

# ONTOLOGIES AND SEMANTIC WEB TECHNOLOGIES TO SUPPORT KNOWLEDGE- BASED TRANSFORMATION AND DECISION SUPPORT

Tobias LEHMANN and Andreas KARCHER

**Abstract:** This article describes how semantic web technologies and here ontologies in particular, can support multinational and multi-agency knowledge representation. To this end, model integration and the typically associated with integration problem classes are described. Then, distributed modeling is presented from expert and standard-models perspective, as well as general clustering of common concepts is presented. This is achieved by means of inheritance as known from Object-Oriented Knowledge Representation Systems, used at different levels of abstraction. In addition to the integration facet, new standard languages for the definition of flexible and context-dependent structures, as well as reasoning over these, are briefly introduced and examples of their application are given. As the focal point of this work a Framework for the generic Integration of Structured IT Systems (ISITS) is described, which makes possible the structural and instance integration of systems, using both syntactic as well as semantic means for integration. In conclusion, the applicability of the ISITS-Framework is outlined by enabling EBO-specific planning support. This article focuses more on providing an overview. Details can be found in other publications or projects published by the authors.

**Keywords:** Ontology, Architecture for Model Integration, Distributed Technical Knowledge Representation.

## Model Integration

The integration of two different models harbors several conceptual classes of problems. If one, for example, plans to make two simulation systems interoperable so that a change within one of the systems is propagated to the other without corrupting, biasing, or losing transmitted information, then, speaking from the standpoint of semantics, two layers have to be taken into consideration. These are the structure as well as the behavior of the elements exchanged or addressed. The behavioral aspects are usually harder to compare, model, and more complex in their interrelations to

other concepts than the structural aspects. As an example, the behavior of a battle tank agent is modeled by a set of several thousand or ten-thousand rules varying in number of actions conducted, comparisons made or other rules included. Its attributes, on the other hand, are limited to a set of several hundred, which are all strictly typed, therefore, meshed but still well comprehensible. The focus of this paper is the latter aspect, structural integration. To this end, the following section describes typical problems faced in structural integration.

When two similar but different structures are going to be integrated, conflicts of different kinds arise. First, this is the very much technical task of making applications interoperable. Here, interoperability means, among other things, understanding of each other's serializations of strings, numbers, and objects, as well as defining and invoking methods. This problem is not addressed here. This article focuses on the following challenge: if two applications can communicate with each other and are interoperable, the information exchanged may still need a complicated interpretation, which includes a lot of not explicitly defined rules. One of the most difficult parts of this set of problems is to overcome the inaccuracy and ambiguity of natural language. Obstacles such as context-dependent interpretation by different people with different background, modeling identical subjects in different ways or even from diverse perspectives, are until now a problem not solved automatically. Although this subject will presumably remain subject for scientific research for decades, the approach presented in this paper makes use of available technologies to decrease the effort needed to integrate different systems in a publicly available framework and might, therefore, reduce the required human action and interaction during integration. The above-mentioned conflicts are related to a representation through classes and instances connected by relations including inheritance. This kind of Knowledge Representation is most common and used in the Unified Modeling Language UML,<sup>1</sup> XML Structure Definition Language XSD,<sup>2</sup> SQL, Resource Description Framework RDF,<sup>3</sup> Web Ontology Language OWL<sup>4</sup> and many other languages. So, conflicts are clustered into the following classes:

- *Structural Conflicts*

This very basic problem occurs when two sets of classes, which are connected via relations, are to be mapped from one system to another.<sup>5</sup> Due to modelers' intents and different perspectives in most cases no direct class to class and relation to relation mapping is possible.<sup>6</sup> As depicted in Figure 1, three classes named A, B, C in one case and A, Bb as well as C1 and C2 (which are semantically equivalent to C) in the other case are connected via relations with names depending on the classes' names.

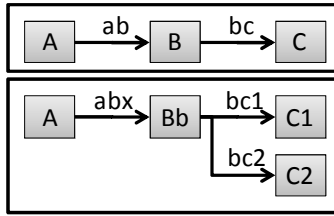


Figure 1: Structure Conflicts.

This conflict is by far more complicated than it might seem to be at first sight. If the number of classes and relations increases to the thousands, then dependencies become unclear since partial classes such as the C1/C2 case might for example be reused in other classes and, therefore, are hard to compare. This conflict is addressed most directly by mechanisms such as “Alignment tools” and “Reasoners” described elsewhere.<sup>7</sup>

- *Label Conflicts*

In general, a label can be understood as a name for a thing that has a meaning.<sup>8</sup> This definition goes eventually back to Aristotle’s semiotic triangle that connects a symbol, a thought or reference, and a referent as described by Ogden and Richards.<sup>9</sup> So, if a meaning has different symbols or a symbol has different meanings (synonyms or homonyms<sup>10</sup>), then interpretation in different contexts mostly leads to ambiguity. This conflict appears in almost every other conflict since it is a kind of basis for addressing things of a world by means of language.

- *Conflicts of Integrity*

This type of conflict is probably one of the hardest ones, since it stems from implicitly defined rules and constraints, which are even sometimes not part of an application’s documentation. A typical representative for this type of conflicts is the usage of primitive data types such as floats for the formalization of particular status. If, as shown in Figure 2, a person has an attribute mood whose range is of type Float, then this is a rather common way of formalization since floats allow fuzzy calculations and are in general easy to handle (fairly high performance, directly comparable, etc.).

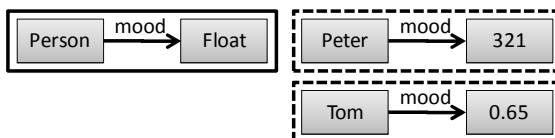


Figure 2: Integrity Conflicts.

The case presented in Figure 2 shows instances of Person named Peter and Tom, both being part of different systems with different integrity constraints. While Peter has mood-value 321, Tom has mood-value 0.65, which does not have meaning at all if the constraints are unknown.<sup>11</sup> Additional information, such as a fuzzy function or any other means (at least documentation) of mapping numerical information to a common understanding is vitally required.

- *Conflicts at Meta-structure Level*

If two modeling languages are to be mapped, then the so called meta-elements of languages might differ. That is, languages allow the usage of different elements such as classes, relations, instances, or constraints. While, for example, OO-Languages such as the .NET family or Java<sup>12</sup> allow multiple inheritance only for interfaces and not for classes, this is a basic feature for Knowledge Representation languages such as RDF(S) or OWL.

- *Conflicts of Data Types*

Primitive data types such as Integer numbers, Strings or Bytes are divergent over different languages. The XML-Schema Definition Language XSD for example, defines Date and Time values as primitive data types, while Java or .NET uses them as classes. Still, a mapping will probably lead to complications since primitive data types are not able to invoke their own methods and therefore have no behavioral aspect. This conflict is listed here for completeness purpose and is considered a minor problem in the context of this work.

At the end of this section, two different ways of conceptual model integration are outlined. The first approach, which is maybe closest to classical military thinking, is a centralized standard to be defined and promoted that all participating parties have to comply to. Industry experience shows that this does in fact work, even for huge structures such as automotive and aeronautic sectors. A mostly European standard<sup>13</sup> is called Standard for the Exchange of Product Data (STEP) and describes many parts used as well as general rules for making objects accessible for distributed applications. But the industry scenario is hardly comparable to the military or the political scenario. The latter contain by far more independent actors with many individual interests and a little will for giving up own products, standards, or inventions.

The second way of integrating models is the so called *ad-hoc* integration. This means a more short-term oriented integration of applications not made for each other, due to an upcoming need. In this approach,  $n$  applications are made interoperable, but not for all available systems. The main purpose is, therefore, not on defining long-term standards.

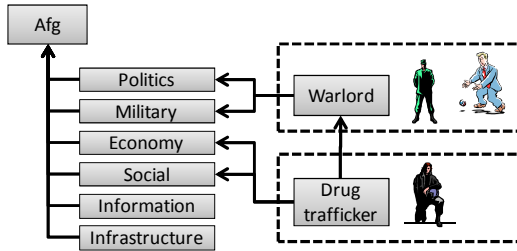


Figure 3: Redundant Information within the PMESII Categories.

The two approaches, centralized standard and *ad hoc* integration, strongly differ in their goals. While the former approach generates benefit in scale by reducing integration costs if people stick to standards and generates costs during its definition and maintenance phase, the latter approach generates benefit in scope by integrating “only”  $n$  applications and generates by far less costs during definition. But if once more applications are to be connected to the *ad hoc* system, integration will most likely become very expensive, because each interface has to be defined individually and cannot refer to an existing standard. Maybe a centralized standard such as STEP stems from a former *ad hoc* integration; it is explicitly not intended to build up an “overhead” for coordination and development.

In general, XSD has become the most popular structure-exchange standard (and XML for data), since it allows a language-independent definition of structured data.<sup>14</sup> But XML describes structures only to a very limited extent, since there is no explicit distinction between classes and relations. For this purpose, XML-derived technologies such as RDF or OWL provide more features as outlined by many researchers.<sup>15</sup>

### Distributed Modeling

Generally speaking, distributed modeling is an analogy to distributed applications with all the hard problems of communication and semantics, redundant data, and parallel changes. Technically, modeling becomes distributed when there are two persons building a common model at separate locations. A crucial advantage of distributed modeling is the fact that experts are available for centralized modeling only for a very limited period of time. Let assume that a model for Afghanistan has to be created from a NATO-perspective; then all kinds of issues become relevant for this model, such as the PMESII (Politics, Military, Economy, Social, Information, and Infrastructure) categories. This example will be used as illustration in what follows to demonstrate the benefits of the new technologies. These technologies are enablers and they represent tools suitable for different purposes. In the given context, a Knowledge Base (KB) would therefore contain information about all six listed cate-

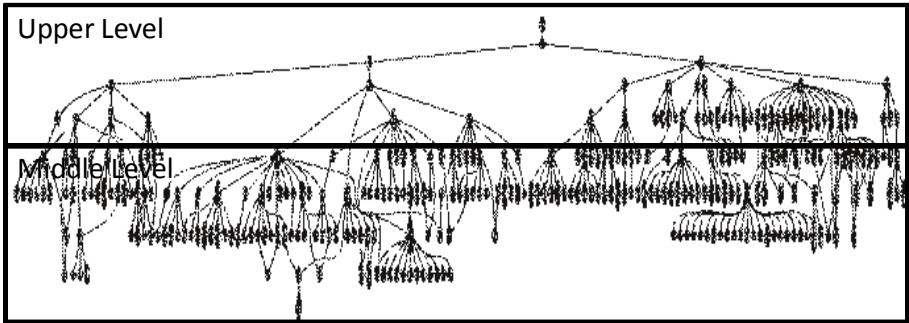


Figure 4: The SUMO Ontology.

gories. For a given category, a number of Subject Matter Experts (SMEs) is responsible to gather, formalize and maintain information. But as depicted in Figure 3, a *Warlord* is of interest for more than one category – here Politics and Military, whereas a *Drug-trafficker* is of interest for the Economy and Social category. In addition, the *Drug trafficker* can be a *Warlord* at the same time.

This simple example shows the need for SME-coordination in order to avoid redundant information as far as possible and therefore to make the KB more reliable and the information that it contains, more up-to-date.

If a KB contains structured information in the sense of at least Object-Oriented (OO) or XSD structures, then a classification that distinguishes through inheriting classes will dramatically reduce the need for classification by individual SMEs. In the ontology world, this distinction is labeled Upper level, Midlevel and Domain level classes (see Figure 4), where Upper-level concepts, such as abstract, physical, time, groups, processes, are included. Figure 4 shows an overview of all classes of the standard ontology SUMO,<sup>16</sup> which is with a relatively small number of classes – 685. The Ontology is available on the Internet.<sup>17</sup> Below this level, the Mid-level classes inherit from the Upper-level classes and are more specific in nature, but still known to the majority of people. These are issues, such as infrastructure, weather, economy, etc. The lowest level, the Domain-level, contains classes, which inherit from Mid-level classes and are so specific that almost nobody except a domain expert can work and judge these concepts. This could be drug cultivation, coordination of military or irregular units, local habits or comparable concepts.

The most significant benefit of such a general classification is not only the re-use of commonly known concepts and their relations, but especially the better comparability of identical classes used in different Domain Ontologies. Once the Domain Level classes have been integrated into this standard Ontology, they become more interre-

lated and, therefore, subject to inference in a wider area of interest, not explicitly defined in advance.

### ***Describing Implicit Structures Using Description Logic***

As outlined in the previous section, using inheritance as a means for avoiding redundancy helps in dealing with distributed models. But in the case when different perspectives on identical classes have to be considered, the classical OO-modeling lacks flexible classification and often even with multiple inheritance. A next step solving this problem is the use of the so called Description Logic (DL). DL's purpose is to describe classes, properties and their relations in terms of inheritance relations at structure level.<sup>18</sup> This explicitly excludes inference on instance level, such as that provided by First Order Logic systems like Datalog, Prolog, F-Logic, or many others.<sup>19</sup> Therefore, DL and its inference engines called Reasoners do not—as so often—cover all needs for inference with one piece of software. But it can be used efficiently as a component for the definition of complex structures that are later on imported by other systems such as Data Bases or OO-Languages. And this is where the benefit for distributed models is generated, as well as where this article intends to make an important contribution. First Order Logic provides a multiple perspective on the issues, while Description Logic allows standardized exchange of structured information not depending on technical systems. In order to give an idea of these aspects, the article, intended to be more of an overview character, provides an implicit PMESII-classification in what follows.

Assume that a *Person* in Afghanistan plays an ambiguous role in the sense that s/he is leader of a legal political organization, as well as a *drug Trafficker* and a *Warlord*. Depending on the context, this person then has to be treated differently, although it is of highest importance for each involved NATO structure or NGO to know the background of this player. Within PMESII, this person would be of relevance for at least four categories and therefore would require (in addition to its problematic activities) high amount of coordination between the responsible analysts. Not regarding any real-world persons, this particular person is called *Muhammad Khan* in what follows. Mr. Khan is—using a DL-based KB described elsewhere by one of the authors<sup>20</sup>—known to be of class *Person* only. This so-called asserted relation is not even necessary, since it could be inferred from the subsequent statements assigned to Mr. Kahn. But for a more intuitive understanding, Mr. Kahn has been classified as such, as depicted in Figure 5 (each ellipse stands for a class, where each arch stands for a subclass relation).

Each *Person*, listed as a subclass here, is described according to his/her relations (in DL, relations (or properties as they are also called) are defined analogously to classes; therefore, they can inherit from each other and connect defined domain-

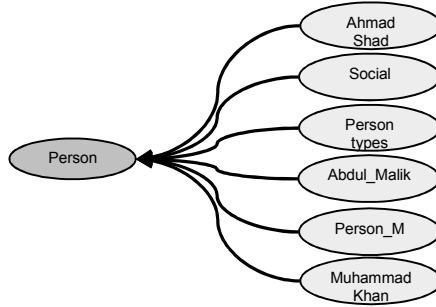


Figure 5: Asserted Classification of Mr. Kahn.

classes with range-classes). What is known about Mr. Khan is that he leads a *Big Militia* (M-category) and that he conducts drug trade (E-category). To give an impression how these definitions look like, Figure 6 depicts them visualized in a user-friendly manner.

Now, the interesting point is that Mr. Khan is classified according to other DL-statements stored in the KB, which are too many to be explained here but again can be found in another publication by the authors.<sup>21</sup> The result obtained in the *Person*-related section of the example KB is illustrated in Figure 7 and shows a fairly complex class hierarchy, containing several multiple inheritances as well as cyclic inheritance relations. On the right hand, Mr. Kahn (the solid-line rectangle) has been assigned to be a *Trader*, since he is involved in *drug trade*. Due to the fact that *drug trade* is considered strongly illegal activity, Mr. Kahn becomes a member of the *Anti-Government* class. This makes him interesting to the P-Analyst. Each *Trader* is important for the E-Analyst. That is the reason he to be classified as element of *Economic* importance.

Due to his role as *Leader of Big Militia* he has not only become a *Warlord* but, in addition, he became, based on the potential influence of *Big Militia*, a *Dangerous Warlord*. Each *Warlord*, and the *Dangerous Warlord* in particular, is classified as M-important. This example shows only a small excerpt of the KB. What is of interest here is the fact that the relations are inferred during reasoning and do not have to be



Figure 6: Description Logics Statements for Mr. Khan.

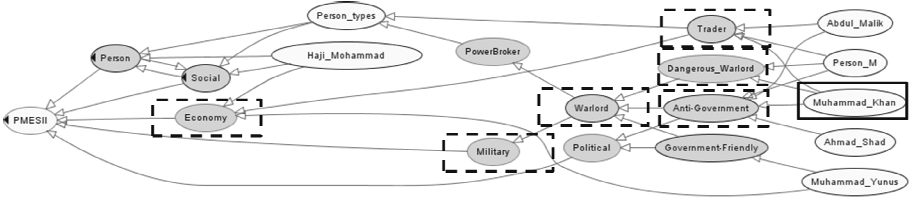


Figure 7: Inferred Classification in the Afghanistan Example.

defined statically. This means that, for example, if the status of *Drug Trade* is changed from illegal to legal, automatically Mr. Khan would no longer be part of the *Anti-Government* class (at least not due to his trading activities). No analyst would need to change this status from true to false manually.

The next feature DL provides is to query a KB for the class a concept belongs to within the context of the particular KB if only part of its properties and constraints are known. If a concept was at hand, for which is known only that it conducts drug trade and leads a militia, then classification results would have been identical to those reasoned in the Muhammad Khan example. In return, this is a better means to detect similar or already existing concepts within an existing KB.

It should be mentioned here that properties play a special role in DL.<sup>22</sup> They can inherit from each other and therefore become important means for PMESII or any other general classification. An example of specialization is *Turnover* of certain goods. *Trading* and *Production* could become subrelation of this relation since both somewhat have to do with a certain class (e.g., *Drugs* are produced as well as traded). For a more detailed example the reader may refer to another work of the authors.<sup>23</sup> In addition, Lehmann and Karcher<sup>24</sup> in their work on modeling complex systems describe ways and options for a broader exploitation of DL-statements as defined by its W3C standard.<sup>25</sup>

## A Framework for Integration of Structured IT Systems

As already presented, DL is an appropriate means for modeling and maintaining distributed KBs within heterogeneous environments. This section outlines a Framework, which now enables not only the integration of different models based on technologies from the Semantic Web community such as Standard Libraries or structure Alignment Tools, but also a generic integration of different technological systems such as databases, mindmaps, OO-Systems, XML or Ontologies. Integration in the context of this article means propagation of changes over equal or similar structures, cloning and synchronization of structured information.

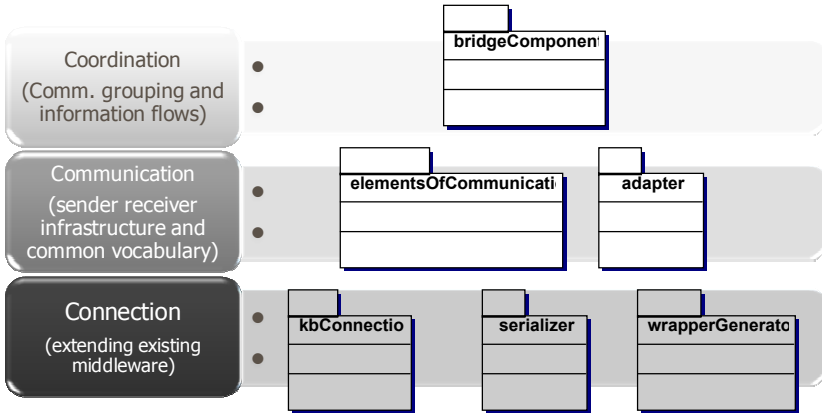


Figure 8: Framework for the Integration of Structured IT Systems.

The Framework is publicly available on the site of the Sourceforge project.<sup>26</sup> The Framework itself consists of several abstract packages, each with one or more default implementations in order to fulfill certain functionality. As depicted in Figure 8, there are three different layers of functionality.<sup>27</sup> These are (from bottom to top) the *Connection Layer*, which defines generic access to Knowledge Bases, as well as serializer components and wrapper-generator to make contents available in OO-Languages.

The second layer defines allowed *Elements of Communication* (EoCs) for the interchange of structured information between different systems. In order to guide and coordinate the messages exchanged, so called *Adapters* are used as “wrappers” for systems. *Adapters* are described within the *Adapter* package.

The last package to be mentioned here is the *bridgeComponent* package that allows to register and deregister adapters and to transform the transmitted messages. This is rather similar to Microsoft’s Biztalk Server’s XML-integration component, however all without GUI, but under the use of Alignment results.<sup>28</sup> This allows a fast, machine-aided detection of similarities between large structures, whilst XML-based integration can be conducted in parallel. Without going too deep into details, the following very brief example shows how an Ontology and a MySQL Data Base are integrated, although they do not follow identical structure definitions. Let us assume that there is, as shown in Figure 9, a *Request* for something, e.g., a spare part. This *request* is performed from the Ontology side (left hand side). It is created within an Ontology and can be transmitted to an integration *Bridge* via Adapter which is called AJ in the figure. AJ knows how to read within an Ontology, how to listen to change-events, as well as how to propagate incoming changes into the KB. By using the

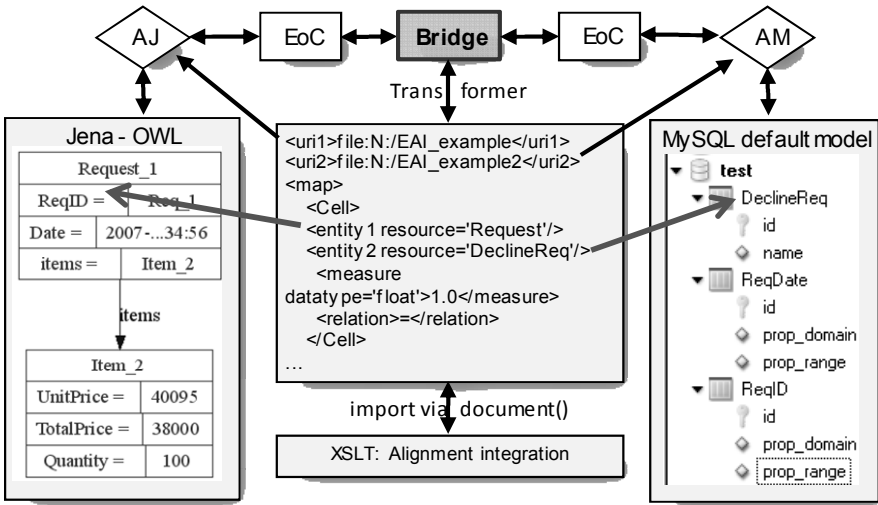


Figure 9: Structure Integration using Alignment Results.

*Elements of Communication*, which reduce all exchanged elements to classes, superclass-relations, instances, properties and restrictions, this request is broadcasted to the central bridge. A *Bridge* knows its registered *Adapters* and makes use of implementation of generic *Transformers*, which are identified by their domain and range *Adapter-Identifiers*. In this particular case, there is one implementation for the communication between AJ and AM (which adapts the MySQL Data Base at the right hand side of the figure).

There can be an unbounded number of transformers although they have to be prioritized since many transformations are not transitive actions. Such an alignment tool (here Falcon<sup>29</sup> is used) found automatically is that *Request-Decline* class is a superclass of *Request*. This very technical view leads to the possibility to create such an object just based on the existence of a request-object, which has been sent by a system unknown to the MySQL Data Base. The mechanism behind this is not only syntactical comparison but structural comparison based on mechanisms from graph-theory, as well as lexical look-ups from dictionaries such as the WordNet.<sup>30</sup> In the center of Figure 9, example-result of an alignment process is presented, which shows a XML-based map for classes and properties, including a similarity measure (which is 1.0 here).

As a result of this framework, each structured system can be brought to Ontology level, serve as input for alignment and then benefit from these results. Especially, in an environment where many similar models in different systems on different plat-

forms are operated, this framework provides a good basis for an extensible and flexible hub-and-spoke integration.

## **Outlook**

This article has demonstrated how new technologies from the Semantic Web Community can provide help within the domain of distributed (in this case multinational and multi-agency) Knowledge Representation. Not only the standardization of models, but also the flexible structures implemented in Description Logic can reduce redundancy by enabling multiple perspectives and semantic queries through DL-statements resolved by Reasoners.

In order to bring this variety of systems together under one common umbrella, a framework has been created to integrate systems on the basis of common Elements of Communication (EoC). This integration is done by Adapters that know how to address and access a model within a given system, listen to its changes and have a connection to bridges. Bridges, as the hub-element within this hub-and-spoke integration architecture, can transform the EoCs according to mechanisms known until that moment and are in general open to any other future transformation, which can confirm its applicability depending on two given KBs.

In the future, the Sourceforge project will further be developed, especially by uploading standard adapters and an interactive library for user-developed solutions. As a future application, the Multination Experiment 5 in summer 2008 will provide various challenges for integration and will eventually lead to new insights for improvement, as well as already existing good features.

## Notes:

---

- <sup>1</sup> Object Management Group, *UML Resource Page*, <[www.uml.org](http://www.uml.org)> (30 April 2007).
- <sup>2</sup> David C. Fallside and Priscilla Walmsley, eds., “XML Schema Part 0: Primer Second Edition,” W3C Recommendation, 28 October 2004, <[www.w3.org/TR/xmlschema-0/](http://www.w3.org/TR/xmlschema-0/)> (5 April 2006).
- <sup>3</sup> Graham Klyne and Jeremy J. Carroll, eds., “Resource Description Framework (RDF): Concepts and Abstract Syntax,” W3C Recommendation, 10 February 2004, <[www.w3.org/TR/rdf-concepts/](http://www.w3.org/TR/rdf-concepts/)> (28 November 2005).
- <sup>4</sup> Grigoris Antoniou and Frank van Harmelen, “Web Ontology Language: OWL,” in *Handbook on Ontologies in Information Systems*, ed. Steffen Staab and Rudi Struder (Berlin: Springer-Verlag, November 2003), 76–92.
- <sup>5</sup> Cheng Hian Goh, *Representing and Reasoning about Semantic Conflicts in Heterogeneous Information Systems*, Ph.D. Thesis (Massachusetts: Massachusetts Institute of Technology, 1997).
- <sup>6</sup> Holger Wache, *Semantische Mediation für heterogene Informationsquellen* (Berlin: Akademische Verlagsgesellschaft Aka GmbH, 2003).
- <sup>7</sup> Marc Ehrig and York Sure, “FOAM – Framework for Ontology Alignment and Mapping; Results of the Ontology Alignment Initiative,” in *Proceedings of the Workshop on Integrating Ontologies*, ed. Benjamin Ashpole, Marc Ehrig, Jérôme Euzenat, and Heiner Stuckenschmidt, Volume 156, <[CEUR-WS.org](http://CEUR-WS.org)> (October 2005), 72–76; Paolo Bouquet, Marc Ehrig, Jérôme Euzenat, Enrico Franconi, Pascal Hitzler, Markus Krötzsch, Luciano Serafini, Giorgos Stamou, York Sure, and Sergio Tessaris, *Specification of a Common Framework for Characterizing Alignment*, Knowledgegeweb deliverable 2.2.1v2. (Karlsruhe: University of Karlsruhe, December 2004), <[www.inrialpes.fr/exmo/cooperation/kweb/heterogeneity/deli/kweb-221.pdf](http://www.inrialpes.fr/exmo/cooperation/kweb/heterogeneity/deli/kweb-221.pdf)>; Jérôme Euzenat, David Loup, Mohamed Touzani, and Petko Valtchev, “Ontology Alignment with OLA,” in *Proceedings of the 3<sup>rd</sup> International Workshop on Evaluation of Ontology-based Tools (EON) Workshop, 3<sup>rd</sup> International Semantic Web Conference*, ed. York Sure, Oscar Corcho, Jérôme Euzenat, and Todd Hughes (Hiroshima, Japan, 2004), 59–68, <[ftp://ftp.inrialpes.fr/pub/exmo/publications/euzenat2004d.pdf](http://ftp.inrialpes.fr/pub/exmo/publications/euzenat2004d.pdf)>; Jérôme Euzenat and Petko Valtchev, “An Integrative Proximity Measure for Ontology Alignment,” in *Proceedings of the First International Workshop on Semantic Information Integration, Second International Semantic Web Conference (ISWC-03)* (Sanibel Island (FL/US), 2003), 33–38, <[ftp://ftp.inrialpes.fr/pub/exmo/publications/euzenat2003h.pdf](http://ftp.inrialpes.fr/pub/exmo/publications/euzenat2003h.pdf), <http://CEUR-WS.org/Vol-82/>>.
- <sup>8</sup> Hans Geißlinger, *Die Imagination der Wirklichkeit. Experimente zum radikalen Konstruktivismus* (Frankfurt am Main, New York: Campus Verlag, 1992).
- <sup>9</sup> Charles K. Ogden and I.A. Richards. *The Meaning of Meaning: A Study of the Influence of Language upon Thought and of the Science of Symbolism* (New York: Harcourt, June 1989).
- <sup>10</sup> Christiane Fellbaum, eds., *WordNet: An Electronic Lexical Database* (Cambridge, MA: The MIT Press, May 1998).
- <sup>11</sup> B. Peter, “Semantische Datenintegration: Strategien zur Integration von Datenbanken,” 2005 <[www.tzi.de/~cyco/lehre/sem-dat-int/\\_material/sem-dat-int\\_peter.pdf](http://www.tzi.de/~cyco/lehre/sem-dat-int/_material/sem-dat-int_peter.pdf)> (7 December 2005); Andreas Schmidt, “Architekturen und Systeme zur Integration autonomer Informationsquellen,” 2004 <[www.tzi.de/~cyco/lehre/sem-dat-int.html](http://www.tzi.de/~cyco/lehre/sem-dat-int.html)> (25 August 2005).

- <sup>12</sup> Microsoft, *C# Version 2.0 Specification* (2005), <msdn.microsoft.com/vcsharp/programming/language/> (26 May 2006); Dare Obasanjo, “A Comparison of Microsoft’s C# Programming Language to Sun Microsystems’ Java Programming Language,” 2001, <www.25hoursaday.com/CsharpVsJava.html> (5 March 2007).
- <sup>13</sup> Martin Eigner and Ralph Stelzer, *Produktdatenmanagement-Systeme*, Volume 2 (Berlin: Springer, October 2006).
- <sup>14</sup> Fallside and Walmsley, “XML Schema Part 0: Primer Second Edition.”
- <sup>15</sup> Michael K. Smith, Chris Welty, and Deborah L. McGuinness, eds., “OWL Web Ontology Language Guide,” W3C Recommendation, 10 February 2004, <www.w3.org/TR/owl-guide/> (25 September 2007); Alan L. Rector, Nick Drummond, Matthew Horridge, Jeremy Rogers, Holger Knublauch, Robert Stevens, Hai Wang, and Chris Wroe, “OWL Pizzas: Practical Experience of Teaching OWL-DL: Common Errors & Common Patterns,” in *Proceedings of the European Conference on Knowledge Acquisition* (Northampton, England, 2004), Lecture Notes in Computer Science 3257, ed. Enrico Motta, Nigel Shadbolt, Arthur Stutt, and Nicholas Gibbins (Berlin, Heidelberg: Springer-Verlag, September 2004), 63–81, <www.cs.man.ac.uk/~rector/papers/common\_errors\_ekaw\_2004.pdf> (27 April 2007); Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein, “OWL Web Ontology Language Reference,” W3C Recommendation 10 February 2004, <www.w3.org/TR/owl-ref/> (30 June 2006).
- <sup>16</sup> Ian Niles and Adam Pease, “Towards a Standard Upper Ontology,” in *Proceedings of the 2<sup>nd</sup> International Conference on Formal Ontology in Information Systems (FOIS-2001)* (Ogunquit, Maine, 17-19 October 2001), ed. Chris Welty and Barry Smith, 2–9, <home.earthlink.net/~adampease/professional/FOIS.pdf> (20 September 2007); Standard Upper Ontology Working Group (SUO WG), IEEE P1600.1 – References to Ontologies,” 2003, <suo.ieee.org/SUO/Ontology-refs.html> (31 July 2006).
- <sup>17</sup> IEEE, “SUMO Ontology,” 2005 <reliant.tekknowledge.com/DAML/SUMO.owl> (27 July 2007).
- <sup>18</sup> Steffen Staab and Rudi Studer, eds., *Handbook on Ontologies* (International Handbooks on Information Systems) (Berlin/ Heidelberg/ New York: Springer, January 2004); Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, *The Description Logic Handbook: Theory, Implementation and Applications*, Volume 1 (Cambridge: Cambridge University Press, March 2003).
- <sup>19</sup> Franz Baader, Ian Horrocks, and Ulrike Sattler, “Description Logics,” in *Handbook on Ontologies*, 3-28.
- <sup>20</sup> Tobias Lehmann, “A Description Logics View on PMESII Classification in Afghanistan,” 2007, <emma.informatik.unibw-muenchen.de/\_portal/\_content/persons/assistents/lehmann\_files/diss/ontologies/DL\_ex\_Afg.owl> (23 July 2007).
- <sup>21</sup> Tobias Lehmann and Andreas Karcher, “Logic Agents and Conceptual Interoperability through Semantic Web Technologies” (paper presented at the 13<sup>th</sup> Americas Conference on Information Systems (AMCIS 2007), Keystone, Co., 2007).
- <sup>22</sup> Bechhofer, van Harmelen, Hendler, Horrocks, McGuinness, Patel-Schneider, and Stein, “OWL Web Ontology Language Reference.”
- <sup>23</sup> Lehmann and Karcher, “Logic Agents and Conceptual Interoperability through Semantic Web Technologies.”

- <sup>24</sup> Tobias Lehmann and Andreas Karcher, “Modeling Complex Systems using the Web Ontology Language” (paper presented at the 5<sup>th</sup> Industrial Simulation Conference, Delft University of Technology, Delft, The Netherlands, 11-13 June 2007).
- <sup>25</sup> Deborah L. McGuinness and Frank van Harmelen, “OWL Web Ontology Language Overview,” W3C Recommendation 10 February 2004, <www.w3.org/TR/owl-features/> (24 November 2005); Boris Motik, Peter F. Patel-Schneider, and Ian Horrocks, “OWL 1.1 Web Ontology Language Structural Specification and Functional-Style Syntax (Editor’s Draft of 23 May 2007),” <www.webont.org/owl/1.1/owl\_specification.html> (19 June 2007).
- <sup>26</sup> Tobias Lehmann, “Integration of Structured IT-Systems – The Sourceforge Project,” <sourceforge.net/projects/isits> (23 July 2007).
- <sup>27</sup> Tobias Lehmann, “Framework for the Integration of Structured IT Systems” (paper presented at the European Semantic Web Conference Workshop for Semantic Web Enabled Software Engineering, Salzburg, 2007).
- <sup>28</sup> Bouquet, Ehrig, Euzenat, Franconi, Hitzler, Krötzsch, Serafini, Stamou, Sure, and Tessaris, *Specification of a Common Framework for Characterizing Alignment*.
- <sup>29</sup> Ningsheng Jian, Wei Hu, Gong Cheng, and Yuzhong Qu, “Falcon-AO: Aligning Ontologies with Falcon,” in *Proceedings of the K-CAP 2005 Workshop on Integrating Ontologies* (Banff, Canada, October 2005), 85–91.
- <sup>30</sup> Fellbaum, *WordNet: An Electronic Lexical Database*.

**TOBIAS LEHMANN** is a Ph.D. candidate at the University of the Federal Armed Forces in Munich, Germany, where he received his B.Sc. and his M.Sc. in Information Systems. He is currently a Captain of the German Armed Forces’ IT-Service Department. His research is focused on technical Knowledge Representation within heterogeneous Environments and Decision Support through Logic Agents. *E-mail*: tobias.lehmann@unibw.de.

**ANDREAS KARCHER** studied Computer Science and graduated with a diploma (equivalent to a M.S.) in 1986. After working five years as a project director for an IT-consultancy in Karlsruhe, he joined the Technische Universität München in 1992 where he received his doctoral degree in 1996. Since February 2004 Prof. Karcher holds a tenure track University Professorship for Application Integration at the Institute of Applied System Sciences and Business Informatics at the University of the Federal Armed Forces in Munich, Germany. His main research topics are Reference Architectures, Knowledge Based Decision Support, Semantic Models, and Information Integration and Middleware; *E-mail*: andreas.karcher@unibw.de.