FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Ontology-Based Data Integration and Knowledge Change Management in Multi-Disciplinary Engineering Environments

DISSERTATION

zur Erlangung des akademischen Grades

Doktor der Technischen Wissenschaften

eingereicht von

Fajar J. Ekaputra, MT.
Matrikelnummer 01228673

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Stefan Biffl
Zweitbetreuung: Dr. Marta Sabou, Dr. Estefanía Serral

Diese Dissertation haben begutachtet:

Prof. Enrico Motta

a.Univ.-Prof. Josef Küng

Wien, 8. August 2018

Fajar J. Ekaputra

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Ontology-Based Data Integration and Knowledge Change Management in Multi-Disciplinary Engineering Environments

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor der Technischen Wissenschaften

by

Fajar J. Ekaputra, MT.
Registration Number 01228673

to the Faculty of Informatics

at the TU Wien

Advisor: Ao.Univ.Prof. Stefan Biffl
Second advisor: Dr. Marta Sabou, Dr. Estefanía Serral

The dissertation has been reviewed by:

Prof. Enrico Motta	a.Univ.-Prof. Josef Küng

Vienna, 8th August, 2018

Fajar J. Ekaputra

Erklärung zur Verfassung der Arbeit

Fajar J. Ekaputra, MT.
Gumpendorferstrasse, 1060 Wien, Österreich

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 8. August 2018

Fajar J. Ekaputra

Danksagung

Lobpreis sei Gott, dem Herrn der Welten, der uns Leben, Wissen und Weisheit gegeben hat.

Zunächst möchte ich meinen Eltern von ganzem Herzen für ihre liebevolle Erziehung danken und besonders dafür, dass Sie immer ihr Bestes gegeben haben um sicherzustellen, dass ich und meine Geschwister die beste Ausbildung genießen können. Besonders dankbar bin ich dafür, dass Sie immer an mich geglaubt haben und überzeugt davon waren, dass ich mein Studium abschließen werde. Ebenso möchte ich meinen Schwiegereltern für ihre große Unterstützung und ihr Verständnis im Laufe meines Studiums danken sowie meinen beiden Großmüttern -Andung und der leider bereits verstorbenen Oma Helena, dafür, dass sie mich immer ermutigt haben weiter zu lernen und meinen Weg zu gehen.

Darüber hinaus gilt mein besonderer Dank meinem Betreuer Prof. Stefan Biffl, der mich zu Beginn als unerfahrenen Studenten im Bereich der Forschung aufgenommen hat, mit seiner Expertise und seinem Wissen bereichert und mir vieles über Forschungsarbeit und Management beigebracht hat. Prof. Estefanía Serral, welche mit mir in den frühen Jahren meiner Studienzeit am engsten zusammenarbeitete und mich geduldig lehrte Publikationen zu schreiben, meine albernen Grammatikfehler korrigierte (dies resultierte in einem „Best Paper Award“ für meine erste Konferenzpublikation, danke Fani!) und mich auch nach ihrem Wechsel an die KU Leuven stetig bei der Umsetzung meiner Dissertation unterstützt hat. Dr. Marta Sabou, einer exzellenten Forscherin mit einem großen Forschungsnetzwerk, die es trotzdem schafft, dass die Familie nicht zu kurz kommt, ganz nach ihrem Motto „family-first“ und welche durch ihr Feedback und ihre tatkräftige Unterstützung immer sehr hilfreich für mich war und mich regelrecht dazu angetrieben hat meine Dissertation fertigzustellen.

Ebenso möchte ich all meinen Kollegen der Forschungsgruppe Information and Software Engineering (ISE) danken, insbesondere Prof. A Min Tjoa für all seine Unterstützung und Ratschläge während meiner Studienzeit. Dr. Elmar Kiesling, Peb Aryan, Ba-Lam Do, Niina Novak und Kabul Kurniawan für all die interessanten Diskussionen in unserem Linked Data Lab. Prof. Dr. Manuel Wimmer, Dr. Dietmar Winkler, Dr. Richard Mordinyi, Dr. Petr Novak, Dr. Luca Berardinelli, Olga Kovalenko, Kristof Meixner, Felix Rinker, Jürgen und Angelika Musil für all die wertvollen Diskussionen, Ideen und Gespräche während unserer Zeit bei CDL-Flex. Genauso wie Michael Schadler, Martin Krajiczek, Maria

Schweikert und Natascha Zachs, die mich während meines Studiums immer unterstützt haben.

Die Absolvierung meines Studiums wäre nicht möglich gewesen ohne die finanzielle Unterstützung des Österreichischen Austauschdienst (OeAD) während der ersten drei Jahre meiner Studienzeit und der Unterstützung durch mehrere Forschungsprojekte (unter anderem: CDL-Flex, Siemens-STAR und FFG-CitySPIN) im weiteren Studienverlauf, für die ich sehr dankbar bin.

Mein besonderer Dank gilt all meinen Freunden und Kollegen in Österreich, die mich in unterschiedlichen Rollen auf dieser einzigartigen Reise begleitet haben. Denjenigen die mit mir Zeit verbracht haben, Dinge unternommen haben und mir ein Ohr geliehen haben, besonders in der Zeit bevor meine Familie angekommen ist (Gunar, Keni, Dhota). Denjenigen, die mich einladen mit ihnen Fußball zu spielen (Anta, Ochid, Edo, Radit, Tölga, Yazan); denjenigen mit denen ich immer wieder interessante und bereichernde Gespräche und Diskussionen über wichtige Dinge führen kann (Pak Rachmad, Pak Syahril, Pak Taufik, Yusfan, Mas Adhi), meinen Laron – Band Mitgliedern (Mas Indra, Ario, Pandu, Frisky, Melissa, Juan), den Mitgliedern meiner Liqo-Gruppe (Jali, Peb, Sisko, Kabul, Rachmad, Pak Ismail, Mas Catur, Fainan, Robby), denjenigen die sowohl mich als auch andere Studenten aus Indonesien einladen indonesisches Essen weit entfernt von der Heimat zu genießen (Aldi, Ilka, Ashrie, Pak De, Mas Dindin), denjenigen die seit vielen Jahren hier in Österreich studieren und mich immer wieder ermutigt haben nicht aufzuhören und gezeigt haben dass es funktioniert (Rasmi, Mba Ika), denjenigen, die mich immer mit ihren „falschen Ermutigungen“ necken und zugleich motivieren (Dhota, Dewi) und jenen Familien, die meiner Familie so viel Freundschaften und Komfort hier in Österreich bieten (der Familie Mertiansya, der Familie Zuhri, der Familie Wibowo, der Familie Al Hakim und der Familie Rizaldi).

Zu guter Letzt möchte ich noch erwähnen, wie überaus glücklich und gesegnet ich mit Ayu und Rayyan bin, einer wundervollen Frau und einem klugen, aufgeweckten und manchmal frechen Sohn, der mich immer aufheitert. Sie haben ihre, mehr als zehntausend Kilometer entfernte, Komfortzone sowie die engste Familie und Freunde hinter sich gelassen, als Rayyan lediglich zehn Monate alt war, um mich auf dieser Reise zu begleiten, in dem Wissen, dass diese Reise nicht einfach sein wird. Ohne ihre Geduld und Unterstützung während dieser verrückten Reise – hätte ich es niemals geschafft – Vielen Dank!

Acknowledgements

All praise be to The Lord of the worlds, who has given us life, knowledge, and wisdom.

My first and foremost gratitude goes to my parents, who raised me with all their hearts and always gave their best to make sure that my siblings and I have the best education through our life. They never stop believing that I can make it through all my studies and include me in their prayers, and I am sincerely grateful for that. I would also like to express my gratitude to my parents-in-law for their massive support and understanding, and to both my grandmothers –Andung and the late Nenek Helena– for always encouraging me to further my study.

Furthermore, I would like to express my gratitude to my advisors: Prof. Stefan Biffl, who accept me to become his student when I am still a novice in research, helped me through his wisdom, and teach me a lot on management in research; Dr. Estefanía Serral, who has been working closest to me during my early years of study, patiently teaching me how to write a paper, revising my silly grammar mistakes (which resulted in a best paper award in my first conference paper, thanks Fani!), and consistently helped me with my thesis even after she went to KU Leuven; And Dr. Marta Sabou, who is, well... a role model for me, an excellent researcher with large research networks and a family-first motto, she is always helpful with all her feedback and actions, and become a significant catalyst for me to finish the thesis finally.

Also, I would like to thank all my colleagues at the Information and Software Engineering (ISE) research group, especially Prof. A Min Tjoa for all his help and guidance during my hardships; Dr. Elmar Kiesling, Peb Aryan, Ba-Lam Do, Niina Novak, and Kabul Kurniawan for all the interesting discussions in our Linked Data Lab; Prof. Manuel Wimmer, Dr. Dietmar Winkler, Dr. Richard Mordinyi, Dr. Petr Novak, Dr. Luca Berardinelli, Olga Kovalenko, Kristof Meixner, Felix Rinker, Jürgen and Angelika Musil for all fruitful discussions, ideas, and debates during our time at CDL-Flex; Michael Schadler, Martin Krajiczek, Maria Schweikert and Natascha Zachs, who always provide excellent support during my study. Bonus thanks for Kristof and Niina for helping me translating the thesis abstract and acknowledgement!

I would have never started my study and finished it without the financial supports from the Österreichischer Austauschdienst (OeAD) during the first three years of the study and several research projects afterwards (CDL-Flex, Siemens-STAR, and FFG-CitySPIN, among others), of which I am grateful.

My special thanks go to all my friends and colleagues in Austria for their different “roles” during this unique journey; those who provide me with places to go and somebody to talk to, especially before my family arrived (Gunar, Keni, Dhota); those who invite me to play football (Anta, Ochid, Edo, Radit, Tölga, Yazan); those who challenge me with intelligent discussions on important matters (Pak Rachmad, Pak Syahril, Pak Taufik, Yusfan, Mas Adhi); those who play together with me in the Laron Band (Mas Indra, Ario, Pandu, Frisky, Melissa, Juan); those who were in my Liqo group (Jali, Peb, Sisko, Kabul, Rachmad, Pak Ismail, Mas Catur, Fainan, Robby); those who invite me and other “single” students to taste Indonesian food away from home (Aldi, Ilka, Ashrie, Pak De, Mas Dindin); those who endure the longevity of studying here in Austria and shows me never to give up (Rasmi, Mba Ika); those who always motivate me with their “reverse encouragements” (Dhota, Dewi); and those families who provide friendships and comforts for my family in Austria (The Mertiansya’s, The Zuhri’s, The Wibowo’s, The Al Hakim’s, and The Rizaldi’s).

Lastly but definitely not the least, I would like to mention how lucky I am, blessed with Ayu and Rayyan, a wonderful wife and a smart, cheeky son who always cheer me up. They left their comfort zone more than ten thousand kilometers behind when Rayyan was ten months old, away from their close families and friends, to accompany me within this journey, understanding that the trip would not be an easy one. Without their patience and support during this crazy ride, I would never make it through - thank you.

Kurzfassung

Aktuelle Entwicklungen im Bereich der Ingenieurswissenschaften, die besser unter dem Begriff „Industrie 4.0“ bekannt sind, erfordern flexiblere Produktionssysteme. Diese Art von modernen Produktionssystemen werden typischerweise als disziplinübergreifende Engineering-Umgebungen entworfen, in denen Akteure unterschiedlichster Ingenieursdisziplinen, Daten von verschiedenen, heterogenen Datenquellen integrieren, und Änderungen innerhalb dieser integrierten Datenquellen verwalten müssen, um Änderungen an einem Datensatz in anderen relevanten Datensätzen widerzuspiegeln. Aus diesem Grund sind disziplinübergreifende Engineering-Umgebungen stark auf eine Integration der Daten zwischen den verschiedenen Akteuren und Ingenieursdisziplinen angewiesen. Technologien aus dem Forschungsbereich Semantic Web unterstützen eine solche übergreifende Datenintegration mittels sogenannten ontologiebasierten Datenintegrationsansätzen, die ihren Bedingungen entsprechend, Voraussetzungen für die Unterstützung von Managementansätzen von Wissensänderungen in disziplinübergreifenden Engineering-Umgebungen sind.

Für ontologiebasierte Datenintegrationen wurden bisher drei Ansätze vorgeschlagen: kombinierte Ontologien, die domänenübergreifend, alle Entitäten beinhalten, multiple Ontologien, die untereinander auf Entitäten verweisen, und hybride Ontologien, die auf ein gemeinschaftliches Vokabular zurückgreifen. Eine solche Klassifizierung wurde jedoch als zu allgemein erachtet, um zu entscheiden, welcher der Ansätze am geeignetsten im Bereich von disziplinübergreifenden Engineering-Umgebungen ist. Eine erste Herausforderung besteht darin, zu verstehen was unter der Eignung von ontologiebasierten Datenintegrationsvarianten für disziplinübergreifende Engineering-Umgebungen verstanden wird und wie die Möglichkeiten, die diese Ansätze bieten erweitert werden können. Nachdem das Management von Wissensänderungen hauptsächlich auf Ansätzen für kombinierte Ontologien untersucht wurde, besteht die zweite Herausforderung in der Bereitstellung von Techniken für das Management von Wissensänderungen, die auch für komplexere ontologiebasierte Datenintegrationsansätze, wie zum Beispiel hybride Ansätze anwendbar sind.

Um die erste Herausforderung zu adressieren, wird in dieser Dissertation die Machbarkeit und Eignung von ontologiebasierten Datenintegrationsansätzen in disziplinübergreifenden Engineering-Umgebungen bewertet. Dafür wird ein neuer Ansatz für ontologiebasierte Datenintegration namens Global-as-View vorgeschlagen und die Kriterien für eine Klassifizierung von ontologiebasierten Datenintegrationvarianten ausgearbeitet, welche danach eine qualifizierte Entscheidungsgrundlage für unterschiedlichen Szenarien disziplinübergreifender Engineering-Umgebungen bilden. Der zweite Teil der Arbeit adressiert die Forschungslücke im Bereich der Unterstützung von Wissensänderungen in komplexen, ontologiebasierten Datenintegrationen. An dieser Stelle wird in der Arbeit eine Auswahl an Anforderungen für das Management von Wissensänderungen für komplexe ontologiebasierte Datenintegrationen im disziplinübergreifenden Engineering-Umgebungen, auf Basis von Anwendungsfällen der Domäne, definiert. Aus diesen zugrundeliegenden Anforderungen wird ein generisches, technologieunabhängiges Rahmenwerk für das Management von Wissensänderungen für hybride und Global-as-View Szenarien in der ontologiebasierten Datenintegration entwickelt. Um diesen Ansatz zu evaluieren, wird ein spezifischer Bezugsrahmen für das Management von Wissensänderungen als Forschungsprototyp realisiert, und anhand dessen eine Machbarkeitsstudie durchgeführt.

Die wichtigsten Beiträge der Dissertation sind: (1) eine systematische Literaturrecherche, die Anwendungsarten von ontologiebasierten Datenintegrationen in disziplinübergreifenden Engineering-Umgebungen beinhaltet, auf deren Basis ein neue Variante einer ontologiebasierten Datenintegration und ein dementsprechender Leitfaden für die Auswahl einer solchen Variante, basierend auf den Charakteristiken des jeweiligen Anwendungsaspekts, vorgeschlagen wird; (2) eine Evaluierung des vorgeschlagenen Leitfadens für die Auswahl von ontologiebasierten Datenintegrationen, mit dem Ziel, spezifische, prototypische Anwendungen für ontologiebasierte Datenintegration für zwei konkrete Anwendungsfälle aus dem Bereich des disziplinübergreifenden Ingenieursumgebung zu verwirklichen; (3) die Definition eines generischen Rahmenwerks für das Management von Wissensänderungen für disziplinübergreifende Engineering-Umgebungen und dessen Evaluierung.

Abstract

The recent developments in the engineering domain, often associated with the German term “Industrie 4.0”, require more flexible production systems. Such modern production systems are typically designed in Multi-Disciplinary Engineering Environments (MDEEs), where stakeholders from diverse engineering disciplines need to integrate data from heterogeneous data sources and manage changes within the integrated data sources, so that changes in one dataset are reflected in other relevant data sources. MDEEs, therefore, strongly rely on data integration across various stakeholders and engineering disciplines. Semantic Web Technologies support data integration using Ontology-Based Data Integration (OBDI) approaches, which, on their term are a prerequisite to support knowledge change management in MDEEs.

Three variants of OBDI approaches have been proposed: single-ontology, multiple-ontology, and hybrid approaches. However, this classification is deemed too generic for understanding which approaches are most suitable in MDEEs. A first challenge, on the one hand, consists in understanding the suitability of OBDI approach variants for diverse MDEE scenarios and considering potential extensions thereof. On the other hand, since the topic of knowledge change management has been primarily investigated for single-ontology OBDI approaches, a second challenge therefore consists in providing knowledge change management techniques that are also applicable for more complex OBDI approaches (e.g., hybrid approaches).

To address the first challenge, in this thesis, we evaluate the feasibility and suitability of OBDI approaches for diverse MDEE use cases. We propose a new OBDI variant called Global-as-View OBDI and distill criteria for classifying OBDI variants that allow for their informed selection for use in diverse MDEE scenarios. Our second line of work addresses the gap in supporting knowledge change management in complex OBDI variants. To that end, we define a set of knowledge change management requirements for complex OBDI in MDEE based on use cases from the engineering domain. Based on these requirements, we develop a generic, technology-agnostic framework of knowledge change management for hybrid and Global-as-View OBDI scenarios. To evaluate our approach, we instantiate the reference framework as a research prototype and conduct an initial feasibility study.

The main outputs of the thesis are: (1) a *systematic literature review* on OBDI applications in MDEE, based on which we propose a new OBDI variant and guidelines for the selection of OBDI variants based on application context characteristics; (2) the evaluation of the

OBDI selection guidelines to design concrete prototypes of OBDI applications for tackling two use cases in MDEEs; and (3) the definition of a generic knowledge change management framework for MDEEs and its feasibility evaluation.

Contents

CHAPTER 1

Introduction

Current developments in the engineering domain, often associated with the German term "Industrie 4.0" [BtHVH14], require more flexible production systems (i.e., factories) that rely on strong data integration across various stakeholders and engineering disciplines. The lifecycle of such production systems typically involves contributions by engineers from a variety of disciplines that collaborate in a Multi-Disciplinary Engineering Environment (MDEE) [BGL17], an environment where stakeholders from several engineering disciplines work together to achieve a common goal (e.g., the design of a factory). For instance, the engineering of a modern hydro power plant usually involves a main contractor, subcontractors, and component vendors [SKEB16]. These stakeholders cover a variety of engineering disciplines including mechanical, electrical, and software engineering, and make use of various engineering software tools, datasets, and terminologies, with limited overlap.

An illustrative example of an MDEE from the engineering of mechatronic systems from Automation System Engineering domain is shown in Figure 1.1. This figure shows engineers from software, electrical and mechanical engineering that are working together to build a complex mechatronic system (e.g., a production system) using specific tools from their respective engineering disciplines (left hand side of Figure 1.1).

Collaboration among these stakeholders requires synchronizing and exchanging data produced by software tools specific to their disciplines. In order to reach their goals, on the one hand, it is important to be able to define and share the necessary knowledge for common work processes between these engineers by the means of data integration. On the other hand, project managers, knowledge engineers and domain experts need to see an integrated view of the mechatronic system in order to conduct the necessary analyses, e.g., to analyze artefact changes during the engineering process (right hand side of Figure 1.1).

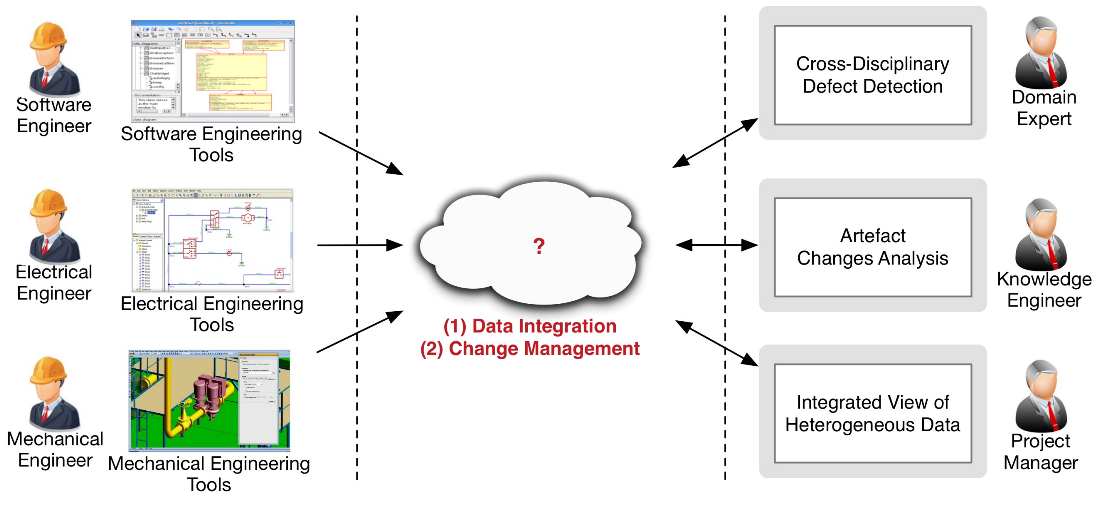

Software Engineer
Software Engineering Tools
Electrical Engineer
Electrical Engineering Tools
Mechanical Engineer
Mechanical Engineering Tools
?
(1) Data Integration
(2) Change Management
Cross-Disciplinary Defect Detection
Artefact Changes Analysis
Integrated View of Heterogeneous Data
Domain Expert
Knowledge Engineer
Project Manager

Figure 1.1: An MDEE problem setting example: the engineering of mechatronic systems

The cross-disciplinary knowledge in MDEEs, however, is often available only implicitly and is therefore challenging to share. This results in time-consuming and repetitive tasks of manually retrieving and processing such knowledge [MB12]. In many cases of MDEEs, domain experts and project managers also have to invest considerable effort to provide the required cross-disciplinary functions due to the unavailability of explicit knowledge and semantic gaps between disciplines [MMW^{+}11].

Consequently, **processes in MDEEs that create modern and flexible production systems have strong needs for data integration**, a crucial prerequisite to enable advanced capabilities that support the work of engineering teams, such as *early defect detection* [KWK^{+}14] or *knowledge change management* [WMM^{+}11b]. MDEEs must therefore evolve from current, primarily manual practices towards the use of more flexible and knowledge-driven technologies.

These advanced capabilities are important to improve work efficiency and overall project management (e.g., to mitigate common risks in such projects). In addition, the desired shorter life-cycles and higher variation of products in modern production systems also require better integration between (i) the life cycles of products and the associated production systems; and (ii) the engineering and operation phases of these production systems [SEB17]. A key challenge in this context is therefore caused by the heterogeneous and semantically overlapping models [FHK^{+}15].

Recently, research on applying Semantic Web (SW) technologies, especially the Ontology-Based Data Integration (OBDI), to address the challenge of data integration in MDEE has been intensified, e.g., for engineering design quality improvement [SRFF11, HVKE16], for simulation generation and evaluation [TU14, DVYP14], for knowledge representation [LLP^{+}15, NFGT16], and for team collaboration [WMM11a]. OBDI, with ontologies as key resources, facilitates data integration by capturing the implicit knowledge across heterogeneous data sources and establishing semantic interoperability between them [WVV+01].

However, given the complexity of data integration scenarios in MDEEs, choosing the most appropriate OBDI variant, as well as suitable technologies for implementing it is challenging, since appropriate choices are mainly determined by the specific characteristics of the problem setting, such as data source heterogeneity or mapping complexity between data sources. Therefore, providing an understanding on how suitable the OBDI approach and its variants are for the diverse data integration scenarios is highly important, both for researchers and practitioners.

Another challenge lies in providing support for data transformation and change propagation, which is part of the **Knowledge Change Management (KCM)** process for OBDI systems. The current approaches to deal with KCM from the SW community do not sufficiently address such requirements, which are crucial to ensure an efficient and correct knowledge management and foster further adoption of OBDI in MDEEs and other similar environments.

1.1 Research Questions

Following the challenges previously introduced, the central research question in the thesis is the following:

> ***Which mechanisms and methods from Semantic Web technologies are suitable to address challenges of Data Integration and Knowledge Change Management in Multi-Disciplinary Engineering Environments?***

Three variants of Ontology-Based Data Integration (OBDI) approaches have been proposed: single-ontology, multiple-ontology and hybrid approaches [WVV+01]. However, this classification is deemed too generic and it is not clear how the OBDI variants relate to scenarios in Multi-Disciplinary Engineering Environments (MDEEs). Therefore, **the first challenge** in the thesis consists in understanding the suitability of the OBDI approach and its variants for diverse MDEE scenarios as well as identifying possible gaps and considering potential extensions thereof.

> **RQ1:** ***"How suitable are the Ontology-Based Data Integration approach variants for the diverse data integration scenarios in Multi-Disciplinary Engineering Environments?***

The OBDI approach allows development of knowledge graphs in the MDEEs as a basis for advanced applications. There is, however, only limited support for managing explicit (i.e., in opposite of tacit) knowledge graph changes in OBDI, since the topic of Knowledge Change Management (KCM) has been primarily investigated for single-ontology OBDI approaches [Eka15].

This aspect of KCM is important for scenarios in the MDEE where changes are integral parts of the process, e.g., the iterative engineering process of hydro-power plant systems. Due to this issue, **the second challenge** in the thesis consists of providing KCM methods and techniques for more complex OBDI approaches (e.g., the hybrid approach) in the context of MDEEs.

> **RQ2:** ***"How to provide sufficient support for Knowledge Change Management in Multi-Disciplinary Engineering Environments in systems using Ontology-Based Data Integration approach?"***

1.2 Research Contributions

The first contribution of the thesis is a literature review on existing Ontology-Based Data Integration (OBDI) applications in Multi-Disciplinary Engineering Environments (MDEEs) to address RQ1 on the suitability of OBDI approach variants for MDEE scenarios [ESS+17]. The review is based on Systematic Literature Review (SLR) method [KC07, ZRM+15] and covers 23 OBDI applications reported in 29 papers.

During the review process, we observed that there are OBDI applications that are similar, but do not exhibit all the characteristics of hybrid OBDI. We refer to this approach as Global-as-View (GAV) OBDI due to its similarity with the GAV approach from the relational databases [DHI12]. To differentiate this OBDI approach from existing variants, we propose to add GAV to the existing OBDI typology from Wache et al. [WVV+01]. Based on the review, we develop a guideline to support the selection of suitable OBDI approaches on the context of MDEEs.

To support our findings above, **the second contribution** of the thesis lies on the evaluation of the OBDI selection guidelines in the engineering context. To this end, we use the guideline to help designing and developing OBDI applications for two distinct use cases in MDEEs: AutomationML Analyzer and Ontology-based Cross-disciplinary Defect Detection (OCDD).

AutomationML Analyzer [SEKB16] is an OBDI application that aims to support analysis of engineering data and was later extended to support generation of simulation models [NEB17]. The OBDI approach in this context aims to address the challenge of (1) understanding complex data structures with intricate links between the elements of engineering data, and (2) providing support for conducting cross-disciplinary analytics by end users. The application uses a single-ontology OBDI variant due to the availability of AutomationML[1] –an open standard for engineering data exchange– in the use case.

[1]www.automationml.org

OCDD approach is an OBDI application that aims to support defect detection across heterogeneous engineering data models [KWK+14]. We applied the OBDI selection guideline against the use case characteristics, which resulting in the selection of the multiple-OBDI variant for OCDD approach.

The third contribution answers RQ2 by designing a Knowledge Change Management (KCM) framework for MDEEs based on the OBDI approach. To design such framework, we first define the KCM requirements for MDEEs [ESSB15b]. Considering these requirements, we make an analysis of Semantic Web (SW) related work, which shows the gap on the KCM support beyond single-ontology OBDI systems [Eka15], especially in the context of MDEEs. The proposed KCM framework addresses this gap by building on the the GAV OBDI approach [Eka16], which can be generalized beyond the SW research communities.

To evaluate the proposed framework, we instantiate the framework as a research prototype using SW technologies and conducts an initial feasibility evaluation in the use case of a hydro-power plant engineering process, which is based on a real-world use case from our industry partner [ESSB16]

In summary, the three main results from this thesis are the following:

- A comparison and analysis on current OBDI variants in MDEE, which proposes **an extension of OBDI variant classifications** and a **guideline for the selection and adaptation** of OBDI variants depending on project contexts;
- The evaluation of the proposed guideline to design and develop two **concrete prototypes of OBDI applications** for tackling distinct use cases and their challenges in the MDEEs; and
- The definition of a **generic KCM framework** for MDEE applications built on the OBDI approach, which would allow advanced features such as engineering project progress analysis and monitoring.

1.3 Thesis Structure

Chapter 2 introduces the research context and motivates the needs of our research. In Chapter 2, we explain important terms and related work, e.g., Semantic Web (SW), Ontology-Based Data Integration (OBDI), Ontology Change, and Multi-Disciplinary Engineering Environments (MDEEs). Furthermore, we explain how SW technologies are being utilized within the MDEEs [SKEB16].

Chapter 3 reviews OBDI applications in the MDEEs from both architectural (i.e., OBDI variants) and technical points of view [ESS+17]. Furthermore, we introduce an additional variant of OBDI, called Global-as-View (GAV) OBDI and propose a guideline to help users in selecting the most suitable OBDI variant for their scenario.

Chapter 4 tests the guidelines created in Chapter 3 within two concrete use cases from our industry partners. As the result, we develop two concrete OBDI prototypes in MDEEs that are enabled by OBDI, namely AutomationML Analyzer [SEKB16] and Ontology-based Cross-discplinary Defect Detection Tool [KSS$^+$14]. These prototypes demonstrate contexts and needs from MDEE projects, together with approaches and concrete prototypes to solve them. In addition, we briefly discuss the benefit and drawbacks of Semantic Web Technologies (SWT) in comparison with alternative technologies.

Chapter 5 provides the definition of a generic framework for Knowledge Change Management (KCM) in MDEE as an extension of OBDI. We build on a set of requirements from MDEEs to develop the framework [ESSB15b, Eka16]. Further, we develop an instance of the framework using SW technologies based on a use case of hydro-power plant engineering from an industry partner [ESSB16] to conduct a feasibility evaluation.

Chapter 6 concludes the thesis with a discussion about research questions and results, describes the thesis limitations, and finally provides a set of potential research issues and challenges to be pursued in further research.

1.4 Publications

During this PhD, we have co-authored 27 peer-reviewed scientific publications, including 2 journal papers, 5 book chapters, 14 full conference papers and 6 workshop/in-progress papers. We won two best paper awards (i-Semantics 2013 and ICoDSE 2014) and received one nomination for best paper award (SEMANTiCS 2016). Parts of this thesis are based on a subset of the aforementioned publications, as explained next.

Parts of Chapter 2 are published in the following publications:

- Marta Sabou, Olga Kovalenko, Fajar J. Ekaputra, and Stefan Biffl. Semantic Web Solutions in Engineering. In Stefan Biffl and Marta Sabou, editors, *Semantic Web Technologies for Intelligent Engineering Applications*, chapter 11, pages 281–296. Springer International Publishing, Cham, 2016 [SKEB16].
- Marta Sabou, Fajar J. Ekaputra, and Stefan Biffl. Semantic Web Technologies for Data Integration in Multi-Disciplinary Engineering. In Stefan Biffl, Detlef Gerhard, and Arndt Lüder, editors, *Multi-Disciplinary Engineering for Cyber-Physical Production Systems*, pages 301–329. Springer, 2017 [SEB17].

Chapter 3 of the Thesis is written based on the following publication:

- Fajar J. Ekaputra, Marta Sabou, Estefanía Serral, Elmar Kiesling, and Stefan Biffl. Ontology-Based Data Integration in Multi-Disciplinary Engineering Environments: A Review. *Open Journal of Information Systems (OJIS)*, 4(1):1–26, 2017 [ESS$^+$17].

Chapter 4 of the Thesis is written based on the following publications:

- Olga Kovalenko, Estefanía Serral, Marta Sabou, Fajar J. Ekaputra, Dietmar Winkler, and Stefan Biffl. Automating Cross-Disciplinary Defect Detection in Multi-disciplinary Engineering Environments. In *Knowledge Engineering and Knowledge Management: 19th International Conference, EKAW 2014, Linköping, Sweden, November 24-28, 2014. Proceedings*, volume 8876, pages 238–249, 2014 [KSS$^+$14]

- Olga Kovalenko, Manuel Wimmer, Marta Sabou, Arndt Lüder, Fajar J. Ekaputra, and Stefan Biffl. Modeling AutomationML: Semantic Web Technologies vs. Model-Driven Engineering. In *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*, pages 1–4. IEEE, 2015 [KWS$^+$15]

- Marta Sabou, Fajar J. Ekaputra, Olga Kovalenko, and Stefan Biffl. Supporting the engineering of cyber-physical production systems with the AutomationML analyzer. In *2016 1st International Workshop on Cyber-Physical Production Systems (CPPS)*, pages 1–8. IEEE, apr 2016 [SEKB16]

- Petr Novák, Fajar J. Ekaputra, and Stefan Biffl. Generation of Simulation Models in MATLAB-Simulink Based on AutomationML Plant Description. In *IFAC - Papers Online*, pages 7613–7620, 2017 [NEB17]

And finally, Chapter 5 is written based on the following publications:

- Fajar J. Ekaputra. Ontology change in ontology-based information integration systems. In *12th European Semantic Web Conference, ESWC 2015*, volume 9088, pages 711–720. Springer International Publishing, 2015 [Eka15].

- Fajar J. Ekaputra, Estefanía Serral, Marta Sabou, and Stefan Biffl. Knowledge Change Management and Analysis for Multi-Disciplinary Engineering Environments. In *Joint Proceedings of the Posters and Demos Track of 11th International Conference on Semantic Systems - SEMANTiCS2015 and 1st Workshop on Data Science: Methods, Technology and Applications (DSci15)*, volume 1481, pages 13–17, 2015 [ESSB15b].

- Fajar J. Ekaputra. Knowledge Change Management and Analysis in Engineering. In Stefan Biffl and Marta Sabou, editors, *Semantic Web Technologies for Intelligent Engineering Applications*, chapter 7, pages 159–178. Springer International Publishing, Cham, 2016 [Eka16].

- Fajar J. Ekaputra, Marta Sabou, Estefanía Serral, and Stefan Biffl. Knowledge change management and analysis during the engineering of cyber physical production systems: A use case of hydro power plants. In *Proceedings of the 12th International Conference on Semantic Systems - SEMANTiCS 2016*, volume 13-14-Sept, pages 105–112, New York, New York, USA, 2016. ACM, ACM Press (**nominated for best paper award**) [ESSB16].

In addition to the papers that directly contributed to the thesis, we have co-authored a number of other papers[2]. A selected list of these papers includes:

- Marta Sabou, Fajar J Ekaputra, Tudor Ionescu, Juergen Musil, Daniel Schall, Kevin Haller, Armin Friedl, and Stefan Biffl. Exploring Enterprise Knowledge Graphs : a Use Case in Software Engineering. In *Extended Semantic Web Conference (ESWC)*, 2018 [SEI+18].
- Juergen Musil, Fajar J. Ekaputra, Marta Sabou, Tudor Ionescu, Daniel Schall, Angelika Musil, and Stefan Biffl. Continuous Architectural Knowledge Integration: Making Heterogeneous Architectural Knowledge Available in Large-Scale Organizations. In *Proceedings - 2017 IEEE International Conference on Software Architecture, ICSA 2017*, pages 189–192, 2017 [MES+17].
- Richard Mordinyi, Estefanía Serral, and Fajar J. Ekaputra. Semantic Data Integration: Tools and Architectures. In *Semantic Web Technologies for Intelligent Engineering Applications*, pages 181–217. Springer, 2016 [MSE16].
- Richard Mordinyi, Dietmar Winkler, Fajar J. Ekaputra, Manuel Wimmer, and Stefan Biffl. Investigating model slicing capabilities on integrated plant models with AutomationML. In *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, volume 2016-Novem, 2016 [MWE+16]
- Fajar J. Ekaputra and Xiashuo Lin. SHACL4P: SHACL constraints validation within Protégé ontology editor. In *Proceedings of 2016 International Conference on Data and Software Engineering, ICoDSE 2016*, 2016 [EL16].
- Stefan Biffl, Marcos Kalinowski, Fajar J. Ekaputra, Estefanía Serral, and Dietmar Winkler. Building Empirical Software Engineering Bodies of Knowledge with Systematic Knowledge Engineering. In *International Conference on Software Engineering and Knowledge Engineering (SEKE)*, 2014 [BKE+14].
- Stefan Biffl, Marcos Kalinowski, Rick Rabiser, Fajar J. Ekaputra, and Dietmar Winkler. Systematic Knowledge Engineering: Building Bodies of Knowledge from Published Research. *International Journal of Software Engineering and Knowledge Engineering*, 24(10):1533–1571, dec 2014 [BKR+14].
- Fajar J. Ekaputra, Estefanía Serral, Dietmar Winkler, and Stefan Biffl. A semantic framework for data integration and communication in project consortia. In *Proceedings of 2014 International Conference on Data and Software Engineering, ICODSE 2014*, pages 1–6. IEEE, IEEE, nov 2014 (**best paper award**) [ESWB14].
- Fajar J. Ekaputra, Marta Sabou, Estefanía Serral, and Stefan Biffl. Supporting Information Sharing for Reuse and Analysis of Scientific Research Publication

[2]https://scholar.google.at/citations?user=ELwbaWYAAAAJ&hl=en

Data. In *Proceedings of the 4th Workshop on Semantic Publishing (SePublica 2014)*, volume 1155. CEUR-WS, 2014 [ESSB14].

- Fajar J. Ekaputra, Estefanía Serral, Dietmar Winkler, and Stefan Biffl. An analysis framework for ontology querying tools. In *Proceedings of the 9th International Conference on Semantic Systems - I-SEMANTICS '13*, page 1, New York, New York, USA, 2013. ACM Press (**best paper award**) [ESWB13].

CHAPTER 2

Background and Related Work

This chapter explains our research context and motivation, important terms, and related works that are relevant to this thesis.

Parts of this chapter are published in several publication venues, including the general introduction of semantic web solutions [SKEB16] in engineering and semantic web-based data integration approaches [SEB17] for multi-disciplinary engineering process. Furthermore, we adapt the related work from several of our previous publications on relevant subjects, such as Ontology-Based Data Integration (OBDI) [ESS$^+$17] and Knowledge Change Management (KCM) [Eka15, Eka16], providing an in-depth discussion to the chapter.

2.1 Multi-Disciplinary Engineering Environments

The typical traditional industrial production systems are focused on a rigid, mass productions with limited produce varieties and customization. In recent years, there is an increasing trend in industries to move towards more modern production systems with flexible, low-volume productions with large varieties based to customer demands. This movement is reflected in several initiatives across the globe. E.g., the Industrie 4.0 initiative in Germany is a vision of an advanced production system control architecture and engineering methodology [BtHVH14]. Similar initiatives have also been initiated in other countries, such as Industrial Internet consortium in the US and Factory of the Future in France and UK [RCW13].

To achieve such a vision, it is necessary that the current traditional production systems go through major changes in their whole life cycle. As the first step of this cycle, the engineering of these production systems needs to produce higher quality results that allow such flexible productions.

To this end, there is a need to streamline the work of the engineering organization (i.e., a set of multi-disciplinary engineering teams that is involved in the planning, realization, and commissioning of new technical systems [Ver09]) responsible for the engineering process, utilize diverse discipline-specific tools, and exploit heterogeneous data and data models with limited overlap coming from aforementioned stakeholders and tools [SLSB14]. In this context, an engineering organization becomes the execution environment (referred as the **Multi-Disciplinary Engineering Environment (MDEE)**) of an engineering process that requires collaboration between involved engineering disciplines to develop products and the associated production systems [BGL17].

However, the optimization of engineering processes is often hampered by their heterogeneous and collaborative nature of the MDEE of an engineering organization. Heterogeneity is a key characteristic because a large and diverse set of stakeholders is involved in an engineering organization, often spanning across several divisions within and between companies. Despite their heterogeneity, the involved stakeholders in an MDEE need to collaborate toward designing and building a complex production system. Indeed, they all provide data and engineering effort to the engineering process. Based on these inputs, many engineering decisions are taken that shape the detailed engineering and implementation of the intended production system [SLSB14]. Therefore, **a key challenge for realizing such flexible engineering of production systems lies on solving the data integration among these stakeholders** (e.g., engineers from mechanical, electrical and software disciplines; project managers) and artefacts (e.g., discipline-specific tools, data model, and data) across engineering disciplines.

Knowledge-based approaches in general have been observed to be particularly suitable to deal with data heterogeneity as well as to enable advanced functionalities of such systems (e.g., handling disturbances adapting to new business requirements) [LLVH13]. This opens up the need for knowledge-based approaches, such as Semantic Web Technologiess (SWTs) where ontologies [Gru93, SBF98] are used as models [BLHL01, SBLH06]. SWTs extend the principles of knowledge-based approaches to Web-scale settings which introduce novel challenges in terms of data size, heterogeneity, and level of distribution [BLHL01]. In such setting, SWTs focus on large-scale (i.e., Web-scale) data integration and intelligent, reasoning-based methods to support advanced data analytics. SWTs enable a wide range of advanced applications [SBLH06] and they have been successfully employed in various areas, ranging from pharmacology [GGL$^+$14] to cultural heritage [Hyv12] and e-business [Hep08].

The adoption of SWTs in the industrial production settings and theMDEE in particular is comparatively slower than the aforementioned areas [SKEB16]. A potential explanation is the complexity of the MDEE hampers a straightforward adoption of SWTs in both aspects of the core data integration challenge and crucial advanced engineering process supports, such as **Knowledge Change Management (KCM)**.

This chapter aims to provide the background and motivates the needs of our research, which is focus on the following research gaps: (1) The limited understanding of potential stakeholders on how Ontology-Based Data Integration (OBDI) approach and its variants can be adopted to address the core data integration challenge due to the complexity and the diverse data integration scenarios in MDEE, and (2) The lack of advanced engineering process supports of KCM, which is a necessary requirement in many use cases of MDEE, in systems using OBDI approaches.

2.1.1 The Needs for Semantic Supports

In their work [BLW16], Biffl et al. identifies a number of needs for semantic supports derived from production systems engineering use case and scenarios –which subsumes the MDEE settings– in a similar manner as the discussion on the enterprise systems ecosystem [Obe14]. These needs (Nx) are as follows:

N1-Explicit engineering knowledge representation. The stakeholders in the engineering domain use a wide-range of models regularly to represent certain aspects of the engineering knowledge [NBJ11]. However, in many cases, the modeling languages do not provide sufficient level of expressiveness needed to automate the production systems engineering. Therefore, there is a need of knowledge representation support to analyze the requirements for the level of representation needed and providing the stakeholders with the necessary tools and methods.

N2-Engineering data integration. The engineering tool network in the MDEE of production system engineering environment contains a collection of tools with heterogeneous data models, which use different terms and data formats for similar concepts [MB12]. Due to this heterogeneity it is difficult, costly, and error prone to provide a consistent production system plant model for parallel engineering. In particular, in MDEEs there is the need for an engineering data integration approach, which therefore provides an integrated data model of the common concepts of stakeholders across disciplines to enable the linking of engineering knowledge across disciplines.

N3-Engineering knowledge access and analytics. Knowledge access and analytics in production system engineering builds on the availability of formally represented (N1) and integrated (N2) engineering data in an MDEE. Stakeholders need basic functions operating on common data model, e.g., reports and analyses to check the project progress and the quality of the results from parallel engineering. Therefore, effective and efficient mechanisms are needed for (a) querying of engineering models, *including versions and changes*; and (b) defining and evaluating engineering model constraints and rules across several views.

N4-Efficient access to semi-structured data in the organization and on the Web. Production systems engineering process automation is mostly based on structured data, e.g., in databases or documents that follow a structured data model. In addition to structured data in databases or knowledge bases, the use of semi-structured data, e.g., technical fact sheets including natural language text, or linked data, e.g., component information in the organization and on the Web can help improve the automated support for reuse processes. Therefore, there is a need for more efficient access to semi-structured data in the organization and on the Web.

N5-Flexible and intelligent engineering applications. Assuming the capability of knowledge access and analytics (N3) on an integrated production system plant model, stakeholders can now design and develop intelligent engineering applications, such as defect detection and *knowledge change management*. In a production system context, these engineering solutions need to be flexible to adapt to the changes in the production system both at design time and at runtime. Furthermore, an intelligent engineering application needs to go beyond hard-coded programs and is driven by the description of the production system plant.

N6-Support for multidisciplinary engineering process knowledge. One of the major goals of the stakeholders in production system engineering is improving the productivity of the engineering project, e.g., by avoiding unnecessary repeatable work. To this end, there is a need to support increasing the quality and efficiency of the engineering process by representing engineering process responsibilities and process states linked to the production system plant model. This need extends N3 with respect to knowledge on engineering processes

N7-Provisioning of integrated engineering knowledge at production system runtime. In a flexible production system context, domain experts need engineering knowledge at runtime to assess in a situation, which needs changing the system, the set of options for a successful change. In addition, changes have to be documented in a way that supports future change analysis. Therefore, there is a need for providing integrated engineering knowledge at system runtime beyond simple printouts of engineering plans. The knowledge has to be available in a sufficiently timely manner to support applications that depend on reacting in time to real-time processes.

2.1.2 The Challenges

Lüder and Schmidt [LS17] identify a set of concrete technical tasks that are still challenging to perform in the general area of mechatronic engineering, which subsumes the MDEE setting. They identify that techniques are needed for *model generation*, *model transformation*, *model integration*, and *model consistency management*. We consider these tasks put forward by Lüder and Schmidt [LS17] as good indicators for typical technical tasks that engineering applications should solve.

In addition, there is a challenge to allow KCM tasks to support flexible model integration and analysis of engineering processes [WMM$^+$11b, MB15]. KCM is often required since the process of designing complex mechatronic objects, such as industrial production systems, requires iterations and redesign phases, which lead to continuous changes of the data and knowledge within the MDEEs. To deal with these changes, stakeholders need to keep data versions, move backwards to previous versions, and query different versions of large data from heterogeneous local data sources. Furthermore, the effective and considerate propagation of changes is essential to ensure a consistent view of the project, to minimize defects and risks, as well as facilitate acceptance of new solutions by domain experts.

In this section and our thesis in general, we focus our attention on the following two engineering challenges in the context MDEEs: (C1) The core challenge of model/data integration[1], which requires **N1-Explicit engineering knowledge representation** and **N2-Engineering data integration**, and (C2) The advanced challenge of KCM support, which requires **N3-Engineering knowledge access and analytics** and **N5-Flexible and intelligent engineering applications**. We will elaborate on each of these challenges next.

C1: Data Integration

Data integration aims to bridge semantic gaps in engineering environments between project participants and their tools, who use different local terminologies [AvdADtH06, HW03, MMMB10], thus ultimately supporting the analysis, automation, and improvement of multidisciplinary engineering processes. Semantic data integration is defined as solving problems originating from the intent to share information across disparate and semantically heterogeneous data [Hal05]. These problems include the matching of data schemata, the detection of duplicate entries, the reconciliation of inconsistencies, and the modeling of complex relations in different data sources [NDH05]. Noy [Noy04] identified three major dimensions of the application of ontologies for supporting semantic data integration: the task of finding cross-source mappings (semi-)automatically, the declarative formal representation of these mappings, and reasoning using these mappings.

Engineering setups introduce important constraints and challenges for the semantic integration of engineering knowledge, namely: (1) The high number of involved engineering disciplines with a limited terminological overlap between them, thus further hampering data integration possibilities; (2) The variety of software tools and tool data models in these engineering disciplines; (3) The requirement of domain experts to continue using their well-established tools and processes; (4) The use of domain-specific jargon to represent a (large) part of the engineering knowledge; and (5) The distributed and concurrent nature of engineering projects, with geographically dispersed experts working on the project at the same time. Such constraints make semantic integration challenging in engineering environments.

C2: Knowledge Change Management

The engineering process of industrial production systems (e.g., modern power plants or steel mills) often requires teams of engineers from diverse engineering domains (e.g., mechanical, electrical and software engineering) to work together. As a result, this design process typically takes place in a MDEE, in which experts from various engineering domains and organizations work together toward creating complex engineering artifacts (Serral et al. 2013). In such as process, domain specific engineers use their own tools to create models that represent parts of the final system. Therefore, the MDEEs is highly

[1]Please note that we use the term of *model integration* and *data integration* interchangeably in the context of this thesis

heterogeneous, as it involves a wide range of data models, processes, and tools that were originally not designed to cooperate seamlessly. Despite this situation, other engineers and project managers need to perform tasks that require access to project-level data as opposed to domain specific data alone. For these stakeholders, there is a need for accessing integrated data at project level.

In addition to the characteristics described above, the process requires iterations and redesign phases, which lead to continuous changes of the data and knowledge within the MDEE [MB15]. To deal with these changes, stakeholders need to keep data versions, move backwards to previous versions, and query different versions of large data (schema and instances) from heterogeneous local data sources. Furthermore, the effective and considerate propagation of changes is essential to ensure a consistent view of the project, to minimize defects and risks, as well as facilitate acceptance of new solutions by domain experts [WMM$^+$11b]. To achieve this, changes originating from one engineering discipline need to be communicated and coordinated with participants of other disciplines, where those changes are relevant. Ideally, this communication should focus on high-level changes (e.g., defined in terms of domain concepts such as “Motor X updated to new version”) as opposed to low-level individual changes (i.e., change operations on versioned files) to ease the data analysis process.

To cater for all these needs, the stakeholders need processes and tools support for knowledge change management and analysis (KCMA) within the MDEE.

2.2 The Semantic Web: Background and Relevance to Engineering

The successful implementation of the World Wide Web led to an explosive growth of data available on it [HA99]. A search of "*Ekaputra*" on Google to look for this thesis author, for example, will lead to more than 1.5m pages in result, including a number of (distinct) research scientists and Indonesian travel agents, among others. This growth posed challenges for information retrieval processes, where one of the proposed solutions was to annotate web content with machine-processable representations.

This idea of applying formal knowledge representation on the web to help addressing the information retrieval challenge was started in the 1990s, e.g., SHOE (Simple HTML Ontology Extensions) [LSR96] and OntoBroker [FDES98]. The idea was later associated with the Semantic Web (SW) vision, originated from a seminal paper of Tim Berners Lee [BLHL01], which described as "*an extension of the current Web, in which information is given well-defined meaning, better enabling computers and people to work in cooperation*". In this definition, the "well-defined meaning" is established through semantic descriptions, e.g., metadata of web pages.

In order to make these semantic descriptions interpretable by machines and to support information retrieval from the web, several principles must be followed [Sab16]. **First**, semantic descriptions should describe information in terms that impose precise meaning and reflect agreement of a wider community. Further, a collection of these semantic terms and the relations between them will form an ontology [Gru93]. **Second**, semantic descriptions should be expressed in a representation language that can be parsed and interpreted by computer programs. In particular, these languages have to have clearly specified semantics that can be leveraged to enable computer programs to derive new information, a process referred to as inference or reasoning.

2.2.1 The Use of Semantic Web Technologies

SWTs were originally developed with the aim to implement the vision of SW [HKR09]. To this end, the W3C as the standardization body of the web has published a number of standards for SWTs that, although originally developed for the web, can and have been applied in many other areas, such as, for instance, integration of genome data and media publishing [SBLH06].

A very good example shown in the e-Science domain where ontologies facilitate data interoperability between scientific communities in different sub-fields and allow them to share and communicate with each other that may in turn lead to new scientific discoveries. In this case, ontologies are used as a "*means of communicating and resolving semantic and organizational differences between "biological databases*" [SK08]. Semantic integration of datasets is achieved using ontologies as mediators. Furthermore, based on the formal nature of ontology languages, automated reasoning can be used to derive new knowledge as well as to detect potential errors and inconsistencies in ontologies [KBM+11].

Similarly to the scenario in the e-Science settings, the challenges of engineering of complex production systems (including CPPS) as discussed in Section 2.1.2 involve integrating and making sense of heterogeneous datasets produced by different engineering disciplines. To realize these use cases, the following needs –using the same coding with the needs from engineering domain explained in Section 2.1.1– must be fulfilled: (N1) abilities to explicitly represent engineering knowledge, (N2) to integrate engineering knowledge, (N3) to provide access on (*the integrated and versioned*) engineering knowledge; and (N5) to develop a flexible and intelligent applications, such as KCM supports for MDEE.

This and the following two sections (2.2-2.4) will explore SWTs required for addressing the needs from the selected set of challenges previously mentioned: We address the needs of explicit representation of knowledge (N1) using Ontologies (Section 2.2.2) and Knowledge Representation languages (Section 2.2.3). Further, we tackle the data integration need (N2) with the OBDI approach (Section 2.3). Knowledge access and analytics (N3) is enabled by the Semantic query languages (Section 2.2.3), and lastly we handle the specific application support of KCM (N5) with the ontology change supports (Section 2.4).

2.2.2 Ontologies

An ontology is a technical artifact that acts as a centerpiece of any Semantic Web-based solution and allows the explicit and formal representation of knowledge relevant for the application at hand. Adopters of Semantic Web solutions therefore need to acquire an

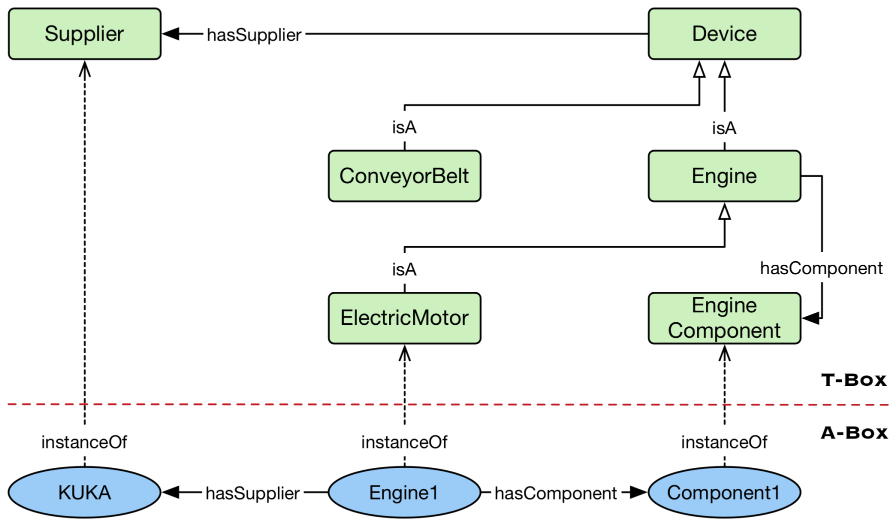


Figure 2.1: An example ontology in the engineering domain

ontology either by creating it themselves or by reusing one from similar applications. This section defines ontologies, explains the main elements of an ontology and describes a set of characteristics to be considered when reusing ontologies.

Studer et al. (1998) define an ontology as "*a formal, explicit specification of a shared conceptualization*". In other words, an ontology is a domain model (conceptualization) which is explicitly described (specified). An ontology should express a shared view between several parties, a consensus rather than an individual view. Also, this conceptualization should be expressed in a machine-readable format (formal). As consensual domain models, the primary role of ontologies is to enhance communication between humans (e.g., establishing a shared vocabulary, explaining the meaning of the shared terms to reach consensus). As formal models, ontologies represent knowledge in a computer-processable format thus enhancing communication between humans and computer programs or two computer programs.

For example, a mechanical engineering ontology, such as depicted in Figure 2.1, could describe concepts such as `Conveyer` or `Engine` and their relations, such as `hasSupplier`[2]. Data items (e.g., a specific engine referred to as `Engine1`) are then described in terms of ontology concepts (e.g., by associating `Engine1` to the concept `ElectricMotor` by means of the `instanceOf` relation)[3].

2.2.3 The Semantic Web Languages

To represent Semantic Web specific data, a set of languages have been developed, most notably RDF (Resource Description Format), RDFS (RDF Schema) and OWL (Web Ontol-

[2]Concepts and relations in an ontology are also called terminological components or T-Box)
[3]The data items in an ontology are called assertion components or A-Box)

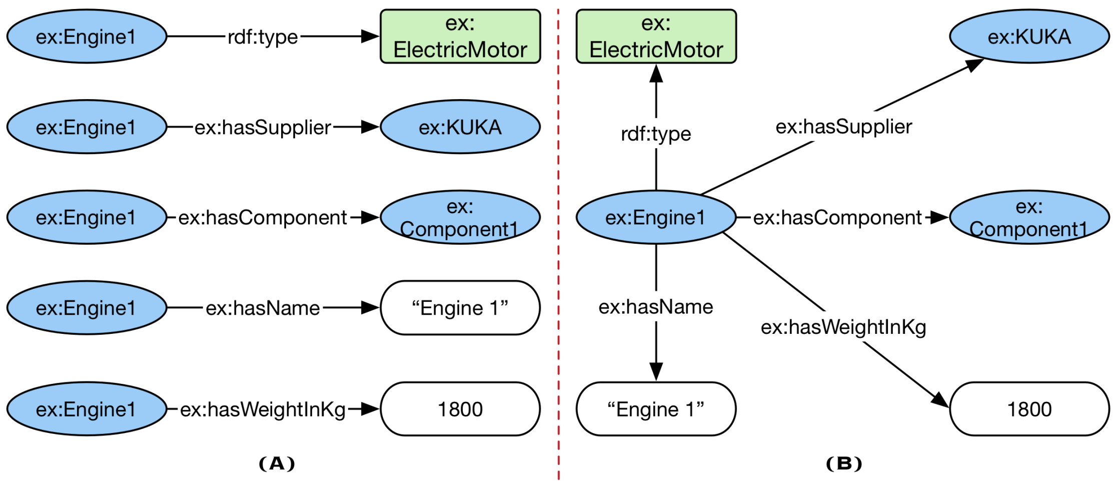


Figure 2.2: Example RDF triples (A) and their integration in an RDF graph (B)

ogy Language). While relational databases rely on a relational (i.e., table like) data model, Semantic Web specific languages adopt a triple (or graph based) model with data being represented as triples. For example, to declare that `Engine1` is an `ElectricMotor`, a triple is created stating that `<Engine1, isA, ElectricMotor>`. Figure 2.2 illustrates triples that refer to the `Engine1` resource (part A) and show how these are combined into an equivalent graph based structure (part B).

In addition to the data representation languages, W3C introduces SPARQL query language, which allows querying semantic data represented in RDF and therefore plays a key role in many applications built using SWTs. We will briefly explains each of these aforementioned languages in the following.

The Resource Description Framework[4] **(RDF)** is a language for describing resources on the Web and was adopted as the data interchange model for Semantic Web languages. Resources can refer to anything including "*physical things, documents, abstract concepts, numbers, and strings*" [CWL14]. RDF allows expressing relationships between two resources through RDF statements. RDF statements consist of three elements: a subject (a resource or a blank node), a predicate (a property/relation), and an object (a resource, a blank node or a literal value), which are collectively referred to as triples. Furthermore, a set of RDF triples constitutes an RDF graph.

An important principle of RDF is that individual triples can be merged whenever one of their resources is the same, as shown in Figure 2.2. This characteristic of RDF facilitates tasks that require integrating data from various sources, for example, from different webpages that provide (potentially) different information about the same entity (for example, the same person or company). This characteristic differentiates the graph-based RDF data model from more traditional, relational data models.

[4]RDF: https://www.w3.org/RDF/

The RDF Schema[5] **(RDFS)** is a lightweight data modeling vocabularies which is compatible with RDF. RDFS provides basic means of structuring data, e.g., `rdfs:Class` to declare that an object is a class (i.e., ontology concepts), `rdfs:subClassOf` to declare a subsumption relation between two classes and `rdfs:subPropertyOf` to declare a subsumptions relation between two properties. The availability of these classes and properties in RDFS making it possible for a reasoner to run the *inference mechanism*, e.g., to deduce that the instance of class A is also an instance of class B, if class A is a subclass of class B.

In addition, RDFS offers a (limited) set of properties to relate a resource with literal values, such as `rdfs:label` to provide a human-readable version of a resource name and `rdfs:comment` to describe a resource in a human-readable way.

Web Ontology Language (OWL) is a Semantic Web language designed to represent rich and complex knowledge about things, group of things, and relations between things. OWL could also be seen as an RDF vocabulary since it is compatible with the RDF data model. OWL is a W3C recommendation since 2004 and its successor, OWL 2, is a W3C recommendation since 2012.

An example of a basic concept in OWL 2 lies in its class hierarchy. In an OWL 2 class hierarchy, there are two predefined class identifiers, namely `owl:Thing` and `owl:Nothing`. The class extension of `owl:Thing` is all individuals, while the class extension of `owl:Nothing` is an empty set. Consequently, every OWL 2 class is a subclass of `owl:Thing` (since it is the super class of any class) and has `owl:Nothing` as its subclass (since owl:Nothing is always at the bottom of the class hierarchy). For classes, OWL 2 provides constructs beyond simple subsumption hierarchies of RDFS. One can specify, e.g., classes that have exactly the same set of instances with `owl:equivalentClass` or specify classes that share no instances at all with `owl:disjointWith`.

Unlike RDF(S), OWL 2 distinguishes object properties (i.e., properties where the range is another instance identified by an IRI, declared with `owl:ObjectProperty`) and datatype properties (i.e., properties where the range is a literal —e.g., string, number, date— declared with `owl:DatatypeProperty`).

SPARQL Query Language is a W3C recommendation [HS13] for querying and manipulating RDF graph data and is widely used in the Semantic Web community. SPARQL[6] can be used for querying and manipulating RDF graph data across different data sources as well as a single source. Semantic Web datasets are can be made accessible through a SPARQL Endpoint, which enables querying that dataset via HTTP (Hypertext Transfer Protocol). In principle, SPARQL works on the principle of mathing query patterns over an RDF data graph. Listing 2.1 shows a basic SELECT query that returns all the component of `ex:Engine1`.

Listing 2.1: A SPARQL query example

[5]RDFS: https://www.w3.org/TR/rdf-schema/

[6]SPARQL 1.1: https://www.w3.org/TR/sparql11-query/

```
PREFIX ex: <http://data.ifs.tuwien.ac.at/ns/mdee#>
SELECT ?component
WHERE {
    ex:Engine1 ex:hasComponent ?component
}
```

In addition to the basic SELECT queries that return tabular results, there are also other types of queries, i.e., CONSTRUCT queries, which return RDF graphs, ASK queries that return boolean, and DESCRIBE queries that return a single result RDF graph containing RDF data about resources. Further functionalities, such as FILTER, ORDER, and GROUP BY are available for advanced querying and manipulation of RDF graph data.

2.3 Ontology-Based Data Integration

Semantic Web technologies are well suited to support large-scale data integration scenarios [WVV^{+}01, Noy04]. Ontologies can be used to provide a semantic bridge for information integration. Concretely, ontologies are often developed with the goal to support data integration [Noy04]. For example, developers of several applications can agree on a general ontology and then extend this ontology with concepts and properties specific to their own applications. Since individual applications share a common semantic ground, this enables easily finding correspondences between them and therefore integrating their data. A set of high level ontologies such as SUMO [NP01] and DOLCE [GGMO03] have been developed specifically for supporting data integration scenarios.

Ontology-Based Data Integration (OBDI) refers to the use of (potentially several layers of) ontologies that capture knowledge across heterogeneous data sources to achieve semantic interoperability between these sources [WVV^{+}01]. Figure 2.3 illustrates three OBDI variants and their components based on a categorization introduced by Wache et al. [WVV^{+}01]: *single-ontology*, *multiple-ontology*, and *hybrid* OBDI . This classification reflects the number and type of ontologies used for data integration.

We distinguish among four layers of OBDI components as shown in Figure 2.3: (1) Data sources represent the (heterogeneous) local data repositories, which need to be integrated. (2) The local ontology layer contains so-called “local ontologies”, which represent the content of each individual data source repository. (3) The global ontology layer contains so-called “global ontologies”, which are semantically sufficiently broad to represent the data from all data sources to be integrated. (4) The software applications layer represents the applications, which access the data integrated with OBDI.

[7] Red arrows indicate access from an application to data, black arrows represent transformation/virtual access to the data; dotted green arrows represent implicit relations between involved ontologies, and numbered items show the sequence of system development.

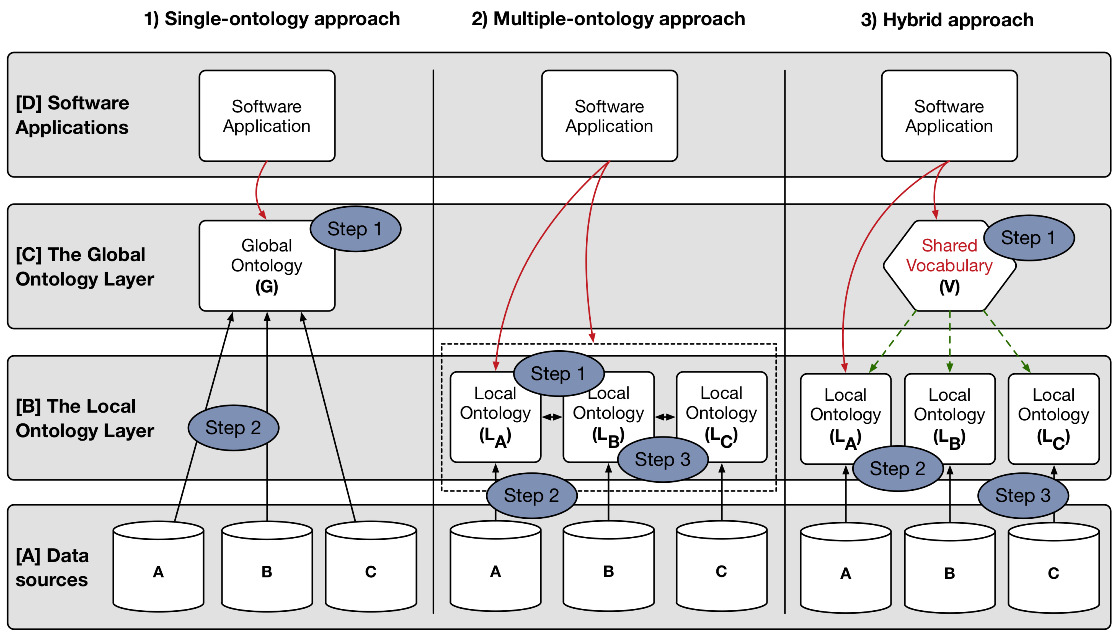


Figure 2.3: Three variants of OBDI from [WVV+01]: (1) single-ontology, (2) multiple-ontology, (3) hybrid.[7]

Assuming three data sources A, B and C, their integration can be achieved by means of three alternative OBDI variants as follows:

The single-ontology OBDI relies on a single global ontology to integrate all data sources (cf. Figure 2.3-1). In this approach, the integration process consists of two steps: (i) define a single global ontology G and (ii) transform source data from A, B, and C into the global ontology G. This integration process is typically hard to maintain because it is susceptible to changes in each data source. Any time a change occurs in one of the data sources, a decision has to be made whether to push the change to the global ontology. If so, to ensure compatibility, the global ontology as well as all mappings to all data sources must be updated.

The multiple-ontology OBDI involves a local ontology per integrated data source and an alignment of these ontologies with each other using semantic mappings (cf. Figure 2.3-2). Examples for this mappings include SPIN [KHI11], SPARQL Construct [HS13], and EDOAL. The integration process consists of three steps: (i) create local ontologies LA, LB, and LC for data sources A, B, and C, respectively, (ii) transform source data of A, B, and C according to their respective local ontologies, and (iii) establish semantic mappings between related ontologies. The drawback of this approach is that semantic mappings among involved ontologies are hard to define and maintain due to varying granularities of the local ontologies. Also, each inclusion of a new data source requires additional semantic mappings to all existing local ontologies.

Finally, **the hybrid OBDI** is similar to the multiple-ontology OBDI as it is characterized by definitions of a local ontology per data source. However, instead of independent alignments among local ontologies, this approach defines a shared vocabulary (i.e., a set of basic terms of a domain, which sometimes is also an ontology [VSWV00]) to be used and/or extended within local ontologies, i.e., by means of ontology refinements (cf. Figure 2.3-3). In this approach, the integration process consists of three steps: (i) define a shared vocabulary V that contains basic terms/concepts of the domain, (ii) create three local ontologies LA, LB and LC by using and/or extending the shared vocabulary V for data sources A, B, and C respectively, and (iii) transform/annotate source data from A, B, and C according to local ontologies LA, LB, and LC.

Research Gaps. Despite the availability of OBDI and all its variants, taking into account the complexity of data integration scenarios in MDEE, choosing the most appropriate OBDI variant, as well as particular suitable technologies is still challenging. Appropriate choices are mainly determined by the specific characteristics of the problem setting, such as data source heterogeneity or mapping complexity between the data sources. We identify such a challenge as an important research gap to address in this thesis.

2.4 Knowledge Change Management in Ontology-Based Data Integration

The emergence of OBDI approach for data integration raises issues on how to maintain the integrated system, i.e., how to manage the knowledge change within the system. KCM support is an important requirement of an OBDI system, especially in a mission- and safety-critical systems such as those in the production systems engineering [MB10] 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 parts of the system to ensure its consistency.

The current approaches to deal with ontology change from the SW community are mostly focused on a single OBDI variant and therefore these approaches are not sufficient to support ontology changes in other OBDI variants with multiple ontologies and complex mappings.

Note that in the SW research community, many relevant (and sometimes confusingly related) terms that are used to describe changes in ontologies and their instances. Flouris et al. provide an excellent summary and distinction of many ontology change terms that are used in the Semantic Web community [FMK$^+$08] Three of the most relevant terms are the following, of which we refer throughout our work:

- **Ontology change**, which is defined as "*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*". Note that we use the term of "knowledge change management" and "ontology change" interchangeably within the context of the thesis.

- **Ontology evolution**, which is a process of modifying an ontology in response to a certain change in the domain or its conceptualization, and
- **Ontology versioning**, an ability to handle an evolving ontology by creating and managing different variants/versions of this ontology.

We will describe next major related works relevant to the KCM in OBDI approaches from the SW research communities, namely Processes in Ontology Change, Change Detection and Representation, and Tool Support for Ontology Change.

Processes in Ontology Change Recent work from Zablith et al. [ZAD$^+$15] has summarized major ontology evolution process approaches [Kle04, NCLM06, Zab09, VPT$^+$05, Sto04]. They proposed five steps for the ontology evolution process:

1. **Detecting Evolution Need**. This step initiates the ontology evolution process by detecting the need for such change, which can be internal or external to the ontology. The goal is to determine whether to an extent whether changes should be conducted.
2. **Suggesting Changes**. This step collects and suggests a set of possible concrete ontology changes that can be executed according to the need of changes in the previous step. The goal is provide the decision maker with all possible options and let him or her decides which operations should be executed.
3. **Validating Changes**. This step filters out all changes from the suggested changes that are not not necessary to address the evolution needs. There are two typical types of this validation: (i) *domain-based validation* that check potential changes to ontology content, and (ii) *formal properties-based validation* that validate changes againts a set of formal techniques.
4. **Assessing Impact**. This step focuses on assessing the external impact of changes that is being applied to an ontology. This means how the changes on the ontology affect applications and usages that depend on the ontology.
5. **Managing Changes**. This step address tasks related to tracking performed changes and versions of ontologies, including their provenance information. The step would serve as a basis for advanced features to be executed, e.g., restoration of previous version of ontologies.

These processes are designed with the focus only on the changes in an ontology schema. A research that is closer to ours comes from Papavassiliou et al., where they take into account changes both in the ontology schema and data [PFF09b]. However, these approaches mainly consider changes in a single ontology instead of in the context of OBDI systems, which presents the challenge of data transformation and change propagation across the system.

Change Detection and Representation. A major requirement for the knowledge change management in OBDI 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 [PFF09b] and represent them in a machine readable format for future analytics, while considering their effects on the change propagation process. To address change detection between two ontology versions, the use of heuristics algorithms [NM02], structural differences [PFF09b, RN11], and OWL reasoning [GPS10] have been proposed and evaluated. To support the change detection mechanism, approaches for change representation as change ontologies [NCLM06, PHCGP09] and change languages [PFF09a] have also been proposed. A different change scenario happens when two copies of repositories changed in parallel, which may results in conflicts in such co-evolution settings. To address this, Faisal et al. propose a set of conflict resolution strategies [FES+16]. The aforementioned approaches assume that users are interested in the whole Knowledge Graph. Endris et al. suggests that this is not always the case, since many users are interested only in a certain part of the graph, which can be represented as interest expressions [EFO+15].

Similar to ontology change process, the approaches in this area are typically focused on detecting changes of a single-ontology OBDI. 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 OBDI system ontologies and their complex mappings.

Tool Support for Ontology Change. 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 [SM02]. In the similar timeframe, PrompDiff change detection algorithm was integrated into Protégé tool [NM02]. 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 [NCLM06]. The latest addition to the impressive set of Protégé ontology change support tools is the Protégé versioning server, which is based on the previously proposed architecture client-server architecture [RSDT08].

Recent researches on ontology change focus more on ontology versioning. One of the first in this area ia an interesting work comes from the adaptation of distributed versioning systems, SemVersion [VG06] and R&Wbase [VCV+13], which provide support for ontology versioning similar to source code versioning systems. Arndt et al. [ARM16] goes a step further, which proposed a Git adaptation for collaborative authoring of RDF data. Graube et al. proposes another approach that utilizes SPARQL extensions to query older versions of data using query re-writing technique [GHU16], as an extension to the original R43ples versioning system that utilizes named graph to handle RDF Graph versioning [GHU14]. Another approach comes from Frommhold et al., which proposes a method for capturing information from changes in arbitrary RDF datasets [FPA+16].

Their approach emphasizes support for protection against version history manipulations and blank nodes, which is mainly left unhandled in other approaches.

To ensure and compare the quality of the proposed versioning methods, the first benchmark on querying RDF data archives is proposed [FUPK16]. In the paper, they explain the theoretical foundations of the benchmark, as well as the prototypical implementation of the benchmark, referred as BEAR (BEnchmark of RDF ARchives).

Research Gaps. The current approaches to deal with KCM from the SW community are mostly focused on a single-ontology OBDI and therefore not sufficiently address requirements for KCM in other OBDI variants, e.g., data transformation and change propagation. Such supports are crucial to further the adoption of OBDI in MDEEs due to its importance in many MDEEuse cases [WMM+11b, MB15], and therefore worthy of further research.

2.5 Summary

The SWTs, especially the OBDI approach, with ontologies as key resources, has the potential to capture knowledge across heterogeneous data sources and create semantic interoperability between them to facilitate data integration [WVV+01], which is highly suitable to address data integration requirements in MDEEs. However, given the complexity of data integration scenarios in MDEEs, choosing the most appropriate OBDI variant, as well as particular suitable technologies is challenging. Therefore, providing an understanding on how suitable is the OBDI approach and its variants for the diverse data integration scenarios is highly important, both for researchers and practitioners. We regard such importance deserves further investigations in the context of our thesis. To this end, **Chapter 3** focuses on reviewing OBDI applications in the MDEEs from both architectural (i.e., OBDI variants) and technical points of view. In the same chapter, we introduce an additional variant of OBDI, called Global-as-View (GAV) OBDI and propose a guideline to help users in selecting the most suitable OBDI variant for their scenario. Furthermore, in **Chapter 4**, we tests the guideline created in Chapter 3 within two concrete use cases from our industry partners.

Another challenge lies in providing a better support for ontology change, especially on KCM on OBDI systems. The current approaches to deal with KCM from the SW community are mostly focused on a single-ontology OBDI and therefore not sufficiently address requirements for KCM in other variants of OBDI system, e.g., data transformation and change propagation. Such supports are crucial to further the adoption of OBDI in MDEEs and other similar environments, such as empirical research publication [BKR+14, ESSB15a], and therefore worthy of further research. We dedicate **Chapter 5** to provide the definition of a generic framework for KCM in MDEE as an extension of OBDI. We build on a set of requirements from MDEEs use cases to develop the framework. Further, we develop an instance of the framework using SW technologies based on a use case of hydro-power plant engineering from an industry partner to conduct a feasibility evaluation.

CHAPTER 3

Ontology-Based Data Integration in Multi-Disciplinary Engineering Environments: A Review

The lifecycle of modern production systems typically involves stakeholders from several engineering disciplines to collaborate in a Multi-Disciplinary Engineering Environment (MDEE), an environment where they work together to achieve a common goal [BGL17]. Processes in MDEEs have strong needs for data integration to overcome the inherent semantic heterogeneity in the environments. Data integration is also a basis for advanced capabilities to support the work of engineering teams, such as early defect detection [KSS$^+$14] and knowledge change management [ESSB16]. The Ontology-Based Data Integration (OBDI) approach, i.e., the use of ontologies that capture implicit knowledge accross heterogeneous data sources to achieve semantic interoperability [WVV$^+$01], has recently emerges as a potential solution. The OBDI has been utilized for several purposes in MDEE use cases, e.g., engineering design quality improvements [SRFF11, HVKE16] and simulation generation and evaluation [TU12, DVYP14].

However, given the complexity of data integration scenarios in MDEEs, choosing the most appropriate OBDI variant, as well as particular suitable technologies is challenging. Therefore, providing an understanding on how suitable is the OBDI approach and its variants for the diverse data integration scenarios is highly important, both for researchers and practitioners, as reflected in the first research question of this thesis **(RQ1)**: *How suitable are the Ontology-Based Data Integration approach variants for the diverse data integration scenarios in Multi-Disciplinary Engineering Environments?*

To address **RQ1**, in this chapter we reports on the literature survey of OBDI approaches in MDEEs, which has been published as a journal paper [ESS$^+$17]. The review is motivated by the increasing interest from both Automation System Engineering and

Semantic Web (SW) communities to adopt OBDI to tackle the challenge of integrating heterogeneous data in MDEEs. This interest has resulted in a growing body of literature that is dispersed across both research communities and has not been systematically reviewed so far. We address this gap with a survey reflecting on OBDI applications in the context of MDEEs. To this end, we analyze and compare 23 OBDI applications reported in 29 papers published in venues from both communities.

As the main results of our survey, we (i) propose an extension to the OBDI variant classifications based on our observation of their applications in MDEEs, (ii) identify key problem context characteristics, (iii) compare strengths and limitations of OBDI variants as a function of problem context, and (iv) provide a recommendation guideline for the selection of OBDI variants based on characteristics of MDEEs. These results provide a basis to answer **RQ1**, which will be evaluated in the following Chapter 4.

This Chapter is structured as the following: Section 3.1 explains the survey methodology to select and retrieve research publications following the Systematic Literature Review (SLR) approach [KC07, ZRM$^+$15]. Section 3.2 briefly introduces the OBDI applications retrieved in the survey, which consist of 23 OBDI applications from 29 publications. Section 3.3 describes the characteristics of data integration scenarios in MDEEs, grouped by their objectives. Section 3.4 reports on the technical realizations of OBDI elements, categorized in the following categories: Ontology Languages and Frameworks, Data Acquisition methods, Semantic Mapping and Transformations, and Storage and Data Access. Section 3.5 reports on our analysis of the OBDI variant adoptions against the data integration characteristics of MDEEs. In this section, we also describe an additional OBDI variant beyond the three variants already described in Section 2.3 as a result of our analysis. Section 3.6 presents our guideline for selecting the most suitable OBDI variant based on use case characteristics, and finally we summarize the chapter in Section 3.7.

3.1 Paper Selection Methodology

To understand the current landscape of OBDI applications in MDEEs, we identified relevant research articles from SW and Automation System Engineering (ASE) communities through the SLR method [KC07, ZRM$^+$15] covering the following steps (see Figure 3.1):

1. **Keyword-based search** on article title published at selected conferences in the SW and ASE research communities. Different sets of keywords were used for the two target communities, which resulted in a large number of research papers for further processing.

2. **Definition and application of the inclusion/exclusion criteria**. Taking into account the inclusion and exclusion criteria, we analyzed the paper titles, abstracts, and content. Based on this analysis, we reduced the number of selected papers to include only the most relevant papers.

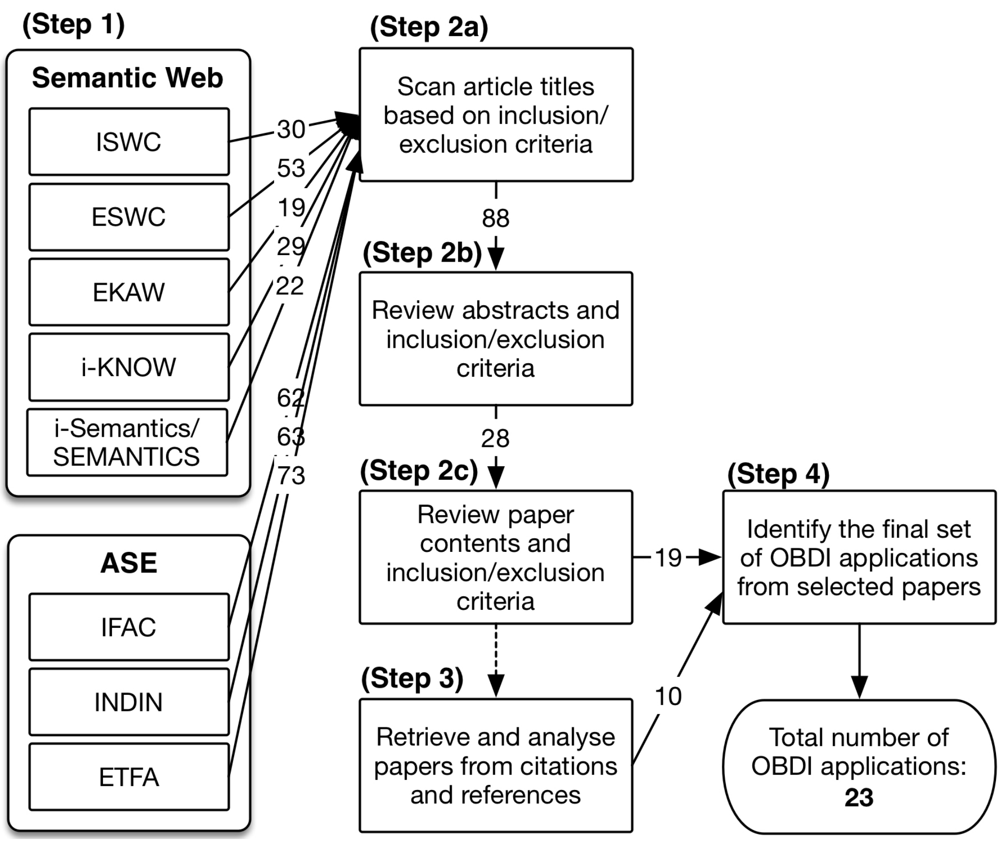

(Step 1)
Semantic Web
ISWC
ESWC
EKAW
i-KNOW
i-Semantics/ SEMANTICS
ASE
IFAC
INDIN
ETFA
30
53
19
29
22
62
63
73
(Step 2a)
Scan article titles based on inclusion/ exclusion criteria
88
(Step 2b)
Review abstracts and inclusion/exclusion criteria
28
(Step 2c)
Review paper contents and inclusion/exclusion criteria
19
(Step 3)
Retrieve and analyse papers from citations and references
10
(Step 4)
Identify the final set of OBDI applications from selected papers
Total number of OBDI applications: 23

Figure 3.1: Number of articles and OBDI applications retrieved during literature search

3. **Retrieval of further potential articles** from citations and references of selected papers. To avoid missing important papers, we also took a look on paper references and papers that cite the selected papers.

4. **Identification of the final set of OBDI applications** from selected papers to be further analyzed. Some of the papers reported different applications of the same or similar methods. In this step, we group these applications as one.

We will explain each of the SLR steps of the survey in Section 3.1.1 - 3.1.3.

3.1.1 Step 1: Keywords-based search

In our survey, we limit our keyword search to research articles published in five main conferences of the SW community (ISWC, ESWC, i-Semantics/SEMANTiCS, i-KNOW, and EKAW) and three main conferences of the ASE community (ETFA, IFAC, and INDIN). We executed the keyword search on all selected conferences between 2010 – 2016 using the Scopus search engine, with the exception of the 2016 edition of i-KNOW (not indexed by Scopus – skipped) and ISWC (metadata did not mention ISWC – manual search). The keyword search yielded more than 350 papers (Figure 3.1, Step 1).

We use a separate set of keywords for SW and ASE conferences. Both sets of keywords omit the "data integration" term, as this keyword typically does not appear in the title. For SW conferences, we assume that ontology-related keywords are unnecessary, as it is implied with the article submissions to conferences in this research area. Therefore, we focus on keywords related to the domain, e.g., engineering or production (cf. Listing 3.1). In contrast, for conferences in the ASE domain, we focus our search on ontology-related keywords with supplementary terms that specify our focus on the domain, which are "production system" or "production plant" (cf. Listing 3.2).

Listing 3.1: Keywords for SW conferences

```
automation OR engineering OR product* OR system OR
production OR manufactur* OR energy OR plant.
```

Listing 3.2: Keywords for ASE conferences

```
ontolog* OR semantic OR knowledge*base OR
"linked data" OR "production system" OR "production plant"
```

3.1.2 Step 2: The Definition and Execution of Inclusion and Exclusion Criteria

We set the following inclusion and exclusion criteria to remove irrelevant papers from the list of papers identified with the keyword-based search in the previous step.

Inclusion Criteria:

- IC1. Paper contains scenarios or use cases of data integration using ontologies in the automation system engineering domain. We consider ASE applications that are using ontologies for integration purposes.

- IC2. Ontology languages or frameworks used for data integration are explicitly mentioned and explained. We consider ASE applications that explicitly mentioned the ontology languages or frameworks that is used within their proposed method.

Exclusion Criteria:

- EC1. The reported approach involves only a single data source. We only consider ASE applications that integrate data from several data sources to exclude applications that use ontologies only for analysis purposes.

- EC2. Non-OBDA relational database or purely *Eclipse Modelling Framework* (EMF)-based approaches. We consider EMF-based approach as a different method for data integration and therefore did not include them as targets of this survey.

Step 2a: Inclusion/Exclusion. We applied the inclusion and exclusion criteria first on the paper titles, which resulted in a set of 88 papers (Figure 3.1, Step 2a).

Step 2b: Abstract Analysis. In the next step, we applied the criteria to the abstracts of the remaining 88 papers, reduced the overall set of papers to 28 papers (Figure 3.1, Step 2b).

Step 2c: Content analysis. There were cases where the abstract did not clearly justify an article's inclusion or exclusion. In these cases, we analyzed the content of the paper to take the final decision (Figure 3.1, Step 2c). As a result, we shortlisted 19 papers.

3.1.3 Step 3 & 4: Retrieval of further potential articles and Identifying the final set of OBDI applications

The keyword-based search only covered a limited number of publications on the topic. To extend our set of considered papers, we conducted an additional search based on references and papers that were cited by the 19 papers from the shortlist we obtained in the previous step. As a result of the third step, we added ten additional papers (Figure 3.1, Step 3) and resulting in a set of 29 papers.

After further reading, we realized that several of these 29 papers covered the same approaches or extensions thereof. Therefore, as a last step, we grouped these papers accordingly and arrived at the final 23 OBDI applications (Figure 3.1, Step 4). These are listed in Table 3.1, which summarizes a set of OBDI applications in MDEEs classified along the life cycle of production systems as the result of our survey.

Eleven applications focus on the design phases (planning and design) for purposes such as design validation, quality improvement, simulation generation and evaluation (Section 3.2.1). Six applications focus on the run-time phases (startup, production, and service) for system monitoring, diagnostic, evaluation and transient data integration (Section 3.2.2). The remaining six applications address both design and runtime phases to support tasks such as integrated data analysis (Section 3.2.3).

3.2 Survey Results

In this section, we provide a brief introduction of selected OBDI applications in MDEEs retrieved for our survey. We classify the OBDI applications along production system lifecycle stages [BÖF+10] as shown in Table 3.1 and briefly explain in the following:

- **Planning of assembly and production processes**. In this phase, plant planners decide on manufacturing processes and resources necessary for building a plant. This phase can also be considered as the pre-project phase.

- **Production plant design**. In this phase, engineers work within their respective domains to build the design of a production system. The phase includes exchange of design data among involved engineering disciplines.

- **Virtual and actual start-up**. The virtual start-up validates the production plant design by systematically iterating through planned and potential plant operation scenarios. The actual start-up of a plant involves plant adjustments on the shop floor after the plant assembly process.

- **Production and service**. Monitoring and improvement of the production plant, manufacturing execution, predictive maintenance, and plant re-configuration are examples of tasks in this phase.

3.2.1 OBDI in the Design Phases

Dibowski and Kabitzsch [DK11] propose an Ontology-based Device Description approach, which aims to provide a formal, unified, and extensible production system device specifications using SW technologies. This approach uses several layers of ontologies, where the top level contains generic domain vocabularies that will be reused and extended in lower layers. Their approach implements a hybrid OBDI, where the top-level ontology is comparable to the shared vocabularies.

Imran and Young [IY16] demonstrate the potential of formal reference ontologies to support interoperability with a study case of manufacturing bill of materials. They use a Common Logic-based Knowledge Frame Language framework to define concepts within assembly systems in a multi-layered ontology approach. Their approach implements a GAV OBDI with a foundation ontology as shared vocabularies.

Lin and Harding [LH07] propose using ontologies to support collaboration of engineers involved in a manufacturing system engineering process. The proposal implements a Global-as View (GAV) OBDI (see Section 5.1), where the involved organizations develop their independent local ontologies and then map these to the global ontology. These mappings serve as a semantic bridge to exchange and integrate the data across these organizations.

Wiesner et al. [WMM11a] build on their previous work of the OntoCAPE ontology [MWM09] to develop an information integration approach in chemical process engineering, which is called the Comprehensive Information Base (CIB). CIB adopts the hybrid OBDI where they derive the shared vocabulary from OntoCAPE and develop source (local) ontologies for several local data sources based on the global ontology. They use a two-layer local ontology approach: (i) Import ontologies, which are derived directly from data sources (e.g., XML files) using (semi)-automatic data transformation, which are later transformed into (ii) Document ontologies that are conformed to the shared vocabularies. They use F-Logic instead of the standard RDF/OWL languages to represent all facts,

Table 3.1: An overview of OBDI approaches in MDEEs classified according to relevant production plant life cycle phases (no shadow) and OBDI variants (gray shadow)

OBDI approach classifications	Planning	Design	Start-up	Production & Service	Single-ontology	Multiple-ontology	Hybrid	Global-as-View
Aarnio et al. [ASF14]				X			X	
Abele et al. [ALGM13, AGZK14]				X	X			
Brecher et al. [BÖF^{+}10]	X	X	X	X	X			
Dibowski & Kabitzsch [DK11]		X					X	
Dubinin et al. [DVYP14]		X						X
Ekaputra et al. [ESSB16]		X						X
Feldman et al. [FHK^{+}15]		X	X			X		
Graube et al. (2013) [GZUH13]		X	X	X		X		
Graube et al. (2016) [GUH16]				X	X			
Hennig et al. [HVKE16]		X			X			
Imran and Young [IY16]		X						X
Kovalenko et al. [KSS^{+}14]		X				X		
Lee & Kim [LK07]				X	X			
Lin & Harding [LH07]		X						X
Natarajan et al. [NS14a]				X	X			
Novak and Sindelar [NS13]		X	X		X			
ONTO-PDM [PDT12, GAP^{+}12]				X	X			
Optique [KJRZ^{+}13, KSÖ$^{+}$14, SHL^{+}14]		X		X		X		
Sabou et al. [SEKB16]		X			X			
Softic et al. [SRD^{+}13]		X			X			
Strube et al. [SRFF11]		X			X			
VFF [TU14, KTS13, TTU15]		X	X	X	X			
Wiesner et al. [WMM11a]		X					X	

rules, and queries. The authors argue that F-Logic is more suitable for defining rules for integration and mapping purposes as well as for the formulation of expressive queries.

Strube et al. [SRFF11] propose an approach to combine the CAEX data format [IEC08] and SW technologies to support re-developments/modernization of plant automation. The approach involves integrating several CAEX instance files containing plant designs and their proposed changes, together with a set of rule definitions to validate plant changes. These data are integrated using a single-ontology OBDI that is using an adaptation of the CAEX data model as a global ontology. They define a set of SWRL [HPsB+04] rules for validating the proposed changes in the modernization process of plant automation.

Softic et al. [SRD+13] semantically integrate data from several data sources to track engineering tasks in an automotive product lifecycle within a single-ontology OBDI. Their architecture consists of three layers: (1) Data layer, where their approach acquires data from local data sources, (2) Entities layer, where they store and link data, and (3) View layer, where users interact with the integrated data. Two different views of data are defined in the view layer: (a) project managers' view and (b) engineers' view, which allow the system to provide different focus on the integrated data.

Dubinin et al. [DVYP14] introduce an approach based on GAV OBDI for integrating information across data sources in the automation domain. Rather than the typical local ontologies development based on a shared global ontology, they develop the global ontology independent of the local ontologies. To transform local ontology data into instances of the global ontology, they introduce the eSWRL transformation language as an extension of SWRL [HPsB+04] for RDF-to-RDF transformation.

Kovalenko et al. [KSS+14] focus on the use of SW technologies to detect defects early in the power plant engineering process. To this end, they adopt the multiple-ontology OBDI to integrate heterogeneous data from several engineering disciplines. They cooperate with domain experts to define links between data from several involved disciplines, i.e., mechanical engineering, electrical engineering, and project management. Furthermore, they develop a set of SPARQL queries to detect defects and validate power plant engineering data.

Ekaputra et al. [ESSB16] primarily focus on using SW technologies to support data change management within MDEE, where data changes in one engineering discipline need to be validated and propagated to other disciplines. To this end, they adopt a GAV OBDI to represent the heterogeneous data as local and global ontologies. Similar to Dubinin et al. [DVYP14], they develop both local and global ontologies independently from each other, and they use SPARQL queries to transform, validate and propagate changes between several local ontologies via the global ontology.

Hennig et al. [HVKE16] propose a SW-based approach to improve the semantic validity and the analysis capability of the multi-disciplinary engineering/system engineering of space systems. To this end, they integrate data from various engineering disciplines within the space system engineering (e.g., mechanical, electrical, instruments, control and software engineering) using the ECSS-E-TM-10-23A data exchange standard as a common (global) data model in a single-ontology OBDI. They focus on the inferencing capability of OWL2 to provide advanced analysis in their scenario.

Sabou et al. [SEKB16] develop the AutomationML Analyzer tool to support engineering of Cyber-Physical Production Systemss (CPPSs) according to the single-ontology OBDI, where they use an ontology form of the AutomationML data exchange format as the global ontology for integrating and analyzing AutomationML data from engineering disciplines. The combined data serves as a baseline to provide advanced capabilities to engineers, e.g., analysis and visualization of CPPS engineering design.

3.2.2 OBDI in the Runtime Phases

Aarnio et al. [ASF14] propose an adaptation of a hybrid OBDI to support condition-based monitoring in automation systems. They conduct a four-steps transformation process from local data to RDF: (1) Automatic transformation of source data from local source formats to temporary RDF data, (2) Transformation of temporary RDF data into instances of local ontologies, where the local ontologies conform to shared vocabularies, (3) Use of the SILK [64] tool to link between data from local ontologies, (4) Development and execution of rulesets on top of local ontologies to infer new information.

The two-level local ontology approach is similar to the approach in Wiesner et al. [WMM11a], with the difference that they are using the standard RDF/OWL language to represent both local and global ontologies with the help of SILK. They evaluate their approach with a set of SPARQL queries targeting both local and global ontologies.

Abele et al. [AGZK14] suggest utilizing SW technologies to support monitoring and diagnostic systems (MDS) in industrial applications. This approach builds on their previous work on the single-ontology OBDI that utilizes the Semantic Media Wiki infrastructure, rule ontology and Drools engine [AG13]. To this end, they integrate both static plant artifacts data from the design-time engineering and plant component states from run-time engineering to provide users with relevant MDS information.

Graube et al. [GUH16] propose a “mixed” solution based on a single-ontology OBDI to integrating static data (e.g., RDF data) and transient data (i.e., sensor data that is coming from web services) based on the URI dereferencing feature of SPARQL 1.1. An evaluation of the proposed solution offers sufficient performance to access transient data as an alternative to the currently available solutions (e.g., SSN, SensorML, and Linked Sensor Middleware).

Lee and Kim propose a framework for engineering collaboration for distributed product development [LK07]. They use SW technologies to integrate and facilitate the exchange of context information from several data sources (e.g., Bluetooth, PDA, Etc.). To this end, the framework deploys a single-ontology OBDI to model engineering contexts (e.g., locations of users and roles) and uses it to determine relevant services for stakeholders based on context data derived through inference.

Natarajan et al. [NGS12] propose an extension of the OntoCAPE ontology [MWM09], which is called OntoSAFE, to provide an application-oriented ontology focused on process supervision in large chemical plants. Later on, they utilize OntoSAFE as a basis for integrating and exchanging complex plant supervision data using the single-ontology OBDI [NS14a].

Kharlamov et al. [KJRZ^{+}13] explain the underlying OBDI approach (OPTIQUE) that can be used in MDEE to facilitate data integration using a multiple-ontology OBDI and OBDA. Two example applications in MDEE based on this approach are: Kharlamov [KSÖ+14] and Solomakhina [SHL+14]. **Kharlamov et al.** [KSÖ+14] propose an OBDA approach to improve access to large, heterogeneous and stream data at a large organization. To support the proposed OBDA approach, they develop a query repository to store both predictive and reactive analysis queries. They evaluate their approach in a large-scale scenario that involves a combination of static data and dynamic data from sensors ($>$ 30GB of new data produced every day). **Solomakhina et al.** [SHL+14] propose an ontology-based approach to improve the precision and recall of statistical data analysis in the domain of production systems. They integrate data from three different local ontologies that represent power generation facilities (i.e., Turbine, Sensor, and Diagnostics ontologies) with different OWL2 dialects (OWL2-QL and OWL2-DL). They show that their integration methods, which combine explicit domain models with SW technologies and statistical analysis, yield a better result compared to a pure statistical analysis.

Panetto et al. [PDT12] develop an approach to support product data interoperability between applications and stakeholders involved within manufacturing process environments. Their approach implements a single-ontology OBDI, with their proposed ONTO-PDM ontology based on two industry standards (i.e., ISO 10303 [ISO14] and IEC 62264 [IEC03]) as a common data model and mediator between applications during manufacturing process lifecycle. They implement the ontology in both OWL and relational database, and use First Order Logic (FOL) patterns to map between data coming from the two industry standards within the ONTO-PDM ontology. **Giovannini et al.** [GAP+12] extend ONTO-PDM with concepts and rules on sustainability principles and technology knowledge. In addition, the authors propose a knowledge base system that use formalized knowledge for supporting product design and process planning. The approach uses SWRL rules to infer additional information and conduct analysis related to sustainability of products.

3.2.3 OBDI in the Production System Lifecycle

Brecher et al. [BÖF+10] aim for software tool integration in production plant lifecycles with SW technologies. Their approach implements the single-ontology OBDI. They develop an information model as a common ontology for production plant lifecycles and connect a set of software tools via data interpreter and generic interfaces. They use the Globally Unique Identifiers or unique names to identify the same objects in different data sources. The integrated data is used to navigate through production plant lifecycles, including the planning phase of the production process and the assembly process.

Feldmann et al. [FHK+15] introduce an inconsistency management approach based on SW technologies. The approach integrates two types of data: SysML4Mechatronics data that represent the mechatronics architecture and Matlab/Simulink data representing workpieces throughput of the plant in a system that implements a multiple-ontology OBDI. In this approach, relations between the two ontologies are defined manually by domain experts. They develop a set of SPARQL queries to detect inconsistencies in the integrated data and successfully retrieve inconsistency of the data as intended in their evaluation.

Graube et al. [GZUH13] suggest using linked data to allow orchestration of software applications in the production system environment. Their approach implements a multiple-ontology OBDI, where they represent various data sources (e.g., device details, plant structure, report-and-form information, and live data access) as separate local ontologies, and store the information about and the relation among these ontologies in a separate ONT ontology. These ontologies are then orchestrated to build various applications (e.g., Task-List applications and Neighborhood-Browser for data flow explorations) related to production systems.

Novak and Sindelar [NS13] proposes a single-ontology OBDI to support simulation design and integration of simulation models in industrial automation. The authors develop the automation ontology that serves as the global ontology of the approach that is wrapped in a java-based tool. The tool receives input data from engineers (plant designs) as well as knowledge about devices in the particular industrial plant and available simulation libraries. As outputs, it produces executable simulation configuration files for simulators based on SPARQL query result on automation ontology instances.

Kádár et al. [KTS13] propose the Virtual Factory Framework (VFF), an integrated collaborative environment to support the design and management of factory entities. VFF initiate a global ontology (Virtual Factory Data Model - VFDM) for integrating and representing factory objects related to production systems, resources, processes, and products, resembling the single-ontology OBDI. A Virtual Factory Manager builds on top of the VFDM to manage and provide access to the VFDM data from various connected tools. These tools act both as data providers as well as data users. **Terkaj and Urgo** [TU14] focus on integrating static data of production systems and their performance history, build on their previously explained VFF. The method allows evaluation of a system design by simulating its performance based on system and simulation logs. **Terkaj et al.** [TTU15] extends VFF to evaluate the impact of planning and maintenance decisions during the operation phase of a manufacturing system. They report on an application case of roll-shop system designs, where they develop a graphical user interface and combine it with a Discreet Event Simulation tool to evaluate the performance of roll-shop system configurations.

3.3 Characteristics of Data Integration Scenarios

As discussed in Section 2.1, MDEEs are characterized by the involvement of engineers from various engineering disciplines. This collaboration results in the need for integrating heterogeneous data sources produced by domain-specific software tools. We discuss characteristics of data integration scenarios in MDEE that we identified and generalized in our survey to address the following question: *What key characteristics of data integration scenarios in MDEE affect the choice of an adequate OBDI variant?*

Identifying these characteristics is also the first step to establish criteria that practitioners can use to choose appropriate OBDI variants for their settings.

3.3.1 Data Integration Objectives

There is a wide range of objectives for data integration in Multi-Disciplinary Engineering (MDE) settings. In this chapter, we do not directly derive recommendations for OBDI variant selection based on these objectives, but focus on the relationships between setting characteristics – explained in Section 3.3.2 and 3.3.3 – and OBDI variants. The data integration objectives we compiled from the papers are as follows (summarized in Table 3.2):

Objectives related to data:

- **Data Change Management** refers to the process of managing local data changes and their effects on the overall system [ESSB16]. For this particular scenario, data integration serves as a foundation to enable data change management.
- **Transient Data Integration**, such as stream data integration, aims to integrate transient data sources with combination with non-transient data [GUH16, KSÖ+14].
- **Centralized Engineering Repository** data integration scenarios aim to provide a centralized engineering repository (e.g., [HVKE16, TU14, LH07]).
- **Integrated Data Analysis** refers to typical OBDI approaches that aim to enable data analysis on top of integrated OBDI data (e.g., [BÖF+10, ASF14, SEKB16]).

Objectives related to the overall system:

- **Design Quality Improvement** aims at improving the quality of system design in MDEE, e.g., with inconsistency management [FHK+15] or defect detections [KSS+14, SRFF11, HVKE16] over a global view of data sources.
- **Design Validation** aims to validate system designs against a set of validation criteria based on integrated data [SRFF11, HVKE16, TU14, KTS13].
- **Simulation Generation and Evaluation** aim to generate [TU14, DVYP14] and evaluate [TU14] system simulation in MDEE.
- **System Monitoring, Diagnostic and Evaluation** aim for system monitoring [ASF14, AGZK14, NS14a], diagnostic [AGZK14] and evaluation [KTS13] in MDEE.

Objectives related to collaborations:

- **Team Collaboration**. This goal refers to the use of integrated data for supporting team collaborations [ESSB16, LH07, WMM11a, SRD+13].

Table 3.2: Data Integration Objectives for OBDI in MDEE *(No shading: data-related objectives; light shading: overall-system objectives; dark shading: collaboration objectives)*

OBDI Objectives	Data Change Management	Transient Data Integration	Centralized Engineering Repository	Integrated Data Analysis	Design Quality Improvement	Design Validation	Simulation Generation & Evaluation	System Monitoring, Diagnostics, & Evaluation	Team Collaboration	Software Interoperability
Aarnio et al. [ASF14]				X				X		
Abele et al. [ALGM13, AGZK14]								X		
Brecher et al. [BÖF$^+$10]			X	X						
Dibowski & Kabitzsch [DK11]				X						X
Dubinin et al. [DVYP14]							X			
Ekaputra et al. [ESSB16]	X			X					X	
Feldman et al. [FHK$^+$15]				X	X					
Graube et al. (2013) [GZUH13]										X
Graube et al. (2016) [GUH16]		X	X							
Hennig et al. [HVKE16]			X		X	X				
Imran and Young [IY16]										X
Kovalenko et al. [KSS$^+$14]				X	X					
Lee & Kim [LK07]										X
Lin & Harding [LH07]			X						X	
Natarajan et al. [NS14a]			X					X		
Novak and Sindelar [NS13]							X			
ONTO-PDM [PDT12, GAP$^+$12]									X	X
Optique [KJRZ$^+$13, KSÖ$^+$14, SHL$^+$14]		X	X	X						
Sabou et al. [SEKB16]			X	X						
Softic et al. [SRD$^+$13]									X	
Strube et al. [SRFF11]					X	X				
VFF [TU14, KTS13, TTU15]			X			X	X	X		X
Wiesner et al. [WMM11a]			X						X	

- **Software Interoperability**. This goal aims to provide a “common language” for software partners to interact with each other (e.g., for app orchestration [GZUH13], intelligent service finder [LK07], or data exchange [KTS13]).

3.3.2 Data Sources

In this section, we explain data-source related characteristics of OBDI scenarios in MDEEs.

Table 3.3: Data source types for OBDI in MDEE

Data source types	Relational Database	Spreadsheets	XML-based formats	RDF	Streaming / Sensor data	Others / Specific formats
Aarnio et al. [ASF14]	X	X		X		
Abele et al. [ALGM13, AGZK14]					X	
Brecher et al. [BÖF^{+}10]						X
Dibowski & Kabitzsch [DK11]						X
Dubinin et al. [DVYP14]				X		
Ekaputra et al. [ESSB16]		X				
Feldman et al. [FHK^{+}15]				X		X
Graube et al. (2013) [GZUH13]	X				X	
Graube et al. (2016) [GUH16]					X	
Hennig et al. [HVKE16]						X
Imran and Young [IY16]						X
Kovalenko et al. [KSS^{+}14]			X	X		
Lee & Kim [LK07]					X	
Lin & Harding [LH07]				X		
Natarajan et al. [NS14a]					X	
Novak and Sindelar [NS13]	X					X
ONTO-PDM [PDT12, GAP^{+}12]						X
Optique [KJRZ^{+}13, KSÖ$^{+}$14, SHL^{+}14]	X				X	
Sabou et al. [SEKB16]			X			
Softic et al. [SRD^{+}13]			X			
Strube et al. [SRFF11]			X			
VFF [TU14, KTS13, TTU15]	X					X
Wiesner et al. [WMM11a]			X			

Data types. The primary focus of a multi-disciplinary engineering process is on the structured data. Spreadsheets, XML-based data formats, RDF, streaming/sensor data, and relational databases are the most common data types in the MDEE as shown in Table 3.3. Several scenarios also report the use of specific data formats, e.g., AutomationML for data exchange, SysML for plant design, and ECSS-E-TM-10-23A for space engineering.

Number of data sources. Due to our focus on OBDI approaches in research communities, data integration scenarios typically report on the integration of a small number (i.e., less than ten) of data sources.

Size of data. There is a large variety in the size of data, ranging from cases with the least amount of tens of data points [ESSB16] up to those that can handle more than 30 GB of sensor data daily [KSÖ+14].

Data source dynamics. The addition and removal of data sources can be crucial for engineering scenarios. Several engineering scenarios consider this data source dynamics [WMM11a, ASF14], while others do not.

Data access. Most scenarios need access to the integrated data as a whole. Some scenarios, however, report on the requirement to access both local and global (parts of integrated) data for various reasons, e.g., to compare local data from different sources [FHK+15, WMM11a, ASF14] or to enable data change propagation [ESSB16].

3.3.3 Semantic Heterogeneity

Semantic heterogeneity reflects differences between two or more data sources. The heterogeneity in data integration systems varies between individual cases in MDEEs. As an example, the semantic heterogeneity is small in data integration cases where engineers develop most of their local data sources according to a data standard (e.g., AutomationML [SEKB16], CAEX [SRFF11], OPC-UA [GUH16] and ECSS-E-TM-10-23A [HVKE16]).

However, there are cases where local data source structures are created independently without prior agreement or standard as a basis (e.g., hydropower plant UC [KSS+14, ESSB16]). In these cases, we cannot assume any prior agreement among data owners and must rely on mapping definitions of source structures (or between data sources and common data structure, depending on the chosen data integration approach) to enable interoperability. In these cases, the semantic heterogeneity is in general considerably higher.

Mapping complexity reflecting the complexity of relations among involved data sources varies across scenarios. This characteristic is important due to the differences of OBDI variant capabilities to represent mappings.

3.4 Technical Realization of Ontology-Based Data Integration Elements

This section explains technical realization options for OBDI elements from our survey. We focus our investigation to the main OBDI elements shown in Figure 3.2, including: **(1) Ontology Language and Framework**, the ontology languages and subsequent frameworks used to represent knowledge in the OBDI application; **(2) Data Acquisition**, concerning the methods for acquiring data from non-ontology data sources to the OBDI system; **(3) Mapping and Transformation**, focusing on methods and tools for mapping and aligning data from heterogeneous data sources within OBDI application;

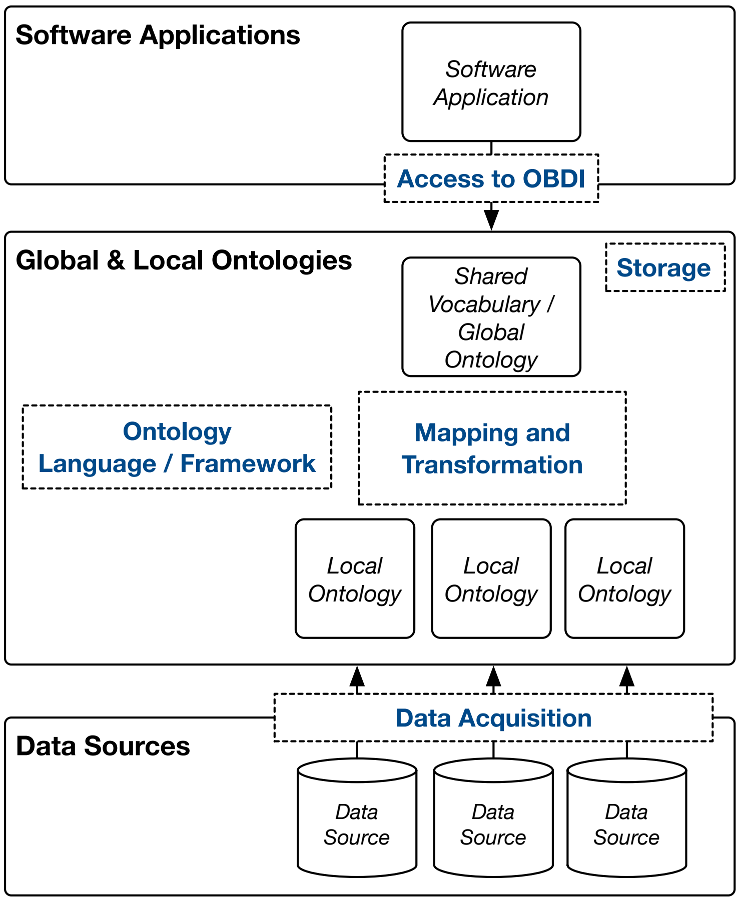


Figure 3.2: OBDI solution elements (blue texts)

and **(4) Storage and OBDI data access**, which deals with the storage solutions of ontology data in the OBDI solution and how software applications access such data.

We report on the results of our survey for each of these elements in Sections 3.4.1 - 3.4.4.

3.4.1 Ontology Language and Framework

In recent years, the Resource Description Framework (RDF) for expressing information about resources [HS13], together with RDF Schema as a data modeling vocabulary [GB14], and Web Ontology Language (OWL) as an ontology language [W3C12] emerged as the de facto standard for representing ontologies on the Semantic Web (SW). Most SW-based OBDI applications use these three standards. Abele et al. [AGZK14] propose an alternative approach on top of the RDF-based Semantic Media Wiki. Several of these approaches use standard and custom RDF vocabularies, e.g., SSN and DUL to represent sensor data [SHL+14], IEC-61499 ontology [DVYP14], SysML and Matlab/Simulink ontologies [FHK$^+$15].

Only a few of the surveyed approaches do not use W3C standard-based ontology languages and frameworks. Wiesner et al. [WMM11a] rely on F-Logic [KL89] to represent all facts, rules, and queries. They argue that even though the combination of OWL and the rule language SWRL [HPsB$^+$04] can in principle provide the same level of expressiveness as F-Logic, it has drawbacks, e.g., the lack of negations. F-Logic could define rules for integration and mapping purposes as well as formulations of expressive queries. Imran and Young [IY16] use similar arguments for their selection of Common Logic-based Knowledge Frame Language (KFL) and emphasize that KFL is more expressive and has more powerful reasoning capabilities compared to OWL. Lee and Kim [LK07] use XML Topic Maps, which were proposed as an alternative to RDF at the time of their research. Because W3C standards are the dominant approach, the following sections will focus on the RDF(S) and OWL.

3.4.2 Data Acquisition

OBDI approaches in the engineering domain typically integrate structured data in various formats. Most approaches in our survey integrate relational databases [TU14, ASF14, GZUH13, KSÖ$^+$14, SHL$^+$14, KTS13], spreadsheets [ESSB16, ASF14], XML [KSS$^+$14, SRFF11, WMM11a, SRD$^+$13, SEKB16], and RDF graph data [KSS$^+$14, FHK$^+$15, DVYP14, LH07, ASF14]. Several OBDI approaches also integrate specific or legacy data formats, e.g., SysML [FHK$^+$15], CAEX [SRFF11], web services and ECAD [BÖF$^+$10], and ECSS-E-TM-10-23A [HVKE16].

Several approaches are possible to integrate non-ontology data into an ontology graph. The Extract, Transform, and Load (ETL) mechanism is one of the most used, where OBDI approaches develop custom applications to convert data (e.g., [TU14, DK10, KTS13]).

The Extract, Load, and Transform (ELT) mechanism represents another method, which may involve automatic conversion to an ontology graph (e.g., [KSS$^+$14, ESSB16, WMM11a, ASF14]). This mechanism first transforms data source instances to an arbitrary ontology graph and then transforms the resulting graph into a target ontology representation. In comparison to ETL, ELT transforms data within a single ontology language.

The Ontology-Based Data Access (OBDA) method allows users to access virtual RDF graphs of non-RDF data source instances, mainly from relational databases (e.g., [KSÖ$^+$14, SHL$^+$14] use Ontop [CCKE$^+$16]). RML Mapping Language [DSC$^+$14] facilitates OBDA for other data sources (e.g., XML, JSON, and CSV). However, we have not found an RML application in approaches within our survey.

Graube et al. [GUH16] propose a method to acquire transient data (e.g., web services that contain sensor data) as part of their OBDI implementation. They adapt the URI dereferencing functionality of SPARQL 1.1 Service Description [Wil13] to retrieve web services data during SPARQL query executions. Due to the preliminary nature and the small amount of data involved, Hennig et al. [HVKE16], Dibowski and Kabitzsch [DK10], and ONTO-PDM [PDT12, GAP+12] used manual data acquisition/transformation of source data to RDF.

3.4.3 Semantic Mapping and Transformation

We observe that most OBDI approaches in our survey rely on either a single or one from the following combinations of methods for mapping definitions: RDF property mapping, Globally Unique Identifier (GUID) matching, a combination of both RDF property mapping and GUID matching, or property value matching.

- RDF property mapping relies on a set of RDF properties to link classes, properties and instances of different ontologies, e.g., owl:sameAs, rdfs:subClassOf, rdfs:subPropertyOf, owl:equivalentClass and custom RDF properties (e.g., [KSS+14, FHK+15, KSÖ+14]).
- URI/GUID matching links instances of ontologies with identical URIs (e.g., [BÖF+10, GUH16, ASF14]). The approach rests on the assumption that individuals will be assigned a unique identifier across different local ontologies in the acquisition process.
- Property value matching is another method used for instances mapping, where two or more objects in different ontologies are considered the same if certain property values of these instances are the same [DVYP14, PDT12, GAP+12].

To define these mappings and create the actual relations, OBDI applications employ RDF to RDF transformation methods and tools, such as SILK [VBGK09] (e.g., [ASF14]), SPARQL construct queries (e.g., [ESSB16]) and arbitrary transformation code based on RDF APIs (e.g., [50]). Within these tools, algorithms for finding links among these ontologies are deployed, e.g., string matching or custom user-defined rules.

An alternative to the transformation methods and tools are reasoners and rule engines. We found a number of them in our survey, e.g., Wiesner et al. [WMM11a] use the OntoBroker [AKL09] rule engine to define rules for mapping, Natarajan et al. [NS14a] use the Hermit reasoner to improve the querying process, and ONTO-PDM [PDT12, GAP+12] use first order logic (FOL) to define instance relations based on property values. Hennig et al. [HVKE16] use Pellet and Strube et al. [SRFF11] use SWRL with Jess to derive implicit knowledge.

3.4.4 Storage and Data Access

In our survey, we identify three RDF-based storage options: RDF triplestore, in-memory store and relational databases. Wiesner et al. use the OntoBroker storage system for their F-Logic based ontologies.

OBDI Approach	OBDI Variant				Language and Framework							Data Acquisition					Mapping			Transformation				Data Storage				Data Access			
	Single-ontology	Multiple-ontology	Hybrid	Global-as-View	RDF	OWL	OWL2	F-Logic	Topic Maps	SMW	Common Logic (CL)	ETL	ELT	OBDA	Transient Data Acquisition	Manual	RDF Property	URI or GUID Matching	Property Value Matching	SILK	SPARQL Construct	Arbitrary Transformation Code	Reasoner & Rule Engine	Triplestore	In-memory or file-based	Relational DB	Others (e.g., ontobroker)	SPARQL Endpoints	Custom APIs	Custom GUIs	Stream Data Engine
Aarnio et al. [ASF14]			X		X	X							X				X	X		X				X				X			
Abele et al. [ALGM13, AGZK14]	X				X					X		X						X		-				X		X		X	X		
Brecher et al. [BÖF+10]	X				X	X						-						X				X		-					X		
Dibowski & Kabitzsch [DK11]			X		X		X					X						X				X		X				X			
Dubinin et al. [DVYP14]				X	X	X						-							X				X		X			-			
Ekaputra et al. [ESSB16]				X	X	X							X					X			X				X				X		
Feldman et al. [FHK+15]		X			X	X						X					X					X		X				X			
Graube et al. (2013) [GZUH13]		X			X	X						X						X				X		X				X			
Graube et al. (2016) [GUH16]	X				X	X									X			X				X		X				X			X
Hennig et al. [HVKE16]	X				X		X									X		X				X	X		X			-			
Imran and Young [IY16]				X							X					X	-					X			X				X		
Kovalenko et al. [KSS+14]		X			X	X							X				X					X		X				X			
Lee & Kim [LK07]	X								X			X					-					X		-					X		
Lin & Harding [LH07]				X	X	X						-					X					X			X			-			
Natarajan et al. [NS14a]	X				X		X					-					-						X	-						X	
Novak and Sindelar [NS13]	X					X						X						X				X			X				X		
ONTO-PDM [PDT12, GAP+12]	X				X		X									X			X				X	X				X			
Optique [KJRZ+13, KSÖ+14, SHL+14]		X			X		X							X			X			-						X		X		X	
Sabou et al. [SEKB16]	X				X	X							X					X				X		X				X			
Softic et al. [SRD+13]	X				X	X						X					-			-				X						X	
Strube et al. [SRFF11]	X				X	X							X					X					X	-					X		
VFF [TU14, KTS13, TTU15]	X				X	X						X					-					X		X					X	X	
Wiesner et al. [WMM11a]			X					X					X					X					X				X		X		

Table 3.4: Technology options for OBDI elements and their adoptions in MDEE *("X" indicates adoption; "-" indicate that no clear information available in the paper)*

- RDF triple stores (e.g., Virtuoso, Jena TDB, StarDog or RDF4J) allow users to store large RDF data as triples (e.g., [DK10, PDT12, GAP+12] - Cf. [MDT+17] for a comparison of selected RDF store solutions in MDEE.
- The in-memory store [LH07, IY16, NS13] is often used for smaller-scale data, e.g., for prototypes or proof-of-concepts.
- The use of relational databases via an OBDA layer are also common [KSÖ+14, SHL+14]. Despite efforts from the SW community, the capabilities of RDF triple-stores are still lacking behind relational databases. Relational databases with an OBDA layer are often used in scenarios that need to cope with large amounts of data.

The three most widely used mechanisms to access OBDI data from software applications are SPARQL endpoints (e.g., [ASF14, GZUH13, SEKB16]), API-based services (e.g., [IY16, NS13, SRFF11]), and custom-build GUIs (e.g., [SRD+13, NS14a, TTU15]). Furthermore, extensions of SPARQL endpoints are being developed to allow access to streaming data [GUH16].

Table 3.4 summarizes technology options used as part of OBDI approaches on papers within our survey.

3.5 Analysis of Ontology-Based Data Integration Variant Adoptions

In this section, we evaluate each OBDI variant (i.e., single-ontology, multiple-ontology, hybrid and GAV OBDI) against a set of MDEE scenario characteristics observed in Section 3.3 (i.e., semantic heterogeneity, data access, mapping complexity, and data source dynamic). Furthermore, we consider the ontology implementation effort as an additional criterion.

3.5.1 Single-ontology OBDI

Single-ontology OBDI is common in MDEE – more than half of the papers surveyed belong to this category.

Semantic heterogeneity. Single-ontology OBDI is convenient when data sources are semantically close (e.g., [TU14, SRD+13, BÖF+10, AGZK14]) or when they can be aligned according to a common data standard (e.g., AutomationML [SEKB16], CAEX [SRFF11], OPC-UA [GUH16] and ECSS-E-TM-10-23A [HVKE16]).

Data access. Software applications built on top of a single-ontology OBDI infrastructure can only access the global ontology, i.e., they cannot access data that are not captured in the global ontology.

Mapping complexity. Because only a single (global) ontology is used, single-ontology OBDI typically does not require any mapping definitions. In some cases, where semi-automatic global ontology acquisition is possible (e.g., [59]), mappings are needed to transform intermediate ontology instances (i.e., the automatically generated local ontologies from data sources) according to the global ontology.

Data source dynamics. Changes to the global ontology are costly, also because they may affect transformation mechanisms from local ontologies. Therefore, the single-OBDI approach is more suitable for scenarios with infrequent data source additions or if addition of a data source does not affect the global ontology.

Ontology implementation effort. Single-ontology OBDI requires only the development of a global ontology, but no additional inter-ontology mappings.

3.5.2 Multiple-ontology OBDI

Semantic heterogeneity. Each data source is described independently using a local ontology, without an implicit assumption that these local ontologies share vocabularies. Therefore, multiple-ontology OBDI is suitable in scenarios with high semantic heterogeneity.

Data access. Each local ontology can be accessed independently, an aggregation of local ontologies can be made accessible using named graphs [KSS+14, FHK+15] or an aggregated ontology [KSS+14, KSÖ+14, SHL+14] can be used. In principle, the aggregated local ontology could also be accessed via SPARQL Federated Queries [SPFW13], although we did not encounter an implementation of it in the survey.

Mapping complexity. Multiple-ontology OBDI requires a set of mappings that define relations among the involved local ontologies. We found that most applications of multiple-ontology OBDI ([KSS+14, KSÖ+14, SHL+14, FHK+15]) use RDF property mappings to represent these relationships. There is only one exception [GZUH13] that uses instance mappings instead.

Data source dynamics. Each addition of a new data source to a multi-ontology OBDI infrastructure requires (i) the definition of new local ontology and (ii) mappings from the new local ontology to other local ontologies. This implies that adding data sources involves considerable effort. Most implementations in our survey involve a fixed number of data sources and a limited number of mappings and do not consider data source dynamics. Graube et al. [GZUH13] hint at the possibility of adding new data sources, but the authors do not explain how their application would address such dynamics.

Ontology implementation effort. The approach requires development of a set of local ontologies and the definition of a set of mappings among them. This is acceptable for scenarios with a limited number of local sources and mappings, which were common in our survey [KSS+14, KSÖ+14, SHL+14, FHK+15]. For more complex cases, however, alternative OBDI approaches are necessary.

3.5.3 Hybrid OBDI

Semantic heterogeneity. A central concept in hybrid OBDI is the availability of a shared vocabulary that facilitates the integration of data sources, not only those that have a similar view of a domain (i.e., low semantic heterogeneity), but also those with a high level of semantic heterogeneity.

Data access. Hybrid OBDI provides two ways to access data: (i) direct access to the (aligned/restructured) local ontologies, and (ii) access to the shared vocabulary, where the system queries each local ontology and merges the results. Aarnio et al. [ASF14] demonstrate and evaluate both access methods, and they report that direct access to local ontologies is faster than access to the shared vocabulary. Wiesner et al. [WMM11a] focus more on accessing the integrated data via shared vocabularies.

Mapping complexity. Hybrid OBDI defines mappings between local and global ontologies using semantic relations. To this end, this approach typically uses a set of RDF properties as reported in [ASF14] (e.g., owl:sameAs and owl:subClassOf). In applications that do not rely on SW technologies (but rather, e.g., F-Logic [WMM11a]), authors typically do not report on how relationships among involved ontologies are established.

Data source dynamics. Hybrid OBDI makes integration of additional data sources easier through the shared vocabulary refinement method. Reports on hybrid OBDI [ASF14, WMM11a] hint at this capability without discussing it in detail or considering dynamics in their application.

Ontology implementation effort. Initial development of a hybrid OBDI system involves considerable effort. Stakeholders need to reach an agreement on the definition of shared vocabularies and need to develop (or redesign, if local ontologies are already available) local ontologies for each data source based on the shared vocabulary. However, these efforts then result in aligned local ontologies without need for additional mappings.

3.5.4 Global-as-View OBDI

Looking back into OBDI categorizations from Wache et al. [WVV$^+$01] (cf. Figure 2.3), we observe in our literature study that there are OBDI applications that are similar (i.e., they make use of a global ontology and several local ontologies), but do not exhibit all the characteristics of hybrid OBDI, such as [ESSB16, DVYP14, LH07]. Specifically, these applications develop local ontologies before the definition of the global ontology (cf. Figure 3.3). Therefore, the local and global ontologies are independent from each other. In this situation, interoperability is achieved either by (1) transformation of local ontologies' instances into the global ontology instances or by (2) query re-writing based on predefined mappings alignments. We refer to this approach as Global-as-View (GAV) OBDI due to its similarity (i.e., it contains a global schema that is independent of local schemas) with the GAV approach from the relational databases [DHI12]. To differentiate this OBDI variant from existing variants, we distinguish the integration process in the equal context with other OBDI variants (cf. Figure 3.3 and 2.3).

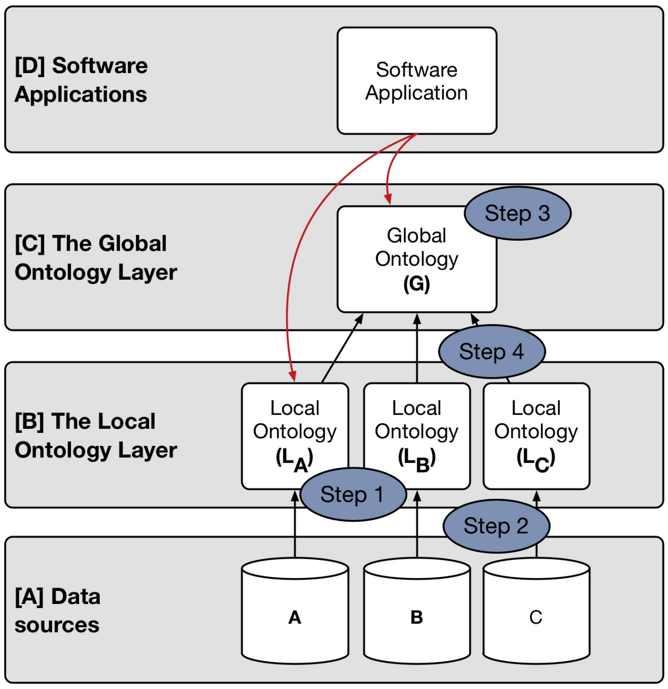


Figure 3.3: The fourth OBDI variant: Global-as-View

GAV OBDI requires the definition of one local ontology per data source, similar to multiple-ontology and hybrid OBDI. In this variant, the integration process consists of four steps (cf. Figure 3.3): (i) Creation of three independent local ontologies LA, LB, and LC (or reuse of existing local ontologies) for data sources A, B, and C respectively. (ii) Transformation of source data in local sources A, B, and C according to local ontologies LA, LB, and LC. (iii) Development of a global ontology G represents a set of common concepts relevant to scenarios, and (iv) Definition of independent mappings between each local repository (i.e., LA, LB, and LC) and the global ontology G to facilitate data transformation from local ontologies to the global ontology.

Several researchers, e.g., Gagnon [Gag07], Modoni et al. [MDT$^+$17], and Moser [Mos09, Mos16] have proposed ideas similar to, or having common points with the GAV OBDI without differentiating GAV OBDI from existing variants, while Juarez et al. report an adoption of GAV OBDI in a related domain, home automation [JRMGC$^+$14]. In this Section and Figure 3.3, we formulate and differentiate GAV from existing OBDI variants shown in Figure 2.3 and explained in Section 2.3. Next, we will analyze the GAV OBDI adoption in MDEEs.

Semantic heterogeneity. Similar to the hybrid OBDI approach, the availability of a “common view” of a global ontology in Global-as-View (GAV) OBDI can address various levels of heterogeneity.

Data access. GAV OBDI provides access on the global and local ontology levels. In line with this capability, MDEE data integration scenarios using GAV OBDI provide access to both local and global ontologies [ESSB16, LH07, DVYP14].

Mapping complexity. Mappings between local and global ontologies are represented by a set of transformation rules or queries. Depending on the scenario, the mappings can be one-way (local-to-global, e.g., [DVYP14, LH07]) or two-ways (local-to-global and global-to-local [ESSB16]), with various levels of complexity.

Data source dynamics. GAV OBDI requires several steps to include an additional data source. First, it is necessary to define or reuse a local ontology for the new data source. Then, transformation rules to the global ontology have to be established. It does not, however, require other local ontologies and mappings to change. Two reports [ESSB16, LH07] highlight this as an advantage of the approach.

Ontology implementation effort. The effort required to establish the ontologies and their mapping is comparable to the effort for hybrid OBDI, albeit with a different use of such mapping (i.e., for transforming instead of linking RDF data instances). SPARQL Construct [ESSB16], eSWRL (an extension of SWRL rule language) [DVYP14], and arbitrary transformation code [LH07] are example languages that are used for this kind of transformation. TopBraid SPIN can also serve as an alternative, however, so far none of the approaches has been used in an MDEE.

3.6 Guidelines for the selection of Ontology-Based Data Integration approach variant in Multi-Disciplinary Engineering Environments

In this section, we discuss the strengths and limitations of OBDI variants with respects to key characteristics of data integration scenarios in MDEEs and conclude with a guideline for the selection of OBDI variant in MDEEs. Table 3.5 summarizes comparison results and highlights the strengths and limitations of OBDI variants in MDEE based on the analysis in Section 3.5.

Wache et al. [WVV$^+$01] consider hybrid OBDI the most effective among the three OBDI variants. We observe, however, that single-ontology OBDI is the most popular OBDI approach in MDEE, presumably due to its simplicity (i.e., it is suitable for scenarios where there is no need to preserve local data structures). If users need to keep local data source structures and compare instances from these sources, other OBDI variants are more suitable.

Single-ontology OBDI. In this OBDI variant, the shared vocabulary of all the data sources that need to be integrated is defined in a single global ontology. Data from various data sources are transformed into instances of the global ontology to achieve the data integration.

The approach is convenient to use when various data sources are semantically close or when data sources can be transformed into a “common language” of the domain (e.g.,

Table 3.5: Characteristics, strengths and limitations of OBDI variants *(Green: strengths, yellow: slight limitations; red: significant limitations)*

	Single-ontology	Multiple-ontology	Hybrid	Global-as-View
Semantic Heterogeneity	best applied for data sources similar view of a domain	support heterogeneous views	support heterogeneous views	support heterogeneous views
Data Access	only allows access on global data	allows access on each (original, if any) local ontology and the aggregated local ontologies.	allows access on each (restructured) local ontology and the global ontology.	allows access on each (original, if any) local ontology and the global ontology
Data Source Dynamics	needs for some adaptation in the global ontology	needs to provide a new local ontology and map the new local ontology to other local ontologies	only needs to provide (or restructure) local ontology based on the shared vocabulary	needs to provide a new local ontology and define mappings to the global ontology
Mapping Complexity	N/A	supports simple mappings (semantic relations)	supports simple mappings (vocabulary refinement)	supports simple and complex mappings (queries and rules)
Ontology Implementation Effort	straightforward	costly	reasonable	rather costly

AutomationML). If such semantic closeness or a "common language" are not available, any addition or removal of data sources may require adaptation of the global ontology to avoid loss of information. Our survey revealed, however, that this approach appears to be sufficient for most MDEE scenarios: more than half of the studied cases adopt this approach. We assume that this popularity is due to the low implementation effort it requires (e.g., only one ontology needs to be built, no ontology mapping/alignment is required).

Multiple-ontology OBDI. Each data source in a multiple-ontology OBDI is described using its local ontology. We cannot assume that these local ontologies share any joint vocabulary. Mappings are established between the local ontologies.

The advantage of this approach is that there is no commitment among local ontologies to shared vocabularies or a global ontology; however, this is also the most significant disadvantage due to the difficulties of relating content in different local ontologies. To overcome this drawback, inter-ontology mappings between local ontologies have to be added. However, these mapping definitions become more difficult when more data sources are being introduced to the system, since local ontologies have to be mapped to each other, which constitutes an exponential problem. The multiple-ontology OBDI is hence more suitable for scenarios where there are a limited number of data sources and therefore a manageable number of inter-ontology mappings is needed. For more complex data integration scenarios, other OBDI variants are more appropriate.

Hybrid OBDI. Hybrid OBDI is characterized by the availability of a shared vocabulary that contains basic terms of a domain that local ontologies should build on via vocabulary/ontology refinement.

The shared vocabulary allows linking and comparing instances from multiple local ontologies, which are relevant for multiple data integration scenarios in multidisciplinary environments. This approach reduces the effort required to define inter-ontology mappings among local ontologies. However, this approach has its drawback: it forces re-development of local ontologies – including their mappings to local data sources – in order to comply with the shared vocabulary. As such, the hybrid OBDI is less suitable for MDEE cases where local ontologies are already established (e.g., in a brownfield OBDI scenario) or they can be automatically generated from data sources.

Global-as-View (GAV) OBDI. The central concept of the GAV approach lies in the global ontology definition, which is similar to the hybrid OBDI. GAV OBDI, however, does not require re-development of existing local ontologies due to inter-ontology transformation definitions between local and global ontologies similar to those used in the multiple-ontology OBDI. In this way, existing local ontologies can be preserved and mapping definitions can be added to allow comparison among local ontologies. Furthermore, data sources can be added with moderate effort (i.e., mappings between the local ontology representing the new data source and the global ontology). Additionally more complex relations beyond ontology representation capabilities are possible (e.g., to represent complex engineering mappings from [GUH16]).

3.6.1 OBDI Recommendation Tree

We developed the OBDI approach recommendation tree (Figure 3.4) based on the OBDI characteristics (cf. Table 3.5) in MDEE scenarios, the OBDI comparison table by Wache et al. [WVV^{+}01], and our analysis result in Section 3.5. The tree summarizes our discussion in section 3.6, which considers several factors (i.e., semantic heterogeneity, resource limitations, mapping complexity, local data access/preservation, and data source dynamics) and can serve as a guideline for practitioners and researchers in selecting the most suitable OBDI approach for the characteristics of their scenario.

There are four main questions in the tree for guiding users in choosing the most suitable OBDI approach in their use cases as shown in Figure 3.4. We will briefly describe these questions in the following:

Question 1: Are local views of data (i.e., local ontologies) needed for the use case? This question helps users to decide whether it is suitable for them to choose a

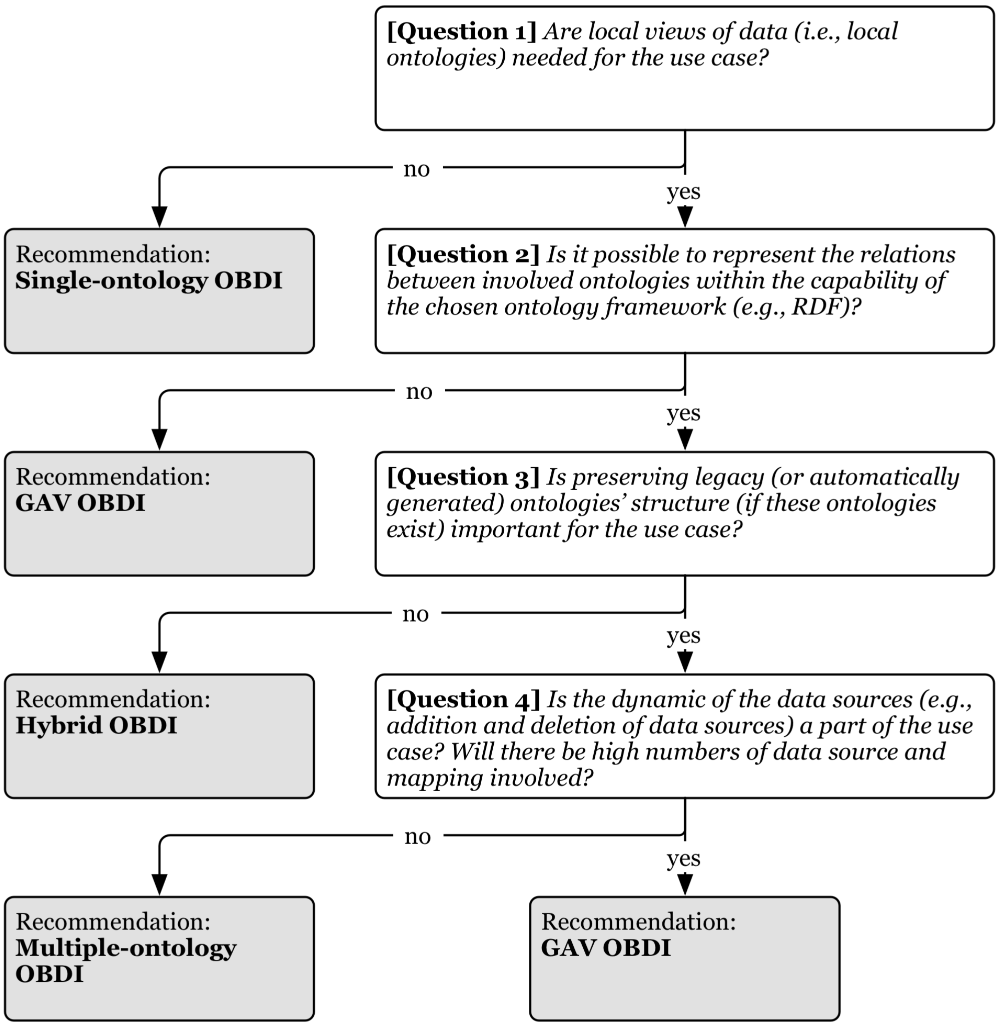

[Question 1] *Are local views of data (i.e., local ontologies) needed for the use case?*
no
yes
Recommendation: **Single-ontology OBDI**
[Question 2] *Is it possible to represent the relations between involved ontologies within the capability of the chosen ontology framework (e.g., RDF)?*
no
yes
Recommendation: **GAV OBDI**
[Question 3] *Is preserving legacy (or automatically generated) ontologies' structure (if these ontologies exist) important for the use case?*
no
yes
Recommendation: **Hybrid OBDI**
[Question 4] *Is the dynamic of the data sources (e.g., addition and deletion of data sources) a part of the use case? Will there be high numbers of data source and mapping involved?*
no
yes
Recommendation: **Multiple-ontology OBDI**
Recommendation: **GAV OBDI**

Figure 3.4: The OBDI approach recommendation tree

single OBDI for their use case. If local views of data are not necessary and a global ontology is sufficient for a use case, a single OBDI approach would be the most suitable for the case.

Several other factors weighting toward choosing a single OBDI in a use case are: (1) the local data sources contain similar views of a domain and therefore could be represented in a single global ontology (i.e., low semantic heterogeneity); (2) the availability of a common/exchange data standard as a global ontology (i.e., lowering the effort to transform data between the global ontology and local data sources considerably); (3) limitation on the resources, e.g., manpower and time to implement the OBDI application, given that the single OBDI requires least effort compared to others (i.e., resource limitations).

Question 2: Is it possible to represent the relations between involved on-

tologies within the capability of the chosen ontology framework (e.g., RDF)? This question focuses on the mapping complexity between among local ontologies and between local and global ontologies. MDEE use cases that contain complex relations or mappings[1] between ontologies may require mapping representation beyond standard SPARQL querying capabilities. In these cases, the GAV OBDI is a suitable approach since it could rely on custom mapping beyond ontology framework, e.g., rule-based and arbitrary programming that support complex mappings.

Question 3: Is preserving legacy (or automatically generated) ontologies' structure (if these ontologies exist) important for the use case? The focus of this question lies on the suitability of the use case with the Hybrid OBDI approach, since the hybrid OBDI requires restructuring local ontologies according to the global ontology. If the answer for Question 3 is "no", Hybrid OBDI is a suitable solution, since the use case does not need access to the original local ontologies.

Question 4: Is the dynamic of the data sources (e.g., addition and deletion of data sources) a part of the use case? Will there be high numbers of data source and mapping involved? This question considers the dynamics of data sources, i.e., how often new data sources are added and removed from the system, which will affect choice between the two possible OBDI approach: multiple and GAV OBDI.

The main disadvantage of a multiple-ontology OBDI solution is the high costs of manually defining mappings between data sources. Therefore, the GAV OBDI is more suitable in cases involving a high number of data sources and mappings between ontologies or concerning dynamic system where data sources are frequently added or removed. In the opposite, we recommend the multiple-ontology OBDI in use cases with a limited amount of data sources and do not emphasis addition and removal of data sources.

3.7 Summary

In this chapter, we report on a review of Ontology-Based Data Integration (OBDI) approaches in Multi-Disciplinary Engineering Environments (MDEEs). Our survey covers both the Semantic Web (SW) and Automation System Engineering (ASE) research communities.

Based on the papers identified in a systematic literature review, we derive a set of data integration characteristics in the MDEEs, propose an extension to the classification of OBDI conceptual approaches, and evaluate the suitability of different OBDI variants against the derived characteristics. Our proposed classification is useful not only in the multi-disciplinary engineering domain, but also in other domains with similar characteristics, e.g., research experiment data [BKE$^+$14, ESSB14].

Furthermore, we identify an additional OBDI variant not considered in prior categorizations, the so-called Global-as-View (GAV) ontology approach. We differentiate the GAV

[1]See [KE16] for a more comprehensive overview about semantic mapping in the engineering domain.

from other OBDI variants and discuss the strengths and limitations of these variants. One of the main advantage of the GAV approach is its ability to preserve existing local ontology structures for analysis purposes.

In addition to the contributions above, we observe technology options for OBDI elements from the selected papers. We find that most of OBDI implementations are using W3C standards of SW technologies (i.e., RDF-based approaches). There are, however, several approaches using alternative technologies, mainly due to their maturity for industrial uptake, e.g., F-Logic as an alternative of RDF [WMM11a]. We also observed feedbacks from the engineering community with regards to their adoption of SW technologies in their domain, e.g., inadequate storage performance [KSS$^+$14], high-learning curve [HVKE16], and the unavailability of rules and transformation standards [ESSB16, DVYP14].

In the following Chapter 4, we will use our analysis result of the OBDI characteristics and recommendations from this chapter to evaluate a set of MDEE scenarios.

CHAPTER 4

Applications of Ontology-Based Data Integration in Engineering

A key output of Chapter 3 is a decision tree model for selecting the right Ontology-Based Data Integration (OBDI) approach for given application contexts (Section 3.6.1). In this chapter, we evaluate these guidelines by demonstrating how we have used them to decide on OBDI approach variants to be used in two concrete settings from Multi-Disciplinary Engineering Environments (MDEEs) described in Sections 4.1 and 4.2. Besides the main goal of evaluating the guidelines, an alternative goal of this chapter is also to reflect on other technologies than Semantic Web (SW) that are used for data integration in MDEEs (as discussed in Section 4.3). Therefore, this chapter, together with Chapter 3 become important elements for answering **RQ1**:*How suitable are the Ontology-Based Data Integration approach variants for the diverse data integration scenarios in Multi-Disciplinary Engineering Environments?*

The first application report (Section 4.1) of AutomationML Analyzer is mainly based on our publication at *The 1st International Workshop on Cyber-Physical Production Systems* [SEKB16]. AutomationML Analyzer is a concrete implementation of a single-ontology OBDI variant and aims to support the analysis of engineering data based on the AutomationML data exchange format and SW technologies. In addition, we will also explain how AutomationML Analyzer is additionally utilized further for simulation generation, which is published at *The 20th International Federation of Automatic Control (IFAC) World Congress 2017* [NEB17].

The second application (Section 4.2), called Ontology-based Cross-disciplinary Defect Detection (OCDD), focuses on defect detection in MDEEs' setting. The OCDD prototype built is an application of the multiple-ontology OBDI variant based on an industrial use case from the industry partners of Christian Doppler Laboratory "*Software Engineering Integration for Flexible Automation Systems*" (CDL-Flex). Section 4.2 explaining OCDD is adapted from our publication at *The 19th International Conference on Knowledge Engineering and Knowledge Management (EKAW)* [KSS+14].

Our reflection on the comparison of Semantic Web Technologies (SWT) and alternative technologies (Section 4.3) is summarized from on our publication at *The 20th IEEE International Conference on Emerging Technologies & Factory Automation (ETFA) 2015* [KWS+15] and the conclusion of *The Semantic Web Technologies for Intelligent Engineering Applications* book [BS16].

4.1 AutomationML Analyzer

In this section, we present AutomationML Analyzer, an application that combines the use of SW technologies and with the AutomationML data exchange standard to enable efficient integration, browsing, querying and analysis of diverse engineering models.

4.1.1 Use Case and Motivation

AutomationML is an open, XML-based data exchange format that was developed to support the exchange of engineering data within the engineering process in production systems. It includes information about system topology, geometry, kinematics, and control behavior and allows inclusion of links to detailed engineering models representing such information, e.g., COLLADA or PLCopen files. For more details about AutomationML please consult [CO15].

Thanks to providing a “common language” for different engineering tools as well as the ability to provide links to detailed engineering information, AutomationML simplifies data exchange across different engineering disciplines and tools in multidisciplinary engineering projects. Nevertheless, there are still issues to address in order to ensure efficient and effective engineering processes. In particular, there are a number of limitations of the engineering processes that solely rely on the use of AutomationML, which are depicted in Figure 4.1 and explained in the following:

1. **Complex data structures with intricate links between engineering disciplines**. Engineering data in multi-disciplinary engineering projects (e.g., such as for designing a power plant) tends to be highly heterogeneous, contains complex structures and has to be managed at a large scale even within one engineering discipline. Also, there are naturally intricate links and correspondences between the data set and data models developed within the different disciplines, since they represent the same object (e.g. a power plant), but from different engineering viewpoints. AutomationML has limited support for defining such links. For example, one can define links between engineering components within AutomationML files. But there are no means to specify cardinality, constraints or the nature of such a relation (e.g., subsumption, sameAs, dependsOn).

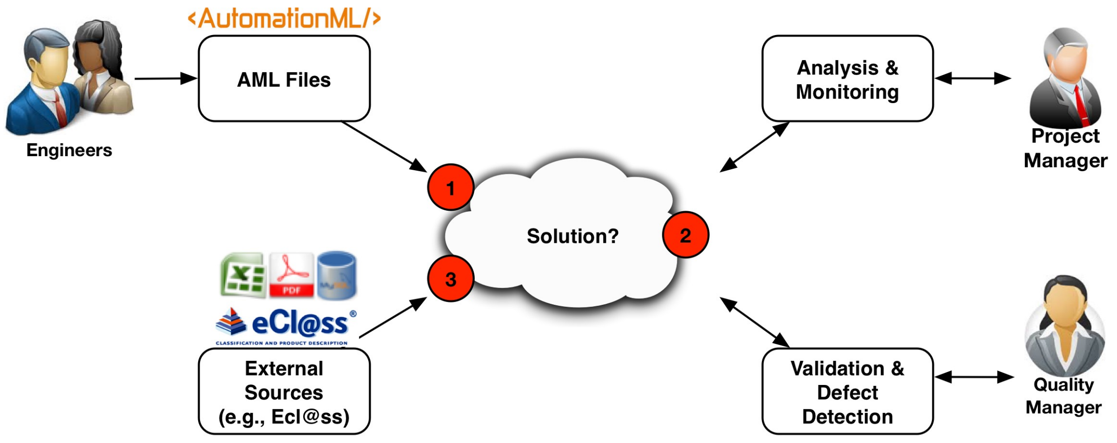


Figure 4.1: Problem Setting: AutomationML Limitations in MDEE settings

2. **Support for cross-disciplinary analytics by end users**. Several stakeholders have a key interest in performing data analytics over the integrated engineering data. For example, project managers often need to monitor and analyze cross-disciplinary engineering data to track the advances of engineers from different disciplines (e.g., milestones reached, the number of fixed issues per project participant). At a more technical level quality managers and domain experts need to validate the integrated engineering data. Defect detection across the integrated engineering models is an example of such a validation process.

3. **Integration of AutomationML data with external data**. The integration of AutomationML data with relevant data from external sources (e.g., with standard product catalogs, such as Ecl@ss) would be highly beneficial. Although the AutomationML consortium currently works on providing some support for integrating Ecl@ss data [GHJ+15], the proposed AutomationML specific constructs are not sufficiently generic for integrating data from other domain-specific standards and catalogs than Ecl@ss.

4.1.2 Selecting an appropriate OBDI Variant

The first question for selecting the OBDI variant according to the recommendation tree in Chapter 3 is to check the semantic heterogeneity of the use case and the resource availability of stakeholders. In this step, the main question to be asked is whether local views of data are necessary for the use case.

In the AutomationML analyzer use case, heterogeneous local views of data is not necessary, as the use case focuses on integrating heterogeneous AutomationML data and enables cross-disciplinary analytics on the integrated data. Additionally, the fact that AutomationML can be seen as a "common language" for integrating engineering data support the decision to use the Single-Ontology OBDI approach, since the heterogeneous tools from

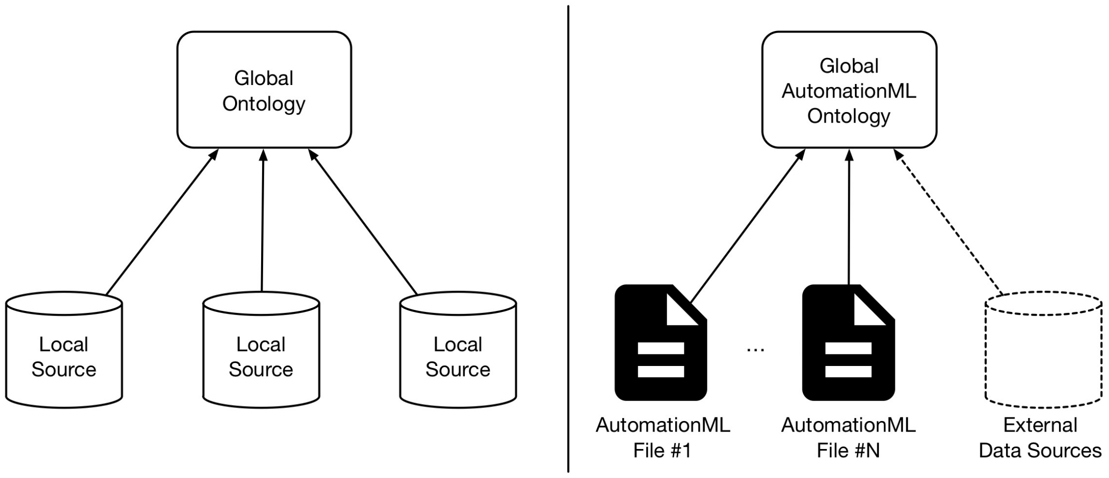


Figure 4.2: The generic Single-Ontology OBDI approach (left) and AutomationML Analyzer approach (right)

different disciplines are assumed to be able to export their data in a "common language". AutomationML import and export plugins are being implemented for various engineering tools, e.g., Enterprise Architect, OPC UA Modeller[1]. Due to these reasons, we decided to use the Single-Ontology OBDI approach for this use case and do not go further in the decision tree steps.

Figure 4.2 depicts the AutomationML Analyzer approach in parallel with the generic Single-Ontology OBDI approach. It shows that AutomationML files and external data sources serve as local data sources for the global AutomationML ontology.

4.1.3 OBDI System Design and Implementation for AutomationML

To address the AutomationML limitations previously described, we designed and implemented the AutomationML Analyzer tool based on the single-ontology OBDI approach. The first part of this section explains the conceptual design of the tool and how it addresses the limitations detailed previously. The later part provides implementation details of the AutomationML Analyzer tool.

Conceptual Design

In the conceptual level, AutomationML Analyzer consists of three major components as shown in Figure 4.3. These major components are marked with number 1-3, and will be explained next.

1. **Knowledge Acquisition**. The knowledge acquisition component reads input files from internal (i.e., AutomationML files) and external sources (e.g., DBPedia

[1]https://www.automationml.org/o.red.c/tools.html

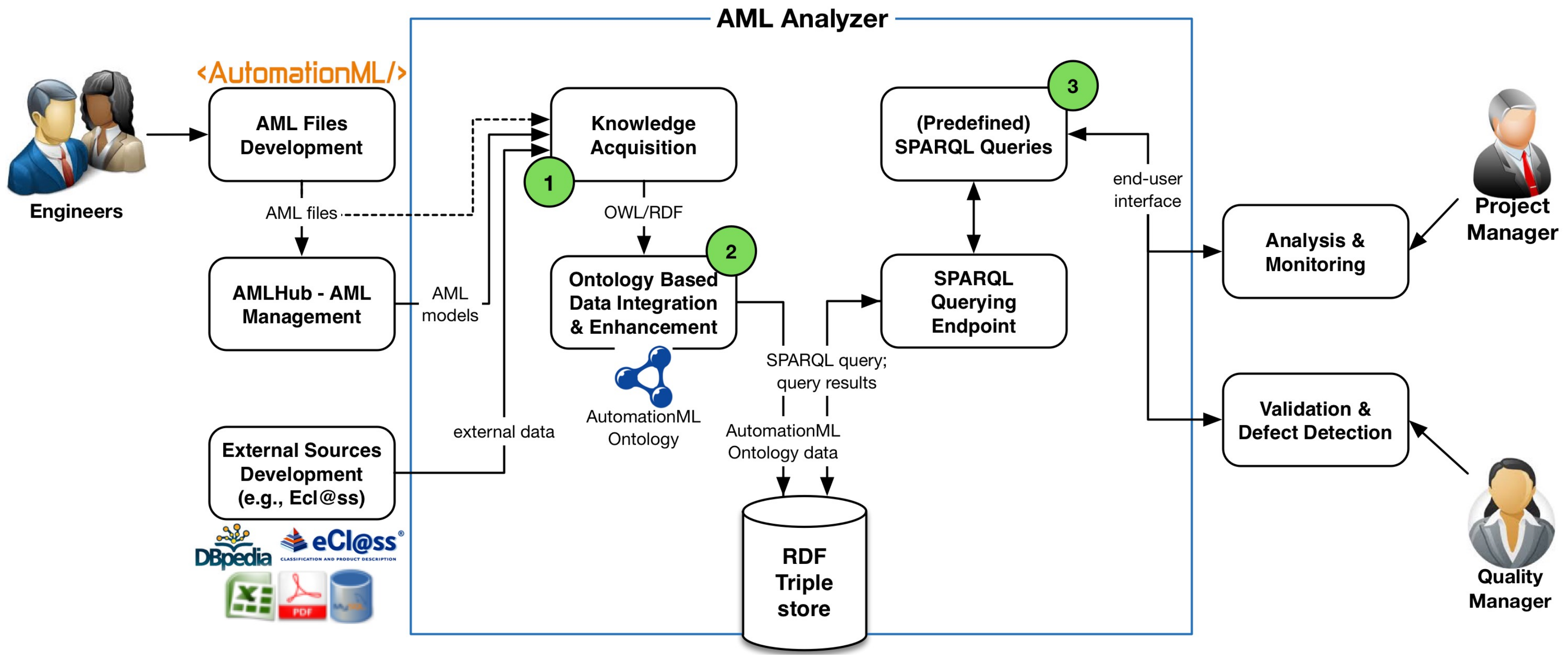


Figure 4.3: AutomationML Analyzer: The Conceptual System Design

and spreadsheet files) and transforms these into an RDF representation. This component performs the first step towards addressing the 1st and 3rd limitation of AutomationML related to the integration of AutomationML data by transforming all data into the global AutomationML ontology as the core of AutomationML Analyzer.

2. **Data Integration & Enhancement**. The functions of this component are twofold. Firstly, it supports data integration between different AutomationML files from heterogeneous data sources. Secondly, it simplifies the complex AutomationML data structure and makes intricate links between components explicit as modeled in the AutomationML ontology (the global ontology of our Single-Ontology OBDI approach). This component mainly addresses the 1st and 3rd limitations of AutomationML and as such contributes towards enabling easier AutomationML data analysis.

3. **Validation Checks Repository**. This component addresses the 2nd limitation of AutomationML and supports the cross-disciplinary analytics and validation of integrated engineering data by the tool users. To that end, it contains a collection of frequent analytical tasks and validation checks formulated as questions over the engineering data integrated using Single-Ontology OBDI approach. Queries are made in terms of the global AutomationML ontology.

Prototype Implementation

In the following, we briefly explain the implementation details of our prototype based on the conceptual system design depicted in Figure 4.3. The AutomationML Analyzer adopts the Single-Ontology OBDI approach and embraces open standards (AutomationML, RDF(S)), while it relies on a mature technical solution, as described in the following:

- **Knowledge Acquisition and Integration**. AutomationML input files are automatically transformed to RDF(S) format and enriched through (1) alignment to an AutomationML ontology and (2) mechanisms to make implicit cross-disciplinary links explicit. Core to the Single-Ontology OBDI approach of the AutomationML Analyzer is the AutomationML ontology that provides a richer semantic representation of AutomationML data. An earlier version of the AutomationML ontology and its creation process are described in [KWS$^+$15]. The current AutomationML ontology provides a semantically rich model into which data from various engineering tools, available in different data formats, is transformed. It has to be noted that external sources integration, such as DBPedia and Ecl@ss is not yet fully implemented and still in progress at the moment.

  We provide options to transform data into the AutomationML ontology from different data formats (e.g., AutomationML and XMI data format). The procedure to transform AutomationML data to ontologies is adapted from the transformation from Ecore[2] (Eclipse Modeling Framework metamodel) into OWL (Web Ontology Language) [KKKK06]. Here, we benefit from the fact that transformations from the CAEX model that underlie AutomationML to Ecore are already in place[3]. Ecore is used as an intermediate format before conversion to OWL and could be replaced by direct translations in the future.

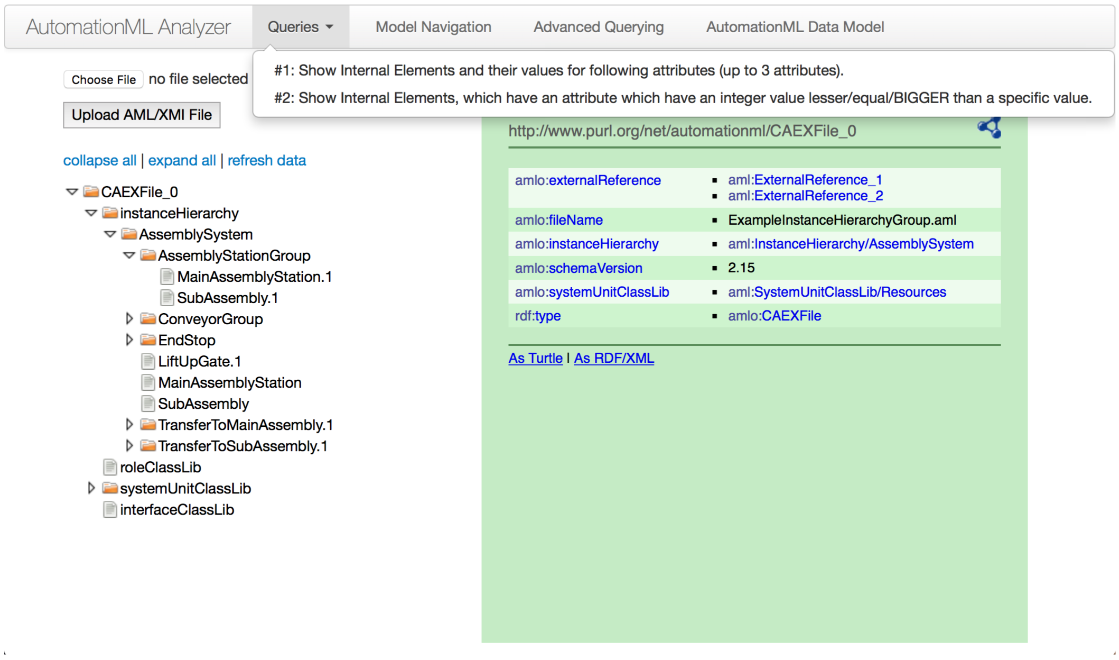
- **Data Storage and Data Access**. We utilize OpenRDF Sesame (now it is renamed into Eclipse RDF4J[4]), an open source, mature and user-friendly RDF triplestore to store the semantic representation of the AutomationML data created in the knowledge acquisition and integration step. Sesame offers an embedded SPARQL endpoint – an interface that enables querying integrated engineering data. We utilize SPARQL queries to perform typical engineering level consistency checks as well as project management level analysis on the global ontology.
- **User Interface**. The AutomationML Analyzer provides a set of project-wide validation checks defined according to the requirements from domain experts and AutomationML experts to be checked against the AutomationML ontology data. These checks are materialized through SPARQL queries. However, the tool's interface only presents users with the textual descriptions of the checks to hide technical complexities (cf. Figure 4.4). The associations between textual descriptions and SPARQL realizations of the checks are stored in a repository that can be further extended with additional checks. The query-based validation interface was created using Apache Wicket[5].

4.1.4 Feasibility Study

For demonstrating the feasibility of the tool, we use a set of AutomationML files obtained from the lab-size simulation model of a production system created at the IAF of the Otto-v.-Guericke University Magdeburg[6], the so-called "Produktionsmodell". This simulation model represents a research prototype of Cyber-Physical Production Systems (CPPS) and comprises ten conveyers, eight turntables and three multipurpose machines. The wiring is done by a modular fieldbus I/O system (WAGO). A coupler and several digital I/O modules connect to Raspberry Pi based controllers. The controllers host PLC code for the aforementioned model components.

We used three AutomationML files each representing a different viewpoint on the production model from the perspective of mechanical, electrical and software engineering respectively. Within the mechanical file, the system physical topology is defined together with the geometry and mechanical properties, e.g. weight, material or load capacity of

[2]https://www.eclipse.org/modeling/emf/
[3]https://github.com/fekaputra/amlMetaModel
[4]http://rdf4j.org/
[5]http://wicket.apache.org/
[6]http://www.iaf-bg.ovgu.de/en/technische_ausstattung_cvs.html



#1: Show Internal Elements and their values for following attributes (up to 3 attributes).

#2: Show Internal Elements, which have an attribute which have an integer value lesser/equal/BIGGER than a specific value.

http://www.purl.org/net/automationml/CAEXFile_0

amlo:externalReference	aml:ExternalReference_1 aml:ExternalReference_2
amlo:fileName	ExampleInstanceHierarchyGroup.aml
amlo:instanceHierarchy	aml:InstanceHierarchy/AssemblySystem
amlo:schemaVersion	2.15
amlo:systemUnitClassLib	aml:SystemUnitClassLib/Resources
rdf:type	amlo:CAEXFile

Figure 4.4: AutomationML Analyzer UI for querying and browsing engineering data

the devices. In addition, the information about the maximum and actual working hours for the model components is defined. Within the electrical file, a wiring for system's components is defined together with electrical properties, e.g. power consumption or switching time. The software engineering file contains information about used PLC interfaces and variables.

The ontology-based representation and integration of AutomationML data using the Single-Ontology OBDI approach enables querying (via SPARQL [HS13]) of data originating from different disciplines, thus, enabling the cross-disciplinary and cross-tool data analysis and consistency checking on heterogeneous CPPS project data. Some example queries are:

- show all composite devices and their sensors;
- show all interfaces for all sensors;
- find all devices that exceeded their maximum working hours.

We describe two sample cross-disciplinary analysis tasks and their implementation as SPARQL queries in the following:

- **Task 1**: The process engineer would like to check that the overall weight and power consumption of the designed plant do not exceed the customer's settings at the location where the plant will be deployed. Listing 4.1 shows the corresponding SPARQL query.

- **Task 2**: This task presents an example of combining information from both design and operation time of a plant. Here an engineer from the plant maintenance team would like to know, which devices have already exceeded their maximum working hours threshold and need, therefore, to be changed as soon as possible. Values for actual working hours are obtained from the sensors streaming data in the plant at operation time and a type of specific machinery chosen during the plant design and development specifies the maximum working hours. Listing 4.2 shows the corresponding SPARQL query.

Listing 4.1: SPARQL query showing the overall weight and power consumption of the production model

```
SELECT
    SUM(xsd:integer(?deviceWeight)) AS ?systemWeight)
    SUM(xsd:integer(?devicePC)) AS ?systemPC)
WHERE {
    ?device a aml:InternalElement .

    ?device aml:attribute ?attribute .
    ?attribute aml:name "Gewicht" .
    ?attribute aml:value ?deviceWeight .

    ?device aml:attribute ?attribute .
    ?attribute aml:name "Leistung".
    ?attribute aml:value ? devicePC .
}
```

Listing 4.2: SPARQL query selecting all devices that already exceeded their maximum working hours and need to be replaced

```
SELECT ?container ?device ?actHours ?maxHours
WHERE {
        ?device a aml:InternalElement .

        ?device aml:attribute ?atActHours .
        ?atActHours aml:name "AktuelleBetriebsstunden" .
        ?atActHours aml:value ? actHours.

        ?device aml:attribute ?atMaxHours .
        ?atMaxHours aml:name "MaxBetriebsstunden" .
        ?atMaxHours aml:value ?maxHours.

        FILTER
            (xsd:integer(?actHours) > xsd:integer(?maxHours))

        MINUS { ?device aml:internalElement ?subPart } .
        OPTIONAL {?container aml:internalElement ?device . }
}
ORDER BY ?container ?device
```

Simulation Generation with AutomationML Analyzer

Besides supporting exploration and analysis of engineering data, AutomationML Analyzer was also adopted for the use case of simulation generation in the MDEE settings, as reported in our paper at IFAC World Congress 2017 [NEB17] and will be explained briefly next.

Simulation models are becoming efficient test-beds that can support decision-making, estimating unmeasured variables, or training human operators. They are necessary for advanced process control, including model predictive control. However, the design phase of the simulation model life-cycle is extremely time-consuming and error-prone, as it is stated in VDI 3633 [Ver13]. Due to the development-time and economic reasons, simulations are not used as widely as they could be.

The AML2SIM method is proposed to ease such challenges in simulation design complexity. It is a systematic method that utilizes AutomationML Analyzer and extended bond graph theory [NŠ14b] for creating simulation models for industrial plants. It generates a simulation model (SIM) semi-automatically based on combining (i) a given real plant topology represented in AutomationML and (ii) available simulation blocks that are included in a simulation block library.

The overall AML2SIM method is summarized in Figure 4.5. The basic idea of the AML2SIM method is that it assembles simulations as a combination of available simulation blocks semi-automatically, based on existing CAD (computer-aided design) description of industrial plants. The steps of generating the simulation in AML2SIM will be briefly explained as follows:

- **Step 1**. The creation of CAD descriptions of industrial plants. In this step, engineers draw the plant descriptions in any CAD tool and serialize the data as AutomationML files.

- **Step 2**. The AutomationML files are transformed into RDF graphs and integrated with the AutomationML Analyzer to allow a queryable and traceable model. We develop a set of REST web services that contain specific SPARQL construct queries to extract the required information to build the simulation.

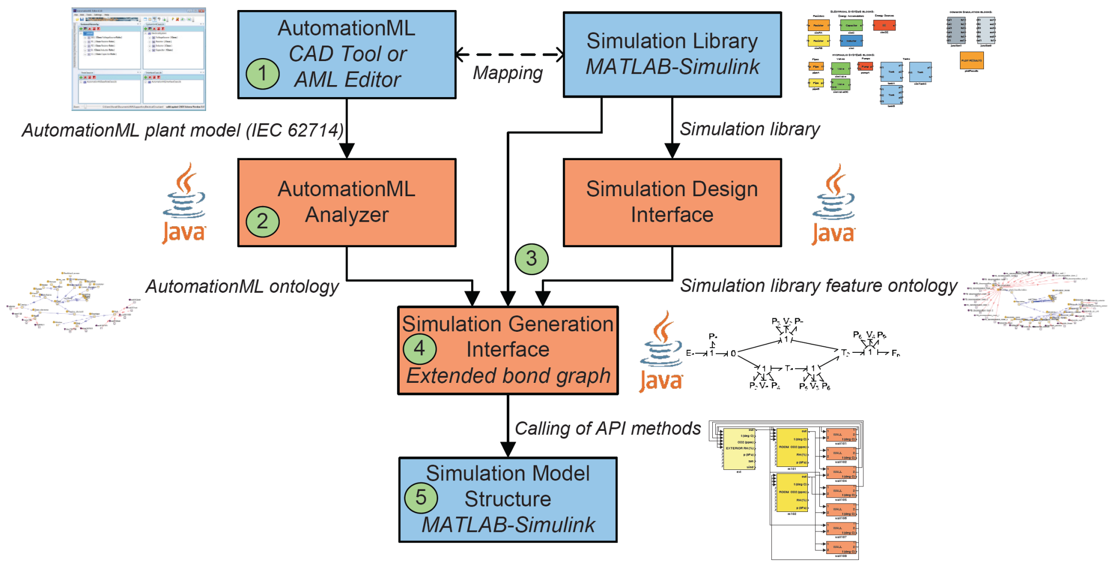

AutomationML CAD Tool or AML Editor
1
Mapping
Simulation Library MATLAB-Simulink
AutomationML plant model (IEC 62714)
Simulation library
AutomationML Analyzer
2
Simulation Design Interface
3
AutomationML ontology
Simulation library feature ontology
Simulation Generation Interface Extended bond graph
4
Calling of API methods
Simulation Model Structure MATLAB-Simulink
5

Figure 4.5: The workflow of the AML2SIM approach that utilizes AutomationML Analyzer

- **Step 3**. The extracted information from AutomationML analyzer is combined with the available simulation blocks, their interfaces, and parameters stored in the simulation libraries.
- **Step 4**. In this step, we utilize the extended bond-graph algorithm to match the available simulation blocks with the plant descriptions.
- **Step 5**. This step represents the generated executable simulation models as a result from the Step 4 above. This simulation model can be then used directly in the simulation tool, such as MATLAB-Simulink.

The adoption of AutomationML Analyzer in AML2SIM method helps generating simulation models by providing means for querying an integrated data model of CAD description of production plants. As a result, the implemented prototype of AML2SIM shows that the time and costs needed to develop a simulation model can be reduced by an estimate of 30-40% compared to the legacy methods [NEB17].

4.1.5 Conclusion

We investigated the use of the OBDI Single-Ontology variant to support the engineering of CPPSs. The aim was to conduct a feasibility check on the Single-Ontology OBDI approach capabilities to integrate and analyze heterogeneous engineering data, which was made available using the AutomationML data exchange standard. We found that the resulting prototype of AutomationML Analyzer is able to address the challenges faced when engineers and domain experts solely relying on AutomationML data exchange format in the following:

- Facilitating integration of AutomationML data and its components from heterogeneous data sources, and
- due to the AutomationML data integration, allowing project level analysis and quality assurance (e.g., defect detection) of engineering data.

In the engineering of systems such as CPPS, the quality assurance of engineering knowledge with advanced checks is highly relevant. Given the mission-critical character of engineering projects, inconsistencies, defects and faults among diverse engineering models should be discovered as early as possible. However, state of the art engineering models tend to capture only limited aspects of cross-domain interdependencies with links between models that are understandable by humans but not by machines [KVH13]. Therefore, the proposed adoption of the Single-Ontology OBDI approach improves the state of the art because it enables formally representing and integrating engineering knowledge. The formal nature of the engineering models and the links between them allow interpretation through reasoning and querying mechanisms, thus, enabling the automation of various quality assurance tasks, as was shown in the case of the AutomationML Analyzer.

We conclude that the Single-Ontology OBDI approach was suitable for AutomationML Analyzer due to its ability to integrate heterogeneous engineering data based on the global AutomationML ontology and facilitating project level analysis and quality assurance of the integrated data. However, the approach has limitations when it comes to integrating heterogeneous external data sources, as this requires an adaptation of the global ontology. If a data source is significantly different from the other data sources, such an adaptation might not be possible. In such cases, other OBDI variants should be investigated based on the decision tree approach we develop in Chapter 3.

4.2 Ontology-Based Cross-Disciplinary Defect Detection

In this section, we describe OCDD, a multiple OBDI application that aims to support defect detection in MDEE use cases. OCDD supports automated defect detection across discipline boundaries. The ontology and SW technologies allow representing the data models of different engineering disciplines and tools and their interrelations in a machine-understandable form, thus, enabling automated processing and analysis across disciplines.

4.2.1 Use Case and Motivation

Multi-Disciplinary Engineering (MDE) projects, e.g., in the automation systems engineering (ASE) domain, typically follow a sequential engineering process. However, in the industry practice, tight project delivery times and project constraints typically require concurrent engineering [Kus93], with engineering teams working in parallel rather than sequentially. This requires adjusting the engineering process and including a set of synchronizations at each project stage to identify early defects and inconsistencies across disciplines from concurrent changes in heterogeneous project data.

Two important types of defects affect engineering processes of MDEE projects:

- intra-disciplinary defects (affect data within one discipline) and
- cross-disciplinary defects (affect data in more than one discipline), e.g., changing a sensor might have an effect on the related software component variable and might lead to defects if not addressed properly.

While defects within a single engineering discipline are usually discoverable by discipline-specific tools, cross-disciplinary defects are detected and fixed mainly during the synchronization phase. Because of the lack of tool support, project engineers usually perform cross-disciplinary data analysis manually, which is time-consuming and error-prone.

Based on the problem description above, it is necessary for MDEE projects to provide efficient and effective mechanisms for defect detection. These mechanisms should be aware of the following requirements: (a) multiple engineering disciplines involvement; (b) cross-disciplinary dependencies in the project data; and c) concurrent engineering. In this section, we focus on addressing these requirements in a specific case study described next.

Case Study Description. The case study is performed in an MDEE project of powerplant engineering. The case study involves data from two engineering domains: Hardware Configuration (HC) data of the Mechanical Engineering domain that contains the physical structure design of devices and their connections, and Control System (CS) data of the Software Engineering domain that corresponds to the control system design. Additionally, we also integrate MDEE Project Configuration (PC) data, which contains information about MDEE projects and their relations with engineering data.

In additional to the data, there are mappings/links between these data that are known by project engineers and domain experts. We extract this information from an interview, and instantiate two of them in our case study: (1) variables that link CS and HC domains, i.e. for each device a set of software variables are defined for its (hardware) inputs and outputs; and (2) artifacts that link the PC domain with HC and CS domains, i.e. an artifact can represent a piece of PLC code on the CS side or a certain hardware device from the HC data, which can be linked to the users responsible for these parts from PC data.

4.2.2 Selecting an appropriate OBDI Variant

In this use case, we are following the questions on the OBDI variant recommendation tree from Chapter 3 as follow:

- **Question 1: Are local views of data (i.e., local ontologies) needed for the use case?** In our case study, the availability of the local views of data is necessary, mainly for analysis purposes. To this end, the use of Single-Ontology OBDI is not suitable, since it does not provide local views of data, which lead us to move to the next step.

- **Question 2: Is it possible to represent the relations between involved ontologies within the capability of the chosen ontology framework (e.g., RDF)?** Our case study is designed as a proof-of-concept prototype in the industrial settings, which will only involve simple (non-complex) mappings between ontologies from different engineering disciplines. Therefore, in this case we move forward to the next question.

- **Question 3: Is preserving legacy (or automatically generated) ontologies' structure (if these ontologies exist) important for the use case?** The data sources used within the use case consists of several XML files generated from various engineering tools. Due to the availability of the ontology generator from XML[7], we can automatically retrieve the local ontologies from the data sources. In the use case, we assume that there are analysis of data both on the local views and the global views, and therefore, we want to avoid re-structuring of the ontologies. Due to these reasons, we are not using Hybrid OBDI and move further to the next question.

- **Question 4: Is the dynamic of the data sources (e.g., addition and deletion of data sources) a part of the use case? Will there be high numbers of data source and mapping involved?** In our use case, it is assumed that there will be no addition of data sources beyond the pre-defined data sources and the number of mapping will be limited to manageable amount. These aspects leads to the conclusion that the multiple-ontology OBDI variant is the most suitable in our case study.

The selection of OBDI variant for OCDD is much dependent on the case study setting. Our case study, in particular, is suitable to be addressed with the multiple-ontology OBDI due to the fixed and small number of data sources and semantic mappings involved. Furthermore, source data are available as a rich structured format (i.e., XML) that allows automatic translation to RDF format as local ontologies. The manual effort of adding inter-ontology mappings between resulted local ontologies is negligible due to the limited number of mapping required.

In comparison with other OBDI variants, the usage of a single-ontology OBDI requires additional efforts of building a global ontology definition and maintaining the consistency between local data sources and the global ontology. The other option of using hybrid-ontology OBDI also requires additional efforts of developing a shared vocabulary and

[7]https://bitbucket.org/fekaputra/xml-tab-jena

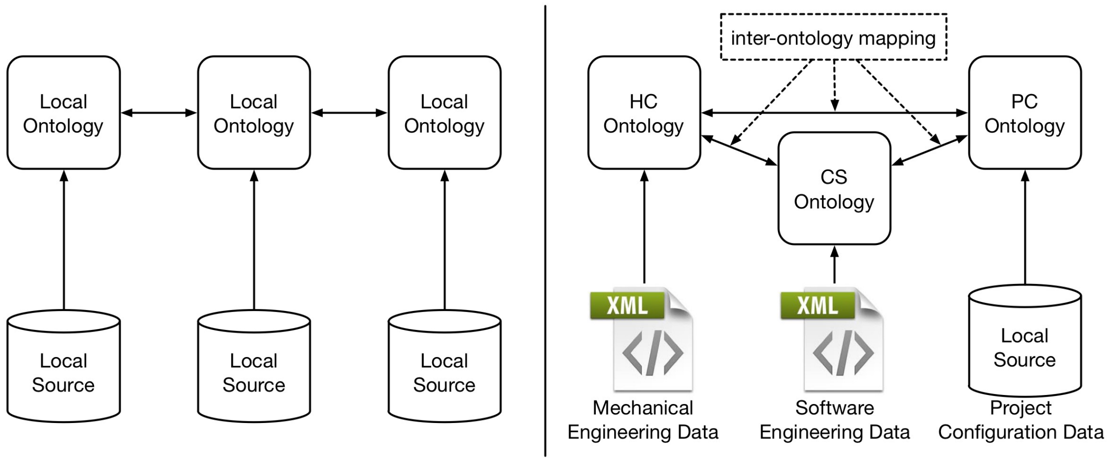


Figure 4.6: The generic multiple-ontology OBDI (left) and its adaptation for OCDD (right)

re-development of local ontologies to make it work. Therefore, we conclude that the use of a multiple-ontology OBDI variant for this particular use case is the best option.

4.2.3 OBDI System Design and Implementation for OCDD

To address the challenge of providing effective and efficient defect detection methods in MDEE projects, we together with colleagues propose OCDD, an Ontology-Based Data Integration (OBDI)-based approach for facilitating automatic defect detection across engineering disciplines.

Conceptual OBDI Design

We follow the principle of the multiple-ontology OBDI approach with a development of local ontologies from data sources as the first step. To this end, we develop three local ontologies of Hardware Configuration (HC), Control System (CS) and Project Configuration (PC). We will briefly explain the content of these ontologies in the following[8]:

- **Hardware Configuration Ontology**. The Hardware Configuration (HC) ontology consists of information about engineering devices, their inputs and outputs and variables that correspond to values on certain hardware inputs/outputs.
- **Control System Ontology**. The Control System (CS) ontology contains engineering artifacts that concern control system software. In the case study, our industry partner uses the IEC61131-3 standard – a standard of International Electrotechnical Commission (IEC) for representing programmable logic controllers.

[8]For more comprehensive information about these ontologies, we refer interested readers to the original paper [KSS+14]

- **Project Configuration Ontology**. The Project Configuration (PC) ontology comprises more general project related information such as: engineering projects; project members and their responsibilities; and the history of changes in the engineering data of these projects.

In addition to the three ontologies, we also define inter-ontology mappings between these ontologies as part of the proposed solution. The overview of our conceptual solution of the OCDD in comparison to the generic multiple-ontology OBDI is shown in Figure 4.6.

Prototype Implementation

The technical implementation overview of the OCDD approach for our case study is shown in Figure 4.7. The HC ontology is created based on the automatic transformation of XML export files of mechanical engineering tools used by our industry partner. For transforming the data, we develop an adaptation of the XMLTab[9] plugin of Protégé[10] called XMLTabJena[11]. In a similar manner, the CS ontology of software engineering is also automatically procured using the XMLTabJena from the XML source files produced by our industry partner (these XML files conform to the IEC61131-3 standard). The PC data source is already in RDF format, and therefore do not need any transformation.

We have previously explained in the Section 4.2.1 that there are two known mappings in our case study (shown in Figure 4.7). The first mapping links the HC and CS ontology. Fortunately, we do not need to explicitly add this mapping to the ontologies since it can be implicitly determined (and queried) by string matching function between CS and HC variable names. In practice, we can also make the mapping explicit by adding a dedicated property to link HC devices and CS software properties, but we decided against it as it was not necessary for the case study.

The second mapping links PC with both HC and CS ontologies. The original local PC ontology stores information about MDEE projects, project members and activities. We add one class and two properties in the PC ontology as mappings between PC, HC and CS ontologies:

1. The `pc:Artifact` class as a superclass of both `hc:Variable` and `cs:Variable` classes.
2. Property `pc:belongsTo` as a relation between `pc:Artifact` and the project that it belongs to.
3. Property `pc:wasPerformedOn` as a relation between an activity with the affected `pc:Artifact`.

[9]http://protegewiki.stanford.edu/wiki/XML_Tab
[10]http://protege.stanford.edu/
[11]https://bitbucket.org/fekaputra/xml-tab-jena

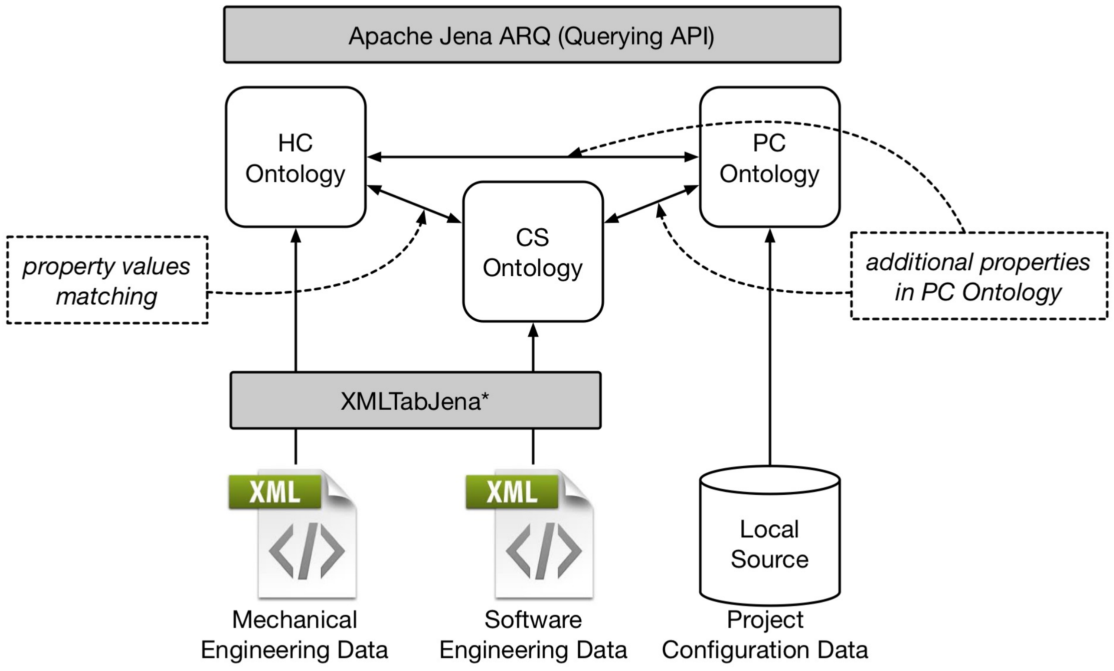

Apache Jena ARQ (Querying API)
HC Ontology
PC Ontology
CS Ontology
property values matching
additional properties in PC Ontology
XMLTabJena*
XML
XML
Local Source
Mechanical Engineering Data
Software Engineering Data
Project Configuration Data

Figure 4.7: Prototype Implementation of OCDD approach based on multiple-ontology OBDI.

We are using the Apache Jena ARQ[12] to allow access and execution of SPARQL queries on top of the integrated data of our OCDD approach.

4.2.4 Feasibility Study

The defined mappings between local ontologies enable formulations and executions of comprehensive SPARQL queries to perform various checks on variable data across domain boundaries, which was not possible to perform automatically before. We describe two cross-disciplinary checks that were implemented for the case study and successfully executed.

Q1: Which global variables on CS side are not declared on HC side? Each global variable declared in the control system software (CS ontology) should be declared as a specific device input or output in the hardware system topology (HC ontology). If a corresponding declaration is missing on the HC side, this might indicate two possible problems: a) either there is a redundant global variable (CS side); b) or a variable declaration is missing in the physical system topology (HC side).

Listing 4.3: Q1: Which variables in HC are not used in CS (as a global variable)?

```
# PREFIXES: hc - hardware configuration ontology
```

[12]https://jena.apache.org/documentation/query/

```
#            cs - control system ontology
#            pc - project configuration ontology

SELECT ?hc_var_id ?hc_var_name
WHERE {
        ?hc_var a hc:Variable .
        ?hc_var hc:hasVarID ?hc_var_id .
        ?hc_var hc:hasItemName ?hc_var_name .

        OPTIONAL
        {
             ?cs_var a cs:Variable .
             ?cs_var cs:hasName ?cs_var_name .
             FILTER (?hc_var_name = ?cs_var_name) .

             # checking whether variable is global in CS
             ?global_vars_container a cs:globalVars .
             ?global_vars_container cs:hasVariableSlot ?artifact .
        }
        FILTER (!bound(?cs_var))
}
ORDER BY ?hc_var_name
```

Q2: Which activities were performed on global variables, which are declared at a certain device, and were not allowed by a project role of the project member, who performed them? This check concerns data from all three domains. Every project member working in a project has project roles, which specifies what kind of activities that is allowed in a project. If there are doubts on the consistency of global variables within the project, one way to discover the cause of defects could be checking whether a project member has performed an activity not allowed by his role. If such cases are found, the engineering artifacts involved (i.e., global variables) are the first candidates to be tested for consistency.

Listing 4.4: Q2: Which activities were performed on global variables that are declared at a certain device and were not allowed by a project role of the project member who performed them?

```
# PREFIXES: hc - hardware configuration ontology
#            cs - control system ontology
#            pc - project configuration ontology

SELECT  ?var_name ?person_name ?role
        ?activity_date ?activity_type
WHERE {
        ?activity a pc:Activity .
        ?activity pc:wasPerformedBy ?person .
        ?activity pc:wasPerformedOn ?artifact .
        ?activity pc:wasPerformedAt ?activity_date .
        ?activity pc:hasActivityType ?activity_type .

        ?person a pc:ProjectMember .
        ?person pc:hasFullName ?person_name .
        ?person pc:hasResponsibility ?resp .
        ?resp pc:hasCorrProject ?project .
        ?resp pc:hasCorrProjectRole ?role .
        ?role pc:hasAllowedActivities ?allowed_activity_type

        ?artifact pc:belongsTo ?project .
        ?artifact a cs:Variable .
        ?artifact cs:hasName ?var_name .

        # checking whether variable is global in CS
        ?global_vars_container a cs:globalVars .
        ?global_vars_container cs:hasVariableSlot ?artifact .

        # global var is set on a specific device in HC ontology
        ?hc_var a hc:Variable .
        ?hc_var hc:hasItemName ?hc_var_name .
        FILTER (?hc_var_name = ?var_name) .
        ?device a hc:Device .

        # device is specified by its id
        # (?param_device_id is given as an input in Jena
        # before query execution)
        ?device hc:hasDeviceID ?device_id
        FILTER (?device_id = ?param_device_id)

        ?device hc:hasVarGrpSlot ?var_group .
        ?var_group hc:hasVarSlot ?hc_var .
        # ——————————————————————————————

        # performed activity was of not allowed type
        FILTER (?allowed_activity_type != ?activity_type)
}
ORDER BY ?person_name ?activity_date
```

4.2.5 Conclusion

Multi-Disciplinary Engineering project participants from different engineering disciplines collaborate to deliver a high-quality end product. Typically, engineering disciplines involved are rather isolated and therefore, it is difficult to efficiently analyze data and perform defect detection activities across the disciplines.

In this section, we introduced the ontology-based cross-disciplinary defect detection (OCDD) approach, which applies the multiple-ontology OBDI as an information integration mechanism, to support automated cross-disciplinary defect detection. In our case study, we have shown that the adoption of the multiple-ontology OBDI approach is able to address the challenges posed by MDEE projects in detecting cross-disciplinary defects in an efficient manner.

The selection of multiple-ontology OBDI variant for OCDD in our case study is based on the fact that only a small number of data sources and semantic mappings involved. Moreover, since the engineering source data is available as a rich structured format, it allows automatic transformation of the data into local ontologies with a minimal effort. In different cases with more dynamic data sources involved, however, other OBDI variant should be considered.

4.3 Notes on Semantic Web and Alternative Technologies

In practice, a number of alternative technologies are used to address challenges in Multi-Disciplinary Engineering Environment (MDEE). We will discuss the comparison between these technologies and Semantic Web Technologies (SWT) next.

4.3.1 General-purpose End-user Approaches

In MDEEs, there is a wide variety of tools and data formats used, and these tool networks often use general-purpose end-user approaches, such as databases, spreadsheets, and scripting, as means to integrate, transform, and reuse data. While these general-purpose end-user approaches are widely used, they suffer, in comparison to the SWT, from low formality and flexibility and, therefore, rely heavily on domain experts to apply and maintain the code and to interpret the results [FBW+13, WB12].

Therefore, general-purpose end-user approaches fall short in addressing the needs identified in Section 2.1.1. However, it is important that we conduct a careful analysis of general-purpose end-user approaches in use before introducing SWT into an engineering environment. Such an analysis is a prerequisite for understanding the existing expertise and for minimizing the risk of failing to provide the benefits expected from applying SWT.

4.3.2 Model-Driven Engineering

Model-Driven Engineering and Semantic Web are different approaches to creating intelligent application in MDEEs that provide several similar capabilities, but also capabilities that differ in important ways. We identify the following similarities between the two technologies:

- **Making knowledge explicit with conceptual modelling**. In Model-Driven Engineering, this is achieved with Ecore[13] metamodels, models, and transformations [WLRW15]; in Semantic Web with ontologies, ontology instances, and reasoning.
- **Direct interaction with knowledge bases**. Both Model-Driven Engineering and Semantic Web communities provide user-friendly generic tools for creating, changing, and populating knowledge bases: in Model-Driven Engineering with Eclipse-based tools and plug-ins [BCW12]); in Semantic Web based on an open development environment for semantic web applications [Knu04].
- **Data integration**. Both Model-Driven Engineering and Semantic Web communities provide mechanisms for creating and maintaining mappings to integrate heterogeneous data sources, in particular, links between schemas and between instances [Obe14].

To provide a better insight into the comparison between the two approaches, we will discuss next our experience in working with both approaches to model engineering data based on AutomationML. This comparison is based on our earlier publication [KWS$^+$15].

Modeling of AutomationML using Semantic Web Technologies vs Model Driven Engineering

We analyse the differences of Model-Driven Engineering and Semantic Web approaches for modelling AutomationML and discuss the potential benefits and limitations of the adopted model creation process and resulting models. The discussion is built over the following aspects: a) model creation process; b) resulting model (metamodel and ontology); and c) potential usages of the resulting model for improving engineering processes in production systems engineering (PSE).

Process. In terms of the effort needed to build a model, creating a model with Model-Driven Engineering approach can be done fast and easy, if the semi-structured representation (e.g., XML schema definition (XSD) in case of AutomationML) is available as a starting point. The conversion of the XSD into an Ecore metamodel is straight-forward and requires no involvement of domain experts, thus, avoiding many iterations and reducing the time needed to have a working model. As the resulting metamodel is a one-to-one match to the original XSD structure, data transformation from the original

[13]http://eclipse.org/modeling/emf/

AutomationML format is therefore straight-forward and easy to perform. However, as soon as changes are made in the metamodel, e.g., for optimization purposes, also the data transformations have to be adapted.

In contrast, when modelling knowledge with Semantic Web technologies and according to a top-down approach, designing an ontology was an iterative process with domain experts in the loop. This leads to a relatively long time for model creation, as the knowledge engineer has to get a detailed understanding of the modelled domain together with expected application to design a suitable model. Additionally, (at least) several verification iterations with domain experts are required during the model creation. Also, defining a complete data transformation from AutomationML format into ontology is more complex, as the resulting model structure differs from the original AutomationML XSD (both on syntactic and semantic level). However, the advantage is obtaining a light-weight, concise and application-tailored model as a result.

Resulting model. The obtained resulting models (metamodel and ontology) differ in many aspects for two major reasons: a) the data representation language (Ecore and OWL); and b) the provided reasoning capabilities.

Ecore enables specifying dynamic behaviour inside the model for instance to define a model simulator for AutomationML, which is not possible in OWL. The Object Constraint Language (OCL) [14] complements Ecore and enables specifying query operations, derived values, pre- and post-conditions. Another advantage is the straight-forward mapping from Ecore into Java objects, which makes the programmatic access to the model easier and supported out-of-the-box. A key limitation, however, is that the Ecore model must stay stable w.r.t. schema, i.e., once defined and populated with data, it is challenging to modify the metamodel, e.g., changing a class or a property, and would lead to co-evolution problems for the existing models. Because of this limitation, it was decided to store all AutomationML data on the instance level, e.g., all Role and SystemUnit classes are stored as instances as well as the Internal Elements from the Instance Hierarchy. This makes the model more complex and may make the OCL rules formulation (e.g., for consistency checking) potentially more challenging. Furthermore, there is no out-of-the-box semantics defined for the roles and classes, e.g., the inheritance between roles and classes is just a simple data link which is not considering inherited elements in query tasks.

Speaking of OWL, the language is very flexible w.r.t. class descriptions. OWL supports various ways to define classes: by a class identifier, an exhaustive enumeration of individuals, property restrictions, or by reusing already existing class descriptions (i.e., an intersection of class descriptions, a union of class descriptions, or the complement of a class description). Classes can also constitute nested hierarchies, which makes explicit type definition (e.g., for consistency checking) not mandatory. Type inference can be performed based on class descriptions, thus enabling dynamic classification of objects into class. Also, unlike Ecore, OWL is very flexible w.r.t. modifications on schema level, i.e. it is relatively easy adding new classes or properties, if there is a need to extend an

[14]http://www.omg.org/spec/OCL/

existing ontology. Another important feature is that OWL allows definition of transitive properties, which simplifies query structures for some cases. This is not explicitly possible with Ecore and OCL and would require some workaround with derived features.

Usage. The resulting AutomationML ontology is well suited for applications such as data analysis and consistency checking, both because it was designed with those in mind and because of the features of the OWL language, which make ontologies particularly useful for performing those tasks. For example, reasoners are available to process ontologies for consistency checking, concept satisfiability, instance classification and concept classification. Some of the reasoners also provide axiom explanation in case inconsistencies/violations have been found, facilitating understanding, localizing and fixing activities for engineers.

The resulting AutomationML metamodels allow applying model transformations languages to perform in-place transformations on AutomationML data, such as providing improvements to a model as well as out-place transformations such as transforming AutomationML to other systems modeling languages or for mapping AutomationML to formal languages, which facilitate model analysis. Furthermore, highly task specific languages such as model validation, model comparison, model merging languages, are available as demonstrated before which proved already useful in several engineering scenarios.

The important difference for applying AutomationML Ecore metamodels and AutomationML ontology is that semantics in MDE approach adopts Closed World Assumption, while OWL adopts Open World Assumption by default. This might lead to different results while performing querying and reasoning on model and corresponding data. However, it is also possible to force OWL reasoning and querying under the closed world assumption. Speaking of limitations, applying both models for applications in industrial and factory automation requires sufficient understanding from developers and engineers. Having bridges between the modelware and ontoware, the benefits of both worlds may be combined.

Conclusion

Semantic Web approaches have some advantages over Model-Driven Engineering regarding agile schema development, i.e., schema evolution at runtime, reasoning-based checks [KVH13], knowledge reuse. Semantic Web approaches have strong advantages over Model-Driven Engineering in the Semantic Web home grounds of linked data, e.g., with the unique resource identifier (URI) identification, and linking URIs with the sameAs mechanism, and of browsing and exploring distributed data sets, e.g., engineering models and external data sources [GM07]. [Obe14] discusses why and when to apply SWT in enterprise systems, which are in some ways similar to engineering project support systems, and characterizes the state of Semantic Web usage as considerable academic interest and early industrial products.

Model-Driven Engineering has advantages over Semantic Web with a strong open source community in business and industry, and a skill set that is better compatible with existing expertise in typical software engineering projects.

4.4 Summary

In this chapter, we report on the applications of Ontology-Based Data Integration (OBDI) in two different use case scenarios with different motivations and goals: (1) The AutomationML Analyzer that focus on supporting integrated engineering data analysis based on AutomationML data standard, and (2) The Ontology-based Cross-disciplinary Defect Detection (OCDD) tool, which aims to provide a defect-detection mechanism for in Multi-Disciplinary Engineering Environment (MDEE) settings.

This chapter shows that differences in use case characteristics can have a big impact on the choice of OBDI variant for the application. To this end, the decision tree presented in Chapter 3 shown to be very helpul in deciding the most suitable OBDI variant for the use cases: the single-ontology OBDI for AutomationML Analyzer use case and the multiple-ontology OBDI for OCDD use case.

In both cases, we evaluate the OBDI variant decision with the development of a working prototype. Afterwards, we populate the prototype with real-world data from our industry (for OCDD) and research partners (for AutomationML Analyzer), and successfully conducted feasibility studies on both prototypes.

In addition to the two applications, we discuss the comparison between Semantic Web (SW) and alternative technologies for MDEE use cases. Toward this end, we provides both generic and a more in-depth comparison between technologies, showing advantages and drawbacks from these approaches.

CHAPTER 5

Knowledge Change Management in Multi-Disciplinary Engineering Environments

This chapter builds on the foundation of Ontology-Based Data Integration (OBDI) provided in the Chapter 3 and 4 to develop a solution approach for Multi-Disciplinary Engineering Environments (MDEEs) to answer **(RQ2)**: *How to provide sufficient support for Knowledge Change Management in Multi-Disciplinary Engineering Environments beyond the Ontology-Based Data Integration approach?*.

Parts of this chapter is published in several publication venues. We first propose the definition of a generic framework for Knowledge Change Management (KCM) in MDEE as an extension of OBDI [Eka16] based on a set of requirements derived from MDEEs [ESSB15b, Eka15]. Further, we develop an instance of the framework using Semantic Web (SW) technologies based on a use case of hydro-power plant engineering from an industry partner [ESSB16] to conduct a feasibility evaluation.

5.1 Introduction

Knowledge is changing rapidly within the engineering process of Cyber-Physical Production Systems, which is typically conducted within a MDEE. Such rapid changes lead to the need for management and analysis of knowledge changes in order to preserve knowledge consistency. Furthermore, KCM in MDEEs is a challenging task since it involves heterogeneous, versioned, and linked data in a mission-critical fashion, where failure to provide correct data could be costly.

Figure 5.1 depicts a problem setting of KCM in an MDEE, which could be seen as an extended version of Figure 1.1 on the generic problem setting of an MDEE. Figure 5.1

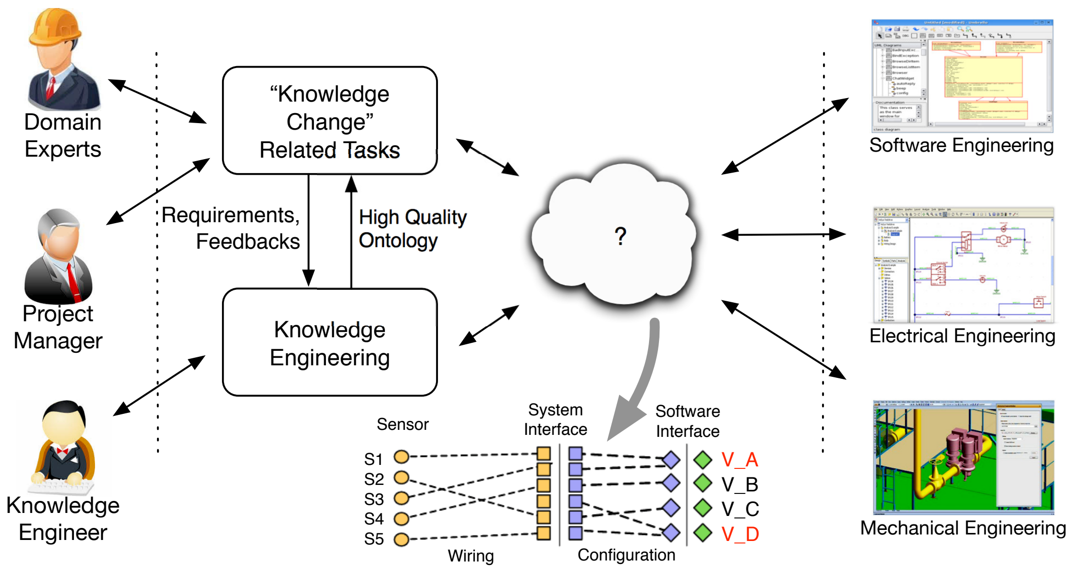


Figure 5.1: The Problem Setting of KCM in an MDEE

shows that domain specific engineers (shown on the right hand side) use their own tools, data models, and data to create models that represent parts of the final system. Due to this, the environment is highly heterogeneous, as it involves a wide range of data models, processes, and tools that were originally not designed to cooperate seamlessly. Despite this situation, as shown on the left hand side of Figure 5.1, there are engineers and project managers that need to perform tasks that require access to project level data as opposed to domain specific data alone. In response to this need, knowledge engineers aim to integrate models and data from different engineering domains based on the requirements and feedback of engineers and project managers. We addressed this need in Chapter 3 and 4.

Additionally to the characteristics described above, the process of designing complex mechatronic objects, such as Cyber Physical Production Systems, requires iterations and redesign phases, which lead to continuous changes of the data and knowledge within the MDEEs. To deal with these changes, industrial partners need to keep data versions, move backwards to previous versions, and query different versions of large data (schema and instances) from heterogeneous local data sources. Furthermore, the effective and considerate propagation of changes is essential to ensure a consistent view of the project, to minimize defects and risks, as well as facilitate acceptance of new solutions by domain experts. To achieve this, changes originating from one engineering discipline need to be communicated and coordinated with participants of other disciplines, where those changes are relevant. Ideally, this communication should focus on high-level changes (e.g., defined in terms of domain concepts such as "Motor X updated to new version") as opposed to low-level individual changes (i.e., change operations on versioned files) to ease the data analysis process. To cater for all these needs, a process and tool support should be available for KCM within MDEEs.

Technology solutions for knowledge change management in general, and to some extent for multi-disciplinary engineering in particular, have been proposed by research fields as diverse as Database Systems, Model Based Engineering and the SW. Although the general strengths and weaknesses of these solution approaches are somewhat known, a precise comparison of how well they could support KCM in the MDEEs is hampered by two major factors. Firstly, there is a lack of understanding of requirements for KCM in MDEE, i.e., a characterization of the problem that needs to be solved. Secondly, a baseline setting that would allow an objective comparison of these technologies is also missing.

To overcome these shortcomings, in this chapter, issues related to KCM in MDEEs are investigated. Specifically, we aim to provide a KCM solution that built on the OBDI approach, which address the second research question of our thesis (**RQ2**): *"How to provide sufficient support for Knowledge Change Management in Multi-Disciplinary Engineering Environments beyond the Ontology-Based Data Integration approach?"*.

To this end, this chapter makes two important contributions. Firstly, it provides a characterization of KCM by means of key requirements that should be fulfilled (Section 5.2). These requirements were derived from concrete industry specific projects where the author investigated the need for KCM. Secondly, this chapter analyzes alternative approaches from relational database and Model-Driven Engineering research communities (Section 5.3) and related work from SW research communities in relation with the elicited requirements (Sections 5.4).

Based on this analysis, the chapter provides a generic reference framework for solving KCM (Section 5.5) that builds on top of the OBDI approach discussed in Chapter 3 and 4. This reference framework is suitable to play the role of a baseline for comparing the strengths and weaknesses of implementations relying on either SW or other relevant technologies, such as database or model-driven engineering. To evaluate our framework, a prototype implementation based on SW technologies and its feasibility study are presented in Section 5.6 and Section 5.7 respectively. We summarize the chapter in Section 5.8.

5.2 Motivating Use Case

For illustration purposes, we provide an excerpt of the data (usually called a "signal list") exchanged between the engineers participating the engineering process of a hydro power plant (Table 5.1).

A signal list is typically serialized and used by engineers as spreadsheet files. The header of Table 5.1 represents the data schema used within the signal list, while its body represents data instances. The combination of the first four columns (ANR, L1, L2, and SIG)

Table 5.1: Excerpt of the engineering data of a power plant engineering process.

ANR	L1	L2	SIG	S3 DIFF MAX	S3 GRADIENT MAX	RESISTANCE VALUE
0	BAA30	GS100	XB01	0,025	0,8	500
0	BAA30	GS191	YB01	0,025	0,8	500
0	BAA30	GS191	YB02	0,025	0,8	500
0	BFB10	GS100	XB01	0,025	0,8	500
0	BFB10	GS100	XM01	0,025	0,8	500
0	BFB10	GS100	YB01	0,025	0,8	500

identifies the engineering signals/objects, while the later three (DIFF MAX, GRADIENT MAX, and RESISTANCE VALUE) represent signal/object properties.

In such MDEEs, the project models and the project data change over time due to:

- **Changes in the represented domains**, such as the introduction/removal of domain concepts (e.g., the removal of RESISTANCE VALUE column from Table 5.1 since it is not relevant anymore in the domain) or granularity changes (e.g., adding more detailed information than at signal level);
- **Changes in the underlying data sources**, such as when new data elements become available and old data elements become obsolete (e.g., a new spreadsheet file is produced to replace the old file without changing the data schema); or
- **Changes in the intended use of the models and data**, such as by changing requirements of the currently supported tools or the design of new tools (e.g., a new data schema is introduced and it has to be mapped into the old schema in Table 5.1).

To address these changes, an integrated versioning of the MDEE data needs to be prepared for facilitating this evolution and the consequent data transformations and propagation, according to the evolved model.

5.2.1 Requirement Analysis

Dealing with the types of possible changes described above requires both activities for managing and analyzing changes. In terms of change management, it is important to record data versions and to be able to move backwards to previous versions, as well as to query different versions of integrated data (schema and instances) originating from heterogeneous local data sources.

Furthermore, on a more analytic related level, it is essential to enable the effective and considerate propagation of changes across data from different disciplines. This will ensure a consistent view of the project and it will minimize defects and risks. To achieve this, changes originating from one discipline need to be communicated and coordinated with participants of other disciplines, where those changes are relevant, especially in cases when data from different engineering disciplines is closely interlinked. The KCM approach should also provide high-level change definitions instead of low-level ones to ease the analysis process of data.

Based on our involvement and experiences in several industrial engineering settings where KCM was required, we have identified a set of requirements and characteristics of KCM in MDEEs as follows:

1. **Closely interlinked knowledge**. In the engineering process within an MDEE, engineering models and data created by different engineering disciplines reflect diverse views on the same system and are therefore naturally interlinked (e.g., an electrical signal activates a mechanical component as a result of executing a certain software routine). Therefore, knowledge changes within one discipline may require changes in other disciplines too due to this relation between data in different disciplines. For example, a signal change in electrical engineering area will require the adaptation of the corresponding machinery part (mechanical engineering) or reprogramming of the relevant software components (software engineering).

2. **Support for change management of large amounts of data**. Engineering projects typically deal with large amounts of data required to describe any complex system. For example, an average size power plant design data contains data about hundreds of thousands to tens of millions of signals. This already large data size is further multiplied due to many iteration processes during the design time of the system, which should all be versioned and stored [MSWB14].

3. **Changes in schema and instances**. In MDEEs, both data models (i.e., schema) and actual data (instances) is likely to change. Indeed, the heterogeneity of data sources within MDEEs and the environment's dynamism mean that additional tools could be added anytime, which may imply changes in the data models of all engineering disciplines involved. At the same time, data instances (e.g., signals with changed characteristics, added or deleted signals) within MDEEs will change even more frequently due to revisions and engineering process iterations.

4. **Change validation support**. Given the mission critical nature of projects in MDEEs, domain experts and engineers do not want to fully rely on automatic change validation mechanisms (e.g., to decide whether changes initiated by one discipline will break the overall consistency of the project wide data). Therefore, instead of fully automated change validation, the involvement of domain experts in the validation workflow is important for making critical decisions about changes.

5. **High-level change definition and detection**. Typical tools currently used in individual engineering disciplines are able to produce report data that consists of signal lists that represent those parts of a CPPS, which these specific tools handle [VHLFR14]. The differences between two versions of signal lists represent changes between them. However, it is challenging for a project manager to grasp the

meaning of such low-level changes in data, i.e., signal changes. Instead, they would highly benefit from changes to data being presented in a more meaningful manner as high-level changes. Such presentation could be achieved in terms of domain level common concepts, e.g., relocation of specific engine to different machine rack.

6. **Support for data evolution and versioning**. A KCMs approach should be able to address both data evolution and data versioning. Data evolution focuses on how to modify the data in response to the changes in the surrounding. Data versioning covers the management of different versions of data schema caused by the data evolution [NK04, Rod95].

5.3 Alternative Technologies

This section introduces brief summaries of approaches related to the KCM in the engineering domain from the Database systems (Section 5.3.1) and Model-Driven Engineering research communities (Section 5.3.2).

5.3.1 Database Schema Evolution and Versioning

One of the earliest works concerning knowledge change management is reported in the field of database systems. Roddick summarized the issues of schema evolution and versioning in database systems [Rod95]. He explains that change management is closely related to data integration, claiming that both areas are the flavors of a more generic problem: using multiple heterogeneous schemata for various tasks. To solve the issues suggested by Roddick, there were several proposed conceptual approaches. One of them is the Hypergraph Data Model (HDM), which targets schema evolution for heterogeneous database architectures [MP02]. The HDM schema consists of Nodes, Edges, and Constraints. Nodes and Edges in the schema define a labeled (Nodes require unique names), directed (the Edges may link sequences of Nodes or Edges), and nested (Edges could link an unlimited number of other Nodes and Edges) hyper graph, while Constraints define a set of Boolean valued queries over the HDM schema.

In the field of MDEEs, there are limited concrete solutions that are utilizing relational databases as the basis for managing and analyzing changes in engineering data from multiple engineering datasets. One exception is the Engineering Database (EDB), a solution based on relational databases that was introduced as an attempt to provide versioning management of engineering data using database technology [MWZ^{+}10]. The Engineering Database is a concrete implementation of Engineering Knowledge Base (EKB) [Mos16]. The EDB stores engineering data as a flat database table, consisting of objects, properties, values and important metadata information such as change commit information and provenance from the original data sources. The approach is capable of handling closely linked knowledge from different engineering disciplines.

To conclude, the maturity of database approaches in general provides a solid basis for handling change management of a large number of data. Additional KCM related solutions from database systems, not covered in this chapter, are also worth further investigations.

5.3.2 Model-Driven Engineering Co-Evolution

A lot of attention has been given to the comparison and versioning of software models in the Model-Driven Engineering research community, with more than 450 papers written in this area[1], covering various topics such as change modeling and definition of evolution frameworks. Similarly to other research areas, the domain of Engineering is not the main application area of these works. An exception is the work of Göring and Fay, which proposes a meta-model language for modeling temporal changes of automation systems together with their physical and functional structures [GF12]. Their work extends the IEC 81346 standard [IEC09], which already includes product, function and location aspects. In the approach, however, they do not explain how to map their meta-model to other meta-models that are potentially used in the same system and therefore it is not clear how data integration is achieved.

Another line of Model-Driven Engineering work focuses on change propagations of model variants of a single meta-model, to ensure the consistency of changes as well as the adoption of relevant changes in different model variants [KKT14]. Recently, the authors of [BBM$^+$15] provide means of a prototype-based model co-evolution, showing the capability of providing various levels of validation configuration to be applied in a top-down coevolution approach.

Meyers and Vangheluwe propose one of the most recent frameworks for evolution of model and modeling languages, which claim that any possible model co-evolution could be derived as a composite of the basic co-evolution schema shown in Figure 5.2 [MV11]. It consists of a model m that conforms to meta-model domain (MM$_D$). Model m needs to be transformed into T(m) via transformation T, which again will conform to image meta-model (MM$_i$). This co-evolution framework could theoretically address the requirements of knowledge change management within closely linked discipline data.

In conclusion, these works from the Model-Driven Engineering research community partially address the requirements of KCM in MDEE as mentioned in Section 5.2. Benefits of using Model-Driven Engineering techniques for knowledge change management in engineering projects include the availability of a good tool support and solid theoretical frameworks [MV11, TELW14].

5.4 Related Work

In this section, we summarize related work from the SW research community in terms of the KCM requirements stated in Section 5.2. Table 5.2 provides an overview of the extent to which each requirement is addressed by each work we overview.

[1]http://pi.informatik.uni-siegen.de/CVSM

$$\begin{tikzcd}
MM_D \arrow[r] & MM_I \\
m \arrow[u, dashed] \arrow[r, "\mathbf{T}"] & T(m) \arrow[u, dashed]
\end{tikzcd}$$

Figure 5.2: Basic co-evolution schema, adapted from [MV11]

	Type of interlinking			Amount of data	Changes granularity		Changes validation		Changes detection		Changes type	
	Single-ontology	Loose-interlinking	Close-interlinking	>1M triples	Instance changes	Schema changes	Semi-automatic	Automatic	Low-level changes	High-level changes	Ontology evolution	Ontology versioning
KCM in MDEE			X	X	X	X	X		X	X	X	X
[Kle04]		X				X					X	
[Sto04, SM02]		X	(X)		X	X	X	X			X	X
[NCLM06]	X					X			X		X	
[PFF09b]	X			(X)	X	X			X	X	X	
[GPS10]	X					X			X	X	X	
[Zab11]	X					X		X			X	
[VCV^{+}13]		X			X				X			X
[HRTM13]		X		X	X	X	(X)		X		X	(X)
[GHU14]	X			(X)	X				X			X

Table 5.2: Related work on KCM requirements

- **Closely interlinked knowledge**. Closely interlinked knowledge is a condition where changes in one ontology within a system of interlinked ontologies may require propagation of the changes to the linked ontologies to maintain the global validity of the knowledge. This is not the typical setting for KCM in SW community, which primarily focuses with open web data. This difference is reflected within most of traditional [GHU14, GPS10, Zab11, PFF09b] or multiple loosely interlinked ontologies [HRTM13, Kle04, RSDT08, VCV$^+$13], where changes in an ontology are independent and do not have to be propagated in order to maintain the validity of overall knowledge within a system. The work of Stojanovic is an exception to this trend, where she provided an attempt to propagate changes to relevant ontologies [Sto04]. However, her work is not further continued.

- **Support for change management of large amounts of data**. Horridge et al. provide an answer to the large-scale challenge of the changed data by introducing binary formats for storing ontology data and differences between ontology versions [HRTM13]. Their approach is claimed to handle more than one million triples. A different approach is adopted by Graube et al., where named graphs are used to store changes and ontology versions [GHU14]. Their approach did not scale well for change data analysis, since the query performance on the change data dropped significantly after several thousands of triples. Papavassiliou et al., on the other hand, successfully experimented their approach on almost 200k triples [PFF09b]. While the current approaches seem promising, given the closely-coupled nature of the engineering data, these approaches need to be re-evaluated in order to asses their feasibility.

- **Changes in schema and instances**. Instance changes are required for addressing changes in the underlying data sources, whereas schema changes are crucial for addressing changes in the represented domains and changes in the intended use of the models and data, as previously mentioned in Section 7.2. Several ontology change management approaches are already able to deal with both schema and instance level changes [HRTM13, PFF09b, Sto04], where other approaches either focus on schema [GPS10, NCLM06, Zab11] or focus on instances [GHU14, VCV$^+$13].

- **Change validation support**. This aspect of validation provides a mechanism to ensure the validity of data changes according to a predefined set of validation rules (automatic validation) or in combination with domain experts' involvement according to certain workflows (semi-automatic validation). Several approaches already support automatic change validations [HRTM13, Sto04]. Furthermore, there are approaches from general SW concerning data validation and linked data quality, e.g., RDFUnit [KWA$^+$14] and Shape Expression [BGP14] that can be adapted to support ontology change validation. In the direction of semi-automatic validation, Stojanovic et al. proposed a mechanism to involve domain experts to check the semantic validity of ontology changes over multiple ontologies [SM02]. This involvement of stakeholders is indeed important in the MDEE due to the mission-critical characteristic of the domain, as we previously mentioned in Section 7.2.1.

- **High-level change definition and detection**. One of the goals of KCM is to provide stakeholders with a better decision support system. The high-level change definition and detection process helps to achieve this goal by providing a mean to detect and encapsulate atomic changes into more meaningful and higher-level changes in terms of domain concepts, which are easier to understand, especially to non domain experts. In this regards, Papavassiliou et al. have developed an algorithm to support the detection of high-level changes from low-level changes, simplifying the effort to analyze changes in large datasets and without compromising performance [PFF09b]. Alternatively, Gröner et al. used a subset of OWL-DL reasoning to recognize high-level change patterns [GPS10]. One of the prerequisites for high-level change definition and detection is the formalization of low-level changes. This formalization can be achieved by using triple patterns [PFF09b, GPS10, VCV+13, HRTM13, GHU14] or specialized ontologies [PFF09b, PHCGP09].

- **Support for data evolution and versioning**. Ontology evolution (i.e., how to modify the data according to relevant changes in the surrounding) and versioning (i.e., management of different versions of data schema and instances caused by ontology evolution) are both important to the KCM process and should be available and easily accessible by relevant stakeholders. Most of the ontology change management approaches focus either on ontology evolution [GPS10, Kle04, NCLM06, Zab11] or on ontology versioning [GHU14, VCV+13]. The rest of the approaches we surveyed try to address both ontology evolution and versioning [Sto04, HRTM13].

To conclude, parts of KCM requirements in MDEE are already well explored in SW research. Schema and instance changes, for example, are addressed already by most approaches. Likewise, approaches for change detection, ontology evolution and ontology versioning are well researched and reported, providing ample options to choose from. **However, due to the open nature of web data, approaches for ontology changes in closely interlinked knowledge settings are rarely investigated**. Similarly, approaches for handling changes of large amounts of data and validating changes are currently limited, probably since these aspects are not the focus in current ontology change management research. There are options to use general ontology validation approaches for ontology change validation, i.e., by adapting RDFUnit [KWA+14] or Shape Expression [BGP14] approach, but these adaptations are not yet seen as integral part of general ontology change management approaches. We therefore see the need to advance and combine existing approaches such that all KCM requirements are sufficiently addressed. The KCM reference framework presented next provides a conceptual framework to guide this process.

5.5 The Generic Reference Framework

In order to address the challenges of providing support for KCM in MDEE, we propose a generic reference framework shown in Figure 5.3. This is a technology agnostic process that could be implemented with technologies drawn from SW or other related technologies, such as databases or model-based engineering. Implementations using different techniques but following this reference framework will be easier to compare and will support a more objective comparison of the strengths and weaknesses of the available technologies. There are KCM requirements for which we cannot cater at the process level but which should be considered during the implementation of the reference framework (e.g., dealing with large amounts of data).

The reference framework was derived by adapting and extending the OBDI approach (cf. Section 2.3) thus closely connecting process steps for data integration and change management. The reference framework is technology agnostic: for this, we replaced all SW specific terms with general terms, e.g., 'ontologies' with 'data models'. The extension of OBDI consisted in adding four more phases. These phases were derived from relevant related work and requirements and are shown as white boxes in the Figure 5.3 while the original phases are shown as gray boxes. We utilize an IDEF-0 style diagram to structure the proposed approach, in which processes are shown as boxes and resources are shown as directed arrows. The diagram clearly defines input (incoming arrows from the left hand side of the box), output (outgoing arrows to the right hand side of the box), consumable resources and stakeholders (input arrows from the bottom of the box) and standards (incoming arrows from the top of the box) used in the reference framework.

There are three domain expert roles involved in the framework: Knowledge Engineer, Project Manager, and Domain Expert. Input and output of the system is shown in the left and right side of the diagram respectively. In the following, we explain the seven main phases of the KCM reference framework:

1. **Local Data Model Definition**. This phase requires the Knowledge Engineer and Domain Experts to translate the local tools data structure (e.g., MCAD model for mechanical engineer) to the local data model instance definition.
2. **Common Data Model & Mapping Definition**. Knowledge Engineer and Domain Expert will define the common data model and its mappings to the local data models. To support this goal, vocabularies and standards are required to formalize the data model and mapping.
3. **Local Data Model Extraction, Transformation and Load (ETL)**. With regards to the heterogeneous domain tools and their data formats within the MDEE, we need to provide the suitable extract, transform, and load (ETL) functions phase to produce the data in the required data model formats.
4. **Change detection**. This phase focuses on the detection of low-level (i.e., triples) and high-level (e.g., semantic and domain-specific) changes between two versions of

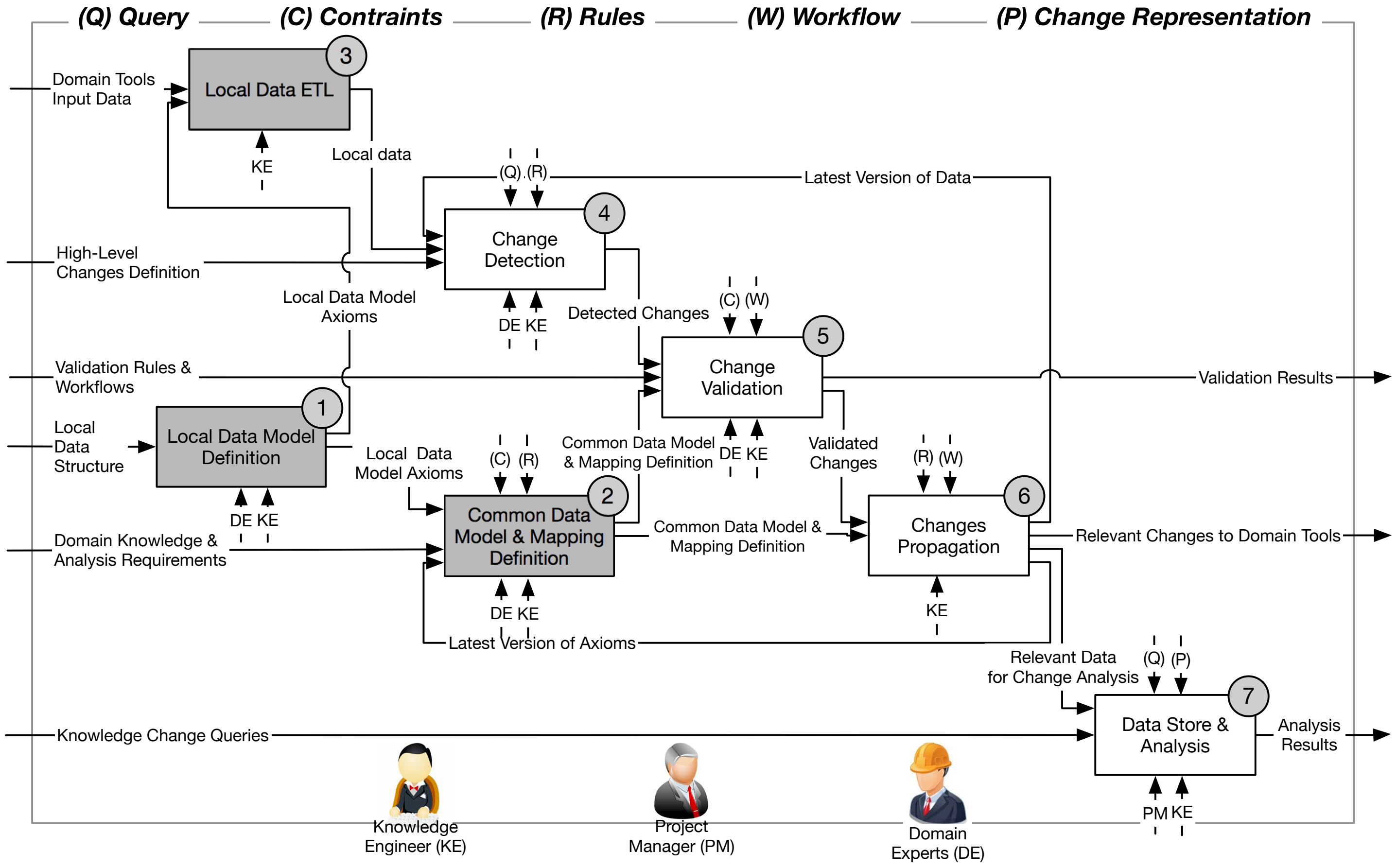


Figure 5.3: reference framework model for KCM in the MDEEs. The white boxes extend a typical OBDI process (grey boxes).

engineering data. An important point to consider within this phase is to balance the expressiveness of high-level changes defined as input and the computational complexity of the detection algorithm, as mentioned in [PFF09b].

5. **Change validation**. The phase of change validation requires the definition of constraints for preserving the validity of data in the local (e.g., mechanical engineering) and global data models (e.g., power plant). Workflow definition is another important element, in order to configure involvement of validation components (e.g., constraint validation engine and domain experts) in the validation process.

6. **Change propagation**. Changes in the MDEE need to be propagated to the relevant components (i.e., common data model and other relevant local data models). This phase requires the common data model and mapping definitions, as well as validated changes. The knowledge engineer will configure the propagation based on the mapping definitions to make sure that no corrupted or irrelevant data is included in the propagation process.

7. **Data Store and Analysis**. The goal of this phase is to enable relevant stakeholders (e.g., project manager) to access and analyze the data and its changes within the projects. The changed data will be stored within a designated data store. Examples of queries that will be relevant to this data are: (1) Provenance information of the changes (e.g., committer, date, reasons of change), (2) Change overview on specific objects, and (3) Analysis of completeness and inconsistencies over changes.

5.6 Semantic Web-Based Prototype

To demonstrate the feasibility of the proposed framework explained in Section 5.5 within a real-world scenario and answering our RQ2: *How to provide sufficient support for Knowledge Change Management in Multi-Disciplinary Engineering Environments beyond the Ontology-Based Data Integration approach?*, an implementation in a specific technology is needed. To address this challenge, this section reports on the development and evaluation of our SW-based KCM tool prototype as an instance of the framework. This section consists of the following: Section 5.6.1 describes the use case of engineering Hydro Power Plants as an example of MDEEs, Section 5.6.2 reports on the KCM prototype development using SW technologies, and Section 5.7 discusses the feasibility evaluation of the prototype.

5.6.1 Use Case: Hydro Power Plant Engineering

This section presents a multi-disciplinary engineering use case provided by one of our industrial partners developing, creating, and maintaining hydro power plants.

A hydro power plant manages from 40 to 80 thousand signals, depending on the size of the commissioned plant, in different tools of different engineering disciplines. Signals consist of structured key value pairs that represent communication links between different power plant components (an example of a signal list is shown in Figure 5.1). They are one of the core information artifacts in the course of developing power plants. In order to exchange data with engineers from different engineering disciplines, signal information is typically exported from the discipline-specific tools in machine-readable formats, such as Comma Separated Values (CSV) or eXtensible Markup Language (XML), to be used by other disciplines.

The engineering of the hydro power plants is conducted in parallel. Engineers from different disciplines work on their own part but rely on the exchanged signal data to coordinate their work on the system with other engineering teams. Here, the challenge of providing a KCM arises due to the heterogeneity of terms used for the same concept and the implicit linking between engineering objects in different engineering specific tools. For example, information about a CPU is stored as part of the composite programmable logic controller (PLC) address in EPLAN (electrical engineering tool) data, while it is stored as property LK_BSE in OPM (mechanical engineering tool) data. Such heterogeneous representations of the same engineering artifact within diverse engineering models raises the need for propagating changes across data from different engineering disciplines. In this particular example, changes to the CPU at the mechanical level (e.g., replacement with a new version) must be communicated with the electrical engineers to update their models accordingly.

In the hydro power plants use case, there are two engineering specific tools used to produce signal information: (1) OPM is used in mechanical engineering to develop the plant topology and its components and (2) EPLAN[2] is used by electrical engineers to develop the electrical component of the power plants. Additionally to signal information, relevant general information includes information about the engineering project, customer, and engineering activities.

To sum up, in this use case there are three different local data sources whose changes need to be managed in an integrated way:

- **Mechanical Engineering Data (OPM)**. The mechanical data source is available as CSV files, exported from the OPM tool. It represents the design of the overall structure of the mechanical components. However, depending on the project type, the exported file can also contain information about software components.

- **Electrical Engineering Data (EPL)**. The electrical data source is available as CSV files, exported from the EPLAN tool used in the electrical engineering domain. This data contains information about the electrical setup and its link to the mechanical components. It also contains specific information about electrical components, which may or may not be useful to stakeholders from other engineering domains.

[2]http://www.eplanusa.com/

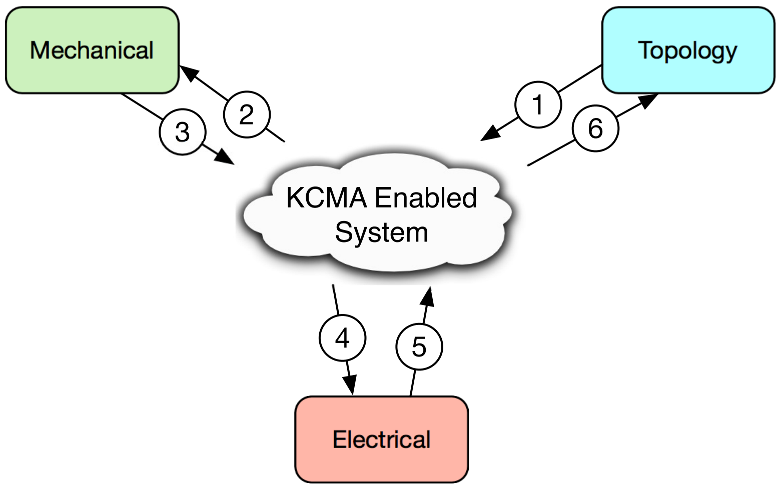


Figure 5.4: The hydro power plant engineering process

- **Project Management Data (PMO)**. This data source is available as spreadsheet files that contain the project information as well as engineers and customers involved in the project. This document is used across the system for project identification purposes.

Figure 5.4 shows a simplified process of the use case. This process is iterative, i.e., it will be conducted repeatedly, and concurrent, i.e., engineers from different engineering disciplines may be working at the same time. The project steps indicated with numbers on Figure 5.4 are:

1. The project manager commits the initial project information into a KCM enabled system on request from the Client.
2. A mechanical engineer reads the initial project information from the system, and designs the mechanical part of the plant based on project information.
3. The mechanical engineer commits the first version of mechanical plant design into the system.
4. An electrical engineer retrieves the plant topology and mechanical component design from the system, and designs the electrical components of the plant.
5. The electrical engineer commits the electrical components design of the plant into KCM enabled system.
6. The project manager retrieves the hydro power-plant engineering data (including changes' information) from the KCM enabled system and analyzes the data to understand the project progress.

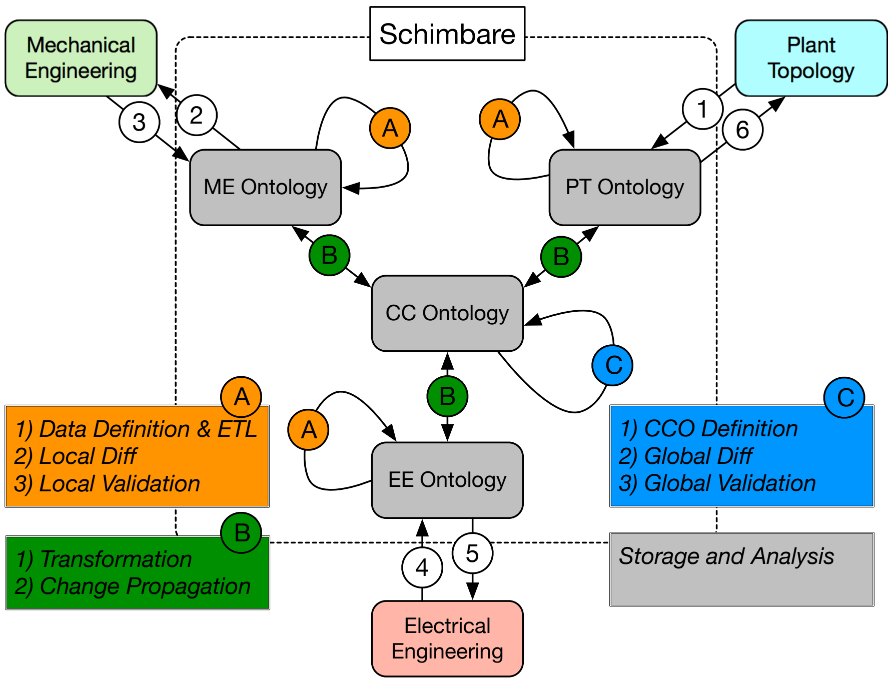


Figure 5.5: The SW-based KCM solution for hydro power plant use case

5.6.2 KCM Prototype Development

In this section, we will explain the application process of the framework (cf. Section 5.5) into a solution prototype for the hydro power plant use case using SW technologies. In the prototype development, we are using the following assumptions:

- Fully automatic process. In order to simplify the KCM process, we assume that it is fully automatic, i.e., no human interaction is accommodated, for example, for validating the proposed changes. It is essential to keep it this way to avoid cluttering the process in this feasibility study.
- Strict importance hierarchy of local data sources for change propagation. We assume that there exists a strict order for propagating changes among the local data sources. In our use case, Project Management Ontology (PMO) holds the highest priority, followed by OPM and EPL data. The strict importance hierarchy reduces the risk of deadlock during change propagation of data and minimizes the need of human interaction with the prototype.

The prototype contains four RDF repositories, one for each local data model: OPM for mechanical data, PMO for project management data, EPL for electrical data, and Common Concept Ontology (CCO) for the common concepts. It relies of the following three components that build on top of semantic data repositories: (A) Local Data

Management, including local change detection and validation, (B) Transformation and Change Propagation mechanisms between local and common concept repositories, and (C) Common Data Management, including CCO change detection and validation. These components are shown in Figure 5.5 and described next.

A. Local Data Management

CSV to one-class RDF transformation. We first clean the CSV input data using OpenRefine[3], a cleaning tool for tabular data. In this process, we clean up split the data to extract only relevant data from the raw CSV input data. Afterwards, we convert each CSV from the local data source into an RDF Graph with one concept and number of properties using CommonCSV[4], a CSV manipulation library for Java, and Apache Jena[5], a Java RDF API, and store it in the repository. While the one-class RDF does not provide much semantic information in the repository, it allows easier data transformation between different data repositories (i.e., between local data and CCO). We treat this one-class RDF as the local data model for each engineering discipline. This step corresponds with the 1st and 3rd phases in our generic reference framework shown in Figure 5.3.

Local Change Validation. The next step is to conduct the validation of the input data. There are several domain rules that are applied here, e.g., Kraftwerk-Kennzeichensystem (KKS)[6] keys data has to be complete for OPM data and PLC address has to be complete for EPL data. If the data is not adhering to the domain rules, it will be rejected. The validation is implemented using SPARQL queries, and executed before any other process is conducted. This step corresponds with the 5th phase in our reference framework.

Local Change Detection. We first detect the low-level changes (i.e., triple-level changes) between the new input and the current version of stored local data. We then transform these triple-level changes to conform with our change representation format, which combines PROV-O [BCC+13] for engineering document provenance (i.e., commit versions from different disciplines) and Change Set ontology for content changes (i.e., triple level changes). This step corresponds with the fourth phase in the reference framework.

B. Transformation and Propagation Management

Transformations between various RDF data sources (e.g., between local and global level data) are required to enable efficient change propagation of engineering data. In the following we explain the RDF-to-RDF transformation mechanism and the change propagation steps that utilize this transformation.

RDF-to-RDF transformation. Because we store the local data in one-class RDF a transformation mechanism is needed to transform it into CCO data and vice-versa. There

[3]OpenRefine: http://openrefine.org

[4]CommonCSV: https://commons.apache.org/proper/commons-csv/

[5]Apache Jena: http://jena.apache.org

[6]https://www.vgb.org/en/db_kks_eng.html

are several options to execute the transformation, including SPARQL Construct and SPIN. In the end, we choose SPARQL Construct since it is a W3C standard. This step corresponds with the 2nd phase in our generic reference framework.

Change Propagation. In the use case, we use RDF-to-RDF transformations to propagate the data from local to common model and from the common model to other valid models. Additionally, the metadata information about the change is also propagated to keep local repositories aware of the relevant changes coming from other data sources. This step corresponds with the sixth phase in our generic reference framework shown in Figure 5.3.

C. Common (Global) Data Management

The Hydro Power Plant common concept ontology (CCO) represents the common (global) information relevant for different engineering disciplines involved in the use case.

The CCO Ontology. The CCO consists of two major parts. First, as depicted on the left hand side of Figure 5.6, the ontology contains concepts that describe organizational level aspects. These concepts include the Project, the Customer for whom the project is performed, as well as the Engineers (and Engineering Roles) necessary to realize the project. Engineers conduct Engineering Activities, which take as input and create as their output various Engineering Documents (e.g., signal lists, design documents). Engineering documents are versioned and reviewed by the customer, thus constituting an important exchange medium between the customer, who requested a project, and the engineering team executing that project. This step corresponds with the second phase in our reference framework.

Second, the CCO describes various Engineering Objects created during the engineering project (right-hand side of Figure 5.6). The ontology identifies different types of engineering objects, such as Software Objects, Mechatronic Objects, and Electrical Objects.

The ontology also clarifies the various parts of a mechatronic object, their internal structure and connections among them. To that end, the ontology captures different types of Mechatronic, Electrical and Software objects, and details their internal structure at a high level of abstraction. Physical Signals and Logical Signals represent the links between engineering objects created by different engineering disciplines and how these diverse components can command or exchange data with each other. In addition to these signals, detailed descriptions of the various mechatronic components are also available as engineering documents (e.g., PLC programs for software objects, or ECAD diagrams for the electrical wiring).

The internal structure of the components captured by the ontology emerged during several projects, and can also be represented using other approaches, such as the AutomationML instance hierarchy or domain-specific structuring standards.

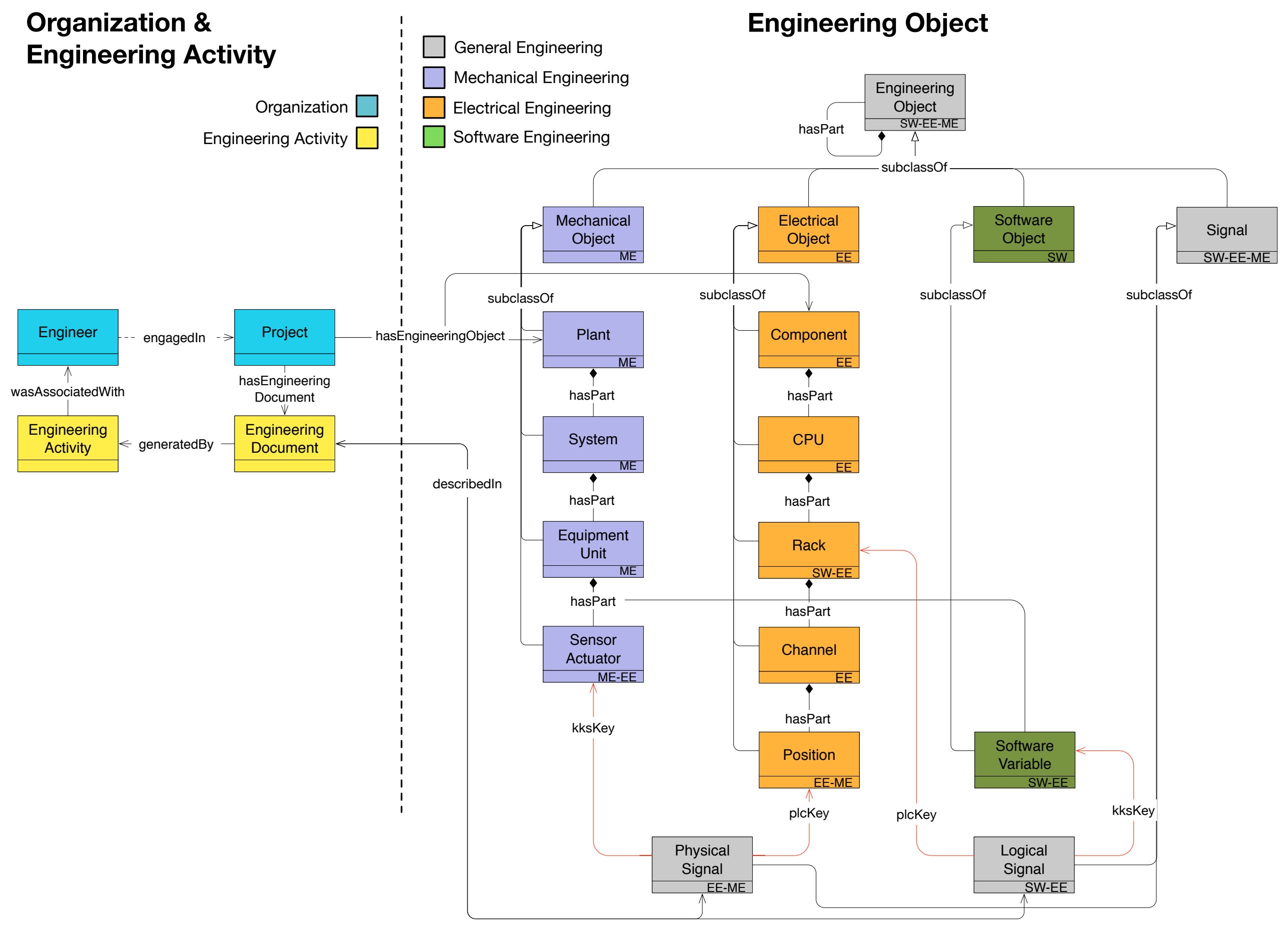


Figure 5.6: The Common Concept Ontology for Hydro Power Plant

Common Changes Validation and Propagation. When a new change from one discipline is validated in that discipline, it will be transformed into CCO format and checked against data from higher-level hierarchy. In our case, PMO data holds the highest importance, followed by OPM data and EPL data. For example, if there are new data coming in, it will always be accepted regardless of whether other local sources are affected. On the other hand, if data from EPL is committed, if it updates of deletes data from OPM or PMO, it will be rejected. This step corresponds with the sixth phase in our reference framework.

Change Vocabularies and Data Storage

Previous components are built on top of a foundation of data storage facility and change vocabularies to enable analysis of KCM data. We will explain both elements in the following, which correspond with the seventh phase in our reference framework.

Change representation. Ontology changes can be represented either as triples (e.g., DBPedia change representation [SMLH10]) or specialized ontologies for change representation (e.g., change representation for OWL2 [PHCGP09], CHAO [NCLM06] and Talis change set[7]). For change representation, W3C provides a recommendation for provenance information (PROV-O [BCC+13]) that can be used in conjunction with the change representation ontologies. In our use case, we decided to use the combination or PROV-O and Talis change set to represent the changed data.

PROV-O (together with FOAF ontology) is integrated with the PMO data (e.g., `cco:EngineeringDocument` is a subclass of `prov:Collection` class), while all triple changes between two engineering documents are represented as Talis change set (i.e., `cs:ChangeSet` is the range of `prov:hadMember` property of `cco:Engineering-Document`). A complete overview of the change representation is shown in Figure 5.7.

Table 5.3: Set of commit sequences for the feasibility evaluation.

Person	Activity	Document	Type
Andy	pm_design_00	andy-pm_design_00-pmo.csv	PMO
Brad	m_design_00	brad-m_design_00-opm.csv	OPM
Casey	e_design_00	casey-e_design_00-epl.csv	EPL
Casey	e_design_01	casey-e_design_01-epl.csv	EPL
Casey	e_design_02	casey-e_design_02-epl.csv	EPL
Casey	e_design_03	casey-e_design_03-epl.csv	EPL
Brad	m_design_01	brad-m_design_01-opm.csv	OPM
Brad	m_design_02	brad-m_design_02-opm.csv	OPM

[7] http://vocab.org/changeset/

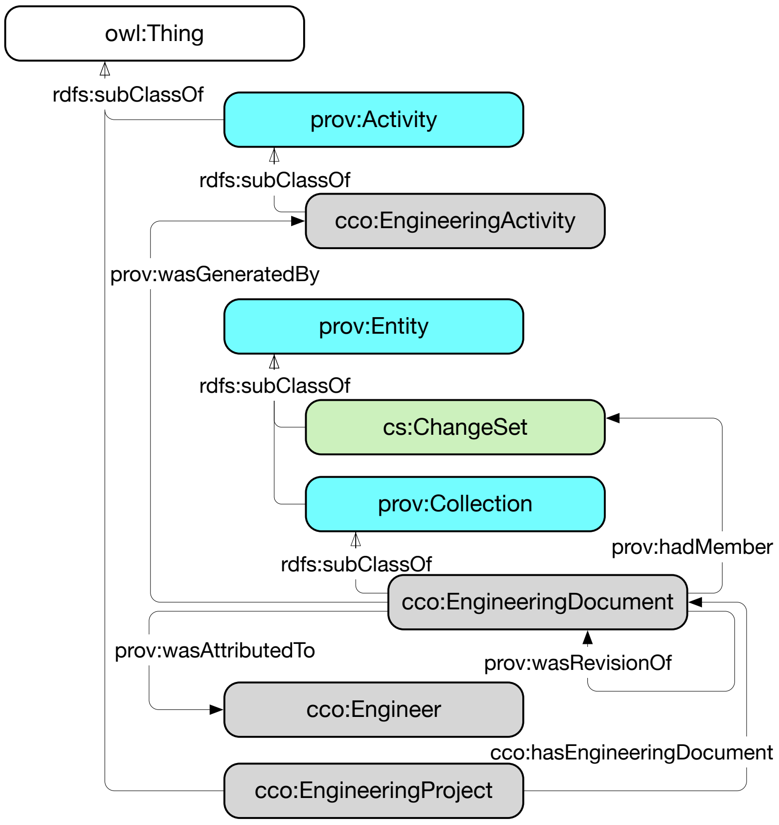


Figure 5.7: Change representation in hydro power plant use case

5.7 Feasibility Study

To test our prototype[8], we extract a portion of data from hydro power plant local data sources. Table 5.3 shows a generic information sheet about the data to be inserted into three local data sources. We commit the data according to the sequence in Table 5.3, and we provide invalid data on purpose for two of the steps above (marked with bold-italic text) to check the validation step.

The result from our feasibility evaluation shows that the following stages of the KCM framework were successfully covered: the CSV data was successfully transformed into local RDF data (1st and 3rd phases of the KCM generic reference framework), it was possible to detect data change from its previous versions (4th phase), the data was mapped to the common data (2nd phase), changes from both local and global perspectives were validated (5th phase), changes were propagated to other local data sources (6th phase), as well as stored and analyzed (7th phase).

[8]The source code of the prototype and our feasibility study is available at `https://gitlab.isis.tuwien.ac.at/Ekaputra/schimbare-old`

In the process of feasibility evaluation, we define a set of SPARQL construct queries for transforming and propagating data from local to common model (e.g., except the SPARQL Construct for transforming PMO to CCO data as shown in Listing 1) and vice versa, and also a set of SPARQL queries for validating changes both in local and common models. The time required for insertion, propagation and querying is negligible due to the sample size, but the scalability of the approach should be checked with larger datasets.

Listing 5.1: SPARQL query to find which objects have changed in the last commit.

```
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl:<http://www.w3.org/2002/07/owl#>
PREFIX cs:<http://purl.org/vocab/changeset/schema#>
PREFIX prov:<http://www.w3.org/ns/prov#>
PREFIX sch:<http://juang.id/ontology/schimbare#>

SELECT distinct ?type (count(?subject) as ?subject_count)
WHERE
{
        GRAPH sch:DIFF
        {
                ?commit prov:hadMember ?change .
                OPTIONAL {
                        ?commit2 prov:wasRevisionOf ?commit .
                }
                FILTER (!BOUND(?commit2)) .
                ?change a cs:ChangeSet .
                ?change cs:subjectOfChange ?subject .
        }
        GRAPH sch:ABOX
        {
                ?subject a ?type
        }
}
group by ?type
```

After all the data were committed, the repository allows queries related to changes to be asked to the CCO repository. Example queries are:

- *What are the objects that have been changed in the last commit*? (cf. Listing 5.1)
- *How many racks added in the overall commit*? or
- *How many changes occurred to a signal named "KOM/0. BBA00.GS104.XB01_050.04.02.5.07" in the last 4 commits*?

In Listing 5.1, we show the query for summarizing the number of objects that have changed in the last commit, grouped by their type as shown in Figure 5.6. In the prototype, we used several named graphs to represent different parts of the data, e.g., `sch:DIFF` to store the changes, `sch:ABOX` to store the latest version of the data, and `sch:TBOX` to store the CCO ontology. These named graphs are reflected in the query shown in Listing 5.1.

Two example results are shown in the Table 5.4 and 5.5, representing the object changes from the second and third commit in our scenario (cf. Table 5.3) respectively.

Table 5.4: Object changes for the second commit

Object type	Object count
<http://data.ifs.tuwien.ac.at/engineering/cco#Channel>	5
<http://data.ifs.tuwien.ac.at/engineering/cco#Rack>	2
<http://data.ifs.tuwien.ac.at/engineering/cco#Cpu>	1
<http://data.ifs.tuwien.ac.at/engineering/cco#PhysicalSignal>	13
<http://data.ifs.tuwien.ac.at/engineering/cco#EngineeringProject>	1
<http://data.ifs.tuwien.ac.at/engineering/cco#SensorActuator>	13
<http://data.ifs.tuwien.ac.at/engineering/cco#System>	2
<http://data.ifs.tuwien.ac.at/engineering/cco#Position>	13
<http://data.ifs.tuwien.ac.at/engineering/cco#Plant>	1
<http://data.ifs.tuwien.ac.at/engineering/cco#EquipmentUnit>	5
<http://data.ifs.tuwien.ac.at/engineering/cco#ElectricalComponent>	1

Table 5.5: Object change for the third commit

Object type	Object count
<http://data.ifs.tuwien.ac.at/engineering/cco#Channel>	6
<http://data.ifs.tuwien.ac.at/engineering/cco#Cpu>	1
<http://data.ifs.tuwien.ac.at/engineering/cco#Rack>	1
<http://data.ifs.tuwien.ac.at/engineering/cco#PhysicalSignal>	8
<http://data.ifs.tuwien.ac.at/engineering/cco#EngineeringProject>	1
<http://data.ifs.tuwien.ac.at/engineering/cco#SensorActuator>	8
<http://data.ifs.tuwien.ac.at/engineering/cco#System>	1
<http://data.ifs.tuwien.ac.at/engineering/cco#Position>	8
<http://data.ifs.tuwien.ac.at/engineering/cco#Plant>	1
<http://data.ifs.tuwien.ac.at/engineering/cco#EquipmentUnit>	2
<http://data.ifs.tuwien.ac.at/engineering/cco#ElectricalComponent>	1

5.8 Summary

In this chapter, we have defined the context and challenges of KCM in MDEEs. To address the challenges, we have identified key requirements and provided an overview of techniques to address these challenges from the SW research community.

Furthermore, we generalize and extend the OBDI approach, previously proposed for the purposes of data integration, to develop a generic reference framework of KCM in MDEEs. This generic and technology-agnostic reference framework is meant to lay the foundation towards a solution for providing a fully functional KCM solution for MDEEs, which address our second research question of the thesis (**RQ2**): *"How to provide sufficient support for Knowledge Change Management in Multi-Disciplinary Engineering Environments beyond the Ontology-Based Data Integration approach?"*.

To evaluate our framework, we implemented a prototype based on the framework using SW Technologies to show its feasibility for MDEEs. Through creating a prototype, we have shown that it is possible to adapt the reference framework for a real-world scenario. For it to be adopted, however, the current prototype needs to be improved and better evaluated to address the challenges arising from the use case, especially with regards to the scalability and ease of use.

CHAPTER 6

Conclusion

The fourth industrial revolution or "Industrie 4.0" brings forth the need for more flexible production systems that requires strong data integration between involved stakeholders and engineering disciplines. The lifecycle of such production systems typically happens in a Multi-Disciplinary Engineering Environment (MDEE), where stakeholders from various engineering disciplines work together in a highly heterogeneous environment. In an MDEE, collaboration among stakeholders requires synchronization and exchange of data produced by different tools and methods specific to their respective engineering disciplines. Despite of this, cross-disciplinary knowledge in MDEEs often available only implicitly, which results in tedious tasks of manually retrieving and processing such knowledge. Therefore, addressing the challenge of providing a strong data integration support and advanced applications based on this integration, such as Knowledge Change Management (KCM) in MDEEs is an important and timely topic.

The work in this thesis investigates mechanisms and methods from Semantic Web (SW) research to support data integration in MDEEs as well as advanced applications, such as KCM and cross-disciplinary data analysis that build on top of the data integration foundation.

6.1 Reviewing the Research Questions

In the introductory chapter, we formulated the following central research question.

> ***Which mechanisms and methods from Semantic Web technologies are suitable to address challenges of Data Integration and Knowledge Change Management in Multi-Disciplinary Engineering Environments?***

In order to answer this research question, we will discuss the two more specific research questions in which the general question is split up.

RQ1: ***"How suitable are the Ontology-Based Data Integration approach variants for the diverse data integration scenarios in Multi-Disciplinary Engineering Environments?"***

When first considering the central research question, the answer for the data integration part seemed to be straightforward: there is already an established method called Ontology-Based Data Integration (OBDI) with its three variants: single-ontology, multiple-ontology, and hybrid, where the hybrid variant was recommended for typical use cases due to the strengths and benefits that combines the other two variants [WVV+01].

However, from our initial investigation, we found that different OBDI variants are being adopted in Multi-Disciplinary Engineering Environments (MDEEs)'s use cases. Such findings motivated us to conduct a literature study on the applications of OBDI in MDEEs as reported in **Chapter 3**.

In the study, we further our understanding on how OBDI approaches are being applied in MDEEs, including the technical realization options and the characteristics of data integration scenarios in the environment. Additionally, we analyze the strengths and limitations of each OBDI variant against MDEE scenarios' characteristics that we observe. During the literature study process, we notice an additional OBDI variant of Global-as-View (GAV) that is not yet covered, and propose to add it to the existing typology of OBDI. Finally, the study proposes a guideline to support users in choosing the most suitable OBDI approach variant based on their MDEE scenario characteristics.

To validate our understanding of the OBDI variants and their suitability in MDEE, we develop concrete OBDI prototypes to address challenges of advanced applications built on data integration, such as engineering data analysis and cross-discplinary defect detection. **Chapter 4** reports on selected prototypes, their design and developments, as well as justifications on why we select a certain OBDI variant for each use case.

RQ2: ***"How to provide sufficient support for Knowledge Change Management in Multi-Disciplinary Engineering Environments beyond the Ontology-Based Data Integration approach?"***

While OBDI allows development of Knowledge Graphs in a MDEE as a basis for advanced applications, it does not fully support the management of knowledge graph changes, partly due to fact that the topic of Knowledge Change Management (KCM) has primarily been investigated for the single-ontology variant of OBDI. This is an important topic in MDEE, especially in cases where changes are integral part of the routine engineering process.

As an effort to address this issue, **Chapter 5** focuses on providing KCM capabilities for MDEE scenarios that build on the glsgav OBDI approach. The chapter begins with the definition of a set of KCM requirements for MDEEs [ESSB15b] and follows it up with an analysis of related works on KCM from the Semantic Web (SW) research community with regards to the requirements. Based on this analysis, we design and develop a generic framework for KCM for MDEE that builds on the OBDI components. [Eka16].

In order to evaluate the proposed framework, we instantiate the framework as a research prototype and conduct an initial feasibility study in the use case of a hydro-power plant engineering process, which is based on a real-world use case from our industry partner [ESSB16].

To sum up, based on the presented contributions, the central research question can be answered positively, since we show that the OBDI variants –including the newly observed GAV– are particularly suitable to support various data integration scenarios in MDEEs and our proposed KCM framework for MDEEs could support the management of knowledge change beyond single-ontology OBDI approaches.

6.2 Limitation of the Thesis

The work in this thesis has a number of limitations. With regards to **RQ1**, our work focused on the applicability of the Ontology-Based Data Integration (OBDI) variants in Multi-Disciplinary Engineering Environments (MDEEs) use cases. Due to this focus, we do not provide comprehensive comparisons between OBDI approaches and alternative technologies. Nevertheless, we include comparisons of Semantic Web (SW) technologies in general to other alternative technologies used in MDEEs (Section 4.3). We also address the topic of alternative technologies for Knowledge Change Management (KCM) in Section 5.3.

In addressing **RQ2**, we spent significant efforts in providing a solid conceptual approach towards supporting the KCM processes in Multi-Disciplinary Engineering Environment within the context of the Christian Doppler Laboratory "*Software Engineering Integration for Flexible Automation Systems*" (CDL-Flex) project. However, due to the shift of focus in the project towards its end in 2016, the resources –especially from our industry partners– were only scarcely available to support the development and evaluation of the KCM prototype. Therefore, we could only conduct the prototype development and its evaluation to a limited extent.

6.3 Future Work

In this section, we outline our planned future works in our two main research topics: Ontology-Based Data Integration (OBDI) and Knowledge Change Management (KCM).

6.3.1 Ontology-Based Data Integration

An Extension of the OBDI Survey Beyond the Current Domain. Currently, the scope of our survey and its results is limited to the engineering domain, more specifically in those applications within the boundary of the Multi-Disciplinary Engineering Environment (MDEE) settings. We are cautiously optimistic that our findings can be relevant beyond this specific domain. To this end, we would like to extend the validation of our proposed guideline with general data integration scenarios. An initial work within this direction is currently underway to conduct a similar survey in the Tourism domain.

Expressiveness of the Ontology Framework Another line of possible future work is a research on the expressiveness of the ontology framework. We observed during our literature study that several OBDI-based systems in MDEE use non-Semantic Web (SW) ontology frameworks (i.e., KFL and F-Logic) in their application, arguing that Semantic Web Technologiess (SWTs) are not sufficiently expressive for their data integration needs. Since the limitations of SWTs have not been systematically investigated in this context, we identify this issue as a research gap and therefore worthy of further research.

RDF Graph Transformations. During our thesis, we realize that there is a need for an investigation on RDF graph transformation methods to provide a better support for the mapping and transformation processes in GAV OBDI approach beyond SPARQL construct with rules. Currently, a standardization effort in this direction is ongoing within the SHACL W3C working group. To be precise, they are working towards SHACL Advanced Features (AF)[1], of which one of the main key component is RDF graph rules for transforming RDF graph data.

OBDI Applications in Other Domains. In the last few years, we have been involved in many discussions with colleagues and domain experts from within and outside of the Semantic Web area. To this end, we discussed the possibilities to bring our knowledge of OBDI to other application domains outside engineering, such as personal data management (i.e., in the wake of the recent enforcement of EU-GDPR law [Eur16]) and Digital Humanities.

As results, we have been involved and co-authored two research proposals on the topic related to the application of OBDI in the domain of personal data management and one short proposal in the digital humanities domain. One of these proposal, that we wrote together with our colleagues from WU Wien[2] and OwnYourData[3] entitled "*EXPEDiTE: EXPloring opportunities and challenges for Emerging personal DaTa Ecosystems: Empowering humans in the age of the GDPR - A Roadmap for Austria*" has been accepted and will be started on October 2018, focusing on the designing the blueprint of personal data ecosystems in Austria using SWTs and based on OBDI principles.

6.3.2 Knowledge Change Management in Hybrid OBDI

KCM Tool Extensions. There is a clear need of the KCM support for OBDI approach in the context of MDEEs, as explained in Chapter 5. To this end, we are planning to extend of our current KCM framework prototype to fully support the necessary requirements in MDEEs. Our current KCM prototye have shown the potential of our approach to address some of the identified requirements. The prototype, however, requires

[1]https://www.w3.org/TR/shacl-af/
[2]WU Wien: www.wu.ac.at
[3]OwnYourData: www.ownyourdata.eu

further developments to fully address all the requirements needed in MDEEs. To this end, we are planning to identify and evaluate several improvements ideas and options based on the recent advances on ontology change research, such as RDF Graph co-evolution [FES$^+$16] and Git-based RDF versioning [ARM16].

We also consider to reuse the *Semantic Container* concept coming from the aviation domain [NGS$^+$17] and adapt it for the KCM context. The Semantic Container concept allows data providers to conduct efficient distribution without giving up control over its usage while providing data consumers with efficient and well-managed mechanisms to obtain and use data in a trustworthy and reproducible manner. Since Semantic Container allows packaging data and processing capabilities into reusable containers, describing the semantics of the content and permissible usage, and providing uniform interfaces, a data set contains well-defined content and quality, as well as clear ownership and usage provenance. We foresee that such approach would allow a more transparent, trustworthy, and reproducible KCM approach, which is crucial in the safety- and mission-critical domain such as industrial production systems engineering and beyond.

List of Figures

List of Tables

Listings

Acronyms

ASE Automation System Engineering. 28–30, 54, 115

CCO Common Concept Ontology. 97–99, 101–104

CDL-Flex Christian Doppler Laboratory "*Software Engineering Integration for Flexible Automation Systems*". 57, 109

CPPS Cyber-Physical Production Systems. 34, 35, 63, 67

GAV Global-as-View. 4, 5, 26, 48, 49, 108, 109

KCM Knowledge Change Management. 3–6, 11–15, 17, 23, 24, 26, 81–83, 85–87, 89–97, 101, 102, 104, 105, 107–111, 113

MDE Multi-Disciplinary Engineering. 38, 68

MDEE Multi-Disciplinary Engineering Environment. xiii, xiv, 1–6, 12–17, 26–28, 31, 33, 37, 39, 41, 49, 50, 52, 54, 57, 59, 66, 68, 69, 71, 72, 76, 77, 80–87, 89–92, 94, 104, 105, 107–111, 113, 114

OBDI Ontology-Based Data Integration. xiii, xiv, 2–6, 11–13, 17, 21, 23, 24, 26–29, 31–55, 57, 59, 60, 62, 64, 67, 68, 70–72, 76, 80, 81, 83, 92, 104, 105, 108–110, 113, 114

OCDD Ontology-based Cross-disciplinary Defect Detection. 4, 5, 57, 68, 70–73, 76, 80, 113

PMO Project Management Ontology. 97, 101, 102

SLR Systematic Literature Review. 4, 28, 29

SW Semantic Web. 2, 3, 5, 6, 16, 17, 23, 24, 26, 28–30, 32–36, 46, 48, 54, 55, 57, 58, 68, 80, 81, 83, 87, 90–92, 94, 97, 104, 105, 107–110, 115

SWT Semantic Web Technologies. 6, 12, 17, 19, 26, 58, 76, 79, 110

Bibliography

[AG13] Lisa Abele and Stephan Grimm. Knowledge-based integration of industrial plant models. In *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*, pages 4392–4397. IEEE, nov 2013.

[AGZK14] Lisa Abele, Stephan Grimm, Sonja Zillner, and Martin Kleinsteuber. An ontology-based approach for decentralized monitoring and diagnostics. In *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, pages 706–712. IEEE, jul 2014.

[AKL09] Jürgen Angele, Michael Kifer, and Georg Lausen. Ontologies in F-Logic. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, pages 45–70. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[ALGM13] Lisa Abele, Christoph Legat, Stephan Grimm, and Andreas W. Muller. Ontology-based validation of plant models. In *2013 11th IEEE International Conference on Industrial Informatics (INDIN)*, number July 2013, pages 236–241. IEEE, jul 2013.

[ARM16] Natanael Arndt, Norman Radtke, and Michael Martin. Distributed Collaboration on RDF Datasets Using Git. In *Proceedings of the 12th International Conference on Semantic Systems - SEMANTiCS 2016*, pages 25–32, New York, New York, USA, 2016. ACM Press.

[ASF14] Pekka Aarnio, Ilkka Seilonen, and Mats Friman. Semantic repository for case-based reasoning in CBM services. In *19th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2014*, pages 1–8. IEEE, sep 2014.

[AvdADtH06] L Aldred, W van der Aalst, M Dumas, and A ter Hofstede. Understanding the challenges in getting together: The semantics of decoupling in middleware. Technical report, 2006.

[BBM+15] Luca Berardinelli, Stefan Biffl, Emanuel Mätzler, Tanja Mayerhofer, Manuel Wimmer, E. Maetzler, Tanja Mayerhofer, and Manuel Wimmer. Model-Based Co-Evolution of Production Systems and their Libraries with

AutomationML. In *Proceedings of the 20th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2015)*, volume 2015-Octob, pages 1–8, 2015.

[BCC+13] Khalid Belhajjame, James Cheney, David Corsar, Daniel Garijo, Stian Soiland-Reyes, Stephan Zednik, and Jun Zhao. PROV-O: The PROV Ontology, 2013.

[BCW12] Marco Brambilla, Jordi Cabot, and Manuel Wimmer. *Model-Driven Software Engineering in Practice*, volume 1. sep 2012.

[BGL17] Stefan Biffl, Detlef Gerhard, and Arndt Lüder. Introduction to the Multi-Disciplinary Engineering for Cyber-Physical Production Systems. In *Multi-Disciplinary Engineering for Cyber-Physical Production Systems*, pages 1–24. Springer International Publishing, Cham, 2017.

[BGP14] Iovka Boneva, Jose Emilio Labra Gayo, and Eric G. Prud'hommeau. Semantics and Validation of Shapes Schemas for RDF. *arXiv preprint arXiv:*, pages 1–35, 2014.

[BKE+14] Stefan Biffl, Marcos Kalinowski, Fajar J. Ekaputra, Estefanía Serral, and Dietmar Winkler. Building Empirical Software Engineering Bodies of Knowledge with Systematic Knowledge Engineering. In *International Conference on Software Engineering and Knowledge Engineering (SEKE)*, 2014.

[BKR+14] Stefan Biffl, Marcos Kalinowski, Rick Rabiser, Fajar J. Ekaputra, and Dietmar Winkler. Systematic Knowledge Engineering: Building Bodies of Knowledge from Published Research. *International Journal of Software Engineering and Knowledge Engineering*, 24(10):1533–1571, dec 2014.

[BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, may 2001.

[BLW16] Stefan Biffl, Arndt Lüder, and Dietmar Winkler. Multi-Disciplinary Engineering for Industrie 4.0: Semantic Challenges and Needs. In *Semantic Web Technologies for Intelligent Engineering Applications*, pages 17–51. Springer International Publishing, Cham, 2016.

[BÖF+10] C. Brecher, D. Özdemir, J. Feng, W. Herfs, K. Fayzullin, M. Hamadou, and A.W. W Müller. Integration of Software Tools with Heterogeneous Data Structures in Production Plant Lifecycles. In *10th IFAC Workshop on Intelligent Manufacturing Systems*, volume 43, pages 48–53, 2010.

[BS16] Stefan Biffl and Marta Sabou. *Semantic Web Technologies for Intelligent Engineering Applications.* Springer, Cham, 2016.

[BtHVH14] Thomas Bauernhansl, Michael ten Hompel, and Birgit Vogel-Heuser. *Industrie 4.0 in Produktion, Automatisierung und Logistik: Anwendung, Technologien, Migration.* Springer-Verlag, Wiesbaden, 2014.

[CCKE$^+$16] Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. Ontop: Answering SPARQL queries over relational databases. *Semantic Web*, 8(3):471–487, dec 2016.

[CO15] International Electrotechnical Commission and Others. IEC 62714 - Engineering data exchange format for use in industrial automation systems engineering-AutomationML, 2015.

[CWL14] Richard Cyganiak, David Wood, and Markus Lanthaler. RDF 1.1 Concepts and Abstract Syntax. *W3C Recommendation 25 February 2014*, (February):263–270, 2014.

[DHI12] AnHai Doan, Alon Halevy, and Zachary Ives. *Principles of Data Integration.* Elsevier, 2012.

[DK10] H. Dibowski and K. Kabitzsch. Ontology-based Device Descriptions and triple store based device repository for automation devices. In *15th IEEE Conference on Emerging Technologies & Factory Automation (ETFA 2010)*, pages 1–9. IEEE, sep 2010.

[DK11] Henrik Dibowski and Klaus Kabitzsch. Ontology-Based Device Descriptions and Device Repository for Building Automation Devices. *EURASIP Journal on Embedded Systems*, 2011(October 2010):1–17, 2011.

[DSC$^+$14] Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, Rik Van de Walle, and Rik Van De Walle. RML: A generic language for integrated RDF mappings of heterogeneous data. In *Linked Data on the Web Workshop (LDOW 2014)*, volume 1184, 2014.

[DVYP14] Victor Dubinin, Valeriy Vyatkin, Chen-Wei Yang, and Cheng Pang. Automatic generation of automation applications based on ontology transformations. In *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, pages 1–4. IEEE, sep 2014.

[EFO$^+$15] Kemele M. Endris, Sidra Faisal, Fabrizio Orlandi, Sören Auer, and Simon Scerri. Interest-Based RDF Update Propagation. In *In Proceedings of the 14th International Semantic Web Conference*, volume 9366, pages 513–529, 2015.

[Eka15] Fajar J. Ekaputra. Ontology change in ontology-based information integration systems. In *12th European Semantic Web Conference, ESWC 2015*, volume 9088, pages 711–720. Springer International Publishing, 2015.

[Eka16] Fajar J. Ekaputra. Knowledge Change Management and Analysis in Engineering. In Stefan Biffl and Marta Sabou, editors, *Semantic Web Technologies for Intelligent Engineering Applications*, chapter 7, pages 159–178. Springer International Publishing, Cham, 2016.

[EL16] Fajar J. Ekaputra and Xiashuo Lin. SHACL4P: SHACL constraints validation within Protégé ontology editor. In *Proceedings of 2016 International Conference on Data and Software Engineering, ICoDSE 2016*, 2016.

[ESS+17] Fajar J. Ekaputra, Marta Sabou, Estefanía Serral, Elmar Kiesling, and Stefan Biffl. Ontology-Based Data Integration in Multi-Disciplinary Engineering Environments: A Review. *Open Journal of Information Systems (OJIS)*, 4(1):1–26, 2017.

[ESSB14] Fajar J. Ekaputra, Marta Sabou, Estefanía Serral, and Stefan Biffl. Supporting Information Sharing for Reuse and Analysis of Scientific Research Publication Data. In *Proceedings of the 4th Workshop on Semantic Publishing (SePublica 2014)*, volume 1155. CEUR-WS, 2014.

[ESSB15a] Fajar J. Ekaputra, Marta Sabou, Estefanía Serral, and Stefan Biffl. Collaborative Exchange of Systematic Literature Review Results. In *Proceedings of the 24th International Conference on World Wide Web - WWW '15 Companion*, pages 1055–1056, New York, New York, USA, 2015. ACM, ACM Press.

[ESSB15b] Fajar J. Ekaputra, Estefanía Serral, Marta Sabou, and Stefan Biffl. Knowledge Change Management and Analysis for Multi-Disciplinary Engineering Environments. In *Joint Proceedings of the Posters and Demos Track of 11th International Conference on Semantic Systems - SEMANTiCS2015 and 1st Workshop on Data Science: Methods, Technology and Applications (DSci15)*, volume 1481, pages 13–17, 2015.

[ESSB16] Fajar J. Ekaputra, Marta Sabou, Estefanía Serral, and Stefan Biffl. Knowledge change management and analysis during the engineering of cyber physical production systems: A use case of hydro power plants. In *Proceedings of the 12th International Conference on Semantic Systems - SEMANTiCS 2016*, volume 13-14-Sept, pages 105–112, New York, New York, USA, 2016. ACM, ACM Press.

[ESWB13] Fajar J. Ekaputra, Estefanía Serral, Dietmar Winkler, and Stefan Biffl. An analysis framework for ontology querying tools. In *Proceedings of the 9th International Conference on Semantic Systems - I-SEMANTICS '13*, page 1, New York, New York, USA, 2013. ACM Press.

[ESWB14] Fajar J. Ekaputra, Estefanía Serral, Dietmar Winkler, and Stefan Biffl. A semantic framework for data integration and communication in project consortia. In *Proceedings of 2014 International Conference on Data and Software Engineering, ICODSE 2014*, pages 1–6. IEEE, IEEE, nov 2014.

[Eur16] European Parliament. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, 2016.

[FBW$^+$13] Alexander Fay, Stefan Biffl, Dietmar Winkler, Rainer Drath, and Mike Barth. A method to evaluate the openness of automation tools for increased interoperability. *IECON Proceedings (Industrial Electronics Conference)*, pages 6844–6849, 2013.

[FDES98] D Fensel, S Decker, M Erdmann, and R Studer. {O}ntobroker or {H}ow to enable intelligent access to the {WWW}. *11th Knowledge Acquisition for Knowledge-Based Systems Workshop (KAW'98)*, 1998.

[FES$^+$16] Sidra Faisal, Kemele M. Endris, Saeedeh Shekarpour, Sören Auer, and Maria-Esther Vidal. Co-evolution of RDF Datasets. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9671, pages 225–243, 2016.

[FHK$^+$15] Stefan Feldmann, Sebastian J.I. Herzig, Konstantin Kernschmidt, Thomas Wolfenstetter, Daniel Kammerl, Ahsan Qamar, Udo Lindemann, Helmut Krcmar, Christiaan J.J. Paredis, and Birgit Vogel-Heuser. Towards Effective Management of Inconsistencies in Model-Based Engineering of Automated Production Systems. In *IFAC-PapersOnLine*, volume 48, pages 916–923, 2015.

[FMK$^+$08] Giorgos Flouris, Dimitris Manakanatas, Haridimos Kondylakis, Dimitris Plexousakis, and Grigoris Antoniou. Ontology change: Classification and survey. *The Knowledge Engineering Review*, 23(2):117–152, 2008.

[FPA$^+$16] Marvin Frommhold, Rubén Navarro Piris, Natanael Arndt, Sebastian Tramp, Niklas Petersen, and Michael Martin. Towards Versioning of Arbitrary RDF Data. In *Proceedings of the 12th International Conference on Semantic Systems - SEMANTiCS 2016*, pages 33–40, New York, New York, USA, 2016. ACM Press.

[FUPK16] Javier D. Fernández, Jürgen Umbrich, Axel Polleres, and Magnus Knuth. Evaluating Query and Storage Strategies for RDF Archives. *Proceedings of the 12th International Conference on Semantic Systems - SEMANTiCS 2016*, pages 41–48, 2016.

[Gag07] Michel Gagnon. Ontology-based integration of data sources. In *2007 10th International Conference on Information Fusion*, pages 1–8. IEEE, jul 2007.

[GAP+12] Antonio Giovannini, Alexis Aubry, Hervé Panetto, Michele Dassisti, and Hind El Haouzi. Knowledge-Based System for Manufacturing Sustainability. *IFAC Proceedings Volumes*, 45(6):1333–1338, may 2012.

[GB14] Ramanathan Guha and Dan Brickley. RDF Schema 1.1. {W3C} Recommendation February 2014, W3C, 2014.

[GF12] M Goring and A Fay. Modeling Change and Structural Dependencies of Automation systems. In *IEEE 17th Conference on Emerging Technologies & Factory Automation (ETFA).*, pages 1–8, 2012.

[GGL+14] Alasdair J.G. Gray, Paul Groth, Antonis Loizou, Sune Askjaer, Christian Brenninkmeijer, Kees Burger, Christine Chichester, Chris T. Evelo, Carole Goble, Lee Harland, Steve Pettifer, Mark Thompson, Andra Waagmeester, and Antony J. Williams. Applying linked data approaches to pharmacology: Architectural decisions and implementation. *Semantic Web*, 5(2):101–113, 2014.

[GGMO03] Aldo Gangemi, Nicola Guarino, Claudio Masolo, and Alessandro Oltramari. Sweetening WORDNET with DOLCE. *AI magazine*, 24(3):13, 2003.

[GHJ+15] O. Gräser, L. Hundt, M. John, G. Lobermeier, A. Lüder, S. Mülhens, N. Ondracek, M. Thron, and J. Schmelter. AutomationML and eCl@ss integration, 2015.

[GHU14] M Graube, S Hensel, and L Urbas. R43ples: Revisions for Triples. In *Proceedings of the 1st Workshop on Linked Data Quality co-located with 10th International Conference on Semantic Systems (SEMANTiCS 2014)*, 2014.

[GHU16] Markus Graube, Stephan Hensel, and Leon Urbas. Open Semantic Revision Control with R43ples. In *Proceedings of the 12th International Conference on Semantic Systems - SEMANTiCS 2016*, pages 49–56, New York, New York, USA, 2016. ACM Press.

[GM07] Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using wikipedia-based explicit semantic analysis. *IJCAI International Joint Conference on Artificial Intelligence*, pages 1606–1611, 2007.

[GPS10] G Gröner, FS Parreiras, and S Staab. Semantic Recognition of Ontology Refactoring. In *9th International Semantic Web Conference (ISWC 2010)*, pages 273–288, 2010.

[Gru93] Thomas R Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, jun 1993.

[GUH16] Markus Graube, Leon Urbas, and Jan Hladik. Integrating industrial middleware in Linked Data collaboration networks. In *Proceedings of 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 2016-Novem, pages 1–8. IEEE, sep 2016.

[GZUH13] Markus Graube, Jens Ziegler, Leon Urbas, and Jan Hladik. Linked data as enabler for mobile applications for complex tasks in industrial settings. In *Proceedings of the 18th IEEE Conference on Emerging Technologies & Factory Automation (ETFA)*, pages 1–8. IEEE, sep 2013.

[HA99] B. Huberman and Lada Adamic. Internet: Growth dynamics of the World-Wide Web. *Nature*, 401(6749):131–131, 1999.

[Hal05] Alon Halevy. Why your data won't mix. *Queue*, 3:50–58, 2005.

[Hep08] Martin Hepp. GoodRelations: An Ontology for Describing Products and Services Offers on the Web. In *Knowledge Engineering: Practice and Patterns*, volume 5268 LNAI, pages 329–346. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[HKR09] Pascal Hitzler, Markus Krötzsch, and Sebastian Rudolph. *Foundation of Semantic Web Technologies.* 2009.

[HPsB$^+$04] Ian Horrocks, Peter F Patel-schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL : A Semantic Web Rule Language Combining OWL and RuleML. {W3C} Submission May 2004, W3C, 2004.

[HRTM13] Matthew Horridge, Timothy Redmond, Tania Tudorache, and Mark Musen. Binary OWL. In *OWL Experiences and Directions Workshop (OWLED 2013)*, 2013.

[HS13] Steve Harris and Andy Seaborne. SPARQL 1.1 Overview. {W3C} recommendation, W3C, 2013.

[HVKE16] Christian Hennig, Alexander Viehl, Benedikt Kämpgen, and Harald Eisenmann. Ontology-Based Design of Space Systems. In Paul Groth, Elena Simperl, Alasdair Gray, Marta Sabou, Markus Krötzsch, Freddy Lecue, Fabian Flöck, and Yolanda Gil, editors, *The Semantic Web–ISWC 2016*, volume 9981 of *Lecture Notes in Computer Science*, pages 33–48, Cham, 2016. Springer International Publishing.

[HW03] Gregor Hohpe and Bobby Woolf. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions.* AddisonWesley Longman Publishing, 2003.

[Hyv12] Eero Hyvönen. *Publishing and Using Cultural Heritage Linked Data on the Semantic Web*, volume 2. Morgan Claypool, oct 2012.

[IEC03] IEC. IEC 62264 - International Standard for Enterprise-Control System Integration, 2003.

[IEC08] IEC International Electrotechnical Commission. IEC 62424:2008, Representation of process control engineering – Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools, 2008.

[IEC09] IEC. IEC 81346 - Industrial systems, installations and equipment and industrial products, 2009.

[ISO14] ISO. STEP ISO 10303 - Standard for the Exchange of Product Model Data, 2014.

[IY16] Muhammad Imran and R.I.M. Young. Reference ontologies for interoperability across multiple assembly systems. *International Journal of Production Research*, 54(18):5381–5403, sep 2016.

[JRMGC+14] J. Juarez, J. A. Rodriguez-Mondejar, R. Garcia-Castro, J. Juárez, J.A. Rodríguez-Mondéjar, and R. García-Castro. An ontology-driven communication architecture for spontaneous interoperability in Home Automation systems. In *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, pages 1–4. IEEE, sep 2014.

[KBM+11] Sebastian Köhler, Sebastian Bauer, Chris J. Mungall, Gabriele Carletti, Cynthia L. Smith, Paul Schofield, Georgios V. Gkoutos, and Peter N. Robinson. Improving ontologies by automatic reasoning and evaluation of logical definitions. *BMC Bioinformatics*, 12(1):418, 2011.

[KC07] Barbara A Kitchenham and Stuart Charters. Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE-2007-01, Keele University and University of Durham, 2007.

[KE16] Olga Kovalenko and Jérôme Euzenat. Semantic Matching of Engineering Data Structures. In *Semantic Web Technologies for Intelligent Engineering Applications*, pages 137–157. Springer, 2016.

[KHI11] Holger Knublauch, James A Hendler, and Kingsley Idehen. SPIN - Overview and Motivation. {W3C} submission, W3C, 2011.

[KJRZ+13] Evgeny Kharlamov, Ernesto Jiménez-Ruiz, Dmitriy Zheleznyakov, Dimitris Bilidas, Martin Giese, Peter Haase, Ian Horrocks, Herald Kllapi, Manolis Koubarakis, Özgür Özçep, Mariano Rodríguez-Muro, Riccardo Rosati, Michael Schmidt, Rudolf Schlatte, Ahmet Soylu, and Arild Waaler. Optique: Towards OBDA Systems for Industry. In *The Semantic Web:*

ESWC 2013 Satellite Events, volume 7955 LNCS, pages 125–140. Springer-Verlag, 2013.

[KKKK06] G Kappel, E Kapsammer, H Kargl, and G Kramler. *Lifting metamodels to ontologies: A step to the semantic integration of modeling languages.* Springer Berlin Heidelberg, 2006.

[KKT14] Timo Kehrer, Udo Kelter, and Gabriele Taentzer. Propagation of Software Model Changes in the Context of Industrial Plant Automation. *at-Automatisierungstechnik*, 62(11):803–814, 2014.

[KL89] Michael Kifer and Georg Lausen. F-logic: a higher-order language for reasoning about objects, inheritance, and scheme. In *Proceedings of the 1989 ACM SIGMOD international conference on Management of data - SIGMOD '89*, volume 18, pages 134–146, New York, New York, USA, 1989. ACM Press.

[Kle04] Michel Klein. *Change Management for Distributed Ontologies.* Phd thesis, Vrije Universiteit Amsterdam, 2004.

[Knu04] Holger Knublauch. Ontology-Driven Software Development in the Context of the Semantic Web: An Example Scenario with Protege/OWL. *1st International Workshop on the ModelDriven Semantic Web MDSW2004*, 2004.

[KSÖ+14] Evgeny Kharlamov, Nina Solomakhina, Özgür Lütfü Özçep, Dmitriy Zheleznyakov, Thomas Hubauer, Steffen Lamparter, Mikhail Roshchin, Ahmet Soylu, and Stuart Watson. How Semantic Technologies Can Enhance Data Access at Siemens Energy. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandečić, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble, editors, *13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, volume 8796 of *Lecture Notes in Computer Science*, pages 601–619, Cham, 2014. Springer International Publishing.

[KSS+14] Olga Kovalenko, Estefanía Serral, Marta Sabou, Fajar J. Ekaputra, Dietmar Winkler, and Stefan Biffl. Automating Cross-Disciplinary Defect Detection in Multi-disciplinary Engineering Environments. In *Knowledge Engineering and Knowledge Management: 19th International Conference, EKAW 2014, Linköping, Sweden, November 24-28, 2014. Proceedings*, volume 8876, pages 238–249, 2014.

[KTS13] Botond Kádár, Walter Terkaj, and Marco Sacco. Semantic Virtual Factory supporting interoperable modelling and evaluation of production systems. *CIRP Annals - Manufacturing Technology*, 62(1):443–446, 2013.

[Kus93] A Kusiak. *Concurrent engineering: Automation, tools, and techniques.* WileyInterscience, 1993.

[KVH13] Konstantin Kernschmidt and Birgit Vogel-Heuser. An interdisciplinary SysML based modeling approach for analyzing change influences in production plants to support the engineering. In *IEEE International Conference on Automation Science and Engineering*, pages 1113–1118, 2013.

[KWA$^+$14] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. Test-driven evaluation of linked data quality. In *Proceedings of the 23rd international conference on World Wide Web*, pages 747–758, 2014.

[KWK$^+$14] Olga Kovalenko, Dietmar Winkler, Marcos Kalinowski, Estefania Serral, and Stefan Biffl. Engineering Process Improvement in Heterogeneous Multi-disciplinary Environments with Defect Causal Analysis. In *Communications in Computer and Information Science*, volume 425, pages 73–85, 2014.

[KWS$^+$15] Olga Kovalenko, Manuel Wimmer, Marta Sabou, Arndt Lüder, Fajar J. Ekaputra, and Stefan Biffl. Modeling AutomationML: Semantic Web Technologies vs. Model-Driven Engineering. In *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*, pages 1–4. IEEE, 2015.

[LH07] H.K. K Lin and J.A. A Harding. A manufacturing system engineering ontology model on the semantic web for inter-enterprise collaboration. *Computers in Industry*, 58(5):428–437, jun 2007.

[LK07] Jae Yeol Lee and Kwangsoo Kim. A distributed product development architecture for engineering collaborations across ubiquitous virtual enterprises. *The International Journal of Advanced Manufacturing Technology*, 33(1-2):59–70, may 2007.

[LLP$^+$15] Yongxin Liao, Mario Lezoche, Hervé Panetto, Nacer Boudjlida, and Eduardo Rocha Loures. Semantic annotation for knowledge explicitation in a product lifecycle management context: A survey. *Computers in Industry*, 71(C):24–34, aug 2015.

[LLVH13] Christoph Legat, Steffen Lamparter, and Birgit Vogel-Heuser. Knowledge-Based Technologies for Future Factory Engineering and Control. In *Studies in Computational Intelligence*, volume 472, pages 355–374. Springer, 2013.

[LS17] Arndt Lüder and Nicole Schmidt. Challenges of Mechatronical Engineering of Production Systems: An Automation System Engineering View. In *Math for the Digital Factory*, pages 93–114. Springer, 2017.

[LSR96] Sean Luke, Lee Spector, and David Rager. Ontology-based Knowledge Discovery on the World-Wide Web. *Proceedings of the Workshop on Internet-based Information Systems at the 13th National Conference on Artificial Intelligence (AAAI-96)*, pages 96–102, 1996.

[MB10] Thomas Moser and Stefan Biffl. Semantic tool interoperability for engineering manufacturing systems. In *2010 IEEE 15th Conference on Emerging Technologies & Factory Automation (ETFA 2010)*, pages 1–8. IEEE, sep 2010.

[MB12] Thomas Moser and Stefan Biffl. Semantic Integration of Software and Systems Engineering Environments. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(1):38–50, jan 2012.

[MB15] Richard Mordinyi and Stefan Biffl. Versioning in Cyber-physical Production System Engineering – Best-Practice and Research Agenda. In *Proceedings of the 2015 IEEE/ACM 1st International Workshop on Software Engineering for Smart Cyber-Physical Systems*, SESCPS '15, pages 44–47, Piscataway, NJ, USA, 2015. IEEE Press.

[MDT$^+$17] G. E. Modoni, M. Doukas, W. Terkaj, M. Sacco, and D. Mourtzis. Enhancing factory data integration through the development of an ontology: from the reference models reuse to the semantic conversion of the legacy models. *International Journal of Computer Integrated Manufacturing*, 30(10):1043–1059, oct 2017.

[MES$^+$17] Juergen Musil, Fajar J. Ekaputra, Marta Sabou, Tudor Ionescu, Daniel Schall, Angelika Musil, and Stefan Biffl. Continuous Architectural Knowledge Integration: Making Heterogeneous Architectural Knowledge Available in Large-Scale Organizations. In *Proceedings - 2017 IEEE International Conference on Software Architecture, ICSA 2017*, pages 189–192, 2017.

[MMMB10] Thomas Moser, Richard Mordinyi, A Mikula, and Stefan Biffl. Efficient Integration of Complex Information Systems in the ATM Domain with Explicit Expert Knowledge Models. In *Complex Intelligent Systems and Their Applications*, pages 1–19. Springer-Verlag, New York, 2010.

[MMW$^+$11] Thomas Moser, Richard Mordinyi, Dietmar Winkler, Martin Melik-Merkumians, and Stefan Biffl. Efficient automation systems engineering process support based on semantic integration of engineering knowledge. In Zoubir Mammeri and Richard Zurawski, editors, *ETFA2011*, pages 1–8. IEEE, sep 2011.

[Mos09] Thomas Moser. *Semantic Integration of Engineering Environments Using an Engineering Knowledge Base.* PhD thesis, TU Wien, 2009.

[Mos16] Thomas Moser. The Engineering Knowledge Base Approach. In Stefan Biffl and Marta Sabou, editors, *Semantic Web for Intelligent Engineering Applications*, chapter 4. Springer, Vienna, Austria, 2016.

[MP02] Peter McBrien and Alexandra Poulovassilis. Schema evolution in heterogeneous database architectures, a schema transformation approach. In *Advanced Information Systems Engineering*, pages 484–499. Springer, 2002.

[MSE16] Richard Mordinyi, Estefanía Serral, and Fajar J. Ekaputra. Semantic Data Integration: Tools and Architectures. In *Semantic Web Technologies for Intelligent Engineering Applications*, pages 181–217. Springer, 2016.

[MSWB14] Richard Mordinyi, Estefania Serral, Dietmar Winkler, and Stefan Biffl. Evaluating software architectures using ontologies for storing and versioning of engineering data in heterogeneous systems engineering environments. In *19th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2014*, pages 1–10, Vienna, Austria, sep 2014. Vienna University of Technology, IEEE.

[MV11] B Meyers and H Vangheluwe. A Framework for Evolution of Modelling Languages. *Science of Computer Programming*, 76(12):1223–1246, 2011.

[MWE$^+$16] Richard Mordinyi, Dietmar Winkler, Fajar J. Ekaputra, Manuel Wimmer, and Stefan Biffl. Investigating model slicing capabilities on integrated plant models with AutomationML. In *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, volume 2016-Novem, 2016.

[MWM09] Jan Morbach, Andreas Wiesner, and Wolfgang Marquardt. OntoCAPE—A (re)usable ontology for computer-aided process engineering. *Computers & Chemical Engineering*, 33(10):1546–1556, oct 2009.

[MWZ$^+$10] Thomas Moser, Florian Waltersdorfer, Alois Zoitl, Stefan Biffl, and Thomas Moser. Version Management and Conflict Detection Across Heterogeneous Engineering Data Models. In *Proceedings of the 8th IEEE International Conference on Industrial Informatics (INDIN 2010)*, pages 928–935, 2010.

[NBJ11] A Newen, A Bartels, and E.-M. Jung. *Knowledge and representation.* CSLI Publications, 2011.

[NCLM06] NF Noy, Abhita Chugh, William Liu, and MA Musen. A Framework for Ontology Evolution in Collaborative Environments. In *Proceedings of the 5th international conference on The Semantic Web (ISWC 2006)*, pages 544–558, 2006.

[NDH05] Natalya F. Noy, AnHai Doan, and Alon Y. Halevy. Semantic Integration, mar 2005.

[NEB17] Petr Novák, Fajar J. Ekaputra, and Stefan Biffl. Generation of Simulation Models in MATLAB-Simulink Based on AutomationML Plant Description. In *IFAC - Papers Online*, pages 7613–7620, 2017.

[NFGT16] Elisa Negri, Luca Fumagalli, Marco Garetti, and Letizia Tanca. Requirements and languages for the semantic representation of manufacturing systems. *Computers in Industry*, 81:55–66, 2016.

[NGS12] Sathish Natarajan, Kaushik Ghosh, and Rajagopalan Srinivasan. An ontology for distributed process supervision of large-scale chemical plants. *Computers & Chemical Engineering*, 46:124–140, nov 2012.

[NGS$^+$17] Bernd Neumayr, Eduard Gringinger, Christoph G Schuetz, Michael Schrefl, Scott Wilson, and Audun Vennesland. Semantic data containers for realizing the full potential of system wide information management. In *2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC)*, pages 1–10. IEEE, sep 2017.

[NK04] NF Natalya F Noy and Michel Klein. Ontology evolution: Not the same as schema evolution. *Knowledge and information systems*, 6(July 2002):428–440, 2004.

[NM02] NF Noy and MA Musen. Promptdiff: A fixed-point algorithm for comparing ontology versions. *AAAI/IAAI*, 2002.

[Noy04] Natalya F Noy. Semantic integration: a survey of ontology-based approaches. *SIGMOD Rec.*, 33:65–70 ST – Semantic integration: a survey of onto, 2004.

[NP01] Ian Niles and Adam Pease. Towards a standard upper ontology. In *Proceedings of the international conference on Formal Ontology in Information Systems - FOIS '01*, volume 2001, pages 2–9, New York, New York, USA, 2001. ACM Press.

[NS13] Petr Novak and Radek Sindelar. Ontology-based industrial plant description supporting simulation model design and maintenance. In *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*, pages 6866–6871. IEEE, nov 2013.

[NS14a] Sathish Natarajan and Rajagopalan Srinivasan. Implementation of multi agents based system for process supervision in large-scale chemical plants. *Computers and Chemical Engineering*, 60:182–196, 2014.

[NŠ14b] Petr Novák and Radek Šindelář. Component-Based Design of Simulation Models Utilizing Bond-Graph Theory. In *IFAC Proceedings Volumes*, volume 47, pages 9229–9234, 2014.

[Obe14] Daniel Oberle. Ontologies and Reasoning in Enterprise Service Ecosystems. *Informatik-Spektrum*, 37(4):318–328, aug 2014.

[PDT12] H. Panetto, M. Dassisti, and A. Tursi. ONTO-PDM: Product-driven ONTOlogy for Product Data Management interoperability within manufacturing process environment. *Advanced Engineering Informatics*, 26(2):334–348, apr 2012.

[PFF09a] V Papavassiliou, G Flouris, and I Fundulaki. Formalizing High-Level Change Detection for RDF/S KBs. Technical report, 2009.

[PFF09b] V Papavassiliou, G Flouris, and I Fundulaki. On Detecting High-Level Changes in RDF/S KBs. In *8th International Semantic Web Conference (ISWC 2009)*, pages 473–488, 2009.

[PHCGP09] R Palma, P Haase, O Corcho, and A Gómez-Pérez. Change representation for OWL 2 ontologies. In *6th International Workshop on OWL: Experiences and Directions (OWLED 2009)*, 2009.

[RCW13] K Ridgway, CW Clegg, and DJ Williams. The Factory of the Future. Technical report, 2013.

[RN11] T Redmond and N Noy. Computing the Changes Between Ontologies. In *Joint Workshop on Knowledge Evolution and Ontology Dynamics*, pages 1–14, 2011.

[Rod95] John F Roddick. A Survey of Schema Versioning Issues for Database Systems. *Information and Software Technology*, 37(7):383–393, 1995.

[RSDT08] T Redmond, M Smith, N Drummond, and T Tudorache. Managing Change: An Ontology Version Control System. *OWLED*, 2008.

[Sab16] Marta Sabou. An Introduction to Semantic Web Technologies. In *Semantic Web Technologies for Intelligent Engineering Applications*, pages 53–81. Springer, 2016.

[SBF98] Rudi Studer, V.Richard Benjamins, and Dieter Fensel. Knowledge engineering: Principles and methods. *Data & Knowledge Engineering*, 25(1-2):161–197, mar 1998.

[SBLH06] Nigel Shadbolt, Tim Berners-Lee, and Wendy Hall. The Semantic Web Revisited. *IEEE Intelligent Systems*, 21(3):96–101, may 2006.

[SEB17] Marta Sabou, Fajar J. Ekaputra, and Stefan Biffl. Semantic Web Technologies for Data Integration in Multi-Disciplinary Engineering. In Stefan Biffl, Detlef Gerhard, and Arndt Lüder, editors, *Multi-Disciplinary Engineering for Cyber-Physical Production Systems*, pages 301–329. Springer, 2017.

[SEI$^+$18] Marta Sabou, Fajar J Ekaputra, Tudor Ionescu, Juergen Musil, Daniel Schall, Kevin Haller, Armin Friedl, and Stefan Biffl. Exploring Enterprise Knowledge Graphs : a Use Case in Software Engineering. In *Extended Semantic Web Conference (ESWC)*, 2018.

[SEKB16] Marta Sabou, Fajar J. Ekaputra, Olga Kovalenko, and Stefan Biffl. Supporting the engineering of cyber-physical production systems with the AutomationML analyzer. In *2016 1st International Workshop on Cyber-Physical Production Systems (CPPS)*, pages 1–8. IEEE, apr 2016.

[SHL$^+$14] Nina Solomakhina, Thomas Hubauer, Steffen Lamparter, Mikhail Roshchin, and Stephan Grimm. Extending statistical data quality improvement with explicit domain models. In *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, pages 720–725. IEEE, jul 2014.

[SK08] Andy Schürr and Felix Klar. 15 Years of Triple Graph Grammars. In *Graph Transformations*, volume 5214, pages 411–425, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[SKEB16] Marta Sabou, Olga Kovalenko, Fajar J. Ekaputra, and Stefan Biffl. Semantic Web Solutions in Engineering. In Stefan Biffl and Marta Sabou, editors, *Semantic Web Technologies for Intelligent Engineering Applications*, chapter 11, pages 281–296. Springer International Publishing, Cham, 2016.

[SLSB14] Nicole Schmidt, Arndt Luder, Heinrich Steininger, and Stefan Biffl. Analyzing requirements on software tools according to the functional engineering phase in the technical systems engineering process. In *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, pages 1–8. IEEE, sep 2014.

[SM02] L Stojanovic and B Motik. Ontology evolution within ontology editors. *Proceedings of the OntoWeb-SIG3 Workshop*, 2002.

[SMLH10] C Stadler, M Martin, J Lehmann, and S Hellmann. Update Strategies for DBpedia Live. In *Proceedings of the Sixth Workshop on Scripting and Development for the Semantic Web (SFSW 2010)*, 2010.

[SPFW13] Andy Seaborne, Axel Polleres, Lee Feigenbaum, and Gregory Todd Williams. SPARQL 1.1 Federated Query. {W3C} Recommendation March, W3C, 2013.

[SRD$^+$13] Selver Softic, Manfred Rosenberger, Andrea Denger, Johannes Fritz, and Alexander Stocker. Semantically based visual tracking of engineering tasks in automotive product lifecycle. In *Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies - i-Know '13*, pages 1–4, New York, New York, USA, 2013. ACM Press.

[SRFF11] Martin Strube, Stefan Runde, Helmut Figalist, and Alexander Fay. Risk minimization in modernization projects of plant automation - A knowledge-based approach by means of semantic web technologies. In *Proceedings of the 2011 IEEE Emerging Technology and Factory Automation*, pages 1–8. IEEE, sep 2011.

[Sto04] L Stojanovic. *Methods and tools for ontology evolution.* PhD thesis, University of Kalsruhe, 2004.

[TELW14] G Taentzer, C Ermel, P Langer, and M Wimmer. A Fundamental Approach to Model Versioning Based on Graph Modifications: From Theory to Implementation. *Software & Systems Modeling*, 13(1):239–272, 2014.

[TTU15] Walter Terkaj, Tullio Tolio, and Marcello Urgo. A virtual factory approach for in situ simulation to support production and maintenance planning. *CIRP Annals - Manufacturing Technology*, 64(1):451–454, 2015.

[TU12] Walter Terkaj and Marcello Urgo. Virtual factory data model to support performance evaluation of production systems. In *Proceedings of OSEMA 2012 workshop, 7th International conference on Formal Ontology in Information Systems, Graz, Austria*, pages 44–58, 2012.

[TU14] Walter Terkaj and Marcello Urgo. Ontology-based modeling of production systems for design and performance evaluation. In *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, pages 748–753. IEEE, jul 2014.

[VBGK09] Julius Volz, Christian Bizer, Martin Gaedke, and Georgi Kobilarov. Silk - A Link Discovery Framework for the Web of Data. In *Linked Data on the Web Workshop (LDOW 2009)*, volume 538, Chichester, UK, sep 2009. John Wiley & Sons, Ltd.

[VCV$^+$13] Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Sam Coppens, Erik Mannens, and Rik Van de Walle. R&Wbase: Git for Triples. In *Linked Data on the Web Workshop (LDOW 2013)*, 2013.

[Ver09] Verein Deutscher Ingenieure. VDI Richtlinie 3695 - Engineering von Anlagen - Evaluieren und optimieren des Engineerings, 2009.

[Ver13] Verein Deutscher Ingenieure. VDI Richtlinie 3633 - Simulation von Logistik-, Materialfluss- und Produktionssystemen - Begriffe, 2013.

[VG06] M Völkel and T Groza. SemVersion: An RDF-based ontology versioning system. In *Proceedings of the IADIS international conference WWW/Internet*, page 44, 2006.

[VHLFR14] Birgit Vogel-Heuser, Christoph Legat, Jens Folmer, and Susanne Rösch. Challenges of Parallel Evolution in Production Automation Focusing on Requirements Specification and Fault Handling. *at-Automatisierungstechnik*, 62(11):758–770, 2014.

[VPT^{+}05] Denny Vrandecic, Sofia Pinto, Christoph Tempich, York Sure, J Davies, York Sure, and Denny Vrandecic. The DILIGENT knowledge processes. *Journal of ...*, 9(5):85–96, jan 2005.

[VSWV00] U Visser, Heiner Stuckenschmidt, H Wache, and T Vögele. Enabling technologies for interoperability. *Workshop on the 14th International Symposium of Computer Science for Environmental Protection*, pages 35–46, 2000.

[W3C12] W3C OWL Working Group. OWL 2 Web Ontology Language Document Overview. {W3C} recommendation, W3C, 2012.

[WB12] Dietmar Winkler and Stefan Biffl. Improving Quality Assurance in Automation Systems Development Projects. *Quality Assurance and Management*, pages 379–398, 2012.

[Wil13] Gregory Williams. SPARQL 1.1 Service Description. {W3C} recommendation, W3C, 2013.

[WLRW15] Tim Weilkiens, Jesko G. Lamm, Stephan Roth, and Markus Walker. *Model-Based System Architecture*. John Wiley & Sons, Inc, Hoboken, NJ, USA, sep 2015.

[WMM11a] Andreas Wiesner, Jan Morbach, and Wolfgang Marquardt. Information integration in chemical process engineering based on semantic technologies. *Computers & Chemical Engineering*, 35(4):692–708, apr 2011.

[WMM^{+}11b] Dietmar Winkler, Thomas Moser, Richard Mordinyi, Wikan Danar Sunindyo, and Stefan Biffl. Engineering Object Change Management Process Observation in Distributed Automation Systems Projects. In *Proceedings of 18th European System & Software Process Improvement and Innovation (EuroSPI 2011)*, pages 1–12, 2011.

[WVV^{+}01] Holger Wache, Thomas Voegele, Ubbo Visser, Heiner Stuckenschmidt, Gerhard Schuster, Holger Neumann, and Sebastian Hübner. Ontology-based integration of information-a survey of existing approaches. In *IJCAI-01 Workshop on Ontologies and Information Sharing*, pages 108–117. Citeseer, 2001.

[Zab09] F Zablith. Evolva : a comprehensive approach to ontology evolution. *The Semantic Web: Research and Applications*, 2009.

[Zab11] F Zablith. *Harvesting Online Ontologies for Ontology Evolution*. Phd thesis, The Open University, UK, 2011.

[ZAD$^+$15] Fouad Zablith, Grigoris Antoniou, Mathieu D'Aquin, Giorgos Flouris, Haridimos Kondylakis, Enrico Motta, Dimitris Plexousakis, and Marta Sabou. Ontology evolution: a process-centric survey. *The Knowledge Engineering Review*, 30(01):45–75, 2015.

[ZRM$^+$15] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Sören Auer. Quality assessment for Linked Data: A Survey. *Semantic Web*, 7(1):63–93, mar 2015.