

Ontology Change in Ontology-Based Information Integration Systems

Fajar Juang Ekaputra^(✉)

Institute of Software Technology CDL-Flex, Vienna University of Technology,
Favoritenstrasse 9-11/188, 1040 Vienna, Austria
fajar.ekaputra@tuwien.ac.at

Abstract. Ontology change is an important part of the Semantic Web field that helps researchers and practitioners to deal with changes performed in ontologies. Ontology change is especially important in Ontology-Based Information Integration (OBII) systems, where several ontologies are interrelated and therefore, changes raise various complexities and implications, such as modifications of ontology mappings and change propagation. Current approaches to ontology change mainly focus on a single ontology and therefore do not properly address the constraints specific to OBII systems. To address the challenge of ontology change in OBII contexts, we plan to adapt successful techniques proposed both by Semantic Web and Model-Driven Engineering communities. We discuss the research goals, methods, and evaluation options to address this challenge. Real-world case studies are used for the development and evaluation of the proposed methods.

Keywords: Ontology change · Ontology evolution · Ontology versioning · Ontology-Based information integration · Model-Driven engineering

1 Context and Motivation

The notion of using ontologies for information integration has been applied for around two decades [1]. Wache et al. have reviewed various approaches for Ontology-Based Information integration (OBII) that use ontologies to integrate information from multiple heterogeneous sources [2], while Calvanese et al. explain typical components of an OBII consisting of a shared (common) ontology, a number of local ontologies, and mappings between the local and common ontologies [3].

Such an integration raises issues on how to maintain the integrated system, i.e., how to manage the ontology change within the system. Ontology change support is an important requirement of an OBII system, especially in a software-intensive system such as business information and industrial automation [4] where change support is often needed to deal with changes in ontology schemas (i.e., T-Box) and data (i.e., A-Box). These changes have to be validated, applied and propagated to all relevant

F.J. Ekaputra—Christian Doppler Research Laboratory “Software Engineering Integration for Flexible Automation Systems”; <http://cdl.ifs.tuwien.ac.at>.

parts of the system to ensure its consistency. “**Ontology Change**”, as described by Flouris et al. [5], refers to “the problem of deciding the modifications to perform upon ontologies in response to a certain need for a change as well as the implementation of these modifications and the management of their effects in depending data, services, applications, agents or other elements”. In the OBII context, we define “change propagation” as part of ontology change process that deals with the management of the ontology change effects.

The current approaches to deal with ontology change from the Semantic Web community are mostly focused on a single ontology and therefore these approaches are not sufficient to support ontology changes in an OBII system that consist of multiple heterogeneous ontologies with complex mappings. Also, the Model-Driven Engineering (MDE) community has proposed the notion of model co-evolution for expressing composite changes within a model and for propagating these changes to re-establish global consistency [6, 7]. As part of our research, we aim to investigate whether and how MDE co-evolution approach could be adapted to ontology change in the OBII settings.

In our research, we plan to address the gap of ontology change support in OBII systems by identifying, defining and developing the required mechanisms and methods, while studying and adapting appropriate approaches both from Semantic Web and MDE communities. As the first step towards that goal, we have studied the literatures from both communities and identified requirements for ontology change within OBII systems from two case studies: (1) **Power Plant Design** from the Industrial Automation System (IAS) domain and (2) **Integration System for Scholarly Data** of the domain of Empirical Software Engineering (EMSE). These initial requirements from the case studies will be discussed within Sect. 5.

The rest of the paper is structured as follows: Sub-Sect. 1.1 illustrates an OBII problem setting example. Section 2 defines the state of the art in ontology change and model evolution. Section 3 provides the problem statement and motivates the research contribution. Section 4 derives the research approach and Sect. 5 shows the preliminary results. We introduce our evaluation plan in Sect. 6, and conclude the paper in Sect. 7.

1.1 An Ontology-Based Information Integration System Example

As an illustrating OBII system example, let us take a look on the case of information integration in a modern power-plant planning from the IAS domain. Figure 1 shows the simplified representation of the system that mainly consists of the following three elements (marked with the corresponding numbers in the figure):

- (1) **Local Ontologies** represent the data from the different tools and domains involved in a power plant system engineering. In the example, the local ontologies come from mechanical, electrical and software engineering.
- (2) **The Common Ontology** represents the aggregation of important concepts (e.g., signal and CPU) of local ontologies from the perspective of the system stakeholders. In the example, the common ontology is the global ontology of the power-plant.

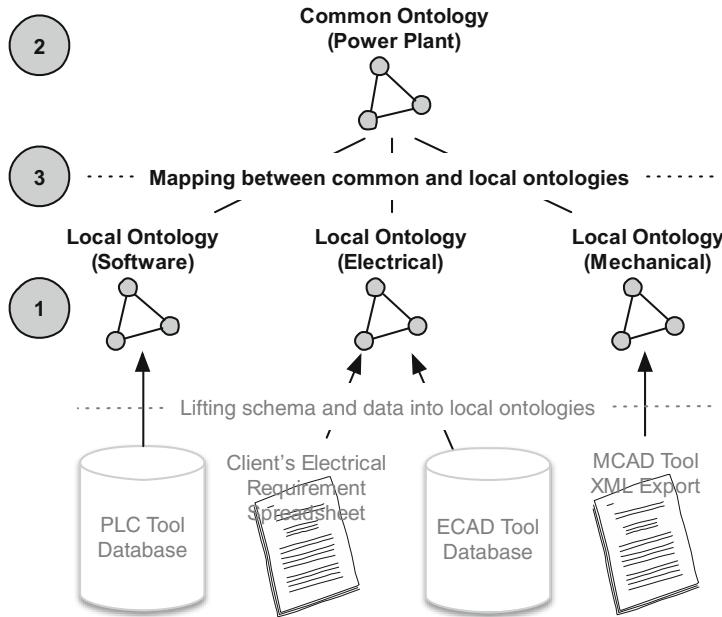


Fig. 1. Components of IAS power-plant design system

- (3) **Mapping between Common and Local Ontologies** represent the semantic overlaps between local and common ontologies.

In the setting shown in Fig. 1, the *mechanical* engineers will use their domain-specific tool like MCAD¹ to design the machinery. In parallel, *electrical* engineers use their specialized tool such as *EPlan*² to design the wiring for each machine and the connections between machines. There are also *Programmable Logic Controller* (PLC) software components to support the production automation, created by *software* engineers. In between the process, clients will provide feedback to the engineers regarding their adjustment of requirements.

Since the specific tools from each domain are typically proprietary, the means to exchange data are limited to tool data export (e.g., spreadsheet, XML) or database views. Ideally, the exported data or database views would be lifted into local ontology and then be mapped with the global ontology, in line with the principles of the Global-as-View approach [8]. The mapping between local and global ontologies may be not straightforward, e.g., concatenation or computation of two attributes values in local ontology will be mapped to one attribute value in the global ontology. Important challenges in such environment are: (1) change process identification, i.e., to identify the necessary change processes for performing ontology changes within the environment, e.g., change propagation, where changes in one element should be propagated to

¹ <http://www.solidworks.com/sw/products/3d-cad/packages.htm>.

² <http://www.eplanusa.com/us/solutions/electrical-engineering/eplan-electric-p8/>.

relevant ontologies; (2) change detection and representation, i.e., how to detect and represent the changes of local and global ontologies with their mappings; and (3) change tool support, i.e., how to adequately provide tool support to perform ontology change within the environment. The common concepts that relate two or more domains, such as Signal and CPU (Software, Electrical domain) will provide additional complexity since a change in one domain could affect different parts of the system.

2 State of the Art

In this section, we revisit the state of the art from two research communities that are relevant for our research, namely ontology change from Semantic Web and model evolution from MDE.

2.1 Ontology Change

Flouris et al. provide an excellent summary of many ontology change terms that are used in the Semantic Web community [5]. We refer to their definition throughout our work. Two of the most important terms are: (1) **Ontology evolution**, a process of modifying an ontology in response to a certain change in the domain or its conceptualization, and (2) **Ontology versioning**, an ability to handle an evolving ontology by creating and managing different variants/versions of this ontology.

Change Process. Recent work from Zablith et al. [9] has summarized major ontology evolution process approaches [10–14]. They proposed five steps for the ontology evolution process: (1) Detecting Evolution Need, (2) Suggesting Changes, (3) Validating Changes, (4) Assessing Impact, and (5) Managing Changes. These processes are designed with the focus on the changes in an ontology schema. A closer approach to our research comes from Papavassiliou et al. They take into account changes both in the ontology schema and data [15]. However, these approaches mainly consider changes in a single ontology instead of in the context of OBII systems, which presents the challenge of complex changes and its propagation across the system. We identify this as a gap and we plan to include the solution as part of our ontology change process definition.

Change Detection and Representation. A major requirement for the ontology change in OBII is the ability to detect low-level (i.e., addition and deletion of triples) changes and high-level (e.g., concept move and deletion) changes between different ontology versions [15] and represent them in a machine readable formats for future analytics, while considers their effects on the change propagation process. To address change detection between two ontology versions, the use of heuristics algorithms [16], structural differences [15, 17], and OWL reasoning [18] have been proposed and evaluated. To support the change detection mechanism, approaches for change representation as change ontologies [11, 19] and change languages [20] have also been

proposed. Similar to ontology change process, the approaches in this area are typically focused on detecting changes of a single ontology. In our research, we aim to build on the state of the art of ontology change detection and representation to detect and represent low-level changes and selected sets of high-level changes of OBII system ontologies and their complex mappings.

Change Tool Support. To provide tool support for ontology change, an initial set of requirements that focused on ontology evolution was introduced by Stojanovic and Motik for the KAON tool [21]. In the similar timeframe, *PrompDiff* change detection algorithm was integrated into *Protégé* tool [16]. Later on, Noy et al. introduced support for different scenarios of ontology editing in *Protégé*, providing background support for storing ontology metadata using CHAO vocabulary [11]. The latest addition to the impressive set of *Protégé* ontology change support tools is the *Protégé* versioning server,³ which is based on the previously proposed architecture client-server architecture [22]. An interesting line of work comes from the adaptation of distributed versioning systems, *SemVersion* [23] and *R&Wbase* [24], which provide support for ontology versioning similar to source code versioning systems. However, the aforementioned tools are mainly designed to work with a single ontology and therefore, they do not fully address important requirements of an OBII system, such as change propagation to provide global consistency across the system.

2.2 Model Evolution

The evolution aspects of MDE are getting more attention with the growing importance of modeling in software development [25]. Various tools for supporting model refactoring have been proposed in the literature [26, 27]. Unlike change in ontologies, which focused primarily on single ontology scenarios, MDE research has a richer set of approaches in the area of model co-evolution [6, 7] and model change analysis (e.g., composite change detection [28] and change sequence identification [29]). However, these approaches have not been studied in the context of OBII. As part of our research, we will investigate whether and how co-evolution and change analysis algorithms could be adapted to ontology change in OBII settings.

3 Problem Statement and Contribution

In order to address the challenge explained in Sect. 1, we identify the main research question (RQ) of our work as follow: “*Which mechanisms and methods are required to cope with ontology change in an Ontology-Based Information Integration (OBII) system, where ontologies are used as means for data integration for heterogeneous data sources?*”

This main research question is refined into smaller research questions as follows:

³ http://protegewiki.stanford.edu/wiki/Protege_5_Development_Environment.

RQ1: What kinds of ontology changes, process and analysis are required in an OBII system? What are the specific characteristics of ontology change in such a system? The research question aims for the identification and description of ontology change types (e.g., mapping changes), change processes (e.g., change propagation), and change analytics (e.g., detection of complex composite changes) that are needed in an OBII system. The answer to the question should be based on a comprehensive interpretation of an OBII system from the ontology change perspective.

RQ2: What methods and techniques can solve problems caused by ontology change in an OBII system? Do model co-evolution methods from MDE provide relevant solution alternatives? In order to solve the challenge identified in the RQ2 while consider the specific characteristic that answer RQ1, what kind of methods and techniques are necessary? Are the current available techniques and technologies from both the Semantic Web and the MDE communities able to address the requirements? We expect that the answer for this research questions could also useful in more general environments as alternative to the current state of the art.

4 Research Methodology and Approach

Within this research we strive to solve how to deal with ontology change within OBII system. The proposed research will be developed within the design science methodology, using the regulative cycle as conceptual framework and, specifically, following the guidelines provided by Wieringa in [30]. At the moment, we are in the first phase, focusing on gathering the requirements and conducting the literature study, to provide a solid ground for further steps.

- **The first phase in this methodology is the problem investigation,** focusing on the analysis of the problems that this research should confront as well on the properties that a good design solution should have. We have done preliminary literature studies and interviews with domain experts from two application domains. We have to look into the data and identify sets of changes that often happen in the system, e.g., frequent instance changes and change propagation over an OBII system. Next, we are planning do a comprehensive literature research on ontology change and related fields (e.g., MDE model co-evolution and MDE change detection algorithms). The goal of this phase is to define a set of requirements and the associated evaluation metrics that will be used throughout the research.
- **The second phase is the solution design.** This phase explores the possibility of applying or adapting the currently available approach to design a good solution. The process includes the realization of manual and alternative approaches to address the challenges, including those coming from the MDE community and evaluating the feasibility of those approaches. Based on requirements, literature studies, and manual approach realizations results, the goal of this phase is to create a solution design as a conceptual framework that will define the required ontology changes, processes, and analytics necessary for the OBII context.
- **The third phase is the solution design validation.** This phase aimed to check the solution design against the previously identified requirements and gather feedbacks

for the proposed approach before the implementation phase. We plan to involve relevant researchers for checking the logical feasibility of our approach, while providing the experts in our application domains with a conceptual prototype of our solution (e.g., scenario, persona) to collect and analyze their feedbacks.

- **The fourth phase is the implementation of the solution.** In this phase, we plan to realise the solution design within the selected application domains by implementing the prototype according to our solution design and the case study scenario definition.
- **The fifth phase evaluates the solution in the selected scenarios.** This phase aims to find out how well the solution approach worked in real-world settings. The validation process will be carried out as case-studies in selected application domains.

5 Preliminary Results

In order to capture the requirements for ontology change in OBII systems, we have conducted semi-structured interviews (according to [31]) with domain experts from two different application domains, IAS and EMSE, as mentioned in Sect. 1. We analyzed the interview results and take into account the current state of the art in the research.

From IAS domain experts, we learned that they deal with daily updates of instance changes and three to four data model changes per year. They would like to be able to store versions for instance and ontology changes, and to be able to query the ontology change history within a selected time period. We sum up the requirements from IAS domain as follows:

- Ontology evolution and change propagation.
- Ontology versioning
- Query across changed data and metadata/analytics of changes.
- System scalability (to millions of data instances)

From EMSE domain experts, we learned that, in spite of slight differences, the core issues for ontology change are similar to the IAS domain. They have less frequent changes compared to the IAS. Additionally, the contribution for the knowledge changes are less structured and dependent to the EMSE community activity. The requirements from EMSE community can be summarized as follows:

- Ontology versioning.
- Query across changed data and metadata for concept tracing
- (Semi-)automated query updates based on data model changes (as typical domain experts are not familiar with Semantic Web technology).
- (Semi-)automated ontology debugging support
- Ontology mapping across different instances of EMSE knowledge bases.

6 Evaluation Plan

We plan to base our evaluation approach on two case studies in the domains of IAS and EMSE. We will follow the case-study-based evaluation presented in [32] to validate our approach. Following the research methodology that we have defined in Sect. 4, we plan two separate evaluations in our research.

- **Evaluation of the solution design.** Within this step, we will validate our design with the stakeholders, including relevant scientific researchers and application domain experts. The evaluation goal is to provide a clear view on how the solution design will answer the set of stated requirements and the associated evaluation metrics.
- **Evaluation of the solution implementation.** The purpose of this evaluation is to validate whether the proposed solution works better in addressing the requirement of ontology change of an OBII system compared to the alternative solutions. We plan to build our implementation according to our solution design and use the evaluation metrics for comparison with solution implementation alternatives. Afterwards, we plan to do user studies to check the solution implementation in real-world settings according to our application domain case studies.

7 Conclusions

In this work, we have provided an overview on the research plan for addressing challenges of ontology change of an OBII system. We positioned our work within the state of the art, explained the problem statement and our planned contribution, and proposed an approach to address the challenges.

As part of the research, we are currently in close cooperation with stakeholders from the domains of our two planned case studies. This advantage allows us to receive better feedback regarding the requirements and to take more informed and better decisions regarding the design and development of our solution. These success factors are likely lead us to get better results. We hope that our research will bring advantages to both research communities and industry partners.

Acknowledgements. This work was supported by the Christian Doppler Forschungsgesellschaft, the Federal Ministry of Economy, Family and Youth, Österreichischer Austauschdienst (ÖAD) and the National Foundation for Research, Technology and Development, Austria. I want to thank my supervisors, Prof. Stefan Biffl, Dr. Estefanía Serral, and Dr. Marta Sabou, the ESWC mentor, Prof. Steffen Staab, and anonymous reviewers for their helpful comments and feedbacks to improve this paper.

References

1. Uschold, M., Gruninger, M.: Ontologies: principles, methods and applications. *Knowl. Eng. Rev.* **11**, 93–136 (1996)
2. Wache, H., Voegele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., Hübner, S.: Ontology-based integration of information-a survey of existing approaches. In: IJCAI 2001 Workshop: Ontologies and Information Sharing, pp. 108–117 (2001)
3. Calvanese, D., De Giacomo, G., Lenzerini, M.: Ontology of integration and integration of ontologies. In: International Description Logics Workshop, p. 2 (2001)
4. Moser, T., Biffl, S.: Semantic tool interoperability for engineering manufacturing systems. In: Proceedings of 15th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2010), pp. 1–8 (2010)

5. Flouris, G., Manakanatas, D., Kondylakis, H., Plexousakis, D., Antoniou, G.: Ontology change: classification and survey. *Knowl. Eng. Rev.* **23**, 117–152 (2008)
6. Cicchetti, A., Di Ruscio, D.: Automating co-evolution in model-driven engineering. In: Enterprise Distributed Object Computing Conference, pp. 222–231 (2008)
7. Wimmer, M., Moreno, N., Vallecillo, A.: Viewpoint co-evolution through coarse-grained changes and coupled transformations. In: Furia, C.A., Nanz, S. (eds.) *TOOLS 2012*. LNCS, vol. 7304, pp. 336–352. Springer, Heidelberg (2012)
8. Ullman, J.D.: Information integration using logical views. *Theor. Comput. Sci.* **239**, 189–210 (2000)
9. Zablith, F., Antoniou, G., D'Aquin, M., Flouris, G., Kondylakis, H., Motta, E., Plexousakis, D., Sabou, M.: Ontology evolution: a process-centric survey. *Knowl. Eng. Rev.* **30**, 45–75 (2015)
10. Klein, M., Noy, N.: A component-based framework for ontology evolution. In: Proceedings of IJCAI (2003)
11. Noy, N.F., Chugh, A., Liu, W., Musen, M.A.: A framework for ontology evolution in collaborative environments. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) *ISWC 2006*. LNCS, vol. 4273, pp. 544–558. Springer, Heidelberg (2006)
12. Zablith, F.: Evolva: a comprehensive approach to ontology evolution. In: Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., Simperl, E. (eds.) *ESWC 2009*. LNCS, vol. 5554, pp. 944–948. Springer, Heidelberg (2009)
13. Vrandecic, D., Pinto, S., Tempich, C., Sure, Y.: The DILIGENT knowledge processes. *J. Knowl. Manag.* **9**, 85–96 (2005)
14. Stojanovic, L.: Methods and tools for ontology evolution (2004). <http://d-nb.info/1001606787/34/>
15. Papavassiliou, V., Flouris, G., Fundulaki, I., Kotzinos, D., Christophides, V.: On detecting high-level changes in RDF/S KBs. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) *ISWC 2009*. LNCS, vol. 5823, pp. 473–488. Springer, Heidelberg (2009)
16. Noy, N., Musen, M.: Promptdiff: a fixed-point algorithm for comparing ontology versions. In: AAAI/IAAI (2002)
17. Redmond, T., Noy, N.: Computing the changes between ontologies. In: Joint Workshop on Knowledge Evolution and Ontology Dynamics, pp. 1–14 (2011)
18. Gröner, G., Silva Parreiras, F., Staab, S.: Semantic recognition of ontology refactoring. In: Patel-Schneider, P.F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J.Z., Horrocks, I., Glimm, B. (eds.) *ISWC 2010*, Part I. LNCS, vol. 6496, pp. 273–288. Springer, Heidelberg (2010)
19. Palma, R., Haase, P., Corcho, O., Gómez-Pérez, A.: Change representation for OWL 2 ontologies (2009)
20. Papavassiliou, V., Flouris, G., Fundulaki, I., Kotzinos, D., Christophides, V.: Formalizing high-level change detection for RDF/S KBs. Technical report TR-398, FORTH-ICS (2009). <http://users.ics.forth.gr/~fgeo/Publications/TR398.pdf>
21. Stojanovic, L., Motik, B.: Ontology evolution within ontology editors. In: Proceedings of OntoWeb-SIG3 Work (2002)
22. Redmond, T., Smith, M., Drummond, N., Tudorache, T.: Managing change: an ontology version control system. In: OWLED (2008)
23. Völkel, M., Groza, T.: SemVersion: an RDF-based ontology versioning system. In: Proceedings of the IADIS International Conference WWW/Internet, p. 44 (2006)

24. Vander Sande, M., Colpaert, P., Verborgh, R., Coppens, S., Mannens, E., Van de Walle, R.: R & Wbase: git for triples. In: Linked Data on the Web Workshop (2013)
25. France, R., Rumpe, B.: Model-driven development of complex software: a research roadmap. In: International Conference on Software Engineering: Future of Software Engineering (2007)
26. Moha, N., Gueheneuc, Y.: DECOR: a method for the specification and detection of code and design smells. *Trans. Softw. Eng.* **36**, 20–36 (2009)
27. Liu, H., Yang, L., Niu, Z., Ma, Z., Shao, W.: Facilitating software refactoring with appropriate resolution order of bad smells. In: ESEC/FSE, pp. 265–268 (2009)
28. Langer, P., Wimmer, M.: A posteriori operation detection in evolving software models. *J. Syst. Softw.* **86**, 551–566 (2013)
29. ben Fadhel, A., Kessentini, M.: Search-based detection of high-level model changes. In: ICSM, pp. 212–221 (2012)
30. Wieringa, R.: Design science as nested problem solving. In: Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology, p. 1. ACM Press, New York (2009)
31. Gray, D.E.: Doing Research in the Real World. SAGE Publications, London (2009)
32. Runeson, P., Höst, M., Rainer, A., Regnell, B.: Case Study Research in Software Engineering: Guidelines and Examples. Wiley, Hoboken (2012)