Exploring Model Engineering Techniques for the Integration of Heterogeneous Context Models

Wieland Schwinger [1]

Abstract

The provision of context-aware services in terms of the anytime/anywhere/anymedia paradigm requires the consideration of certain context information in terms of, e.g., time, location, device, and user's preferences. These various context information will often be provided through a series of various sources being available in diverse formats and standards, e.g. device profiles, user databases, or Geo-Services. Nevertheless, not before being able to consider context information in combination during the development of such context-aware systems, the full potential of context-awareness cannot be effectively exploited. Whereas existing work mostly perceives the integration of different context sources at the system level, e.g., by a framework approach or dedicated middleware, we tackle the problem in this paper differently, namely at the model level. We propose the use of model-driven engineering techniques to facilitate the integration of context sources in a bottom-up way. For this, ModelCVS, a semantic infrastructure for model-based integration allows to overcome existing heterogeneities of various context sources through a metamodel-based transformation approach. In the scope of this paper, we demonstrate the applicability of ModelCVS in the course of three example scenarios pointing out potentials and limitations of such a model-based integration approach.

1. Introduction

The provision of context-aware services in terms of the anytime/anywhere/anymedia paradigm requires the consideration of certain context information in terms of time, location, device, and user's preferences. To provide the most added-value for a user, context-aware services need to be capable to fulfill complex use cases like: "Show me those multimedia presentations of Points of Interest I am interested in, currently open to be visited within my vicinity, which I have not yet seen, taking into account my current device's capabilities." To enable context-awareness of a system, some sort of reasoning on the context need to be conducted and appropriate reaction in the system have to be foreseen. The context information necessary to serve such requirements will often be provided through a *series of various sources being available in diverse formats and standard*s, e.g., Context Capabilities Preferences Profile (CC/PP) [23], Geo-Services, or user databases. Those various sources need to be considered in an integrated way to be able to determine the relevant context to fulfill the requirement exemplified above.

With respect to the development of such context-aware systems and information systems in general, models have been used successfully to reduce complexity, lay down design decisions, and facilitate communication within project teams. The advent of the model-driven development paradigm takes modelling even a step further as it puts models as first class artefacts in the software development process. When following a model-driven approach in developing such systems, it is necessary to

[1] Johnannes Kepler University Linz, Altenberger Straße 69, A-4040 Linz, Austria

model, besides the "basic functionalities" of the system, also its context-awareness, i.e., the appropriate consideration of the application according to certain context situations. Consequently, to allow for a model-driven development of context-aware systems, one of the essential prerequisites is the modelling of context.

Representing context information at a model level is not only beneficial for the development of context-aware systems as a whole, but rather brings up another major advantage. It allows dealing with the heterogeneities of the different context sources at the model level, thus abstracting from the traditional system-level approaches as done, e.g., by frameworks or dedicated middleware (cf., e.g., [14], [15], [22]). In particular, metamodel standards such as OMG's MetaObject Facility (MOF)[2] together with standardized transformation languages like Queries/Views/Transformations (QVT)[3] can be used to deal with heterogeneous context information in an elegant way.

The goal of this paper is therefore to elaborate on an approach for the integration of context sources at a model level, employing techniques in the realm of model engineering. For this, we advocate the employment of ModelCVS – a semantic infrastructure for model-based integration – which can serve the need for integration of context sources in two ways. Firstly, it can serve for the identification of similarities between different context sources overcoming data model heterogeneities and, secondly, it allows for a transformation from one context model to another by means of automatically generated transformations.

The paper structures as follows: In the following, heterogeneities of context models are discussed and the need for integration at model level is motivated. Section 3 presents ModelCVS in general and Section 4 gives a discussion about the applicability of ModelCVS by three representative example scenarios. Section 5 addresses related work before, Section 6 finally concludes and points out future research directions.

2. Heterogeneities Between Context Models

In the following, the problems faced when considering different context sources at the model level are laid out and issues when integration context models are discussed.

2.1. Diversity of Context Sources and Context Models

Developers of context-aware systems face the formidable task of expressing the context-awareness of the system-to-be in terms of appropriate models. Thus, the developer needs to express – in terms of a model – besides the "basic functionalities" of the system, for which a general purpose modeling language like the Unified Modeling Language (UML) might well be the first hand choice, also the system's context-awareness. Regardless of the modeling formalism used for the latter, appropriate adaptations of the system according to certain context situations need to be expressed. This requires that the context information provided during runtime by the various context sources is also captured in terms of a context model and in turn the specification of reasoning about these.

Considering modeling of context, one can observe that several (specialized) context modeling approaches have already been proposed in literature stemming from various disciplines like, e.g., adaptive hypermedia systems (cf., e.g., [25]), personalized systems (cf., e.g., [8]), pervasive computing (cf., e.g., [10]), sensor-based systems (cf., e.g., [9]), or geo-systems (cf., e.g., [13]), each

[2] http://www.omg.org/mof/

[3] http://www.omg.org/docs/ptc/05-11-01.pdf

focusing on modeling certain aspects of context relevant to their domain of application, e.g., user model, environmental information, or geographic models. Although some approaches, e.g., [5] or [19] advocate a rather generic knowledge of context-awareness, up to now, currently no single approach to modeling context provides for all relevant contexts in the necessary detail.

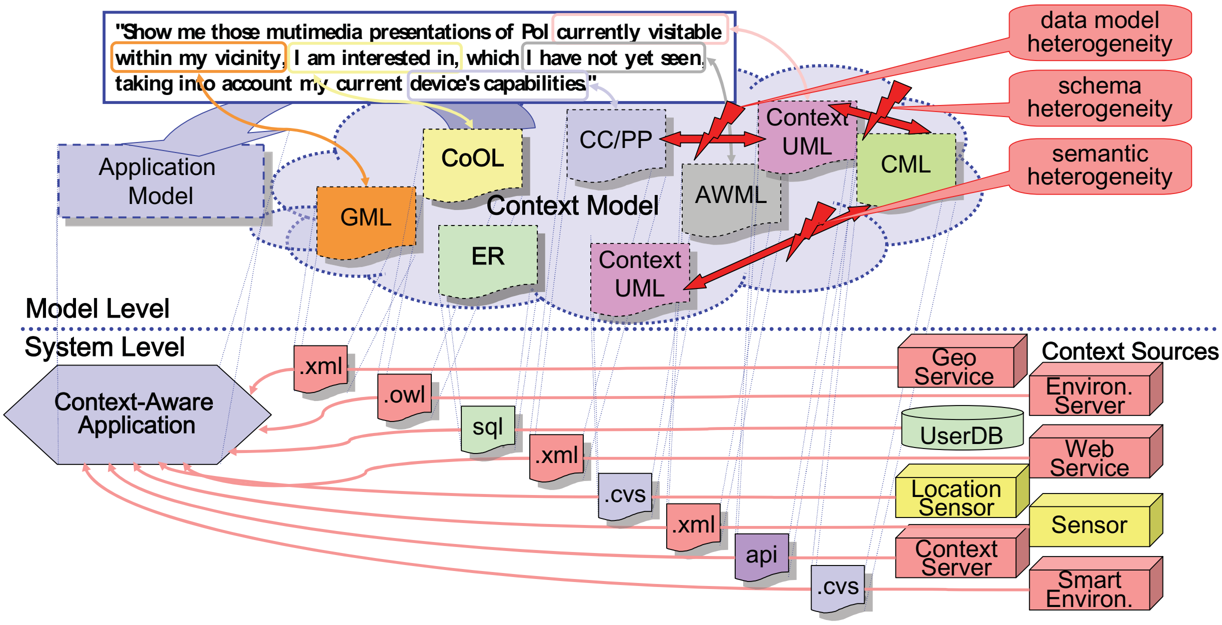

Figure 1: Scenario of different context sources / context models

Nevertheless, some initiatives have come forward with context models not bound to a specific context-aware system. They capture context in terms of generic models, which can be reused in different classes of context-aware systems, opening the possibility of context model reuse.

With respect to the employed formalisms on basis of which context models are captured, one can distinguish in line with [21] and [7]: key-value pair models, e.g., [18], markup-models, e.g., [3], object-oriented models, e.g., [19], [20], relational models [4], graph-based models, e.g., [2], models based on first order logic, e.g., [9], and ontology-based models, e.g., [5], [22] to mention only the most prominent ones. All of them share the goal to provide a high-level abstraction of context information.

2.2. Need for Integration

Despite the fact of various context sources and in turn context models, as the example requirement in Figure 1: explicates, developers of context-aware systems need to reason such that the different context sources / context models are incorporated. However, most of these models lack the ability to be used directly in conjunction with each other, because various types of heterogeneities are encountered.

Specifically, with respect to diverse context models, one faces the problem of the following types of heterogeneities as can be found in data integration scenarios of many domains [25]:

(i) ***Data model heterogeneity,*** since context models are expressed in diverse formalisms, e.g., XML vs. UML-based context models

(ii) ***Structural heterogeneities,*** since the same facts are possibly being expressed differently even when applying the same formalism, e.g., a user's interest can be represented by an attribute of a class *user* or alternatively by a relationship to a class *interest*

(iii) ***Semantic heterogeneities***, since the same concepts can used to represent differently interpreted facts, e.g., different scales for environmental information being applied

Not before taking into consideration these heterogeneities abridging the various context models, a comprehensive reasoning about all context sources is possible.

In a first attempt towards the integration of context models, this paper addresses, as a first step, the issues of data model heterogeneity, which is a prerequisite for being able to tackle structural heterogeneities and semantic heterogeneities in a subsequent step. Therefore, schema and instance level are unattended for the moment.

The abridging can be achieved – as one alternative – within the specification of the reasoning about the diverse context models. This carries the inherent disadvantage that the reasoning specification recurrently needs to abridge the context model's heterogeneities – a tedious and error-prone process that is naturally not desired. This undesired specification during reasoning can be avoided if – alternatively – the reasoning can operate on an integrated context model despite the diverse context sources and context models faced.

In general, two strategies can be employed to provide for integration of context models. On the one hand, it can be approached in a "top-down" manner, re-modeling context information in your formalism of choice whereby later on it has to be taken care of "mapping" to the base formalism of the context source. This admittedly will work for "simple" models like user preferences etc. but certainly not for complex context like for example if a geo-system or a pervasive environment would have to be re-modeled. Additionally, the formalism of choice, by ways of the thereby provided concepts, also determines the extent of context and the point of view from which context can be modeled. On the other hand, taking into consideration what is at hand in terms of models and context sources, one could apply a "bottom-up" strategy, trying to overcome their heterogeneities and thus come up with a comprehensive context model re-using existing ones. The benefit of the latter approach would be that the knowledge of the integration between the various sources is also available for the model-driven development of context-aware systems as a whole.

3. Basic Building Blocks of ModelCVS

Currently we are building *ModelCVS* [11], [12], a semantic infrastructure for model-based integration (cf. Figure 2:). Though tackling a different domain – namely the integration of modelling tools adhering to different modelling formalisms such as UML and ER – the intention is the same, namely supporting integration of models through model engineering techniques. Instead of defining the necessary transformation ad-hoc at model level, ModelCVS advocates an integration of the models' metamodels via specific metamodel integration operators called *bridgings* (1). Bridgings can express, e.g., that some concepts from one metamodel are *equal*, or *conditional equivalent* to concepts of the other metamodel. The definition of bridgings is supported by ModelCVS's Bridging Tool Kit able to be employed on any Ecore-based metamodel[4]. From those bridgings, executable model *transformations* can be derived (2), which can be executed upon the models to be transformed (3) provided that the models are available in terms of an XMI serialization[5].

4 http://www.eclipse.org/emf/

5 http://www.omg.org/technology/documents/formal/xmi.htm

Defining the correct bridgings is a manual task for which expert knowledge is required, but is assisted through ModelCVS by an additional optional component. This component helps to identify possible bridging candidates. For this, ModelCVS advocates the employment of semantic technologies, in terms of ontologies, to enhance the quality of integration in terms of conciseness, effectiveness and reuse of existing integration solutions. It supports the creation of ontologies from metamodels to be integrated, through a semi-automatic so called *lifting* process (cf. 0a in Figure 2:). The lifting is supplemented by refactoring patterns (0b) that can be applied to unfold typically hidden concepts in metamodels that should better be represented as explicit concepts in an ontology. Furthermore, ontologies can be enriched (0c) by putting them into relation with other existing ontologies. Through this lifting of metamodels to ontologies, the gap between the implementation-oriented focus of metamodels and the knowledge representation-oriented focus of ontologies has to be closed. These ontologies capture semantically enriched descriptions of the metamodels allowing to better find bridging candidates, also since existing ontology matching approaches can be utilized. On basis of those mappings and by keeping traces of the lifting process, a semi-automatic derivation of the bridgings (0e) between the original metamodels can be achieved. Nevertheless, the employment of ontologies in ModelCVS is an optional step and, since the focus of the paper is on the bridging of context models, will not be considered further.

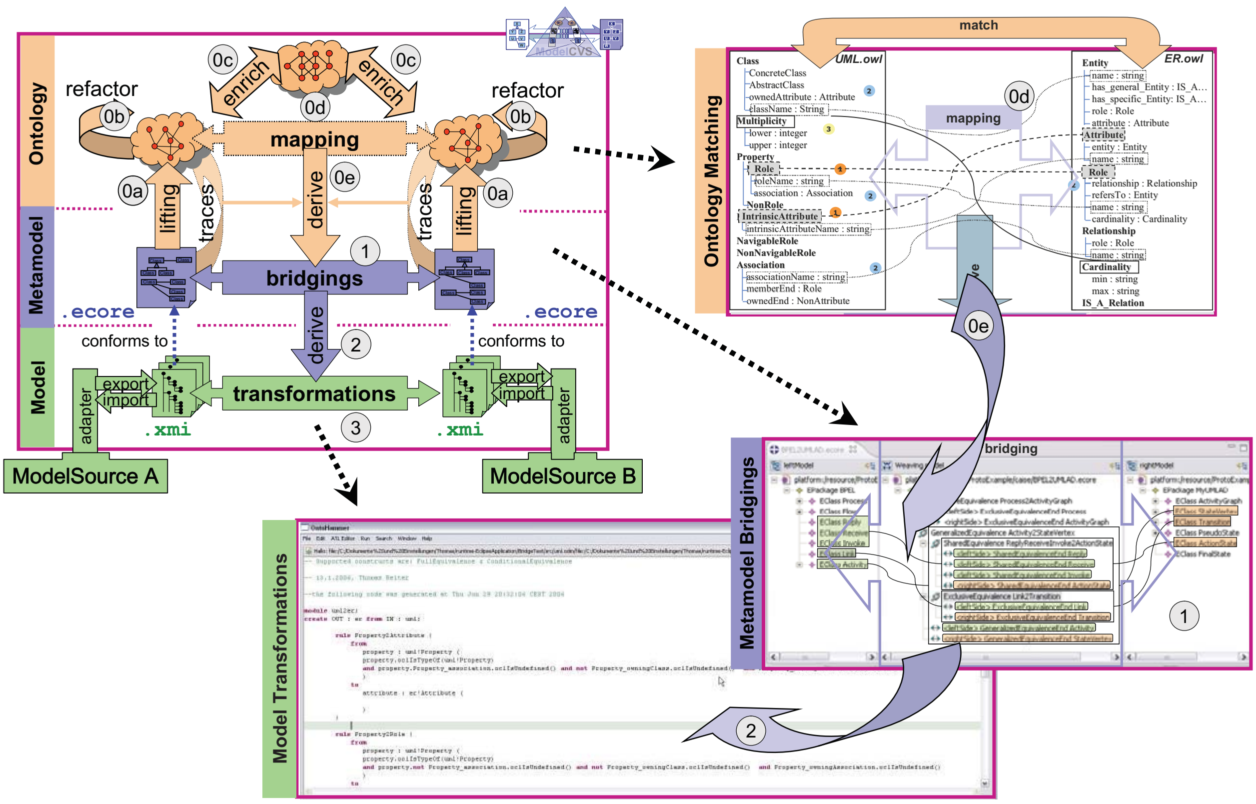

Figure 2: The Overall Architecture of ModelCVS

A first prototype of ModelCVS is already operational being based on ATL[6] and AMW[7] for transforming models and for bridging metamodels, respectively, and Jena[8] for ontology management. Pseudo-ontologies can be refactored on basis of an EMF-based[9] editor which supports

[6] http://www.eclipse.org/m2m/atl/

[7] http://www.eclipse.org/gmt/amw/

[8] http://jena.sourceforge.net

[9] Eclipse Modelling Framework, http://www.eclipse.org/emf

a number of refactoring patterns. For mapping refactored ontologies, an existing tool called COMA++[10] is used. Mappings established at the ontology level can be derived into bridgings of a prototypical bridging language and in turn, by means of ATL transformations, into executable model transformation code.

4. The Approach by Example

In the following we investigate in terms of three representative scenarios the applicability of our ModelCVS architecture for integrating between different context models. The scenarios consider, on the one hand, ContextUML [19], providing for generic context modeling, and on the other hand, OOWS 0 as an approach providing for personalization (Scenario 1), CoOL [22] as an approach for pervasive systems (Scenario 2), and AWML [1] dedicated for spatial systems (Scenario 3). Since being generic, ContextUML is used as formalism for the integrated context model the others should be mapped to (cf. Figure 3:).

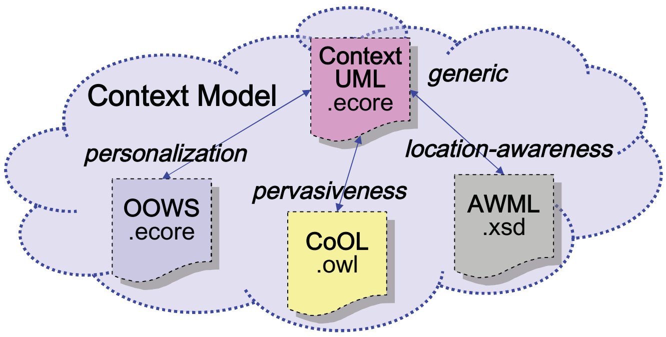

Figure 3: Integrating ContextUML with OOWS, CoOL and AWML

4.1. Scenario 1: ContextUML ↔ OOWS

As a first scenario let us consider the integration of context models formulated in terms of ***ContextUML*** [19] and ***OOWS*** (Object-Oriented Web Solution) 0.

ContextUML is a generic approach providing a UML-based metamodel for modeling context, services and context-aware mechanisms. Modeling context centers on a concept *Context*, a generic representation of context information, which is detailed into *AtomicContext* and *CompositeContext* being composed out of other context. Additional concepts describe the source from which the context is provided in terms of a *ContextSource* and its specific subconcepts as well as concepts describing the binding of context to context-aware objects (*CAObject*). Since ContextUML is UML-based, it draws from Ecore and defines extensions to standard UML by means of UML's profiling mechanism.

OOWS is an approach considering personalization for the model-driven development of web applications. OOWS provides for a role-based user modeling being expressed in terms of OOWS' *User Diagram* allowing the definition of roles of users using sub-classing of *UserRole*. These *UserRole*s in turn allow restricting the accessibility of parts of the web application, i.e. the navigations and tasks users are allowed for. Although OOWS is not based on UML, it is, nevertheless, defined in terms of Ecore and borrows from UML-notation.

[10] http://dbs.uni-leipzig.de/de/Research/coma.html

Since both modeling languages are described in Ecore, the formalism required by ModelCVS, ModelCVS can be employed as depicted in Figure 4: to map between concepts of ContextUML and OOWS by means of bridging operators. Thus, just to give some examples[11], it can be specified that the concept of a *UserRole* used by OOWS, is to be mapped to *AtomicContext* provided by ContextUML at a metamodel level. Furthermore, e.g., *NavigationalContext* of OOWS, which can be constrained by *UserRole*s, can be mapped to the according *CAObject* and *ContextBinding* in ContextUML, respectively. From such mappings, executable ATL-code can be derived as exemplified in Figure 4:.

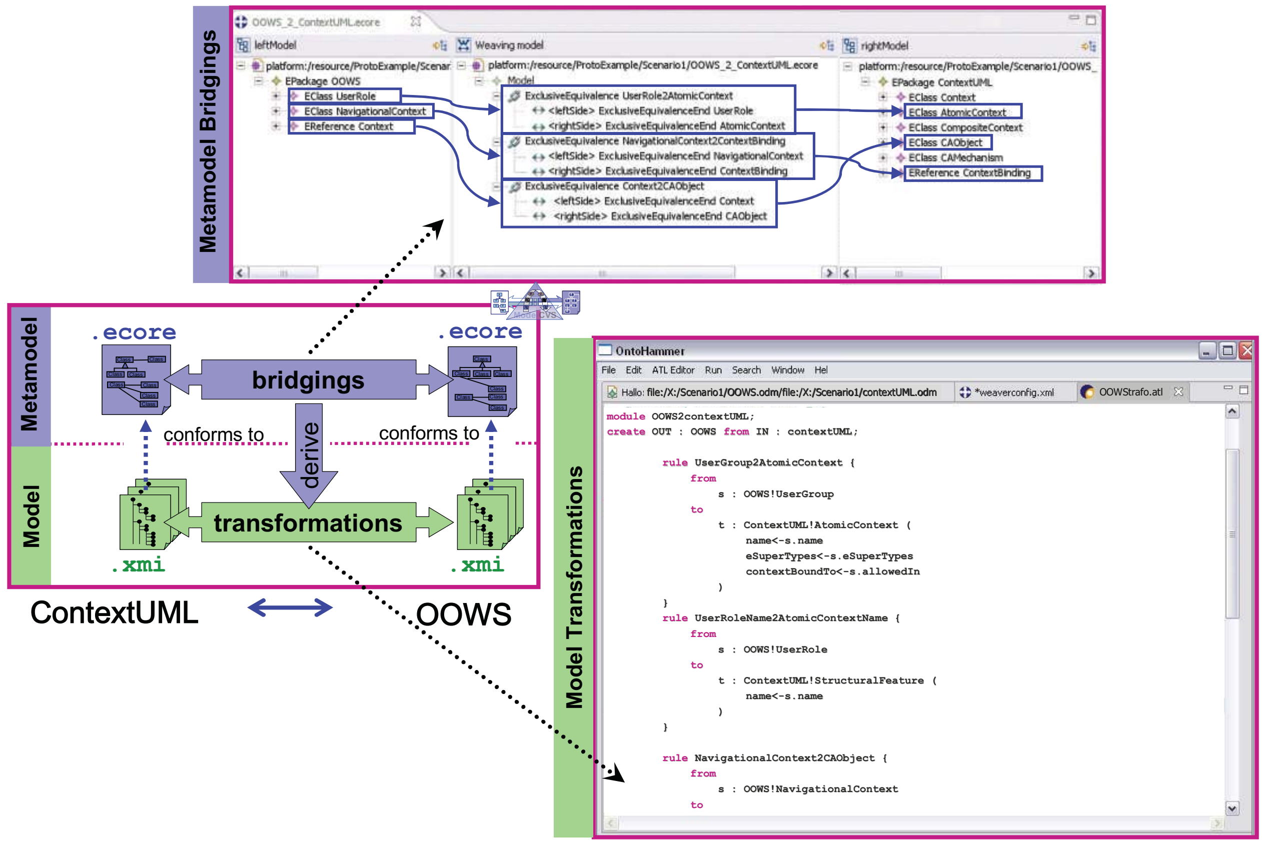

Figure 4: Application of ModelCVS to Transform From OOWS to ContextUML

For being able to actually transform between models of both modeling formalisms, models need to be available in a common representation, namely XMI. Since ContextUML is established in form of a profile, standard UML tools can be employed, most providing for XMI as exchange standard, although it is admitted that some diversity in XMI realizations exists. OOWS in contrast employs its own modeling tool for conceptual modeling relying on dedicated transformations to the solution space. Thus, an actual transformation of OOWS models is dependent on the development of a tool-specific adaptor being able to export and import models in XMI representation.

In this scenario, the integration between ContextUML and OOWS can exploit synergies of different modeling foci. OOWS can supplement ContextUML with a concrete language to model a web-based application whereas ContextUML can provide to OOWS the incorporation of additional context information beyond personalization.

---

[11] Please note, the metamodels in the example have been simplified to increase understanding and consequently some assumptions about the metamodels had to be made, which, however, are not limiting the applicability of our approach.

This scenario is somehow representative for all models relying on UML or object-oriented modeling exposing their concepts in terms of Ecore, like *CMP* [20]. In such type of scenarios ModelCVS, in general, can be employed to firstly identify similarities between model concepts abridging data model heterogeneities and also to automatically generate model transformations, given that models are provided in XMI.

4.2. Scenario 2: ContextUML ↔ CoOL

This scenario comprises the integration of ContextUML and ***CoOL*** (Context Ontology) [22]. CoOL is an ontology for describing and exchanging context in pervasive computing environments based on the Aspect-Scale Context (ASC) model. For example, it provides the concept *Context Information* (i.e., context that can be measured), which characterizes the state of an *Entity* (e.g. person, place) concerning a specific *Aspect* describing the reachable subsets of states. *Context* is the set of all context information characterizing the Entities relevant for a specific task.

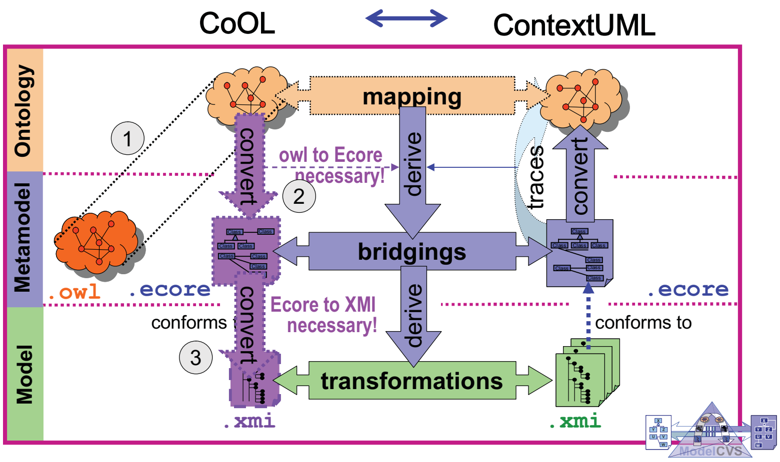

Figure 5: Application of ModelCVS to Integrate ContextUML and CoOL

CoOL is not expressed in terms of Ecore but in terms of OWL, thus, at a first sight, being not able to be handled by ModelCVS for defining the bridges on the metamodel level. Nevertheless, ModelCVS provides for a lifting of metamodels to an ontological level at which the mapping between CoOL and ContextUML can be specified with ModelCVS (cf. Figure 5:). Thus, at the ontological level, it will be possible to identify similar concepts and consequently to map, e.g, CoOL's *Context Information* to ContextUML's *Context*, an *Entity* to *Owner*, an *Aspect* to *Context Constraint*, etc. Nevertheless, the semi-automatic generation of the bridgings is limited since bridgings operate on Ecore-metamodels and lifting traces are not availiable. For the employment, a converter would be necessary to convert the OWL description of CoOL to Ecore and an additional converter as provided for EMF[12] would be necessary to transform CoOL models into XMI.

The findings exemplified in this scenario are equal for other ontologies representing context like, e.g., *CONON* 0. In such cases, ModelCVS is capable of identifying similarities but is limited with respect to automatic transformations.

4.3. Scenario 3: ContextUML ↔ AWML

In the final scenario the ***Augmented World Modeling Language (AWML)*** [1] shall be integrated with ContextUML. AWML is an object-oriented data model for location-based information systems

[12] http://www.eclipse.org

in the realm of the Nexus project[13]. With AWML it is possible to specify the spatial location (*Nexus Object Locator*) of objects (*SpatialObject*). Models are represented by means of XML, and thus described at the metamodel level using XML-schema[14].

Likewise the previous scenario, additional conversions have to be conducted to be able to draw from ModelCVS' mapping capabilities (cf. Figure 6:). It is necessary, first, to convert the XML-schema description into an Ecore-based description to be able to define the bridgings and, second, to convert the individual models from their XML-based representation to XMI. For both, existing converters available for EMF can be employed.

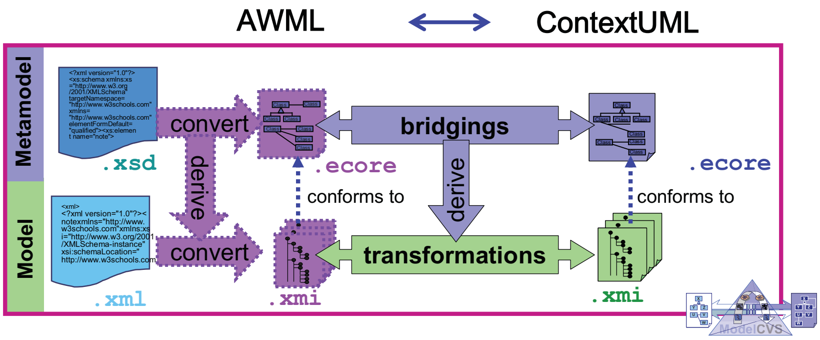

Figure 6: Application of ModelCVS to Integrate ContextUML and AWML

This scenario is again prototypical for scenarios where XML is used to express context models (cf., e.g., [17]). In such cases, conversions are required before either similarities can be identified or any transformation between models can be established at all.

5. Related Work

Though information integration in general has drawn attention already some time ago (cf. [16], [25]), the integration of different context models, due to the infancy of context-aware systems, has only recently been regarded.

Nevertheless, an approach sharing similar intentions has been proposed by *De Virgilio et al.* [6]. Similarly to our approach, it aims at overcoming the variety of context models by means of a rule-based transformation process that transforms RDF-based models on bases of RDF-schema mapping specifications to a dedicated reference context model. However, our approach is more flexible, since ModelCVS is not limited to a dedicated formalism for the transformation between models and can be employed on any Ecore based formalism employing standards in the field of model-driven development. Also relying on a single dedicated model are the approaches of *Ou et al.* [30] and *Blackstock et al.* 0, deferring the integration to a later phase by means of platform independent model to platform specific model transformations and dedicated adaptors, respectively. Our approach tackles the problem differently in that it allows integration in a model-to-model fashion and does not enforce a single universal model. ModelCVS may complement these approaches by enabling to integrate various context models, also those not based on ontologies, to their single dedicated model.

[13] http://www.nexus.uni-stuttgart.de/

[14] http://www.nexus.uni-stuttgart.de/de/forschung/dokumente/information-exchange/NexusAwmlSchema.xsd

With respect to model integration in general *Rondo* 0 provides high-level operators facilitating the integration of relational schema and XML schema. In the modeling realm, *Clarke* 0 and *Straw* 0 propose approaches for model composition, but both are limited to UML-based models only.

6. Summary and Outlook

Context-aware systems require context information, which is likely provided by various context sources. Consequently, when developing such context-aware systems in a model-driven way, the modeling of context is a prerequisite. Regarding the different sources with its diverse formalisms, integration of context models is required. This paper pursues the vision of a model-level integration of context models. For this, it envisions the employment of ModelCVS, an architecture for model-based integration. In terms of three representative example scenarios, this paper has discussed the applicability and limitations of ModelCVS to overcome data model heterogeneities as a first necessary step to achieve integration of context models.

Future work will focus on overcoming the current limitations of ModelCVS as laid out. Specifically, it will be necessary to provide for OWL to Ecore converters to incorporate ontology-based context models as well as converters to describe XML-Schemas in terms of Ecore. On bases of those, the applicability of ModelCVS to actually transform models can be demonstrated. Since current experimentation with ModelCVS has reviled some limitations with respect to specifying the bridging of more complex metamodel heterogeneities, currently model-operators for ModelCVS are developed to allow for easier definition of the necessary bridgings.

7. Acknowledgement

This work has been partly funded by the Austrian Federal Ministry of Transport, Innovation and Technology (BMVIT) and FFG under grant FIT-IT-810806. The author whishes to thank the members of the ModelCVS-project for their valuable input and discussion.

8. References

[1] Nicklas D., Mitschang B.: "Contact Information On building location aware applications using an open platform based on the NEXUS Augmented World Model", Journal Software and Systems Modeling, 3/ 4, December, 2004.

[2] Brézillon, P.: "Context-based Modeling of Operators’ Practices by Contextual Graphs". Proceedings of the 14th Mini-Euro Conference on Human Centered Processes. Euro, 2003.

[3] Buchholz S., Hamann T., Hübsch G.: "Comprehensive Structured Context Profiles (CSCP): Design and Experiences". Proc of the 2nd IEEE Conf. on Pervasive Computing and Communications Workshops, IEEE Computer Society, Washington, DC, March 2004.

[4] Ceri S., Daniel F., Matera M., Facca F. M., "Model-driven Development of Context-Aware Web Applications". ACM Transactions on Internet Technology (TOIT), Volume 7, Issue 1, February 2007.

[5] Chen H., Perich F., Finin T.; Joshi A.: "Mobile and Ubiquitous Systems: Networking and Services". Proc. of the Int. Conf. on Mobile and Ubiquitous Systems: Networking and Services, Boston, USA, August 2004.

[6] De Virgilio R. and Torlone R. 2006. Modeling heterogeneous context information in adaptive web based applications". Proc. of the 6th Int. Conf. on Web Engineering, ACM Press, Palo Alto, California, USA, July, 2006.

[7] De Farias C. R., Leite M. M., Calvi C. Z., Pessoa R. M., Filho, J. G.: "A MOF Metamodel for the Development of Context-Aware Mobile Applications". In Proc. of the ACM Symp. on Applied Comp., ACM Press, Seoul, 2007.

[8] Goker A. and Myrhau H. I.: "User Context and Personalisation". Proc. of the Workshop on Case Based Reasoning and Personalisation, Aberdeen, UK, 2002.

[9] Gray P. D. and Salber D.: "Modelling and Using Sensed Context Information in the Design of Interactive Applications". Proc. of the 8th Int. Conf. Engineering for Human-Computer Interaction, LNCS 2254, Toronto, 2001.

[10] Henricksen, K., Indulska, J. and Rakotonirainy, A.: "Modeling Context Information in Pervasive Computing Systems". Proc. of the Int.l Conf. on Pervasive Computing, LNCS 2414, Zurich, Switzerland, August 2002.

[11] Kappel G., Kapsammer E., Kargl K., Kramler G., Reiter Th., Retschitzegger W., Schwinger W., Wimmer M.: "On Models and Ontologies - A Layered Approach for Model-based Tool Integration". Proc. of Modellierung, Innsbruck, March 2006.

[12] Kappel G., Kapsammer E., Kargl K., Kramler G., Reiter Th., Retschitzegger W., Schwinger W., Wimmer M.: "Lifting Metamodels to Ontologies: A Step to the Semantic Integration of Modeling Languages". Proc. of the ACM/IEEE Int. Conf. on Model Driven Engineering Languages and Systems, Genova, Italy 2006.

[13] Lehmann O., Bauer M., Becker Ch., Nicklas D.: "From Home to World - Supporting Context-aware Applications through World Models". Proc. of the 2nd Int. Conf. on Pervasive Comp. and Communications, USA, March 2004.

[14] Mikalsen M., Paspallis N., Floch J., Stav E., Papadopoulos G. A.., Chimaris A.: "Distributed Context Management in a Mobility and Adaptation Enabling Middleware (MADAM)". Proceedings of the 2006 ACM Symposium on Applied Computing (SAC), ACM 2006, ISBN 1-59593-108-2, Dijon, France, April 23-27, 2006.

[15] Paspallis N., Chimaris A. Papadoupoulos G. A.: "Experiences from Developing a Distributed Context Management System for Enabling Adaptivity". Proc. of the Int. Conf. on Distributed Applications and Interoperable Systems, LNCS 4531 Paphos, Cyprus, June, 2007.

[16] Rahm E., Bernstein P.A.: "A Survey of Approaches to Automatic Schema Matching". VLDB Journal 10/4, 2001.

[17] Robinson, R., Henricksen, K., and Indulska, J.:"XCML: A Runtime Representation for the Context Modelling Language". Proc. of the Int. Conf. on Pervasive Computing and Communications Workshops, IEEE Computer Society, Washington, DC, March, 2007.

[18] Schilit, W. N., Adams, N. I. and Want, R.: "Context-Aware Computing Applications". Proc. of the Workshop on Mobile Computing Systems and Applications. IEEE Computer Society, 1994.

[19] Sheng Q. Z., Benatallah B.: "ContextUML: A UML Based Modeling Language for Model-driven Development of Dontext-aware Web Services". Proc. of the Int. Conf. on Mobile Business, Sydney, Australia, July 2005.

[20] Simons, C.: "CMP: A UML Context Modeling Profile for Mobile Distributed Systems". Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS 2007), Hawaii, USA, Jan 3-6, 2007.

[21] Strang, T. and Linnhoff-Popien, C., "A Context Modeling Survey". Proc. of the 1st International Workshop on Advanced Context Modeling, Reasoning and Management (UbiComp 2004), Nottingham, England, Sept. 2004.

[22] Strang T., Linnhoff-Popien C., Frank K. "CoOL: A Context Ontology Language to Enable Contextual Interoperability. In Proceedings of the 4th International Conference on Distributed Applications and Interoperable Systems (DAIS), volume 2893 of Lecture Notes in Computer Science, Springer, 2003.

[23] Mikalsen, M., Paspallis, N., Floch, J., Stav, E., Papadopoulos, G. A., and Chimaris, A.: "Distributed Context Management in a Mobility and Adaptation Enabling Middleware (MADAM)! In Proceedings of the 2006 ACM Symposium on Applied Computing, SAC '06. ACM Press, Dijon, France, April, 2006.

[24] W3C: Composite Capabilities/Preference. Profiles, http://www.w3.org/Mobile/CCPP/, 2007.
[25] Paul De Bra, Wolfgang Nejdl: Adaptive Hypermedia and Adaptive Web-Based Systems, Third International Conference, AH 2004, Eindhoven, The Netherlands, August 23-26, 2004, Proceedings Springer 2004.

[26] Shet A. P., Larson J. A.: "Federated Database Systems for Managing Distributed, Heterogeneous and Autonomous Databases", ACM Computing Surveys, 22(3), September 1990.

[27] Pastor O., Fons J., Pelechano V:. "OOWS: A method to develop web applications from web-oriented conceptual models". Proceedings of the International Workshop on Web Oriented Software Technology (IWWOST), 2003.

[28] Wang, X. H., Zhang, D. Q., Gu, T., and Pung, H. K. 2004. Ontology Based Context Modeling and Reasoning using OWL. In Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (March 14 - 17, 2004). PERCOMW. IEEE Computer Society, Washington, DC, 18.

[29] Blackstock M., Lea R., Krasic C.: "Toward a Shared Model for Wide Area Interoperability of Ubiquitous Computing Environments". Proc. of the System Support for Ubiquitous Computing Workshop, USA, Sep. 2006.

[30] Ou S., Georgalas N., Azmoodeh M., Yang K., Sun X.: "A Model Driven Integration Architecture for Ontology-Based Context Modelling and Context-Aware Application Development". Proc. of the 2nd European Conference Model Driven Architecture - Foundations and Applications, (ECMDA-FA 2006), LNCS, Bilbao, Spain, July, 2006.

[31] Melnik S., Rahm E., Bernstein P. A.: "Rondo: a programming platform for generic model management". Proc. of the ACM SIGMOD International Conference on Management of Data, ACM Press, New York, USA, June 2003.

[32] Clarke S.: "Extending standard UML with model composition semantics". Science of Computer Programming, Elsevier Science, 44(1), July 2002.

[33] G. Straw et al.: Model Composition Directives, Proc. of the 7th UML Conference, Lisbon, Portugal, October 2004.