ABSTRACTION AND TRANSPARENCY IN META MODELING

Hans-Georg Fill

Privatdozent, University of Vienna, Research Group Knowledge Engineering
Währingerstraße 29, 1090 Wien, AT
hans-georg.fill@univie.ac.at; http://homepage.dke.univie.ac.at/fill

Keywords: ***Modeling Methods, Abstraction, Transparency***

Abstract*:* *Modeling methods have been traditionally used for many aspects in legal informatics. Several of them feature visual notations to enable the understanding of complex relationships and ease human interaction with the corresponding models. Some also put a focus on the interpretation by machines and according processing functionalities. In this paper we regard abstraction and transparency as two functions in meta modeling, i.e. in the conceptualization of modeling methods. This permits us to highlight the nature of these two aspects and thus provide a basis for discussing the specific concepts of extraction and obfuscation, which play an important role for processing information in multi-sensory law and from a legal perspective in general. In this way applications for legal informatics can be analyzed and designed in regard to their abstraction and transparency capabilities.*

1. Introduction

Today, models are used for a variety of purposes in many areas of legal informatics. Based on their core property of reducing the complexity of real or virtual systems, they facilitate interactions both for human actors as well as machine-based agents – see also the discussion on complexity reduction through models by [Fiedler (2008)]. Examples for models include but are not limited to: models for verifying the application of legal norms, e.g. [Kahlig (2011)], models in the form of rules for the automatic classification of legal documents, e.g. [Kienreich et al. (2013)], or, models as formal ontologies for the automatic verification of service descriptions against legal information, e.g. [Wacker and Peitz (2012)].

Depending on the underlying *modeling method* that is used for their creation, models may be depicted in visual notation and some or all of their constituents may be described formally in order to enable an unambiguous, intersubjective interpretation as well as machine-processing. The modeling method thus plays an essential role in determining the value of the resulting models and therefore needs to be carefully chosen and/or designed in order to achieve a maximum benefit. The design of a modeling method not only encompasses the specification of a modeling language, the definition of a modeling procedure, i.e. the rules how to apply a modeling language, and mechanisms and algorithms that operate on the modeling language, c.f. [Karagiannis and Kühn 2002]. It also needs to be taken into account in the early phases of the design if and how a modeling method should be technically conceptualized and implemented.

Although these technical aspects have so far in many cases been disregarded for the design of modeling methods, they are essential due to the following aspects: primarily, the interaction with modeling methods and models is today almost exclusively accomplished using information technology. This stems from the requirement of being able to create, share, and process models

electronically as well as for handling large amounts of models and their contained information at all. As a consequence, it needs to be considered how the interaction of human actors and machine-based agents with a modeling method and the models shall be enabled.

Based on the diversity of modeling methods and the different purposes for which they are used, the ways of interaction may differ considerably. Whereas in one case it may be beneficial to give a user a certain degree of freedom in creating models without continuously checking if all constraints are satisfied, in another case, e.g. when a machine-based agent shall create a model, the strict enforcement of checking routines may be crucial to ensure formal consistency. However, such decisions have an impact on the design of the modeling method as, e.g. in the first case, constraints do not need to be formally described whereas in the second case machine-processable descriptions of the constraints are inevitable. Similar considerations need to be taken for example in regard to the handling, i.e. the storage and efficient processing of model information. If the content of the models is to be used for further processing, the choice of appropriate technical formats or structures, e.g. in the form of an optimized database schema, or a particular syntax structure for algorithms is essential. Again, this can lead to adaptations of the modeling method that need to be synchronized with the interaction requirements.

It thus becomes obvious that the design of modeling methods in order to use them in the form of according IT-based tools is a challenging undertaking. At the Research Group Knowledge Engineering at the University of Vienna we currently investigate the underlying theoretical and technical foundations of these activities under the term *meta modeling*. Meta modeling thereby stands for all activities related to the conceptualization of modeling methods which includes their creation, design, formalization, development, and deployment[1]. The ultimate goal of meta modeling in our understanding is to provide usable modeling methods in the form of executable software tools. These tools may comprise functionalities such as visual model editors, simulation engines, transformation and analysis tools, etc. that are specifically adapted for a particular modeling method.

In the following we will regard *abstraction* and *transparency* as two functions in meta modeling. Whereas abstraction refers to the identification of terms and ideas on a meta level [Lachmayer and Schweighofer (2013)], transparency refers to the traceability, visibility, availability, and verifiability of information, c.f. for example [Tscheliesnig and Weinzettl (2013)]. The goal of this discussion is on the one hand to contribute to a further understanding of these functions and to lay the basis for discussing additional theoretical concepts in the context of legal informatics and more specifically in multi-sensory law that build upon them. On the other hand, also practical implications for conceiving and realizing new applications in multi-sensory law and for services that process legal information can be derived in this way. In particular we will apply the insights gained by this discussion to the concepts of extraction and obfuscation, which play an important role in many applications of legal informatics.

The remainder of the paper is structured as follows: in section two we will provide the reader with some foundations regarding our view on meta modeling. In section three we will discuss how we can characterize abstraction and transparency in meta modeling and show in section four how these findings can be applied to the concepts of extraction and obfuscation. The paper will be concluded with a summary and an outlook on further research activities.

[1] For further information see the website of the Open Models Initiative Laboratory www.omilab.org.

2. Foundations

In this section we will clarify some terms in the context of modeling and meta modeling in order to achieve a common understanding. In general model theory, a model is characterized by three aspects: *mapping*, *reduction*, and *pragmatic* characteristics [Stachowiak (1973)]. Mapping means that a model has some relation to an original, may it be real or virtual where some attributes of the original are mapped to attributes of a model. Reduction means that not all attributes of the original are present in a model. In this way only relevant attributes are selected that are necessary for achieving the core benefit of a model by reducing complexity. Finally, the pragmatic characteristic states that models are intended for a specific purpose in a specific time period and for distinct subjects.

With this description of models we can now advance to an understanding of *meta models*. In computer science and in particular in artificial intelligence and software engineering, the concept of models has been found particularly suitable to describe many diverse aspects such as for example knowledge structures, rules, software systems or also requirements for the implementation of software systems. Thereby, models are not only used by humans to facilitate their understanding of some complex originals [Mylopoulos (1992)]. They can also serve as input for the processing by machines, e.g. to derive new facts from existing knowledge structures, operate systems by executing rules, generating source code or analyzing and simulating the behavior of requirements of systems.

To realize such functionalities, a focus had to be put on the description of models in machine-processable formats. And, as computer systems today are fundamentally based on languages and in particular formal, languages, also models need to be described in such languages for machines to process them. This leads us to the question which language to use and how this language itself can be described in order to be processable. Of course, we can use any kind of general purpose programming language that is known – or even mathematically proven – to be executable on some computer system. We would then have to use the constructs given by the programming language, i.e. typically some data and control structures, to describe our models and their behavior. However, we would have to do this for any kind of model more or less again from the beginning.

It therefore makes sense to identify recurring patterns in the descriptions of models and try to unify them. For example, we could identify the concept of a ‘circle’ that is used in several models with a similar meaning, e.g. to stand for an ‘event’, but with different labels attached to it, e.g. ‘event X’ and ‘event Y’. Through a unification we can already provide the construct of ‘circle’ as a basic construct in a machine-processable language – this is also denoted as a domain-specific language (DSL), c.f. [Frank (2011)]. Thereby, the concept does not need to be defined from scratch but can be used directly as a first-order concept. Readers familiar with the concepts of object orientation in software engineering or subsumption relations in ontologies will recognize the similarities here, though the goal of the illustrated principles is more general[2] and not exclusively targeted at creating software or inference schemata – although it may serve these purposes too.

The set of all unified concepts that are necessary for creating models for a certain domain or purpose is then what we will call a *meta model*. In visual terms, a meta model resembles something like a map key or a drawing stencil. A meta model however goes beyond just a collection of entities and may also contain for example attributes that specify the details of entities or also certain behaviors of an entity, e.g. how it can be connected with other entities or how it reacts to changes in attribute values. A meta model can therefore also be characterized as a model of a modeling language [Favre (2005)] or as the abstract syntax of a modeling language [Harel and Rumpe

[2] See also the discussion of predicaments and subordination in Aristotle’s Categories.

(2004)]. A formal, mathematical description of the relationships between models and meta models is for example given in [Fill et al. (2012)].

As we already mentioned in the introduction, the creation of models not only depends upon a meta model that specifies the modeling language. As models today are typically created using information technology, several additional aspects need to be taken into account. Besides the specification of the modeling language, which includes the definition of the syntax, semantics and notation of the language, also a modeling procedure and mechanisms and algorithms are part of modeling methods [Karagiannis and Kühn (2002)]. The modeling procedure specifies how the modeling language is to be used to achieve certain results. For this purpose it can be referred to mechanisms and algorithms that define how elements of the modeling language are processed. Whereas algorithms are used to define complex behaviors such as analysis or simulation functionalities, mechanisms typically provide less complex functionalities, e.g. to support a user during the creation of models by recommending suitable elements or for checking constraints.

In case a modeling procedure and/or mechanisms and algorithms are added to a modeling method, the modeling language sometimes needs to be adapted accordingly. For example, the addition of a simulation algorithm may require the existence of certain elements or attributes in a modeling language to be able to generate results [Fill and Karagiannis (2013)]. Similarly, the existence of a specific modeling procedure that requires different views for multiple users may lead to the partitioning of a modeling language into several diagram/model types [Bork and Sinz (2014)].

Regarding the technical realization of a modeling method, it also needs to be considered how human actors and machine-based agents shall interact with the method and how the information contained in the models is to be stored. Again this may impact the design of the modeling method in various ways. E.g. in the earlier mentioned example of different constraint checking behaviors for human actors and machine-based agents, this has to be reflected in the definition of the modeling language, the modeling procedure and the corresponding mechanisms: the modeling language has to contain appropriate formal specifications of the constraints, the modeling procedure has to define in which cases constraints are checked, and the actual constraint checking mechanisms need to be configurable so that either constraints are strictly enforced or for example only evaluated upon a user request.

The storage and technical handling of the model information has similar consequences. Although for example XML formats can be created from several different designs of a modeling language, a specific XML schema may require particular information in the models. If this information is not directly available but for example hidden in free text attributes from where it needs to be separately parsed, this may create unnecessary additional effort that can be avoided by an appropriate design of the modeling language, i.e. during meta modeling. With these foundations we can now turn to the discussion of abstraction and transparency in the context of meta modeling.

3. Abstraction and Transparency as Functions in Meta Modeling

As we saw in the previous section, abstraction in the sense of the identification of terms and ideas on a meta level is an inherent characteristic of meta modeling and modeling. Thereby we witness two relations where abstraction takes place: first, the relation between a model and an original and second, the relation between a modeling method and several originals. Although also the relation between a model and its corresponding modeling method is subject to an abstraction – i.e. from one or more models to their meta model – this kind of abstraction depends on the chosen technical environment or formalism for instantiating models, e.g. as described by the FDMM formalism [Fill et al. (2012)]. The abstraction that can thus be influenced during meta modeling is how the modeling method abstracts from originals. Similarly, during modeling the abstraction can be chosen for a model based on the constructs provided by the meta model and in regard to an original. On the

model level the abstraction from an original can thus only be accomplished in the boundaries set by the modeling method, although the modeler may still retain some freedom if the modeling procedure and the corresponding mechanisms and algorithms permit it. On the meta model level, the abstraction from originals can be freely chosen, only limited by the constraints set forth by a formalism or technical environment for describing the modeling method.

In an attempt to arrive at a more formal notation we suggest to define an abstraction function for modeling methods f_{MM}^{abst} that maps a combination from the set of formalisms FO and originals O to the space of modeling methods where each modeling method j is characterized by a combination of a set of modeling languages $\mathrm{ML_j}$, a set of modeling procedures $\mathrm{MP_j}$ and a set of mechanisms and algorithms $\mathrm{MA_j}$. On the model level we can define a function f_{M}^{abst} as a map from a combination of the components of a modeling method j and the set of originals to the union of the sets of models that belong to the modeling method j.

$$f_{MM}^{abst}: (\mathrm{FO} \times \mathrm{O}) \rightarrow \bigcup_j \left(\mathrm{ML}_j \times \mathrm{MP}_j \times \mathrm{MA}_j\right)$$

$$f_{M}^{abst}: \bigcup_j \left(\mathrm{ML}_j \times \mathrm{MP}_j \times \mathrm{MA}_j \times \mathrm{O}\right) \rightarrow \bigcup_j \mathrm{M}_j$$

With these definitions there are several assumptions involved: first, it is assumed that the formalisms in FO provide all necessary constructs to define a valid map to modeling languages, procedures and mechanisms and algorithms. However, the formalisms that we have come across so far in meta modeling only target selected parts of modeling methods. Mostly, only the modeling language part is covered. Second, the combination of the components of modeling methods is not yet as well understood on a theoretical level as the shown relations may imply. Rather it is still being investigated in current research how for example formally described modeling procedures are combined with a modeling language and mechanisms and algorithms in a universally applicable fashion [Bork and Sinz (2014)]. Third, it needs to be pointed out that the application of the abstraction functions, in particular of f_{MM}^{abst}, requires considerable human involvement and seems to be hardly automatable. Therefore, the contribution of these definitions is to establish a more concrete working hypothesis for characterizing abstraction in the context of meta modeling.

When focusing now on the role of transparency and the above mentioned characteristics of traceability, visibility, availability, and verifiability of information, we can also approach it from the viewpoint of meta modeling. Again, what draws our attention is the relation between models and originals and modeling methods and originals. However, in contrast to abstraction where we can clearly distinguish two levels and the focus seems to lie more on the meta modeling level by providing more freedom, transparency seems to be essentially dependent on a combination of both levels. For the properties of transparency it is important what kinds of inferences are possible from models to originals. These inferences are of course also determined by the modeling method that is used for the creation of the models. We can therefore propose a first idea of a transparency function f^{transp} that maps combinations of a modeling method and its models to originals O.

$$f^{transp}: \bigcup_j \left(\mathrm{ML}_j \times \mathrm{MP}_j \times \mathrm{MA}_j \times \mathrm{M}_j\right) \rightarrow \mathrm{O}$$

As in the previous definition of the abstraction functions, we only discuss the domain and codomain of the transparency function here without going into further detail. Nevertheless this already permits us to gain some first insights into the theoretical conception of transparency in the context of meta modeling. For example, we can find that transparency depends on all components of a modeling method and not necessarily only on the constructs provided by a modeling language, which may have been an intuitive approach, e.g. when just regarding a visual representation of a model. Based

on this definition it may thus be possible that a certain algorithm provided in a modeling method allows to verify information in the models in a way that is not immediately visible, e.g. through the deduction of new facts from complex logical definitions in the models. In a similar manner, also the modeling procedure may have an influence on transparency, e.g. by requiring a modeler to fill in certain attribute values. And of course, the modeling language and the actual models themselves play an essential role which insights can be derived in regard to the originals.

With these first theoretical sketches of abstraction and transparency functions, a foundation is provided for further analyses. From these definitions e.g. a discussion of metrics for the degrees of abstraction and transparency could be initiated. Thereby different modeling methods and models could be compared for example in regard to their suitability for the identification of meta level concepts or for the traceability of information. In this context, also the integration of time and evolutionary aspects, e.g. to trace changes in transparency over time, are potentially interesting. However, to conduct such analyses in a formal way, further definitions of the constituents of the domains and codomains of the functions are necessary. Although formal definitions are available for the part of modeling languages and meta modeling formalisms, it needs to be further investigated how mechanisms and algorithms and modeling procedures can be formally defined in a universal way. Candidates that may contribute input for this purpose can be found in the area of constraint languages for models such as the UML object constraint language OCL or for certain modeling languages that are graph-based in the field of graph algorithms.

What can immediately follow from the definitions are also further argumentative analyses of abstraction and transparency. Of particular interest here seem the pragmatic implications that can be systematically derived and that we will illustrate in the following.

4. Application Examples

To give a first idea how the derived theoretical foundations can contribute to practical applications, we will discuss in the following two concepts that can be analyzed with the presented definitions.

At first we will take up the concept of *extraction*, which has been discussed in legal informatics for example to recognize, understand and normalize normative references in legal texts and thus permit the processing by legal information systems [Palmirani et al. (2003)]. Extraction is thus a basic requirement for technically-oriented multi-sensory law applications as it can serve as input for the translation into various multi-sensory interpretable formats.

Consider for example that the information contained in the Austrian RIS legal information system needs to be made accessible in a barrier-free manner, e.g. through special XML formats for processing by Braille devices [Bernier and Burger (2012)]. All texts in RIS are today stored in an XML format with a corresponding XML schema [Stöger and Weichsel (2008)]. The XML documents can thus be regarded as models whose modeling language is set by the XML schema. According to the abstraction function f_{MM}^{abst} we thus recognize that the used formalism is set to the XML definitions as specified by W3C and the regarded original that served as input for the abstraction are the legal texts that are to be stored in RIS. The modeling procedure that is used to populate the models is today based on forms for entering meta data and texts as well as algorithms that are able to extract the information from Word-Files [Weichsel (2013)]. Based on this definition of the modeling method, we can also analyze the abstraction function for models f_{M}^{abst}. It is obvious that the resulting models, i.e. the XML documents in this case, depend on the characteristics of the XML schema, the modeling procedure and the used transformation algorithms. If we decided now to design an application that extracts legal information from the models, e.g. through according regular expressions [Palmirani et al., (2003)], we would have to take these aspects into account. For example, it could be beneficial to either slightly modify the used XML

schema to better support the intended extraction. Another option could be to think about a modification of the modeling procedure and the underlying algorithms, e.g. to encode the original textual information with meta data that are targeted towards the later extraction but that do not need to be added to the RIS XML schema.

A second application example that we will discuss is based on the concept of *obfuscation*. Obfuscation relates to the hiding of some information parts while maintaining processability to some extent. It is typically used in software engineering for complicating reverse engineering or in data analysis for desensitizing data sets from personal information. Also in legal informatics obfuscation has been discussed, for example in the context of the Austrian transparency database [Kappl (2013)]. Also in the RIS database obfuscation occurs where personal information, e.g. from appellants who are mentioned in court decisions, needs to be removed due to data privacy so that persons cannot be directly identified. Based on the transparency function f^{transp} we can analyze the according relationships by viewing again the schema of the databases as the modeling language. Although the schema of the Austrian transparency database is – to the best of our knowledge – not publicly accessible in a technical format, the literature discussing its properties indicates that there are different views ("Leistungsbereich") provided for different stakeholders [Kappl (2013)]. In addition, the actual obfuscation of personal information in the transparency database is based on specific algorithms that create encrypted, section specific identifiers ("verschlüsseltes bereichsspezifisches Personenkennzeichen"). At IRIS 2012 the details as well as an enhancement of these algorithms have been presented by [Schartner (2012)]. Regarding the modeling procedure, i.e. how the data is actually entered in the database, it seems to be based on an automatic procedure that assembles the relevant information from various existing databases. Together with the interface for the transparency database as it is available for the public at https://transparenzportal.gv.at/, we can thus investigate which inferences are possible, i.e. which originals can be identified. Thereby, an interesting aspect is for example the comparison to the obfuscation used in RIS. In RIS, the names are not obfuscated using the above mentioned algorithms, but are replaced by XXX or the initials of the names – see e.g. the decisions in the judicature section of RIS at http://www.ris.bka.gv.at/Judikatur/. Therefore, a direct coupling based on personal identifiers with the information from the transparency database does not seem possible at the moment.

5. Conclusion and Outlook

With the first characterization of functions for abstraction and transparency in the context of meta modeling, we could provide a theoretical foundation for analyzing these phenomena in a formal way. Based on these functions it was possible to conduct a theoretical analysis of the concepts of extraction and obfuscation. Thereby the benefit of the theoretical foundation became obvious as it provided a frame for deriving potential new designs through a systematic analysis and permitted the comparison of different realizations.

From here more fine grained characterizations of the abstraction and transparency functions can now be investigated. For this purpose, the constituents of the domain and codomain of the functions will have to be described in more detail. Although for some constituents universally applicable descriptions are already available, e.g. for formalisms to describe meta models and their instantiation to models, for others, such as modeling procedures and mechanisms and algorithms, these foundations still need to be provided. With a more complete characterization of the functions, also the derivation of metrics for comparing different instantiations of the functions could be accomplished.

References

Berbier, A., & Burger, D.,XML-Based Formats and Tools to Produce Braille Documents. In: Miesenberger, K. et al. (Eds.), ICCHP 2012, Springer, LNCS 7382, Berlin, p. 500-506 (2012).

Bork, D. & Sinz, E., Bridging the Gap from a Multi-View Modelling Method to the Design of a Multi-View Modelling Tool, to appear in: Enterprise Modeling and Information Systems Architectures (2014).

Brunschwig, C. R., Multisensory Law and Legal Informatics - A Comparison of How these Legal Disciplines Relate to Visual Law. In: Geist, A., Brunschwig, C.R., Lachmayer, F. & Schefbeck, G. (Eds.), Strukturierung der Juristischen Semantik – Structuring Legal Semantics, Festschrift für Erich Schweighofer, Editions Weblaw, Bern, p. 573-668 (2011).

Fiedler, H., Komplexitätsreduktion als Thema in Rechtstheorie und Rechtsinformatik. In: Schweighofer, E., Geist, A., Heindl, G., Szücs, C. (Eds.), Komplexitätsgrenzen der Rechtsinformatik, Boorberg Verlag, Stuttgart, p. 459-462 (2008).

Favre, J.-M., Foundations of Meta-Pyramids: Languages vs. Metamodels, In: Language Engineering for Model-driven Software Development, Dagstuhl Report, IBFI (2005).

Fill, H.-G. & Karagiannis, D., On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform. Enterprise Modelling and Information Systems Architectures, Vol. 8(1), p. 4-25 (2013).

Fill, H.-G., Redmond, T., Karagiannis, D., FDMM: A Formalism for Describing ADOxx Meta Models and Models. In: Maciaszek, L., Cuzzocrea, A., and Cordeiro, J. (Eds.), ICEIS 2012, Wroclaw, p. 133-144 (2012).

Frank, U., Some Guidelines for the Conception of Domain-Specific Modelling Languages. In: Nüttgens, M., Thomas, O., Weber, B. (Eds.), EMISA 2011, Vol. P-190, GI, p. 93-106 (2011).

Harel, D. and Rumpe B., Meaningful Modeling: What's the Semantics of "Semantics"?, IEEE Computer, October, p. 64-72 (2004).

Kahlig, W., Visualisierungstypologie des Deutschen Privatrechts. In: Jusletter IT 24. Februar 2011, (2011).

Kappl, A., Die Österreichische Transparenzdatenbank – Die rechtlichen Rahmenbedingungen. In: Schweighofer, E., Kummer, F. & Hötzendorfer, W. (Eds.), Abstraktion und Applikation, book@ocg.at, Band 292, p. 207 (2013).

Kienreich, W., Schulze, G., Lex, E., Rapp, S.,Eine Kombination von regelbasierten und statistischen Verfahren für die hierarchische Klassifikation von juristischen Dokumenten. In: Schweighofer, E., Kummer, F. & Hötzendorfer, W. (Eds.), Abstraktion und Applikation, book@ocg.at, Band 292, p. 73-77 (2013).

Karagiannis, D., Kühn, H., Metamodelling Platforms, Proceedings of the 3rd International Conference EC-Web 2002 - Dexa 2002 (2002), Full version: http://www.dke.at/fileadmin/DKEHP/publikationen/metamodell/FullVersion_MMP _DexaECWeb2002.pdf retrieved 10-01-2013.

Lachmayer, F., Schweighofer, E.,Abstraktion und Applikation. In: Schweighofer, E., Kummer, F. & Hötzendorfer, W. (Eds.), Abstraktion und Applikation, book@ocg.at, Band 292, p. 31-34 (2013).

Mylopoulos, J., Conceptual Modeling and Telos. In: Loucopoulos, P. and Zicari, R. (Eds.), Conceptual Modelling, Databases and CASE: An Integrated View of Information Systems Development, Wiley, p. 49-68 (1992).

Palmirani, M., Brighi, R., Massini, M., Automated Extraction of Normative References in Legal Texts, ICAIL'03, ACM, p. 105-106.

Schartner, P., Sicherheitsanalyse des (Wirtschafts-)Bereichsspezifischen Personenkennzeichens. In: Schweighofer, E., Kummer, F. & Hötzendorfer, W. (Eds.), Transformation juristischer Sprachen, book@ocg.at, Band 288, p. 523-528 (2012).

Stachowiak, H., Allgemeine Modelltheorie, Springer (1973).

Stöger, H., Weichsel, H., Das Redesign des Rechtsinformationssystems RIS. In: Schweighofer, E., Geist, A., Heindl, G., Szücs, C. (Eds.), Komplexitätsgrenzen der Rechtsinformatik, Boorberg Verlag, Stuttgart, p. 235-243 (2008).

Tscheliesnig, S. & Weinzettl, R., Die Österreichische Transparenzdatenbank – Ziele, Nutzen und Technische Umsetzung. In: Schweighofer, E., Kummer, F. & Hötzendorfer, W. (Eds.), Abstraktion und Applikation,book@ocg.at, Band 292, p. 203-206 (2013).

Wacker, R. & Peitz, P., Semantische Erweiterung von USDL zur Vorbereitung einer automatischen Subsumtion. In: Schweighofer, E., Kummer, F. & Hötzendorfer, W. (Eds.), Transformation juristischer Sprachen, book@ocg.at, Band 288, p. 125-130 (2012).

Weichsel, H., Rechtsinformationssystem RIS – Update 2013. In: Schweighofer, E., Kummer, F. & Hötzendorfer, W. (Eds.), Abstraktion und Applikation, book@ocg.at, Band 292, p. 53-55 (2013).