LATVIJAS UNIVERSITĀTES
RAKSTI

757. SĒJUMS

Datorzinātne un informācijas tehnoloģijas

SCIENTIFIC PAPERS
UNIVERSITY OF LATVIA

VOLUME 757

Computer Science and Information Technologies

SCIENTIFIC PAPERS
UNIVERSITY OF LATVIA
VOLUME 757

Computer Science and Information Technologies

Databases and Information Systems

Doctoral Consortium
Riga, July 4, 2010
Ed. Guntis Barzdins and Leo Selavo

Ninth International Baltic Conference
Baltic DB&IS 2010
Riga, Latvia, July 5–7, 2010

University of Latvia

LATVIJAS UNIVERSITĀTES
RAKSTI
757. SĒJUMS

Datorzinātne un informācijas tehnoloģijas

Datu bāzes un Informācijas sistēmas

Doktorantu konsorcijs
Rīga, 2010. gada 4.jūlijs
Red. Guntis Bārzdiņš un Leo Seļāvo

Devītā Starptautiskā Baltijas konference
Baltic DB&IS 2010
Rīga, Latvija, 2010. gada 5-7. jūlijs

Latvijas Universitāte

UDK 004(082)
Da 814

Layout: **Ilze Reņģe**

ISSN 1407-2157
ISBN 978-9984-45-189-3

Contents

Conference Committee

Conference Chair

Juris Borzovs, Faculty of Computing, University of Latvia

Organising Co-Chairs

Guntis Arnicans, Faculty of Computing, University of Latvia
Inara Opmane, Institute of Mathematics and Computer Science, University of Latvia

Conference Secretary

Elina Kalnina, Institute of Mathematics and Computer Science, University of Latvia

Co-ordinators

Saulius Maskeliunas, Institute of Mathematics and Informatics, Lithuania
Hele-Mai Haav, Institute of Cybernetics at Tallinn University of Technology, Estonia
Ahto Kalja, Department of Computer Engineering of Tallinn University of Technology, Estonia

Advisory Committee

Janis Bubenko, Royal Institute of Technology, Sweden
Arne Solvberg, Norwegian University of Science and Technology, Norway

Programme Co-Chairs

Janis Barzdins, University of Latvia
Marite Kirikova, Riga Technical University

Doctoral Consortium Co-Chairs

Guntis Barzdins, University of Latvia
Leo Selavo, University of Latvia

Programme Committee

Witold Abramowicz, Poland
Irina Astrova, Estonia
Guntis Barzdins, Latvia
Janis Bicevskis, Latvia
Juris Borzovs, Latvia
Albertas Caplinskas, Lithuania
Vytautas Cyras, Lithuania
Dale Dzemydiene, Lithuania
Johann Eder, Austria
Hans-Dieter Ehrich, Germany
Janis Grabis, Latvia
Hele-Mai Haav, Estonia
Janis Grundspenkis, Latvia
Mirjana Ivanovic, Serbia
Leonid Kalinichenko, Russia
Ahto Kalja, Estonia
Audris Kalnins, Latvia
Peep Küngas, Estonia
Rein Kuusik, Estonia
Marion Lepmets, Estonia
Audrone Lupeikene, Lithuania
Kalle Lyytinen, USA
Hui Ma, New Zealand
Rainer Manthey, Germany

Saulius Maskeliunas, Lithuania
Jorgen Fischer Nilsson, Denmark
Boris Novikov, Russia
Algirdas Pakstas, UK
Jaan Penjam, Estonia
Karlis Podnieks, Latvia
Gunter Saake, Germany
Kurt Sandkuhl, Sweden
Michal Smialek, Poland
Darja Smite, Sweden
Janis Stirna, Sweden
Uldis Sukovskis, Latvia
Kuldar Taveter, Estonia
Jaak Tepandi, Estonia
Benhard Thalheim, Germany
Enn Tyugu, Estonia
Olegas Vasilecas, Lithuania
Tatjana Welzer, Slovenia
Wita Woitkowski, USA
Robert Wrembel, Poland
Stanislaw Wrycza, Poland
Naoki Yonezaki, Japan
Jozef M. Zurada, USA

Additional Referees

Erika Asnina, Latvia
Edgars Celms, Latvia
Karlis Cerans, Latvia
Boris Cogan, UK
Janis Eiduks, Latvia
Kārlis Freivalds, Latvia
Rūsiņš Freivalds, Latvia
Ingolf Geist, Germany
Martin Henkel, Sweden
Kristi Kirikal, Estonia
Christian Koncilia, Austria
Arne Koschel, Germany
Deniss Kumlander, Estonia
Julius Köpke, Austria
Algirdas Laukaitis, Lithuania
Egons Lavendelis, Latvia
Azeem Lodhi, Germany
Leonids Novickis, Latvia
Martins Opmanis, Latvia
Syed Saif ur Rahman, Germany
Tarmo Robal, Estonia
Eike Schallehn, Germany
Inna Shvartsman, Estonia
Renate Strazdina, Latvia
Vladimir Tarasov, Sweden
Sarah Tauscher, Germany
Juris Viksna, Latvia

Local Organising Committee

Vineta Arnicane
Edgars Celms
Dainis Dosbergs
Girts Karnitis
Lelde Lace
Inga Medvedis
Edgars Rencis
Agris Sostaks
Viesturs Zarins

Preface

The Baltic conference on databases and information systems is a bi-annual international forum for technical discussion among researchers and developers of databases and information systems. The objective of the conference is to bring together researchers, practitioners, and PhD students in the fields of advanced database and information systems research. Besides evolution of the classic database and MDA techniques, the papers submitted this year highlight also interest into following research areas: distributed and web-based systems, attention to security and industry standards, striving towards use of ontologies and natural language processing techniques. The 9th International Baltic DB&IS 2010 Conference held this year is hosted by University of Latvia (Riga, July 5-7, 2010).

The doctoral consortium solicited submissions on research-in-progress that is meant to lead to a PhD dissertation. The goal of the consortium is to provide doctoral students with independent constructive criticism through written reviews of the program committee members as well as through providing the discussion forum with senior scientists and other doctoral students.

In contrast to contributions to the main conference track, submissions to the doctoral consortium are allowed to concentrate on the overall goals of the doctoral thesis rather than on single technical results. Meanwhile mere ideas and statements of intent were not deemed sufficient for acceptance.

Out of 13 papers from 5 countries submitted the program committee accepted 8 papers. Each paper was reviewed by at least three reviewers, which judged the submissions with respect to originality, quality, and consistency with the topics of the conference. The final doctoral consortium program is the result of the constructive work of the following Program Committee members:

Irina Astrova, Estonia
Guntis Barzdins, Latvia
Janis Barzdins, Latvia
Janis Bicevskis, Latvia
Juris Borzovs, Latvia
Albertas Caplinskas, Lithuania
Dale Dzemydiene, Lithuania
Mirjana Ivanovic, Serbia
Ahto Kalja, Estonia
Audris Kalnins, Latvia
Marite Kirikova, Latvia
Peep Küngas, Estonia
Audrone Lupeikene, Lithuania
Saulius Maskeliunas, Lithuania

Jaan Penjam, Estonia
Karlis Podnieks, Latvia
Gunter Saake, Germany
Leo Selavo, Latvia
Darja Smite, Sweden
Kuldar Taveter, Estonia
Jaak Tepandi, Estonia
Enn Tyugu, Estonia
Olegas Vasilecas, Lithuania
Tatjana Welzer, Slovenia
Wita Woitkowski, USA
Robert Wrembel, Poland
Jozef M. Zurada, USA

I thank all the contributors (successful and not so successful), the members of Program Committee, the Conference Chairs and Co-chairs, the members of Organizing Committee, and University of Latvia for their tireless support provided.

Guntis Barzdins

SCIENTIFIC PAPERS, UNIVERSITY OF LATVIA, 2010. Vol. 757
COMPUTER SCIENCE AND INFORMATION TECHNOLOGIES 11–22 P.

The Specific Text Analysis Tasks at the Beginning of MDA Life Cycle

Armands Šlihte
Faculty of Computer Science and Information Technology,
Institute of Applied Computer Systems, Riga Technical University
armands.slihte@rtu.lv

Abstract. This paper recognizes the computation independent nature of a Topological Functioning Model (TFM) and suggests it to be used as the Computation Independent Model (CIM) within Model Driven Architecture (MDA). To step towards the completeness of MDA and enable the automation of system analysis the Topological Functioning Model for Model Driven Architecture (TFM4MDA) method is considered. A project of implementing TFM4MDA as a TFM Tool is suggested to enable artificial intelligence in system analysis and software development. The main components of the tool are a TFM Fetcher for system's informal description analysis, TFM Editor and TFM Transformer for TFM to UML transformation. This paper discusses the specific text analysis tasks at the beginning of MDA life cycle and the implementation challenges of the TFM Fetcher component.

Keywords: Topological Functioning Model, Model Driven Architecture, Language Processing, Meta-Object Facility, Query/View/Transformation

1 Introduction

Software development is a complex process. Every software development project is unique. However in most cases the abstractions or models of the information systems to be developed may be at least similar if not the same. Software developers are often busy with coding similar structures and procedures; the development process becomes somewhat inefficient. Moreover software development is expensive and there are many risks that stakeholders have to take in account. The industry of software development has been approaching and dealing with these issues in different ways.

Model Driven Architecture (MDA) proposes software development to abstract from the code as the uppermost of the functionality of the information system to the model of the information system [1]. That means that first an information system's model is developed and then it is transformed into a ready-to-use information system or at least a ready-to-implement framework of the system. Changes and additions also are made

using the model. The purpose of MDA is to enable software development using the models of an application and generating the source code from these models.

MDA is a software development framework which defines 3 layers of abstraction for system analysis: Computation Independent Model (CIM), Platform Independent Model (PIM), and Platform Specific Model (PSM). MDA is based on 4 level architecture and the supporting standards: Meta-Object Facility (MOF), Unified Modeling Language (UML), and XML Metadata Interchange (XMI) [2]. Moreover, Query/View/Transformation (QVT) is a standard for model transformation, which is also a critical component of MDA [3].

TFM offers a formal way to define a system by describing both the system's functional and topological features [3]. TFM is represented in the form of a topological space (X, Θ), where X is finite set of functional features of the system under consideration, and Θ is the topology that satisfies axioms of topological structures and is represented in the form of a directed graph [5]. TFM represents the system in its business environment and shows how the system is functioning, without details about how the system is constructed. TFM4MDA method suggested in [5] and developed in Riga Technical University allows system's TFM to be composed by having knowledge about the complex system that operates in the real world. This paper suggests using TFM as CIM by composing it using TFM4MDA; acquiring a mathematically formal and thus transformable CIM.

This paper analyses the specific text analysis tasks at the beginning of MDA life cycle and provides solutions to these tasks. The first task is defining a formal data structure for the knowledge about the system. TFM4MDA assumes that this knowledge can be presented as an informal description of the system with text in natural language. But such an informal description is far too complex and redundant for a formal analysis. Another task is to create a formal method or an algorithm for constructing a TFM by analyzing this knowledge about the system. The basic building blocks of the data structure representing the knowledge about the system will be a sentence in natural language, so language processing methods will have to be applied.

The long-term goal of this work is to improve TFM4MDA method and to develop a TFM Tool which would fully implement this method. MDA tools mainly focus on requirements gathering, domain modeling, and code generation [6], not offering a way for defining a formal CIM. This tool starts a new direction of MDA tools by offering construction of a formal CIM and applying elements of artificial intelligence for system analysis and software development. The development of such a tool is a complex and a large scale project, which requires dealing with several issues. This paper talks about the issues related to the task of implementing a TFM Tool.

This paper is organized as follows. Section 2 analyzes related work, discussing the TFM4MDA method and other approaches dealing with the transformation of an informal description of a system to a formal model. Section 3 describes the specific text analysis tasks at the beginning of MDA life cycle. Section 4 provides a solution for representing the knowledge about a system in a formal way and shows an example. Section 5 addresses the task of retrieving functional features from use cases by applying language processing methods. Section 6 defines a method for retrieving topology from use cases and demonstrates it. Conclusions summarize the work done and explain the significance of further research.

2 Related Work

This work continues research on computation independent modeling and specifically on TFM4MDA started in [3], [5], [7] and [8]. As stated in [5] an informal description of the system in textual form can be produced as a result of system analysis. TFM4MDA proposes an approach for transforming this system's informal description into a TFM of the system. The concept of the TFM Tool is described in author's earlier work [9]. A MOF-compatible metamodel of the TFM and the development of a TFM Editor component is also described in [9].

TFM4MDA consists of the following steps: 1) retrieving the system's objects and functional features by analyzing the informal description of a system; 2) constructing a TFM's topological space using the retrieved system's objects and functional features; 3) constructing a TFM's topological graph using its topological space; 4) verifying the functional requirements by mapping them to the corresponding functional features; 5) transforming TFM to UML (a specific formal UML profile). The approach described in [5] still defines some structure of the informal description, thus making it semi-formal. This paper will introduce more formalism into TFM4MDA's conception of an informal system's description.

There have been other attempts to transform an informal description of a system to a formal model. Approach proposed in [10] suggests generating implementation from textual use cases. This approach uses statistical parser on use cases and by analyzing the parse trees compose so called Procases for further use in implementation generation. Procases can be thought of as a formal model of requirements.

Another approach ReDSeeDs [11] defines software cases to support reuse of software development artifacts and code in a model driven development context. This approach is very formal and it depends on writing the software cases very precisely by adding specific meaning to every word or phrase of software case sentences.

The Use Case Driven Development Assistant (UCDA) tool's methodology follows the IBM Rational Unified Process (RUP) approach to automate the class model generation [12]. It starts with analyzing the requests of stakeholders and identifies actors and use cases. From there the tool can generate the system's use case diagram, class diagram, collaboration diagram, and other artifacts. The tool uses natural language parsing to achieve this. This methodology deals only with identifying use cases, but not how they operate. The steps of the main scenario or the basic flow of events have to be defined manually.

Linguistic Assistant for Domain Analysis (LIDA) processes text to help the analyst identify the objects and model elements. By also providing a model editing environment the model elements are refined through a validation process [13]. This approach provides a very handy toolset for a system analyst, but the models still have to be manually constructed.

Approach suggested in this paper provides a way to automatically acquire a formal model from knowledge about the system. Defining this knowledge and then validating the model are done manually by the system analyst. TFM4MDA is devised to enable artificial intelligence methods in software development; after defining the knowledge about a system TFM4MDA would derive its meaning automatically by constructing a TFM.

This paper also suggests textual use cases to be used for defining requirements and as input for text analysis from which a TFM could be composed. Approach described in [10] uses their generated implementation for verifying software requirements and also to use the implementation as a platform to proceed with the development of the software.

3 Specific Text Analysis Tasks

For a demonstration of the TFM4MDA method an example library system described in [5] is considered. This example will be used throughout the paper. For using TFM4MDA as described in [5] first we need an informal description of a system. Let us consider this fragment: "The librarian checks out the requested book from the book fund to a reader, if the book copy is available in the book fund." This fragment is from [5]. Then the system analyst identifies system's objects and composes functional features. The following system's objects can be identified: a librarian, a book copy (a synonym is a book), a book fund, a reader. Every functional feature consists of an object action, a result of this action, an object involved in this action, a set of preconditions of this action, an entity responsible for this action, and subordination.

Using the given fragment of an informal description we can compose the following functional feature: 1) the action is checking out; 2) the result of this action is a book copy is checked out for a reader; 3) the object involved in this action is a book copy; 4) a precondition of this action is that a book copy has to be available; 5) the entity responsible for this action is a librarian; 6) subordination is inner. These attributes of a functional feature are proposed in [5], but for an algorithm to retrieve them it is necessary for all these attributes to be represented in the informal description. It is possible that some of these attributes are absent – a result of the action or object involved in the action. For this reason attributes object action, a result of this action, an object involved in this action, are merged into one attribute – action. This makes the task of retrieving functional features by text analysis a little but easier.

Next step of the method is to construct a topological space of TFM, meaning that the analyst has to identify the cause-effect relations between the composed functional features, define the main functional cycle and verify functional requirements.

TFM Tool will support this process by providing a TFM Fetcher component for retrieving functional features automatically and allowing the user to correct initial functional features and cause-effect relations. In addition the tool will enable the user to manually point to the main functional cycle, define functional requirements, and check their conformity to the functional features. TFM Tool has to support a number of iterations back and forth between description and TFM Fetcher until the analyst has verified every functional requirement and set the main functional cycle. The user of the tool will be able to see the mapping between the description and TFM, and then correct any incompleteness, redundancy or inconsistency.

Where does an informal description of the system come from? The main idea is that this description contains the knowledge about the problem domain, but the representation of it might vary. There are a lot of different methodologies to support software development. All of them require some sort of requirements gathering process, which usually provides software requirements expressed in textual and diagram form.

Some of these methodologies are more formal others less formal, but in most cases textual and diagram requirements of the system can be considered as the knowledge about the problem domain. Constructing a formal model from text analysis is not a simple task. In a realistic case the description can probably be quite long, incomplete, redundant and inconsistent. To make this task a little easier the description of the system has to have some degree of formality. One of the most popular software development approaches today is use case driven software development. Use case driven software development provides a way to define knowledge about the problem domain in a more structured form than plain text. For this reason business use cases are considered as the system's informal description.

4 Use cases

Use cases are not normalized or standardized by any consortium, unlike UML use case diagram by Object Management Group. Moreover, there are many different use case templates and the structure of a use case can be adjusted depending on the situation and the development team [14]. Usually use case structure can consist of the following or similar sections: use case identifier, description, actors, assumptions, steps, variations and non-functional requirements.

Use case: **Requesting a book**
Actors: Client, Librarian
Preconditions: Librarian authorizes reader status
Main scenario:
1. Client searches for a book in the catalogue
2. Librarian hands out a request form
3. Client fills the request form
4. Librarian checks out the book from book fund
5. Librarian gives the book to client
6. Client leaves the library

Extensions:
1a. If client hasn't found a book, client leaves the library

Sub-variations:
4a. If the book is not availiable in the book fund, librarian denies the book request form
4a1. Client searches for a book in the catalogue
6a. If client wants another book, client searches for a book in the catalogue

Use case: **Registering**
Actors: Client, Librarian
Preconditions:
Librarian hands out a registration form
Main scenario:
1. Librarian hands out a registration form
2. Client fills the registration form
3. Librarian creates a new reader account
4. Librarian creates a new reader card
5. Librarian hands out the reader card
6. Librarian authorizes reader status

Extensions:
-

Sub-variations:
-

Use case: **Returning a book**
Actors: Client, Librarian
Preconditions: Librarian authorizes reader status
Main scenario:
1. Client gives the book to librarian
2. Librarian checks condition of the book
3. Librarian checks in the book into book fund
4. Client leaves the library

Extensions:
2a. If the book is damaged, librarian calculates a fine
2a1. Librarian gives the fine ticket to client
2a2. Client pays the fine
2a3. Librarian checks in the book into book fund

Sub-variations:
4a. If client wants another book, client searches for a book in the catalogue

Use case: **Arriving**
Actors: Client, Librarian
Preconditions: -
Main scenario:
1. Client arrives at the library
2. Client shows a reader's card
3. Librarian authorizes reader status

Extensions:
-

Sub-variations:
2a. If client doesn't have a reader card, librarian hands out a registration form

Fig. 1. Use cases for a library. This shows an example of business use cases for a library: arriving, registering, requesting a book and returning a book.

In context of TFM Tool textual business use cases are considered the representation of knowledge about the system. The following structure of use case is considered: 1) use case title, 2) actors, 3) pre-conditions, 4) main scenario, 5) extensions, and 6) sub-variations. Use case title shortly describes the use case; actors are a list of actors involved in the use case; pre-conditions define the conditions that must be in place before this use case starts; main scenario lists the specific steps (written in natural language) that take place to complete the use case; extensions and sub-variations list deviations from the main success scenario - branch actions, with the difference that extensions are performed in addition to extended action, but sub-variations are performed instead of the extended action. This use case structure is very similar to that proposed in [10].

As you can see in Fig. 1, extensions and sub-variations are numbered as follows 1a., 1a1, 1a2, etc. First number represents the step that is being extended or sub-variated, and the first step of extension or sub-variation always has a condition (if) that has to be true for the step to be executed.

Now a fromal data structure that can be used to represent the knowledge about a system is defined. If there is a set of use cases that describe a working system, it is possible to process them with purpose of retrieving functional features.

5 Retrieving Functional Features

Functional features are represented by a tuple consisting of action, a set of preconditions of this action, an entity responsible for this action, and subordination [5]. As mentioned earlier an object action, a result of this action and an object involved in this action are merged into action because of the complexity of text analysis. One of the tasks of the TFM Fetcher component is to retrieve these functional features from use cases.

Use cases are formed by sentences written in natural language. Every sentence, except title and actors, in a use case can be considered as a representation of a functional feature. Use case sentence can sometimes represent more than one functional feature. This can happen when sentence consists of more than one result of the action or objects involved in the action. Such an issue can be dealt with by analyzing sentence's coordinating conjunctions. For example in Fig. 1, if 2nd and 3rd steps of use case "Requesting a book" are combined in one sentence "Librarian hands out a request form and client fills the request form". In this sentence the second reference to a request form could be replaced by a pronoun "it". This should be taken into account. Moreover, the sentences of a use case should be written as simple and unambiguous as possible, but in realistic case this is not always possible. In the examples used in this paper use case step sentences are constructed to answer this question – who does what? For example, "Librarian checks out the book from book fund". The verb phrase of the use case step's sentence is considered the action. Moreover, use case's actors will be considered as objects involved in the action and entities responsible for the action. The title can partly be considered as a functional requirement.

TFM Fetcher component has to be able to form the corresponding functional features by analyzing the use case sentences. For this purpose natural language processing methods have to be applied.

Concrete syntax tree, or parse tree for short, will be used for the analysis of use case sentences. Parse tree is a tree that represents the syntactic structure of a sentence according to some formal grammar [15]. Parse trees are usually output of parsers, which can use different methods for finding the right parse tree for the specific sentence. The most efficient parsers are statistical parsers which associate grammar rules with probability. For example, use case sentences "Librarian checks out the book from book fund" and "Librarian creates a new reader account" will be parsed using The Stanford Parser [16]; results are shown in Fig. 2 By exploiting statistical parser it is possible to acquire the structure of the sentence, and thus analyze it.

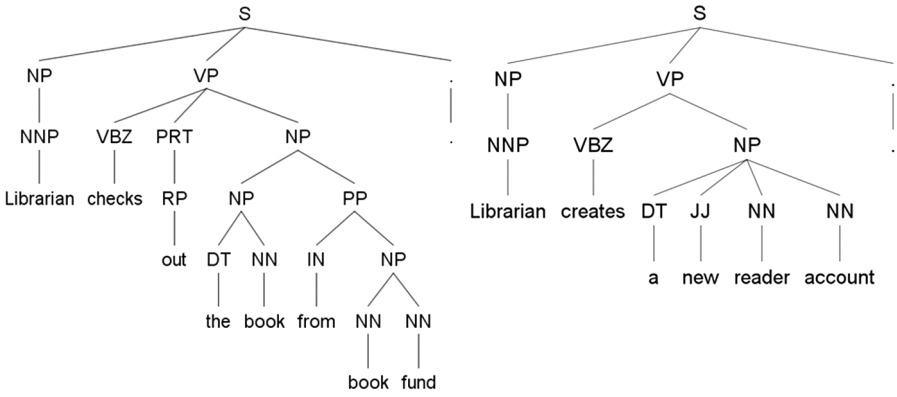

Fig. 2. Parse trees of use case sentences. The abbreviations are Part-Of-Speech tags according to [17]: S – sentence, NP – noun phrase, VP – verb phrase, NNP – proper noun, singular, VBZ – verb, 3rd person singular present, DT – determinant (article), NN – noun singular or mass, PP – prepositional phrase, TO – "to", PRT – particle phrase, RP – particle, IN – subordinating conjunction, JJ – adjective.

Let us analyze the first sentence in Fig. 2. First an action of the corresponding functional feature has to be identified. In this case it is the verb phrase (VP tag) of the sentence – "checks out the book from book fund". It consists of the object action (checks), the result of the action (book) and object involved in the action (book fund). The responsible entity for the action can be determined by comparing the actors list of the use case and the noun phrase (NP tag). In this case the noun phrase is "Librarian" and there is "Librarian" in the actors list as well, so the entity responsible for the action probably is "Librarian". Preconditions can be determined by analyzing the first steps of use case's sub-variations and extensions. If the current functional feature is represented as the first step of use case main scenario, then one additional precondition will match the precondition of the use case itself. If current step has a sub-variation, then the functional feature represented by the next step will have a precondition that is the opposite of the sub-variation condition. For example, sub-variation "If book is not available in the book fund, librarian denies the book request form" will result in a precondition "Book is not available in the book fund" for functional feature "denies the book request form", but an opposite precondition for functional feature "checks out the book from book fund". Use case extensions define their own precondition; obviously the condition in

the extension's sentence is the precondition of the functional feature represented, but the opposite precondition for the next step. Functional feature's subordination can be determined only by the user of the TFM Tool.

By analyzing use case sentences it should be possible to derive functional features. It is important that TFM Fetcher considers functional features the same if they are represented by the same tuple. This means that no duplicate functional features are created and two or more use cases can include the representation of the same functional features.

6 Retrieving Topology

Once there is a set of functional features it is necessary for TFM Fetcher to retrieve the topology of TFM or cause-effect relations between functional features. The structure of use cases will help with this task.

First of all, every use case's main scenario is an ordered sequence of functional features. Additionally, by analyzing the extensions and sub-variations it is possible to detect branching in a TFM. Extension adds an effect to the functional feature represented by the step referenced by the extension. On the other hand, sub-variation adds an effect to the functional feature represented by the previous step referenced by the sub-variation. Therefore, the setting of cause-effect relations between functional features represented within the same use case is very straightforward. As you can see in Fig. 3 the 4 main sequences of functional features come from main scenarios of use cases. As a demonstration of use case extension and sub-variation analysis consider functional features number 1 and 11. Functional feature number 1 has an additional effect because of the sub-variation 2a, but functional feature 11 has an additional effect because of the extension 4a.

A different task is setting the cause-effect relations between functional features fetched from different use cases. Precondition section of use cases are used to define this relation, because it contains the use case step which is the cause of the particular functional feature. For example, use case's "Requesting a book" precondition is "Librarian authorizes reader status", which is the 3rd step of use case's "Arriving" main scenario. Moreover, as different use case sentences represent the same functional feature if their tuples conform, relation between different use cases can be fetched from extensions and sub-variations, too.

Fig. 3 shows the compliance between the steps of use cases and the fetched TFM. By analyzing the use cases it is possible to derive a TFM. TFM Tool will support several iterations back and forth between description and TFM Fetcher until the system analyst can verify every functional requirement. The mapping between use case sentences, functional features and TFM should be intuitively illustrated and easily editable, so that any incompleteness, redundancy or inconsistency could be corrected. The main functioning cycle must be defined and set by the analyst. Cause-effect relation between functional features 13 and 2 in Fig. 3 is set by the system analyst for the completeness of main functioning cycle. It cannot be determined automatically.

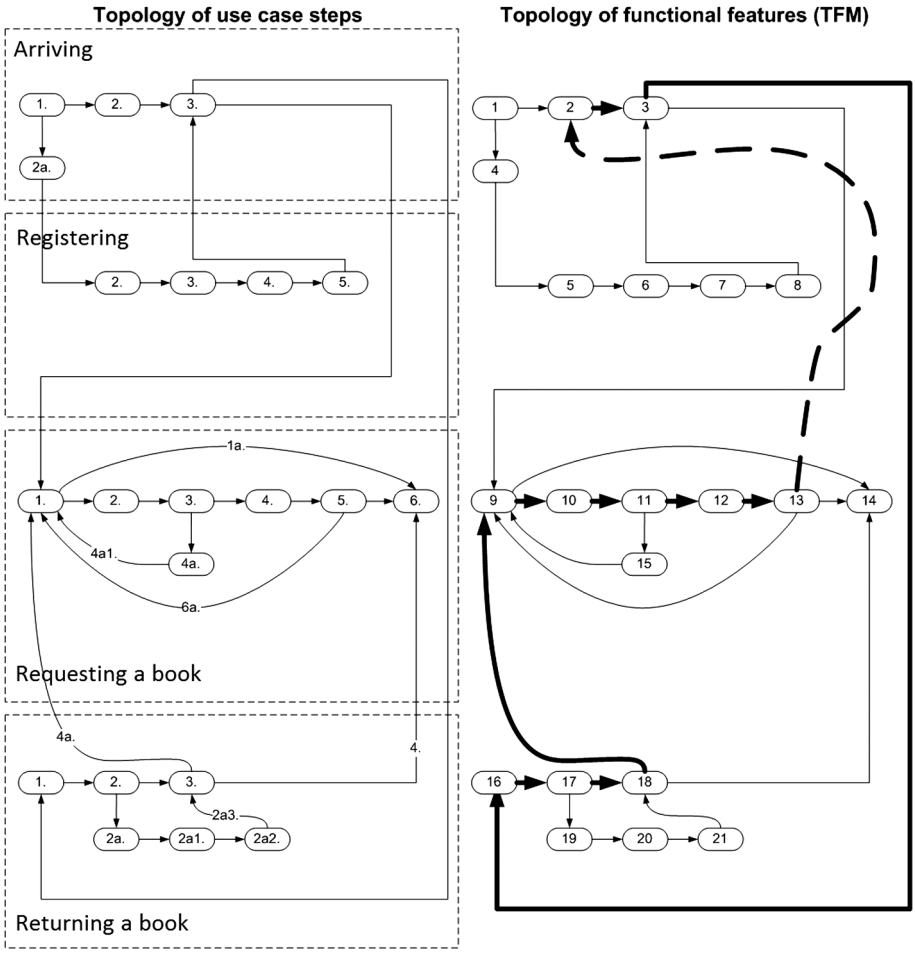

Fig. 3. This shows the topology use case steps in compliance with cause-effect relations between functional features. The bold arrows represent the main functioning cycle of TFM. Functional features of this example will not be listed in this paper.

7 Solution

This paper analyses the specific text analysis tasks at the beginning of MDA life cycle: 1) defining a formal data structure for the knowledge about the system; 2) retrieving functional features for TFM; 3) retrieving topology or cause-effect relations between functional features, thus constructing a TFM for the system.

As discussed earlier, business use cases with a specific template are used to define the knowledge about the system. This is a promising solution because use case driven development is widespread approach in software development. Another advantage is that a MOF compatible metamodel can be created for this use case template using XMI,

as well as for a TFM, which already is defined in [9]. A statistical parser can be used for analyzing the sentences of use cases, and thus retrieving functional features for a TFM of the system. This is a straightforward task as long as the sentences of use cases are kept as simple as possible and in simple present tense. This is not always possible, so it is the task of system analyst to prevent incompleteness, redundancy or inconsistency of use case sentences. At last, for retrieving the cause-effect relations between these functional features the structure of the use cases is exploited.

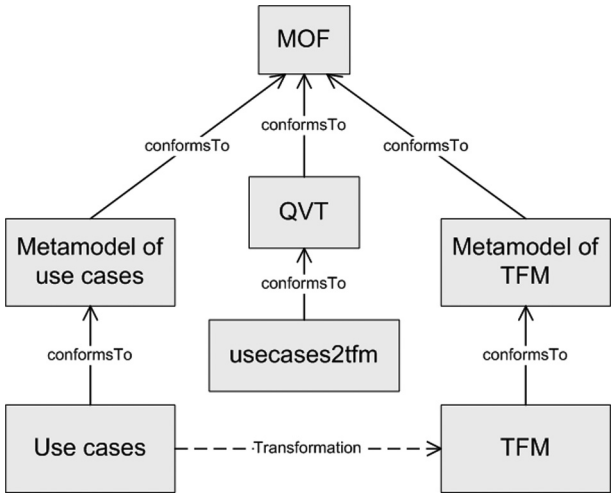

Fig. 4. This shows a scheme of the transformation between a set of use cases that describe a system and a TFM. The definition of transformation itself is usecases2tfm which conforms to QVT.

So we have a metamodel of use cases (a set of use cases), a metamodel of TFM and a set of methods to transform use cases of a system into a TFM for that system. At this point MDA's standard for model transformation QVT comes into the picture. The solution for the tasks defined earlier can be expressed as an exogenous model transformation [3] with the help of QVT. QVT provides the necessary domain-specific languages to define the transformation between use cases and TFM, including a QVT/BlackBox mechanism for integrating existing non-QVT libraries or transformations like a statistical parser. The scheme of transformation is presented in Fig. 4. There are some workable implementations of QVT like Eclipse M2M.

8 Conclusions

This paper discusses the specific text analysis tasks at the beginning of MDA life cycle in context of TFM4MDA method like defining the knowledge about a system in a formal data structure, challenges of retrieving a formal model from this knowledge represented by use cases, implementing a workable transformation between a set of system's use cases and its TFM.

Nowadays software developers often are occupied with similar pattern application coding. MDA proposes to abstract from application source code to the model of the application as the main artifact in software development. Until now in MDA context everyone has his own opinion about what is a CIM. This paper suggests that TFM should be considered as the CIM of a system and proposes data structures and methods for fetching a TFM from system's use cases. Thus a mathematically formal and transformable CIM of a system is acquired.

Further research is related to the evolution of the TFM Tool bringing its functionality closer to TFM4MDA. First thing in queue is implementing the TFM Fetcher. It has to be able to automatically retrieve functional features from its use cases by using a statistical parser and then by the use of model transformation transform use cases to a TFM of the system. For this task a metamodel for a set of these use cases have to be developed and a model transformation conformable to QVT.

Next task is to develop a TFM Transformer component which would transform TFM to UML conforming to TFM4MDA. As mentioned before it will probably be a special UML profile to keep all the valuable information of the TFM. So firstly there is a need for a specific TFM UML profile. Secondly Eclipse offers UML2 and UML2 Tools which can be applied for dealing with TFM Tool's problem of TFM to UML profile transformation. From this point it should be possible to generate some part of the system's code.

With advancements of this TFM Tool research the completeness of MDA will improve. TFM4MDA provides a formal CIM and new horizons by partially automating and improving system analysis.

References

1. Frankel, D.S.: Model Driven Architecture: Applying MDA to Enterprise Computing. Wiley, Indianapolis (2003)
2. Gasevic, D., Djuric, D., Devedzic V.: Model Driven Architecture and Ontology Development. Springer, Heidelberg (2006)
3. Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification, *http://www.omg.org/cgi-bin/doc?ptc/07-07-07.pdf* (02.05.2010)
4. Osis, J.: Topological Model Of System Functioning (in Russian). In: Automatics and Computer Science, J. of Acad. of Sc., pp. 44--50, Zinatne, Riga (1969)
5. Osis, J., Asnina, E., Grave A.: Computation Independent Representation of the Problem Domain in MDA. J. Software Eng. vol. 2, iss. 1, 19--46, *http://www.e-informatyka.pl/e-Informatica/Wiki.jsp?page=Volume2Issue1* (2008)
6. Kontio, M.: Architectural manifesto: Choosing MDA tools, *http://www.ibm.com/developerworks/library/wi-arch18.html* (2005)
7. Asnina, E.: The Formal Approach to Problem Domain Modeling within Model Driven Architecture. In: 9th International Conference on Information Systems Implementation and Modelling, pp. 97 – 104. Prerov, Czech Republic, Ostrava (2006)
8. Osis, J., Asnina, E.: Enterprise Modeling for Information System Development within MDA. In: 41th Annual Hawaii International Conference on System Sciences, pp. 490. HICSS, USA (2008)
9. Šlihte, A.: The Concept of a Topological Functioning Model Construction Tool. In: 13th East-European Conference, ADBIS 2009, Associated Workshops and Doctoral Consortium, Local Proceedings, pp. 476--484. JUMI, Riga, Latvia (2009)
10. Francu, J., Hnetynka, P.: Automated Generation of Implementation from Textual System Requirements. In: Proceedings of the 3rd IFIP TC 2 CEE-SET, pp. 15--28. Brno, Czech Republic, Wroclawskiej (2008)
11. Structural Requirements Language Definition, Defining the ReDSeeDS Languages, *http://publik.tuwien.ac.at/files/pub-et_13406.pdf* (May 2010)

12. Subramaniam, K., Liu, D., Far, B., Eberlein, A.: UCDA: Use Case Driven Development Assistant Tool for Class Model Generation. In: Proceedin of the 16th SEKE. Banff, Canada, *http://enel.ucalgary.ca/People/eberlein/publications/SEKE-Kalaivani.pdf* (2004)
13. Overmyer, S., Lavoie, B., Rambow, O.: Conceptual Modeling through Linguistic Analysis Using LIDA, In: Proceedings of the 23rd International Conference on Software Engineering, pp. 401--410. Toronto, Ontario, Canada (2001)
14. Malan, R., Bredemeyer, D.: Functional Requirements and Use Cases, *http://www.bredemeyer.com/pdf_files/functreq.pdf* (May 2010)
15. Jurafsky, D., Martin, J.H.: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Pearson Education (2000)
16. The Stanford Parser: A statistical parser. The Stanford Natural Language Processing Group, *http://nlp.stanford.edu/software/lex-parser.shtml* (May 2010)
17. Part-Of-Speech Tagging Guidelines for the Penn Treebank Project (3rd revision, 2nd printing), *http://www.ldc.upenn.edu/Catalog/desc/addenda/LDC1999T42/TAGGUID1.PDF* (May 2010)

SCIENTIFIC PAPERS, UNIVERSITY OF LATVIA, 2010. Vol. 757
COMPUTER SCIENCE AND INFORMATION TECHNOLOGIES 23–29 P.

Clarifying a vision on certification of MDA tools

Antons Cernickins
Riga Technical University, Institute of Applied Computer Systems,
Kalku Street 1, Riga, LV-1658, Latvia
antons.cernickins@rtu.lv

Abstract. As software systems tend to play a major role in many areas of activity, the certification of these systems or even small software applications becomes a task of growing importance [1]. Moreover, a software development life cycle itself is also considered as a matter of certification concern [2], especially with the emergence of new approaches. However, the main concern for a new approach is its acceptance from the side of the industry. In particular, the proposal of Model Driven Architecture shifted the process of software development towards modeling [2]. The actual implementation of MDA as an approach still lacks the appropriate implementation framework, resulting in incompliance among software development tools. This article proposes a vision on conceptual framework to be used as a foundation for certification of MDA tools. In particular, this includes the development of certification schemes to assess the compliance of the tools with incorporated standards from various perspectives.

Keywords: Model Driven Architecture, software development life cycle, certification, software development tools

1 Introduction

The continued integration of IT into various areas of activity facilitates the increase in dependence on the reliability, availability, and integrity of software systems [3]. Due to the overall complexity of modern hardware, software, and the ways of communication, the development of quality systems has become both a major scientific and engineering challenge [3].

However, the verification of developed system is also a matter of high importance, especially in safety-critical software systems (e.g., in domain of civil aviation), where system failure must be avoided [3]. Verification process done by a developer cannot guarantee the appropriate level of quality for such systems; usually, a third party is also incorporated into verification. Furthermore, the third party should be independent (i.e., should not represent the supplier or the acquirer side) to produce an objective and complete judgment of the system. Thus, in order to approve that any specific product meets specific requirements or conforms to particular standards, a certification procedure hold by an independent third party is carried out.

Possible benefits from certification procedure are the following: it offers more certainty about or confidence in developed software systems [3]. It helps in software sales, giving more confidence for prospective clients. Certification is also valuable, because the developer of the software can be sure that their system will operate in predictable way as specified in the standards [3].

Software development standards provide complex methods and approaches for defining, specifying, visualizing, and documenting the artifacts of software systems [4]. Furthermore, software tools have been developed to support these artifacts. Since the most current trend in software development is shifted towards modeling [2], the development of software artifacts is being driven by models. In order to support the model-driven trend, as well as to make sure, the process of modeling is done equally (i.e., same rules, notation, etc.), a branch of new standards for software modeling and architecture specifications has been proposed [5]. Model Driven Architecture (MDA), introduced by OMG in 2001, covered a wide range of these specifications [5].

However, the implementation of MDA approach is still challenging enough: there is a lack of information on how the support for MDA approach should be provided [2]. This results in incompliance with proposed standards, because each tool vendor implements these standards in his own way. Due to incompliance between software tools, a model of a software system created in one environment cannot be adequately used in another environment.

The original article contains a research on Model Driven Architecture, reviewing the MDA-oriented software development life cycle in the context of developing a quantifiable certification scheme to assess the compliance of MDA support tools with OMG standards. The goal of the article is to propose a vision on conceptual framework to be used as a foundation for certification of MDA tools. This article represents the most current state of the art in author's doctoral studies.

In turn, the hypothesis intended to be proven in doctoral research is associated with the possibility to develop a quantifiable certification scheme along with a set of rules and guidelines on how to assess the compliance among MDA support tools. The practical value of the research is the development of the certification scheme itself, including the appropriate framework, rules, and guidelines.

The article is organized as follows. Section 2 overviews the background, as well as provides a brief review of related work. Section 3 outlines the vision on conceptual framework to be used for certification of MDA tools. Finally, Section 4 concludes the article and provides pointers to further work.

2 Background and Related Work

The idea lying behind the research is to provide a set of guidelines on the actual implementation of the MDA for the purpose of promoting it as a holistic approach for software development across the IT community. A branch of standards provided within MDA is defined in a form of specification, meaning that the specification-based testing may be used as a basis for compliance assessment [4]. In particular, the conformance statement for CORBA provided by The Open Group [6] is done this way. In fact, the compliance itself is nothing else but the satisfaction of software implementation to the standard specification [4]. [4] comes with a idea of considering the compliance test

suite generation as a branch of constraint satisfaction problem, in which the first-order predicate is given and processed to find models that satisfy it. Following this work, instead of starting from a concrete set of constraints and trying to find the appropriate models, the construction (as well as the further classification) of all possible models is considered.

When it comes to development of a new certification scheme, the first and the foremost task is to define the object of certification [1]. According to [1], the following types of certification are possible:

- Product certification (accordance with particular technical standard);
- Process certification (accordance with ISO 9000 or similar standard);
- Personnel certification;
- Accreditation of certification bodies (the certification of certifiers).

[1] summarizes the study on various certification schemes and categorizes them into several groups, also providing a general structure of certification process itself, as well as presenting a new certification scheme used in space technology.

In fact, the type of certification procedure for current research can be determined as a combination of both the product and the process certification. Such a mixture of types will provide a more detailed outlook on various options to be considered in the certification scheme.

Basically, the former type of certification is considered, as software development tools (i.e., software products) are involved in the research. This may also include the specification of the most common features and options defined to clarify the accordance level of each tool from various perspectives (discussed in [2]).

As far as MDA-oriented software development life cycle represents the process, the latter type of certification should also be considered.

In order to provide a solid background for the certification scheme, as well as to clarify the means of the MDA tool as such, [7] is considered. [7] reviews the MDA approach within the variety of the CASE tools, which are proposed as supporting for MDA activities. The goal of the following research is to investigate the variety of the CASE tools, which are proposed as "MDA compliant," in order to classify them in accordance with the previously defined MDA tool specification. The provided specification of MDA tools consists of seven categories, specified in a hierarchical way flattened in the table (categories are divided into subcategories, subcategories—into groups, and groups—into single entries, accordingly) [7]:

- Accordance with MDA-oriented life cycle—the accordance level of software development life cycle supported by a tool, which includes MDA-oriented activities combined into such subcategories as knowledge formalization (CIM), system model refinement (PIM), PIM-to-PSM mapping, system model implementation (PSM), and transformation support;
- Functional capabilities—the functional capabilities of a tool in such fields as environment, modeling, implementation, testing, documenting, project management, configuration management;
- Reliability—the capability of a tool to maintain the appropriate level of performance under certain conditions for a certain period of time, including repository management, automatic backup capabilities, data access management, error processing capabilities, as well as fault analysis capabilities;

- Usability—usage efforts and individual assessments of such usage, including user interface, licensing and localization options, ease of use, quality of documentation etc.;
- Efficiency—the amount of resources needed to maintain the appropriate level of performance under certain conditions, including technical requirements, workload efficiency, as well as performance;
- Maintainability—efforts needed to make specified modifications;
- Portability—ability of a tool to be transferred to another environment.

3 Vision

In order to clarify a vision on a certification scheme to assess the compliance of MDA tools, a conceptual framework is proposed. In fact, this framework should be used to verify the output produced by MDA tools. Whereas a wide variety of the tools intended for specific purposes (e.g., mapping definition) may be used [2], an additional specification-based assessment of these tools is considered (discussed in [2]).

In short, the following four layers are used to describe the MDA-oriented software development life cycle [2], [5], [8], [9]:

- Computation Independent Model (CIM)—represents the high-level specification of what the system is expected to do (i.e., describes the domain and requirements of the system). It might consist of a model from the informational viewpoint, which captures information about the data of a system;
- Platform Independent Model (PIM)—specifies the functionality of a system. It might consist of a model from the informational viewpoint, which captures information about the data of a system, and a model from the computational viewpoint, which captures information about the processing of a system;
- Platform Specific Model (PSM)—specifies the implementation of system's functionality on specific platform. It might consist of a model from the informational viewpoint, which captures information about the data of a system, and a model from the computational viewpoint, which captures information about the processing of a system, based on a specific platform;
- Implementation Specific Model (ISM) or source code—describes the implementation of a system in source code of specific platform.

However, the only layers to be specified and promoted by OMG (i.e., described in details) are PIM and PSM [8]. In fact, OMG does not provide any specific requirements for CIM (meaning that it is not "computational", not formal enough, etc.), as well as ISM itself—the actual source code generated from PSM—from modeling perspective looks out of scope. Despite this, all four layers are somehow covered by various software development tools.

The conceptual framework considers these four layers as individual blocks, each of them having their own input and output (Fig. 1). The origin of this idea has come from black box testing [10]: whereas software system is considered as a black box, the only thing to be analyzed is the output produced by specific input. Therefore, tester does not need to understand why the compiled code does what it does; here, the requirements are used to determine the correct output of black box testing.

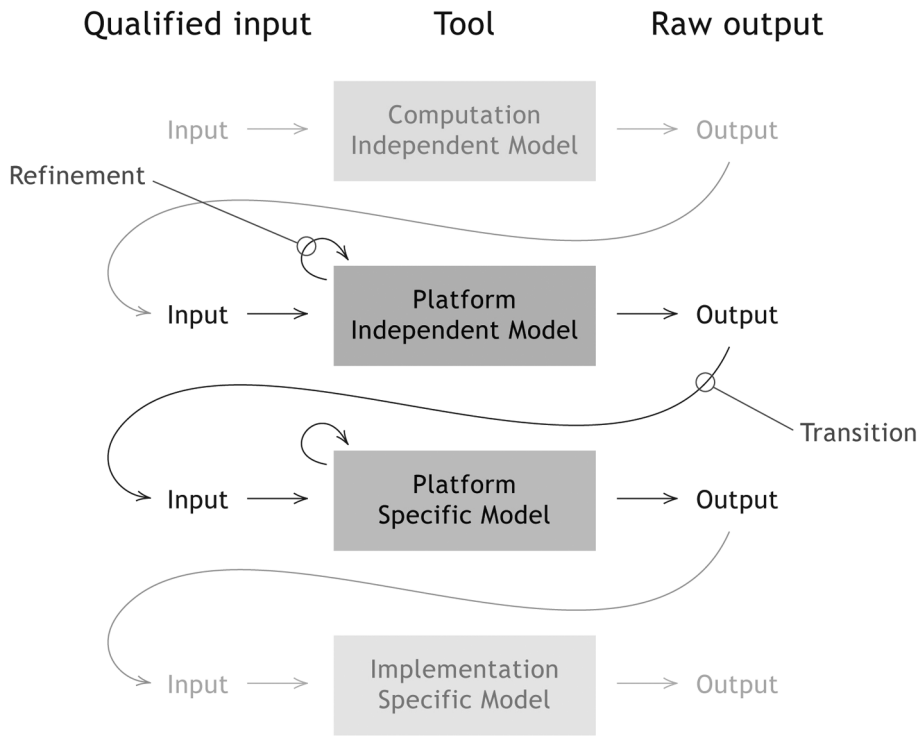

Fig. 1. Graphical representation of the conceptual framework

In fact, the main artifacts for the conceptual framework are inputs and outputs. As far as CIM and ISM are out of scope from the perspective of OMG standards, the conceptual framework does not cover the according artifacts (Fig. 1). The actual tool use on each block (i.e., what operations are performed) is also not the matter of high importance.

However, the main concern for each tool is the support of XMI standard [11]. In order to perform a transition from raw output to qualified input, the conceptual framework assesses the output from each tool. If tool conforms to OMG standards, then the output from this tool should be opened in other tool with no problems. If not, the conceptual framework would provide an appropriate suggestion on where the root of the problem lies.

While OMG does not provide any constraints (i.e., does not restrict) on the modeling language notation used with MDA (however, the use of UML is strongly recommended) [5], [8], the use of XMI for assessment of software development tools seems to be the only valuable option.

3.1 Implementation

In order to implement the conceptual framework, the development of a software tool is considered. The main concern of this tool is proper parsing and analysis of the XMI documents. As far as the technical standard of XMI like any other standard continue

to evolve (the most current release is 2.1.1 [11]), a modular software design model is considered in the developed tool.

The specification of XMI standard as such is used to create the XML Schema of XMI standard [12], which provides a means by which the syntax and the semantics of an XMI document can be validated. XMI Schemas must be equivalent to those generated by the XMI Schema production rules specified in [11]. Equivalence means that XMI documents that are valid under the XMI Schema production rules would be valid in a conforming XMI Schema; in turn, those XMI documents that are not valid under the XMI Schema production rules are not valid in a conforming XMI Schema [11].

After the XML Schema of XMI standard is created, the developed tool creates a document data model, which consists of [12]:

- Vocabulary (element and attribute names);
- Content model (relationships and structure);
- Data types.

This model is used for further validation of XMI documents. Validation can determine whether the XML elements required by [11] are present in the XML document containing model data, whether XML attributes that are required in these XML elements have values for them, and whether some of the values are correct.

After the document has been successfully parsed and analyzed, the developed tool outlines elements that are incorrect from the specification viewpoint, providing a "clean representation" of XMI document.

4 Conclusion and Future Work

This article outlined a vision on conceptual framework to be used as a foundation for certification of MDA tools. The ideas stated in this article are intended to be proven in doctoral research, with an aim to develop a quantifiable certification scheme to assess the compliance of MDA support tools with OMG standards.

In prospective, the conceptual framework described in this article will be used as a basis for XMI validation utility. However, this is yet another component in software certification scheme, which should be used in a combination with the others.

In extending this work, the next step is the development of several reference models to cover the whole MDA-oriented development life cycle. These will provide a step-by-step guide for the developers. As for modeling notation, the use of UML is considered. According to [13], a minimal set of UML diagrams is already defined. In turn, a recommended and complete set of UML diagrams should be proposed, as well as mutually compared.

References

1. Schäbe, H.: A Comparison of Different Software Certification Schemes, *http://www.sipi61508.com/ciks/schabe1.pdf*
2. Cernickins, A., Nikiforova, O.: On Foundation for Certification of MDA Tools: Defining a Specification. RTU 50th International Scientific Conference, Computer Science, Applied Computer Systems (2009)
3. Heck, P., Klabbers, M., van Eekelen, M.: A software product certification model. In: Software Quality Journal, vol. 18 (1), pp. 37–55. Springer Netherlands (2009)

4. Bunyakiati, P., Finkelstein, A., and Rosenblum, D.: The Certification of Software Tools with respect to Software Standards. IEEE International Conference on Information Reuse and Integration (2007)
5. MDA Guide 1.0.1. Object Management Group, *http://www.omg.org/docs/omg/03-06-01.pdf*
6. CORBA 2.3 Conformance statement template, *http://www.opengroup.org/csq/csqdata/blanks/OB1.html*
7. Cernickins, A.: An analytical review of Model Driven Architecture (MDA) tools. Master's thesis. Riga (2009) / Čerņičkins, A.: Modelvadāmās arhitektūras rīku analītisks apskats. Maģistra darbs. Rīga (2009)
8. Mellor, S., Scott, K., Uhl, A., Weise, D.: MDA Distilled: Principles of Model-Driven Architecture. Addison-Wesley, San Francisco (2004)
9. Alhir, S.: Understanding the Model Driven Architecture, In: Methods & Tools 2003, pp.17–24. Martinig & Associates (2003)
10. Sommerville, I.: Software Engineering (8th edition). Addison-Wesley, Wokingham, (2006)
11. MOF 2.0/XMI Mapping, Version 2.1.1, *http://www.omg.org/spec/XMI/2.1.1/PDF*
12. XML Schema, *http://www.w3.org/XML/Schema*
13. Nikiforova, O.: Object-oriented System Analysis. Drukatava, Riga (2007) / Ņikoforova, O.: Objektorientētā sistēmanalīze. Drukātava, Rīga (2007)

SCIENTIFIC PAPERS, UNIVERSITY OF LATVIA, 2010. Vol. 757
COMPUTER SCIENCE AND INFORMATION TECHNOLOGIES 31–42 P.

A Collaborative Web-based and Database-based Meta-CASE Environment

Rünno Sgirka[1],
[1] Department of Informatics, Tallinn University of Technology, Raja 15, 12618 Tallinn, Estonia
runno.sgirka@gmail.com

Abstract. Meta-CASE systems are used for the creation of CASE (Computer Aided Software Engineering) systems. In this paper, a web-based meta-CASE system is presented that uses an object-relational database system (ORDBMS) as its enabling technology. It allows us to integrate different parts of the system and to simplify the creation of meta-CASE and CASE systems. The proposed system includes a query subsystem, which allows developers to use queries to evaluate and gradually improve artifacts.

Keywords: CASE, meta-CASE, object-relational database system, SQL, queries, consistency, completeness, software measure.

1 Introduction

Meta-CASE systems are used for the creation of CASE systems. In [1], [2] and [3], we have introduced a web-based and database-based meta-CASE system, which allows us to create web-based and database-based CASE systems. As a web-based system, it allows distributing the CASE systems more widely, opposed to many traditional CASE systems which are thick-client programs that must be installed and run on the host computer. In addition, web-based systems make the group collaboration between users of the systems more available. Delen et al. [4] write: "Second, there is a need for a completely Web-based integrated modeling environment for distributed collaborative users."

As a database-based system, our meta-CASE system uses an object-relational database management system (ORDBMS) as its *enabling technology*. More precisely, it uses an ORDBMS, the database language of which conforms more or less to SQL:2003 standard [5]. Fugetta [6] explains that *enabling technologies* are integrating platforms that provide extensions to traditional operating systems and support "the creation of logically integrated but physically distributed systems". If a system uses a DBMS as its enabling technology, then developers of the system can use the system-defined (built-in) features and extensibility mechanism of the DBMS to simplify the creation of the entire system. Dittrich et al. [7] analyze the use of DBMSs in the field of software engineering

and conclude that modern systems provide "a rich set of functionality that is able to meet several requirements of software engineering applications in a satisfactory manner".

The author of this paper has chosen to continue the research and development of the aforementioned meta-CASE system in his Doctoral thesis. The system has been extended and improved since then. In [3], we introduced the query subsystem of the meta-CASE system and CASE systems that have been created by using the system. It allows us to take advantage of the powerful query mechanism of the ORDBMS and to create queries that can be executed based on artifacts. For instance, queries can be used to identify consistency and completeness problems of artifacts, to find how well developers have followed modeling guidelines, and to calculate values of software measures.

In this paper, we will demonstrate the use of the meta-CASE system by presenting a small case study. The metamodel of the *Family Tree Modeling Language*, introduced in the tutorial website [8] of one of the leading meta-CASE systems in the market, MetaEdit+ [9], is used as an example. Our meta-CASE system is mainly intended to be a research vehicle and not an industrial strength system and it currently provides less functionality compared to more matured meta-CASE systems.

The rest of the paper is organized as follows. In Section 2, our web-based and database-based meta-CASE system is described, together with its query subsystem. A small case-study using the *Family Tree* metamodel is presented to demonstrate it in practice. In Section 3, we outline topics regarding the future work with meta-CASE systems in general and with the current system. It is planned to study the topics in the Doctoral thesis of the author of this paper. Finally, some conclusions are drawn.

2 A Web-based and Database-based Meta-CASE System

In this section, the proposed meta-CASE system is introduced. Both the meta-CASE system and CASE systems that are created by using the meta-CASE system use an ORDBMS as their enabling technology. A prototype of the system has been created, using PostgreSQL 8.0.4 ORDBMS and PHP 5.0.5 scripting language.

In our system, there are two types of users. An *administrator* has to create a new CASE system by specifying a metamodel. In addition, an administrator has to register settings of the user interface and user identification of the CASE system. Administrators also have to manage queries. *Developers* (end-users) use the CASE systems in order to manage (create, read, update, and delete) artifacts. Developers can execute queries based on artifacts.

Fig. 1 presents the UML class diagram of the *Family Tree* metamodel presented as a case study in our system. We have extended the metamodel compared to the example in [8], by adding *parent_relation* and *parent_relation_type* classes. The former class represents the relationship between two parents; the latter class is a classifier which characterizes the relationship. These classes were added for better demonstration of using the queries to evaluate the artifacts.

In our meta-CASE system, each *metamodel* is used to describe the modeling language of a CASE system. For every metamodel, there is a corresponding SQL schema (an *artifact base*) in the database. For example, if an administrator creates a metamodel *family_tree*, then the meta-CASE system creates a database schema *family_tree,* which

is then used for storing the artifacts of the *family_tree* metamodel. The different schemas have been used to avoid possible name conflicts.

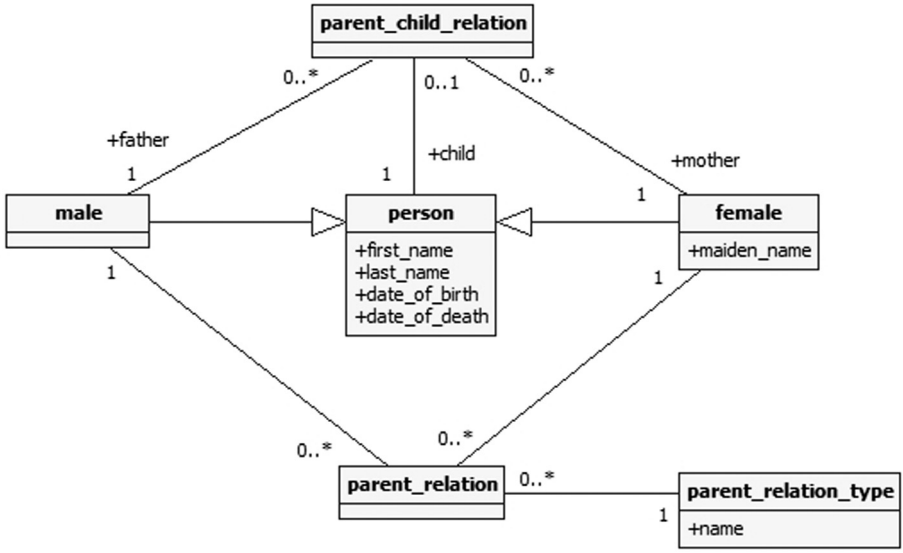

Fig. 1. An UML class diagram representing a Family Tree metamodel.

Each metamodel consists of zero or more *objects*. For every object, there is a corresponding *table* in the database schema corresponding to the metamodel. For example, if an administrator creates object *person* in the metamodel *family_tree*, then the meta-CASE system creates table *person* in the database schema *family_tree*.

Each object has exactly one type. Possible types are *main*, *inherited main*, *relationship,* and *classifier*. A *main object* is the most common one in our meta-CASE system, representing an element in the metamodel. It does not have the constraints describing a relationship object or a classifier object. For example, in the context of the *Family Tree* metamodel, the object *person* is a main object. An *inherited main object* is a main object which is derived from another main object, inheriting all the latter's characteristics. Objects *male* and *female* can be classified as inherited main objects, because both are derived from some other main object, in this case from object *person*. Our system makes use of the INHERITS clause of the PostgreSQL ORDBMS in the table definition statement. The non-standard INHERITS clause specifies "a list of tables from which the new table automatically inherits all columns" [10].

Each *classifier object* is used to characterize other objects in a metamodel. For example, object *parent_relation_type* is a classifier object, which characterizes *parent_relation* object. Possible values for *parent_relation_type* object can be *commitment*, *open marriage*, *marriage* and *divorce*.

Each *relationship object* represents a relationship between two or more main, inherited main, or classifier objects, thus supporting n-ary relationships. Currently our system does not support relationship objects, which connect a relationship object and another object. For example, to achieve the most accurate portrayal of the family *relationship* and parent/child *roles* described in MetaEdit+ [8], the counterpart in our

system *should* be built as follows: a relationship object (e.g. *family*), connecting a *male* and a *female* object, and a relationship object (e.g. *family_child*), connecting the *family* object and a *person* object. Since our system does not currently support relationship objects connecting to another relationship object, we have settled to build the family relationship as shown in Fig. 1: a relationship object *parent_child_relation* connects foreign objects *father*, *mother* and *child* (referencing *male*, *female* and *person* objects, respectively). In addition, to represent the relationship between parents, we have created relationship object *parent_relation*, which connects *male*, *female* and *parent_relation_type* objects.

Fig. 2 represents the screenshot of the objects of the *family_tree* metamodel in our meta-CASE system. Note that there are some additional objects generated by the system itself. Object *sysg_artifact* allows the developers to create different artifacts, which are all recorded in the same artifact base. All these artifacts have the same metamodel. Objects *sysg_user* and *sysg_user_artifact* are used for end-user (developer) management. Object *sysg_file* is used for storing files in the artifact base.

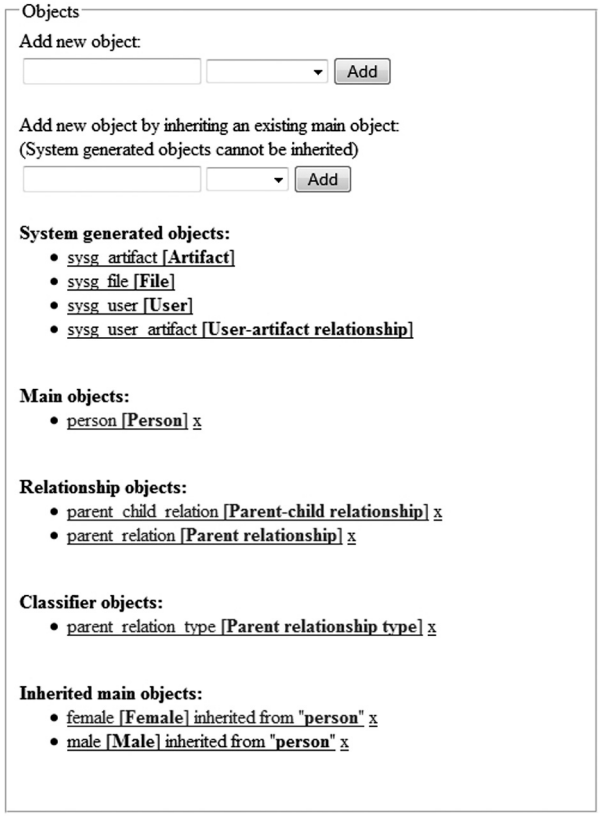

Fig. 2. A screenshot of the objects of the *family_tree* metamodel.

Every object consists of zero or more *sub-objects*. For every sub-object, there is a corresponding *column* in the database table corresponding to the object. The *type*

of the sub-object specifies the type of the corresponding column. For example, if an administrator creates sub-objects *first_name* and *last_name* with the type *varchar(100)* and sub-objects *date_of_birth* and *date_of_death* with the type *date* to object *person*, then the system generates columns *first_name* and *last_name* with the type *character varying(100)* and columns *date_of_birth* and *date_of_death* with the type of *date* to table *person*. If the sub-object is used to reference another object, then the system will generate a foreign key constraint to the corresponding column. This sub-object is called *foreign object* in our system. For example, if an administrator creates the foreign objects *father*, *mother* and *child* to object *parent_child_relation*, referencing to *male*, *female* and *person* objects, respectively, then the system generates columns *father*, *mother,* and *child* to table *parent_child_relation*, and adds foreign key constraint to each of them, referencing the tables *male*, *female,* and *person*, respectively.

Fig. 3 represents the sub-objects of the *parent_child_relation* object in our meta-CASE system. Note that all of the foreign objects are also part of the *additional unique identifier*. For relationship objects, this is done automatically by the system. Each column corresponding to the sub-objects in the additional unique identifier is included to the UNIQUE constraint of the table corresponding to the object. For objects other than relationship object, managing the additional unique identifier and therefore the UNIQUE constraint of the corresponding table is the responsibility of the administrator. If at least one sub-object is added there, the system automatically includes the sub-object *sysg_artifact_id* as well.

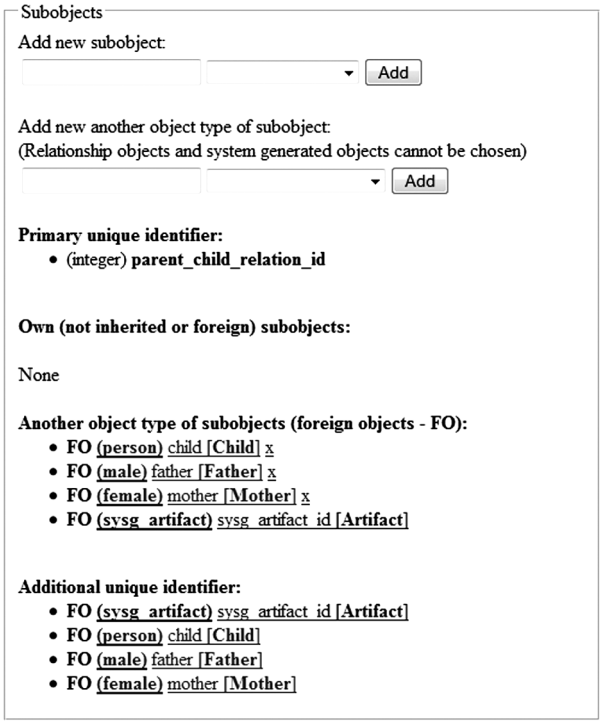

Fig. 3. A screenshot of the sub-objects of the relationship object parent_child_relation.

There are no sub-objects other than foreign objects in object *parent_child_relation*. However, note that in addition to the sub-objects added by an administrator, there is also a system generated sub-object present (*sysg_artifact_id*). This sub-object references the *sysg_artifact* object and is included in every main, inherited main and relationship object. It ensures that the corresponding tables in each artifact base contain identifiers of artifacts. It simplifies the creation of queries about a single artifact – the queries must perform the restriction operation based on column *sysg_artifact_id*.

Fig. 4 represents the objects of inherited main object *female* in our meta-CASE system. Note that the only sub-object not inherited is the sub-object *maiden_name*. Every other sub-object is derived from object *person* (including foreign object *sysg_artifact_id*).

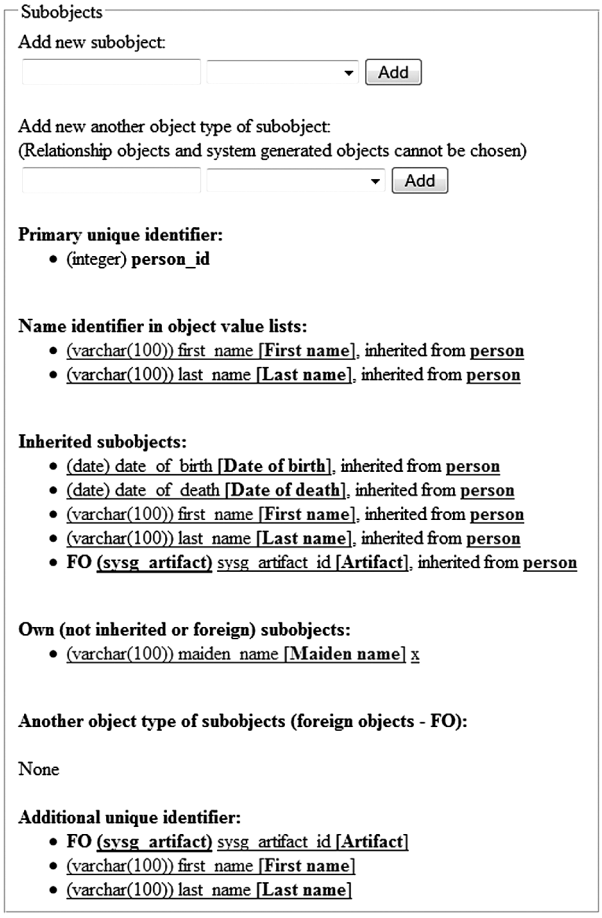

Fig. 4. A screenshot of the sub-objects of the inherited main object *female*.

The *name identifier* is used in our systems to specify the sub-objects which "represent" the objects to the developers in different lists (main menu, combo boxes etc). For example, an administrator has included *first_name* and *last_name* sub-objects

as name identifier for object *person* and the settings are consequently derived by objects *male* and *female*. If an administrator chooses to add object *person* or objects *male* or *female* to the main menu of the *Family Tree* CASE system, then the object instances are displayed there by using the values of sub-objects *first_name* and *last_name* (e.g. *Homer Simpson*, *Marge Simpson* etc). The name identifier is a decorative functionality only and has no corresponding constraint in the database, except that all sub-objects in the name identifier are also added to the additional unique identifier.

Before developers start using a CASE system, an administrator should define the *metamodel settings*. These include *setting up the main menu* of the CASE system, *managing the user settings,* and *adding classifier instances* that correspond to classifier objects. Setting up the main menu allows the administrator to determine the menu tree which is then displayed in the CASE system. By default, the highest element of the menu tree is always the *sysg_artifact* object and it cannot be changed, which means that developers have to choose an artifact instance in order to be able to manage artifact elements. Fig. 5 presents the main menu of the CASE system based on the *family_tree* metamodel. An administrator has added object *person* under the *Artifact* menu item, which means that *person* instances will be shown for each *sysg_artifact* instance, using the name identifier.

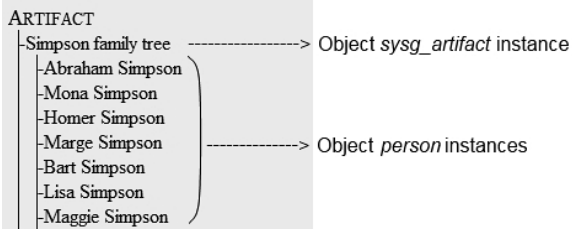

Fig. 5. A screenshot of the main menu of the CASE system based on metamodel *family_tree*.

The user settings management allows administrators to define whether all developers are allowed to see all the artifacts in the artifact base or only those, which are created by the developers themselves. In addition, an administrator can choose if the developers have to register first before accessing the CASE system or the CASE system is available to everyone without registering. System generated objects *sysg_user* and *sysg_user_artifact* are used to handle the necessary data. For example, an administrator states that the CASE system based on *family_tree* metamodel and all artifacts inside are available to all users without registering.

Finally, an administrator must add the classifier instances for each classifier object for developers to use. For example, an administrator adds instances *Commitment*, *Open marriage*, *Marriage* and *Divorce* to classifier object *parent_relation_type*.

Fig. 6 displays objects *parent_child_relation* and *parent_relation* in the CASE system based on the *family_tree* metamodel. A developer has added an artifact *Simpson family tree*, which contains object *male* instances *Abraham Simpson*, *Homer Simpson* and *Bart Simpson*, and object *female* instances *Mona Simpson*, *Marge Simpson*, *Lisa Simpson* and *Maggie Simpson*. All these values are also instances of object *person*.

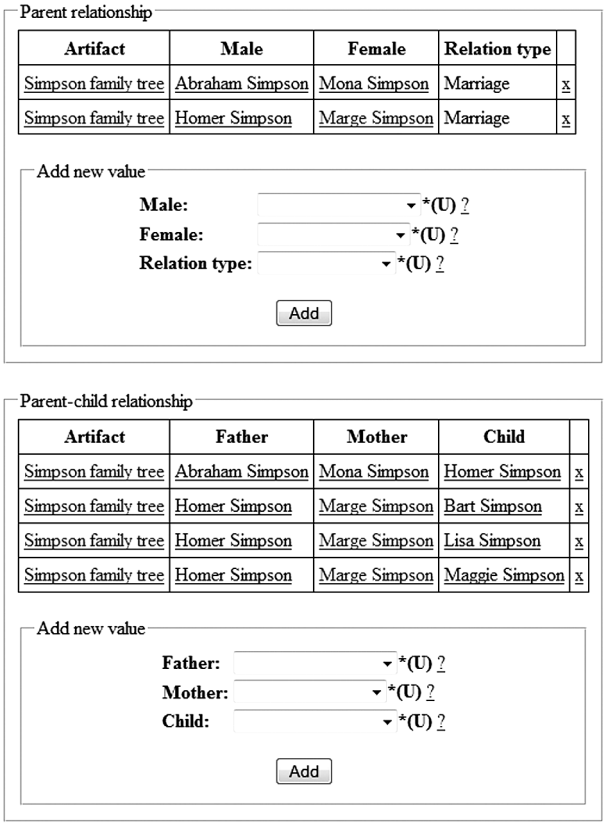

Fig. 6. A screenshot of objects *parent_child_relation* and *parent_relation* in the CASE system.

Administrators can use the *query subsystem* of the meta-CASE system to create queries that help developers to understand and improve artifacts. The queries (SQL SELECT statements) will be executed based on artifacts that are recorded in an artifact base. Simple and consistent syntax of a database language makes it easier to construct such queries and hence makes this kind of system more useful.

It is possible to create queries for different *purposes*, with different *scope* and expecting different *type* of *results*. Query purposes can include, for example, finding violations of constraints that give information about the semantics of the underlying metamodel of a CASE system, finding violations of design guidelines, or calculating values of software measures and general queries that cannot be classified to any of the previous category.

The scope of the queries can be exactly one artifact or all artifacts that are in one artifact base. In the first case, administrators can use generic placeholder #A# when specifying an artifact, using *sysg_artifact_id = #A#*. If a developer selects an artifact *af* and executes the query, then the system replaces the placeholder with the identifier of *af* before executing the query.

The expected results of queries can be either scalar values (for instance, with type Boolean; the results are tables with one column and zero or one row) or tables where

there are one or more columns and zero or more rows. For example, developers can use these queries to find elements of artifacts that have problems.

All the queries that can be used in the system have to be written by using the SQL dialect of the underlying DBMS (PostgreSQL™ in our case). The system allows administrators to save and check only SELECT statements. The statements cannot contain reserved words like INSERT, UPDATE or DELETE that are not delimited (not between “”). It is needed to prevent SQL *injection* attacks.

Choinzon and Ueda [11] present a set of software design measures together with their thresholds of undesirable value. Our meta-CASE system supports the use of *measure defect queries* that allow developers to determine design defects based on the values of measures. For each such query *q* there are one or more non-overlapping ranges of scalar values that have the same type as the expected results of *q*.

Administrators can construct *tests* based on Boolean queries and measure defect queries. The scope of the queries that belong to a test must be one artifact. Developers can execute these tests in order to evaluate artifacts. For example, a test *t* contains queries *q1,..., qn*. Each query in a test is a subtest of this test. For each query that belongs to *t*, administrators have to define a set of acceptable values. If a developer executes *t* in case of an artifact, then *t* fails if at least one query, which belongs to t, fails.

We have created some queries, which allow the developers to check the artifacts created by using the *family_tree* metamodel.

Let us assume that two parents have to have a relationship in order to have children. The following query checks whether there are parent instances in an artifact that have children, but have no relationship. If there are any, the query will return TRUE. In our case study (the *Simpson family tree* artifact), there are no such parents and therefore this query returns FALSE.

```
SELECT public.is_empty('SELECT pr.* FROM parent_relation
pr, parent_child_relation pcr WHERE pr.male = pcr.father
AND pr.female = pcr.mother AND pcr.sysg_artifact_id =
#A#') AS result;
```

IS_EMPTY is a user-defined function that has one parameter, the expected value of which is a query (SELECT statement) *q*. If the execution of *q* results with a table that has zero rows, then the function returns TRUE. If the execution of *q* results with a table that has more than zero rows, then the function returns FALSE.

The following query is an example of a query, the answer of which will be found based on all the artifacts that are recorded in an artifact base. This query allows developers to find the average amount of children in a family over the whole artifact base of *Family Tree* models. In our case study, this query returns “2.0”.

```
SELECT avg(children) AS result FROM (SELECT count(*) AS
children FROM parent_child_relation GROUP BY father,
mother) AS foo;
```

The table names in the query statements are *unqualified* names. If a user executes a query that is associated with a metamodel *m*, then the system modifies the schema search path in order to ensure that the result will be found based on tables that are in the schema that corresponds to *m*.

The following *general query*, the scope of which is one artifact, can be used to find children who are still underage (< 18).

```
SELECT p.* FROM person p, parent_child_relation pcr WHERE
p.person_id = pcr.child AND extract(year from age (p.date_
of_birth)) < 18 AND pcr.sysg_artifact_id = #A#;
```

3 Future Work and Conclusions

The proposed meta-CASE system is completely web-based. Users of the system need only a web browser and do not have to install additional plug-ins. An administrator must specify a metamodel and determine settings of the CASE system that is created by using the meta-CASE system. Based on this information the system is able to create data structures and dynamically generate web pages, through which it is possible to manage models. Dynamic generation means that if a modeler requests a web page, then the system determines the structure of the page based on data that is stored in the metamodel base. The system is in this sense similar to Oracle Application Express ™ development environment [12], which stores specifications of entire applications in an Oracle ™ database. The meta-CASE system is currently form-based and does not allow users to specify metamodels and artifacts by using a visual language.

There are several fields of research, which will further extend the functionality of the meta-CASE system described in this paper. The author of the paper considers the following topics as his future work and possible sections of his Doctoral Thesis. These topics are divided into three groups: *basic research*, *empirical studies* and *extending the functionality of the system*.

Basic research, the results of which are relevant in case of all meta-CASE systems

- *Ontological analysis* of the meta-metamodel of our meta-CASE system to find out whether it is flexible and powerful enough compared to the meta-metamodels of other meta-CASE systems. One has to define the methodology of such analysis.
- *Comparing* other existing meta-CASE systems to the system described in this paper, outlining positive and negative aspects. For evaluation one can use the Analytical Hierarchy Process decision framework [13] and also define a decision model, which can be then used to compare meta-CASE systems.
- Defining a set of *design pattern languages* for creating web-based CASE systems and using the patterns as building blocks of CASE systems.

Empirical studies

- *Empirical studies*, where different CASE systems will be defined using the meta-CASE system. Based on the feedback left by the users of these CASE systems, there might be a need to adjust the meta-CASE system. Examples of the CASE systems in question can be a *patterns management system* [14] or a system enabling the *framework for EIS strategic analysis* [15]. The latter does not currently have a specialized CASE tool.

Extending the functionality of the system

- *Version management subsystem*, which allows the developers to manage different revisions of artifacts, to restore an older state of an artifact, and to merge different

revisions to a new artifact. This can include *managing changes in a metamodel* to support the evolution of information systems and CASE systems that are used to develop evolving information systems. A metamodel is an artifact as well and the system should be able to manage its revisions like with every other artifact.

- The system should allow developers to use *conversion rules*, which allow the system to display the artifacts in some alternate way (HTML webpage, XMI file etc.) or to generate program code, tests, documentation etc., based on the artifacts. The latter is an actual problem in Model Driven Development context where developers often leave the model aside to focus on writing program code; the added code is later left unsynchronized with the model.
- *Interconnection between different metamodels* (and therefore, CASE tools) by linking different artifacts. These artifacts can be based on the same metamodel or several different ones.
- Using *templates* in the query subsystem to simplify the composition of queries. There can be a number of base queries. For each of these base queries one can create a template, which consists of a constant part and a variable part. For creating a particular query an administrator can select a template, value the parameters and the system will generate a query based on this information. Many of these templates can be created based on generalized integrity constraints that one can use in conceptual data models. For instance, one can create a query template based on a multiplicity constraint [Parent]-1-----1.*-[Child] to check whether each parent has at least one child.
- Managing metamodels and artifacts by using a diagramming environment.
- Further development of the query subsystem for administrators to be able to define queries to check the *completeness and consistency of metamodels* (CASE systems). These queries can be run based on the system catalog of the DBMS.

The most important keywords regarding the meta-CASE system described in this paper are “web-based” and “database-based”. Using web-based systems can create more opportunities for group collaboration between and within development teams of information systems of globalised and networked organizations. Such web-based systems can also be used as the basis of communities that are interested in the study and use of a particular modeling language. Database-based systems provide built-in functionality of the DBMS as well as quite mature extension mechanism to simplify the development of systems.

The author of this paper has not yet found any other meta-CASE system which while being web-based and database-based itself, allows creating web-based and database-based CASE systems. There are many existing meta-CASE tools (e.g. MetaEdit+, MetaMOOSE [16], Pounamu [17]), which provide much more functionality than the meta-CASE system described in this paper. However, they all use a desktop application as a modeling tool, which requires the user to install and update the program. There are some web-based extensions, for example in Pounamu, but they focus more on using the metamodels (CASE systems) over the web instead of developing them.

There are many new modeling languages developed nowadays. However, many of them lack a development environment, which allows creating models using the language. Thus, a demand of this kind of environment as described in this paper, still exists.

Acknowledgments. This research was supported by the Department of Informatics in Tallinn University of Technology and the Doctoral Studies and Internationalisation Programme DoRa.

References

1. Sgirka, R.: The Design and Prototyping of Meta-CASE Analysis Environment (in Estonian). Master's Thesis. Tallinn University of Technology, Tallinn, Estonia (2008)
2. Eessaar, E., Sgirka, R.: A Database-Based and Web-Based Meta-CASE System. In: International Conference on Systems, Computing Sciences and Software Engineering (SCSS 08) (2009) (in press)
3. Eessaar, E., Sgirka, R.: A SQL Database Based Meta-CASE System and its Query Subsystem. In: International Conference on Systems, Computing Sciences and Software Engineering (SCSS 09) (2009) (to appear)
4. Delen, D., Dalal, N.P., Benjamin, P.C.: Integrated modeling: the key to holistic understanding of the enterprise. Commun. ACM 48, 4, pp.107--112 (2005)
5. Melton, J.: ISO/IEC 9075-2:2003 (E) Information technology - Database languages - SQL - Part 2: Foundation (SQL/Foundation) (2003)
6. Fugetta, A.: A Classification of CASE Technology. Computer 26, December 1993, 25--38 (1993)
7. Dittrich, K., Tombros, D., Geppert A.: Databases in Software Engineering: A Roadmap. The Future of Software Engineering. In: 22nd International Conference on Software Engineering, pp. 293--302. ACM New York (2000)
8. MetaEdit+ Evaluation Tutorial, *http://www.metacase.com/support/45/manuals/evaltut/et-Title.html*
9. Tolvanen, J., Rossi, M.: MetaEdit+: defining and using domain specific modeling languages and code generators. In: The International Conference on Object Oriented Programming, Systems, Languages and Applications 2003, pp. 92--93 (2003)
10. PostgreSQL 8.3.9 Documentation, *http://www.postgresql.org/docs/8.3/static/*
11. Choinzon, M., Ueda, Y.: Design Defects in Object Oriented Designs Using Design Metrics. In: 7th Joint Conference on Knowledge-Based Software Engineering, pp. 61--72 (2006)
12. Oracle Application Express 3.1 Documentation.
13. Saaty, T.L.: The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation. McGraw-Hill (1980)
14. Zub, D., Eessaar, E.: Pattern-Based Usability Evaluation of E-Learning Systems. In: International Conference on Engineering Education, Instructional Technology, Assessment, and E-learning (EIAE 07), pp. 117--122 (2008)
15. Roost, M., Kuusik, R., Rava, K., Veskioja, T.: Enterprise Information System Strategic Analysis and Development: Forming Information System Development Space in an Enterprise. In: International Conference on Computational Intelligence. pp. 215--219 (2004)
16. Ferguson, R., Parrington, N., Dunne, P., Archibald, J., Thompson, J.: MetaMOOSE – an object-oriented framework for the construction of CASE tools. In: The 1st International Symposium on Constructing Software Engineering Tools (1999)
17. Zhu, N., Grundy, J., Hosking, J.: Pounamu: A Meta-Tool for Multi-View Visual Language Environment Construction. In: 2004 IEEE Symposium on Visual Languages - Human Centric Computing, pp.254--256 (2004)

SCIENTIFIC PAPERS, UNIVERSITY OF LATVIA, 2010. Vol. 757
COMPUTER SCIENCE AND INFORMATION TECHNOLOGIES 43–52 P.

Model-driven secure system development framework

Viesturs Kaugers, Uldis Sukovskis
Riga Technical University, Faculty of Computer Science and Information Technology, Riga, Latvia
{viesturs.kaugers, uldis.sukovskis}@rtu.lv

Abstract. Security is definitely one of the most important aspects in business information systems. This aspect is strongly related to costs, risks and reputation of organization. That's why authors want to propose "secure-by-design" principle by using innovative and proven development approaches in whole system lifecycle to maintain high security level. There are already some existing techniques to solve this problem but they are mainly linked with specific technologies and most frequently focus only on production phase. This paper presents new secure system development methodology based on three general aspects – model-driven secure code development, model-driven policy development and usage of run-time security management system to maintain necessary security level. All these aspects are integrated into one framework.

Keywords:, security, MDA, model, development, framework, UML.

1 Introduction

Model is a representation of system or real-world entity. Metamodel is model for models – it describes model notation. Model-driven Architecture (MDA) is a software development method which uses models to separate the specification of the information system from the details of the implementation in specific platform. Unified Modeling Language (UML) is the basis of model description. MDA was developed by Object Management Group (OMG) in 2001 [1]. OMG is international, open membership, non-profit computer industry consortium since 1989 [2]. This organization has defined four model types – Computation Independent Model (CIM), Platform Independent Model (PIM), Platform Specific Model (PSM) described by a Platform Model (PM), and an Implementation Specific Model (ISM) [3].

MDA tends to be solution for successful overall system development methodology. It gives possibility to introduce much more formal approach in early phases of design and development helping to avoid problems in later phases. This could be done because of detailed system description in high abstraction models. Practice shows that the main source of problems in system development project and later usage is caused by bad design. Time spent on system design is balanced with automatic code generation possibilities. Although currently there is no complete and universal code generation tool situation may change in a few years and developers already can use half-assisted code

generation tools. The code generation process is based on transformation rules described in Query/View/Transformation (QVT) standard. There are also other standards defined for various activities like XML Metadata Interchange (XMI) for metadata exchange and Meta Object Facility (MOF) for model-driven engineering [4].

2 Analysis and design of security aspects

Security nowadays is a critical factor in software systems. Problems with information system security affect both developers and customers [5].

Secure system maintains four main qualities – integrity, confidentiality, authentication and availability. If system does not have the qualities mentioned above, it becomes under risks like identity theft, data alteration, unauthorized access, service interruption and others.

There is a clear need in the IT market for secure software development methods to avoid problems caused by bad information system design, faulty security policies, inadequate implementation and human factor influence. It is of high importance to understand potential risks and view system in context of all organization structure, needs and resources. The most common faults include use of weak cryptography, inadequate user management, unreasoned authorization algorithm, inefficient authentication mechanisms and improper or incomplete data validation [6].

Model-driven system design approach increases system maintainability and security level and it will have significant impact on IT systems in future [7]. This is especially important for systems dealing with money, transactions, human lives and secret information. On the other hand office applications and smaller software can be successfully developed without MDA approach. So there is a domain where system development with MDA can give the best results. MDA has several advantages important also for business environment like fast adaptation to changes (changes in model not code), architecture portability among platforms (transformation from PIM to PSM, rules) and easier to trace errors and fix them.

The main goal of model-driven security is to integrate security features and requirements directly into system architecture, make it secure-by-design. So it is easier to link system design with enterprise security policy. It is also much easier to find and fix bugs in high abstraction level rather than in code. Practice demonstrates that part of bugs in code happens due to incomplete system design [6].

Models are more understandable than code and give an overall view of system being developed. Also high level models are easier to explain to top management staff rather than specific technical data and code.

Below there are described two languages for defining, designing and modeling security mechanisms. Both languages are delivered from UML and are used at PIM level.

2.1 SecureUML

One of secure development approaches is to use SecureUML. This is a modeling language or modeling extension based on UML and role-based access control (RBAC) model. So currently SecureUML is mainly focused at access control. RBAC is more

progressive authorization model than classical access-control lists because of possibility to define what kind of operation or activity user can perform on object [8]. For example, with RBAC administrator can define not only access rights to database but also what kind of data user is allowed to change in specific database table.

SecureUML offers to use some concepts like authorization constraints and state conditions to design more dynamic authorization process. By using SecureUML it is also possible to define security policies. SecureUML metamodel extends UML metamodel with types *User*, *Role*, *Permission* and *ResourceSet* and others [6]. Every UML model element can be protected resource and belong to set of resources for defining permissions and constraints. *Permission* is an object which connects a role to a *ModelElement* or to a *ResourceSet*. Element *ActionType* describes permission or allowed actions for protected resource. For example, developer can assign attribute *MaxLength: 255* to action *SetPassword* for user *John Doe*. Element *ResourceType* defines available all available action types for specific metamodel type. Element *BaseClass* defines connection to metamodel type. Element *AuthorizationConstraint* defines precondition for action to be executed on resource.

Simple example of UML/SecureUML model set is given below in Figure 1. This is basic model set for database access and management with defined users and roles.

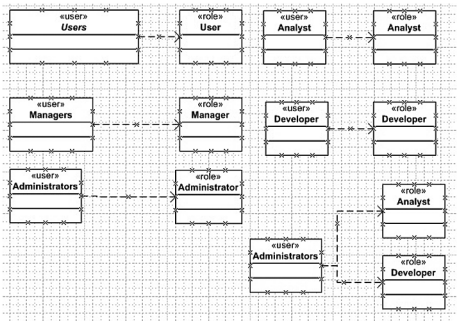

Fig. 1. Database users and roles

As it was mentioned earlier, SecureUML deals only with access control. This language would be more useful if it also could support integrity and data validation. So the authors offer to improve SecureUML with attributes and entities to support integrity, confidentiality and availability.

Integrity support can be achieved with *checksum* as attribute in *Permission* type – Figure 2.

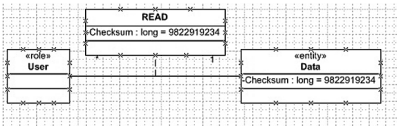

Fig. 2. Integrity support

Confidentiality requires data encryption mechanisms so we offer to add two new types *EncryptionMechanism* and D*ecryptionMechanism* which encrypts and decrypts data flow between *Users* – Figure 3.

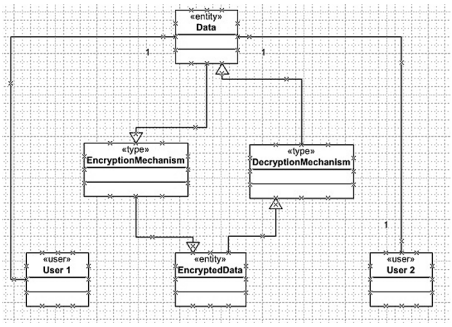

Fig. 3. Confidentiality support

Availability is a bit different characteristic and requires service to be available all time. To guarantee availability system has to be redundant, with alternative resources to perform necessary operation. In the case when main system is unavailable there should be link to alternative system as visible in Figure 4.

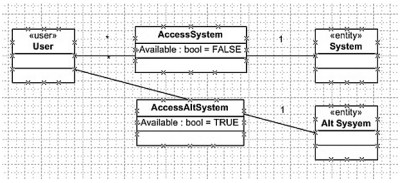

Fig. 4. Availability support

SecureUML notation is not quite suitable and needs to be enhanced for alternative link representation so we used two permissions with opposite attribute values to determine available system. On the other hand SecureUML adds additional layer of abstraction where developers can work directly on security features separated from business process diagrams. If this is not necessary they can choose UMLsec which naturally will fit into whole UML system model.

2.2 UMLsec

UML standard can be extended by using *profile*s, which consists of constraints, tagged values and stereotypes. Stereotypes allow defining new modeling elements. Tagged values are a pair of name-value. Constrains gives possibility to add additional information to model. Stereotypes attach tagged values and constraints to model elements so they define security requirements and assumptions in the system.

In UMLsec there are following stereotypes (Name (Base class, [<Tags>])) – *Internet* (Link), *Encrypted* (Link), *LAN*(Link), *secure links*(subsystem), *Secrecy*(dependency), *Integrity*(dependency), *High*(dependency), *secure dependency*(subsystem), *Critica*

(Object, subsystem, <secrecy, integrity, high>), *no down-flow*(Subsystem, <high>), *data security*(Subsystem), *fair exchange*(Subsystem, <start, stop>) and *provable*(Subsystem, <action, cert>). For example, stereotype *Internet* can be attached to the links between components so it describes how component can interoperate with it. So using mentioned stereotypes developer can design security mechanisms right inside UML model [9]. Deeper discussion of UMLsec is not the goal of this paper so we will give basic example of UMLsec usage in Figure 5.

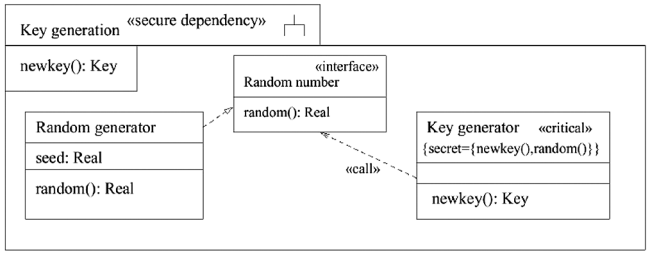

Fig. 5. UMLsec example [10]

3 Model-driven policy modeling

MDA approach is a useful method also for security policy development. It helps to create more consistent and manageable security mechanisms. By using qualification criteria which are common to all systems, designer can validate security model before and after usage and eliminate potential flaws. So security policy development with MDA consists of two phases – design and qualification phase.

In the first phase security policy is transformed into security components. One of the components is Policy Decision Point (PDP) – it is logical entity in the system that performs admission and decision control in response for access to specific resource [11]. Calls or calling points to a PDP for accessing resource are policy enforcement points (PEP). PDP and PEP graphical representation is given in Figure 6.

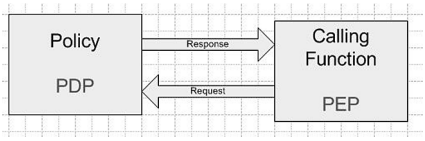

Fig. 6. PDP and PEP

Security policy is built using domain-specific language (DSL). DSL consists of all objects needed to represent policy and helps to automate the process because it provides necessary level of formalization. It is possible to generate automatically security framework, executable PDP and integrate PEP into business logic.

It is very important to ensure that PEP cannot be bypassed so there is a need for qualification phase. Qualification consists of initial security model verification before

component generation and concluding validation. Qualification environment is built on policy metamodel.

Security policy development process starts with requirement analysis then modeling, qualifications and execution come. All the process graphically is summarized in Figure 7.

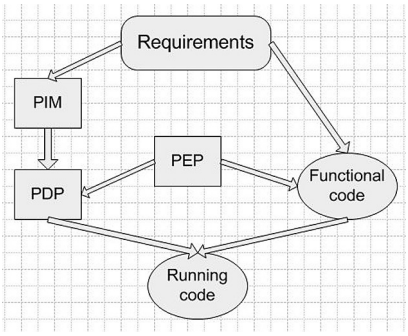

Fig. 7. Security policy design with MDA

Policy can deal with one or more security aspects like integrity, confidentiality, authentication and availability. For example, security policy can deal entirely with user access control. It means that policy model will include information about users, roles and actions. For large enterprise systems number of users can reach several thousands, every user can have some roles and many actions associated with it. So, for example, manual search for conflicting rules (one rule allows particular action, other denies) is very time-consuming job. Also at least one action needs to be associated with specific role. With MDA a developer can do these verifications automatically on platform-independent model. After PIM transformation necessary PDP are generated. PDP contains security policy implementations. These points receive requests from PEP which serves as interface between PDP and application and helps to integrate security policy software systems.

Using aspect-oriented programming business logic and security policy can be separated and security code effectively integrated into overall code of application. After integration phase comes final application testing with automated test cases [12]. Main testing goal is to ensure that security policy implementation corresponds to modeled security policy.

Security policies can be modeled using various metamodels for particular purposes. One of the metamodels is already previous mentioned RBAC. It defines all necessary concepts for access control security policy. Security policy development modeling phase is rather similar to that previously described for database access management. More detailed overview of MDA security policy development can be found in [9]. Security policies are part of model-driven secure system development framework. This is because whole system needs part where business needs and requirements are dynamically implemented and changed according to current situation.

4 Run-time security management

When software system is ready and working, it functions in dynamic and often in complex environment, linked with other systems and used by thousands of users. Software systems can be changed using patches or major upgrades. System analysts and software architects cannot predict and implement all security, usage and change scenarios at design phase, even using model-driven system development approach. On the other hand software users are interested in continuous security and reliability. There comes the challenge to maintain security for software system during all its life cycle. Run-time security management policies can be developed also in the way described in previous chapter. Such policies are understandable also for business executives and managers.

Run-time security management implementation requires defining specific security concept and requirement representation, also known as ontology, ongoing system monitoring and adequate adjustment to changes in external environment [13].

Construction of run-time security management includes finding and modeling alternative security solutions based on ontology, service discovery and adaptation mechanism. Result of construction is architecture for implementation with alternative security solutions for different situations and measurement system. Overview of run-time security management architecture is given in Figure 8.

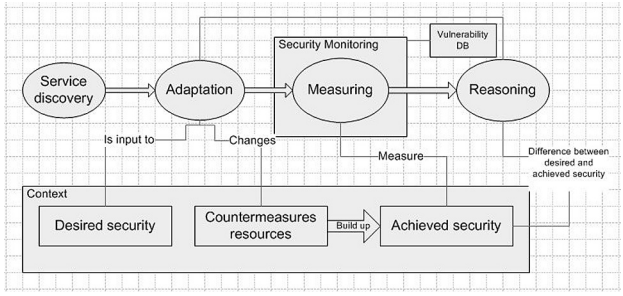

Fig. 8. Run-time security management architecture

In the beginning available services are discovered and adaptation performed on them according to their state and specifics. In adaptation process run-time security management system finds most suitable solution for security threat. After adaptation starts measurement process of base measures and system balancing according to obtained measurements. Necessary security level can change dynamically in response to situation. The whole process usually takes place without user interaction; exception is cases when system cannot maintain necessary security level [14]. In such situation it inquires assistance of the user. Run-time security management adds one more security layer for the system because it can respond to changing outer environment and automatically take preventive actions and generate easy manageable security rules. This system is a part of the model-driven secure system development framework where technical rules are enforced.

5 Model-driven secure system development framework

Model-driven secure system development framework addresses security issues and gives possibility to integrate security features in the whole system while offering innovative view on secure system development. Based on reviewed secure software development principles we are offering new approach that links all mentioned methodologies together. Overall view of the framework in high abstraction is represented in Figure 9.

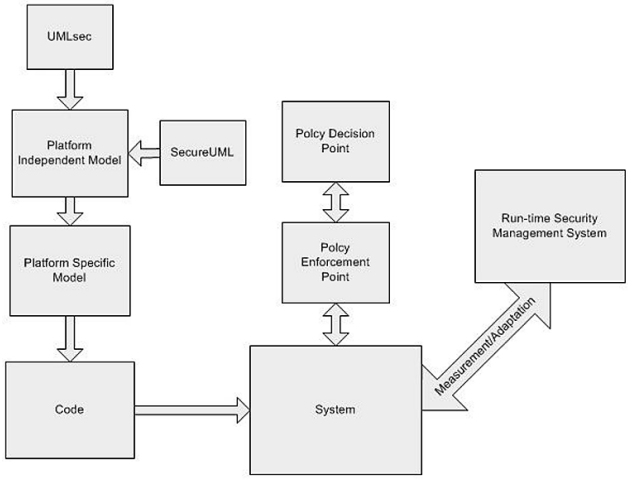

Fig. 9. Secure system development framework

This framework consists of 8 building blocks and system itself. The main goal of the framework is to integrate described security mechanisms in systematic way to provide appropriate security level in design and runtime. PIM is designed in conjunction with SecureUML or UMLsec to produce PSM and system code. System is linked with external security policy via PEP to maintain business rule changes in future. Run-time security management system protects main system from hazardous technical changes in operating environment. This approach ensures that security all considerations are taken into account and development process is model-driven.

PSM is generated from PIM and this is more technically detailed version of system architecture. Developers can mike minor adjustments here although it is recommended to work on PIM. Next level is code from which executable system is produced. In model-driven secure system development code shouldn't be changed manually/directly except if there is strong need for that. In such cases all changes must be reflected in higher level models.

System itself can be described using general system theory. This theory helps to understand working mechanisms of system without going in too much detail and is theoretical basis for the framework. It may be usable when system architects and developers communicate with business people to coordinate design process and gather necessary aspects. Basically it consists of various components and links between them (Figure 10).

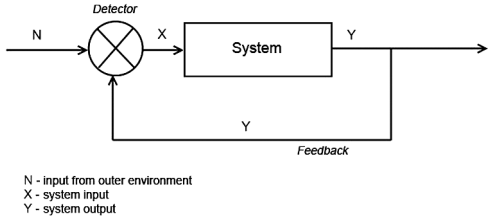

Fig. 10. Scheme of general system

In context of this framework system is controllable; it has inputs and outputs, detector and selector. The role of detector is performed by run-time security management system. Selector is implemented by policy decision point. Inputs may be any kind of business information, policy rules and control signals from run-time security management system. Outputs may be processed inputs. General system can be divided into smaller subsystems but they all apply to the same rules. In context of framework inputs are model and output is executable system.

One of the main advantages of secure system development framework is that external security policy and run-time security management system can be added and changed later during system usage. This is because framework prescribes connection points in design with cooperating systems. When main system is done, developers can continue their work on creation of mentioned systems. Of course development costs are higher in comparison with standard systems but added security requires additional expenses like in other secure system development methodologies.

The power of this framework lies in its flexibility. It is easier to integrate changing business requirements and technical measures if it is predicted in system during its development phase. Development of system can be divided into parts and every part is not limited to specific technology or approach. It is even possible, for example, to develop run-time security management system using rapid prototyping. Also PIM of the system can be developed in conjunction with security policy. Business people can easier deal with security policies and be involved in their development, because many technical things are hidden at this level. System administrators can benefit from run-time security management application which makes their job easier.

6 Results, conclusions and further research

In this paper we examined concepts of MDA applicable to system security, run-time security management, SecureUML and UMLsec languages and policy modeling. We described enhancement for SecureUML modeling language to extend its support for integrity, confidentiality and availability. Based on reviewed secure software development methods and principles we offered a new development framework that links all mentioned methodologies together. This framework aims to provide complete and integrated secure system development methodology

Future work will focus on deeper analysis and description of secure system development framework, making it detailed. Aspects of policy and run-time security

management system development require more research. Framework needs to be verified in real-life development projects. At some point it may be even possible to split system development between different developers to speed up project movement. This also needs to be researched in upcoming work.

References

1. Frank Truyen. The Fast Guide to Model Driven Architecture. Cephas Consulting Corp, 2006, 16 p.
2. About the Object Management Group. Prepared by OMG.: Object Management Group – http://www.omg.org/gettingstarted/gettingstartedindex.htm. – visited at February 2010.
3. An introduction to Model Driven Architecture. Prepared by Alan Brown.: IBM – http://www.ibm.com/developerworks/rational/library/3100.html. – visited at February 2010.
4. Joaquin Miller, Jishnu Mukerji. MDA Guide Version 1.0.1. OMG, 2003, 62 p.
5. The Importance of Information Security for Financial Institutions and Proposed Countermeasures. Prepared by Bank of Japan.: Bank of Japan – http://www.boj.or.jp/en/type/release/zuiji/kako02/fsk0004b.htm. – visited at February 2010.
6. Rudolph Araujo, Shanit Gupta. Design Authorization Systems Using SecureUML. – Foundstone, 2005, 16 p.
7. Model-Driven Security: Enabling a Real-Time, Adaptive Security Infrastructure. Prepared by MacDonald N.: Gartner – http://www.gartner.com/DisplayDocument?id=525109. – visited at February 2010.
8. David F. Ferraiolo, D. Richard Kuhn. Role-Based Access Controls. – Baltimore MD: National Institute of Standards and Technology, 1992, 11 p.
9. P. Shabalin, J. Jürjens. Towards Tool Support for UMLsec (Poster Proposal). – Dep. of Informatics, Munich University of Technology, Germany, 2003, pp. 1-3.
10. Jan Jurjens. UMLsec: Extending UML for Secure Systems Development. – Dep. of Informatics, Munich University of Technology, Germany, 2002, pp. 1-9.
11. PDP – Policy Decision Point. Prepared by Christine Martz.: Birds-Eye – http://www.birds-eye.net/definition/p/pdp-policy_decision_point.shtml. – visited at February 2010.
12. Tejeddine Mouelhi1. et. al. A model-based framework for security policies specification, deployment and testing. – Berlin: Springer, 2008, 15 p.
13. Antti Evesti, Eila Ovaska, Reijo Savola. From Security Modelling to Run-time Security Monitoring. – Finland: VTT Technical Research Centre of Finland, 2009, 11 p.
14. Ulrich Lang, Rudolf Schreiner. Managing business compliance using model-driven security management. - Vieweg&Teubner, 2009, 241 p.
15. General Systems Theory. – http://www.istheory.yorku.ca/generalsystemstheory.htm. – visited at February 2010.

SCIENTIFIC PAPERS, UNIVERSITY OF LATVIA, 2010. Vol. 757
COMPUTER SCIENCE AND INFORMATION TECHNOLOGIES 53–63 P.

Application of CobiT Maturity Model in Information Security Management and Arising Problematic Issues

Dmitrijs Nogicevs
University of Latvia, Faculty of Computing, 19 Raina Blvd., Riga, LV 1586, Latvia
dmitrijs_n@inbox.lv

Abstract. This paper examines a simplified decision making and achieving process that complies with the management of Information Security. The problems to determine a current position are investigated, as well as their impact, in order to diminish the risk of making a faulty decision, which can originate when the current position is incorrectly or inaccurately defined. The work focuses on the following problem: due to imperfect and defective methods used for determination of a current position, in addition to the great impact this process has on the determination of objectives and activity planning, besides the management of the both and IT overall management, there persist great risks of improper decision making, position defining and consequently the use of resources. Not only the decision making process, but also the current methods, as well as alternatives, of determination of Maturity Model (MM) and Maturity Level (ML) are analysed. By forecasting the development of a process, the guidelines for the abatement of the existent imperfections are offered. Decision making stages, particularly the determination of the current position, the use of equitable indicators, automatic collection of data and interconnected processes are being emphasized throughout the paper.

Keywords: CobiT, Maturity Model, Maturity Level, Information Security.

1 Introduction

In recent years, Security Maturity Model (ISMM) and Information Security management issues have been addressed in various studies, hence a sensible progress has been made – from Mikko Sipone to ISM3 [1], which is associated with the application of qualitative indicators. Likewise, the indirect relationship between security activity and security goals [2] has been inspected. ROSI as one of the main MM components is often mentioned in various studies, and in some of them ROSI related problematic issues [3] are covered, which, in my judgment, are more related to the selection of routes and activities as one of many, furthermore ambiguous criterions. By proffering new MM visions, relations, as well as congruencies between ISO and other standards, guidelines and ISMM have been explored [4], [5], and different MM have been compared [6]. This interest is coherent to the fact that MM can readily be put to use in the management of various processes, *inter alia*, Information Security processes.

Determination of the current state is one of the main stages combined into IT governance, decision-making and accomplishment of purposes. However, determination of a current state is required not only *per se*, but also in the context of future actions. Think about the purpose first, and then choose the method [7]. At present there are many MM visions. Avenues of approach vary for so do the purposes of MM application.

In this research process are observed from two points of view, first, the decision-making and the accomplishment of purposes, on the second hand. The role of MM in the process is defined and aims and recommendations are offered accordingly.

Different approaches are used for analysis, mostly CobiT and ISM3. The reasons are: **CobiT** – very popular (classic), relatively well developed, can be analysed together with CobiT Security Baseline (IT Security orientated); **ISM3** – new, IT Security orientated, use quantitative method.

Other reasons are given in the part 3.

2 CobiT and Decision Theory

In order to illustrate position and importance of the Maturity Model (MM) [8] in Information management processes, let us first consider its Decision making main stages (levels) and than the coincident CobiT/ISM3 tools.

Main stages of decision making and goal achievement process:

1. Determination of the initial (current) state;
2. Definition of the desirable position (target state);
3. Determination of the possible routes (from current to desirable position);
4. Selection of the best routes;
5. Checkpoint (sub target) setting (in order to maintain the control of the process and allow the rectification of the path in case of necessity)
6. Inspection, whether the desirable position has been reached.

The visualization of this process is given in the following figure (see Fig.1.).

Let us only briefly describe the process. Primarily, the current state is determined (C.S.). Then, taking the latter into consideration, the target state (T.S.) is defined, moreover – the desirable position is accessible and meets the organization's objectives. In order to shift from the current to the desirable position, the possible routes have to be determined. The routes are formed by successive activities (Act. x.y., where „x" denotes the route number and „y" – the activity number in respective route). Examples of possible activities are alteration of policies or procedures and notable projects, such as DLP (Data Loss Prevention) or IPS (Intrusion Prevention System).

Each activity leads either to the intermediate state or to a sub-target (I.S.x.y., shall „x" be the route number and „y" – the checkpoint number in a respective route). Sub-targets help maintain the control of the process (verification of the direction and position should occur). In case of necessity, revision of the list of activities (or checkpoints) can be implemented.

The best route is the one that satisfies the defined conditions. In our example it could be the route number 2.

As the last the check-out procedure is put in use in order to verify whether the target position is reached. It means that achieved position is compared with the desired by an application of appropriate measurements.

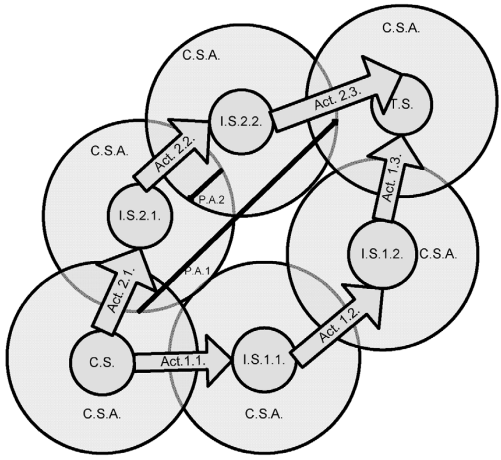

Fig. 1. Illustration of goal setting and achieving process.

Significantly, each time when a current position is determined it is done with a certain precision that has to be taken into account. Thus, there is actually a „current state area" (C.S.A.) rather than a definite position known, moreover the greater the C.S.A. the higher the risk of faulty activity planning. The Fig. 1. depicts a possible mismatch of a planned activity (for example, Act. 2.2.) and an activity that could be practically performed (P.A.2). Consequently, such initial irregularities can substantially affect the achievement of the marked goal. There exist the following risks:

1. The activities require greater resources than expected.
2. Activities cannot be successfully accomplished at the current state.
3. The accomplished activities have unexpected, or even negative, outcomes.

Table 1.

CobiT and ISM3 tools that can be used to support the decision-making process

The Main stages of decision making and consummation	CobiT[8], [2], [7], [10]	ISM3 [11]
Determination of the current state[1]	MM, (Detailed) Control Objectives (DCO)	system configuration, metrics, ISM3 levels
Definition of the target state	MM (Rising Star), KGI (Key Goal Indicators). BSC (Balanced Scorecard) can be used	Service quality (see 4.1.)
Determination of the possible routes	Practically does not support, CSF (Critical Success Factors), DCO [15]	Practically does not support, Process (description), Related Methodologies
Selection of the best routes[2]	Practically does not support, PO domen	Practically does not support, ROSI, Related Methodologies
Subtarget setting	KPI (Key Performance Indicators), see also Determination of the current state[3], MM, DCO	see Determination of the current state
Assessment of the target accomplishment	MM, KGI	see Determination of the current state

[1] Results of the Risk and Vulnerability Analysis have also been frequently used (applying the principles of priority) [12]

[2] ValIT [13], CRAMM [14] have been mentioned

[3] Correlation with Determination of the current state is low

We can now turn to the decision-making supportive tools provided by CobiT and ISM3

Some of the mentioned stages are also considered by such solutions as CRAMM Risk Matrix [14], PRINCE2 *et cetera*.

Functionally these stages form three groups of operations:

1. Determination of the initial position.
2. Detection of the possible ways how the target could be met.
3. Identification of the best route.

In the following sections we will focus on the current state detection process.

3 Determination of the State: Problems and Potential

In order to assess the stage of IT system development, many organizations use CobiT Maturity Model, which enables the determination of so-called Maturity Level or ML. This method allows the governing body to identify the current situation and to decide on needful and desirable steps that should be taken to improve the situation, as well as to realize the extent of required changes.

We will further investigate MM application related problems using CobiT MM as an example.

3.1 Problematic Issues Incidental to the Application of a Maturity Model Method

Since CobiT defines only general principles for the determination of ML, there is a high probability that the value subjectively influences the conclusions. The general principles allow to be treated variously [7], thus the positions can be determined (measured) inaccurately, consequently the targets, which are the purpose of the ML measurements, are met in an inefficient way.

The main disadvantages of the current (CobiT) method are its insufficient formalization, excessive usage of qualitative evaluation and a lack of quantitative measurements.

In the case of ISM3, the following obstacles can be mentioned:

1. It may be complicated to compare two states representing different fields (directions) in order to prioritize and boost the decision-making process.
2. Qualitative measurements are emphasized, thus leaving the vision to MM Control Objectives and activities poorly established (see 4.1). As a consequence, the process that has been carried out does not actually clarify, neither what one should do, nor in which direction.
3. The acquired results are equivocal (also due to the selection of indicators).
4. Process metrics, except for ML 5, are not compulsory. [11]

3.2 Alternative

We have concluded that the extant MM and ML methods are impotent to provide adequate results when applied for the determination of a current state. Despite the approaches' great potential, its current development does not comply with the requirements set by the process of decision-making and consummation. Therefore, there are the following possibilities:

1. To abandon the MM and use other approaches.
2. To explicate MM's avenue of approach by implementing existent tools or by modelling new ones in order to abate the imperfections mentioned above.

Continuing on the first possibility, an example of another approach is the keystone method which is commonly used in guidelines and standards, also in ISO (for example, 27001[16], 27002 that concern information security management [10] or ISO 9001). The application of this method provides the picture of an ideal (target) situation, but does not give any information about the possible inter-states, their interactions or developments. The keystone method can help compose a check list but the method is not useful when the stage of development or analysis of the current position is relevant. Besides it does not make provisions for the priority establishment and goal achievement which is possible if MM is applied. Only with maturity levels it is possible to show progress towards better security management [5].

Available metrics possess presumable complications such as, primarily, detection of those who are purpose-suitable and, secondly, metrics can be of little value when security management has to be ensured [2]. After consideration of the restrictions, we deduce that it would be suitable to develop the method on the basis of MM, applying the approach of keystone method as an adjuvant for composing ML questionnaires.

The current state determination methods should comply not only with metrics criteria, like S.M.A.R.T. [2], but also with the following:

1. They are uniform, ensuring that equate positions after processing give the same result.
2. It is possible to determine the position with a high precision, the C.S.A („current state area"; see Fig. 1.) is modest.
3. The level is defined in terms of controls and activities, thus facilitating the retracing the process of development.
4. The determined state can be compared with others (e.g., historic) correctly.
5. The components of the defined state can be evaluated and analysed asunder enabling the identification of weak (and strong) links.
6. It is possible to define and prioritize controls and activities.
7. These methods are perceivable by the concerned.

Conclusion:

Organizations are still after tools that could help determine the routes to which more attention should be paid in order to improve the consummation of the goal.

The analysis of alternatives has led to a conclusion that the most appropriate action to approach the solution of the problems associated with the determination of the current state is a further development of MM (using CobiT MM as the basis). The odds of other standards and guidelines should be used to improve checklists.

4 Suggestions for Improvement of ML Assessment Method

In order to improve CobiT ML model and to support the congruence with the criteria put forward, it would be rational to work out a checklist that could be implemented to

assess the current position. Please note that this checklist may touch fields irrelevant to some organizations (e.g., the controls for a shop differ from those of a bank). Moreover, it does not necessarily mean that the organization has a higher ML if some of the criteria are not topical.

It is possible to create a ML profile, which is categorized by fields. If an organization has no need for particular controls, they are not evaluated. Filters for requisite criteria can be made by comparing organizations and analyzing the progress.

4.1 Determination of ML: The Basal Principles

There are two main ways how ML of IT processes can be defined:

1. By inspecting the quality of IT management.
2. Considering the quality of IT services.

Let us now pay regard to each of them in a bit more detail.

Evaluation of IT management quality means that primarily, sufficiency of control measures are evaluated, secondly, control and process levels of development are defined. In the case of CobiT, the controls, which are essential to meet the targets, are chosen and it is said that the situation is good and the goals are achieved if only it can be agreed that all controls are sufficiently well-developed.

We can see that here the focus is on the query, whether the IT management is in correspondence with a good practice, rather than on – in truth – the main question, i.e., how safe the IT environment is. The latter remains unanswered.

Another drawback of this approach is the high probability of a mismatch of supposed IT security conditions with the actual situation in the organization. Such risk can be regarded as a multiplication of the following factors:

1. The risk of inaccurate determination of the current situation;
2. The risk to choose inappropriate directions;
3. The possibility to choose unsuitable controls;
4. Wrongly chosen activities for implementation of controls;
5. Faulty assessment of the factual impact the implemented controls have on the information security.

Therefore it is advisable not only to use the assessment of IT management quality, but also to consider the quality of respective services, which could help clarify the *de facto* situation (e.g., DS5).

If the evaluation of IT management shows good results (a fine development) but the evaluation of IT services – modest, unsatisfactory or even negative effect, the following questions should be considered:

1. Does the organization focus on the right IT security areas;
2. Are the controls correct;
3. Do the chosen activities ensure (effective) achievement of controls?

The conclusions could be helpful in planning and implementing activities in the future. On the other hand, if both approaches give a common result, there is a high probability of being on the right track.

Evaluation of IT service quality denotes the inspection of the actual situation. In order to screen the real situation by means of a checklist, ISO (27002 – in the case of information security) or ISM3 can be successfully applied. Yet another option for the question compilation is Universal Vulnerability Matrix which supports a risk-based approach.

The questions should be composed so that the possibility to make a subjective assessment would be diminished or excluded. Besides, they should ensure the comparability of data, i.e., with each other, as well as within time and with those of other organizations. Questions should be as uniform as possible (applicable to a wide range of organizations) and may be meeting numerous requirements usually characteristic of business continuity, availability, protection of classified data, compliance of the necessary regulatory requirements. It is advisable not to overload the questionnaire; otherwise its usage may become inconvenient.

4.2 Suggestions and Recommendations on Compilation, Completion and Analysis of a Questionnaire

The great bulk of suggestions and recommendations are represented in Table 2.

The General Guidance on Selection of Indicators:

1. In order to diminish the impact of subjectivity on the ML assessment process, quantitative, rather than qualitative, approach should be adapted.
2. To a feasible extent the quantitative indicators should be attained from statistical data, e.g., from databases (see Table 2., „Information Source"; paying particular attention to uniformity of configuration, i.e., principles of evaluation), but the remainder – from IS administrators (or technical process owners) provided with concise and unambiguous questionnaires (consequently, not time-consuming). Furthermore, the latter could also be useful for executive officers (to raise the level of understanding) or as a part of *Self assessment*.
3. Sufficient amount of time should be devoted to prepare the statistical data, including their verification, before the ML assessment (meeting) takes place. What is more, the data should tend be indisputable, so that subjective assessments, conflicts of interest and misunderstandings could be avoided.
4. The questions incorporated in the questionnaire should be grouped by common sub-process (controls) rather than by MLs, and formed on the basis of parameters characterizing the quality of a current state (Table 2, column nr. 1).
5. The questions should form short unambiguous sentences, each including only one statement. For instance, an affirmative sentence like this – *„security-relevant information is produced by systems, but not analysed*" – should be split into two separate sentences before being incorporated into the questionnaire.
6. The number of questions that may have more than one answer should be minimized. On the other hand, each question should be unique (should not be repeated), furthermore, independent (thus, two questions that are in conflict with each other, e.g., *„a small part is involved"* and *„the majority is involved"* – cannot be used within the same questionnaire).
7. To determine the current state more precisely, not only the indicators that cover a certain IT process, but also interactions between these processes (and identifiers)

should be incorporated in the ML determination procedure (see Interpretation of Results: Application of Interconnected Processes). Such approach enables the detection of so called „sensitive links"[4].

8. Relevant parameters and metrics have to be defined (Table 2.).

It should be taken into the consideration that the satisfaction with the system-functioning is by definition subjective, therefore it is impossible, as well as unnecessary to make an effort to replace all qualitative indicators with quantitative.

Guidance on Compilation of Assessment Questionnaires

Source material: Depending on the desirable accuracy of the assessment, it is advisable to choose within the ensuing openings:

1. CobiT Maturity Model Questionnaire, ISM3;
2. CobiT Controls [8];
3. CobiT CSF/KGI/KPI (Critical Success Factors, Key Goal Indicators, Key Performance Indicators) [15], CobiT Security Baseline [17];
4. Other organization-relevant standards (e.g., ISO 27002).
5. On the contrary, cumulative parameter compliance to requirements of regulators should be avoided due to following reasons:
   a. Such approach is resource-demanding, therefore would likely impede the ML evaluation process;
   b. Compliance with certain parameters is more essential than the total conformance.
6. A relative importance (weight) can be granted to each question. As means of assistance, ratings (and other methods embodied in Decision-making theory) can be used.
7. When compiling response options the criticality of the object should be considered (crucial IS or processes).

Algorithm for question compilation:

1. Set up the basic structure;
2. Use checkpoints or activities for questions and controls or parameters of quality (Table 2.) for response options;
3. Choose units (e.g., a percentage, a number or a certain state-indicator) and establish an evenly (<25-50-75-100) or unevenly (<80-90-95-100) divided scale;
4. Create a clear structure: group questions thematically. If applicable, extend the existing CobiT „*Maturity Model Attributes*" or ISM3 (this will help determine positions that call for further measures) and apply crucially characterizing parameters („Crucial Systems" (CS), „CS and up to 30 % of other systems" etc.).

[4] Sensitive link – a parameter that is tightly bound to several other indicators, thus even slight its improvement can well affect many other factors. Examples of such „sensitive links" are formalization of a training process and implementaton of examinations.

Collection and assessment of results. Results can be collected by heeding:

1. Directions (fields), e.g., *„Understanding and Awareness", „Goals and Metrics", „Security policies and procedures"* etc. according to CobiT [8] (please pay attention to Table 2.).
2. By processes.
3. Interrelationships (e.g., questions concerning existence of audit trails, their quality and application are all related).

The potency to gain information on different stages enables profound use of available data, thus, empowering the process analysis, consequently, better decisions can be made. Let us now learn how the assessment of results occurs. Let us start with an example.

Shall all answers correspond to ML 1, it is then not difficult to understand that the result (ML of the topic) is as well 1. This also applies to other ML, i.e., ML 2, 3 *et cetera*. Put in mind that the ML of each topic follows the answers to *all* corresponding questions unless the response option that would cover the ML under consideration is not provided. In such a case the question is left out in the particular phase of calculation. The latter grants avoidance of situations when affirmative responses (to all provided questions) within the second level result in ML 3.

When interrelationships are taken into account risk determination analogous treatment is applicable - probability should be multiplied by possible detriment.

Interpretation of Results: Application of Interconnected Processes. The next step within advancement involves the application of consequences emerging from interconnected processes, since indeed many IT processes are closely related, e.g., DS5 [8] (Information provision process) to M1 [8] (monitoring process).

Table 2.

Fragment of a sample questionnaire

	Maturity Level for "Ensure Systems Security" (DS5)						
Maturity Level	0	1	2	3	4	5	Information Source
Understanding and Awareness							
	Not used	Depends on the individual	Formalized and executed for critical	Formalized and executed for all systems, meets the regulator's requirements	3 + periodically reviewed	4 lvl + best practice	technical resourse holder/CMDB
Information classification (quality)							
System classification (quality)							
Authentication and authorisation							
Identity Management system used (for system, in %):	Not at all	Critical <50%	Critical - 100%	Critical, Total <40%	Critical, Total 41 - 90%	Critical, Total >90%	IDM/CMDB
The rights depository are used for (system, in %):							IDM
Data in the rights depository are verified with the data in systems (for system, in %)							IDM settings
User identification is standardised for systems							IDM, systems settings
User authentication is standardised for systems							IDM, systems settings
Data in the rights depository are verified with the data in systems (period)	Not verified	After request (not periodically)	Not all data, periodically, manually	Automatically, not one-line	On-line after changes	On-line after changes + 1 per day	IDM settings

The evaluation can be carried out in two distinct ways:

1) In the frame of DS5 evaluation process 2 questions showing relation with M1 development should be composed. One of them could focus on the range of monitoring (only CS, all systems etc.), the other – on the quality of monitoring, which could be expressed quantitatively;
2) The other approach is applicable when all the processes have already been evaluated (including allocation of relative importance's (weights)) and explicable by means of the following example.

If the weight of DS5 is 3.5, but that of monitoring is 2 and the impact weight is assumed as one twentieth or 0.05, then it can be deduced that monitoring affects DS5 negatively, hampering it by a value of (3.5-2)*0.05=0.075.

The main benefit of application of interconnected processes is the acquired opportunity to reveal the operations substantially impeding or accelerating others.

For instance, as mentioned before, monitoring impedes other processes with a weight of 0.5, while the Change Management - with only 0.25. Consequently, provided equal amount of resources is utilized, monitoring affects the situation more than Change Management. At the same time, these calculations are theoretical, furthermore, the costs to ensure these changes are not taken into account.

Application of results in situation modelling: An additional gain from formalized ML determination is the establishment and introduction after certain rules providing an insight into a generalized course of action. This helps to plan the sequence of necessary actions to approach to and achieve the desired goal.

5 Conclusion

On balance, determination of ML is a prerequisite to making decisions and meeting targets; at present methodologically the best tool to carry out this process is MM. The area for future inquiries is broad.

Primarily, the other stages of the complex decision making and consummation process (see section 2) should be investigated. Furthermore, the adaptation of the commenced approach in the continuing research is advisable, since it could ensure natural integration of the groups in the whole process while assuring effective problem solving and task accomplishment separately - in each of them.

Taking into account that ML should be determined in line with future activities (like implementation of new controls or alteration of the existent), MM helps select the most appropriate directions, or fields, (Table 2. provides an example) for the performance of activities, not the activities themselves. What is more, MM supports the determination of ML, rather than of possible or best routes.

MM requirements, as well as recommendations employable to assure meeting them have been given throughout the inquiry. Though it is proper to emphasize afresh that a greater attention should be paid to connections between processes, controls, states and activities. Information Security process is very complex, therefore deficiencies emerging from stage most likely could prohibit success in another.

Determination of ML and the development of other stages should support the decision-making process, as well as control mechanisms, for they help determine,

whether the goal has been achieved. Simultaneously the ML determination should advance toward boosting the situation modelling, which – as brought to front at the end of section 4.2 – could be possible providing the acquired results were accurately applied.

Situation modelling allows, first, operative audit (whether the (intermediate) target state is reached (see Fig. 1.) and credible forecasts, in the second place, regarding effects of alteration, advancement possibilities and risks. Moreover, it possesses a potential of being helpful in decision-making.

The issues faced in the research, as well as current MM and ISM management tendencies, have lead us to the following conclusions:

1. Other stages of decision making and aim achievement processes should be investigated on the given basis;
2. A clear and formal definition of aims and scope of MM application is needful;
3. Situation modelling should be encouraged;
4. Connections, including multidimensional, between various activities, controls and aims are supportable;
5. The use of Risk and Vulnerability approach should be encouraged.

References

1. Abhishek Vaish and Shirshu Varma: Proposed Next Generation Information Security Management Effectiveness Measurement Model, *http://amrita.edu/cyber-workshop/proceedings/icscf09_submission_101.pdf*
2. ISM3, *http://ism3.wordpress.com/*
3. Wes Sonnenreich, Return On Security Investment (ROSI): A Practical Quantitative Model, *http://www.infosecwriters.com/text_resources/pdf/ROSI-Practical_Model.pdf*
4. Saad Saleh AlAboodi, *http://www.itgi.org/Template.cfm?Section=Home&CONTENTID=34805&TEMPLATE=/ContentManagement/ContentDisplay.cfm*
5. ISM3 Comparet to ISO27001, *www.ism3.com/page2.php*
6. Lessing MM: Best practices show the way to Information Security Maturity, *http://researchspace.csir.co.za/dspace/bitstream/10204/3156/1/Lessing6_2008.pdf*
7. Erik Guldentops: Maturity Measurement—First the Purpose, Then the Method. J. Information Systems Control Journal, Vol 4 (2003), *http://www.isaca.org/Template.cfm?Section=Home&CONTENTID=16267&TEMPLATE=/ContentManagement/ContentDisplay.cfm*
8. The IT Governance Institute®, CobiT 4.1. *http://www.isaca.org/*
9. Andrea Pederiva: The COBIT Maturity Model in a Vendor Evaluation Case . J. Information Systems Control Journal, Vol 3 (2003), *http://www.isaca.org/Template.cfm?Section=Home&CONTENTID=16253&TEMPLATE=/ContentManagement/ContentDisplay.cfm*
10. ISO 27002, *http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50297*
11. ISM3 v2.10, *http://www.ism3.com*
12. Microsoft Corporation. The Security Risk Management Guide 2006, *http://www.microsoft.com/downloads/details.aspx?FamilyID=c782b6d3-28c5-4dda-a168-3e4422645459&displaylang=en*
13. Erik Guldentops: Enterprise Governance of IT Implementation and Assurance Advice when using CobiT and ValIT, *http://www.isaca.lv/gl/easyfile/index.php?folder=14*
14. CRAMM (v 5.1, v5.2), *http://www.cramm.com/*
15. CobiT Management Guidelines, *http://www.isaca.org/*
16. ISO 27001, *http://www.iso.org/iso/catalogue_detail.htm?csnumber=42103*
17. CobiT Security Baseline: An Information Security Survival Kit, 2nd Edition, *http://www.isaca.org/Template.cfm?Section=Research2&CONTENTID=36883&TEMPLATE=/ContentManagement/ContentDisplay.cfm*

SCIENTIFIC PAPERS, UNIVERSITY OF LATVIA, 2010. Vol. 757
COMPUTER SCIENCE AND INFORMATION TECHNOLOGIES 65–76 P.

A Framework for Optimal Selection of a Storage Architecture in RDBMS

Andreas Lübcke and Gunter Saake
School of Computer Science,
Otto-von-Guericke-University Magdeburg, Germany
{andreas.luebcke,gunter.saake}@ovgu.de

Abstract. Self-tuning techniques are widely used in the database community especially for large analyses systems. Within this domain, a new architecture known as column store comes up to analyse and aggregate data. Column stores demonstrate good results in real world examples and benchmarks like TPC-H. In recent years, row stores dominate the data warehousing domain. To the best of our knowledge, there is no advisor for the optimal selection of storage architecture. We discuss this fact with respect to our example based on TPC-H. Afterwards, we present our research steps to develop a decision model for optimal selection of storage architecture. Finally, we motivate and discuss the development of a hybrid architecture.

1 Introduction

Database management systems (DBMSs) are pervasive in current applications. With database tuning, practitioners aim at optimal performance which is a primary objective for database applications. The administration and optimization of DBMSs is costly. Researchers and DBMS vendors are developing *self-tuning* techniques such that DBMSs continuously improve themselves [1,2]. Chaudhuri et al. show in their summary of ten years of self-tuning [3] that almost all techniques have been investigated for *row-oriented* DBMSs (*row stores*).

In recent years, different approaches have come up to fulfil new requirements for applications, e.g., *column-oriented* DBMSs *column stores* [4-6]. Abadi et al. define "Column-stores, in a nutshell, store each database table column separately, with attribute values belonging to the same column stored contiguously, compressed, and densely packed, as opposed to traditional database systems that store entire records (rows) one after the other" [7]. However, column stores have been neglected so far for self-tuning techniques. In this paper, we show that self-tuning techniques are beneficial for column stores as well.

We motivate our research with a small example and discuss the results according to our goals. Afterwards, we discuss the differences among the two storage architectures and show first ideas to develop a decision model for selection of an optimal storage

architecture. Therefore, such a decision model should map different application fields to the best supporting storage architectures. Such a decision model is applicable to all relational DBMSs but our following descriptions are mainly based on the data warehousing (DWH) domain. To the best of our knowledge, there is no decision model to advise one of both architectures for a given application domain. Based on our insights, we argue that there is a high benefit for combining both architectures to a hybrid system. Such a system could use the advantages of both approaches in certain parts of application. We finally note how self-tuning techniques for row stores can be used in hybrid systems and how they can be adapted for column stores parts within the system.

2 Motivating Example

In this section, we present the results of a motivating example. In our example, we decide to use Infobright ICE[1] 3.2.2 and MySQL[2] 5.1.37. MySQL represents the row stores and ICE represents the column stores. At first view, the two DBMSs are as similar as these two because both systems are based on the MySQL kernel/management services except that they use different storage architectures. Due to their affinity, we selected these two DBMSs to compare the architectures instead of different functionality. Afterwards, we discuss the results of our example.

2.1 The TPC-H Benchmark

Our test environment is an Ubuntu 9.10 64bit system running on a Samsung X65 with a 2.2GHz dual core processor, 2GB RAM and 2GB swap partition. Furthermore, both DBMS configurations are adjusted to MySQL standard configuration and we use the standardized benchmark TPC-H (2.8.0) with scale factor 1 (1GB), i.e., both DBMSs use considerably less than 1GB of RAM (e.g., key buffer 16MB). No indexes or views are create to our environment expect indexes and views that are caused by DDL (primary key) or the workload itself. We run a series of tests with the TPC-H benchmark that represents DWH applications on our machine.

Our results are presented in Table 1. The query execution times are in the format hours, minutes and seconds (hh:mm:ss). We note that we are not able to run TPC-H Query 13 with MySQL syntax. We aborted Query 18 running on MySQL after more than 21 hours. All executions of Query 18 (MySQL) led to the same result. Some queries perform similarly on both architectures like Query 10 and Query 12. In contrast, Query 4 performs 20 times faster on MySQL (row store) than ICE (column store) as well as ICE executes some queries much faster than MySQL, e.g., Query 9 and Query 19. Our results show that neither of both architectures can outperform the other for every query.

Due to the significantly different results, we have to analyse several queries in detail, e.g., Query 21 for ICE or Query 18 for MySQL. If we exclude Query 18 and 21, we receive more expressive values, i.e., 58 minutes 33 seconds for ICE and 13 minutes 06 seconds for MySQL. ICE does not outperform MySQL as one could expect.

[1] http://www.infobright.org
[2] http://www.mysql.org

Threats to Validity. In this study, we made some assumptions. We have chosen MySQL and ICE as representatives because both systems are based on MySQL management services. There are many other column and row stores which we do not compare in this study, i.e., our study does not make claims of being complete.

2.2 Discussion

One can argue that our results arise from different functionality and query optimization techniques. We mention, ICE uses query optimization techniques which MySQL does not support and ICE optimizer poorly supports nested queries at the moment. We assume that this behaviour is intensified by necessary tuple operations for certain queries, e.g., Query 21 or 22. ICE has to process the entire tuples for the (not) exist clauses in Query 21 and 22. Additionally, ICE is already designed as read-only DWH system whereas MySQL is designed for transactional processing (OLTP). Nevertheless, we note that this functional gap appears to each system comparison concerning different architectures.

In our example, ICE outperforms MySQL on queries that contain a high number of aggregation and grouping operations, e.g., Query 1 and Query 3. The tuples processed for these queries contain only few columns in the (intermediate) result sets, i.e., a small number of tuple reconstructions is necessary. For (intermediate) result sets with a higher number of columns, ICE loses its advantages, e.g., Query 10 or in the worst case Query 21/22 because a high number of tuple reconstructions increase significantly the costs of query execution.

Table 1.

TPC-H results for MySQL and ICE

Query no.	ICE	MySQL	Query no.	ICE	MySQL
1	00:00:25	00:00:54	12	00:00:03	00:00:04
2	00:00:44	00:02:42	13	-	-
3	00:00:02	00:00:46	14	00:00:01	00:00:37
4	00:02:33	00:00:07	15	00:00:04	00:00:17
5	00:00:03	00:01:28	16	00:00:00	00:00:10
6	00:00:00	00:00:04	17	00:24:12	00:00:01
7	00:00:03	00:00:31	18	00:00:08	> 21:07:44
8	00:00:02	00:00:04	19	00:00:03	00:02:31
9	00:00:06	00:01:11	20	00:10:51	00:02:24
10	00:00:08	00:00:11	21	06:00:27	00:02:48
11	00:00:00	00:00:01	22	00:19:13	00:00:03
carryover	00:04:06	00:07:59	overall	06:59:08	> 21:24:38

Our example shows the need for new approaches in the DWH domain considering the new requirements [5,6,8-11] in this domain. There are queries which perform better on the row store and queries which perform better on column store architecture. The results verify our assumption that different workload parts perform better on one of the architectures. Consequently, different applications and their needs have to be evaluated to use the optimal architecture. The optimal architecture can only be figured out if we have a decision support for architecture selection. Therefore, we recommend the development of a general decision model for row- and column-oriented architectures. We assume that a prototype supporting both architectures independently leads to a more general decision model.

According to the architecture, a decision model will support design and redesign of databases, e.g., DWH, and improves the quality of design decisions.

2.3 A hybrid storage architecture

We argue that different applications, in our case the DWH domain, have common needs and we can decide between both architectures based on a decision model. However, we will have performance losses caused by the architecture because there

are probably also suitable patterns for the converse architecture. Moreover, the storage architecture of a system has to be changed if the workload changes extremely. This involves a complete architectural redesign and is the worst case on operational systems. So, the decision model cannot only be utilized to support design decisions but also to support development of new architectures, e.g., a hybrid architecture.

We see two strong intercessory aspects to develop a hybrid architecture and/or prototype. First, a hybrid system should be more suitable for applications which have mixed workload patterns, thus both storage architectures are required. The performance losses can be reduced compared with the other two architectures by a mixed workload scenario (tuple operations and aggregations). Second, the redesign of a system with one of both architectures can be prevented by a hybrid architecture because a hybrid architecture contains both. The hybrid architecture should reduce the performance losses caused by changing workload in contrast to the other both architectures. Additionally, it is conceivable that a hybrid architecture will be able to adapt constantly to a changing workload with some extensions. The costs of dynamical adaption have to be limited by thresholds in a cost function. These thresholds prevent the growth of the adaption costs beyond the benefit of adaption. Hence, self-tuning in an architectural manner will be possible.

We have to consider several aspects to develop a hybrid system, e.g., compression techniques, query processing. Some aspects are mutually exclusive to each other like different query processing techniques. Nevertheless, some conflicts can be solved by adapting current techniques. These aspects, e.g., cooperative query processing, can be integrated into a hybrid system and into the decision model in further steps. A decision model for the selection of the optimal architecture has to be extended and provides also the basis for self-tuning in hybrid system. Thereby, analyses for the storage architecture selection have to be broken down from whole workloads to parts of it (decomposition), e.g., storage architecture will be advised for each relation.

3 Development Steps for a Decision Model

The database community has mainly considered that row stores can solve all data management tasks [12,13] in the past. Other storage architectures lead to niche solutions [14-16]. Thereby, column stores are most suitable for DWH and business intelligence [7]. Some TPC-H benchmark results emphasize this estimation but one cannot exactly say which application or workload advances the usage of a column store (cf. Section 2). To the best of our knowledge, there is no study or model which determines the use of column stores for a DWH. In this section, we present the first ideas to develop a decision model for storage architectures selection. Afterwards, we discuss the main influence factors for a decision model according to storage architectures[3] and how we can extend this model.

3.1 Workload Patterns

We have examined the general assumption that the typical column store application is DWH. Aggregation and grouping are typically used in the DWH domain but

[3] In the remaining work, we use column and row store as synonyms for the corresponding database storage architectures.

computations based on tuples occur in these workloads, too. As we already mentioned column stores perform worse on tuple operations, thus column stores are not suitable for each application in the DWH domain. Our considerations should help to overcome the main problem in this domain according to the selection of storage architectures. We assume, row and column stores have their own application fields within the DWH domain especially with respect to the new requirements [5,6,8-11]. Hence, we have to analyse workloads and derive workload patterns to estimate or select the most suitable architecture. Additionally, we need to analyse computation types of DBMS operations for the DWH domain.

Based on our analyses, any workload is composed out of three workload patterns that crucially influence the performance on both architectures. First, aggregation and grouping form one pattern because in an abstract manner, the computations proceed the same way. The second pattern covers join operations as essential part of nearly every DBMS. Third, tuple operations/reconstructions are composed in a separate pattern.

Aggregation and grouping are very important in the DWH domain, e.g., cube building or on-the-fly analysis. The aggregation and grouping pattern has to be analysed for row and column stores even if the operations themselves are the same because they are processed in different ways. We need to know how different both architectures perform on this pattern. The comparison of two systems provides system-specific results which we have to generalise. This pattern points out the strengths of columns stores.

The behaviour of join pattern is similar to the aggregation and grouping pattern. There are several join techniques in DBMSs, e.g., merge join, hash join, or nested loop join, which are common in both architectures. Hence, we do not analyse different join techniques themselves but the different computation types in both architectures, e.g., vector and non-vector based. Join operations crucially influence the size of intermediate results, e.g., selectivity join over primary key values or full table join that imply different amount of tuple reconstructions in column stores. Thus, the performance can be changed critically.

The number of tuple operations and/or reconstructions is a critical factor. In contrast to row stores, each tuple operations/reconstruction causes computational costs in column stores because column stores recompose tuples with all necessary columns. Row stores access tuples directly and cause only costs directly from the access path, i.e., this pattern advantages row stores and also represents several new mentioned requirements.

Typically, insert, delete, and update operations are not considered in the DWH domain, so we do not discuss these operations for now. For the new requirements in the DWH domain our patterns should be extended in this direction at a future date. Equally, the projection will not covered in a separate pattern because each system uses projections on intermediate and final results.

3.2 The Decision Model

We mentioned, different workload patterns are more suitable for either row or column store architecture (cf. Section 3.1). None of both architectures, row- and column-oriented, support the whole DWH domain optimally, i.e., we have to assign different workload patterns to one of the two storage architectures. Hence, we have to identify the influence of different workload patterns on the architectures and the performance of these patterns on a certain architecture. The workload patterns are used in our decision

model that gives an advice for the optimal storage architecture. The input parameters are a given workload, statistics that we map into our workload patterns, and derived or user-defined heuristics/rules. The rough functionality is covered below and given in Fig. 1.

A given workload will be analysed and parts of the decomposed workload are mapped to the described patterns. As we mentioned before, the different patterns advantage a certain architecture. We compute the costs of each pattern for the respective architecture and aggregate these costs for instance. In this case, the most qualified architecture for a given workload causes less total costs. Hence, we can recommend a certain architecture for a given workload. Therefore, we can derive heuristics and rules from our workload patterns which contain the workload statistics. The heuristics and rules can be iteratively refined. The different optimization techniques and/or rules of a system have to be hidden for a general model because we want to evaluate the architectures and not the different (query) optimization techniques. This reduces the initial complexity the decision model development.

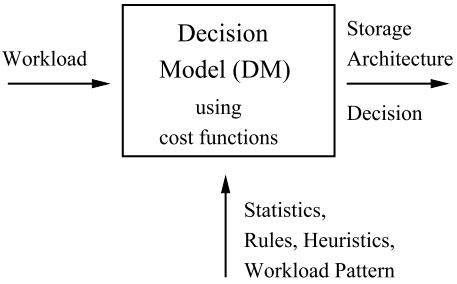

Fig.1. Scheme of our Decision Model

In detail, we analyse the workload based on used operations, e.g., we count the frequency of operations and the participated tuples and columns for each operation. The computation cost for each operation, e.g., average calculation operation (aggregation/ grouping pattern); will be estimated by statistics of corresponding DBMS or empirical determined factors. We state, these factors weight costs of column and row stores for a certain operation to each other. In other words, a heuristic could state that column stores cause costs by factor X per tuple for a certain column and row stores cause costs by factor Y per tuple for a certain column in our average calculation example. This procedure will be repeated for each operation belonging to a pattern. From the heuristics, we can derive decision rules. For the total costs calculation, it is not decisively if the costs are completely computed or some costs are estimated by heuristics or rules. The usage of heuristics or rules will only reduce the computation complexity, i.e., these decision rules replace cost computations in our decision model for pre-defined events.

Due to a fine-granular approach, the complexity of storage architecture decision based on statistics is very high. As we mentioned, we can use rules to reduce the complexity, e.g., column stores perform always better on average calculation for a column than row stores. To derive correctly heuristics and rules, we need to use real statistics from operating DBMSs. Thus, we have to accept the computation costs and start with the fine granular approach to constitute facts for optimization of decision process. Afterwards, we are able to reduce the complexity by our weighted heuristics/rules, e.g., we expect that a certain part of the workload belongs to average calculation operations that perform better on column store by factor X. The complexity will crucially lowered by each heuristic for a certain operation. As a consequence of coarse granularity, we only have to compute the costs for operations where the performance approximation is not decidable by heuristics and rules.

In the design process, we cannot build on given workloads and have to use an estimated workload behaviour. Again, we use weighting factors to estimate the performance of workload parts. Our coarse decision model approach using heuristics and rules will satisfy these demands. Therefore, we use weighting factors which we obtained by complexity reduction process of fine-granular approach. In the most extreme case, the decision rules stored in the decision model can describe entire patterns and replace the complete cost calculation after workload decomposition.

3.3 Development Strategies

In this section, we will present development strategies for a decision model to select the optimal storage architecture. We assume at least two development strategies.

First, we develop a DBMS which supports independently both architectures to obtain and evaluate heuristics and rules for the model, i.e., we can switch the storage engines within the DBMS. Thus, we can hide the optimization techniques for decision model development because both storage engines use the same DBMS functionalities. Finally, we use a set of cost functions to estimate the performance differences between the architectures in our decision model. With the assistance of such a decision model, we can select an architecture in the design process or if necessary the redesign for a certain application.

Second, we develop a decision model that does not hide system-specific optimization techniques. This strategy compares two systems, i.e., a column and a row store, and weights the different workload patterns for both systems. Hence, we can develop an architectural decision model for the two systems. Such a model can give advice which architecture we should use for a given workload. However, this strategy implies new parameter sets for the decision model concerning each pairs of DBMSs.

We argue, one system that supports both architectures is needed to obtain a general model. Hence, our model based on heuristics/rules can be proven and refined using such a system, i.e., we can decide which architecture is more suitable for a given workload.

Independently from the development strategy, the general functionality of the decision model is equally for design tasks (weighted comparison) and self-tuning (hybrid system). The set of rules derived from the workload patterns and heuristics are integrated as cost functions into the decision model. Finally, the given workload will be processed by the decision model based on cost functions.

3.4 Interactions with other DBMS parts

Many aspects influence the storage model decision. We introduce the estimated main factors of influence to support an open and objective discussion and introduce them into the model in later stages. We assume at least four main interactions: the SQL optimizer, the query processor, the query processing, and compression techniques.

First, the SQL optimizer is integrated into nearly every DBMS. The functionality, optimization rules, and implementation techniques vary from system to system and make a comparable evaluation very difficult. At the same time, no current DBMS supports both, column-oriented and row-oriented, architectures. Even if a DBMS supports both architectures, it is still a vague expectation that its SQL optimizer does not affect the query execution in different ways. By hiding the optimization step, we can guarantee that the SQL optimizer does not perform better for one of both storage architectures.

Second, the query processor of a system is an important aspect for the performance of a system. Thereby, the quality of the query processor depends on the functionality, quality of implementation etc. Even worse, column stores can use a row-oriented or a column-oriented query processor. In case of column stores, query optimization depends on the query processor and point of time for tuple reconstruction.

Third, the query processing (and optimization) can be based on tuples like all row stores and most column stores or are based on columns like C-Store/Vertica [5]. By using a column-oriented query processor, advantages of compression techniques can be utilized more efficiently because columns can be processed for a query without decompression [5].

The last aspect regarding different compression techniques is directly linked to query processing on compressed data. Several compression techniques are used in column stores in miscellaneous variants, e.g., dependently on data type. Thereby, it is nearly impossible to observe every aspect of compression across different systems. Some systems use a huge set of compression techniques (nearly for each data type) and other use compression only for specific columns. Adding the compression techniques of row stores will corrupt these observations because row stores process compression tuple-wise or partition-wise with different data types (mostly based on relations).

To hide the mentioned interactions, we propose the extension of a row store, e.g., BerkeleyDB [17] or FAME DBMS [18], with column-oriented architecture. This strategy promises the best results according to the model development. Hence, we guarantee the same environment for both architectures and develop a general model for them.

3.5 Perspective

Nevertheless, we have to recover the interactions and optimization techniques which we had hidden (cf. Section 3.4). These interactions have to be integrated in a general manner into our decision model because we also consider distinct systems in our general decision model besides a prototype. In this way, we can develop a general decision model for column and row stores. But as already mentioned, neither of both architectures can perform efficiently on each workload. We can only overcome this gap between column and row stores with aid of a hybrid architecture and develop a hybrid system based on this architecture.

To develop a hybrid system or architecture, we do not need to integrate the interactions of DBMS parts. On the contrary, a hybrid system should utilize and consider architectural strengths and weaknesses before considering optimization techniques. We argue, these optimization techniques have to be adapted for a hybrid system anyway, thus we do not need to consider them. The reimplementation or adaption of current optimization techniques is also a chance to overcome existing barriers, e.g., the static storage allocation [19], OLTP and/or real-time DWH [6] because they strongly depend on the current architectures. The reached degree of freedom involves administrative overhead which has to be lower than the benefit guaranteed by cost functions.

In contrast to our proposed prototypical implementation for model development, we mention that it could be more advantageous to develop the hybrid architecture based on a column store (or complete reimplementation). We reason this approach by functional gaps between column and row stores, e.g., update-in-place strategies cannot

be applied in column stores or concurrency control in column stores is very different (mostly delta based).

Update operations are another new challenge in the DWH domain and new DWH applications. Updates can critically affect the performance of a DBMS. We note that updates can be implemented by delete and insert operations. So, we determine all three operations as update behaviour. Formerly, the database community estimated, update processing does not influence DWH applications because the DWH works on time-invariant data and executes the updates in an ETL process. Such process requires a time of minor workload or offline time of the DWH but most DWH should be available 24 hours a day. However, new approaches [9,20,21] show that updates processing will be crucially increased for DWH. Updates are an architectural problem in column stores due to tuple partitioning in the column-oriented architecture. To support the tuple reconstruction after partitioning, each column value has to refer to unique tuple identifier or have to keep the columns in physical sequential order especially for sorted columns. Hence, several column store implementations try to reduce drawbacks by updates, e.g., C-Store/Vertica [5]. However, we want to evaluate a decision model for architectural decision between column and row store for a certain application and not specific implementations even if there are already good approaches available.

Finally, a hybrid architecture can help to overcome the update problem of column stores and could open new application fields because it contains both architectures that can be adapted to workloads, e.g., storage architecture will be advised for each relation. A hybrid architecture that utilizes both architecture enables self-tuning techniques in an architectural manner, i.e., the storage architecture of a certain part of the database can be changed and adapted to changing workloads. Additionally, the adaptation of self-tuning techniques from row stores are possible and will increase the overall performance, e.g., self-adapting compressions (techniques) for shifting data distribution or materialized views that store the tuple reconstruction results for frequently accessed partitioned tuples etc. These self-tuning approaches can reduce I/O costs for data access even for changing data sets or reduce tuple reconstruction costs within query execution. These self-tuning techniques can certainly be utilized in column stores, too.

4 Related Work

In recent years, several open- and closed-source column stores have been released [4,5,15,22]. There are well-known column stores but all systems are pure column stores and do not support any row store functionality. Abadi et al. [23] compare row and column stores performance on the star schema benchmark. They simulate the column store architecture by indexing every single column or vertical partitioning of the schema. They show that using column store architecture in a row store is possible but the performance in a row store is poor. In this paper, we do not directly compare optimization techniques of DBMSs. Instead, we consider strengths and weakness of both architectures to encourage our work of combining them in a hybrid system.

Regarding the solutions for architectural problems, there are systems available which are not a hybrid system in our sense (between a column and a row store). Nevertheless, they contain very interesting approaches for development of a hybrid system. C-Store [5]

uses two different storages to overcome the update problems of column stores. A related approach brings together a column store approach and the typical row store domain of online transactional processing (OLTP) data [9]. In contrast to our work, they focus on near real-time processing of OLTP data in a DWH and the necessary ETL components. They hold replicates of all OLTP data which is needed for reporting mechanism (OLAP/DWH). These approaches are not a hybrid system in our sense because we want to combine two architectures to a hybrid storage without replication.

Another interesting approach is Ingres/Vectorwise which applies the Vectorwise (formerly MonetDB/X100) architecture into the Ingres product family [24]. In cooperation with Vectorwise, Ingres is developing a new storage manager ColumnBM for the new Ingres/Vectorwise. From [24] and product documentations, the integration of the new architecture into the existing environment remains unclear.

BigTable [16] or HadoopDB [25] bring together different architectures or competing approaches. However, these systems are developed for quite different application fields (niche solutions) and do not support the relational data model. Thereby, the development of a hybrid system is supported by our decision model which is our main goal in our work. We compare the architectures to obtain a general model and not certain DBMSs which is nearly impossible by the amount of DBMSs in the community. Moreover, a decision model and its resulting knowledge are the basis to develop a hybrid architecture. This strategy is the main distinction of our work to the other approaches.

To develop our decision model, we need to analyse any workloads and derive workload patterns therefrom. Therefore, we can utilize, adapt and extend existing approaches such as Turbyfill's approach [26] that considers mixed database workloads concerning disk access. We can map this approach to the strengths and weaknesses of different architectures. In contrast, the approach by Raatikainen [27] considers workload classes and how to find them based on cluster analysis. We do not want to cluster workloads into a certain number of classes but we want to classify operations. We can use these and other approaches to find and evaluate our workload patterns. The approaches of Favre et al. [28] and Holze et al. [29] step into the direction of self-tuning databases. Favre et al. consider the evolutionary changes within a DWH workload. They also consider the interactions between schema evolution and workload evolution. This approach is very interesting according to our proposed hybrid system. The related approach of Holze et al. concerns clustering of workloads based on distance functions but is developed for lightweight and stream-based operations. Thereby, the development of a hybrid system is supported by the workload pattern development.

The current research reflects new approaches to solve the update problems of OLAP applications, e.g., dimension updates [10]. Moreover, the update problem is increased according to new demands like real-time DWH [8,11]. These new demands call for new decision models in the DWH domain and databases at all.

5 Conclusion

In this paper, we report on a study that we performed on column store and row store systems. In this study, we observed the necessity of a decision model for architectural design in the data warehousing domain. To map a workload to an architecture (column or

row store), we identified three workload patterns which will be refined into fine-grained sub-patterns. The patterns have to be evaluated concerning their performance on different architectures, i.e., the recommended architecture depends on workload or found pattern. Based on these patterns, we presented our ideas to develop our decision model. In future work, we want to advise the architectural (re-) design decisions of DWH systems. We assume that the development of a decision model for the choice of storage architecture (row or column store) is the precondition for new self-tuning techniques in hybrid systems. Furthermore, the generality of a decision model should be verified in a number of case studies. The interactions of DBMS parts have to be discussed afterwards, e.g., SQL optimizer or query processor.

Moreover, we conclude the development of a hybrid architecture to overcome the gap between column and row stores as consequence of our study. This step is an explicit conclusion of our considerations concerning a decision model because we can map the workload pattern to the strengths of either architecture. So, our model will be the base for a hybrid system and we can utilize it for column, row, and hybrid architectures. We argue that our model will foster the development of a hybrid architecture by architectural design decision support and rules/heuristics.

A hybrid system will enable self-tuning approaches in an architectural manner. The monitored workloads will be analysed with the aid of our decision model. The self-tuning approach enables continuous adaption of architectures in small steps for a hybrid system instead of periodical (re-) design of complete systems. The adaptive architecture of a hybrid system will save the costly redesigns of a system and will endorse changing workloads.

In the future, we will determine the necessary workload patterns using a prototypical implementation and evaluate their performance on the different architectures. These patterns will be integrated in our model and refined by evaluation on different existing environments/systems.

Acknowledgments. We thank Christian Kästner, Martin Kuhlemann, Ingolf Geist, and Veit Köppen for helpful discussions and comments on earlier drafts of this paper.

References

1. Weikum, G., and Hasse, C., Moenkeberg, A., Zabback, P.: The COMFORT automatic tuning project, invited project review. Information Systems **19**(5) (1994) 381-432
2. IBM: An architectural blueprint for autonomic computing. White Paper (June 2006) Fourth Edition, IBM Corporation.
3. Chaudhuri, S., Narasayya, V.: Self-tuning database systems: A decade of progress. In: VLDB '07, VLDB Endowment (2007) 3-14
4. Zukowski, M., and Boncz, P.A., Nes, N., Heman, S.: MonetDB/X100 - A DBMS in the CPU cache. IEEE Data Engineering Bulletin **28**(2) (June 2005) 17-22
5. Abadi, D.J.: Query execution in column-oriented database systems. PhD thesis, Cambridge, MA, USA (2008) Adviser: Madden, Samuel.
6. Plattner, H.: A common database approach for OLTP and OLAP using an in-memory column database. In: SIGMOD '09, New York, NY, USA, ACM (2009) 1-2
7. Abadi, D.J., Boncz, P.A., Harizopoulos, S.: Column oriented database systems. PVLDB '09 **2**(2) (2009) 1664-1665
8. Santos, R.J., Bernardino, J.: Real-time data warehouse loading methodology. In: IDEAS '08, New York, NY, USA, ACM (2009) 49-58

9. Schaffner, J., Bog, A., Krüger, J., Zeier, A.: A hybrid row-column OLTP database architecture for operational reporting. In: BIRTE '08. (2008)
10. Vaisman, A.A., Mendelzon, A.O., Ruaro, W., Cymerman, S.,G.: Supporting dimension updates in an OLAP server. Information Systems **29**(2) (2004) 165-185
11. Zhu, Y., An, L., Liu, S.: Data updating and query in real-time data warehouse system. In: CSSE '08, Washington, DC, USA, IEEE Computer Society (2008) 1295-1297
12. Stonebraker, M., Çetintemel, U.: One size fits all: An idea whose time has come and gone. In: ICDE. (2005) 2-11
13. Stonebraker, M., Bear, C., Çetintemel, U., Cherniack, M., Ge, T., Hachem, N., Harizopoulos, S., Lifter,J., Rogers, J., Zdonik, S.B.: One size fits all? Part 2: Benchmarking Studies. In: CIDR. (2007) 173-184
14. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. Communications of the ACM **51**(1) (2008) 107-113
15. Legler, T., Lehner, W., Ross, A.: Data mining with the SAP NetWeaver BI Accelerator. In: VLDB '06, VLDB Endowment (2006) 1059-1068
16. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: A distributed storage system for structured data. ACM Transactions on Computer Systems **26**(2) (2008) 1-26
17. Yadava, H.: The Berkeley DB Book. Apress, Berkely, CA, USA (2007)
18. Rosenmüller, M., Siegmund, N., Schirmeier, H., Sincero, J., Apel, S., Leich, T., Spinczyk, O., Saake, G.: FAME-DBMS: Tailor-made data management solutions for embedded systems. In: SETMDM '08, New York, NY, USA, ACM (2008) 1-6
19. Lübcke, A.: Self-tuning of data allocation and storage management: Advantages and implications. In: GvD '09: Proceedings of the 21. GI-Workshop on Foundations of Databases. (2009) 21-25
20. Langseth, J.: Real-time data warehousing: Challenges and solutions. White Paper (February 2004) Claraview (Teradata Corporation).
21. Vandermay, J.: Considerations for building a real-time data warehouse. White Paper (November 2001) DataMirror Corporation.
22. Ślęzak, D., Wróblewski, J., Eastwood, V., Synak, P.: Brighthouse: an analytic data warehouse for ad-hoc queries. PVLDB '08 **1**(2) (2008) 1337-1345
23. Abadi, D., Madden, S.R., Hachem, N.: Column-stores vs. row-stores: how different are they really? In: SIGMOD '08, New York, NY, USA, ACM (2008) 967-980
24. Ingres/Vectorwise: Ingres/VectorWise sneak preview on the Intel Xeon Processor 5500 series-based platform. White Paper (September 2009)
25. Abouzeid, A., Bajda-Pawlikowski, K., Abadi, D.J., Rasin. A., Silberschatz, A.: HadoopDB: An architectural hybrid of MapReduce and DBMS technologies for analytical workloads. PVLDB '09 **2**(1) (2009) 922-933
26. Turbyfill, C.: Disk performance and access patterns for mixed database workloads. IEEE Data Engineering Bulletin **11**(1) (1988) 48-54
27. Raatikainen, K.E.E.: Cluster analysis and workload classification. SIGMETRICS Performance Evaluation Review **20**(4) (1993) 24-30
28. Favre, C., Bentayeb, F., Boussaid, O.: Evolution of data warehouses' optimization: A workload perspective. In: DaWaK '07. (2007) 13-22
29. Holze, M., Gaidies, C., Ritter, N.: Consistent on-line classification of DBS workload events. In: CIKM '09. (2009) 1641-1644

SCIENTIFIC PAPERS, UNIVERSITY OF LATVIA, 2010. Vol. 757
COMPUTER SCIENCE AND INFORMATION TECHNOLOGIES 77–83 P.

Towards Formalization of a Method for Data Consistency Support in Mobile Ad-hoc Distributed Systems[1]

Maxim Galkin
University of Saint-Petersburg, Universitetsky pr. 28, Peterhof, Saint-Petersburg, Russia
maksim.galkin@gmail.com

Abstract. This paper presents a research-in-progress report on a method for data consistency support in ad-hoc mobile distributed systems. The method is based on a concept of high-level operations "compatibility" and operation history reconciliation with compensations. The described approach is not bound to any particular data storage or communication technology, but in this paper we use tuple spaces as a possible practical medium, which hides network details and mobile nodes heterogeneity. Limits of applicability of the method are described, model and operations are formalized, detailed reconciliation algorithm and communication protocol are discussed.

Keywords: data consistency, ad hoc distributed systems, mobile systems.

1 Introduction

In this paper we present a new method for data consistency support in an ad-hoc mobile distributed system. This method combines some of the optimistic practices known for traditional and nomadic distributed systems (like high-level operations "compatibility" and compensational operations) with special structures such as list-accumulators, which abstract the idea of an operation history [10] (see Section 3). The proposed method specifically deals with reconciliation conflicts separating them into a compensational accumulator, which is only required for external operations, but not necessary for merging. In this paper we also propose using loose communication mechanisms such as tuple spaces, which can hide the intermittent mobile network connection and mobile nodes heterogeneity.

Many researchers contributed to the development and optimization of consistency support methods for traditional client-server and nomadic distributed systems [1, 2, 10, 15]. Most of such methods though rely on some sort of centralized "clusters of consistency" [13], or "master replica", that is a node or a set of nodes, which keeps the latest and most actual data. Nomadic nodes synchronize their state with those

[1] This work was partially supported by Russian Foundation for Basic Research RFBR, grant 10-07-00156.

central nodes and achieve durability for the accepted transactions. Such approach is not applicable in the ad-hoc mobile scenario [3, 9]. Other works that propose weak consistency protocols for distributed systems [6, 8] do not describe any specific conflict resolution protocols, assuming that the conflicts are "unlikely to happen" or giving them only a short remark, though conflict resolution is usually the most complex part of an optimistic system[16]. Instead of avoiding conflicts through locks or quorum protocols (impossible in a disconnected scenario) or ignoring them we have to embrace consistency conflicts as a part of the usual working scenario and give away the durability property. We propose a specific mechanism for handling consistency conflicts with compensational accumulator, which is discussed in details in Section 3. This mechanism supports automatic semantic conflict resolution as well as delayed manual.

On the other hand, one of the strong advantages of ad-hoc distributed systems is in the absence of a "single point of failure". All of the network nodes are equally important, and if we can keep their replicas consistent then our system can continue working even if it loses part of the nodes. Another advantage is the potentially unlimited extensibility (i.e. the ability to add and accommodate new network nodes) as there is no bottleneck like main server performance or main server connection bandwidth. Every node is only going to "talk" to its neighbors. Finally, in a mobile world it's a possible scenario that the known main servers can be inaccessible, while the neighbour devices can be still available. Ideally such availability can be exploited to support the same level of services the node had while it was connected to the static infrastructure. In practice our method can only guarantee limited level of services, as they must comply with the conditions we describe in Section 2 and should not have many conflicting operations. The exact practical limitations are to be verified in the experimental part of my Ph.D. thesis, which is yet in progress.

The proposed method is indifferent to an underlying data model, only operations are important. It demands all nodes to be aware of:

- some common initial data state S_0, which can be just an empty set, or a "map" over which the nodes work;
- a set of high-level operations O=M+C, where M are model operations recorded by nodes and C are compensational operations;
- a compensational function δ: <M> $\rightarrow$ <C>, which maps model's list-accumulator to a compensational list-accumulator;
- a reconciliation operation R.

This is discussed in detail in Section 3. Having that defined the nodes become autonomous and start applying operations from M to their local replicas. At some random (unpredictable) points in time nodes perform binary reconciliation operation R of their model accumulators and through this achieve a consistent and actual data.

$$<M> \text{ `R' } <M> \rightarrow <M> . \quad (1)$$

Due to the general approach this method can be used to build a middleware platform for consistency support in ad-hoc distributed systems.

2 Method Applicability

In this section we discuss two conditions of the proposed method applicability.

First of all, let's clarify what we mean by "mobility" of the network nodes. Basing on the classification given in [11] we assume our nodes: have intermittent network connection; have dynamic location / dynamic context; need to discover other hosts in an ad-hoc manner; are heterogeneous. We abstract from nodes heterogeneity and context change by using loose generic tuple spaces communication mechanism. Also we do not consider our nodes to have any significant limitations in CPU / power resources for the sake of this research, due to the rapid development of mobile devices and because we not only take PDAs and communicators into account, but laptops as well.

The second applicability condition concerns the reconciliation operation 'R' properties. This operation is naturally commutative, because of the system symmetry: all nodes histories are equally important and there is no point in making the reconciliation to prefer one argument over another. What we have to additionally demand is the associativity property for R. Without that property the result of the reconciliation will depend on the particular order of nodes reconciliations thus, as we cannot guarantee any particular order, making the result non-deterministic. For some certain sets of operations and compensations it might be possible to bound this non-determinism, so that, for example, any possible order of R's would return a "consistent enough" result, but this topic is out of the scope of this research. So, we demand R to be commutative and associative.

3 Data Model and Operations

For the sake of the proposed method, a set of defined high-level operations over the data is more significant than the data model itself and further in this paper we will only discuss operation set properties. At the same time we must note that in some related works [10, 14] special data model was crucial in defining such operations, which cause less conflicts and therefore better suitable for collaborative work.

In general, we can name two sides or two states of the data in the system. One side is the real data, which resides in the mobile agents, and which is continuously updated by them and another side is the ideal data, that can be achieved by a full semantically-correct reconciliation of all data replicas. In the process of work our cloud of local data replicas strives to become closer to the ideal state, where every agent knows the current and actual state of data. Of course, in reality the degree of consistency depends on many factors, like the connection quality between agents and the intensity of the continuous local updates. We plan to examine the ratio of these factors in the experimental part of the work.

To formalize that we will use a variation of the version vector approach [16]. Assuming we have N nodes and each node has a version number, which is incremented on every update this node commits, the state of a node i can be expressed as a vector Vi, where Vi[j] is the last version of node j known to node i. Then an ideal fully consistent state would have every Vi[j] = Vj[j], if by Vj[j] we mean the global latest version of node j. Every state different from that would have some node divergence and require

further reconciliation. We must note here that the actual node data is fully determined by its version vector due to the operation 'R' properties, discussed in Section 2.

In most cases, a table of compatibility conditions is filled together with operations definition. In the CoACT model [15] such conditions are called "activity interleaving rules" and given in a form of predicates for each pair of operations. In general, operations compatibility may depend on both their nature (e.g., two "read" operations will never create a conflict) and data they are applied to.

To successfully maintain consistency the set of operations for ad-hoc mobile distributed system must ensure as few conflicts as possible, in other words operations must be as compatible as possible, because every conflict in such a system is a serious problem. Two mobile nodes, which discover a data inconsistency during their interaction don't possess any advantage over each other, unlike it happens in "nomadic" distributed systems, where the fixed nodes have the priority, the most consistent state. Usually in ad-hoc distributed system it's also impossible to draw a human operator into the process of conflict resolution, because the inconsistent state can be discovered at nodes that aren't the authors of the conflicting operations or when the authors are offline [16].

Therefore every conflict resolution must be automatic in accordance with some pre-defined rules of reconciliation, as the system can't be blocked by a conflict. Such rules can be of two major types: differential (e.g. when the system takes one of two conflicting operations basing on its timestamp) and integral (if the system, for example, has some pre-defined objective function or cost function over the data model and it discards one of the conflicting operations basing on the value of the function). The system should also allow delayed manual conflict resolution as this is the only ultimate way to resolve possible operation conflicts.

In this paper we propose using two list-accumulators for each data object to meet the described requirements. Accumulators are intended for "intervention" of the application to the conflict resolution. As against simple types, the accumulator stores operations applied to some initial value of simple type instead of the explicit value of the element. Thus, accumulators allow on the air replaying the operations [10].

The first accumulator <M> is used to store the model operation history over a data element, and additional accumulator <C> is used to store the compensation operations over data elements. On reconciliation 'R' we merge only model accumulators to propagate the known updates throughout the system. After that, nodes use the compensational function δ to calculate the compensations <C> for the new model history <M>. Compensations may include simple cancellations of some operations from the main history, or those can be additional commands to semantically resolve a conflict in <M>. <C> is constructed in such a way, so it can be easily merged with <M> for external queries. But another important usage of <C> is that it can be examined separately by client applications and used to show the collected conflicts to a user for further reconciliation (the user, for example, might want to undo or redo one of the cancelled operations with the newer version of the object).

Formally, the reconciliation operation of nodes i and j can be described as maximization of each version vector component separately, so that

$$Vresult[k] = \max(Vi[k], Vj[k]), \text{ where } k = 1..\ N\ . \qquad (2)$$

4 Transactional Properties

The above-stated implies that even if some operation has been committed at one of the nodes and lived in the system for some period of time it is not guaranteed to be fixed in the system forever as one of the other nodes could have submitted a conflicting operation that will eventually overcome the first one. In other words, the Durability property from the ACID set is not guaranteed by our system for the sake of Consistency. Here we can also notice that in an ad-hoc distributed system no "global commit" or "strong" [13] operations are possible. On the contrary, all operations in the system are "weak", that is they are only applicable to the local replica of the data and they are only guaranteed to be "safe" until next data reconciliation with another node.

The Consistency property in our system should not to be confused with "single-node" data consistency. This type of consistency is always guaranteed in ACID systems, no transaction is allowed to leave the data in an inconsistent or partially consistent state as it would make the data meaningless. Our high-level operations can be seen analogous to such classical transactions as they are supposed to always leave our model semantically correct. So when we talk about the "degree of consistency" and node divergence we mean distributed system consistency. As our consistency support techniques cannot be based on locking in the mobile environment, they can only be optimistic, hence we have weak consistency type of system [6]. We can guarantee eventual type of weak consistency only under certain conditions, as we cannot always expect full convergence of the nodes' states, unless they participate actively in the communication. Another problem for the convergence is the possibility of distributed network partitioning, where some nodes from one network part will never be exposed to nodes from another. This limits the applicability of the proposed method in cases where eventual consistency is demanded.

With respect to the other ACID properties such as Atomicity and Isolation we should notice two levels of their applicability. Again if we consider our high-level operations as a specific form of transactions consisting of some elementary read/write sub-operations, then these transactions will satisfy both A and I properties. If on the other hand we introduce even higher-level transactions that consist of our original operations we will need to give away the highlighted properties, similar to how it happens in the "sagas" transaction model [5]. Otherwise, long-running isolated transactions will inevitably increase the number of conflicts just like it is observed in other types of distributed database systems [7]. In our system this can lead to increased load on mobile nodes and the overall decline in data quality, as compensative transactions can sometimes be ineffective in cleaning up the database due to lack of transaction isolation.

5 Nodes Interaction Protocol

We assume that numerous mobile nodes interact with each other by establishing connection with some of the other nodes in their "visibility range" and sharing a tuple space, for example, LIME [12] or JavaSpaces [4]. The tuple space has an ID, which is bound to the data model, so every node can actually "host" several models and reconcile them in different tuple spaces.

At one time there can be several nodes sharing one tuple space, but only one "editor". The editor is selected in some deterministic and reliable way, for example, by passing a token from one node to another. Tuple space contains the model achieved so far after the reconciliation of all the previous "editors" models. Current editor's task is to reconcile it with his own model, if necessary, update its own model with the shared model and pass the token to the next node. Every node can go offline any time, if the "editor" disconnects a new token must be generated.

As the implementation of the model reconciliation operation 'R' we propose simple timestamp-based ordering of the operations. It's easy to observe such 'R' is both commutative and associative.

After the reconciliation we need to calculate the compensational accumulator <C>, this can be done on each node separately or once per tuple space if one node is selected as compensation accumulator "editor". One obvious further enhancement of the protocol is, of course, in keeping the old compensations unchanged if there were no model changes in the compensated area.

In this scenario the common tuple space can be found to correspond to a notion of a "cluster of consistency" [13], with the only difference that in our case no data in a cluster is durable. For example, if two nodes from different clusters interact any of the operations in their replicas can be potentially conflicting and subject for removal.

6 Conclusion and Further Work

We have introduced a method for consistency support in a mobile ad-hoc distributed system, based on the reconciliation process of high-level operations histories. We also presented a description of supported data models and the requirements for operations set. We have analyzed the transactional properties of the proposed system and outlined a protocol for nodes interaction.

We plan to further develop this method with regards to an experimental proof of concept. In future theoretical work we plan to introduce metrics of the global consistency in such a mobile system based on the version vectors to be able to evaluate the efficiency of the method and/or compare our method with other methods. Another way of applying such metrics is to use them in the described mobile system as an integral rule for operational histories reconciliation, thus we can sort out nodes, which have their replica so obsolete or inconsistent that it doesn't make sense to compare them with the current data.

One more possible further research direction is to look at weaker reconciliation operations 'R', which do not require the associativity property.

7 References

1. Adya, A.: Weak Consistency: A Generalized Theory and Optimistic Implementations for Distributed Transactions. PhD thesis at the MIT, certified by Barbara Liskov (1999).
2. Bosneag, A-M., Brockmeyer, M.: A Formal Model for Eventual Consistency Semantics. In: Proc. of the Parallel and Distributed Computing and Systems Symposium (2002).
3. Capra, L., Emmerich, W., Mascolo, C.: Middleware for Mobile Computing: Awareness vs. Transparency. In: Int. 8th Workshop on Hot Topics in Operating Systems (2001).

4. Freeman, E., Hupfer, S., Arnold, K.: JavaSpaces, Principles, Patterns and Practice. Addison-Wesley (1999).
5. Garcia-Molina, H., Salem, K.: Sagas. ACM SIGMOD Record, Volume 16, Issue 3, pp. 249-259 (1987).
6. Golding, R.: A Weak-Consistency Architecture for Distributed Information Services. Computing Systems Journal, vol. 5 (1992).
7. Gray, J.N., Helland, P., O'Neil, P., Shasha, D.: The Dangers of Replication and a Solution. In Conference on Management of Data, pp. 173-182, Canada (1996).
8. Gustavsson S., Andler, S.F.: Self-stabilization and eventual consistency in replicated real-time databases. In: Proc. of the first Workshop in Self-Healing Systems. WOSS '02, Charleston, SC, USA (2002).
9. Imielinski, T., Badrinath, B.R.: Mobile wireless computing: Challenges in data management. Communications of the ACM, Vol. 37, No. 10, pp. 19—28 (1994).
10. Kozlova, A., Kochnev, D., Novikov B.: The Middleware Support for Consistency in Distributed Mobile Applications. In: Proc. of the Baltic DB&IS'2004, 145-160, Riga, Latvia, Scientific Papers University of Latvia (2004).
11. Mascolo, C., Capra, L., Emmerich, W.: Principles of Mobile Computing Middleware. In: Mahmoud, Q. (ed): Middleware for Communications, John Wiley (2004).
12. Murphy, A. L., Picco, G. P., Roman, G-C.: LIME: A Coordination Model and Middleware Supporting Mobility of Hosts and Agents. ACM Transactions on Software Engineering, Vol. X, No. X, pp. 1-48 (2006).
13. Pitoura, E., Bhargava, B., Wolfson, O.: Data Consistency in Intermittently Connected Distributed Systems. In: Transactions on Knowledge and Data Engineering (1999).
14. Proskurnin, O., Novikov, B.: Towards collaborative video authoring. Proc. of ADBIS'03, 370-384, Dresden, Germany (2003).
15. Rusinkiewicz, M., Klas, W., Tesch T., Wasch, J., Muth, P.: Towards a Cooperative Transaction Model: The Cooperative Activity Model. In: Proc. of the 21st VLDB Conference, Zurich, Switzerland (1995).
16. Saito Y., Shapiro M.: Replication: Optimistic Approaches. Internet Systems and Storage Laboratory, HP Labs, Palo Alto (2002).
17. Weikum, G., Schek, H.-J. Concepts and Applications of Multilevel Transactions and Open Nested Transactions. In: Elmagarmid, A.K. (ed): Database Transaction Models for Advanced Applications, Morgan Kaufmann (1992).

SCIENTIFIC PAPERS, UNIVERSITY OF LATVIA, 2010. Vol. 757
COMPUTER SCIENCE AND INFORMATION TECHNOLOGIES 85–93 P.

Heterogeneous Statistical Language Model

Kairit Sirts
Tallinn University of Tehcnology, Department of Informatics
kairit.sirts@gmail.com

Abstract. We outline the proposal for the doctoral research of heterogeneous statistical language model. The conventional language models can be classified as homogeneous, because the usage of certain structures as the base language units (words or morphemes) in lexicon is fixed. We, on the contrary, propose not to constrain the language structures to be used in the lexicon beforehand. Our model would allow the insertion of morphemes, words as well as multi-word expressions into the lexicon. For finding the optimal lexicon, we propose two criteria: the amount of language covered with the lexicon and the amount of structure of the language preserved. Such structure-preserving model will hopefully lead to better results of certain Natural Language Processing applications, like for example Automatic Speech Recognition or Machine Translation.

Keywords: Statistical language modeling, Heterogeneous language model, Heterogeneous structure of language.

1 Introduction

In the field of Natural Language Processing (NLP) there are many tasks, which use statistical language modeling (SLM). For example Automatic Speech Recognition (ASR), Machine Translation (MT), Information Retrieval (IR) and Automatic Text Segmentation are just some examples, where the statistical language models are used. Most of these tasks could be (and historically also have been) approached also with non-statistical methods, but in the last three decades the usage of statistical models has become common in such applications [17].

The most common building blocks used in statistical language models are the words. This approach has been satisfactorily applied in the languages like English, which is a rather isolating language. In more synthetic languages, where from one word many different forms can be made, the usage of morphemes has proved to be more successful [1,3]. In both cases, the building blocks are chosen keeping in mind the properties of the specific language being modeled.

We want to propose the outline of the research of the so-called heterogeneous language model. In this model the structure of the modeling units would not be restricted. We assume that the information of the most optimal building blocks is present in the

text corpus and we believe that it is possible to devise unsupervised methods, which automatically capture those structural language elements from the corpus that are most beneficial for constructing a statistical language model regardless the language. The only predefined limit would be the size of the lexicon used in the model. One would argue that the lexicon sizes used now in practical applications are not big enough to allow the capture of more complicated structures than words. Yet the success of statistical modeling is mainly due to the increase in computational power and in the future there will be more and more computational power available, which will allow the size of the lexicon to grow.

The rest of this paper is organized as follows: in section 2 we will give a brief overview of statistical language modeling and the basic mathematics involved in it. In section 3 we describe the multi-level structural division and one-level structural division of the language. In section 4 we give a short overview of the published works that might be relevant for the planned research. In section 5 the proposed research of heterogeneous statistical language model is outlined. There we also list the main problems and questions revealed so far. The section 6 gives a small example of the problem. The last section concludes the paper.

2 Statistical Language Modeling

The purpose of statistical language modeling is to capture the regularities present in a language into the statistical framework. The model consists of two parts: a lexicon and the probabilistic parameter space. The lexicon is simply a list of language elements, usually words. The set of parameters describe the regularities between the lexicon elements in a probabilistic manner.

The most common approach is to model the regularities between the words with the chain of conditional probabilities:

$$P(w_1 \dots w_i) = P(w_1) \cdot P(w_2|w_1) \cdot \dots \cdot P(w_i|w_1 \dots w_{i-1}). \quad (1)$$

The probability of the next word w_i is predicted on the history $w_1 \dots w_{i-1}$ observed. Usually not the whole history of a word will be used for prediction, because this would require an immense amount of parameters. It is a common method to make the Markov assumption that the probability of the next word is conditioned only on some fixed length history. For example, considering the history of two previous words only, the model will get the form:

$$P(w_1 \dots w_i) = P(w_1) \cdot P(w_2|w_1) \cdot P(w_3|w_1 w_2) \cdot \dots \cdot P(w_i|w_{i-2} w_{i-1}). \quad (2)$$

In general, such models are called n-gram models where n-1 determines the length of the history the next word is conditioned on. Thus the model, where the next word is conditioned on the last word only, is called bigram model and the model conditioned on the last two preceding words is called trigram model.

The conditional probabilities with history length equal to n-1 are calculated from the training corpus as follows:

$$P(w_i|w_{i-n+1} \dots w_{i-1}) = \frac{count(w_{i-n+1} w_{i-n+2} \dots w_i)}{count(w_{i-n+1} w_{i-n+2} \dots w_{i-1})} \quad (3)$$

One important aspect is how to measure the goodness of a specific model. The goodness can be tested by using the language model in a NLP application like ASR and the results are reported in the means of accuracy of the application, namely word error rate (WER) in ASR. In many occasions the appropriate NLP application is not available or it is too costly to integrate the language model into the application framework just for testing purposes [7]. In such cases a computational measure called perplexity is used. The common approach is to put a small amount of training corpus (usually 10%) aside for the purpose of perplexity evaluation and not to use this part for the model training. The perplexity is given with the equation:

$$pp = 2^{-\sum_x p(x) log_2 p(x)}. \quad (4)$$

In this equation the exponent is equal to the entropy of the data; p(x) over all x is the prior probability of the test data. As the real probability distribution of the test data is not known, p(x) is substituted by q(x) which is the observed probability distribution (posterior probability) of the training data. Thus the equation for calculating the perplexity of the test data gets the form:

$$\widetilde{pp} = 2^{-\sum_{i=1}^{N} \frac{1}{N} log_2 q(x_i)} \quad (5)$$

N is the number of elements in the test set and the reciprocal of N is substituted for p(x) for denoting the empirical distribution of the test data.

Perplexity measures the uncertainty that the model has towards the test data. The better the model fits to the test data distribution the less the model is uncertain about the data and thus the lower the perplexity. It is common practice to evaluate different models on the same test data set and endeavor toward models, which result in smaller perplexity.

Another measure used in statistical language modeling is the Out Of Vocabulary (OOV) rate which measures how many words of the test data were not present in the model's lexicon.

3 Structure of the Language

3.1 Multi-Level Structural Division

Language can be viewed as existing of structural elements of different size. We can organize them as different levels of language structure. The first level constitutes the smallest structural elements – characters. The next structural elements according to size are morphemes. From morphemes the words will be constructed. Words are organized into phrases, which in turn are used to make clauses and finally sentences. Sentences are organized into paragraphs, which in turn are parts of some bigger text.

It is obvious that the bigger the size of the language structure, the more there are different elements on that level. For example, there are only about 100 characters in the extended Latin alphabet, while the amount of different meaningful sentences is practically infinite.

In such a way we have constructed the multi-level structural division of a language. We started with the character level and ended with the level, where the whole texts could

be considered as entities. We can consider each level of this structural imagination of language to be of *homogeneous* type. This means, that each level consists of entities with the same or similar structure: character level contains the whole alphabet, morpheme level consists solely of morphemes, word level contains only words etc. Each level will cover the language totally, when we are allowed to have a theoretical lexicon of infinite size.

3.2 One-level Structural Division

As a kind of opposition we now want to look at a structural division of a language, where there is only one level, which contains the entities that can be defined to be of *heterogeneous* type. This approach has been adopted by cognitive linguists in whose works such entities are called *constructions* (see for example [13]). Constructions can be morphemes, words, idioms, partially lexically filled and fully general linguistic patterns.

Almost the same idea is expressed in [19], where the notion Morpheme Equivalent Unit (MEU) is adopted. The definition of MEU is given: a word or word string, whether incomplete or including gaps for inserted variable items, that is processed like a morpheme.

The constructionists' partially lexically filled patterns conform to partly-fixed frames, which is just one possible type of MEU. The process of deciding whether some pattern is construction or MEU is slightly differently motivated by both cases. A pattern is recognized as a construction as long as some aspect of its form or function is not strictly predictable from its component parts or from other constructions recognized to exist, while the finding of MEU-s follows the concept of Need Only Analysis (NOA), where nothing is broken down into smaller structural elements, unless there is a specific need for it. This means that the input is checked against the existing lexicon units and only where the variation is identified between the lexicon and input text, the further analysis follows, leading to the possible insertion of a new MEU into the lexicon.

4 Related Work

In addition to conventional word-based n-gram language models, among which trigram and bigram models are the most common, there are several more complex methods proposed. In our research we are most interested in the models, where 1) the n-grams are not restricted to the sequence of words observed alongside in the training corpus; 2) the grammatical structure of the language is captured; and 3) some linguistic patterns are revealed. Also we are interested in methods for automatic morpheme discovery, because this might give an insight of how to find the morpheme-like units, which may also extend over several words.

One big class constitute the models that learn the syntactical structure of the training corpus. In [2] first the sentence structure in the means of Parts-Of-Speech (POS) tags is estimated and then most likely words in places of POS tags are found. Models described in [4], [6], [8] and [20] create for each input sentence a parse-tree, which takes into account also head-words of the sentence parsed so far, enabling to capture the dependencies of the words, which are not alongside in the sentence.

The model in [5] makes use of dependency grammar, which is created separately for each training sentence. The grammars are presented by directed graphs, showing which words are dependent on each other.

In [12] instead of the conventional n-grams the multigrams are used. Fixed m is given and all n-s in n-grams must satisfy the inequality n <= m.

The model in [15] is trained using Latent Semantic Analysis (LSA). The words are also provided with their POS-tags, the latter information being used in the training process. In the training also the information about content words and stop words is used, helping to catch the dependencies between the words more accurately.

The methods for expressing the dependencies of the words not been alongside in the training corpus are presented in [16] and [18]. In [16] those dependencies are called triggers, while in [18] they are called intermediate distance n-grams. The essential nature of the both constructions is the same; they are basically n-grams, in which certain number of words is allowed to be between the words the n-gram consists of.

The models with lexicon consisting of morphemes instead of words are presented for example in [1], [3] and [10]. The methods for unsupervised discovery of morphemes from a text corpus are described in [9] and [14]. In [11] the morphemes discovered from a corpus are used for finding the multi-word structural patterns from the language.

5 Heterogeneous Statistical Language Model

The essence of the thesis will be established on the heterogeneity of the language structure. We will make a kind of simplification and from this point forward we will use the notation of MEU to refer to any structural language pattern, be it morpheme, word or a multi-word expression.

The planned scope of the research includes extracting the lexicon consisting of MEU-s from a text corpus and building the statistical language model. Two main questions have to be answered:

- How to find and extract the relevant MEU-s from the text corpus?
- How to build a statistical language model based on this lexicon?

We now inspect the both problems a little bit closer and we start with the problem of segmenting the text corpus into heterogeneous entities MEUs. First it must be pointed out that the objective of segmenting the corpus is to extract a certain amount of elements to be put into the lexicon of a statistical language model. The size of the lexicon can vary to some extent, but for practical reasons it must stay into certain borders, which, as already mentioned, are usually between 50K and 120K items. As we already explained, the bigger structural elements we consider as lexicon items, the more language structure will be stored into the lexicon; alas the bigger lexicon size is necessary for covering the same amount of language and preserving the low OOV rate. In a situation where the lexicon size is fixed in terms that its upper limit is given, two degrees of freedom remain – the amount of structure preserved and the amount of language covered. Now we come to some questions or sub-problems:

- How to measure the amount of preserved structure with a certain lexicon selection?
- How to decide that the balance between structure and coverage in circumstances of certain lexicon size has been achieved?

- How to achieve this balance between the preservation of structure and coverage of the language with the fixed lexicon size?

First we need to get the text corpus segmented into elements to be potentially put into lexicon. When building a lexicon where the items are allowed to have only homogeneous structure, this task is simple. It is easy to find the set of all distinct words of a corpus, calculate the frequencies of occurrences for each word and take the first N most frequent words to be put into lexicon. The task is a bit more complicated with morphemes as specialized algorithms are necessary for finding the morpheme borders in the words. If the morpheme sequence has been discovered the same process as with words can be carried out to form a lexicon. The task is however not trivial with heterogeneous elements, because there are no clear borders between the different elements in the text and also the elements can overlap each other in the sense that a word may constitute a MEU while there might be another multi-word MEU containing the same word. So we need to devise a method for finding the possible MEU-s from the text corpus. We have not yet performed any research in this matter, but the following ideas could serve as a starting point:

- The types of elements the text is to be segmented to could be restricted to morphemes, words and multiword expressions.
- The list of all such elements with their frequencies could be formed;
- There are several possible ways of how to approach to finding the multi-word expressions. For example 1) find the most frequent collocations within the sentence; 2) adopt the Part Of Speech (POS) tags to find the patterns with gaps, where some certain types of words (nouns, verbs etc) can be inserted into.

It must be noticed that the result of such an activity is not a proper segmentation of text but rather a list of textual patterns and this list has the heterogeneous structure as it may contain morphemes, words as well as multi-word expressions.

Next task is to select the proper elements from the list to be inserted into the lexicon. Due to the circumstances that the list will contain also the multi-word MEU-s, there might be several sets of items that can equally well form a lexicon and this means we need to devise a method for selecting the most optimal set of elements, which preserves the structure of the language best, while covering the most language at the same time.

Let us assume now that we have been able to extract the relevant heterogeneous entities into the lexicon which represents an optimal balance between the structure retained and language covered. As our longer goal has been to build a statistical language model then logically the question follows: how to use this lexicon for building a statistical language model? How to build up the conditional probabilities between the lexicon items considering that some of them are multi-word expressions and may even contain gaps and most probably some of them are overlapping each other?

There remains yet one important question of how to measure the goodness of this language model comparing to other statistical language models? As was explained before, there are two common methods of measuring the goodness of a language model – 1) computationally and 2) by evaluating the output of a NLP application where the model has been incorporated into. In both cases the evaluation of the model has to be set up as a comparison of the proposed model with a baseline model using some widely accepted configuration. The most common approach in statistical language modeling is

to use conventional bigram and trigram models as baseline models. The computational evaluation would then involve forming the proposed model and appropriate baseline models based on the same training data and calculating the perplexity measures for all models for training and also some unseen test data.

Different NLP applications set different specific requirements to the statistical language model to be used. Therefore, in order to evaluate the goodness of a model according to the output of an application, it is necessary to choose a specific application the statistical language model has to be adapted to. In addition, as one of the targets of our research is to devise a language-independent method for finding the proper heterogeneous entities that could be used as building blocks in the statistical language model, the validation of the model should involve experiments with different languages.

6 Example

We now try to give some examples of how the heterogeneous segmentation of an English sentence could look like. As we currently do not yet have any method implementation available, which would segment the text into heterogeneous entities, the given examples present just the idea of what kind of segmentation we would expect to emerge. We also have to remind that the aim is to produce a data-driven segmentation method, which is by no means objective and is solely dependent on the data it is applied to. One of the input parameters to the system is planned to be the maximum number of elements the vocabulary can contain. Thus, the resulting segmentation will also be dependent on this parameter. For example, if the capacity of the vocabulary is for some reason only as much as 30 elements then if the corpus is Estonian text, we can put only single characters into the lexicon. While the amount of the lexicon grows, the more morphemes could be put into the lexicon, until at some point most frequent words will emerge as vocabulary element candidates. The moment when bigger structural elements will start to enter into the vocabulary is different with different languages, making the result of different languages structurally very different.

As an example we will look now the following English sentences:

Everything was going by the plan. Dinner was ready at eight as expected.

The possible segmentation of those sentences following our intuition is given below. The result is presented so that after each segment the structural type of the segment is given in brackets. The structural type phrase should not be taken literally, as we use it to describe any parts longer than words.

Everything [word] **was** [first part of the phrase] **go** [morpheme going into the phrase gap] **ing** [second part of the phrase] **by the plan** [phrase]. **Dinner** [word] **was** [word] **ready** [word] **at** [word] **eight** [word] **as expected** [phrase].

We must note that even this short segmentation is definitely arguable, because some of the parts segmented as words could be segmented as phrases with slots: was [part of the phrase] ready [word to be put into the phrase gap], at [part of the phrase] eight [word to be put into the phrase gap]. It is very difficult to estimate, how these things

would work out in a real experiment. The ambiguity of the possible results speaks in the favor of the argument that the true quality of the method will be revealed only through an application.

7 Conclusion

We presented an outline for a doctoral research, which objective is to construct a statistical language model of heterogeneous language entities, by heterogeneousity meaning here that the language elements used could be any of morphemes, words or multi-words expressions. In order the reach the objective, the following problems must be tackled:

1. Find the list of heterogeneous elements of a text corpus;
2. Create an optimal lexicon of fixed size by finding the optimal balance between the preservation of the language structure and coverage of the language.
3. Use this lexicon for building a heterogeneous statistical language model.

There are several advantages we see the proposed model could have over the conventional statistical language models. First, with such model there would be no need to state explicitly beforehand of what base units the model will be constructed. The set of optimal base units to be put into the lexicon would be automatically detected based on the training corpus and structural preference of the items will probably be different for different languages. It might as well happen that with some languages the bordering situation will occur, where the optimal lexicon will consist only of items of homogeneous structure, but with the proposed model, there would be no need for a prior language-specific decision for that.

The another major possible advantage we see, is that the more structure the model lexicon will preserve while at the same time having the high coverage of the language, the better results are expected to be output by a NLP application.

References

1. Alumäe, T.: Methods for Estonian Large Vocabulary Speech Recognition. Ph. D. Thesis. Tallinn University of Technology (2006)
2. Amaya, F., Benedi, J. M.: Improvement of a Whole Sentence Maximum Entropy Language Model Using Grammatical Features. In: Proceedings of the 39th Annual Meeting on Association for Computational Linguistic, pp. 10--17 (1981)
3. Arisoy, E., Saraclar, M., Hirsimäki, T., Pylkkönen, J., Alumäe, T., Sak, H.: Mihelič, Fr., Žibert, J. (eds.) Statistical Language Modeling for Automatic Speech Recognition of Agglutinative Languages. Speech Recognition : Technologies and Applications, Ch. 10, pp. 194--204 (2008)
4. Chelba, C.: A Structured Language Model. Computer Speech and Language, 14, pp. 283--332 (1998)
5. Chelba, C., Engle, D., Jelinek, F., Jimenaz, V., Khudanpur, S., Mangu, L., Printz, H., Ristad, E., Rosenfeld, R., Stolcke, A., Wu, D.: Structure and Performance of a Dependency Language Model. In: Proceedings of Eurospeech, Vol. 5, pp. 2775--2778 (1997)
6. Chelba, C., Jelinek, F.: Exploiting Syntactic Structure for Language Modeling. In Proceedings of COLING-ACL'98, Vol. 1, pp. 225--231 (1998)
7. Chen, S., Beeferman, D., Rosenfeld, R.: Evaluation Metrics for Language Models. In Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop, pp. 275--280, (1998)
8. Collins, M.: Head-Driven Statistical Models for Natural Language Parsing. Ph. D. Thesis. Univeristy of Pennsylvania (1999)

9. Creutz, M.: Induction of the Morphology of Natural Language: Unsupervised Morpheme Segmentation with Application to Automatic Speech Recognition. Ph. D. Thesis. Helsinki University of Techology (2006)
10. Creutz, M., Hirsimäki, T., Kurimo, M., Puurula, A., Pylkkönen, J., Siivola, J., Varjokallio, M., Arisoy, E., Saraclar, M., Stolcke, A.: Analysis of Morph-Based Speech Recognition and the Modeling of Out-Of-Vocabulary Words Across Languages. In: Proceedings of HLT-NAACL 2007, pp. 380--387 (2007).
11. Déjean, H.: Morphemes as Necessary Concept for Structures Discovery from Untagged Corpora. In: Proceedings of the Joint Conferences on New Methods in Language Processing and Computational Natural Language Learning, pp. 295--298 (1998)
12. Deligne, S., Bimbot, F.: Language Modeling by Variable Length Sequencies: Theoretical Formulation and Evaluation of Multigrams. In Proceedings of ICASSP, Vol. 1, pp. 169--172 (1995)
13. Goldberg, A. E.: Constructions: A New Theoretical Approach to Language. Trends in Cognitive Sciences, Vol. 7, Iss. 5, pp. 219--224 (2003)
14. Goldsmith, J., Hu, Y., Matveeva, I.: A Heuristic for Morpheme Discovery Based on String Edit Distance. Technical Report TR-2005-4. University of Chicago (2005)
15. Kanejiya, D., Kumar, A., Prasad, S.: Statistical Language Modeling with Performance Benchmarks Using Various Levels of Syntactic-Semantic Information. In Proceedings of the 20th international Conference on Computational Linguistics, Article 1161, (2004)
16. Rosenfeld, R.: A Maximum Entropy Approach to Adaptive Statistical Language Modeling. Computer Speech and Language, Vol. 10, pp. 18--228 (1996)
17. Rosenfeld, R.: Two Decades of Statistical Language Modeling: Where Do We Go from Here. In Proceedings of IEEE, Vol. 88, pp. 1270--1278, (2000)
18. Saul, L., Pereira, F.: Aggregate and Mixed-Order Markov Models for Statistical Language Processing. In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing, pp. 8--89, (1997)
19. Wray, A.: Formulaic Language: Pushing the Boundaries. Oxford University Press (2008)
20. Wu, J., Khudanpur, S.: Combinging Nonlocal, Syntactic and N-gram Dependencies in Language Modeling. Proceedings of Eurospeech'99, pp. 2179--2182, (1999)

LATVIJAS UNIVERSITĀTES RAKSTI
757. sējums, DATORZINĀTNE UN INFORMĀCIJAS TEHNOLOĢIJAS

Latvijas Universitātes Akadēmiskais apgāds
Baznīcas ielā 5, Rīgā, LV-1010
Tālrunis 67034535

Iespiests SIA «Latgales druka»