Business Requirements Model as an overview (or one could say, an overarching strategically aligned Business Process Architecture), and automatically generate as much business process models from the Business Requirements Model as they require.

Finally, we want to discuss the difference between 'business requirements for business processes' and 'software requirements'. In terms of Fig. 1, business requirements are to be situated in the Strategy Thinking Layer, and relate to the motivational aspects of a company. Based on the Business Requirements Model of the Strategy Thinking Layer, this paper proposes a method to build the Business Process Architecture Layer by generate business process models that describe all kinds of activities (performed by machines or humans). Typically, these business process models are used by requirement engineers to discover software requirements [17] such as functional requirements, non-functional requirements, constraints, interfaces, etc. In contrast, in the context of a BPMS, these business process models do not act as documentation but could be executed -after adding the necessary run-time components- in the BPMS. So in this paper, the notion of 'software requirements' coincides with the business requirements for non-human, automated business processes.

4 Conclusion and Future Work

This paper presents an approach to model Strategy Thinking Layer artefacts as part of the business requirements for business processes and to automatically generate business process design skeletons (captured in models on the Business Process Architecture Layer) that respond to these business requirements. Our first contribution is extending an existing goal-oriented requirements engineering method, called the Business-Strategy Context Process (B-SCP) method, such that it can be used to create the Business Requirement Model for an organisation. Our second contribution is offering automatic transformation mappings to create business process design skeletons out of business requirements models. The expected implications of our research is to empower a (non-technical) business user with a serialized, yet intuitive way to represent his business requirements, going from strategies and goals till business processes and activities, and to allow these business users to generate business process models from these requirements. This should allow the business user to provide technical experts with reusable IT assets instead of providing merely paper-based requirements that requires more interpretation from (non-business) technical experts.

The current limitations of our work are the limited support for workflow control-flow patterns (only WCP-1 until WCP-5), and the lack of full-scale validation. In order to get initial feedback on the use and perceived value of our method, we decided to conduct a number of small-scale pilot studies in a specific context (Policy Modelling), in order to evaluate and refine our solution before considering larger-scale and more general case study research. The main finding [18] of our pilot studies was the need for a thorough preparation of the participants in understanding our definitions, tools and method steps. For instance, a correct understanding among the participants should be reached about what is a mission, strategy, tactic, business process, or task, as participants could have different interpretations. The full-scale validation should check whether the newly added activities in Step 4 could work well and contribute to the production of artefacts of higher quality in the latter steps. Next, we need to discuss the quality of the resulting BPMN artefact, and the applicability of transformation such that our transformation technique could be used for other goal-oriented RE techniques.

References

1. OMG: Business Motivation Model - http://www.omg.org/spec/BMM/. (2009)
2. Shaw, D.R., Holland, C.P., Kawalek, P., Snowdon, B., Warboys, B.: Elements of a business process management system: theory and practice. BPMJ **13** (2007) 91-107
3. Woods, J., Genovese, Y.: Delivery of Commodity Business Applications in a BPMS Does Not Mean You Should Customize the Applications. Gartner Research **G00139084** (2006)
4. 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** (2006) 846-868
5. Yu, E.S.-K.: Modelling strategic relationships for process reengineering (PhD Thesis). University of Toronto (1995)
6. Jackson, M.: Problem Frames: Analysing and Structuring Software Development Problems. Addison-Wesley, New York (2000)
7. Ould, M.A.: Business Processes: Modelling and Analysis for Reengineering and Improvement. Wiley, Chichester, New York (1995)
8. OMG: BPMN 1.2 specification - http://www.omg.org/spec/BPMN/1.2/. (2009)
9. Lapouchnian, A., Yu, Y., Mylopoulos, J.: Requirements-Driven Design and Configuration Management of Business Processes. Business Process Management (2007) 246-261
10. Bensaou, B.M., Uchino, H., Mitani, K., Noishiki, M.: Seven-Eleven Japan: Managing a Networked Organisation. INSEAD Euro-Asia Centre - Case Study 05/97-4690 (1997)
11. Hammer, M., Champy, J.: Reengineering the corporation : a manifesto for business revolution. HarperBusiness, New York (1994)
12. Russell, N., Hofstede, A.H.M.t., Aalst, W.M.P.v.d., Mulyar, N.: Workflow Control-Flow Patterns : A Revised View. BPM Center Report BPM-06-22, BPMcenter.org (2006)
13. Eclipse: ATL - ATLAS Transformation Language - http://www.eclipse.org/m2m/atl/.
14. 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** (2002) 113-123
15. 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. Advanced Information Systems Engineering (2004) 3-21
16. Mylopoulos, J.: Information modeling in the time of the revolution. Information Systems **23** (1998) 127-155
17. IEEE: Recommended Practice for Software Requirements Specifications - IEEE Standard 830-1998. (1998)
18. Decreus, K., Poels, G., Kharbili, M.E., Pulvermueller, E.: Policy-Enabled Goal-Oriented Requirements Engineering for Semantic Business Process Management (To Be Published). International Journal of Intelligent Systems (2009)

Toward a Privacy Enhancing Framework in E-government

Wiem Hammami[1], Lamjed Ben Said[1], Sameh Hadouaj[1], François Charoy[2], Khaled Ghedira[1],

[1] Intelligent Information Engineering Laboratory
Higher Institute of Management of Tunis
41, Rue de la Liberté, Cité Bouchoucha 2000 Le Bardo, Tunis, Tunisia
[2] Loria, Lorrain Research Laboratory in Computer Science and its Applications
Campus Scientifique, 54506 Vandoeuvre-lès-Nancy, Cedex, France,
{lamjed.bensaid, khaled.ghedira}@isg.rnu.tn, hadouaj@yahoo.fr,
wiem.hammami@ymail.com, Francois.Charoy@loria.fr

Abstract. E-government involves data sharing between different partners such as citizens and government agencies. Thus, the use of personal data in such cooperative environment must be done in legal ways and for legal purposes. In this context, issues related to data protection, such as privacy, have to be considered. This paper adopts a multi-agent based approach to manage privacy concerns in e-government systems. The proposed model provides a mechanism for e-government systems to evaluate trust degree reached by digital government processes. For this purpose, concepts of responsibility proposed in multi-agent systems and access rights used in security models, are integrated in this work. The research provides an evaluative framework for trust degree related to e-government process.

Keywords: E-government, privacy, trust, multi-agent systems, simulation.

1 Introduction

Privacy refers to the ability of individuals to control the collection, retention, and distribution of information about themselves [1]. In the context of e-government, privacy is a critical issue as there is an increased amount of private data shared between different agencies. For example, to access public service online, citizens must fill in some forms that require Personally Identifiable Information (PII) such as name, social security number, credit card number, etc. Citizens need to know whether their PII are used in the right way and for the right purpose or not. This can be achieved by an enhanced ability of control over their personal information. In this paper, we focus on data privacy, in particular, on privacy protection of personal data exchanged, processed and stored in e-government systems. As citizens act at the front office side of the e-government system, they do not know what happens to their personal information handled in the back office side by government agencies. Agent technology can be a suitable solution for this situation as they can act on behalf of the user. An agent is a computer system situated in some environment, and that is capable of *autonomous* actions in this environment in order to meet its design objectives [2]. The main issue of this paper is to propose a mechanism to control the use of personal information by e-government agents based on restrictions imposed on their behavior and to evaluate the trust degree related to e-government process.

The rest of this paper is structured as follows. In the second section we present different visions for privacy protection in the literature. In section 3 we describe our proposed model including the fundamental concepts used and their formal representations. Section 4 is devoted to experimentations and simulation results. In section 5, we make a comparative study of our work with other proposed model in the literature. Finally, section 6 summarizes the contribution of this work, and provides conclusions and future work.

2 Related works

Many approaches have been used to manage privacy concerns. We specially note those based on users preferences such as the P3P (Platform for Privacy Preferences) [3]. P3P provides technical mechanism to inform web sites users about privacy policy before they release their personal information. However, P3P does not provide mechanism for ensuring that sites act according to their policies [4]. Additionally, we mention approaches based on security modeling such as the Access Control Decision System. We note for example, the Role-Based Access Control model (RBAC) [5] that manages privacy through access control systems. In RBAC, users are assigned to roles to have permissions that allow them to perform particular job functions. However, we regret the lack of mechanism ensuring privacy protection of data after their collection in both P3P and RBAC approaches.

There are also a number of agent based privacy schemas in the literature. We note Hippocratic Multi-Agent Systems (HiMAS) [6]. HiMAS model define the concept of private sphere of an agent or a user that enable to structure and to represent data involved in privacy management. However, HiMAS model do not define metrics for trust evaluation. We note that existing approaches are often concerned about privacy protection in the data collection phase. But, actually we need to control data after their collection by e-government systems. Our main contribution is that we propose a new privacy schema based on multi-agent systems to handle such drawbacks of the existing knowledge.

3 The Privacy Enhancing Model

In this section, we introduce our multi-agent based model that we call ABC (Agent-Based Control). First, we describe the concepts used. Then, we present the ABC mechanism, and finally we describe our proposed techniques.

The ABC model enables to manage privacy concerns in e-government systems. This model offers a mechanism based on the use of a set of privacy rules and a set of information on the parties' rights, roles, responsibilities and restrictions to make a statistical assessment of *trust*. Consequently, ABC model enables the subsequent authorizations to transfer private data.

3.1. Model description

In the ABC model (see Fig. 1), we define two kinds of agents: *Admin agents* and *AP agents* (Authorization Provider agents). Admin agents represent the staff working in government agencies. AP agents are charged to provide *authorizations* to Admin agents in order to communicate with each other or to access objects in the system. Each Admin agent in our model plays a set of *roles* (e.g. the tax controller, the mayor, etc). A role includes a set of *responsibilities* (e.g. mayor roles: sign documents, validate, etc) that are restricted by *access-rights* (e.g. read only, write, read-write, etc). These access-rights are used to protect the *resources* and they are managed by a set of *privacy rules*. Access rights are also justified by a specific *context*. In fact, what is appropriate in one context can be a violation of privacy in another context.

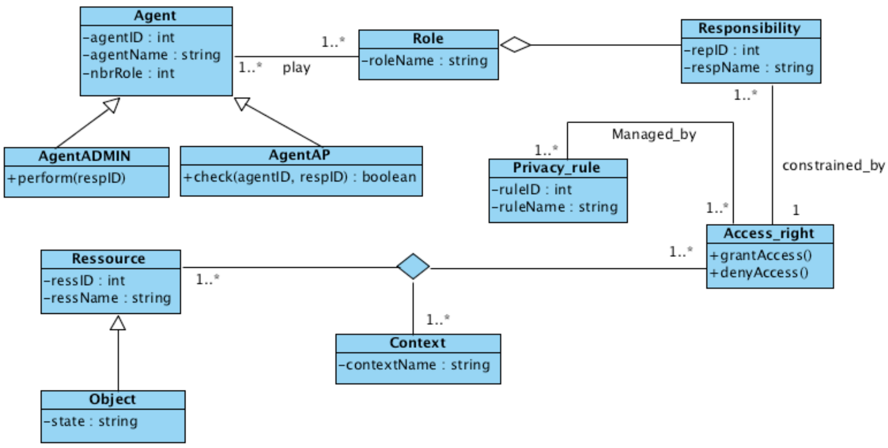


Fig. 1. ABC model

3.1.1. Definition of agent responsibility

We define responsibilities as the restricted behavior of a given agent. In other words, responsibility is the behavior that the system expects from an agent based on *restriction rules* (RR). We note that restriction rules are used to manage agent behavior at an internal level (e.g. making temporary results in a standard format before continuing execution, requesting specific authorizations before performing some actions, etc.). To more explain the concept of responsibility, let's take this example: we suppose that A_1 is an agent responsible for sending e- mails to agent A_2 (we suppose that e-mails are confidential). When we observe A_1's behavior, we find that A_1 sends e-mails to agent A_2 and sends a copy to the agent A_3 at the same time.

Thus, A_1 in this case did not assume his responsibility as some restriction rules are not respected.

3.1.2. Definition of privacy rules

Citizens must have control over their personal information handled by government systems. The Fair Information Practices (FIPs) [7] is an example of control enhanced by legislation, such as: limiting collection and disclosure, identifying purposes, etc.

In the ABC model, we define the following privacy rules that are in compliance with the FIPs and considered as the core needed to test and apply ABC model:

- R1: each agent must assume his responsibilities.

- R2: each agent has access only to objects needed for doing the set of his responsibilities.

- R3: agent cannot use the context to access linked data outside the set of his responsibilities.

For the privacy rules specifications, we are based on the notation of the rule-based systems [8] used in artificial intelligence, such that:

R1: Agent (A) $\wedge$ *Responsibility (Re)* $\wedge$ *Responsible-for (A,Re)*$\rightarrow$*Authorization(A, Re)*

R2: Responsible-for (A,Re) $\wedge$ *Access (R,O,Re)*$\rightarrow$ *Authorization(A, Re, R,O)*

R3: Responsible-for(A,Re)$\wedge$*Access(R,O,Re)*$\wedge$*(O*$\rightarrow$*J)* $\rightarrow$ *NOT(Authorization(A,Re, R,J)*

3.2. Description of ABC mechanism

In the ABC model, we define a distributed architecture in which sets of the agent in different groups are interacting with each other. Each group (container) represents the set of Admin agents running in the same e-government agency. To have access to objects in the system or to communicate with other agents, each Admin agent must be authorized from AP agent that exists in his group. To keep control of the system, AP agents use a Rule Base (RB) and a Knowledge Base (KB). RB includes the set of privacy rules and KB includes the set of knowledge in the system: agents, their roles, their responsibilities, their RR, their access-rights and their resources. In ABC model, we suppose that in case of failure, AP agents can switch roles dynamically with Admin agents. AP agents delegate the control of the system to the most trusted Admin agent. This delegation decision is based on the computation of Admin agent's honesty degree that will be explained in the next section.

3.3. Description of ABC techniques

In this section, we define techniques used for privacy protection in ABC model: the computation of the trust and the honesty degrees. Formally, our ABC model correspond to the following set: {A, Re, Au, PR, T, R} such that:

A: the set of agents in the system
Re: the set of agent responsibilities
Au: the set of authorizations
PR: the set of privacy rules

T: the trust degree reached in an e-government process. T is defined as*:*

$$T = \sum_{\substack{i=1 \\ i \neq j}}^{k} h(i,j) * \alpha_j / k. \tag{1}$$

where *k* represents the total number of agents in the system, *h* represents the honesty of agent j estimated by agent i (see Fig. 2). *h* is defined as follows:

$$h = \sum_{i=1}^{i=k} h(i,j). \tag{2}$$

After each transaction, an Admin agent i can give feedback to Admin agent j according to the service quality of j. Thus, a feedback score S is calculated as follows: S= P-N, where P is the number of positive feedbacks left by agents and N is the number of negative feedbacks from agents. The S value is disclosed to all agents in the system. This reputation model has been presented in [9]. *h* is decreasing when the agent is performing unauthorized actions. We define two types of such actions: unauthorized access to objects, and unauthorized communications with other agents. We suppose that honesty value is between 0 and 1 and it is disclosed to all agents in the system. α_j represents the interaction degree related to agent j. it denotes a weight used to balance T value because we must consider that agents behave differently. In our model, honest agents (having h=1) are rewarded. However, dishonest agents (having h<0,5) are punished.

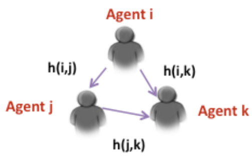


Fig. 2. Representation of Admin agent's honesty

We represent risk degree associated to the use of personal information (R), by the following:

$$R = 1 - T \cdot \tag{3}$$

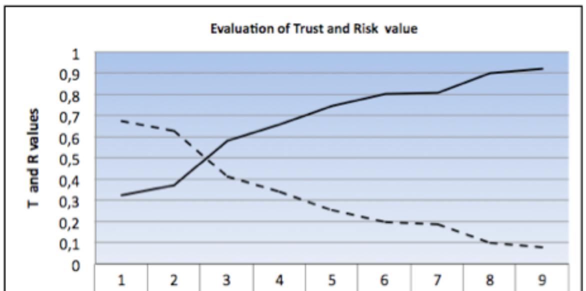
4 Simulation and experimental results

Using agent technology, we can make *« behavioral simulation »* that enable us to create a virtual image of the reality. This is considered as a powerful predictive analysis tool that enable decision makers to test their idea via scenario in an artificial environment before implementing their decision in the real world. In this section, we present our Multi-Agent Based Simulation (MABS) of agents' behavior during the company formation process in Tunisia. We chose this scenario because it is complex (involves numerous administrations and many interactions) and requires the collection of many sensitive data at every step. We represent governmental agencies involved in this process by a set of agents (twelve agents) interacting with each other on behalf of the user (the citizen) and we simulate their behavior during the whole process using the ABC model. Our MABS is implemented using JADE platform [10]. We used JADE because it is a distributed platform and supports agent mobility. The mobility is an important issue of our work for a real application of the proposed model. For the privacy rules' specifications, described in section3, we used the rule engine Jess [11] to make authorization' decision based on both agent's responsibilities and access rights. To realize our simulation, we made the following hypothesis:

- All agents initially are honest (having h = 1)
- A simulation step corresponds to n successive interactions. In the following simulation results, we assume that n=10.

At every simulation step, we evaluate Admin agent's honesty and the trust value.

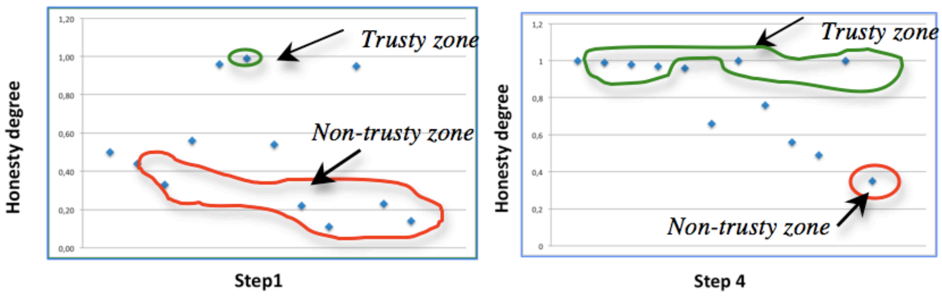
The first experimental result is the evaluation of trust degree reached during the running of company formation process. The trust (versus the risk) degree obtained during nine steps of our simulation is plotted in Fig. 3.



	1	2	3	4	5	6	7	8	9
T	0,3251	0,3718	0,5847	0,6607	0,7443	0,8038	0,8106	0,8989	0,9211
R	0,6748	0,6281	0,4152	0,3392	0,2556	0,1961	0,1893	0,1010	0,0788

Fig. 3. Evaluation of Trust (T) and Risk (R)

As observed, we note that T is increasing notably during the simulation and R is decreasing (as there is a dual relation between T and R).



(a) (b)

Fig.4. (a) Comparison of agent's honesty in step1, (b) Comparison of agent's honesty in step 4

According to our model, honest agents (having h=1) are placed in a trusty zone and dishonest agents (having h<0,5) are placed in non-trusty zone. As shown in Fig. 4 (a), we note that in the first step of our MABS only one agent is honest. However, when we observe agent's behaviors during next steps, we note that the number honest agents increase (Fig. 4 (b) shows an example of this increase during the step 4 of our MABS).

According to this interpretation, we find that agents placed in the non-trusty zone want to behave like honest agents, they want to move to the trusty zone. This enabled us to interpret the increase of the trust value during the simulation.

5 A comparative study

Regarding to some standard evaluation criteria, such as the use of access control mechanisms, the use of user preferences, the trust evaluation metrics [12] and the use of anonymity techniques [13] our work is the most appropriate one that is able to support all of these criteria. The following table summarizes the comparative study of our model with P3P and RBAC models. For each criterion we attribute (+) to the model that supports it and (-) to the model that does not support it.

Table 1. A comparative study

Evaluation criteria	Related works (1) RBAC	Related works (2) P3P	Our work ABC
Access control	+	-	+
User preferences	-	+	+
Trust evaluation metrics	-	-	+
Anonymity	-	-	+

In fact, the use of agent technology in our work, has many advantages:
Agent-based models provide a more convincing approach to modeling the real world behaviors due to their ability to explicitly model a component of the real world such as: human, organizations, etc. The dynamicity of the real world including environmental, political and social behaviors can be captured within a software agent. Also, in multi-agent systems, we have the possibility to encapsulate private data. So, we do not need additional security mechanisms to ensure data protection. One of the main characteristics of Multi-Agents Systems (MAS) is the distribution. Using MAS we can have a decentralized framework in which tasks are dispatched to agents in the system. So, we can distribute the control of data transfer and access. Therefore, we take profit from the task-delegation via agents, which is impossible with other approaches.

6 Conclusion and future work

In this paper, we proposed a new model for privacy enhancing in e-government context. This model enabled us to build an evaluative framework for trust degree reached for a given e-government process. In the context of e-government it is crucial to build a trust relationship to ensure and enforce the adoption of e-government systems by citizens. Also the proposed approach has the potential benefit of the use of only one trusted entity: the AP agent. For future works we propose to enrich our model by adding further privacy rules. We also plan to incorporate different types of risks related to privacy protection such as risks related to: data collection, data processing, data sharing, etc. Finally, we suggest managing task delegation between Admin agents to ameliorate performances of e-government systems.

Acknowledgments. This work has been supported by a common grant from Tunisian and French government relative to the project STIC DGRSRT/INRIA entitled: multi-agent coordination models for e-government systems.

References

1. Goldberg, I., Wagner, D., Brewer, E.: Privacy-Enhancing Technologies for the Internet. In: COMPCON'97, Proceedings, IEEE, pp. 103--109. IEEE Press (1997)
2. Wooldridge, M., Jennings, N.R. : Intelligent agents: Theory and practice. The Knowledge Engineering Review 10(2), 115--152 (1995)
3. Cranor, L., and al.: The Platform for Privacy Preferences1.1 (P3P1.1) Specification W3C (2004)
4. Agrawal, B., Bhattacharya, J., Gupta, S.K.: Protecting Privacy of Health Information through Privacy Broker. In: 39th International Conference on System Sciences, Proceedings (2006)
5. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-based access control models. IEEE Computer 29 (2), pp. 38--47 (1996)
6. Crépin, L., Vercouter, L., Jaquenet, F., Demazeau, Y., Boissier, O.: Hippocratic multi-agent systems. In: 10th International Conference of Entreprise Information Systems, pp. 301--308 (2008)
7. Organisation for Economic Co-operation and Development: OECD Guidelines on the Protection of Privacy and Transborder Flows of Personal Data (1980)
8. Sydenham, P.H. and Thorn, R.: Handbook of Measuring System Design. John Wiley & Sons, ISBN: 0-470-02143-8 (2005)
9. K.J. Lin, H. Lu, T. Yu, C. en Tai.: A reputation and trust management broker framework for web applications. In: IEEE International Conference on e-Technology, e-Commerce and e-Service, pp. 262--269 (2005)
10. Bellifemine, F., Bergenti, F., Caire, G., Poggi, A.: JADE- A Java Agent Development Framework. In: Multi-Agent Programming, pp. 125--147 (2005)
11. Friedman-Hill, E. : Jess The Rule Engine for the Java Platform Version 7.1p2 (2008)
12. Yang, Y., Lin, K.J., Wong, D.S. and Varadharajan, V.: Trust Management Towards Service-Oriented Applications. In: The IEEE International Conference on e-Business engineering (ICEBE 2007), pp. 129-- 146 (2007)
13. Flegel, U.: APES: Anonymity and Privacy in Electronic Services. Advances in Information Security, vol. 35, Springer US, pp. 171--176 (2007)