Using Obfuscating Transformations for Supporting the Sharing and Analysis of Conceptual Models

Dr. Hans-Georg Fill

University of Vienna, Research Group Knowledge Engineering, 1210 Vienna, AT, E-Mail: fill@dke.univie.ac.at

Abstract

Several initiatives have been started that promote the collaborative creation, sharing and analysis of conceptual models. In order to maintain confidentiality and protect intellectual property, sensitive data has to be removed from the models or at least sufficiently abstracted. We derive and analyze four types of obfuscating transformations for conceptual models that have been inspired by existing methods from the area of source code obfuscation and privacy preserving data mining. Finally, a use case from the area of banking is presented to illustrate the concrete application of the transformations.

1 Introduction

In the last years, several initiatives have been started that promote the collaborative creation, sharing and analysis of conceptual models [5][11][14]. The common goal thereby is to apply aspects of the world of open source to conceptual modeling and thus involve a large number of participants in such public processes [11]. Similar to the benefits of open source, the advantages are the provision of large repositories of high quality models that can be freely accessed for learning, research and development purposes, the establishment of a community of practitioners and academics for exchanging and combining their expertise, and the access to IT concepts and tools for realizing these ideas. Apart from the technical challenges, such as the development of collaboration tools and according organizational concepts for bringing this to life, a central aspect of sharing models with a large public audience is the dealing with intellectual property and confidential information.

In particular, models that are used in real-life business settings typically contain large amounts of confidential information that should not be disclosed. This concerns work practices, business relationships or other information that could potentially harm an organization by making it available to competitors. On the other hand, not all parts of the contained information are potentially harmful when they are disclosed, but may even serve in helping businesses to improve their operations by enabling comparisons and receiving feedback [15]. In addition, the public access to the models may even lead to further business opportunities by documenting how customers and suppliers may optimally interact with a business and exchange goods and services. From a research perspective, the access to such business data is very valuable for gaining insights into actual operations and for developing and testing new concepts, even if only certain parts of the information are available.

It thus seems worthwhile to investigate techniques for protecting confidential information in conceptual models while at the same time allowing for certain types of analyses. For this purpose it can be reverted to the areas of source code obfuscation and privacy preserving data mining (PPDM) that have discussed similar approaches in the past. However, to adapt these approaches for the field of conceptual modeling, the specific characteristics of conceptual models have to be taken into account. We will therefore briefly outline the basic foundations in regard to the components of conceptual modeling methods, PPDM and code obfuscation techniques in section 2. This will permit us to derive selected representation, structural, data, and semantic information transformations for obfuscating information in conceptual models in section 3 and discuss them in a use case in section 4. The paper will be concluded by discussing work related to our approach in section 5 and by giving an outlook on potential future developments in section 6.

2 Foundations

In order to describe the transformations for obfuscating information in conceptual models we will give a short overview of the particular characteristics of conceptual modeling methods and existing approaches for code obfuscation and privacy preserving data mining.

2.1 Components of Conceptual Modeling Methods

For clarifying the terms we will use in the following we will base our discussion on a framework that has been proposed by Karagiannis and Kühn [12]. In their framework, a modeling method is composed of a modeling technique and mechanisms and algorithms. The modeling technique is further refined by a modeling language and a modeling procedure that defines the steps for applying the modeling language and the results. The modeling language is composed of a syntax, semantics, and notation. Thereby, the syntax defines the grammar of the modeling language and the semantics the meaning. This is accomplished by mapping the elements of the syntax to a semantic schema, which may be formally specified or given in natural language. The notation takes into account the semantics and defines the visualization of the modeling language. Mechanisms and algorithms can be used by the modeling procedure and related to the modeling language. They can be either generic, i.e. applicable to all modeling languages, specific, i.e. applicable only to certain modeling languages or hybrid in the sense of configurable for certain classes of modeling languages.

Although conceptual models are typically not directed towards the processing by machines and thus usually do not provide formal semantics, also a formal syntax permits to construct algorithms that can process the contained information. Especially in fields such as business process modeling or enterprise architecture management the application of analysis and simulation algorithms is highly valuable [10]. When investigating techniques to hide or remove confidential information, it is therefore desirable to preserve at least certain parts of the model information so that such algorithms can still be executed. For example, to run a simulation algorithm that calculates the average cycle time of a process only the structure including the contained activities, splits, joins etc. together with the attached time attributes are required [10]. At the same time it is however not necessary to disclose information about the labels and descriptions of the contained activities.

2.2 Privacy Preserving Data Mining and Source Code Obfuscation

For preserving data privacy and intellectual property in terms of source code, a wide range of methodologies have been proposed in the research literature [9][4]. In particular there are two main fields that discuss approaches that can serve as input for protecting information in conceptual models: *privacy preserving data mining* and *source code obfuscation*. In privacy preserving data mining (PPDM) it has been distinguished between methodologies that protect sensitive data itself in the mining process and methodologies that protect the results after applying data mining [9]. Both types of methodologies protect information by sanitizing the original datasets in a way that sensitive information is shielded or certain associations in the datasets cannot be derived. At the same time it is aimed for the preservation of properties that are similar to the original datasets as well as reasonably accurate data mining results [9]. In line with the first type it can also be reverted to methods of *secure multi-party computation* where computation tasks are conducted that are based on the private, i.e. non-disclosed input of collaborators [6].

Similarly to PPDM, source code obfuscation discusses ways to convert the source code of a program into one that has the same observable behavior, i.e. that produces the same output based on the same input but which is harder to understand and reverse engineer [16]. The obfuscating transformations used for this purpose have been classified based on the kind of information they target [4][16]: *layout transformations* that change the formatting of source code and identifiers of variables, *control transformations* that modify the representation of the control flow of a program to hide its functioning, and *data transformations* that target the data structures of an application.

3 Obfuscating Transformations for Conceptual Models

With the foundations outlined above we can now investigate in detail how such approaches for protecting information can be applied to conceptual models. Therefore we will derive transformations that target the *representation*, *structure*, *data*, and *semantic information* of conceptual models and illustrate those using concrete examples. These transformations will be denoted as *obfuscating transformations for conceptual models* as they aim primarily at modifying information rather than solely removing information.

In contrast to data sets, the nature of conceptual models requires that even if certain information is hidden or not accessible any more, the models are still of value in regard to human understanding and communication. It is therefore possible to change the contained information in a way that is not suitable for machine processing but that is still beneficial for humans. Therefore, the methodologies of secure multi-party computation that are based on a strict non-disclosure of all involved inputs and purely directed towards machine processing do not seem to be suitable [6]. They may however be a good choice for comparing the results of algorithms that have been executed on conceptual models.

For maintaining the benefits of conceptual models, the main structure and the contained semantic information should be preserved as much as possible. Other aspects such as numerical details or simulation-relevant data may be - despite their great value for analyses - of less importance.

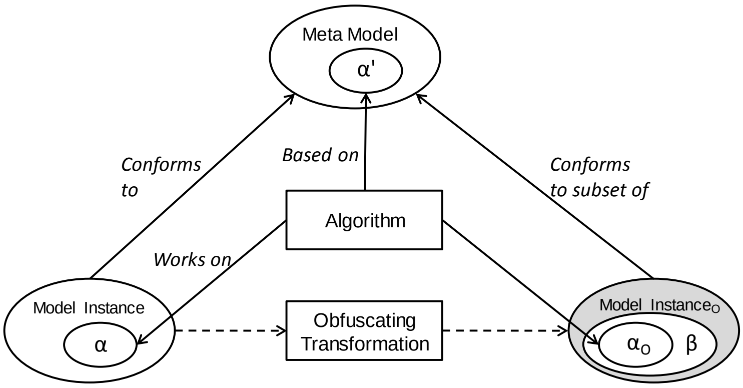


Figure 1: Concept for Applying Obfuscating Transformations to Conceptual Models

In addition, control transformations that add several levels of complexity to disguise the actual functioning of a program, neither seem applicable for conceptual models as this would considerably limit the purpose of conceptual models in terms of understanding.

To describe in detail how obfuscating transformations are applied to conceptual models we built upon the concept shown in Figure 1: Thereby, we use the notion of *meta model* for the representation of a modeling language and *model instance* for a concrete realization of a conceptual model by using this language. When designing algorithms for arbitrary model instances, they have to be based on a subset of the meta model - denoted as α’ in Figure 1. This subset then corresponds to the subset α in the model instance. When an *obfuscating transformation* is applied to the model instance, a new instance is created, denoted by subset “O”. In our approach this new instance still conforms to a subset of the meta model in order to preserve the understandability but may contain information that has been modified in regard to the original. However, not the full amount of information may be available as indicated by the grey area on the right hand side of Figure 1. The amount of information that is accessible is denoted by β. A subset of β, which is indicated as $α_O$, is the information required by the algorithm.

For deriving suitable obfuscating transformations we investigated existing transformations in the areas of privacy preserving data mining and source code obfuscation. Thereby we focused on one-way and highly resilient transformations [4], i.e. transformations where it is not possible or very unlikely to discover the original information. From this we derived potential candidate solutions that matched the aspects described above for conceptual models. In the next step we adapted these candidate solutions for conceptual models.

3.1 Representation Transformations

The first type of transformations is analogous to the layout transformations used in source code obfuscation. There they are used for example to scramble the identifiers of variables, change the formatting of the source or remove comments.

For the domain of conceptual models we derived two adaptations of this type: The first is the trivial solution for obfuscating information, which consists of *hiding or removing* certain elements in the obfuscated model instance. This is illustrated Figure 2 by using a conceptual model type for representing organizational structures. On the left hand side the original model is shown that represents the organizational unit "Customer service department", staff assigned to this department and their organizational roles. For the obfuscated model on the right hand side one role definition and a representation of a staff member have been removed. Although this is a simple example, it illustrates how the main structure of the model is preserved while some analyses such as any capacity simulations involving the role "Customer agent" cannot be performed anymore. However, simulations only involving the second role are still feasible.

The second type of representation transformation is one that abstracts the structure of the original model. The identifiers and the types of the elements are removed and only the elements and their relations are shown. In Figure 3 this is illustrated by a model in BPMN notation. Thereby it becomes obvious that after the application of such a transformation only a very basic structure of the original model is preserved. Possible analyses of such an abstracted model would comprise for example complexity measures such as the number of nodes and edges or distance measures.

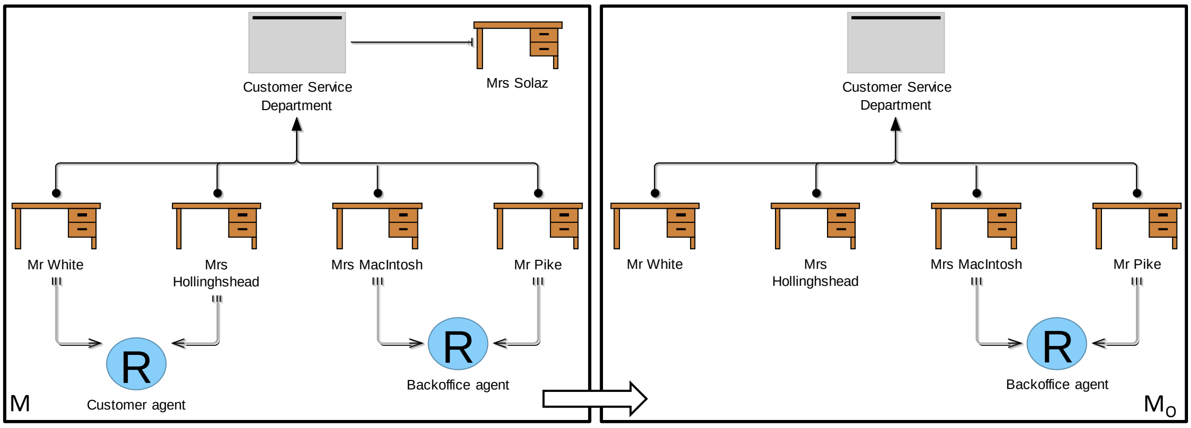


Figure 2: Illustration of an Information Hiding Transformation

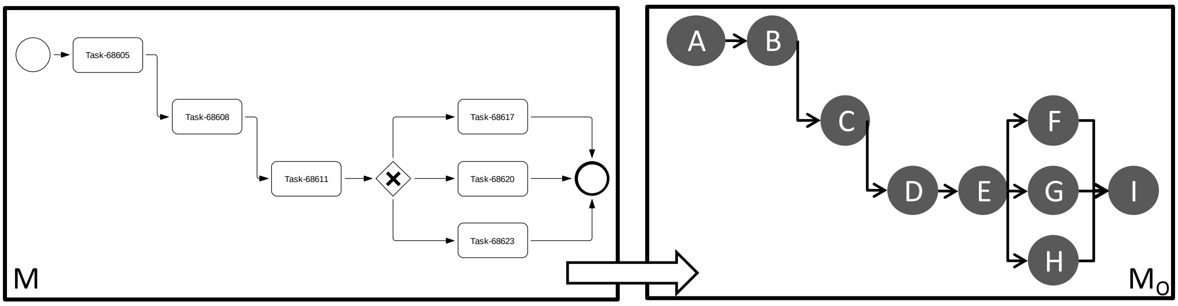


Figure 3: Illustration of an Abstraction Transformation

3.2 Structural Transformations

The second type of transformations has been derived from control transformations for obfuscating source code. They are targeted towards changing the structure and complexity of programs in order to hide their actual functioning. For conceptual models we adapted this in the way that several elements are condensed to one element. This is illustrated in Figure 4 by condensing two actors to one element and renaming it to "Multiple Actors". Additionally, the new element is marked by a star to indicate that it stands for more elements of the same type. In this way for example the actual number of resources can be hidden.

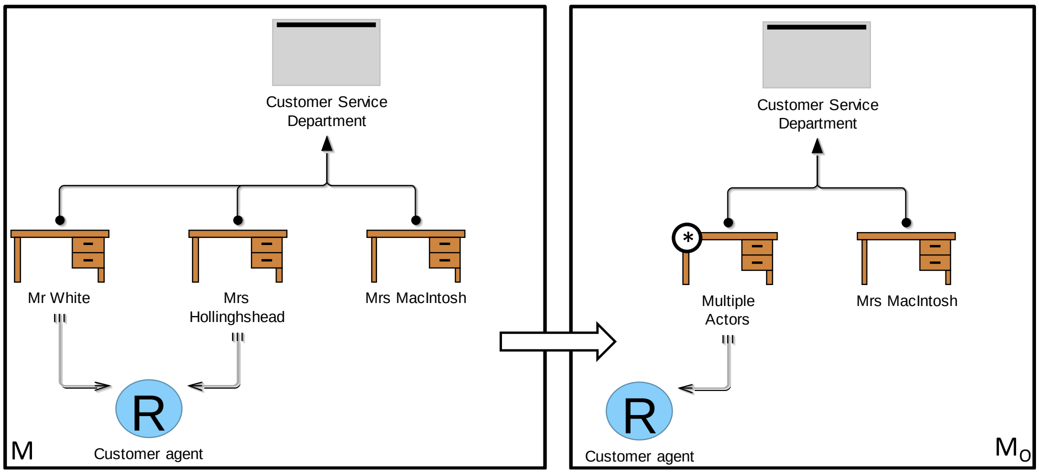


Figure 4: Illustration of a Structural Transformation

Further transformations concerning structural properties can be derived similarly to code obfuscations where method and function calls are obfuscated. This applies in particular to conceptual model types that make use of *nesting*. Thereby, conceptual models can be embedded in each other by defining links between different model instances. As an example we refer again to BPMN where parts of processes can be embedded by using calls to sub processes – see Figure 5: Here, a call to a sub-process is used to obfuscate the execution of the tasks “C” and “D” in the original model M. The obfuscated model M_O only shows the call to the sub-process “Sub-Process 1”. The contents of this sub-process are not disclosed and are preserved internally in the model “$M_{\text{Sub-Process 1}}$”. Such nesting could also be used to realize another obfuscation transformation that is analogous to the obfuscation of method calls in the area of source code: Identical parts of a model can be replaced by calls to different submodels that in turn point to the same model parts. If the sub-models are not disclosed it cannot be assessed that the seemingly different parts are actually the same.

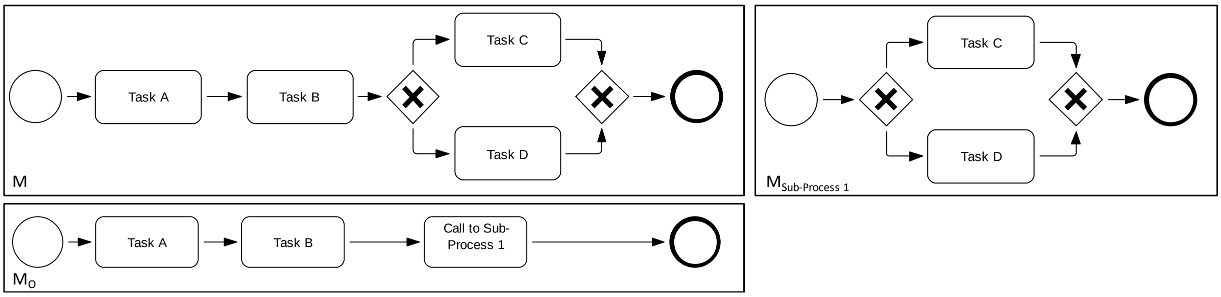


Figure 5: Illustration of a Transformation Using Nesting

3.3 Data Transformations

Based on transformations used in privacy preserving data mining, also the data of labels and attributes of conceptual models can be obfuscated. This can be done for example by scrambling the labels in the models so that their original meaning is not any more accessible. As an example we used again the organizational model where the labels have been replaced by random alphanumeric identifiers – see Figure 7. Although this is similar to the representational transformations discussed before, the scrambling can also be applied only to parts of the original model. Thus, some information can be preserved for analyses while other information is rendered inaccessible.

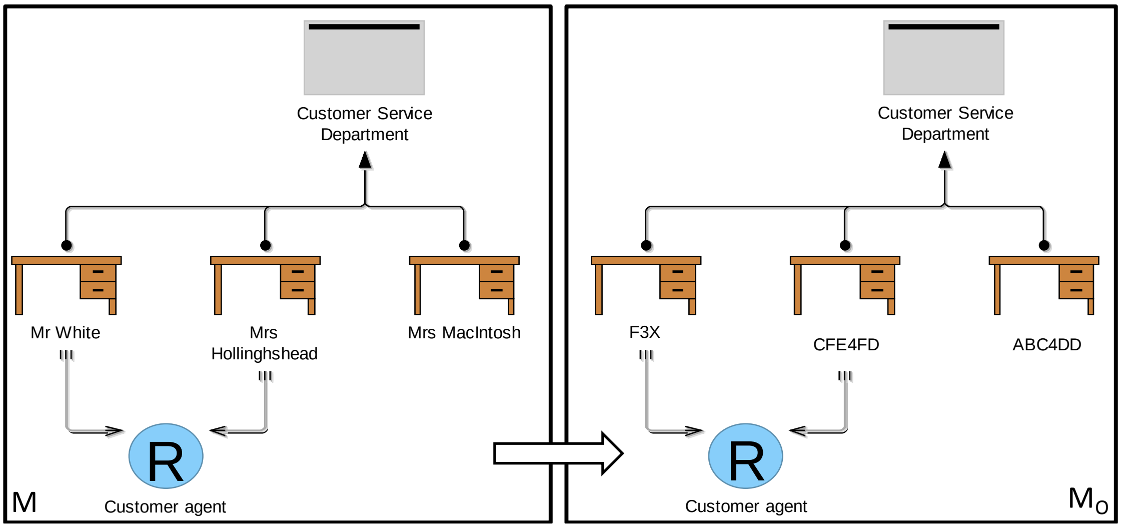
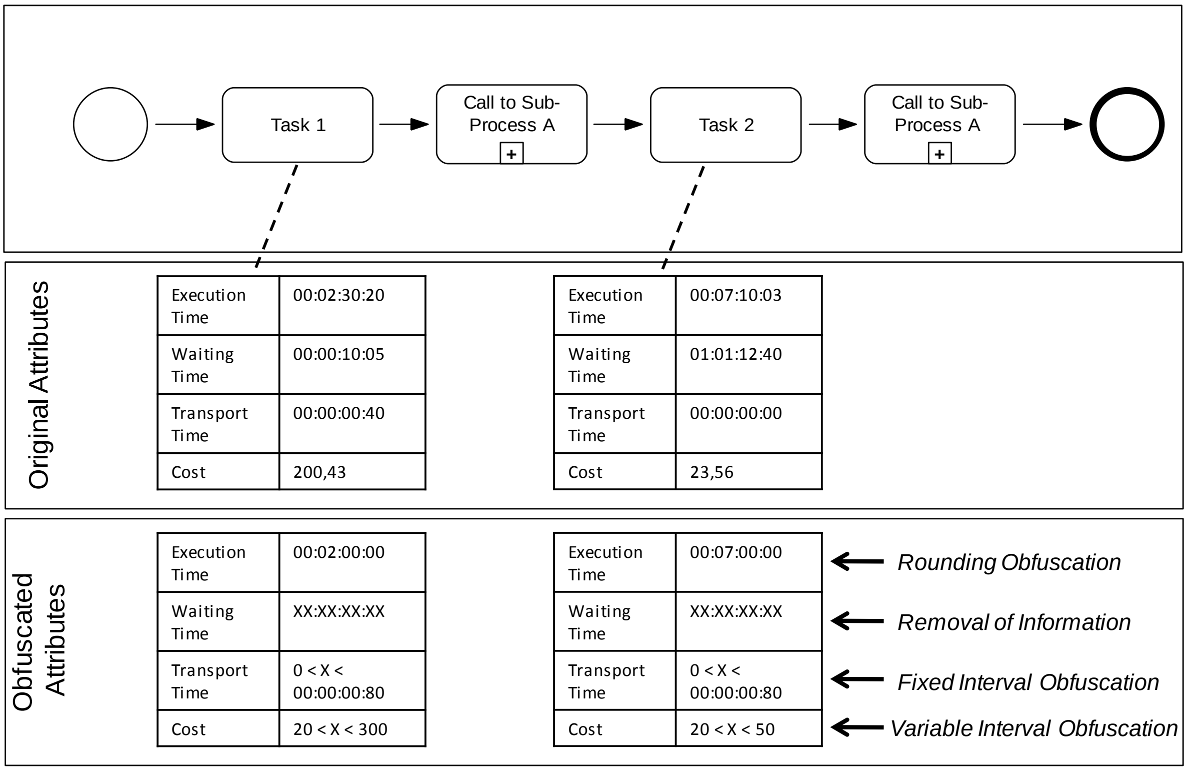


Figure 7: Illustration of an Attribute Scrambling Transformation



Original Attributes

Execution Time	00:02:30:20
Waiting Time	00:00:10:05
Transport Time	00:00:00:40
Cost	200,43

Execution Time	00:07:10:03
Waiting Time	01:01:12:40
Transport Time	00:00:00:00
Cost	23,56

Obfuscated Attributes

Execution Time	00:02:00:00
Waiting Time	XX:XX:XX:XX
Transport Time	0 < X < 00:00:00:80
Cost	20 < X < 300

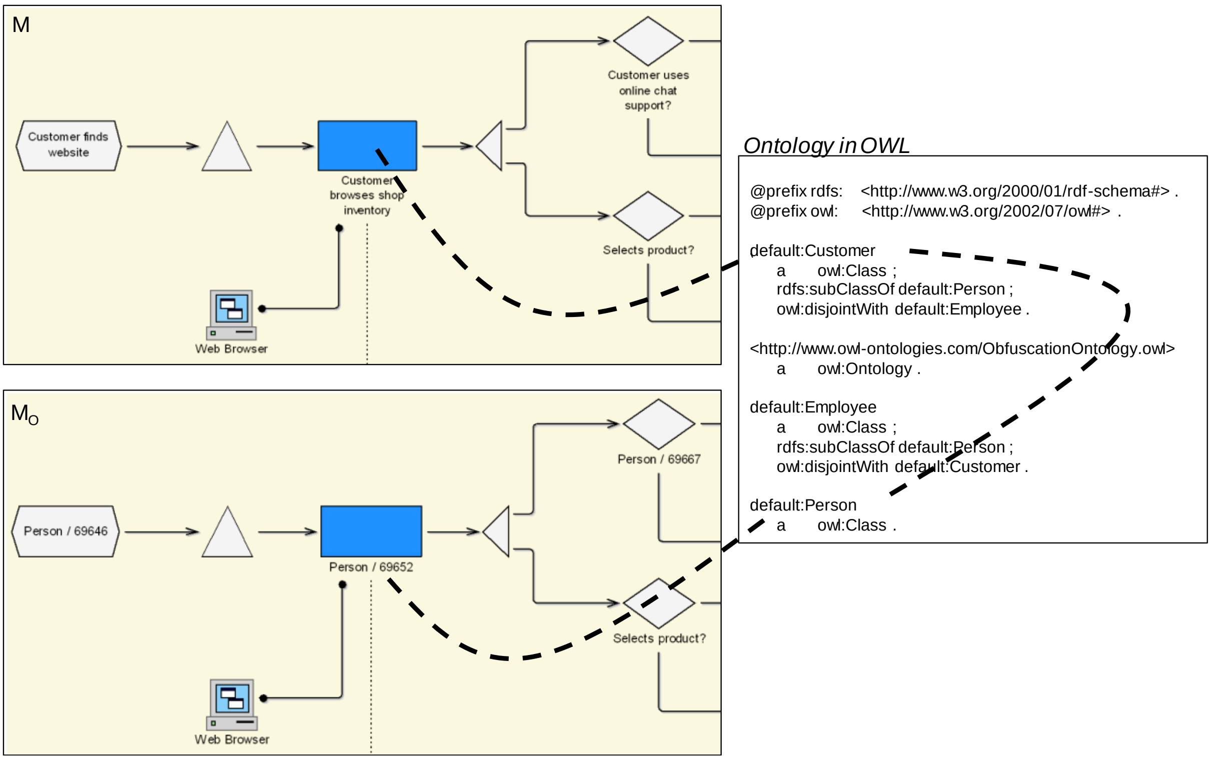
Execution Time	00:07:00:00
Waiting Time	XX:XX:XX:XX
Transport Time	0 < X < 00:00:00:80
Cost	20 < X < 50

Figure 6: Illustration of Numerical Value Transformations

When obfuscating the values of numerical attributes in conceptual models, the existing transformations used in privacy preserving data mining can be directly applied. This concerns the use of *rounding* and *fixed and variable interval* obfuscation as well as the removal of attribute values as depicted in Figure 6.

3.4 Semantic Obfuscation Transformation

In addition to the previously described transformations we developed another type of transformation that targets the semantic information contained in the conceptual models. It is based upon the notion of *ontological instantiations* in the sense of annotating models with domain concepts from a formal ontology as described for example by [8]. Through using specific properties of formal ontologies, additional ways of obfuscating transformations can be added. As illustrated in Figure 8 the use of a subsumption hierarchy as it can be formally defined by using the web ontology language OWL, information can be anonymized by replacing attribute or label values with less specific values in the sense of generalization. As a consequence, parts of the semantic information expressed in the original conceptual model instance can be preserved and processed without disclosing the exact details.



```
@prefix rdfs:   <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl:    <http://www.w3.org/2002/07/owl#> .

default:Customer
    a      owl:Class ;
    rdfs:subClassOf default:Person ;
    owl:disjointWith default:Employee .

<http://www.owl-ontologies.com/ObfuscationOntology.owl>
    a      owl:Ontology .

default:Employee
    a      owl:Class ;
    rdfs:subClassOf default:Person ;
    owl:disjointWith default:Customer .

default:Person
    a      owl:Class .
```

Figure 8: Illustration of a Semantic Obfuscation Transformation

4 Use Case

To illustrate the benefits of using obfuscation transformation for conceptual models and to evaluate the transformations we will describe in the following a use case from the area of banking. For this purpose we revert to the description of an account opening process that is publicly available[1]. The process description has been first represented in the form of a busi-

[1] The process description was elaborated and published by Gerardo Palmisano for the Hypothekarbank Lenzburg on the platform http://www.lernender.ch last access 18-09-2011

ness process model type in ADONIS® notation and then complemented with fictitious data for the attribute values of IT systems, activity costs and execution time as well as an organizational structure. The original conception is shown by the M^1 and M^2 models in Figure 9 and the resulting models after the application of obfuscating transformation by M^1_{OB} and M^2_{OB}. As shown in M^1, the activity “Acquire customer data (personal data, address, ID)” contains two attributes: “Execution time” and “Average cost”. The activity “Create new customer profile and enter data” uses the IT system “Finstar”. This system is detailed by the two attributes “Host name” and “IP-Address”. In parallel to these activities, the activity “Customer fills out application form W4801” is conducted. Furthermore, the model in M^2 shows an organizational model for the unit “Customer service center”. The unit is led by “Mrs White” who has the

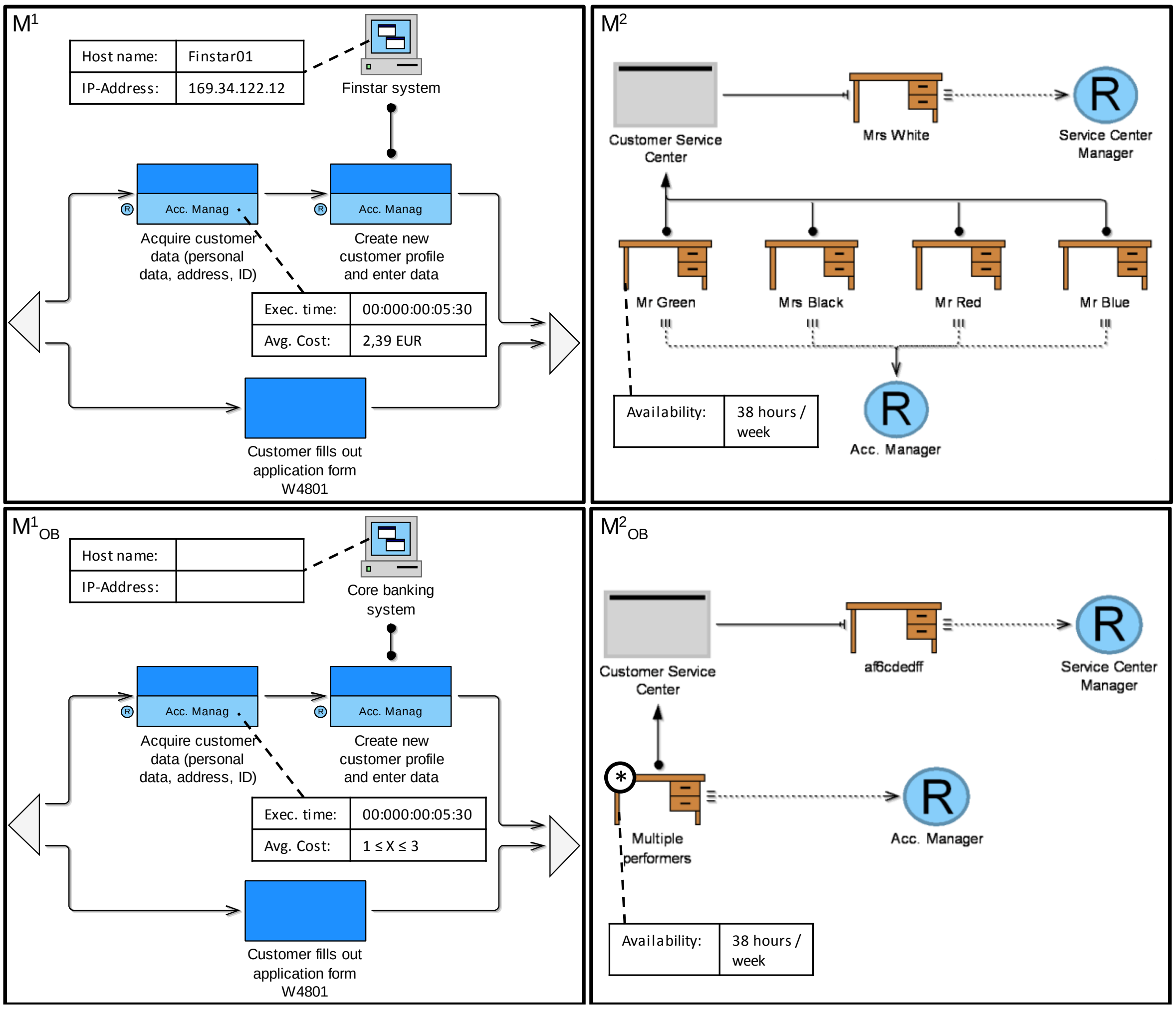


Figure 9: Application of Obfuscating Transformations to Parts of a Banking Business Process

role “Service Center Manager”. The further members of the unit with the role “Account Manager” (“Acc. Manager”) are shown below. The role “Account Manager” is responsible for the top two activities in the process. For each performer the “Availability” in terms of hours per week is assigned as an attribute.

As a basis for applying the obfuscating transformations we identified three issues that can potentially lead to requirements in altering process information before sharing it:

- Competitive issues, i.e. information that should not be disclosed to competitors

- Security issues, i.e. information that potentially permits to access non-public systems or could facilitate such access
- Privacy issues, i.e. information that concerns personal data

In the process at hand potential competitive issues could concern the used IT systems, the number of employees for particular roles in the process or the details about the activity costs. Security issues could concern the details of the involved IT systems such as host name and IP address. Privacy issues could come up in regard to the names of the employees and their detailed organizational roles and status.

Therefore, the following obfuscating transformations were applied – see M^1_{OB} and M^2_{OB}: To abstract the name of the used IT system a semantic obfuscation transformation was applied to these elements. For this purpose, the IT system elements have to be annotated with concepts from a subsumption hierarchy that contains "Finstar" as a specific concept and "Core banking system" as a more general concept. Additionally, the values for "Host name" and "IP-Address" were completely removed. To prevent competitors from assessing the capacities of the "Customer Service Center", the performers of the role "Account Manager" were aggregated to an element named "Multiple Performers". The name of the manager of the organizational unit was scrambled and the values of the cost attribute of the process activities were obfuscated using variable interval obfuscation.

After the application of the transformations it can now be discussed what kinds of insights can still be gained by the obfuscated models and which analyses can still be applied. As becomes apparent by the model shown in M^1_{OB}, the core structure of the process is still preserved here. It can thus be used to learn about the core functioning of the process such as the sequence and structure of the activities and in which parts of the process IT systems are used. As the execution time is maintained in its original form, algorithmic analyses such as the determination of longest and shortest paths can still be conducted. To a limited degree, also the costs of the process can be investigated. However, due to the variable interval obfuscation, only very rough estimates can be calculated for this. Concerning the organizational environment, it can still be concluded, which organizational roles are responsible for which parts of the process. Therefore, calculations such as a capacity analysis per role can be done by reverting to the execution time and availability attributes. It can however not be assessed what capacities are available in the organization depicted in the models.

Although considerable parts of the original model information have been removed or obfuscated, the resulting models still provide a lot of value. Based on the individual requirements of an organization in regard to competitive, security and privacy issues it has to be chosen which of the obfuscating transformations suits best and which types of analyses thus can be safely permitted.

5 Related Work

To the best of our knowledge the application of source code obfuscation techniques for the purpose of promoting the sharing of conceptual models has so far not been discussed, although some of the investigated transformations have been discussed previously in other contexts. Apart from the already mentioned approaches in source code obfuscation and privacy preserving data mining, we can relate the concept to two prior approaches: The first concerns a theoretical approach for the extraction, removal and desensitization of sensitive concepts in ontologies [13]. However, this approach refers to formal knowledge representation models and is targeted towards erasing information rather than obfuscating it and still permitting analyses. Another related approach has been described in the area of databases for obfuscating data schemas [15]. It is however directed towards the processing of database queries and replaces potential sensitive data by randomly chosen synonyms that are processed by a mediator so that no parts of the original models are actually shared but can only be queried.

From the area of business process management the approaches of process inheritance and nesting, view-based process visualizations and semantic process model abstraction can be related to some of the transformations shown here. Process inheritance deals with the abstraction and specialization of processes in order to ease the definition of new processes based on already existing representations [19]. This is related to the well-known techniques in object orientation but also takes into account the behavioral properties of a process. Similarly, the areas of process nesting and view-based process visualization developed techniques similar to the ones we have shown for removing parts of models and segmenting complex models into parts of less complex, respectively aggregated ones [19][2]. In contrast to these approaches, we state, however, that such techniques are applicable not only to process models but arbitrary types of conceptual models. This also applies to the techniques of semantic obfuscation and semantic abstraction of process models that have been discussed previously [7][17]. Regarding the semantic abstraction for process models presented in [17], the semantic obfuscation transformation does not revert to the structural semantic information but rather to the semantics expressed by the labels of the modeling elements and their annotations. Thereby, we can preserve all parts of the original structure for analyses and hide confidential information in the labels at the same time.

6 Conclusion and Outlook

In this paper we derived a number of transformations for obfuscating information in conceptual models. The benefit of these transformations is seen in supporting the sharing of conceptual models while at the same time enabling certain kinds of analyses of the original model information. As a next step, it is planned to detail the transformations using mathematical formalisms in order to implement them using modeling tools. In particular, it will be investigated which of the described functionalities can be realized in a generic way so that they are directly applicable to arbitrary types of modeling languages. On top of that it will then be possible to conduct experiments to further evaluate the proposed obfuscating transformations in regard to user acceptance and their suitability for concrete business use cases.

7 Acknowledgement

The work on this paper has been partly supported by the Austrian Science Fund in the course of a research project financed by an Erwin-Schrödinger Fellowship, project number J3028-N23.

8 References

[1] Bakken, D.E. et al. (2004): Data Obfuscation: Anonymization and Desensitization of Usable Data Sets. IEEE Security and Privacy, 2(6):34–41.

[2] Bobrik, R.; Reichert, M.; Bauer, T. (2007): View-Based Process Visualization. In: BPM'2007, 88-95. Springer.

[3] Bussler, C. (2002): Process Inheritance. In: Banks Pidduck, A. et al.: CAiSE 2002. Springer, 701-505.

[4] Collberg, C.; Thomborson, C.; Low, D. (1997): A Taxonomy of Obfuscating Transformations. Technical Report 148, University of Auckland, New Zealand. https://researchspace.auckland.ac.nz/bitstream/handle/2292/3491/TR148.pdf?sequence=2. Accessed on 30-03-2011.

[5] Decker, G.; Overdick, H.; Weske, M. (2008): Oryx – Sharing Conceptual Models on the Web. In: Proceedings of Conceptual Modeling - ER 2008, 536-537. Springer.

[6] Du, W.; Atallah, M. J. (2002): Secure Multi-Party Computation Problems and Their Applications: A Review and Open Problems. In: NSPW'01. ACM.

[7] Fill, H.-G. (2011): Using Semantically Annotated Models for Supporting Business Process Benchmarking. In: BIR'2011, 29-43. Riga, Latvia. Springer.

[8] Gailly, F.; Espana, S.; Poels G.; Pastor, O. (2008): Integrating Business Domain Ontologies with Early Requirements Modelling. In: ER Workshops 2008, 282–291. Springer.

[9] Gkoulas-Divanis, A.; Verykios, V.S. (2009): An Overview of Privacy Preserving Data Mining. ACM Crossroads, 15(4):23–26.

[10] Herbst, J.; Junginger, S.; Kühn, H. (1997): Simulation in Financial Services with the Business Process Management System ADONIS. In: ESS97. Soc. for Computer Simulation.

[11] Karagiannis, D.; Grossmann, W.; Höfferer, P. (2008): Open Model Initiative - A Feasibility Study. http://cms.dke.univie.ac.at/uploads/media/Open_Models_Feasibility_Study_SEPT_2008.pdf. Accessed on 20-09-2011.

[12] Karagiannis, D.; Kühn, H. (2002): Metamodeling Platforms. In: Bauknecht, K.; Tjoa, A.M.; Quirchmayr, G. (eds.). 3rd Conference EC-Web 2002 Dexa 2002. Springer.

[13] Kaushik, S.; Wijesekera, D.; Ammann, P. (2005): Policy Based Dissemination of Partial Web Ontologies. In: SWS'05, Fairfax, Virginia. ACM.

[14] Koch, S.; Strecker, S.; Frank, U. (2006): Conceptual Modelling as a New Entry in the Bazaar: The Open Model Approach. In: Open Source Systems, 9–20. IFIP.

[15] Mitra, P.; Pan C.-C.; Liu, P.; Atluri, V. (2006): Privacy-preserving Semantic Interoperation and Access Control of Heterogeneous Databases. In: ASIACCS 06, Taipei, Taiwan.

[16] Preda, M.D.; Giacobazzi, R. (2005): Control Code Obfuscation by Abstract Interpretation. In: 3rd IEEE Conference on Software Engineering and Formal Methods. IEEE.

[17] Smirnov, S.; Rejiers, H.A.; Weske, M. (2011): A Semantic Approach for Business Process Model Abstraction. In: CAiSE'2011, 497-511. London. Springer.

[18] Yang, G. (2004): Process library. Data and Knowledge Engineering (50). 35-62.