2010

Model-Based Management – Design and Experimental Evaluation

Gunnar Dietz
Dongbei University of Finance and Economics, mail@gdietz.de

Martin Juhrisch
Dresden University of Technology, martin.juhrisch@tu-dresden.de

MODEL-BASED MANAGEMENT – DESIGN AND EXPERIMENTAL EVALUATION

Gunnar Dietz, International Education Center, Dongbei University of Finance and Economics, Dalian, China, mail@gdietz.de

Martin Juhrisch, Chair for Information Systems, Dresden University of Technology, Dresden, Germany, martin.juhrisch@tu-dresden.de

Abstract

Business Process Models contain a lot of information. They are created with different objectives in mind by people with different background. Many models contain information about an organization's structure and the application systems or services used within the organization. Several tasks are done by people with different roles using different resources. Identity Management Systems (IDMS) try to offer a way to manage all these information automatically. After introducing an IDMS it is easy to cope with changes in identities (persons) and their roles. However, one main question often remains: How to identify good roles that are capable to ease the task of assigning people to resources? In this article a model-based approach using ratios is presented. Complexity, cohesion and coupling for roles are introduced and evaluated to come to a good set of roles representing what they should represent: a bundle of similar organizational functions and resources.

Keywords: Conceptual Modeling, Model-Based Management, Guidelines, Semi-formal Models.

1 INTRODUCTION

Today, conceptual models are intensively used for the development and adaptation of software systems as well as for the (re-)design of organizations. To support the efficient use of models and to support the creation of models itself, several methods have been established within the information systems (IS) discipline. IS developers use conceptual modeling languages to represent information of a business domain by composing the modeling grammar's constructs to a conceptual model (Wand and Weber 2002; Frank 1999).

In model-based management, conceptual models are the essential instrument in the administration, configuration and improvement of an enterprise. Usually management, i. e. quality management, business process management, system simulation, is a highly manual challenge and causes extensive effort. An increased efficiency could be achieved if the necessary activities are solved completely model-based. In a model transformation, the source model is modified until a solution to the problem that is described is found (Brinkkemper 1996; Juhrisch 2010). The appropriate solution may again initially be displayed as a model.

Within the various stages of management, a model is both – result and necessary condition – for certain activities. Then algorithms, based on a set of rules, perform model operations. Thus, the required time and effort for solving the overall task is reduced. As part of the model-based solving process the business problem is initially transmitted in the model space through the development of a problem model. Once a model of the business problem has been created, the problem solving is carried out in the model space.

Regarding the conceptual modeling process in general, modeling can be seen as the use of a typing, that is represented by the language definition of a method. Thereby, elements of the real world are assigned to a category (a language construct) and are put into the model as an instance of this category. Modeling methods focus only on the creation of models, without giving concrete instructions of an appropriate use of models. This is not sufficient, since in model-based management models are not the goal, but always additives to resolve specific operational problems.

Methods that help to resolve business problems with the use of models should therefore describe and integrate three processes: the model creation, the model transformation, and the model use. Such methods are called model-based methods (Brinkkemper 1998; Juhrisch 2010).

One of manifold business problems is the management of service oriented architectures (SOA) (Erl 2005; Walsh 2002). This again is a wide field, where one problem is to manage the security of the underlying (Web-)service layer, namely the access rights for the services within a SOA. It seems that this problem is already addressed by Identity Management Systems, however, most identity management solutions only address application access control instead of access control for a fine-granular SOA. Furthermore, and more important, most identity management solutions implement (a certain interpretation of) role based access control (RBAC) (Ferraiolo and Kuhn 1992). Access rights are controlled by roles and a set of rules and policies that determine which role includes which access rights. All that remains to do is to sort people into roles so that they automatically get those access rights they need.

The main question in this scenario is: How to come to a useful set of roles that can be used in this approach? The answer to that question heavily depends on the organizational structure, the business processes and their implementation as a SOA. Role management means to answer the question: Who is doing what within the organization and why? The three question words here represent the following: "Who" here means user management, "what" means to have an overview of necessary applications, services and access rights, while "why" reflects the role management. (An accountant has access to accounting information because his role as an accountant reflects the need for corresponding access rights.) After a useful set of roles has been identified, the ordinary process of access management becomes relatively easy with the help of identity management systems.

In this paper we analyze the characteristics of a model-based method by exemplarily answering the above question. Therefore we propose a model-based method to derive a useful set of roles for a given organizational scenario. Models that reflect the organizational structure, business requirements and process information as well as service implementations should be used to derive a set of role candidates and a set of rules and policies that map these role candidates to access rights.

There are two main scenarios for this:

- An identity management that contains a certain set of roles is not yet existent. That means that role candidates have to be identified "from the scratch" using process responsibilities that are documented in business process models. This scenario again splits into two scenarios:
    - Also a service oriented architecture is still not existent. That means also service candidates have to be identified (see Juhrisch and Dietz, 2010). The identification of role candidates and service candidates can be done in one step.
    - A service oriented architecture is already existent. Design (or implementation) models for the SOA or a service catalog is used to map the role candidates to access rights.
- A set of roles already exists, so that one needs to map the business requirements of process responsibilities to the already existing and map these roles to the resulting access rights for the services in question. This scenario also includes the change management for changed (implementations) of business requirements, that may result in new rules and policies.

In both scenarios one has to identify responsibilities (organizational units, organizational functions which may be recorded in personal data, ...) for certain process activities, map these process activities to services and question, derive necessary access rights and agglomerate this information into useful sets of roles. A role represents both a group of persons with certain responsibilities as well as a group a services that needs to be accessed by these persons and therefore acts as a link (some kind of "glue") between these two sides. The result in both cases is not just a set of rules but also a set of rules and policies that connect these roles to access rights in the sense of RBAC. This set of rules and policies can be pictured by a model transformation of the original models into models that show the relationship between responsibilities and access rights.

The paper is structured as follows: After this introduction we introduce both aspects of a model-based identity management – access control and role identification. Based on these requirements this paper then presents a solution in chapter 4, which then should be used in the original motivation: The model-based configuration of application access. The paper ends with a discussion and the future research.

2 MODEL-BASED ACCESS CONTROL

The requirement to manage the underlying security aspects for (Web-)services of a SOA comes along in most cases with the need of a management of the SOA itself. The security and access control not only changes when responsibilities within the organization change but they also have to be altered when either implementation aspects of the SOA or functional requirements, that have to be covered by the SOA, change.

As mentioned, the security management done by identity management systems relies on roles (Ferraiolo and Kuhn 1992; Esswein 1993). While an identity management is mainly responsible for person to role assignments, synchronizing personal data within several systems and realizing or enforcing certain rights based on these roles within target application systems, the right management for a fine granular SOA is normally out of scope of an identity management system.

The following information is needed or has to be created for this task:

- A catalog of all services within the SOA and all access rights possible for these services
- A catalog of all roles within the organization
- Rules and policies for mapping roles to access rights

- Technical information about how to access the right management for particular services to provide the necessary information for enforcing rights

Especially the second point – to come to a set of roles that really covers all the needs for controlling the access rights – together with the third point – derive a set of rules and policies that map the roles to access rights – is one of the main challenges when introducing an identity management. This is not only important during the introduction phase. Even after the identity management system has become active it remains a very important task to have a role management that respects changes of requirements over time.

As mentioned, changing requirements may either alter the SOA itself or alter the roles and rules within the identity management. Therefore it is self-evident to use the same approach for a model-driven control for both scenarios. The approach proposed in this article is based on the description kit approach (DKA) that is shortly introduced in Chapter 3. The DKA here is used to enrich models by additional information in the form of so-called descriptions. The use of descriptions is controlled, so that these descriptions result in computer-readable information that can be used for a model-based problem solution. The control is achieved by so-called description kits (DK) and description kit types (DKT).

To address the role management problem we created description kit types and description kits for entering role information (not necessarily roles itself) in a way that it mirrors personal information stored within the IDMS into models (see Juhrisch et al. 2010). They are used to create descriptions for persons or roles respectively that should be able to use a certain process within a process model (e.g. “a student within department 3”, “secretary of head of department 3”, etc.).

The DKA also includes a generic mapping algorithm that is able not only to identify service candidates within process models and to map (in a semi-automated way) processes to services within a service catalog, but also – completely in the same way – to identify role candidates based on the role information and to map role information to certain already productive roles within some “role catalog” (offered by the identity management) (see also Juhrisch and Dietz 2010). This procedure results in a model transformation that yields a model that contains the list of services, the list of roles and all connections between these two sides.

Since this transformation process is nearly completely electronically, the next step is to access this information and make it available to the identity management. (In fact the transformation is semi-automatic and involve decisions of an expert to enhance and ensure the quality of the results.) To use this data, the identity management system (IdMS) must be able access the role information for each process, translate this information to account data and propagate this information to the Web service or user management that is responsible for the process in question (Juhrisch et al. 2010). To models that serve as an input to the transformation process used here rely on the information entered into the descriptions. These descriptions can either form models on their own or can be used to enrich existing models like event-driven process chain (EPC) diagrams (Scheer 2000), which hold role information for each process. The information within the descriptions is then made accessible via a Web service of the modeling toolkit.

The DKA is generic, and integrating role information into models (in an electronically accessible form) is easily possible using this approach. As mentioned above, a description kit type for entering role information is created. This description kit type is used to create description kits that serve as templates for entering specific role information in descriptions. A description kit may e.g. represent courses of study for a student or institutes for research staff members. The DKA includes hierarchic ordering of information that makes it possible to describe different aspects of a role. A concrete description represents concrete role information, e.g. a specific course, but may also include “wildcard” information. In addition to the specific roles the organizational structure is modeled at Description Kit level to make it easy to refer to the organizational structure in a defined way on description level.

Now, on the ordinary modeling level concrete roles like “student at the department of biology”, “university administration staff member” or “head of the department 2.1” can be described using the

Descriptions. The role information is either a given specific role – pre-modeled at DescKit level - or an arbitrarily created role descriptions using a certain Description Kit. When using e.g. EPC models as mentioned above, the role information is added to each activity in an EPC chart.

In the concrete implementation of a concrete identity management scenario we have done the following: Using description kit types we enriched ARIS models and created – among others – a "role" DKT. This DKT defines corresponding parameter types, embedded DKTs and may include the definition of concrete values for the some parameter, e.g. "student", "employee", …. This allows at DescKit to create either instances of fixed roles or to create arbitrary roles on model level – or a combination of both.

Furthermore we created Web services to access this role data (see Juhrisch, Weller and Esswein 2010) within the modeling tool. These Web services grant access to the specific role descriptions (Descriptions of DescKitType ‚role') in a process model. An identity management system then can use this Web service to implement the roles within the IdM given by the models. A first proof of concept has been done using the IBM Tivoli Identity Manager (ITIM). A second proof of concept has then been done using the Novell Identity Manager. In the latter case a so-called driver was created for the Novell IdM that is able to access the Web service to synchronize the role information into the so-called identity vault. An internal logic has then been implemented to translate the role information into entitlements for the services in question. Now the IdMS can react immediately to a change of role information within the models and chance the corresponding access rights for some services.

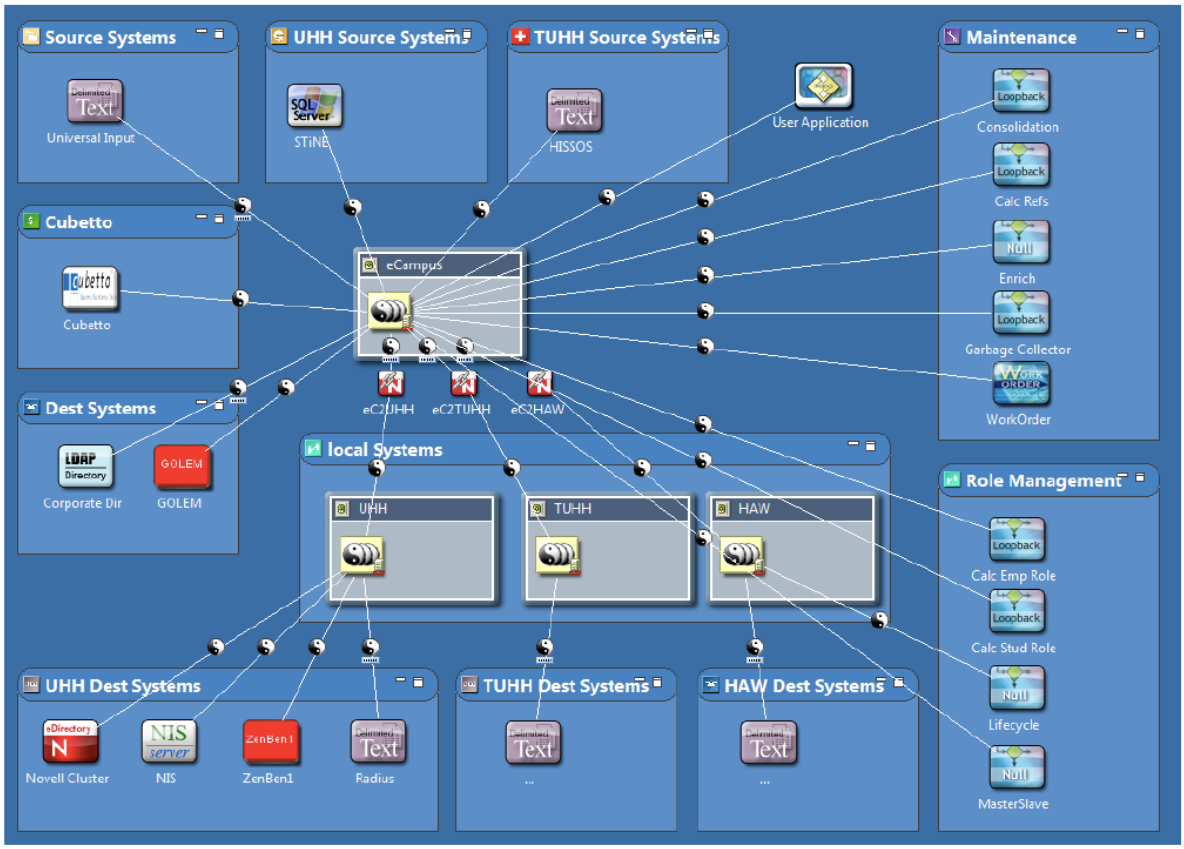

Figure 1. Design of the eCampus III IdMS and the integration of cubetto Toolset.

3 MODEL-BASED ROLE IDENTIFICATION

While the previous chapter was focussing on how to manage access to resources (with help of identity management systems) in a model based way, the main question remains: How to come to a useful set of roles? The previous chapter assumed that models are already enriched with useful role information. However, roles are often modeled "ad hoc" without thinking of their usefulness in an identity

management scenario. So is for example the role “secretary of Mr. Miller” not a useful role for an identity management. It is too fine granular and also has a reference to a person that may change. Some other roles may be useful or not useful depending on the context. The role “Excel user” is for example a technical role that does not really reflect the function of a person. It may be useful nevertheless when talking of role implication. On the other hand, if every person (in a company) that uses Excel also uses Word, then it would be more useful to use a role “MS Office user” instead.

A role (in an identity management scenario) should be the glue between persons and their (access rights to) resources. Depending on the roles of a person a specific right will be granted or not. The question is to find the right balance between using one role for each person and each access right and only one role “access to everything”.

We propose here a model driven way that helps answering this question by generating ratios for “role candidates”. These ratios reflect the usefulness of a role within the whole organizational scenario for implementing it into an identity management system. After this role identification and role implementation process the foundation is laid so that all ideas of the previous chapter can take in.

Starting with the ideas of the previous chapter, we use the Description Kit Approach for creating models that contain role information in a restricted way. However this time the role modeling should be nearly not restricted at all, so that the domain experts can describe any role they have in mind. Only the way how these roles are modeled should be restricted. Here the DKA enfolds its full strength by providing guidelines for a restricted modeling so that this information then can be processed by some algorithms. The DKA also includes algorithms that supply a basis for creating the ratios (see Juhrisch and Dietz 2010).

3.1 Description Kit Approach Algorithms

The DKA not only offers a framework for modeling, but also includes a set of algorithms. Since the DKA is generic to be applicable for different scenarios, the same is true for the algorithms. To address a model-driven problem solution, it is necessary to evaluate the models. “Evaluation” here is means understanding the models, while “understanding” is based on comparison. Therefore the main part of the algorithms of the DKA is about model comparison.

Comparison does not necessarily mean to compare two different models, but is also quite important if there is only one model. In this case it is still interesting to compare different parts of the model: Do the different parts of a model contain very different information or are the nearly the same? The answer to this question yield to a measurement of complexity and cohesion.

The DKA come with an algorithm that is able to compare two different models or two different parts of one model (in the following called sub-model), which use the same so-called description kit language (DKL). This means they use a common set of description kit types and and overlapping set of description kits. This algorithm has essentially two steps (see Juhrisch and Dietz, 2010). The first step is a 1-to-1 comparison of two single descriptions (which may have other descriptions within their embedding hierarchy). This step generates a ratio (a number between 0 and 1) that expresses to which degree these two descriptions are the same. The second part compares two sub-models (that may consist of several descriptions and relations). It highly relies on the convolution of a (complex) sub-model structure into one single (but maybe rich) description, so that the first part of the algorithm can be used. For details on the algorithm see again Juhrisch and Dietz, 2010.

Figure 2: The Description Kit for Role

The convolution operation is done by folding the sub-model step-by-step together along the relations. This means that for each relation a folding operation has to be defined. This folding operation highly depends on the context and the modeling problem in question. In the present case, analyzing role information, the operation used here is quite simple compared to other scenarios (compare again Juhrisch and Dietz, 2010). It collects all needed access rights that are necessary to perform a certain sub-process that is represented in the sub-model. The operation can be written as follows, where it uses description kits of the form given in Figure 2:

$$\Big(\big\{\text{Process}\{\text{Role}\{\text{OrgUnit } O_1, \text{Function } F_1\}\}\{\text{Resource } R_1\}\big\}$$
$$\xrightarrow{\text{Process Flow}} \big\{\text{Process}\{\text{Role}\{\text{OrgUnit } O_2, \text{Function } F_2\}\}\{\text{Resource } R_2\}\big\}\Big)$$
$$\xrightarrow{\text{Convolution Result}} \big\{\text{Process}\{\text{Role}\{O_1, F_1\} \cup \text{Role}\{O_2, F_2\}\}\{\text{Resource } R_1 \cup R_2\}\big\}$$

The curly braces denote the embedding hierarchy reflecting the structure as in Figure 1. The operation $*$ is a predefined operation meaning □union□ and is based on the 1-to-1 mapping algorithm. If this algorithm detects a match between the two sides of this operator, both descriptions are iteratively combined to one description; if no match is detected then these two descriptions are still left separated (but embedded in the outer description).

With this convolution operation the algorithm then step-by-step “collects” all the different pieces of information of one sub-model into one (probably very big) single description. This happens to both sides of the comparison and yields two “big” descriptions that can be compared by the 1-to-1 comparison algorithm.

3.2 Determining Ratios for Role Candidates

It is important to remark, that the “roles” in the previous part (see also Figure 2) are not ready made roles but rather role descriptions that could be a good role candidate - or could be part of a role combining several functions and access rights.

To determine ratios that can rate the quality of role candidates we need to analyze the models in a way that is able to detect role information that is used in a large and well-defined (coherent) business area. A good role must should be simple, but not too simple. A role would be “too simple” if it would cover only a small business area. That would make it easy to handle the role, but the role would be not really useful, since it can’t be used in other business areas. These two at first glance contradicting facts can be summarized as follows:

- A good role should have a low complexity with regard to the information that is used to form this role. The information here is threefold. It includes the “two sides” of a role, namely organizational structure and responsibilities on the one side and necessary access rights on the other side, but also

includes the “glue” between these two sides, namely the processes that create a link between these two sides.

- The role information that forms a good role should have a high cohesion, which means that a lot of similar information is used to define the role. Cohesion is therefore a measurement for how large the business area is, where the role could be used.

A third requirement for good roles is the following

- The role information that forms a good role should have a low coupling, which means the role is (quite) independent of other roles.

In the following we define three ratios that measure the quality of a role with respect to each of these requirements. The ratios make heavy use of the convolution algorithm to search for areas (sub-models) within a model that can be combined to good role candidates. The results of the convolution process are then used to generate ratios that determine the applicability (or usefulness) of the resulting role. The trick is that a highly coherent sub-model would “collapse” during the convolution.

First we need a measurement for complexity. It can then be defined as the number of different role descriptions within a certain sub-model. Note that we don’t count the number of resources here, since a higher bundling of resources will make the role even more applicable. Complexity here therefore measures a scattering of responsibilities or tasks throughout the whole organizational structure.

Note that complexity is not an inherent measurement, but highly depends on the granularity of the model description and is therefore only a vague and absolute measurement. We denote the complexity of a sub-model P by $K(P)$.

Now define the cohesion by

$$\Lambda(P) = \frac{K(P)}{K\big(F(P)\big)}$$

where $F(P)$ denotes the convolution result of P. The cohesion defined in this way compares the complexity of a sub-model with the complexity of the same sub-model after the convolution. It therefore measures how much a sub-model collapses when convoluted. Note that a variance in granularity here cancels out so that cohesion is in fact (nearly) an inherent measurement.

The last ratio, the coupling, contrasts the previous ratios that were analyzing the inner structure of a sub-model by comparing the sub-model to the information outside the sub-model. A measurement for the coupling of two sub-models is

$$\overline{\kappa}(P_1, P_2) = \frac{K\big(F(P_1)\big) + K\big(F(P_2)\big)}{K\big(F(P_1 \cup P_2)\big)} - 1$$

(This formula uses the operator $*$ as described above.) This value is 1, if the convolution of the union of P_1 and P_2 has the same complexity as P_1 or P_2 respectively, which means the case of highest coupling, or 0, if the complexity of the union is just the sum of the complexities of P_1 and P_2, which means that both subprocesses have nothing in common and therefore are not coupled at all.

These ratios are used to identify a good set of role candidates. Since the whole process remembers where these roles come from, a model transformation is possible that yields process model that is enriched by the now decided roles and contains enough information to determine the set of rules and policies that links the roles to the necessary access rights. The only task that remains them (the easiest part of identity management) is to map persons within the organization to their roles.

4 IMPLEMENTATION

Many or even most German universities are currently planning or developing identity management systems, or have already done this. Some of the universities that already implemented an identity

management system had to learn (often in the hard way) that identity management is a task that does not stop when the identity management system has gone active. Universities have in common that firstly the set of persons within the organization (the biggest part are the students) changes very frequently. Furthermore often the organizational structure changes from time to time and persons have often not only one role but several roles within the organization.

Hamburg, Germany, is a city state with several universities with different portfolios. The city Hamburg - especially against the background of the Bologna process (European project for creating academic degree standards) - has to compete other cities as a common educational location. Therefore the six public universities of Hamburg, namely the University of Hamburg, the Hamburg University of Technology, the Hamburg University of Applied Sciences, the HafenCity University, the Hamburg University for Music and Drama and the University of Fine Arts have launched together with the university libraries in Hamburg a project called eCampus to meet the upcoming challenges. This project is coordinated by the Multimedia Kontor Hamburg, and is currently in the third phase eCampus III which ends in 2011.

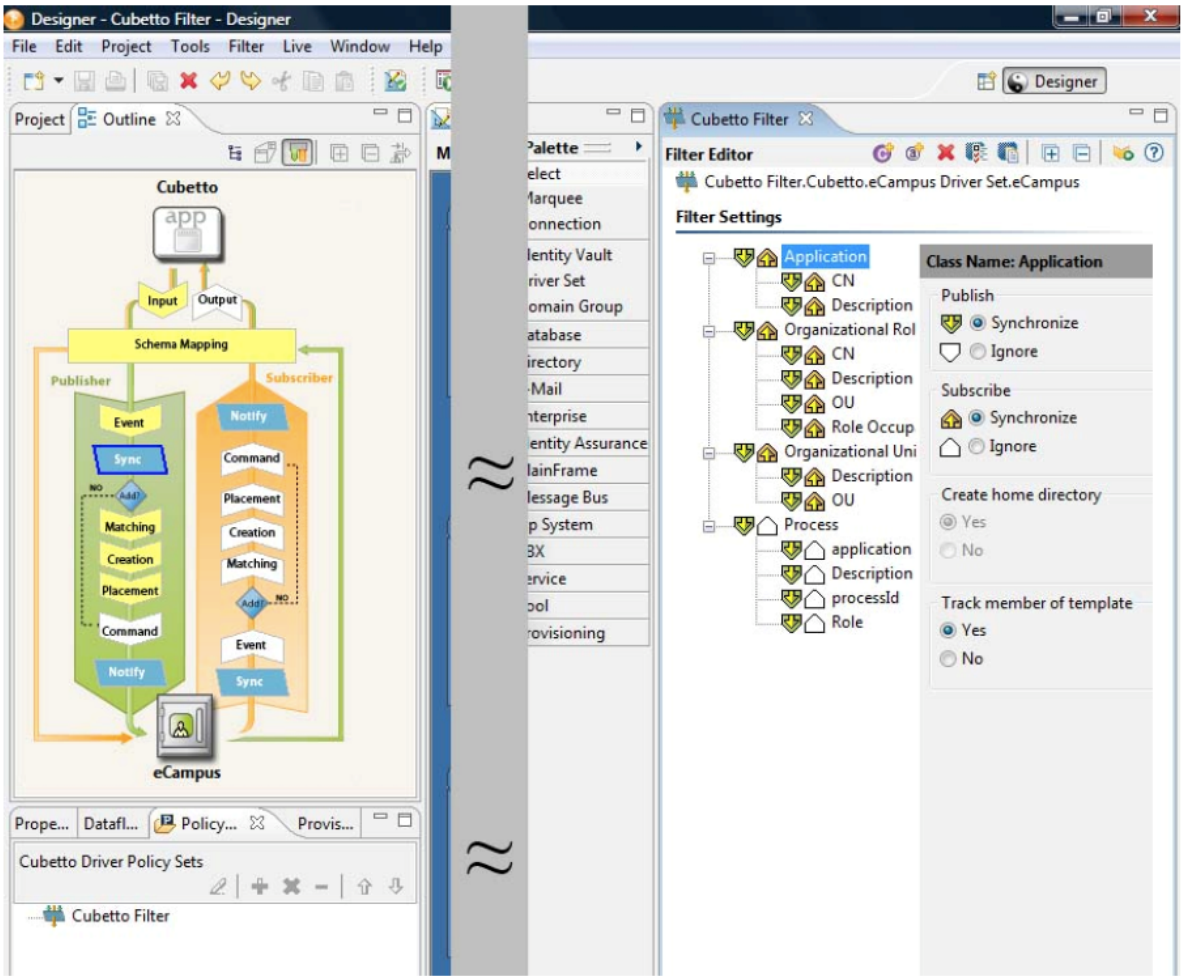

Figure 3. The cubetto driver for exchanging role data between models in cubetto and the IDMS

One goal of the project eCampus is the development of a common identity management system that assigns students and employee of any one of the public universities one unique identity that is valid in all participating universities. This project not only focusses technical issues like the creation of a common metadirectory and an identity management logic, but even more organizational and political challenges. Of high concern (especially in Germany) are data security and privacy aspects that turn the project into a long turn challenge. One organizational aspect of high importance is the identification and implementation of roles.

The cooperation of several universities made a complex requirements analysis necessary which led to a cooperation with the University of Technology Dresden (Germany) and resulted in the development of the approach presented here. In earlier phases the research was focussed on a restricted and distributed modeling, which - together with other model-based methodologies to face certain organizational problems - led to the DKA. The mapping algorithms were developed later on and laid the foundation for the creation of ratios.

For a technical design of the identity management system see Figure 1. To implement the here proposed approach a connection between the modeling tool (which is the “Cubetto Toolset” (see Cubetto 2009)) and the identity management system was developed, see Figure 3. This laid the base for a prototypical implementation of model-driven access control.

Prior to the proposed approach the role management was a highly manual task. The usefulness of models created during the analysis phase for a role analysis was limited due to the complexity and due to differences between distributedly created models. One major difficulty was to connect very technical viewpoints of technical experts with organizational needs especially in administrative processes. The DKA here helps build a bridge between different “domains” (viewpoints and fields of expertise) and to cope with the complexity of the models by an automated analysis.

5 CONTRIBUTION AND FURTHER RESEARCH

This research makes several theoretical contributions. It underlines the usefulness of the description kit approach by defining a concrete scenario (role management) and the prototypical application of the approach. This extends the current research on model-based method theories by taking a focus on - even if not explicitly mentioned in this article - guidelines during the modeling process and algorithms that enable a model comparison (see Juhrisch, Dietz and Esswein 2009 for a discussion on guidelines). Furthermore it demonstrates how the description kit approach is able to solve several types of conflicts in model-based management, especially the domain conflict (see Juhrisch and Dietz 2010 for a discussion on the domain conflict). Additionally, our research offers implications for practitioners. It has taken a specific focus on identity management systems and a model-driven way to answer the most important task of role-engineering: What is a useful set of roles?

Even if the identity management project is still in development, the description kit approach proved its usefulness within the first case studies and seems to be promising to be used within the productive identity management scenario. First tests have been done within the identity management project of the universities of Hamburg. Further research is required in order to examine how well a detailed process analysis can be done with the help of the cubetto Toolset, since the acceptance of the modeling tool within the university administrations is still an open question. Moreover, the local IT-support organizations of the university have to be involved to finalize the implementation of the approach and to prepare the roll-out to the universities' administrations.

We have developed a conceptual framework for role-engineering by adapting the DKA. Our framework addresses our research question by presenting the various sources of information that affect the application of semi-formal models. The framework highlights the fact that roles have to be adapted and used appropriately based on the project, the organizational context, and the development context.

References

Brinkkemper, S. (1996) Method Engineering: Engineering of Information Systems Development Methods and Tools, Journal of Information and Software Technology.

Brinkkemper, S., Saeki, M. and Harmsen, F. (1998) Assembly Techniques for Method Engineering, CAiSE'98 (Eds, Pernici, B. and Thanos, C.), LNCS 1413, pp. 381-400.

Cubetto (2009) Specification, Semture GmbH, Dresden, http://www.semture.de/cubetto.

Erl, T. (2005) Service-oriented architecture: concepts, technology, and design, Prentice Hall PTR.

Esswein, W. (1993) Das Rollenmodell der Organisation: Die Berücksichtigung aufbauorganisatorischer Regeln in Unternehmensmodellen, In Wirtschaftsinformatik 35, pp. 551-561.

Ferraiolo, D. F. and Kuhn, D. R. (1992) Role Based Access Control, 15th National Computer Security Conference.

Frank, U. (1999) Conceptual Modelling as the Core of the Information System Discipline - Perspectives and Epistemological Challenges, Proceedings of the 5th America's Conference on Information Systems, AMCIS'99 (Eds, Haseman, D. W., Nazareth, D. and Goodhue, D.) AIS, Milwaukee, pp. 695-697.

Juhrisch, M. (2010) Richtlinien für die modellgetriebene Integration serviceorientierter Architekturen in Analysemodellen, Logos-Verlag, Berlin, Germany.

Juhrisch, M., Dietz, G., Weller, J. and Esswein, W. (2009) Towards Business Driven Web Service Authorization – Project Experiences in German University Administrations, Proceedings of the 15th Americas Conference on Information Systems (AMCIS'09)

Juhrisch, M., Dietz, G., Esswein, W. (2009): Perspectives on Semantic Business Process Modeling – A Generic Approach. Proceedings of the 13th Pacific Asia Conference on Information Systems (PACIS 2009)

Juhrisch, M., Weller, J. and Esswein, W. (2010) A Model-Driven Framework for Business IT Alignment, In International Journal of Internet and Enterprise Management, 6 (3).

Juhrisch, M. and Dietz, G. (2010) Constraints in Conceptual Modelling – Outlining an Approach to Business Driven Web Service Composition, In International Journal of Internet and Enterprise Management, 6 (3).

Scheer, A.-W. (2000) ARIS - Business Process Modeling, Springer, Berlin.

Walsh, A. E. (2002) UDDI, SOAP, and WSDL: The Web Services Specification Reference Book, Prentice Hall Professional Technical Reference, New Jersey.

Wand, Y. and Weber, R. (2002) Research Commentary: Information Systems and Conceptual Modeling - A Research Agenda, Information System Research, 13 (4), pp. 363-376.

Weller, J., Juhrisch, M. and Esswein, W. (2006) Towards using visual process models to control enterprise systems functionalities, Int. J. Networking and Virtual Organisations, 3 (4), pp. 412-424.