Chapter 11
Semantic Web Solutions in Engineering

Marta Sabou, Olga Kovalenko, Fajar Juang Ekaputra and Stefan Biffl

Abstract The *Industrie 4.0* vision highlights the need for more flexible and adaptable production systems. This requires making the process of engineering production systems faster and intends to lead to higher quality, but also more complex plants. A key issue in improving engineering processes in this direction is providing mechanisms that can efficiently and intelligently handle large-scale and heterogeneous engineering data sets thus shortening engineering processes while ensuring a higher quality of the engineered system, for example, by enabling improved cross-disciplinary defect detection mechanisms. Semantic Web technologies (SWTs) have been widely used for the development of a range of Intelligent Engineering Applications (IEAs) that exhibit an intelligent behavior when processing large and heterogeneous data sets. This chapter identifies key technical tasks performed by IEAs, provides example IEAs and discusses the connection between Semantic Web capabilities and IEA tasks.

Keywords Intelligent engineering applications • Model integration • Model consistency management • Flexible comparison

M. Sabou (✉) · O. Kovalenko · F.J. Ekaputra · S. Biffl
Institute of Software Technology and Interactive Systems, CDL-Flex,
Vienna University of Technology, Vienna, Austria
e-mail: Marta.Sabou@ifs.tuwien.ac.at

O. Kovalenko
e-mail: Olga.Kovalenko@tuwien.ac.at

F.J. Ekaputra
e-mail: Fajar.Ekaputra@tuwien.ac.at

S. Biffl
e-mail: Stefan.Biffl@tuwien.ac.at

S. Biffl and M. Sabou (eds.), *Semantic Web Technologies for Intelligent Engineering Applications*, DOI 10.1007/978-3-319-41490-4_11

11.1 Introduction

The *Industrie 4.0* trend (Bauernhansl et al. 2014) envisions increased flexibility of production systems as well as an improved vertical and horizontal integration of production system components (Lüder and Schmidt 2016) as also explained in Chap. 1. Essential elements for realizing these goals are: self-ware and self-adaptable production components as well as flexible production systems relying on adaptation and plug-and-work capabilities. For example, future factories should be able to flexibly respond to changing business conditions and to handle disruptions in the production process (Legat et al. 2013). To achieve such capabilities, it is essential to modernize factory engineering processes in particular by ensuring an integrated exchange of engineering-component information during the engineering design and run-time phases of the production system life cycle (Legat et al. 2013; Lüder and Schmidt 2016).

The need for more flexible and adaptable production systems, which are redesigned more often than before (Bauernhansl et al. 2014), requires, on its turn, that the process of engineering production systems becomes faster; likely leading to higher quality, more complex plants. However, the optimization of engineering processes is often hampered by their heterogeneous and collaborative nature. Heterogeneity is a key characteristic because a large and diverse set of stakeholders is involved in the process of engineering production systems, often bridging the boundaries of several organizations. Usually, the engineering of a production system, for example, a hydro power plant, involves: the production system owner, a main contractor, between 10 and 500 subcontractors, and up to 1,000 component vendors. These stakeholders span diverse engineering disciplines (including mechanical, electrical, and software engineering), make use of a diverse set of (engineering) tools and use terminologies with limited overlap. Indeed, Feldmann et al. (2015) identified heterogeneous and semantically overlapping models as a key characteristic and challenge of engineering settings.

Despite their heterogeneity, the involved stakeholders 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 (Schmidt et al. 2014).

The study of Lüder and Schmidt (2016) states that engineering of complex mechatronic objects, especially production systems, is increasingly driven by information models that enable representing different aspects of the produced system. This opens up the need for model-driven technologies, such as Semantic Web technologies (SWTs) where *ontologies* (Gruber 1993; Studer et al. 1998) are used as models (Berners-Lee et al. 2001; Shadbolt et al. 2006). Indeed, knowledge-based approaches in general have been observed to be particularly suitable to support the process of engineering production systems as well as to enable advanced functionalities of such systems (e.g., handling disturbances, adapting to new business requirements) (Legat et al. 2013). Knowledge-based systems support "(1) the explicit representation of knowledge in a domain of interest and (2) the exploitation of such knowledge through appropriate reasoning mechanisms in order to provide high-level problem solving performance" (Tasso and Arantes e Oliveira 1998). SWTs, explained in Chap. 3, 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. In such setting, SWTs focus on large-scale (i.e., Web-scale) data integration and intelligent reasoning-based methods to support advanced data analytics. Important data analytics tasks include project monitoring, defect detection, and control.

While the use of SWTs to create intelligent engineering applications (IEAs) has seen considerable uptake, there is a lack of understanding of:

- *Q1: What are the key technical tasks that should be solved by an IEA?* and
- *Q2: How are typical IEA tasks enabled by SWT capabilities?*

As an answer to Q1, clear requirements emerge from the area of mechatronic engineering for the *technical tasks* that should be solved by model-driven technologies such as the Semantic Web. Lüder and Schmidt (2016) identify a set of concrete technical tasks that are still challenging to perform in mechatronic engineering, which should be better supported in *Industrie 4.0* settings. They identify that techniques are needed for *model generation, model transformation, model integration,* and *model consistency managemen*t. These tasks are made more difficult by the fact that engineering models are created by engineers from different disciplines. In addition, to support adaptation and plug-and-work capabilities, approaches are needed that allow *flexible comparison*. Flexible comparison can occur, for example, during the selection of an appropriate mechatronic unit, where matchmaking is performed between the required functionalities and those offered by the mechatronic unit to be selected. We consider the tasks put forward by Lüder and Schmidt (2016) as good indicators for typical technical tasks that Semantic Web-based IEAs should solve. Approaches to model generation and model transformation that rely on SWTs are discussed at length in Chap. 5. In this chapter, we focus, in particular, on *model integration, model consistency management,* and *flexible comparison*.

To answer Q2 we perform the following two analysis tasks. First, for each of the three technical tasks we discuss in Sects. 11.2–11.4 a set of IEAs that achieve one of these tasks. The goal is to provide example applications and therefore the material is not meant as an extensive survey of the domain. For each application, we also discuss how SWT capabilities, presented in Chap. 3, support achieving the technical tasks at hand. As such we go beyond the initial version of this work published in (Sabou et al. 2015). We recall that the SWT capabilities identified in Chap. 3 are: (C1) formal and flexible semantic modeling; (C2) intelligent, Web-scale knowledge integration; (C3) browsing and exploration of distributed data sets; (C4) knowledge quality assurance with reasoning, and (C5) knowledge reuse. Section 11.5 concludes our analysis. Second, in Sect. 11.6 we provide an outlook to the remainder of this book, namely Chaps. 12, 13 and 14, and perform a similar analysis of which technical tasks are addressed and which SWTs are used in these three chapters.

11.2 Semantic Web Solutions for Model Integration

Model integration aims to bridge semantic gaps in engineering environments between project participants (and their tools), who use different local terminologies (Aldred et al. 2006; Hohpe and Woolf 2003; Moser et al. 2009; Moser et al. 2010), thus ultimately supporting the analysis, automation, and improvement of multi-disciplinary engineering processes. Semantic model integration is defined as solving problems originating from the intent to share information across disparate and semantically heterogeneous data (Halevy 2005). These problems include the matching of data schemas, the detection of duplicate entries, the reconciliation of inconsistencies, and the modeling of complex relations in different data sources (Noy et al. 2005). Noy (2004) identified three major dimensions of the application of ontologies for supporting semantic model 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 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; (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.

In the remainder of this section, we exemplify some approaches to SWT-based model integration and we discuss which SWT capabilities are used by each approach. Table 11.1 sums up the dependencies between SWT capabilities and the task of model integration for the examples given in this section.

Terkaj and Urgo (2014) present an ontology-based solution to achieve interoperability between systems and tools that may be developed according to different data structures and by employing heterogeneous technologies. Concretely, by leveraging the benefits of conceptual modeling *(C1)*, an ontology represented in the Web Ontology Language[1] (OWL) is used to integrate partial design information from the stakeholders with different competences and expertise (i.e., plant planner or PLC programmer). For the ontology representation, the authors extend the

[1]OWL Reference: https://www.w3.org/TR/owl2-overview/.

Table 11.1 SWT capabilities used by example approaches for model integration

C1: Semantic modeling	C2: Knowledge integration	C5: Knowledge reuse
(Grünwald et al. 2014; Terkaj and Urgo 2014)	(Grünwald et al. 2014; Kovalenko et al. 2013)	(Terkaj and Urgo 2014)

Virtual Factory Data Model (VFDM), a standard data model for representing factory objects related to production systems, resources, processes, and products (Terkaj and Urgo 2012). The provided infrastructure is tested in the context of supporting the design of production systems and generating simulations to check the performance and other aspects of the production system. For that, a simulation model is generated in a semiautomatic way from the obtained virtual representation of the designed production system. This allows faster verification of the current design (i.e., current model). The produced simulation is used to generate the 3D visualization of the dynamic behavior of the production system, based on which the various parameters of the current design of production system (e.g., performance) can be analyzed. The provided solution therefore facilitates the production system reconfiguration in the design phase. The created ontologies exploit the state-of-the-art technical standards and were designed in an application-oriented fashion in order to ensure their future reuse in similar scenarios (*C5)—knowledge reuse*.

Driven primarily by the context of project management in multidisciplinary engineering settings, the *Semantic Model Editor* (SMEd) provides an intuitive way to model and integrate ontologies describing engineering knowledge (Grünwald et al. 2014). Thanks to the formal representation in *conceptual modeling* (*C1*) of the integrated data, a set of SPARQL queries can be executed to extract information relevant for project management (e.g., the number of open features, the number of collaborations between project participants, and the project status). SWTs that support data integration in this tool are formal and conceptual modeling (*C1*) as well as knowledge integration through model mapping, *knowledge integration* (*C2*). An interesting feature of this tool is the use of UML[2] (Unified Modeling Language) class diagram notations for modeling ontological knowledge, thus making ontology development more intuitive to nonexperts in SWTs.

Kovalenko et al. (2013) analyze the types of relations that have to be modeled to integrate heterogeneous data sets in multidisciplinary engineering environments, such as those specific for mechatronic engineering projects. Based on these relations authors derive the requirements for mapping types, which will be needed while integrating heterogeneous data across engineering disciplines and tools with SWTs. Different technologies for mapping definitions and representation between ontologies are then analyzed with respect to their capabilities and limitations to support the identified mappings types. Authors focus on SPARQL[3] CONSTRUCT,

[2]UML: http://www.uml.org/.

[3]SPARQL Overview: https://www.w3.org/TR/sparql11-overview/.

SPIN[4], SWRL[5], and Jena[6] rules as the most widely used alternatives that enable the Semantic Web capability of mapping-based model integration for *knowledge integration* (*C2)*. This analysis of mapping representation languages and their strengths could be helpful while choosing an appropriate mapping language for a specific application scenario. More details on this line of work are available in Chap. 6.

11.3 Semantic Web Solutions for Model Consistency Management

In multidisciplinary engineering projects, defects in artifacts of individual disciplines can be propagated to artifacts in other disciplines, causing a major impact on product and process quality in terms of additional risk and time for defect repair. For instance, the sensor type specified in the physical topology model (mechanical engineering) of the automation system has to match the information in the corresponding electrical plan (electrical engineering) and the value range for control variables (software engineering) to describe a correct system. Defects may also come from inconsistencies between disciplines, which are not defects in any of the single discipline views. Because these interdisciplinary relations are not represented in a machine-understandable way, they cannot be checked and managed easily with standard tool support.

Model consistency management refers to the task of detecting defects and inconsistencies in models of individual engineering disciplines as well as across interrelated models from diverse engineering disciplines. The following works are examples of addressing model consistency management with SWTs. Table 11.2 sums up how approaches to consistency management discussed in this section make use of diverse SWT capabilities.

Feldmann et al. (2015) focus on providing a solution for identifying inconsistencies that may arise among diverse engineering models created during the engineering process of automated production systems. Such inconsistency detection contributes to the increased productivity of the engineering process as it supports the detection of potentially severe errors early in the engineering process. To that end, the authors propose the use of SWTs, in particular: (1) RDF[7] (Resource Description Framework) is used as a means to represent knowledge from various engineering models and its simple, triple-based data model acts as a common formalism to represent a variety of models with *conceptual modeling* (*C1*);

[4]SPIN Overview: https://www.w3.org/Submission/spin-overview/.

[5]SWRL Overview: https://www.w3.org/Submission/SWRL/.

[6]Jena: https://jena.apache.org/.

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

Table 11.2 SWT capabilities used by example approaches for model consistency management

C1: Semantic modeling	C2: Knowledge integration	C3: Browsing and exploration of distributed data sets	C4: Quality assurance with reasoning	C5: knowledge reuse
(Feldmann et al. 2015; Kovalenko et al. 2014; Abele et al. 2013; Feldmann et al. 2014b; Sabou et al. 2016)	Feldmann et al. 2015; Kovalenko et al. 2014)	(Sabou et al. 2016;	(Feldmann et al. 2015; Kovalenko et al. 2014; Abele et al. 2013; Feldmann et al. 2014b; Sabou et al. 2016)	(Kovalenko et al. 2014; Abele et al. 2013)

(2) representing all models using the same formalism enables specifying explicit links between the elements of those models for *knowledge integration* (*C2*); (3) the SPARQL query language is chosen as a mechanism to explicitly define and verify various inconsistency checks—this benefits from the formality of Semantic Web languages as well as employs reasoning (*C4*). Feldmann et al. (2015) identify that the use of ontologies for capturing knowledge that is shared among various models should be beneficial as a future extension of their work. Such an ontology-based approach to inconsistency detection is already taken by (Kovalenko et al. 2014), as discussed next.

Kovalenko et al. (2014) present an ontology-based approach to automatically detect inconsistencies across heterogeneous data sets produced by different engineering disciplines and tools in multidisciplinary engineering projects. OWL ontologies are used to explicitly represent the discipline/tool-specific knowledge and data in a machine-understandable form with *conceptual modeling* (*C1*). Mappings are then defined between the ontologies to make cross-disciplinary (or cross-tool) relations between the data models and data sets explicit for *knowledge integration* (*C2*). SPARQL queries are executed over the discipline/tool ontologies regarding the defined mappings in order to perform inconsistency detection across discipline/tool boundaries, thus taking advantage of the reasoning capabilities of SWTs (*C4*). Another advantage of using SWTs lies in the possibility to reuse the developed ontologies and formulated checks (SPARQL queries) in subsequent projects for *knowledge reuse* (*C5*).

An approach for the automated validation of plant models is presented in (Abele et al. 2013). Inconsistencies in plant models may arise when integrating different engineering views on the plant created by different experts (e.g., mechanical and electrical engineers), who work concurrently on developing models for the same plant. The approach relies on representing plant models in terms of ontologies and subsequently applying reasoning techniques for validation purposes. In particular, the focus is on models conforming to CAEX data format (Schleipen et al. 2008), a meta-model for the storage and exchange of engineering models defined by IEC 62424 (IEC 2008). An OWL representation of the CAEX plant models is obtained through an automated transformation, which relies on a set of mappings that the authors defined between CAEX and OWL constructs. A modular ontology design is adopted: first, taking advantage of *conceptual modeling* (*C1*), a CAEX base ontology captures in OWL the basic design decisions of representing CAEX models; second, a plant ontology imports the base CAEX ontology and extends it with vendor-specific information and instance data from the specific CAEX file. The base ontology can be *reused* across projects (*C5*). Then SPARQL queries and (for some checks) reasoning are used to perform consistency checks on the obtained ontology. *Reasoning* and querying mechanisms enabled by OWL allow validation and retrieval of implicit knowledge from ontologies (*C4*). The final decision of whether the identified inconsistencies are indeed a problem is left to domain experts, i.e., there is no automatic correction/fixing.

In the area of requirements and test case management, Feldmann et al. (2014b) propose a Semantic Web solution for verifying the consistency of requirements as well as of requirements and test cases. A *conceptual model* that describes the main elements for this use case is developed and then formalized as an OWL ontology (*C1*). *Reasoning mechanisms* (*C4*) such as satisfiability checking, instance classification, and model consistency checking are applied to support various consistency-related use cases in the management of requirements and their associated test cases.

The AutomationML Analyzer[8] (Sabou et al. 2016) uses Semantic Web and Linked Data technologies to provide an interface for analyzing data from integrated AutomationML files. AutomationML (Drath 2010) is an emerging format for exchanging engineering data. While the concerted use of AutomationML in an engineering project makes the integration between data from different engineering disciplines easier, tools are still needed for more easily navigating and analyzing integrated AutomationML data. To that end, the AutomationML Analyzer uses ontology-based technologies to integrate AutomationML data; to provide easy navigation support within the AutomationML data as well as to detect project level inconsistencies and defects through SPARQL querying of the integrated data. The main SWT capabilities used are: semantic modeling (*C1*) of an AutomationML ontology with which the input data is semantically enriched (see also Chap. 5); browsing and exploration of the semantic data through Linked Data based mechanisms (*C3*) and the use of reasoning mechanisms as part of the SPARQL querying activities (*C4*).

[8]AutomationML Analyzer: http://data.ifs.tuwien.ac.at/aml/analyzer.

11.4 Semantic Web Solutions for Flexible Comparison

In engineering settings, comparisons are often performed between engineering objects that should be replaced or interchanged. For example, it is often requested that a comparison be made between the capabilities of an engineering unit to be replaced (e.g., a device) and a new unit. *Flexible comparison* refers to performing such comparison among descriptions of engineering objects, as exemplified in the works described in this section. Table 11.3 depicts how approaches to flexible comparison discussed in this section make use of diverse SWT capabilities.

Feldmann et al. (2014a) consider the problem of parts exchange in an evolving manufacturing system, in particular, checking the compatibility of the old part and the new part. This is a complex problem since insights from multiple contributing disciplines must be taken into account. Therefore, support for such operations leads to an increased productivity of the engineering process. The authors offer a solution where model-based and Semantic Web approaches are combined: the SysML language provides a means for modeling interdisciplinary manufacturing systems using graphical means but lacks the formal foundations to allow automated compatibility checks between various components. This shortcoming is compensated by translating SysML models into OWL ontologies and exploring the formality of OWL for checking compatibility constraints expressed in terms of SPARQL queries. The Semantic Web capabilities of *conceptual modeling* (*C1*) and *reasoning enabled quality assurance* (*C4*) play a key role in supporting this application.

The use of SWTs for creating simulation models is discussed in (Novák and Šindelár 2011). Simulation models are widely used in industrial engineering in general, and for engineering production systems in particular, to perform experiments that would be too dangerous or expensive if performed in vivo. A major task is the generation of a simulation model, which consists of the selection of a set of suitable simulation blocks that accurately represent the state of the real plant. To improve this manual model creation process, the authors propose the use of three ontologies which allow the explicit representation of knowledge with *conceptual modeling* (*C1*) about simulation blocks (simulation ontology), the real industrial plant (plant ontology), and the signals of each simulation block (signal ontology). Mappings are created between these ontologies for *knowledge integration* (*C2*). This formalized knowledge enables the creation of a semantic engine that performs *reasoning*-based,

Table 11.3 SWT capabilities used by example approaches flexible comparison

C1: Semantic modeling	C2: Knowledge integration	C4: Quality assurance with reasoning	C5: Knowledge reuse
(Feldmann et al. 2014a; Novák and Šindelár 2011; Willmann et al. 2014)	(Novák and Šindelár 2011)	Feldmann et al. 2014; Novák and Šindelár 2011; Willmann et al. 2014)	(Novák and Šindelár 2011; Willmann et al. 2014)

flexible comparisons between the available simulation blocks and the industrial plant status, thus automating the creation of the simulation model (*C4*). The created ontologies and engine can be *reused* across diverse simulation events (*C5*).

Flexible comparison among products and production processes is an important aspect of any product ramp-up activity, which aims at identifying a suitable production process at a target site in order to produce a certain product with the same quality as at a source site. Willmann et al. (2014) propose K-RAMP, a knowledge-based production ramp-up process, where a core task is flexibly finding matches between semifinished products (production processes) available at the source and target production sites. SWTs were used to define, design, and evaluate the ramp-up process knowledge-base. SWT capabilities such as *conceptual modeling* (*C1*), *reasoning* for matchmaking (*C4*) as well as *reuse of knowledge* (*C5*) between various ramp-up activities are the most useful features in this setting. Chapter 9 provides more insights into this line of work.

11.5 Conclusions

A key goal of this chapter was to better understand how typical IEA tasks can be enabled by the SWT capabilities introduced earlier in Chap. 3. To that end, the chapter exemplified typical technical tasks supported by IEAs, in particular, a set of tasks that, according to (Lüder and Schmidt 2016), require support in the context of the engineering of production systems (i.e., *model generation, model transformation, model integration, consistency management, and flexible comparison*). Approaches to model generation and model transformation that rely on SWTs are discussed at length in Chap. 5. In Sects. 11.2–11.4 a set of current approaches was analyzed that make use of SWTs for supporting the engineering of production systems and address the tasks of model integration, consistency management, and flexible comparison. Table 11.4 sums up the analysis of the earlier chapters and

Table 11.4 Number of example approaches that solve a given technical task with a certain SWT capability

	C1: Semantic modeling	C2: Knowledge integration	C3: Browsing, exploration of distributed data sets	C4: Quality assurance with reasoning	C5: Knowledge reuse
Model integration	2	2			1
Model consistency management	5	2	1	5	2
Flexible comparison	3	1		3	2

identifies the number of approaches that address certain technical tasks by using the SWT capabilities.

The analysis leads to the following conclusions:

- *SWTs support various aspects of production system's engineering*. Although our focus was restricted to the engineering phase of production systems, it was observed that SWTs have been applied to support various aspects of this process ranging from requirements management (Feldmann et al. 2014b), to simulation (Novák and Šindelár 2011) and project management (Grünwald et al. 2014). Although diverse, use cases form these various life cycle stages or production systems, are enabled at a technical level by a few individual tasks. This chapter focuses on the tasks of model integration, model consistency checking, and flexible comparison.
- *Model consistency management* tasks are of major importance. They have applications in a wide range of settings. The task of *flexible comparison* is more complex and therefore less explored to date. *Model integration* is often not a goal *per se*, but rather an enabler for the other tasks, especially in the settings that consider engineering models from multiple disciplines.
- *Formal and flexible conceptual modeling (C1) is the most used SWT capability.* Often performed by using modeling approaches of wide adoption in engineering such as SysML (Feldmann et al. 2014a) or UML (Grünwald et al. 2014), this feature is essential for attaining all three technical tasks we analyzed. The *reasoning* SWT capability (*C4*) is used by all approaches dealing with consistency management and flexible comparison. *Knowledge integration* techniques (C2), such as model mapping, play an important role for model integration being applicable in all scenarios, in which multiple models are involved that need to be integrated before advanced reasoning-driven analytics can be performed. The possibility to easily *reuse* formally represented conceptual knowledge (C5) was perceived as a clear benefit across the various usage scenarios. This is somewhat conflicting with other evidence from the literature which alerts to the difficulty of performing ontology reuse both in general (Oberle 2014; Simperl 2009) and in engineering contexts in particular (Legat et al. 2014; Terkaj and Urgo 2014).
- *Web-based SWT capabilities are less frequently used.* Although one of the strengths of SWTs is the combination of traditional knowledge representation and reasoning techniques with Web compliance features, there is a clear tendency, at least in the papers we reviewed, to primarily explore the semantic features of these technologies as opposed to those related to Web compliance, in particular, C3 related to *Browsing and Exploration of Distributed Data Sets*. This could be a consequence of the fact that SWTs are primarily used as an enterprise data integration and management solution, where Web-oriented features (unique URIs, reuse from other Web data sets) are of less importance. An interesting future research question is therefore the investigation of how the Web-compliance-related features of SWTs could be of use in engineering of mechatronic systems, and more broadly in *Industrie 4.0*.

11.6 Outlook on Part IV

Part IV showcases three works that use SWTs in engineering settings to solve all three basic tasks discussed above: model integration, consistency management, and flexible comparison. We provide a summary of these chapters and discuss how they use SWT capabilities to address the three technical tasks.

Chapter 12 (*Semantic Web Solutions in the Automotive Industry*) reports on two use cases in the context of the automotive industry solved with SWTs. The first use case aims to support the engineer in deriving an optimized design layout starting from a system specification, which is refined in iterative design steps. To enable these tasks, ontologies are used to represent requirements, to allocate them to parts of the systems, to attach constraints to requirements and parts of the system, and to keep track of different versions of requirements during the subsequent processing (i.e., design phases). The main advantage of using ontologies in this case was representing requirements explicitly and in a machine processable way (*C1*). This allowed the versioning of the requirements and attaching constraints which were then verified with a *Relational Constraint Solver* (RCS), a constraint engine specialized to check numeric equations. Other benefits of this solution were: (1) enhanced *reuse* (C5) of previous knowledge (e.g., requirement templates and system structure information could be reused across design problems); (2) the *formal representation* of requirements and associated constraints was successfully used to guide the engineers during the design phase and prevent them from entering incorrect values; (3) thanks to version control over the iterative steps, this solution proved more manageable than the baseline, an *Excel*-based approach.

The second use case (UC2) focuses on supporting the collaborative development process of an automatic transmission gearbox by distributed engineering groups from different engineering disciplines needed to realize this complex mechatronic object, namely: mechanical, hydraulic, electrical, and software engineering. This use case illustrates the task of *consistency management* of several overlapping models of the same system (the authors refer to these as views, similarly to the terminology in Chap. 13). An additional need was to enable change propagation among the interrelated models, meaning that a change in one model would lead to changes in all the related models whenever necessary (i.e., according to the interdependencies specified between models). SWTs were used to address these needs in particular with the capabilities *C1, C2, C4*.

As the reported work was performed in the early years of Semantic Web research when the representation languages were still under development, the authors made use of classical knowledge representation techniques such as a Frame logic-based knowledge representation and the *Flora2*[9] reasoning framework. This use case also demonstrates solving a *model integration* task. Domain-specific knowledge is integrated by reference to so-called *Engineering ontologies*, which capture common concepts shared across engineering disciplines. The authors observed several

[9]Flora2 reasoning framework: http://flora.sourceforge.net/.

benefits to engineers such as enabling automatic consistency checking and change propagation among engineering models, which was not possible before. In addition, engineers also benefitted from the fact that the dependencies between their models were made explicit—as such they achieved a better understanding of how changes in their model might affect other models.

Chapter 13 (*Leveraging Semantic Web Technologies for Consistency Management in Multi-Viewpoint Systems Engineering*) focuses on *consistency management* among different, overlapping views (or models) of the same complex systems. As such, this chapter captures a typical problem since such overlapping models are often created by engineering tool networks, each model representing one engineering discipline's view on the system. The authors propose a solution where RDF is used to encode different system views in a uniform manner and the emerging *Shapes Constraint Language* (SHACL) is employed to define the inter-viewpoint dependencies (*C1, C2*). These dependencies can be automatically checked during modeling time to uncover potential inconsistencies between the various models. The *Reasoning* SWT capability supports this task (*C4*). Going beyond the case when dependencies are specified between a set of views, the authors also consider a use case which requires the *semantic integration* of multiple viewpoints prior to checking the consistency among these viewpoints. As a technical solution for solving the engineering data integration, they choose the *hybrid ontology integration* approach, where local models are mapped to a global ontology, which contains shared knowledge (such as common concepts discussed in Chap. 5). SHACL expressions are used to define the correspondences between local and global views in line with the SWT capability *C2* on *data integration*. The authors note that the Semantic Web solution described in the chapter is highly compatible with and complementary to Model-Driven Engineering approaches, as models can be easily transformed into Semantic Web-specific languages.

Chapter 14 (*Applications of Semantic Web Technologies for the Engineering of Automated Production Systems—Three Use Cases*) details three use cases from the process of engineering automated production systems where SWTs are used. The first use case (UC1) focuses on ensuring compatibility between mechatronic modules that need to be replaced in a given system configuration, illustrating a setting that needs *flexible comparison* approaches. This use case requires means for (1) identifying modules compatible with a module that needs to be replaced and (2) identifying and resolving conflicts in a given system configuration as a follow-up of a module change. There is a need for *knowledge representation* and for intelligent access mechanisms that can accomplish such comparisons. For this, the authors propose using an ontology for representing compatibility information (*C1*) and encoding and checking compatibility through SPARQL queries.

The second use case (UC2) focuses on the task of *ensuring consistency* between requirements specified for a production system and test cases corresponding to checking that requirement. The authors use semantic modeling to formally *represent domain knowledge* (C1) and reasoning services offered by Semantic Web *reasoners* to check whether test cases are compatible with requirements (C4).

The third use case (UC3) aims to *detect consistency* between different engineering models of the same system. Because the engineering models describe the same system, they overlap to some extent and as such these overlapping parts should be kept consistent. To achieve this task, there is also a need to define which parts of the models correspond to each other as a basis for compatibility checks. The authors propose achieving such *engineering data integration* by (1) defining a base vocabulary that contains the common concepts used by the various models considered (C1) and (2) using a common data representation language (namely RDF) to encode the various models in a uniform way and describe equivalent *mappings* between their corresponding elements (C2).

All use cases rely on a combination of Semantic Web and *Model-based Engineering* (MBE). Concretely, the authors leverage the widespread adoption and good tool support for MBE, especially the SysML4Mechatronics language to collect relevant system models from engineers. These models are then translated into Semantic Web formats that allow specifying correspondences between, querying and reasoning on the various engineering models.

Table 11.5 provides an overview of the discussion in this chapter by depicting the task(s) addressed by each chapter and the SWT capabilities used for solving those tasks. Several of the conclusions drawn from the analysis of example applications (Sect. 11.5) can also be made when considering the chapters of Part IV. We observe that these works cover various aspects of production system's engineering and that model consistency management is a task addressed by all chapters. It is also remarkable that all chapters showcase the need for model integration prior to enabling model consistency management and that they rely on similar data integration approaches. As with the example approaches, for the chapters in Part IV, the conceptual modeling SWT (C1) is most frequently used while capabilities C3 and C5 are not used at all. Another interesting observation is that Chaps. 13 and 14 aim to establish synergies with system engineering languages that are more widely spread among engineers as a way to facilitate the acquisition of domain models.

Table 11.5 Overview of technical tasks addressed by chapters in Part IV and the SWT capabilities used to address those tasks

	C1: Semantic modeling	C2: Knowledge integration	C4: Quality assurance with reasoning
Model integration	Ch12 UC2, Ch13, Ch4 UC3	Ch12 UC2, Ch13, Ch4 UC3	
Model consistency management	Ch12 UC2, Ch13, Ch14 UC2		Ch12 UC2, Ch13, Ch14 UC2
Flexible comparison	Ch14 UC1		

Acknowledgments This work was supported by the Christian Doppler Forschungsgesellschaft, the Federal Ministry of Economy, Family and Youth, and the National Foundation for Research, Technology and Development in Austria.

References

Abele, L., Legat, C., Grimm, S., Müller, A.W.: Ontology-based validation of plant models. In: 11th IEEE International Conference on Industrial Informatics (INDIN), pp. 236−241. IEEE (2013)

Aldred, L., van der Aalst, W., Dumas, M., Hofstede, A.: Understanding the challenges in getting together: the semantics of decoupling in middleware. In: BPM Center Report BPM-06-19, BPMCenter.org, p. 36 (2006)

Bauernhansl, T., ten Hompel, M., Vogel-Heuser, B. (eds.): Industrie 4.0 in Produktion, Automatisierung und Logistik. Springer (2014)

Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Sci. Am., 29−37 (2001)

Drath, R.: Datenaustausch in der Anlagenplanung mit AutomationML: Integration von CAEX. PLCopen XML und COLLADA. Springer, DE (2010)

IEC: IEC 62424. Representation of process control engineering. Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools (2008)

Feldmann, S., Kernschmidt, K., Vogel-Heuser, B.: Combining a SysML-based modeling approach and semantic technologies for analyzing change influences in manufacturing plant models. In: Proceedings of the 47th CIRP Conference on Manufacturing Systems, pp. 451−456 (2014)

Feldmann, S., Rösch, S., Legat, C., Vogel-Heuser, B.: Keeping requirements and test cases consistent: towards an ontology-based approach. In: 12th IEEE International Conference on Industrial Informatics, INDIN, pp. 726−732 (2014)

Feldmann, S, Herzig, S.J.I., Kernschmidt, K., Wolfenstetter, T., Kammerl, D., Qamar, A., Lindemann, U., Krcmar, H., Paredis, C.J.J., Vogel-Heuser, B.: Towards effective management of inconsistencies in model-based engineering of automated production systems. In: Proceedings of IFAC Symposium on Information Control in Manufacturing (INCOM 2015) (2015)

Gruber, T.R.: A translation approach to portable ontology specifications. Knowl. Acquis. **5**(2), 199–220 (1993)

Grünwald, A., Winkler, D., Sabou, M., Biffl, S.: The semantic model editor: efficient data modeling and integration based on OWL ontologies. In: Proceedings of the 10th International Conference on Semantic Systems, SEM '14, pp. 116−123. ACM (2014)

Halevy, A.: Why your data won't mix. In: Queue, vol. 3, no. 8, pp. 50−58 (2005). ISSN 1542-7730

Hohpe, G., Woolf, B.: Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Addison-Wesley Longman, pp. 480 (2003). ISBN 978-0321200686

Kovalenko, O., Debruyne, C., Serral, E., Biffl, S.: Evaluation of technologies for mapping representation in ontologies. In: On the Move to Meaningful Internet Systems: OTM 2013 Conferences, pp. 564−571. Springer (2013)

Kovalenko, O., Serral, E., Sabou, M., Ekaputra, F.J., Winkler, D,, Biffl, S.: Automating cross-disciplinary defect detection in multi-disciplinary engineering environments. In: Knowledge Engineering and Knowledge Management, pp. 238−249. Springer (2014)

Legat, C., Lamparter, S., Vogel-Heuser, B.: Knowledge-based technologies for future factory engineering and control. In: Borangiu, T., Thomas, A., Trentesaux, D. (eds.) Service Orientation in Holonic and Multi Agent Manufacturing and Robotics, Studies in Computational Intelligence, vol. 472, pp. 355–374. Springer, Berlin (2013)

Legat, C., Seitz, C., Lamparter, S., Feldmann, S.: Semantics to the shop floor: towards ontology modularization and reuse in the automation domain. In: Proceedings of 19th IFAC World Congress, Cape Town, South Africa, pp. 355−374 (2014)

Lüder, A., Schmidt, N.: Challenges of mechatronical engineering of production systems—an automation system engineering view. In: Ghezzi, L., Hömberg, D., Landry, L. (eds.) Math for the Digital Factory. Springer, Heidelberg (2016)

Moser, T., Mordinyi, R., Mikula, A., Biffl, S.: Making expert knowledge explicit to facilitate tool support for integrating complex information systems in the ATM domain. In: Proceedings of the International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), pp. 90−97. IEEE (2009). ISBN 978-1-4244-3569-2

Moser, T., Mordinyi, R., Mikula, A., Biffl, S.: Efficient integration of complex information systems in the ATM domain with explicit expert knowledge models. In: Xhafa, F., Barolli, L., Papajorgji, P.J. (eds.) Complex Intelligent Systems and Their Applications, vol. 41, pp. 1−19. Springer (2010). ISBN 978-1-4419-1635-8

Novák, P., Šindelár, R.: Applications of ontologies for assembling simulation models of industrial systems. In: Proceedings of the Confederated International Conference on the Move to Meaningful Internet Systems, pp. 148−157. Springer (2011)

Noy, N.F.: Semantic integration: a survey of ontology-based approaches. ACM SIGMOD Record **33**(4), 65–70 (2004)

Noy, N.F., Doan, A.H., Halevy, A.Y.: Semantic integration. AI Mag. **26**(1), 7 (2005)

Oberle, D.: How ontologies benefit enterprise applications. Semant Web J. **5**(6), 473–491 (2014)

Sabou, M., Kovalenko, O., Ekaputra, F.J., Biffl, S.: Beiträge des Semantic Web zum Engineering für Industrie 4.0. In: Vogel-Heuser, B., Bauernhansl, T., ten Hompel, M. (eds.) Handbuch Industrie 4.0. Springer (2015). doi:10.1007/978-3-662-45537-1_90-1

Sabou, M., Ekaputra, F.J., Kovalenko, O.: Supporting the engineering of cyber-physical production systems with the AutomationML analyzer. In: Proceedings of the CPPS Workshop, at the Cyber-Physical Systems Week, Vienna (2016)

Schleipen, M., Drath, R., Sauer, O.: The system-independent data exchange format CAEX for supporting an automatic configuration of a production monitoring and control system. In: IEEE International Symposium on Industrial Electronics, pp. 1786−1791. IEEE (2008)

Schmidt, N., Lüder, A., Biffl, S., Steininger, H.: Analyzing requirements on software tools according to functional engineering phase in the technical systems engineering process. In: Proceedings of the 19th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). IEEE (2014)

Shadbolt, N., Berners-Lee, T., Hall, W.: The semantic web revisited. IEEE Intell. Syst. **21**(3), 96–101 (2006)

Simperl, E.: Reusing ontologies on the semantic web: a feasibility study. Data Knowl. Eng. **68** (10), 905–925 (2009)

Studer, R., Benjamins, V., Fensel, D.: Knowledge engineering: principles and methods. Data Knowl. Eng. **25**(1–2), 161–197 (1998)

Tasso, C., Arantes e Oliveira, E.D. (eds.) Development of Knowledge-Based Systems for Engineering. Springer, Vienna (1998)

Terkaj, W., Urgo, M.: 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, pp. 24–27. Graz, Austria (2012)

Terkaj, W., Urgo, M.: Ontology-based modeling of production systems for design and performance evaluation. In: 12th IEEE International Conference on Industrial Informatics (INDIN), pp. 748−753. IEEE (2014)

Willmann, R., Biffl, S., Serral, E.: Determining qualified production processes for new product ramp-up using semantic web technologies. In: Proceedings of the 14th International Conference on Knowledge Technologies and Data-Driven Business. ACM (2014)