Towards Semantic Integration of Supply Chain and Production Data

Dissertation
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn.

von
Niklas Petersen
aus
Husum

Bonn, 25.07.2019

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn.

1. Gutachter:	Prof. Dr. Sören Auer
2. Gutachter:	Prof. Dr. Antoine Zimmermann
Tag der Promotion:	03.03.2020
Erscheinungsjahr:	2020

Abstract

Data is a critical ingredient for many analyses. Its recent exponential growth allowed utilizing computers to solve previously considered hard problems. For organizations and entire economies, being able to solve harder problems results in an competitive advantage. However, organizations often struggle to make use of the growing amount of data. Data is often spread in various places, organized in different structures and described in different models. While the rising popularity in initiatives such as *Industry 4.0*, *Smart Manufacturing* and *Cyber-physical systems* shows that there is a strong interest in new data-driven applications in the industry, it is not completely clear yet how these ideas can be realized. Advances in the field of Knowledge Representation, in particular the Semantic Web, provide a promising technology stack to better organize and describe heterogeneous and distributed data.

In this thesis, we investigate how these semantic technologies can be applied in an industrial context. In particular, we look into methods to better describe and integrate data such that it can be used by more individuals and applications outside of the original use case of a particular dataset. With a focus on Supply Chains and production data, the goal is to support decision makers in optimizing supply chains or factory production lines. This includes providing them with a holistic view on the digital and physical assets of an organization and creating an common understanding on the nature of their domain, inside and outside the organization.

In order to tackle these challenges, we propose various ontologies to describe Supply Chains, factories, production lines and enterprises as a whole. We evaluate these ontologies with queries which represent KPIs defined by domain experts. In particular, we show how a supply chain ontology can enable the management of inter-organizational supply chains. We demonstrate how sensor and production data from heterogeneous databases can be integrated using semantic technologies. For the development of ontologies, we further propose an ontology authoring environment. The environment supports industrial domain experts and knowledge engineers to work collaboratively on the formalization of ontologies which represent a common understanding of a particular domain. Our results show that the semantic integration of industrial data comes at a reasonable performance and that it can help an organization to better exploit their existing data.

Acknowledgments

Being a kid who grew up on a farm, I would like to thank a couple of individuals without whom I would have likely not finished my PhD research: First, Prof. Dr. Sören Auer for taking me into his research group, for supervising me, and for giving me a lot of freedom to pursue my own ideas and interests. Second, Dr. Christoph Lange for his continuous support, our interesting scientific and non-scientific discussions and the beers we shared outside the office. To Dr. Steffen Lohmann, who took me into a lot of interesting projects and provided me with valuable feedback on my research. To the entire Enterprise Information Systems group of the University of Bonn and Fraunhofer IAIS, where we shared a lot of good moments at and outside of work. Special thanks to Michael Galkin for the enjoyable trips home and to Christian Mader for sharing his wisdoms of life.

Last but not least, thanks to my family for supporting their youngest traveling around and doing the things he found interesting.

Contents

CHAPTER 1

Introduction

Data is everywhere. Its main purpose is to enable machines to process information more efficiently than a human could do in manual labor. However, the exponential growth of data in recent years raises many new challenges for individuals and organizations alike to efficiently use data for their task at hand. Coined as *Big Data*, these challenges were structured into dimensions called *V6*, which are defined in [1] as follows:

Volume refers to the magnitude of data. Big data sizes are reported in multiple terabytes and petabytes.

Variety refers to the structural heterogeneity in a dataset. Technological advances allow firms to use various types of structured, semi-structured, and unstructured data.

Velocity refers to the rate at which data is generated and the speed at which it should be analyzed and acted upon.

Veracity represents the unreliability inherent in some sources of data.

Variability refers to the variation in the data flow rates. Often, big data velocity is not consistent and has periodic peaks and troughs.

Complexity (sub of Variability) This imposes a critical challenge: the need to connect, match, cleanse and transform data received from different sources.

Value can be obtained by analyzing large volumes of such data.

This thesis addresses the dimensions *Variety* and *Complexity* of data in the industrial domain in general, and supply chain in particular. For manufacturing enterprises, the efficient usage of their data translates directly into gaining a competitive advantage in their domain. The rising interest in *Industry 4.0*, *Smart Manufacturing* and *Cyber-physical systems* initiatives to build more data-driven applications demonstrates that these challenges exist and are real.

In contrast to unstructured information (also called *unstructured data*), data always comes with a model which structures and describes the data. Models exist in the shape of formal languages, schemas, markup languages, taxonomies, vocabularies or ontologies. They differ in their structure (e.g. tabular, trees, graphs) and offer their own serializations. While all models aim to support the user to make sense of the data structure, often a lot of context knowledge which is needed for the correct interpretation (aka *semantics*) is missing such that the usage becomes difficult or even impossible in the time given. According to NASA, their scientists spend 60% of their time preparing data before it can be analyzed for

their research [2]. An AI study put that number to over 80% for tasks involving data preparation and data engineering[1] A Harvard Business Review report documents as well that "80% of analysts' time is spent simply discovering and preparing data."[2].

Advances in the field of Knowledge Representation, in particular the Semantic Web [3], provide an entire technology stack which focuses on preserving the meaning of data and even their models. We believe the Linked Data paradigm [4] of the Semantic Web to describe, publish and interlink data can help organizations in the industrial domain to more efficiently use, interpret, exchange and analyze data. Therefore, the aim of this thesis is to investigate different strategies for organizations to apply semantic technologies for lowering the costs of the Big Data dimensions *variety* and *complexity*. Especially supply chains, where data is in different database management system, conforming to different standards, formalized in different structures, across different organizations, seems to be a prime candidate for applying these technologies.

1.1 Motivation

It is the goal of every individual in an organization to execute their given tasks at an optimum with regard to time, costs and available information. Information, formalized as data, is often spread among different IT systems, databases, formats, conform to different standards and data modelling paradigms. This puts the individual or system to make informed decisions into a difficult spot.

With regard to Supply Chain Management, failures can lead to enormous costs (factory or production line outtakes, product recalls), but may also affect lives if mistakes have been made for example in food supply chains or the aerospace industry. Furthermore, legal restrictions and distrust among supply chain partners contributes to the complexity of the problem, but are beyond the scale of this thesis.

Industry 4.0 initiatives aim to promote the establishment of "global networks that incorporate their machinery, warehousing systems and production facilities in the shape of Cyber-Physical Systems (CPS)" [5]. In particular, production facility have to be "capable of autonomously exchanging information, triggering actions and controlling each other independently" [5]. The potential of reaching that goal is estimated to contribute up to 78 billion to the Germany GDP by the year 2025 [6].

Thus, the overall motivation is to explore how semantic technologies may be used to lay the groundworks for new IT infrastructures, where humans make better decisions and "computers [...] do more useful work"[3].

1.2 Research Questions

To address the stated problems described in the previous section, we have defined the following research questions:

1. **RQ1: How can the Linked Data paradigm be used to represent and evaluate industrial data to improve Supply Chain Management?**

   Supply Chain Management is driven by the constant optimization of individual processes. Therefore, we investigate where the Linked Data approach can help Supply Chain managers to identify strong and weak processes within and outside their organization. In particular, we analyze how an

[1] https://www.cognilytica.com/2019/03/06/report-data-engineering-preparation-and-labeling-for-ai-2019/
[2] https://hbr.org/2017/05/whats-your-data-strategy
[3] https://www.w3.org/standards/semanticweb/

ontology can be used to reflect the agreed process interpretation among multiple organizations and how such an ontology could be used to calculate the KPIs needed by the Supply Chain managers.

2. **RQ2: How can we support domain experts contributing to the engineering of industrial ontologies?**

   Building industrial ontologies requires the input of domain experts in the industry as well as ontology engineers to structure and formalize this input. Therefore, we study on what keeps domain experts from providing useful contributions in existing ontology authoring applications. In particular, we collect the requirements needed to address these identified issues and provide an alternative to lower the entry-barrier for industrial domain experts to engage in ontology engineering projects.

3. **RQ3: How can we leverage semantics in industrial use cases?**

   In this question, we analyze where semantic technologies can play a crucial role to solve problems in industrial applications. In particular, we look how ontologies can provide the necessary context needed to make more informed business decisions. The covered use cases include a view on problems within a factory, within an organization and within the exchange of data among multiple organizations. Furthermore, we also report on our findings on why it can be challenging to implement semantic technologies within an organization.

1.3 Thesis Overview

To prepare the reader for the rest of the document, we present an overview of our main contributions and the research areas investigated and provide references to the scientific publications covering this work, and an overview of the thesis structure.

1.3.1 Research Map

As shown in Figure 1.1, our contributions combine research from the areas of Ontology Engineering, Ontology Authoring Tools, Supply Chain Management, Industry 4.0.

1.3.2 Contributions

1. Methods, architectures, prototype implementations and evaluations for the usage of semantic technologies in various industrial domains: Industry 4.0 in general, Supply Chain Management, steel industry, production systems, decentralized industry networks.

2. *TurtleEditor*, a synchronized visual and text editor for the development of ontologies in the *Turtle* RDF serialization. The source-code of the editor is published on the Git version control hosting platform GitHub[4]. TurtleEditor is integrated into the collaborative ontology engineering environment *VoCol*[5].

3. *SCORVoc SPARQL queries* for calculating Supply Chain KPIs defined by domain experts in the SCOR standard. The queries are published on the Git version control hosting platform GitHub[6].

[4] `https://github.com/EIS-Bonn/TurtleEditor`
[5] `https://github.com/vocol/vocol`
[6] `https://github.com/vocol/scor/tree/master/metrics`

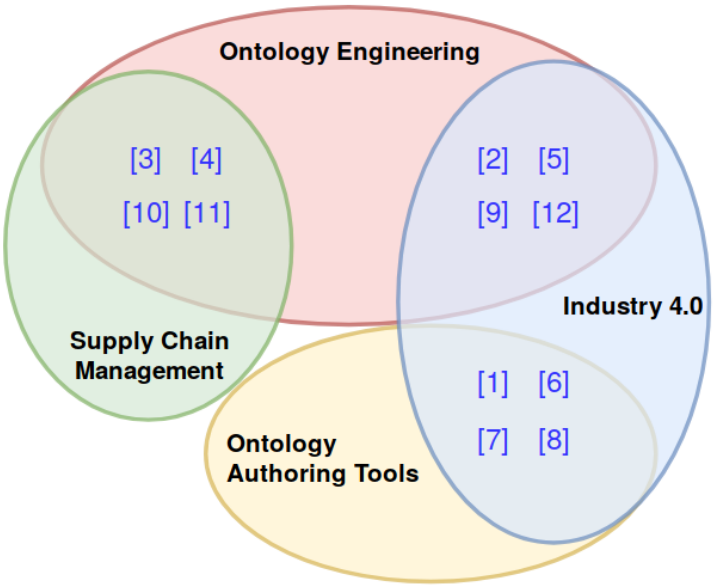


Figure 1.1: Overview of the main research areas and methodologies covered by this thesis. Numbers correspond to the paper contributions listed in Subsection 1.3.3.

4. Vocabularies/Ontologies for industry and other domains
   - *SCORVoc* – An RDFS vocabulary for managing Supply Chains based compliant to the SCOR [7] standard. The methodology for creating the vocabulary and an implementation are described in Chapter 4. The vocabulary is published on the version control hosting platform GitHub[7].
   - An RDF-based *Information Model* for creating a holistic view on physical assets such as machines and buildings and digital assets such as sensor data and production processes. The methodology, architecture and implementation are described in Section 6.1.
   - A *Factory Ontology* for creating a holistic view on the an enterprise with a focus on physical assets in a factory. The methodology, architecture and an implementation are described in Subsection 6.2.1.
   - A *DWD* weather ontology based on the open data published by the German Weather Service (Deutscher Wetterdienst, DWD). The vocabulary is published on the Git version control hosting platform GitHub[8].
   - A *Datex2* ontology for highway roadworks. The ontology is used to transform and integrate data published in XML by the German states into RDF. The vocabulary is published on the Git version control hosting platform GitHub[9].
   - Contribution to the development of the *IDS Information Model* for describing components and datasets in the *International Data Spaces*. The vocabulary is published on the Git version control hosting platform GitHub[10]. Section 6.3 provides a first documented implementation of the IDS Information model.

[7] https://github.com/vocol/scor
[8] https://github.com/vocol/dwd
[9] https://github.com/vocol/datex2
[10] https://github.com/IndustrialDataSpace/InformationModel

- Contributions to the development of *MobiVoc* – A mobility vocabulary describing concepts such as electric charging points, parking facilities, transportation types and highway roadworks. The vocabulary is published on the Git version control hosting platform GitHub[11].
- Contributions to the *MTConnect* vocabulary for describing machine components. The vocabulary is published on the Git version control hosting platform GitHub[12].

Active membership in the following working groups:

- APICS Supply Chain special focus forum to support digitization strategies of the standardization process for the *SCOR* standard.
- Germany federal ministry of transport and digital infrastructure (BMVI) metadata standardization group to discuss metadata activities and goals for German open data portals. Contributor to the *Guidelines for comprehensive metadata*[13] report published by the group.
- IDS information model group for contributing in the development of the IDS Information Model in the *International Data Spaces*.

1.3.3 List of Publications

The following publications constitute a scientific basis of this thesis and serve as a reference point for numerous figures, tables and ideas presented in the later chapters. In the Appendix A, a complete list of all publications by the author is available.

- *Journal Articles (1)*

  1. **TurtleEditor: A Web-based RDF Editor for Supported Distributed Ontology Development on Repository Hosting Platforms** Niklas Petersen, Alexandra Similea, Christoph Lange, Steffen Lohmann. In International Journal on Semantic Computing, Vol. 11, No. 03, 311–323, 2017;

- *Conference Papers (8)*:

  2. **Realizing an RDF-based Information Model for a Manufacturing Company** Niklas Petersen, Lavdim Halilaj, Irlán Grangel-González, Steffen Lohmann, Christoph Lange, Sören Auer. In Proceedings of the 16th International Semantic Web Conference (ISWC '17), 350–366, Springer;
  3. **SCORVoc: Vocabulary-Based Information Integration and Exchange in Supply Networks** Niklas Petersen, Irlán Grangel-González, Gökhan Coskun, Sören Auer, Marvin Frommhold, Sebastian Tramp, Maxime Lefrançois, Antoine Zimmermann. In Proceedings of the Tenth International Conference on Semantic Computing (ICSC '16), 132–139, IEEE;
  4. **Towards Federated, Semantic-based Supply Chain Analytics** Niklas Petersen, Christoph Lange, Sören Auer, Marvin Frommhold, Sebastian Tramp. In Lecture Notes in Business Information Processing (BIS '16), 436–447, Springer;

[11] https://github.com/vocol/mobivoc
[12] https://github.com/vocol/MTConnect
[13] https://www.wik.org/fileadmin/mFUND_VF/201902_mFUND-WIK-Veroeffentlichung_LeitfadenAussagekraeftigeMetadaten.pdf

5. **Monitoring Automating Factories Using Semantic Models** Niklas Petersen, Michael Galkin, Christoph Lange, Steffen Lohmann, Sören Auer. In Lecture Notes in Computer Science (LNCS, volume 10055; Joint International Semantic Web Technology Conference (JIST) '16), 315–330, Springer;
6. **TurtleEditor: An Ontology-Aware Web-Editor for Collaborative Ontology Development** Niklas Petersen, Gökhan Coskun, Christoph Lange. In Proceedings of the Tenth International Conference on Semantic Computing (ICSC '16), 183–186, IEEE;
7. **TurtleEditor 2.0: A Synchronized Visual and Text Editor for RDF Graphs** Alexandria Similea, Niklas Petersen, Christoph Lange, Steffen Lohmann. In Proceedings of the 11th International Conference on Semantic Computing (ICSC '17), 286–289, IEEE;
8. **VoCol: An Integrated Environment to Support Version-Controlled Vocabulary Development** Lavdim Halilaj, Niklas Petersen, Irlán Grangel-González, Christoph Lange, Sören Auer, Gökhan Coskun, Steffen Lohmann. In Knowledge Engineering and Knowledge Management of the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW '16), 303–319, Springer;
9. **Ontology-based Information Modelling in the Industrial Data Space** Jaroslav Pullmann, Christian Mader, Niklas Petersen, Steffen Lohmann. In Proceedings of the 22nd International Conference on Emerging Technologies and Factory Automation (ETFA '17), 1–8, IEEE;

- *Workshop Papers (3)*:

10. **Distributed Linked Data Business Communication Networks: The LUCID Endpoint** Sebastian Tramp, Rubén Navarro Piris, Timofey Ermilov, Niklas Petersen, Marvin Frommhold, Sören Auer. In Proceedings of The Semantic Web: ESWC 2015 Satellite Events (ESWC '15), 154–158, Springer;
11. **Publish and Subscribe for RDF in Enterprise Value Networks** Marvin Frommhold, Natanael Arndt, Sebastian Tramp, Niklas Petersen. In Proceedings of the Workshop on Linked Data on the Web (LDOW '16), Co-located with the 25th International World Wide Web Conference (WWW '16);
12. **An MTConnect Ontology for Semantic Industrial Machine Sensor Analytics** Glykeria Alvanou, Niklas Petersen. Submitted to the Workshop on Semantic Web of Things for Industry 4.0 (SWeTI'18), Co-located with the 15th Extended Semantic Web Conference (ESWC '18);

1.4 Thesis Structure

The thesis is structured into seven chapters:

- Chapter 1 introduces the thesis, motivates the conducted study, lists the scientific contributions, defines the approached research questions, and provides a list of published scientific papers that in detail describe those contributions.
- Chapter 2 presents fundamental concepts and preliminaries in the fields of Supply Chain Management, Industry 4.0 and Semantic Technologies for understanding the research problem and chosen approaches to tackle this problem.

- Chapter 3 discusses state-of-the-art efforts within academia to address the research questions defined in Chapter 1.
- In Chapter 4, we presents a key contribution of this thesis, a semantic vocabulary to manage Supply Chains. We describe the methodology that was followed to build the vocabulary and explain in detail how the semantics of the vocabulary can be leveraged. That is, we explain how data can be expressed using the vocabulary, how it can be queried to calculate the standardized KPIs needed by Supply Chain managers to judge the performance of the Supply Chain.
- Chapter 5 describes our efforts to lower the entry-barrier for domain experts in industrial ontology engineering projects. In particular, we present our collected requirements, a derived architecture and the implementation of the tool we propose named *TurtleEditor*. We evaluated TurleEditor in a user study and further describe its integration into the collaborative ontology developed environment named *VoCol*.
- Chapter 6 provides three industrial Use Case studies where semantics technologies were applied: First, we report on how semantic technologies can be used to build an information model for an organization to manage their data assets. Second, how formalized context information can help to optimize the management of machines in a factory. Third, we describe how vocabularies are used to build a distributed infrastructure for the exchange of data between organizations.
- Finally, Chapter 7 concludes the thesis with a discussion on the results, open problems and possible future directions in research.

CHAPTER 2

Preliminaries

This thesis builds upon the following scientific domains: The use cases are based on the fields of "Supply Chain Management" and the recently emerging field of *Industry 4.0*. The technologies used to address the research question are based on the domain of "Semantic Technologies".

2.1 Supply Chain Management

Figure 2.1 visualizes a 2D-view on an abstract supply chain. The flow of goods (or processes) are represented by the arrows which point to circles which represent an organization. The circle in the middle depicts an *Original Equipment Manufacturer* (OEM) enterprise from whose perspective the supply chain is viewed. While the OEM is aware of the performance of its direct suppliers and clients, outside of their own view, it is unknown how well these processes perform. The percentage is an example of a process performance result. As an example, the percentage could represent the process *all orders delivered in full*.

Interest in Supply Chain Management (SCM) increased in the 1980s once companies saw the benefits of collaborative relationships within and beyond their organization [8]. Ever since, internal enterprise information systems have experienced significant technical and scientific advancement to make their supply chains more productive. However, comparatively little progress has been made to improve the exchange of information *between* organizations [8, 9]. Reasons differ: from the lack of guidelines to create alliances, to the failure to develop measures for monitoring alliances or the lack of trust and the lack of integrated information systems and electronic commerce linking firms [8]. The IT research and advisory firm Gartner expects the SC market to grow up to $13 billion by 2021[1].

As high quality information is a prerequisite for companies to accomplish their business and strategic goals [10], organizations slowly start to treat their "information as a product" [10] and tend to invest more in information quality.

Up till now, still a considerable amount of communication between enterprises is done via informal channels such as emails (including file attachments) or phone calls.Only tier-1 suppliers of major Original Equipment Manufacturers (OEM) are usually fully integrated into the information exchange and corresponding IT support (e.g. EDI - Electronic Data Interchange connections) as they are expensive to deploy and maintain. Since they are the most dependent on well structured value chains, they are the ones who are urging their suppliers to share crucial information [11] to prevent SC failures.

[1] https://www.gartner.com/newsroom/id/3747517

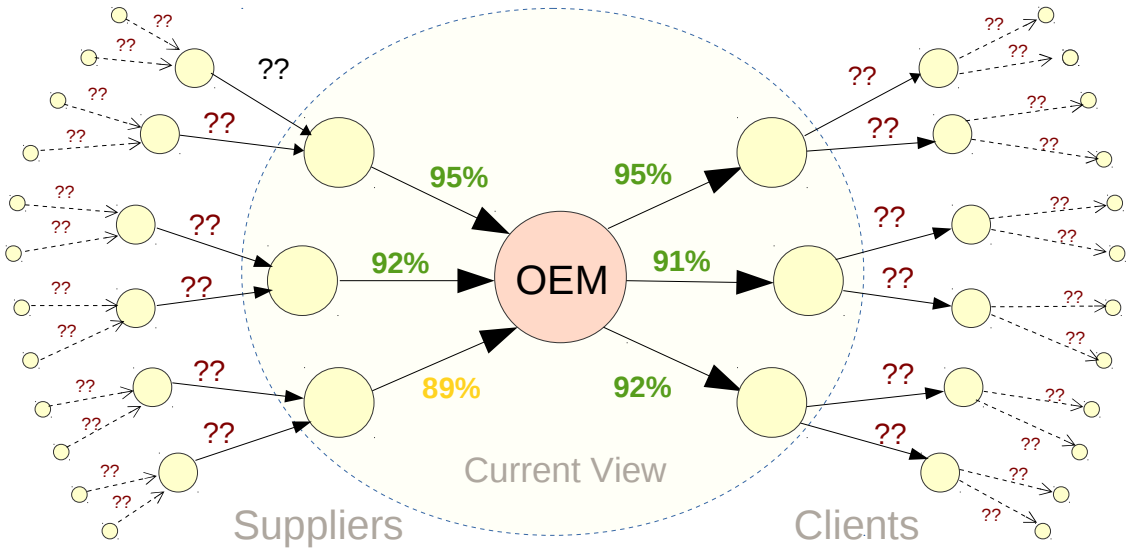


Figure 2.1: Supply Chain workflow example: small circles represent organizations, arrows represent flows of goods. The number next to the arrow is a sample value for example for counting the amount of goods delivered in full between two organizations. The dotted circle *current view* represents the knowledge of the selected organization *OEM* to its direct connection on the supplier side and client side. The question marks represent the uncertainty of that organization outside of their own view.

It is challenging to agree on a standardized way to represent knowledge about the business processes among organizations. The main reasons why they differ are:

1. viewpoint (or perspective)
2. need for process depth (granularity)
3. industrial domain
4. perception of process failures
5. availability of resources to manage supply chains

The *APICS Supply Chain Council*[2] tackles this challenge, and elaborates a reference model named *Supply Chain Operations Reference (SCOR)*[3] [7]. SCOR (see Figure 2.2) is a cross-industry approach to support a global process view defining a conceptual model for Supply Chain related information. The applicability of SCOR, however, is currently limited, since the standard stays on the conceptual and terminological level (document) and major effort is required for implementing the standard in existing systems and processes.

2.2 Industry 4.0

In 2011, the term “Industry 4.0” emerged to describe the need for the convergence of industrial production and information and communication technologies [12]. As of 2019, the term is widely used for marketing

[2] `http://www.apics.org/sites/apics-supply-chain-council/about-apics-scc`
[3] `http://www.apics.org/sites/apics-supply-chain-council/frameworks/scor`

Figure 2.2: High-level overview of Supply Chain Organizations Reference (SCOR version 11 [7]). Circles represent processes defined by SCOR, vertical dashed lines represent the boundaries between organizations.

purposes, and a growing number of scientific publications aims at fulfilling the goal.

Over time, various, mostly national, initiatives have emerged to improve the secure exchange of data among organizations. The *Industrial Internet Consortium* (IIC)[4] in the USA, the *International Data Spaces Association* (IDSA, formerly: Industrial Data Space Organization)[5] in Germany, the *Alliance Industrie du Futur* (AIF)[6], the *Industrial Value Chain Initiative* (IVI)[7] in Japan and the *Alliance of Industrial Internet* (AII)[8] in China. To the best of our knowledge, those are the largest initiatives based on the number of members.

Table 2.1 presents on overview of those five initiatives:

Created	Initiative	Started where	# Members	Infrastructure name
2014	IIC	USA	207	Industrial Internet of Things
2015	AIF	France	35	Industrie du Futur
2016	IDSA	Germany	88	International Data Spaces
2016	IVI	Japan	217	Industrial Value Chain
2017	AII	China	151	Industrial Internet Architecture

Table 2.1: Overview of the largest Industry 4.0 initiatives which focus on building the foundation for inter-organizational IT infrastructures. Member count assessed in March 2019.

They all have in common to be a consortium between academic institutions and private organizations. Furthermore, they are supported by the national government in order to prevent their economy from becoming dependent on foreign IT infrastructure.

In the following, we list the mission statements of each initiative, to get a clearer picture of their goals:

Mission statement by the International Data Spaces Association[9]:

[4] http://www.iiconsortium.org/
[5] https://www.internationaldataspaces.org/
[6] http://www.industrie-dufutur.org/aif/
[7] https://iv-i.org/
[8] http://en.aii-alliance.org/
[9] https://www.internationaldataspaces.org/our-approach/#data-exchange

> The International Data Spaces is a virtual data space that guarantees the secure exchange and easy linking of data in business ecosystems based on standards and joint governance models.

Mission statement by the Industrial Internet Consortium[10]:

> The Industrial Internet Consortium is the world's leading organization transforming business and society by accelerating the Industrial Internet of Things (IIoT).
>
> Our mission is to deliver a trustworthy IIoT in which the world's systems and devices are securely connected and controlled to deliver transformational outcomes.

Mission statement by the Alliance industrie du futur[11]:

> L'AIF organise et coordonne, au niveau national, les initiatives, projets et travaux tendant à moderniser et à transformer l'industrie en France. Sept groupes de travail dédiés et deux actions transversales (International et Filières) sont chargés de leur mise en œuvre.

Translated AIF mission statement translated into English with DeepL[12] and grammatical correction by the author.

> The AIF organizes and coordinates, at a national level, initiatives, projects and work aimed to modernize and transform the industry in France. Seven dedicated working groups and two transversal actions (international and activity sectors) are responsible for their implementation.

Mission statement by the Industrial Value Chain Initiative[13]:

> The Industrial Value Chain Initiative (IVI) is a forum to design a new society by combining manufacturing and information technologies, and for all enterprises to take an initiative collaboratively.

Mission statement by the Alliance of Industrial Internet[14]:

> As an important part and a key instrument in the transformation and upgrading of advanced manufacturing industry in China and abroad, industrial Internet is a major component in the strategic layout for "China Manufacturing 2025" and "Internet + Collaborative Manufacturing" Initiatives.

After all, they all have a common goal: To build an IT infrastructure that facilitates the secure exchange of data between organizations to support entirely new collaborative business models. They are all organized via working groups, led by a created legal consortium entity, and are financially supported by the countries they were founded. In particular, the goal is to establish more collaborative supply chains, to enable Smart Factories and Smart Manufacturing in general. They all see data produced by sensors as a core ingredient in their proposed architectures. They differ in their choice in technology and the focus in respect to security, roles, frameworks, data management and many more.

It is difficult to assess the state of the implementation for multiple reasons:

[10] http://www.iiconsortium.org/
[11] http://www.industrie-dufutur.org/
[12] https://www.deepl.com/
[13] https://iv-i.org/wp/en/about-us/whatsivi/
[14] http://en.aii-alliance.org/index.php?m=content&c=index&a=lists&catid=7

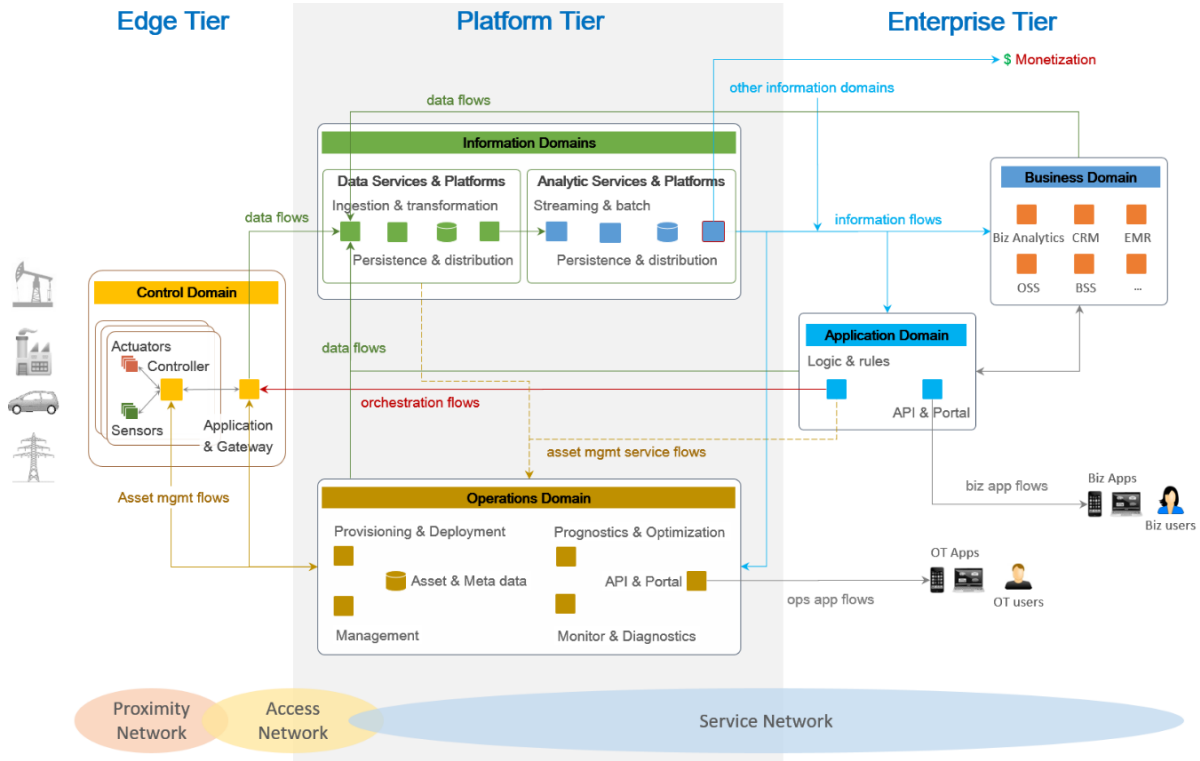


Figure 2.3: *Three Tier System Architecture* proposed by the Industrial Internet Consortium (IIC) [13]

1. In order to justify their funding via private and governmental sources, there are possibly incentives to publish every activity or result as a success story.

2. Their presentation of the system or reference architecture uses different levels of granularity which make it hard to fully access and compare their implementations.

3. Each organization has a closed members area, not all results of the working groups are publicly accessible.

Nevertheless, in the following, the system (or reference) architectures are presented including a brief summary on their focus. The need to give a more detailed comparison should be considered in future research.

Figure 2.3 presents the proposed *Three Tier IIoT System Architecture* [14] by the American Industrial Internet Consortium. What can be observed is that they separated the architecture into three levels: On the left, the data management is labeled the *Edge Tier*. In the middle, the new core technology of their activity is placed to support services, meta data management and stream & batch processing as part of the *Platform tier*. On the right, the integration of the previous tiers is proposed into the legacy enterprise software, with the possibility to develop new interfaces, as part of their *Enterprise Tier*.

Figure 2.4 presents the interaction of the technical components as described in the *Reference Architecture Model* [15] by the German International Data Spaces Association. Their focus is to provide multiple core components to support the creation of a decentralized *International Data Spaces*. The *IDS Connector* plays the role of a gatekeeper through which the data is offered and which consumes it from another IDS Connector of another organization. All legacy enterprise systems are behind those connectors of the respective organization. The *IDS Broker* provides a catalog for all participants such

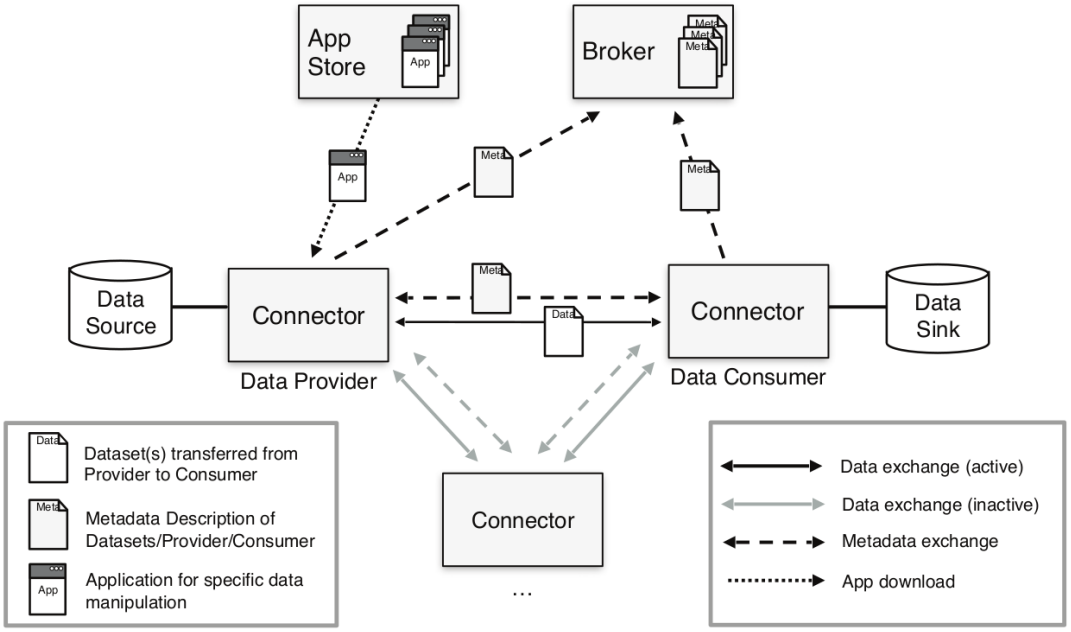


Figure 2.4: The proposed interaction of the technical components described in the *Reference Architecture Model* by the International Data Spaces (IDS) initiative [15]

that members can find each other and possible interesting data offerings. The *IDS AppStore* provides possible standardized applications for processing data and transforming data. All components are fueled by rich metadata descriptions.

To the best of our knowledge, an architecture similar to the ones proposed by the other initiatives is either not public or does not exist by the French Industry du Futur initiative. Multiple contact requests remained unanswered.

Figure 2.5 presents the *Connected Industrie Open Framework for Manufacturing* [16] provided by the Japanese Industrial Value Chain Initiative (IVI). Observed can be the interaction of the components with the internet and how the edge devices related to the introduced *Industrial Ethernet*. The proposed *Hyper Connection Terminal* manages the exchange of data between the Edge components through the internet to other *Hyper Connection Servers*. The proposed *Hyper Dictionary Server* managed the standardized terms & definition needed so different organizations can understand each others data.

Figure 2.6 presents the *Industrial Internet Architecture* [17] proposed by the Chinese Alliance of Industrial Internet. They propose an *Industrial cloud platform* which connects the factory technical system via the internet to other factories/organizations. Inside the factory, legacy software systems are places, including Edge computing nodes, which are supposed to handle the data flow outside the enterprise. The *Factory external network* offers a layer to build new products, end-user applications and a possible *Collaborative enterprise Information system*.

As a summary: The levels of granularity and the different focus makes a fair comparison hard. The general idea is the same: How can data be better used and exchanged among organizations. More research is certainly needed, to get to the depth of those initiatives and how they can be combined in a possible future global enterprise data space.

Section 6.3 describes the contributions of the author to the IDS. The section provides multiple component implementations and explains in two use cases how data is supposed to be exchanged in the IDS.

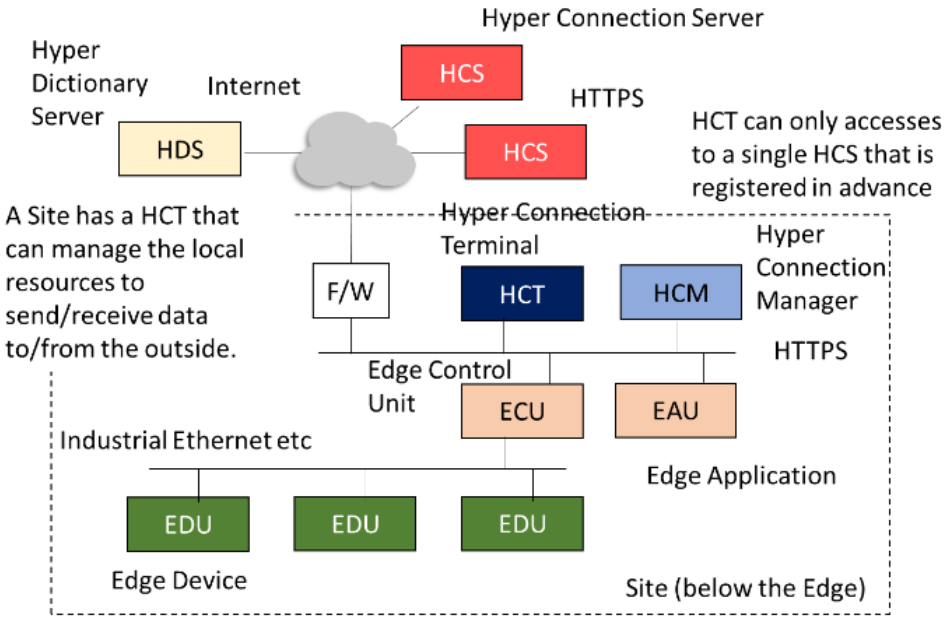


Figure 2.5: *Connected Industries Open Framework for Manufacturing* proposed by the Industrial Value Chain Initiative (IVI) [16]

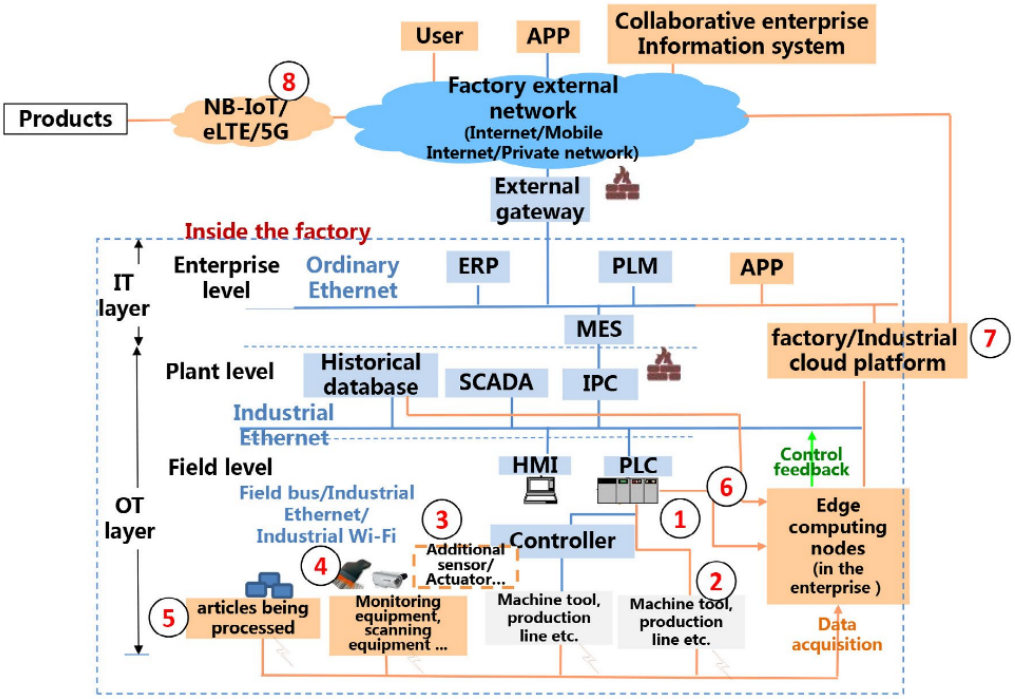


Figure 2.6: *Industrial Internet Architecture* proposed by the Alliance of Industrial Internet [17]

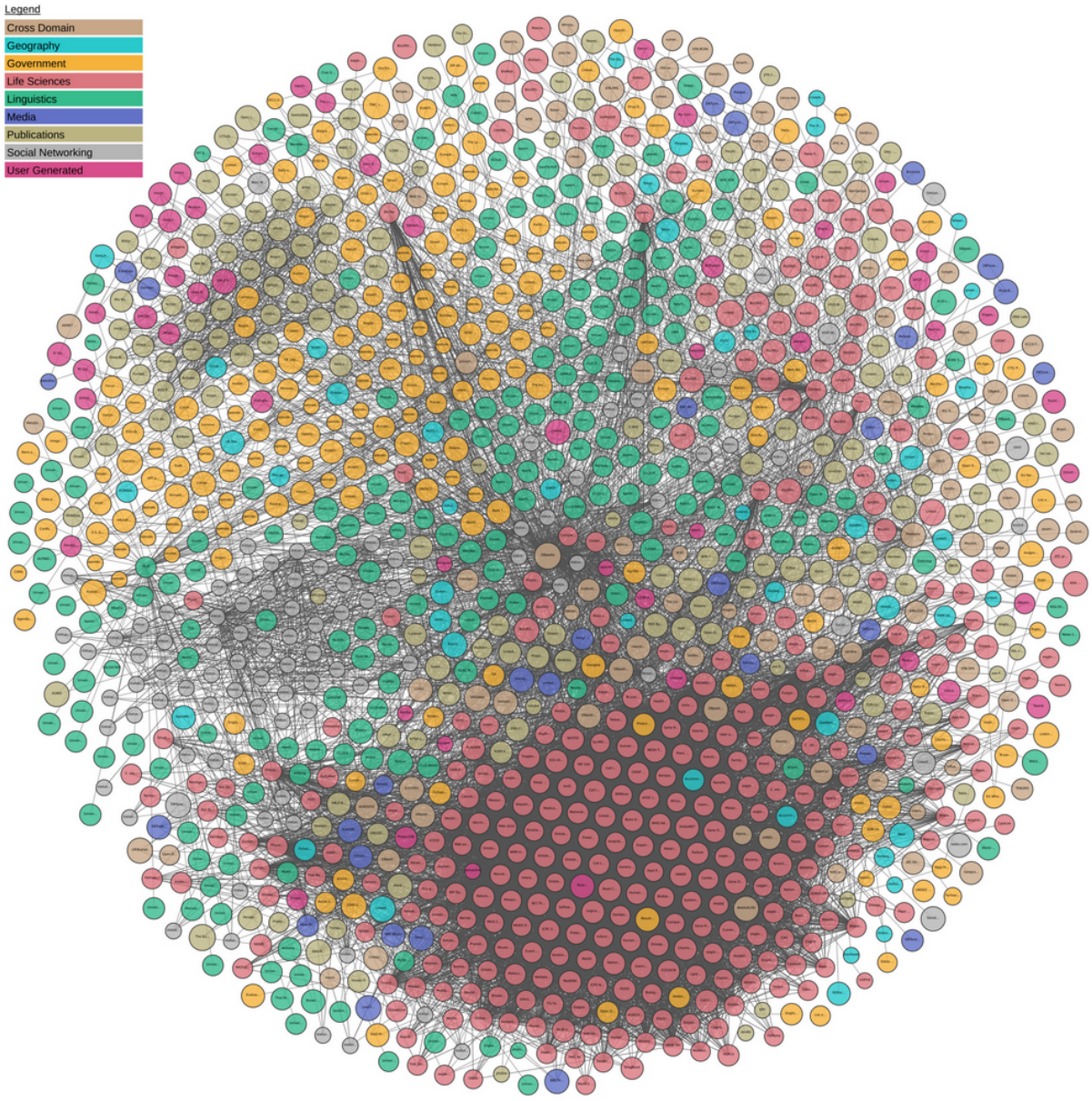


Figure 2.7: The Linked Open Data cloud (CC BY 4.0 `https://lod-cloud.net/`, March 2019). Each circle represents a different dataset. Each edge represents a link between them on a class or instance level.

2.3 Semantic Technologies

Semantic technologies represent a set of standards, protocols and technologies to "allow data to be shared and reused across application, enterprise, and community boundaries"[15]. In 2001, Tim Berners-Lee proposed the so called *Semantic Web* [3]. The vision is to form instead of a "web of documents" a "web of data" such that "computers can de more useful work"[16].

One of the key results of this movement is the Linked Open Data cloud (see Figure 2.7) which

[15] `https://www.w3.org/2001/sw/`
[16] `https://www.w3.org/standards/semanticweb/`

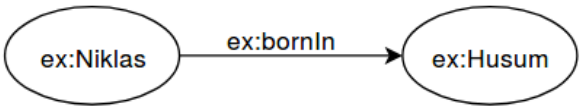


Figure 2.8: An graphical visualization of the RDF triple *Niklas-bornIn-Husum*. A source code representation of the same statement is available in Listing 2.1. The `ex:` represents the prefix of those triples. In this case, `:ex` represents the often used default prefix for testing purposes `www.example.org/`.

represents a growing amount of open datasets. These datasets are *interlinked* with each other, such as, entities which are considered equal in two or more datasets contain explicit links to each other.

In the following, we will describe the general method for expressing information according to the data formats and Knowledge Representation languages in the Semantic Web stack. Furthermore, we explain how the resulting data can be queried using the recommended query language.

2.3.1 Knowledge representation languages: RDF, RDFS, OWL

In 2004, the World Wide Web Consortium (W3C)[17] proposed three different knowledge representation languages for expressing data in the Semantic Web:

- **Resource Description Framework (RDF)** [18] which defines the core structure to express and organize information
- **Resource Description Framework Schema (RDFS)** [19] which provides a syntax for defining schemas
- **Web Ontology Language (OWL)** [20] for expressing logical axioms

RDF

The Resource Description Framework (RDF) provides the fundamental *framework* to *describe* information in web *resources*. A *resource* is considered anything with an identity: web pages, images, abstract concepts or numbers and strings. The abstract syntax (data model) of RDF is based on triples: *subject–predicate–object*. Figure 2.8 provides a visualized example of a triple: *Niklas* represents the subject, *bornIn* the predicate and *Husum* the object.

A set of triples form a graph. Thus, RDF is a graph-based data model. Nodes in an RDF graph can be either in the form of an *Internationalized Resource Identifier* (IRI), a *literal* or a *blank node*. However, subject nodes can only be in the form of an IRI or a blank node. Edges in an RDF graph (aka *predicates*) need to have the form on an IRI.

An IRI (generalization of an URI) is a string of characters (as defined in RFC 3987 [21]) which unambiguously identifies a particular resource. As an example: A resource named *bornIn* can be represented as an IRI (`http://www.example.org/bornIn`) or a literal (`"bornIn"`). However, they represent distinct resources. The additional information about the scheme (http) and the host (www.example.org) provide therefore resources in the form of an IRI with a global scope.

Literals represent in RDF values such as strings, numbers or dates. A literal consists of its lexical form (ex: "Niklas") and their datatype IRI (ex: string, integer, date). Optionally, language tags can be added to strings to specify to which language they belong to: `"born in"@en`.

[17] `https://www.w3.org/`

Blank nodes in an RDF graph are nodes which are neither defined by an IRI or a literal. Their lexical structure is not defined by RDF, but they are part of certain RDF syntaxes to allow an author of an RDF graph to not explicitly label every resource. An example where the subject represents a blank node, the predicate and IRI and the object a literal: `<123> <www.example.org/name> "Niklas"`. In contrast to IRIs, blank nodes are by their nature locally scoped.

The RDF Concepts vocabulary[18] defines the RDF namespace (list of terms) for the previously described fundamental concepts. The term `rdf:type` is further of importance for identifying the type of a resource, for example as an *rdf:Property*.

RDF triples can be serialized in various formats (also called syntaxes). The most popular one, the *Turtle syntax*[22], is used throughout this thesis. Listing 2.1 provides an example how an RDF triple visualized in Figure 2.8 can be expressed in the Turtle syntax.

```
1 @prefix    ex: <http://example.org/#> .
2 ex:Niklas ex:bornIn ex:Husum .
```

Listing 2.1: The source code syntax of the RDF triple *Niklas-bornIn-Husum* formalized in the Turtle serialization syntax. A visual representation of the same statement is depicted in Figure 2.8.

After all, RDF has by itself very little meaning. The next sections describe how RDFS and OWL extend RDF to provide formal semantics on RDF triples.

RDFS

The Resource Description Framework Schema (RDFS) vocabulary[19] is a semantic extension of RDF. Critical terms for this thesis introduced by RDFS are `rdfs:Class` `rdfs:subClassOf`, `rdfs:range`, `rdfs:domain`, `rdfs:comment` and `rdfs:label`.

Resources in an RDF graph can be typed using RDFS as classes and subclasses, thus enabling one to build hierarchical class definitions. As an example, a class *ex:Fish* can be expressed as a subclass of a class *ex:Animal*. RDFS provides the formal semantics to deduce that in this example, any instance of the class Fish is also an instance of the class Animal.

The range and domain of a defined `rdf:Property` is used to state to what class(es) the subject and object (or value) belong to in a particular triple. As an example, the range of a property named `ex:bornIn` may be specified as `ex:City` and the domain as `ex:Person`. The triple `ex:123 ex:bornIn ex:456` now supports the following deduction: `ex:123` is an instance of the class `ex:Person` and `ex:456` is an instance of the class `ex:City`. RDFS reasoners are applications that can make these deductions explicit, as in, they generate the additional triples to make the implicit semantics of RDFS explicit in an RDF graph.

Furthermore, RDFS provides the terms `rdfs:label` and `rdfs:comment` to provide resources with a human-readable label and definition.

OWL

The Web Ontology Language (OWL) 2 Schema vocabulary[20] further extends the semantics of RDFS to support logical statements (constraints). Critical concepts for this thesis introduced by OWL are:

[18] `http://www.w3.org/1999/02/22-rdf-syntax-ns#`
[19] `http://www.w3.org/2000/01/rdf-schema#`
[20] `http://www.w3.org/2002/07/owl#`

`owl:DatatypeProperty`, `owl:ObjectProperty`, `owl:unionOf`, `owl:disjointWith`, `owl:sameAs` and `owl:equivalentClass`.

To distinguish between properties whose range refers to data values or to instances of classes, OWL introduced `owl:DatatypeProperty`, `owl:ObjectProperty`. RDF triples which contain an object property as a predicate and a number as a value (or object) can thus be deduced by an OWL reasoner to be a semantic error.

To identify the equivalence of a class or an instance, OWL defined `owl:equivalentClass` and `owl:sameAs`. These properties are useful to specify that a class or an instance in one RDF graph is equal to a class or instances in another RDF graph possibly published by a different person or community. The Linked Open Data cloud [21] provides an example of more than a thousand RDF datasets which contain interlinked classes and instances. To identify disjointness between classes, OWL introduced `owl:disjoint`.

Furthermore, OWL allows to further specify certain characteristics of properties (`owl:TransitiveProperty`, `owl:SymmetricProperty`, `owl:FunctionalProperty`) and restrictions on them (`owl:Restriction`, `owl:allValuesFrom`, `owl:someValuesFrom`, `owl:onProperty`) and also supports cardinality constraints on their usage.

Overall, OWL semantics helps authors of vocabularies or ontologies to deduce triples in an RDF graph which contradict each other.

2.3.2 The SPARQL Protocol and RDF Query Language (SPARQL)

SPARQL is the recommended query language for RDF, RDFS and OWL by the World Wide Web Consortium (W3C). Similar to SQL, it allows the querying of RDF graph databases instead of relational databases.

There exist multiple SPARQL protocols, for retrieval (SPARQL Query), for updating an RDF graph (SPARQL Update), for federated queries on different SPARQL endpoints (SPARQL Federated Query). SPARQL SELECT 1.1 supports the following result formats: XML, JSON, CSV and TSV.

SPARQL 1.1 Query supports four types of selection queries: `SELECT`, `CONSTRUCT`, `ASK` and `DESCRIBE`. Select queries allow the retrieval of raw values from a RDF graph. Construct queries allow the retrieval of information from an RDF graph returning valid RDF. Ask queries allow simple True/False queries. Describe queries allows the retrieval of an RDF graph based on what the endpoint considered as useful information.

```
1  @prefix    ex: <http://example.org/#> .
2
3  ex:12345 rdf:type ex:Person .
4  ex:12345 ex:firstName "Niklas" .
5  ex:12345 ex:bornIn ex:Husum .
6  ex:Husum rdf:type ex:City .
```

Listing 2.2: An example RDF graph consisting of four statements.

Listing 2.2 displays a small knowledge graph, based on four RDF triples (statements). Listing 2.3 presents SPARQL SELECT query. The query searches for all variables `?firstName` in the RDF graph matching the `WHERE` pattern and returns their variable bindings. Such as, only a `?firstName` is returned, when all three statements in the `WHERE` clause are fulfilled. The displayed query returns the result *Niklas*.

[21] `https://lod-cloud.net/`

```
1  PREFIX    ex: <http://example.org/#> .
2
3  SELECT ?firstName
4  WHERE
5  {
6    ?person rdf:type ex:Person
7    ?person ex:bornIn ex:Husum .
8    ?person ex:firstName ?firstName .
9
10 }
```

Listing 2.3: SPARQL SELECT query example based on the knowledge base provided in Listing 2.2

2.3.3 Ontologies

Ontology is the study of *being* in Philosophy. Broadly, the study includes the identification of what concepts and categories exist and how they are related to each other. With regard to Computer Science, Gruber defines an ontology as "an explicit specification of a conceptualization" [23]. Gruber further points out that for a Computer Scientist, the study of being is reduced to what can be represented in a knowledge-based system. In the Semantic Web, the knowledge-based system are the previously described Knowledge representation languages RDF, RDFS and OWL.

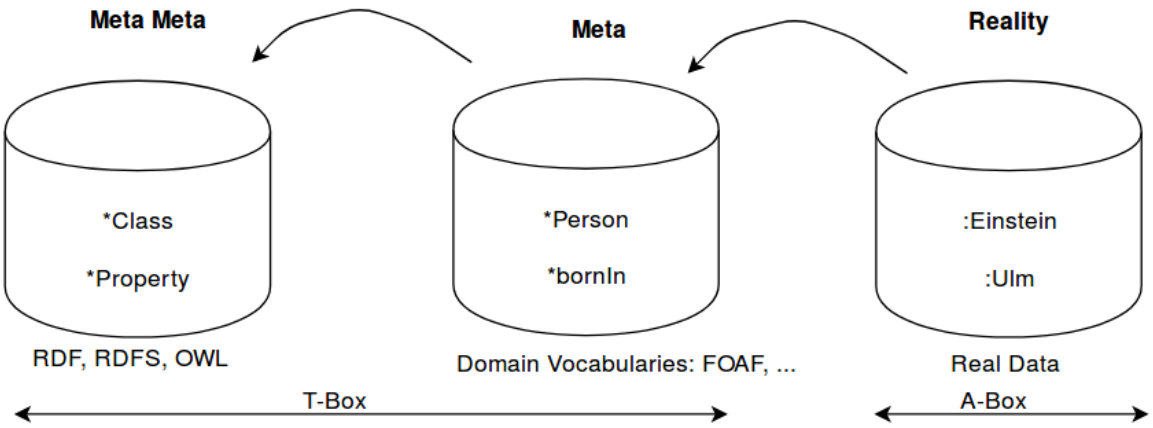


Figure 2.9: Placing the *Resource Description Framework (RDF)*, the *Resource Description Framework Schema (RDFS)* and the *Web Ontology Language* (OWL) into perspective: How they (left database symbol) relate to domain vocabularies (middle database symbol) and how they relate to real data (right database symbol).

Within the Semantic Web, there is no clear division between the usage of the term ontology and a vocabulary[22]. The term ontology is often used to describe more formal conceptualizations of a domain, while vocabularies refer to a more *lightweight* description on the concepts and their relations. Thus, if a particular domain is formalized using RDFS and OWL, it may be called either an RDFS vocabulary or and OWL ontology, even if the RDFS vocabulary contains OWL terms.

Figure 2.9 puts the previously described languages into perspective. It could be argued that RDF, RDFS and OWL form another meta layer (*meta meta*), based on which ontologies and vocabularies are built. Ontologies and vocabularies represent the meta layer to formalize and structure actual data (*reality*)

[22] https://www.w3.org/standards/semanticweb/ontology

for a particular domain. It should be noted that it can be hard sometimes to place certain concepts into one of those three proposed levels: *Meta Meta*, *Meta* and Reality

One of the design choices by the Semantic Web is to enable new vocabularies being built based on concepts of other vocabularies. As an example, in this thesis, the following vocabularies are frequently reused:

- **DCMI Metadata Terms** (dct:)[23] to enrich resources with additional metadata
- **Ordered List Ontology** (olo:)[24] to formalize lists in RDF graphs
- **R2RML: RDB to RDF Mapping Language Schema** (rr:)[25] to define mappings of relational data to RDF
- **NeoGeo Geometry Ontology** (ngeo:)[26] to describe geographical regions

Overall, the strong defined semantics of RDFS vocabularies and OWL ontologies are supposed to form the basies to "enable computers to do more useful work"[27], as the vision of the Semantic Web puts it.

[23] `http://purl.org/dc/terms/`
[24] `http://purl.org/ontology/olo/core#`
[25] `https://www.w3.org/ns/r2rml#`
[26] `http://geovocab.org/geometry#`
[27] `https://www.w3.org/standards/semanticweb/`

CHAPTER 3

Related Work

This chapter reviews previous efforts related to the research questions: First, we discuss proposed vocabularies to describe and evaluate Supply chains. Second, we review ontology authoring environments and tools which focus on the collaborative development of ontologies between domain experts and knowledge engineers. Finally, we discuss initiatives, projects and technologies that are related to the International Data Spaces initiative. In particular, we examine the proposed architectures, APIs, components and technologies to facilitate the exchange of data between organizations, to represent factories and to access relational data.

3.1 Supply Chain Vocabularies

As of version 11.0, SCOR provides 201 process definitions and 286 metrics to evaluate them. Various projects and activities aim at formalizing SCOR into a semantic model to support its execution. Table 3.1 provides an overview on the existing approaches.

	Version	Date	Published	Complete	Metric	Evaluation
Ye [24]	7.0	2005	–	n/a	n/a	–
Fayez [25]	7.0	2008	–	Assumed	Classes	–
Leukel [26]	8.0	2008	–	–	Classes	–
Sakka [27]	9.0	2011	–	Assumed	Classes	–
Zdravkovic [28]	n/a	2011	–	–	n/a	–
Lu [29]	7.0	2013	–	–	Classes	–
SCORVoc	11.0	2015	+	+	Queries & Properties	+

Table 3.1: Existing efforts for creating a vocabulary or ontology representing the Supply Chain Organization Reference (SCOR) specification.

For that purpose, SCOR defines different performance indicators (metrics) including a calculation plan to ensure comparability within the entire Supply Chain. In total, there are 286 metrics which are grouped into five categories: *Reliability*, *Responsiveness*, *Agility*, *Costs* and *Assets*.

Various projects and activities aim at formalizing SCOR into a vocabulary using RDFS and OWL [24–29].

The formalization of the SCOR model into an ontology is addressed in [26]. The authors analyzed the different conceptualization levels of the model and converted them into OWL classes.

In [28], a seminal approach formalizes Supply Chain operations overcoming the semantic weaknesses of the SCOR model. In this work the SCOR-KOS OWL model is presented, which encodes the main entities and properties for SCOR. In addition, the SCOR-Full ontology is a domain ontology for the representation and management of knowledge about Supply Chain operations. The latter presents the core concepts of Supply Chain embedded in SCOR definitions. This effort is also the basis used by Zdravković *et al.* [30] to configure the Supply Chain process. They provide a thread model configuring a specific flow of the Supply Chain studied.

The combination of the SCOR ontology and the *ONTO-PDM*[1] ontology is addressed in [29]. The ONTO-PDM ontology is used to represent information regarding product development, which is not covered by SCOR. The goal is to create a Supply Chain ontology framework for networked enterprises interoperability.

Sakka *et al.* [27] present a SCOR model as a way to align the business processes with strategical objectives for Supply Chains. Concepts like information and input/output are included to face this alignment. SCOR is modeled using ARIS[2] thus obtaining a SCOR/ARIS model. Then, XLST transforms the output of SCOR/ARIS into a SCOR OWL ontology.

The work conducted by [32] provides an ontology model to support Supply Chain process modeling and analysis based on the SCOR model. In this work, only part of the SCOR-KOS model [28] related to the definition of input and output entities in SCOR processes is reused.

However, most of these attempts have limitations, summed up in Table 3.1:

- First, these approaches are based on SCOR versions up to 9.0, while the current version is 12.0
- Then, the Linked Data principles are not satisfied by these works. They do not reuse existing vocabularies when possible, such as *schema.org* and *Dublin Core*. But above all, vocabularies of these works are not publicly available on the web. We contacted the authors in order to get access to their vocabularies, unsuccessfully.
- Most approaches choose to stay close to the hierarchical structure for processes and metrics given by the original source. By not defining the metrics as queries, it is not possible to directly calculate the metrics using their proposed ontologies.
- Finally, none of these vocabularies are 'operationalized', and enabled to automatically compute KPIs using data.

Chapter 4 reports on the development and evaluation of the SCORVoc vocabulary, that aims to formalize and operationalize the SCOR model, while alleviating these issues.

3.2 Ontology Authoring Environments

Distributed ontology development is an active research topic in the Semantic Web community [33]. One line of research is concerned with the development of web applications that offer low-barrier access to ontology editing.

A well-known example in this area is WebProtégé [34], a lightweight version of the Protégé desktop editor. It provides change tracking and collaboration features to support the distributed development of

[1] ONTO-PDM is an ontology for Product Data Management interoperability within manufacturing process environment, presented in [31]

[2] Business Modeling Approach

ontologies, complemented by a customizable user interface that can be adapted to the different expertise levels of the users. WebProtégé uses its own repository service based on the Protégé collaborative framework [35]. To keep track of changes, the client sends a ping request to the server every few seconds and checks if there are any new versions. In case of changes, the server provides the diff between the client and server version such that the client can decide what to keep or discard. Furthermore, a chat service is included, as well as the possibility to annotate ontologies in order to discuss or propose changes.

A related attempt is VocBench [36], which also includes a web editor for distributed ontology development. In contrast to WebProtégé, VocBench does not focus on editing OWL ontologies but is targeted towards SKOS thesauri. Its features include a change history, a validation and publication workflow, and a SPARQL query service. VocBench runs in an application server complemented by an RDF framework and triple store. It separates responsibilities through a role-based access control mechanism that handles the user privileges for the different tasks of thesauri editing. Other SKOS editors are *SOBOLEO* [37], PoolParty [38], and TemaTres[3]: SOBOLEO offers a special browser to navigate and change the thesauri, and a semantic search engine for annotating web resources. PoolParty provides a web interface to create and manage SKOS thesauri; in addition, it enables the extraction of relevant information from Linked Data sources. TemaTres includes an API to access the latest version of the developed ontology, a WYSIWYG editor, and support for quality assurance.

MoKi [39] is a collaborative MediaWiki-based tool to support the ontological modeling of business processes. It adds a wiki page, containing both unstructured and structured information, to each entity of the ontology and process model. The wiki principle is also adopted by OntoWiki [40] and Knoodl [41]: OntoWiki is based on an RDF framework and application server, and offers inline editing of RDF instances and classes. Knoodl combines structured ontology information and free-text wiki pages, with a focus on searching and linking to SPARQL endpoints.

While most of these approaches support history tracking, their implementations only cover a subset of the functionalities that are offered by sophisticated version control systems like Git and that have proved useful in distributed ontology development [42]. Instead of building an isolated solution, TurtleEditor aims at enhancing the existing features of repository hosting platforms. Furthermore, the idea of enabling the users to decide where the ontology should be hosted can be considered a driving factor of our development.

Apart from that, only few of the available web-based ontology editors allow for visual editing. Although several visualization plugins have been developed for the desktop version of Protégé[4] [35], they mostly do not allow for visual editing, cannot be reused in web applications like WebProtégé, and are only loosely synchronized with the rest of the ontology editor. A tightly synchronized textual and visual editing approach is currently not provided by any of the available RDF-based ontology editing tools to the best of our knowledge, i.e., the visual and text-based views in these editors are not permanently in sync but require reloading the graph visualization in case of any updates or changes. Also, clustering is not supported by most of the visual tools, which quickly results in large graph visualizations that are hard to read and handle. This holds similarly for diagrammatic approaches, such as Graffoo [43], or UML-based modeling tools, such as the Visual Ontology Modeler[5] or the Maestro Edition of TopBraid Composer[6].

[3] `http://www.vocabularyserver.com`
[4] `http://protegewiki.stanford.edu/wiki/Visualization`
[5] `http://thematix.com/tools/vom/`
[6] `http://www.topquadrant.com/tools/`

3.3 Industrial IT infrastructures

The section provides an overview on approaches based on semantic technologies to build industrial IT infrastructures.

3.3.1 Inter-organizational Frameworks

In the following, we discuss initiatives, projects and vocabularies that are related to the International Data Spaces initiative and the presented information modelling.

Broad Initiatives

Two major initiatives related to the Industrial Data Space are Plattform Industrie 4.0 (PI4.0, on a national level) and Industrial Internet Consortium (IIC, on an international level). Whereas PI4.0 aims to address all relevant architectural layers in Industry 4.0 contexts, the Industrial Data Space focuses primarily on the data layer. Accordingly, the Reference Architecture Model for Industry 4.0 (RAMI 4.0)[7] of PI4.0 can be considered rather vertical in scope, whereas the Reference Architecture Model of the Industrial Data Space and accordingly also the Information Model presented in this paper are more horizontally concerned with describing data for the interoperable exchange between enterprises [44]. The Reference Architecture of the Industrial Internet Consortium (IIRA)[8] targets systems in the Industrial Internet of Things (IIoT) and consists of similar architectural layers as RAMI4.0 and the IDS Reference Architecture. However, it does not include a dedicated Information Modelling layer but limits the architecture to a functional and implementation layer besides business and usage layers. The semantics of data are only marginally touched by the reference architectures of both PI4.0 and IIC.

Focused Initiatives and Projects

Apart from these broad initiatives, there are some more narrow-focused initiatives and projects that aim to facilitate the usage of heterogeneous data sources by defining a unified semantic layer. For instance, the Center for Expanded Data Annotation and Retrieval (CEDAR)[9] has developed a suite of tools supporting, among others, the annotation process and semantic search of datasets used in scientific experiments (biomedical studies). Sofia 2[10] is a platform that uses a *Semantic Broker* to ingest data based on ontologies and rules into a central storage database. Data-agnostic analytical methods are provided on the top of the central storage database as well as SDKs to build customized applications and various APIs to access the data. BIG IoT[11] envisions a platform to establish interoperability to already existing other IoT platforms. The data integration is supposed to be based on vocabularies to describe the different data sources. A standards-based BIG IoT API is planned to provide the basis for application development in an own marketplace. bIoTope[12] aims at enabling interoperability between *vertical IoT silos* by defining a standardized open API. The data integration is supposed to be based on vocabularies to describe the different data sources.

[7] https://www.zvei.org/en/subjects/industry-4-0/the-reference-architectural-model-rami-40-and-the-industrie-40-component/
[8] https://www.iiconsortium.org/IIRA.htm
[9] https://metadatacenter.org
[10] http://sofia2.com
[11] http://big-iot.eu
[12] http://www.biotope-project.eu

FIWARE[13] provides data through a RESTful API called NGSI[14]. Besides the claim to reduce JSON payload costs and a full REST adoption, it offers a more powerful query language especially for geospatial queries.

Vocabularies and Technologies

A variety of languages for (semantic) service descriptions were proposed. The recent smartAPI specification [45] proposes a catalog of extension attributes for the OpenAPI standard in order to allow for easier indexing and querying of RESTful interfaces. The Web Services Description Language (WSDL), Version 2.0 [46] emphasized the distinction of an abstract service interface (`wsdl:interface`) and actual services (`wsdl:service`) exposed via a number of protocol-specific endpoints (`wsdl:endpoint`). Closely related to our approach the iServe project adopted these concepts into a lightweight, semantic "Minimal Service Model" [47].

Finally, the Data Catalog Vocabulary (DCAT) [48] is a related W3C Recommendation making use of well-established vocabularies to describe the distribution of (static) data sets. However, the expressivity of DCAT is limited and does not cover aspects such as versioning or the temporal and spatial context; it does also neither include relations to originating organizations nor allow for the description of data-related service APIs.

3.3.2 Virtual Factories Frameworks

There are two categories of related work: i) existing frameworks that aim to describe factories as completely as possible, and ii) existing ontologies representing assembly lines.

Terkaj et al. [49] propose a Virtual Factory Data Model represented as an OWL ontology based on the Industry Foundation classes (IFC)[15] standard. The purpose of their ontology is to describe business processes that involve machines requiring specific resources. However, the advantage of modeling each concept twice, once as a class whose instances represent real occurrences (e.g. `IfcProduct`, `IfcProcess`), and then as a class whose instances are supposed to describe generic objects and types (e.g. `IfcTypeProduct` "describes a generic object type that can be related to a geometric or spatial context", `IfcTypeProcess` "describes a generic process type to transform an input into output") is not clearly justified. A significant part of the ontology employs such a logical duplication which is confusing for non-experts. Furthermore, the rationale of proposing property classes such as `VffProcessProperties` to "characterize processes" instead of using object or data properties is not described.

Chen et al. [50] propose a multi-agent framework to monitor and control dynamic production floors. The ontology, serialized in XML, is optimized for the communication between different agents. It describes Radio-Frequency Identification (RFID) tags [51] attached to factory objects and addresses requirements specific to a bike manufacturing use case. Although RFID sensors are an important component of Industry 4.0, the purely XML-based ontology without logical formalisms behind, as they are provided by RDF(S) and OWL, lacks semantics and does not allow for universal and convenient querying.

Büscher et al. [52] introduce the Virtual Production Intelligence platform based on the Condition Based Factory Planning (CBFP) approach. The authors developed an OWL-based CBFP ontology advocating "the decoupling of domain business logic and the technical implementation of a planning

[13] `https://www.fiware.org`
[14] `https://www.fiware.org/2016/06/08/fiware-ngsi-version-2-release-candidate/`
[15] `http://www.buildingsmart-tech.org/specifications/ifc-overview`

system" [52]. The ontology is relatively small, consisting of only five classes. As it is not available online, we consider the CBFP ontology rather abstract not useful for our implementation. Detailed evaluations and experiments are not provided, making it hard to assess the practical contribution of the work.

Kim et al. [53] propose an OWL ontology and an information sharing framework to allow collaborative assembly design. The heart of the ontology is the assembly line and its direct environment. The ontology defines assemblies and constraints leveraging capabilities of SWRL and OWL. However, the lack of a published online version prevented us from reusing it. Nevertheless, the conceptual design influenced the one of our ontology, i.e., several concepts in the classes hierarchy and a few properties have been recreated.

Ameri et al. [54] propose the Digital Manufacturing Market (DMM), a semantic web-based framework for agile supply chain deployment. DMM employs the Manufacturing Service Description Language (MSDL) at a semantic level. MSDL is an upper-level ontology expressed in OWL DL. Description Logic is extensively used to characterize supply and demand entities on several levels, such as the supplier, shop, machine and process levels. However, the granularity and ramification (especially for an upper-level ontology) impose restrictions on the usability, i.e., only a domain expert would have enough expertise to create a working model with accompanying queries. Furthermore, the ontology is again not available online, which prevented us from performing a thorough semantic analysis and considering an adaptation of concepts.

Zuehlke [55] introduces the SmartFactory initiative, which comprises best practices from the technical, architectural, planning, security and human dimensions. The initiative is envisioned to define and elaborate on the concept of *factory-of-things* as a vision of future manufacturing. Although semantic services involving ontologies and knowledge bases are claimed to be a part of the concept, the author does neither provide any examples nor references of such ontologies. Therefore, the presented concept is rather an implementation roadmap than a technical contribution.

3.3.3 Semantic Models

In this section, we give an overview on the development and usage of semantic models in related industrial scenarios:

Siemens developed an *ontology-based access* to their wind turbine stream data [56, 57]. The ontology serves as a global view over databases with different schemata. It thus enables SPARQL queries to be executed on different databases without having to take the different schemas into account. Statoil ASA also established a "single point of semantic data access" through an ontology-based data integration for oil and gas well exploration and production data [58]. They thus reduced the time-consuming data gathering task for their analysts by hiding the schema-level complexity of their databases. Ford Motor Company captures knowledge about manufacturing processes in an ontology such that their own developed AI system is able to "manage process planning for vehicle assembly" [59]. Furthermore, Ford examined the potential of federated ontologies to support reasoning in industry [60] as well as detecting supply chain risks [61]. Volkswagen developed a *Volkswagen Sales Ontology*[16] to provide the basis for a contextual search engine [62]. Renault developed an ontology to capture the performance of automotive design projects [63]. With regard to Ontology-Based Data Access (OBDA), Statoil chose the *Ontop* [64] framework because of its efficient query processing. While Siemens initially favored Ontop as well, they developed their own system in the end to further optimize stream data processing. Based on these experiences and our own tests of OBDA tools (mainly Ontop and *D2RQ*[17]), we chose Ontop as well. Regarding semantic models for companies, none of the existing works has specifically addressed

[16] `http://www.volkswagen.co.uk/vocabularies/vvo/ns`
[17] `http://d2rq.org`

machine tools and factory infrastructures. While it is understandable that companies prefer not to share internal details of their methodologies and infrastructure, there is nevertheless very limited evidence of semantic technologies being deployed in the manufacturing industry.

CHAPTER 4

Semantic Supply Chain

This chapter is dedicated to solve one of the core challenges of this thesis, i.e., to explore how semantic technologies may be used to improve the management of Supply Chains. Furthermore, how Supply Chain processes can be better describe using semantic technologies, to support inter-organizational Supply Chain Management. The content of this chapter is based on the publications [65–68][1]

This chapter answers the following research question:

> **RQ1**: How can the Linked Data paradigm be used to represent Industrial Data to improve Supply Chain Management?

4.1 SCORVoc –An RDFS vocabulary for managing Supply Chains

This section describes how the SCOR[2] standard in formalized as an RDFS vocabulary, called *SCORVoc*.

4.1.1 Methodology

As seen in the Section 3.1, the main issues in existing formalizations of SCOR is that they have not been developed for a specific version of SCOR, and cannot be continuously updated. On the contrary, we develop SCORVoc using the *VoCol* methodology and support environment [69][3] based on the Git version control system. This makes SCORVoc a living artefact, which can be extended and revised by a community of collaborators.

Multiple people were involved in the development of SCORVoc. In order to facilitate the collaborative development, we choose a GitHub repository[4] for managing vocabulary source files, documentation, queries as well as example data. The Turtle serialization format [22] was chosen due to its simplicity. GitHub's web interface further provided our domain experts with a very simple way to access the latest SCORVoc version.

Additionally to the collaborative ontology development environment provided by VoCol, we followed the methodology proposed by Uschold *et al.* [70] for building the vocabulary:

[1] [66] and [67] is a joint work with Marvin Frommhold, a software engineer at eccenca GmbH. Only minor parts are based on these publications. In these papers, I contributed to the development of the requirements and the architecture.

[2] `https://www.apics.org/apics-for-business/frameworks/scor`

[3] `https://github.com/vocol/vocol`

[4] `https://github.com/vocol/scor`

i) define the purpose and scope; ii) capture the domain knowledge; iii) develop the ontology and integrate it with other existing vocabularies; iv) evaluate it and document it properly.

Purpose and Scope. The purpose of SCORVoc is to provide enterprises with a vocabulary which they can use to express any data related to Supply Chain management (SCM). The users of the vocabulary are therefore enterprises which would like to profit from the benefits of expressing their supply chains in SCOR. The vocabulary aims at being light-weight in order to facilitate its usage for future SCOR compliant IT applications.

Capture. The capture of the domain of interest (Supply Chain data management) was achieved in two ways. First, we used the 976-page SCOR reference [7], with its strong terminological definitions as a major source for studying the domain of interest. Second, we had a domain expert with a deep knowledge (a member of the *APICS Supply Chain Council*[5]) which supported us in the entire process. As a result of many interviews with the domain expert, we acquired a more fundamental understanding of the motivation of SCOR, its strengths but also its weaknesses.

Design. This step is decomposed in the design of the two main aspects of SCOR: processes, described in Subsection 4.1.2; and metrics, described in Subsection 4.1.3. Existing semantics for each concept and the lack of accessibility of prior approaches to formalize SCOR lead us to create many concepts by ourselves. Nevertheless, various concepts and properties are integrated from well-known vocabularies such as *schema.org*, *SKOS* and *Dublin Core*. We made this decision based on the semantic description of these terms.

Documentation and Evaluation. As illustrated in Subsection 4.1.2 and Subsection 4.1.3, each concept contains a definition together with a full documentation based on the descriptions provided by SCOR. Subsection 4.1.4 describes the evaluation of SCORVoc.

4.1.2 Formalizing Processes

In SCOR, there are the 201 processes. A process represents any business activity between and within enterprises. For most of them, the reference outlines unambiguous text definitions. Since some of them have a rather long name (e.g. *Identify, Prioritize And Aggregate Supply Chain Requirements*), we decided to keep the short name and attach the long version as a label. To stay coherent, all concept and property names follow the camel case notation.

As proposed in the reference, we created the processes as a hierarchical structure. We defined an abstract super class *Process* with its subclasses *Plan*, *Source*, etc. While certain terms (e.g. *Make*, *Deliver*) do not seem to be self-explanatory, we nevertheless adopted them due to the clear meaning in the SCM domain. All together, the hierarchy consists of three levels (Scope, Configuration, Step). Each level fulfills a certain purpose:

- Level 1 groups processes together,
- Level 2 comprises events, that are to be instantiated in the real world, and
- Level 3 explains in detail how level 2 processes are to be executed (step by step).

Furthermore, the reference defines IDs and clear text definitions for each process. These are reused as is in SCORVoc. Figure 4.1 visualizes the general structure of the vocabulary with the processes and others main concepts.

Listing 4.1 shows an example for the full definition of the concept *Enable*. *Enable* is a subclass of the abstract concept `scor:Process`. Each concept contains a definition together with further descriptions provided by SCOR. We further added translations for a variety of languages.

[5] `http://www.apics.org/sites/apics-supply-chain-council`

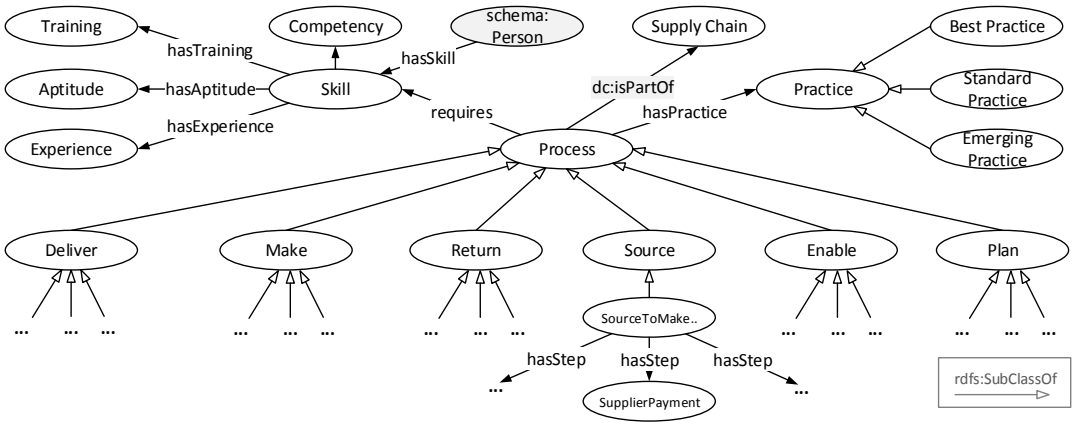


Figure 4.1: Visualization of the proposed *SCORVoc* vocabulary (the namespace prefix `schema:` refers to schema.org and `dc:` to the Dublin Core Metadata Initiative).

There are some specificities in SCOR that required the addition of more fined-grained knowledge in the SCORVoc vocabulary.

```
1  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
2  @prefix skos: <http://www.w3.org/2004/02/skos/core#> .
3  @prefix xsd:  <http://www.w3.org/2001/XMLSchema#> .
4  @prefix     : <http://www.purl.org/eis/vocab/scor#> .
5
6  :Enable rdfs:subClassOf :Process ;
7          rdfs:comment    "Enable describes the ..."@en;
8          rdfs:label      "Enable"@en ,
9                          "Permitir"@es ;
10         skos:notation   "E" .
```

Listing 4.1: An shortened example of a class definition in *SCORVoc*: The SCOR process `:Enable`.

As an example, SCOR level 3 processes (also called *steps*) have an order of execution of these processes. Therefore, we used the Ordered List Ontology`http://smiy.sourceforge.net/olo/spec/orderedlistontology.html` property `olo:next` to express this relation in our ontology.

SCOR further defined 179 *Practices* to provide a collection of industry-agnostics practices which were recognized for their value. In order to manage talent in the supply chain, concepts such as *Person*, *Skills*, *Experiences*, *Aptitudes* and *Trainings* were introduced. While these concepts are not necessarily needed for the calculation of KPIs, they are all added for sake of completeness.

4.1.3 Formalizing Metrics

Metrics are defined by SCOR to evaluate Supply Chains on certain aspects such as reliability or responsiveness.

Similar as the aforementioned processes, the SCOR reference organizes metrics into a hierarchical structure, level 1 being the highest, and level 3 the lowest. SCOR provides for each metrics in a level a calculation plan that takes as an input the value of certain metrics in a lower level.

Let us describe one selected metric for each category [6]:

[6] For reference, we added SCOR identifiers in parentheses where applicable.

Reliability Level 2 metric *Orders delivered in full* (RL 2.1) of performance indicator *Reliability* measures whether orders are received by the customer in the quantities committed. Its calculation plan is:

$$\frac{\#Orders\ delivered\ in\ full}{\#Orders\ delivered} * 100\%$$

An order is considered as delivered in full once it contains the correct items (RL 3.33, level 3 metric) with the correct quantity (RL 3.35, level 3 metric).

Responsiveness Level 1 metric *Order Fulfillment Cycle Time* (RS 1.1) measures the average cycle time in days it requires to achieve customer orders. Its calculation plan is:

$$\frac{\sum Actual\ Cycle\ Times\ for\ All\ Orders\ Delivered}{\#Orders\ Delivered}$$

It depends on multiple other metrics of level 2 such as the time it takes to procure goods and services (RS 2.1), its production time (RS 2.2), the delivery and the delivery retail time (RS 2.3).

Agility The metric *Upside Supply Chain Flexibility* (AG 1.1) counts the number of days to achieve an unplanned increase (20% suggested by the reference) in the output of Source, Make and Deliver components.

$$\max(Source, Make, Deliver, SReturn, DReturn)$$

By assuming the production can run concurrently (one strategy provided by SCOR), it requires to identify the process (see metrics AG 2.1-5) within the enterprise whose adaption takes the most time.

Costs The metric *Production Cost* (CO 2.004) accounts for all costs involved in the production process.

$$\sum Labor + Automation + Property + Inventory$$

Thus, it depends on metrics which assemble the labor costs (CO 3.014), the automation costs (CO 3.015), the property, plant and equipment costs (CO 3.016) and the governance, risk, compliance, inventory and overhead costs (CO 3.0017).

Assets The metric *Cast-to-Cash Cycle Time* (AM 1.1) represents the time it takes for an enterprise to earn money on raw material investments.

$$\sum SalesOutstanding + Inventory - PayableOutstanding$$

Thus, it is necessary to summarize the days between a sale is made and the cash is received (AM 2.1) with the days of sales they were in the inventory (AM 2.2). That sum needs to be subtracted with the days between purchasing raw materials and their actual payment (AM 2.3).

Previous approaches defined a concept for each metric. However in SCORVoc, metrics are represented as data properties, and their calculation plan is represented as an SPARQL query. SCOR metrics are hence ‘operationalized’, in the sense that all information required for computing the metric is made available in an interoperable way (viz. as Linked Data) and the metrics itself can be translated into queries operating on this information (i.e. SPARQL queries [71]) and returning the respective KPI.

The level 3 metrics are the *data capture* entry point in SCORVoc. They are hence defined as data properties. Their `rdfs:domain` points to their respective processes (given by SCOR) and their range is `xsd:decimal` since they all describe a number between 0-100 (percent values).

```
1  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
2  @prefix skos: <http://www.w3.org/2004/02/skos/core#> .
3  @prefix  xsd: <http://www.w3.org/2001/XMLSchema#> .
4  @prefix     : <http://www.purl.org/eis/vocab/scor#> .
5
6  :hasMetricRL_33 a              owl:DatatypeProperty ;
7                  rdfs:comment   "Percentage of orders ..."@en ;
8                  rdfs:label     "Delivery Item Accuracy"@en,
9                                 "Exactitud en Entrega de Items"@es;
10                 skos:notation  "RL.3.33" ;
11                 rdfs:range     xsd:decimal ;
12                 rdfs:domain    :ItemAccuracyProcesses .
```

Listing 4.2: An shortened example of a datatype property definition in *SCORVoc*: The SCOR reliability metric *Delivery Item Accuracy* (`:hasMetricRL_33`). The number 33 represents the identifier within the SCOR spec.

Listing 4.2 shows an example for the full definition of the property `hasMetricRL_33`. Equally as for the documentation of processes, we expressed each property with the definition and the additional information provided by SCOR.

Then, the level 1 and 2 metrics are formalized as SPARQL queries. When triggered, they compute the value of the metric using the values of lower level metrics. Subsection 4.1.4 contains multiple examples of such SPARQL metric queries.

4.1.4 Evaluation

We evaluate our approach using a qualitative and quantitative method based on the metrics discussed in the previous section. Our qualitative evaluation describes the usage of the SPARQL metrics in a business scenario. In order to do a quantitative evaluation, we developed a SCOR test data generator and measured the execution time of typical queries.

Qualitative evaluation

Listing 4.3 demonstrates a simple example of data expressed using SCORVoc. Since the reference limits itself entirely to processes and performance indicators, we added additional information (using the namespace prefix `ex:`) to make the example more realistic.

The example describes a scenario (`:process_1`) in which goods (viz. keyboards) are received by an enterprise on a certain date. These goods are forwarded to the warehouse which is the reason for the classification of the process as a `:SourceStockedProduct`. Alternatively, if these goods are directly used in the production or for specialized client orders, the process may have been classified as `:SourceMakeToOrderProduct` or `:SourceEngineerToOrderProduct`. This enables enterprises to distinguish more easily between possible unnecessary orders, which end up as *dead capital* in the stock.

As a next step, all information related to this event is captured. `:hasMetricRL_33` represents the accuracy of the items and `:hasMetricRL_50` the quantitative accuracy. Thus, our example shows that this order achieved a quantitative accuracy of 90%.

```
1  @prefix xsd: <http://www.example.org/#> .
2  @prefix  ex: <http://www.example.org/#> .
3  @prefix    : <http://www.purl.org/eis/vocab/scor#> .
4
```

```
5  #--- General information --------------
6  :process_1 ex:isSubjectOf   "Keyboard X3" ;
7             ex:hasTimeStamp "2015-10-01T12:52:16"^^xsd:dateTime ;
8             ex:hasSupplier  ex:Logitech ;
9             ex:hasCustomer  ex:Dell .
10
11 #--- SCOR Information -----------------
12 :process_1 a               :SourceStockedProduct ;
13            :hasMetricRL_33 100 ;
14            :hasMetricRL_50  90 .
```

Listing 4.3: Example data compliant to *SCORVoc*. It expresses the definition of a process instance called `:process_1` which is of the type of the SCOR process *Source Stocked Product*. The process instance contains certain measure metric values and additional information for an imaginary delivery: product information, supplier and customer organization and a time stamp.

Once the Supply Chain related information is captured using SCORVoc, the execution of SPARQL query metrics becomes feasible. As described in Chapter 3, this was the driving force in the development of the SCORVoc vocabulary. Listing 4.4 shows the *Perfect Order Fulfillment* SPARQL metric. The query compares all complete deliveries (achieving 100%) with all deliveries in total by relying on the respective properties. Applied on the previous example, it returns 0% due to the delivery being incomplete.

The knowledge of these metrics is considered to become a major competitive advantage in the enterprise world.

Besides the previous data example, we will present and briefly discuss how the metrics are realized as SPARQL queries in the following:

Listing 4.4 presents the *Orders Delivered In Full* metric. Orders are considered to be delivered entirely by SCOR once their item accuracy (RL 33) and quantitative accuracy (RL 50) correspond to 100%. Thus, every order below that value is regarded as incomplete.

```
1  SELECT ((xsd:decimal(?full)  /
2          (xsd:decimal(?notFull))
3             * 100) as ?result)
4  WHERE { {SELECT ((count(?deliveredInFull)) as ?full)
5            WHERE { ?deliveredInFull :hasMetricRL_33 100 .
6                    ?deliveredInFull :hasMetricRL_50 100 . }}
7           {SELECT ((count(?allDeliveries)) as ?notFull)
8            WHERE { ?allDeliveries a :Process . }}}
```

Listing 4.4: Formalized SPARQL query of the SCOR KPI *Orders Delivery in Full*.

The *Order Fulfillment Cycle Time* is calculated by collecting the respective sum (days) of all source (RS 21), make (RS 22), deliver (RS 23) and deliver retail (RS 24) processes and divides them by amount of all orders.

```
1  SELECT ((xsd:decimal(?actualTime)) /
2          (xsd:decimal(?allOrders)) as ?result)
3  WHERE { {SELECT (sum(xsd:decimal(?value)) as ?actualTime)
4           WHERE { ?order :hasMetricRS_21
5                          |:hasMetricRS_22
6                          |:hasMetricRS_23
7                          |:hasMetricRS_24  ?value . }}
8           {SELECT (count(?order) as ?allOrders)
9            WHERE { ?order a :Process . }}}
```

Listing 4.5: Formalized SPARQL query of the SCOR KPI *Order Fulfillment Cycle Time*.

For the calculation of the *Upside Supply Chain Flexibility* metric, it is necessary to gather the value of all flexibility properties (AG1-5) and identify the maximum among them. Similar as a team is only as strong as its weakest part, a Supply Chain is only as agile as its slowest link.

```
1  SELECT (max(xsd:decimal(?flexibility)) as ?result)
2  WHERE { ?order :hasMetricAG_1
3                 |:hasMetricAG_2
4                 |:hasMetricAG_3
5                 |:hasMetricAG_4
6                 |:hasMetricAG_5  ?flexibility . }
```

Listing 4.6: Formalized SPARQL query of the SCOR KPI *Upside Supply Chain Flexibility*.

The *Production Cost* metric (CO 2.004) is dependent on the sum of the metric properties for the costs in Labor (CO 14), Automation (CO 15), Property (CO 16) and Inventory (CO 17).

```
1  SELECT (sum(xsd:decimal(?costs)) as ?result)
2  WHERE {  ?order :hasMetricCO_14
3                  |:hasMetricCO_15
4                  |:hasMetricCO_16
5                  |:hasMetricCO_17  ?costs . }
```

Listing 4.7: Formalized SPARQL query of the SCOR KPI *Production Cost*.

Listing 4.8 presents the *Cast-to-Cash Cycle Time* metric (AM 1.1). The query selects the average time raw materials stays in inventory (AM 2) together with the time the payment is due to by us (AM 1) subtracted by that of our customers (AM 3).

```
1  SELECT (avg(xsd:decimal(?inventoryDays))
2        + avg(xsd:decimal(?salesOutstanding))
3        - avg(xsd:decimal(?payableOutstanding)) as ?result)
4  WHERE { ?order :hasMetricAM_1  ?salesOutstanding .
5          ?order :hasMetricAM_2  ?inventoryDays .
6          ?order :hasMetricAM_3  ?payableOutstanding . }
```

Listing 4.8: Formalized SPARQL query of the SCOR KPI *Cast-to-Cash Cycle Time*.

Quantitative evaluation

We demonstrate the feasibility of our approach on a SCOR dataset. The only known dataset is *SCORmark*[7], which is assembled by a major consulting firm, but is only available for business customers and not open for research. We hence developed a open-source synthetic benchmark data generator, available on GitHub[8].

This generator allows to perform a round-trip between data representation and KPI evaluation. The benchmark allows to assess the performance of an SCORVoc implementation in a systematic and repeatable way. The generator creates data based on a number of parameters: Supply Chain depth, industry and Supply Chain partners. The Supply Chain depth sets the level from one main OEM enterprise to it suppliers' supplier. The industry generates plausible product lines and enterprise names. The Supply Chain partners determine the width of the Supply Chain. A minimum of 2 generates a binary tree to both sides.

Various dataset sizes can be generated in order to assess the scalability as well as the correctness of

[7] http://www.apics.org/sites/apics-supply-chain-council/benchmarking
[8] https://github.com/vocol/scor/generator

Processes	Instances	Metric Properties
Source	1.423	28.576
Make	1.785	23.216
Deliver	2.083	22.917
Plan	1.538	18.462

Table 4.1: Generated data for our experiments: 100k RDF triples example.

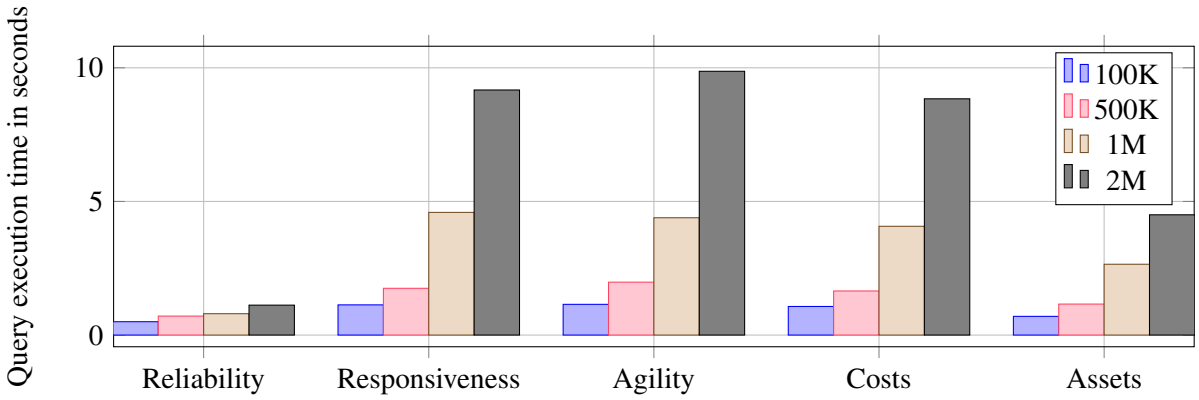


Figure 4.2: SPARQL query execution results in seconds for testing the performance of five SCOR KPIs. Four datasets of different size were created: 100k, 500k, 1M and 2M RDF triples.

the metric SPARQL query implementations. We evaluated the metrics for datasets which contain 100k, 500k, 1M and 2M triples. Table 4.1 presents an overview of the generated data for the 100k scenario. While the instances represent different types of processes, the related properties are mostly 3-level data type properties which are required by the metrics (such as `scor:hasMetricRL_50`). The values randomized within a certain range (e.g. >80% for Reliability).

The queries were executed using the *ARQ* SPARQL processor[9]. The machine we used for the experiment contains 8GB of RAM, 256GB SSD and an Intel i7-3537U CPU with 2.00GHz.

Figure 4.2 summarizes the results of the quantitative evaluation. The *Reliability* and *Assets* queries performed best in our scenario and were also able to be executed fast with datasets larger than 2M triples. The other queries do not scale as well, but still perform with less than 10s query execution time. This is enough for real-world settings. Even for much larger supply networks, we deem query execution performance not to be a bottleneck, since queries are executed relatively infrequently and not by thousands of users. Also, since the number of metrics is limited it is possible to optimize query execution even more, but creating specific indexes or applying caching strategies. Overall this evaluation shows that the approach of having an executable vocabulary is feasible.

Furthermore, the SCOR data generator can be used to systematically assess and evaluate SCORVoc compliant software solutions. Such solutions could be Specific supply network visualizations tools, Supply Chain robustness assessment frameworks, or scenario planning tools.

In contrast with Figure 2.1, Figure 4.3 represents the full view on the entire Supply Chain.

[9] `https://jena.apache.org/documentation/query/`

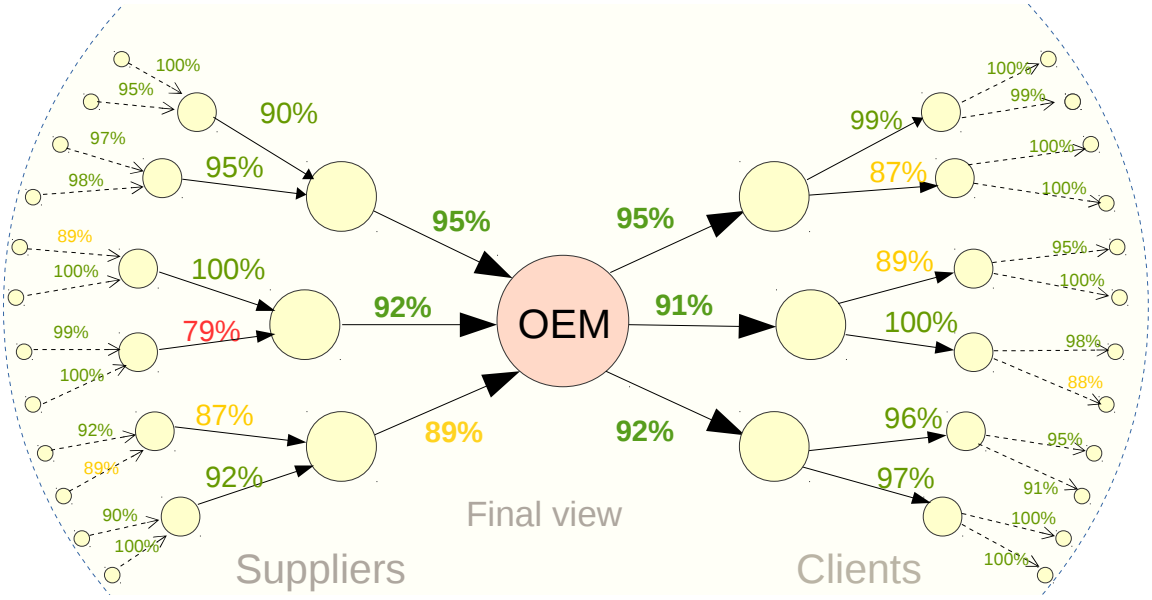


Figure 4.3: Final possible result view on the Supply Chain, where KPIs are calculated with the *SCORVoc* SPARQL queries and are propagated through the network. The colors represent possible threshold for acceptable values (green), warning values (yellow) and danger value (red).

4.2 A Prototype for Federated, Semantics-based Supply Chain Analytics

This section presents the developed prototype called *scmApp* which makes use of *SCORVoc* data and provides a user interface to manage Supply Chains.

4.2.1 Requirements

Our requirements are driven by the *Linked valUe ChaIn Data* (LUCID) research project [10] which aims at improving the information flows between highly connected enterprises. Furthermore, a global operating manufacturer provided us with useful insights on day-to-day supply chain challenges and helped us to gain a better understanding of the SCOR standard. Instead of human intervention, the rationale is to automate data exchange of logistical information, order management or event propagation. Automating information exchange can, for example, help preventing the bullwhip effect, which causes supply chain inefficiencies due to excessive production line adaption as a response to business forecasts [72]. To attain the goal of an interactive, federated analysis tool for supply chain management, we first collect all requirements it needs to fulfill.

These are the requirements:

R1 Data sovereignty The major requirement is that each enterprise keeps the full sovereignty of its data. This requirement precludes systems based on external data hubs or cloud solutions.

R2 Secure data access Giving external enterprises access to one's own information system requires secure interfaces.

[10] `http://www.lucid-project.org/`

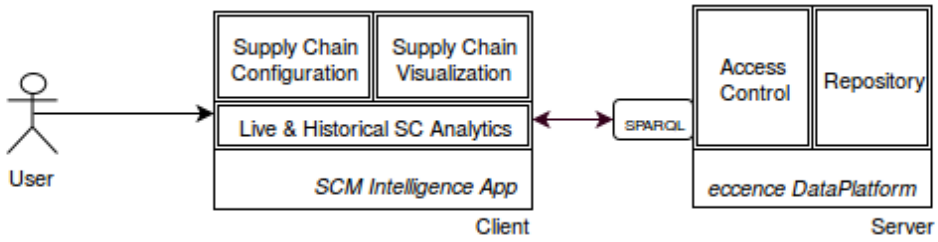


Figure 4.4: Client-Server Architecture to enable the calculation of KPIs with the proposed *SCM Intelligence App*. The server holds the *SCORVoc* data while the client sends SPARQL queries of the respective SCOR KPI selected by the end user.

R3 Configurable supply chain networks Since every supply chain partner maintains a different view on the entire network, it is important that each partner can configure its own view in the system.

R4 Customizable data models Since there is no single globally accepted process model, one needs to be able to customize and extend the data models.

R5 Customizable analysis techniques key performance indicators (KPIs) and analysis methods vary by industry, region and abstraction model.

4.2.2 Architecture

Figure 4.4 displays the overall architecture of our approach including the services that a server is going to provide and which the front-end takes advantage of. From the host's SPARQL endpoint, SCM Intelligence App is able to retrieve specific information relevant to the supply chain, or to compute KPI metrics, implemented as SPARQL queries, directly on the endpoint. The provider of the server thus provides access control (via standard HTTP authentication mechanisms) and a triple store. The SCM Intelligence App provides three services in total described in the sequel.

Configuration. First, the application allows each enterprise to configure their individual view on the supply chain. To achieve this, they are required to specify the endpoint and the credentials of each node in the supply chain. Furthermore, every connection within the supply chain needs to be described, that is: Which enterprise is the supplier/client of another enterprise.

Visualization. Second, once the information is specified in the configuration step, the application is able to generate a supply chain visualization. This visualization generally enables users to view supply chain information related to different aspects (e.g. supply chain throughput, health) and on different levels of granularity (e.g. tier 1 suppliers). In particular, it enables users to identify critical links and suppliers/clients.

Analytics. Third, the user can run pre- or self-defined analytic methods to compute KPIs on the entire supply chain. This main service helps to identify weak performing links which then can be either optimized or replaced. Thus, helping to keep the entire supply chain efficient and competitive.

4.2.3 Implementation

Our implementation follows the client-server model. We first introduce how the client is implemented and then describe the server including the services it provides.

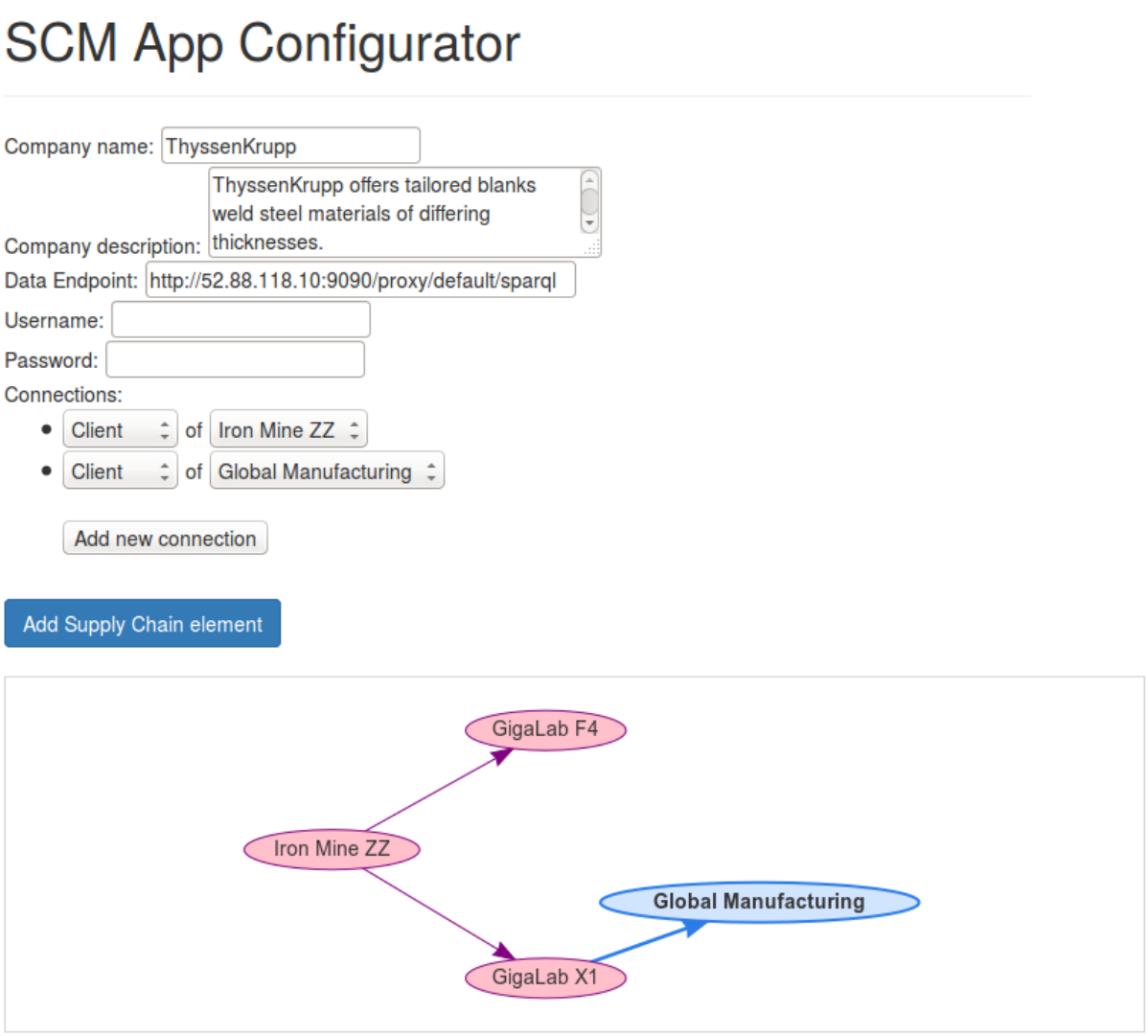


Figure 4.5: Web-based user interface *SCM Intelligence App Configurator*. The upper part allows one to specify the information about a specific organization which is part of the Supply Chain. By clicking on the blue *Add Supply Chain element* button, this organization is added to the lower graph visualization. Thus, step by step, the entire Supply Chain can be modelled using this user interface

Client

The client is realized as a web application for two reasons: First, it prevents possible installation issues or required operating-system dependent preconfigurations. Thus, having a web browser is the only prerequisite for using our system. Second, due to the vast use of open web standards such as RDF or SPARQL, providing a web client is only consequential, considering that, without an active network connection, data exchange is impossible.

The user interface is developed by using the markup language HTML5[11] and the JavaScript libraries `jQuery`[12] and `vis.js`[13]. The entire source code is available at the repository hosting platform

[11] `https://www.w3.org/TR/html5/`
[12] `http://jquery.com/`
[13] `http://visjs.org/`

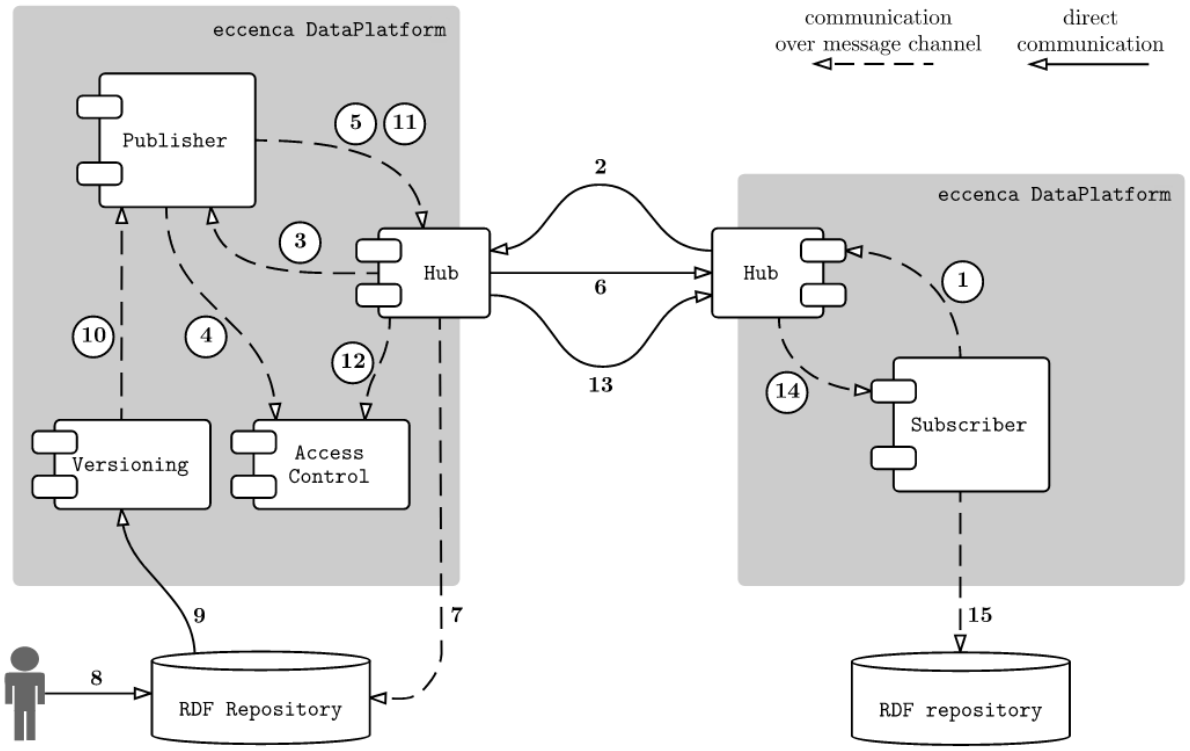


Figure 4.6: eccenca DataPlatform architecture and workflow overview.

GitHub[14].

Figure 4.5 depicts the SCM configuration view for the user. Step by step the user can add each enterprise to the supply chain and thus model the entire view. For each enterprise, one needs to specify the SPARQL endpoint, the access credentials and its position in the supply chain. Once all this information has been defined, the entity is available for future supply chain analyses.

Figure 4.7 depicts the analysis view of the client. By specifying the timespan and analysis method (in the example: "Order Delivered in Full"), one can run the analysis on live or historical data and have strong and weak links highlighted. Thus, one can identify connections that require special attention and should therefore be acted upon or be replaced with alternative suppliers.

Predefined methods are described in [65]. Example dimensions include *Reliability*, *Responsiveness*, *Agility*, *Costs* and *Asset Management*.

Server

As a server, we chose the eccenca DataPlatform[15] because of its access control support and SPARQL endpoint.

The authentication and authorization mechanisms guarantee that data consumers access only the data that the endpoint owner explicitly permits. To ensure consumer authentication, OAuth2.0 [73] is used. Access control rules can be defined on a named graph level.

Once a local graph is modified, the endpoint will notify its subscribers by sending the latest change

[14] https://github.com/EIS-Bonn/SCMApp
[15] https://www.eccenca.com/en/products/linked-data-suite.html

sets for inclusion[16]. An example of this approach is depicted in Figure 4.6. The workflow starts with a new subscription message from the subscriber to its hub (1). The hub then authenticates itself in the name of the subscriber at the advertised hub of the publisher and sends a PubSubHubbub subscription request (2). The hub of the publisher hands over the subscription request for validation to the publisher (3) which validates the authorization of the subscriber with the help of the access control component (4). In case of success, the publisher informs its hub (5) which verifies the subscription (6) before storing it in the local RDF repository (7). If an update is made to the subscribed data (8), the versioning component creates a patch (9) and informs the publisher about the change (10). The publisher sends a change notification to its hub (11) that validates for each affected subscription if it is still authorized to get updates (12). In the positive case, the hub sends a PubSubHubbub content distribution request to the affected subscriber hubs(13), else the subscription is canceled. The subscriber hub now informs its subscriber (14) that checks the integrity of the patch before applying it to its data (15). When applying patches, especially in the presence of blank nodes, it must be ensured, with respect to non-lean graphs and isomorphism, that the correct triples get removed.

Authorization is based on the Authorization vocabulary[17] as described in [75]. By providing context-based access conditions (`eccauth:AccessCondition`), it allows providers (suppliers or clients) to decide which data to share with whom on an RDF named graph level.

However, the server may be exchanged with any other system that provides a secured data access interface. The knowledge base, a triple store, may also be exchanged as long as SPARQL queries can be executed. While in theory access control is not important for the analysis itself, in reality, for enterprises it is. Thus, if publishing supply chain information openly is an option, any open SPARQL endpoint may be sufficient.

4.2.4 Feasibility Assessment

We provide a proof-of-concept evaluation for the feasibility of our approach.

Figure 4.7 describes the following scenario: The raw material producer *Iron Mine ZZ* supplies the enterprises *Daimler*, *GigaLab F4* and *GigaLab X1*. Subsequently, *GigaLabX1* then supplies *Global Manufacturing* and *Press Logistics MML* and so on. Once this network is specified, one can choose the analytic method to measure the quality. In our example, we choose *Order Delivered in Full* (according to SCOR). Listing 4.9 represents the SPARQL query for this metric. In order to calculate this result, it is necessary to first select all deliveries executed and then to divide their count by the count of those that were not fully executed. Whether a delivery is considered full or not full is specified by the properties `hasMetricRL_33` and `hasMetricRL_50`.

In the SCOR standard, this metric is identified by *RL 2.1* and contains the following general definition:

```
1  @prefix  xsd: <http://www.w3.org/2001/XMLSchema#> .
2  @prefix     : <http://purl.org/eis/vocab/scor#> .
3
4  SELECT (xsd:decimal(?full) / (xsd:decimal(?notFull)) * 100) AS ?result
5  WHERE { { SELECT COUNT(?deliveredInFull) AS ?full
6            WHERE { ?deliveredInFull  :hasMetricRL_33  100 .
7                    ?deliveredInFull  :hasMetricRL_50  100 . } }
8          { SELECT (COUNT(?allDeliveries)) AS ?notFull
9            WHERE { ?allDeliveries  a  :Process . } }}
```

Listing 4.9: Formalized SPARQL query of the SCOR KPI *Order Fulfillment Cycle Time*.

[16] The overall process is compatible with the PubSubHubbub working draft v0.4 [74].
[17] `https://vocab.eccenca.com/auth/`

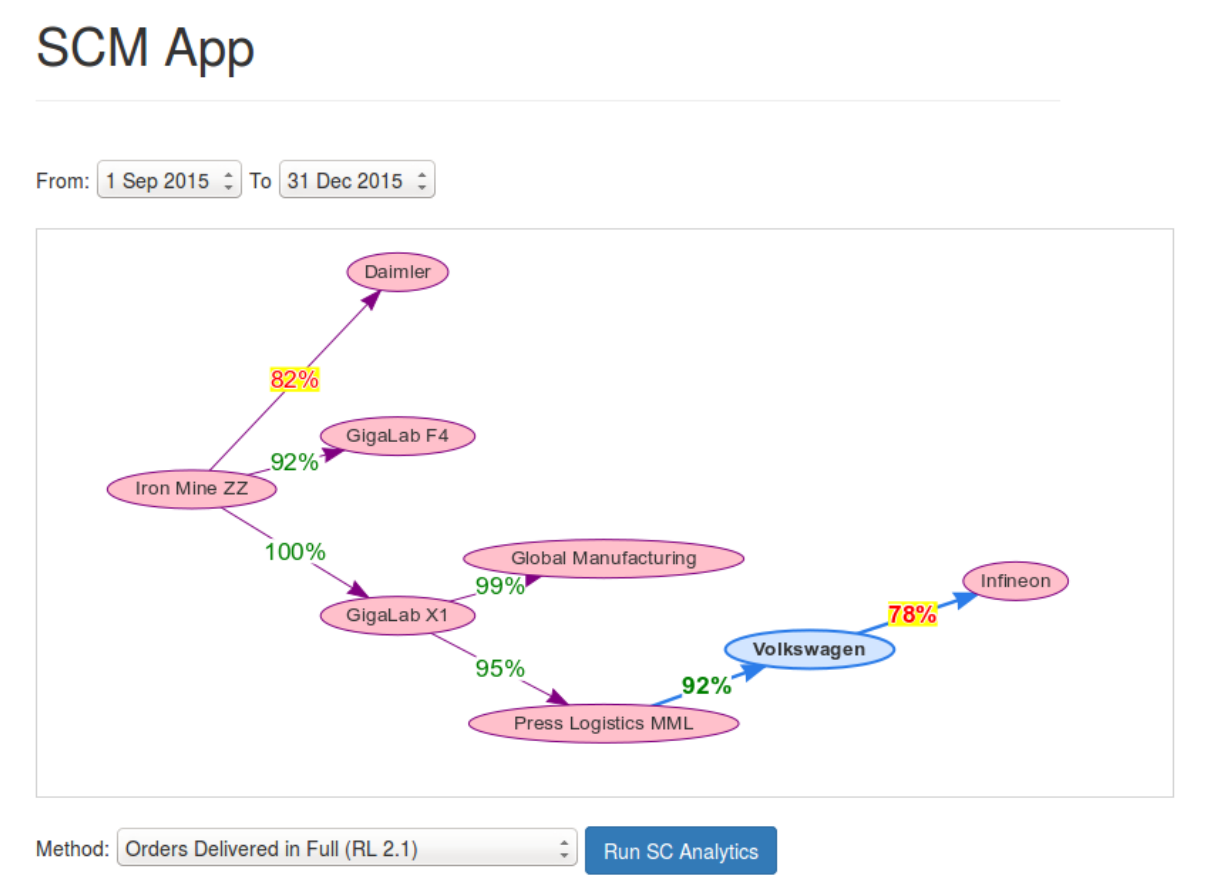


Figure 4.7: Web-based user interface of the *SCM Intelligence App Analytics*. The user needs to specify in the upper part the timespan of the Supply Chain to be analyzed. At the bottom, one SCOR KPI method (*Orders delivered in Full (RL 2.1)*) needs to be selected, which will be executed on the *SCORVoc* data on the server-side

> *Percentage of orders which all of the items are received by customer in the quantities committed. The number of orders that are received by the customer in the quantities committed divided by the total orders* [7]

Thus, the percentage *92%* between *Iron Mine ZZ* and *GigaLab F4* means that *8%* of all orders received by *GigaLab F4* were not received in the quantities originally ordered in the timespan of September 1 2015 up to December 31 2015. For the sake of better identifying problematic links, values below 90% were highlighted with a red font color and a yellow background. We have deployed a demo[18] with multiple server instances including artificial test data.

4.3 Concluding Remarks

The use of data-centric approaches in engineering, manufacturing and production are currently widely discussed topics (cf. Industry 4.0, smart manufacturing or cyber-physical systems initiatives). The complexity in Supply Chain Management in general is thought to be one of the major bottlenecks of the

[18] `https://rawgit.com/EIS-Bonn/SCMApp/master/scm-app.html`

field. A key issue in engineering, manufacturing and production is to be able to efficiently and effectively manage Supply Chains.

The complexity of supply chain management in general is considered to be one of the major bottlenecks of the field. Interoperability, decentralization, real-time capability and service orientation are four of the six design principles of the Industry 4.0 initiative [76].

This chapter introduces the SCORVoc RDFS vocabulary, that formalizes and operationalizes the SCOR standard. SCORVoc is available on GitHub[19] for collaborative further development and at *purl.org*.[20]. We used an innovative methodology to develop the SCORVoc vocabulary, and provided means to automatically compute typical KPIs. Finally, we described comprehensive test scenarios for SCORVoc and implemented a synthetic benchmark. SCORVoc, together with the formalized SPARQL queries, represents a comprehensive approach to facilitate information flows in Supply Chains, and enables the design of SCOR compliant IT applications.

Our prototype tackles these challenges using a decentralized system, which provides direct access to crucial supply chain relevant information. While proprietary software systems are slowly introducing access services such as REST or SOAP, also a semantic description of the data is required to gain a common understanding and thus reduce the semantic barrier. Aligning data structures on industry standards increases interoperability, but in order to meet enterprise-specific demands, any data structure needs to allow customization without breaking the model. From a non-technical perspective, a challenge also lies in establishing trust among supply chain partners. Providing remote access to internal knowledge bases is often hindered by legal concerns and the fear of giving away one's business secrets.

Finally, let us note that we learned from domain experts that SCOR still has a number of limitations. For example: a delivery of 9 out of 10 items is described as a 90% success rate. Although for one company this may be an appropriate measurement, for another company, the missing part may actually stop the entire production part. SCORVoc may help identify such limitations, and accompany the improvement of the SCOR specification.

[19] `https://github.com/vocol/scor`
[20] `http://purl.org/eis/vocab/scor`

CHAPTER 5

Collaborative Industrial Ontology Engineering

This chapter is dedicated to explore how more mature industrial ontologies can be built unisono between domain experts and ontology engineers. Tools are proposed to support collaborative ontology engineering and existing problems are discussed in this chapter. The content of this chapter is based on the publications [69, 77–79][1].

This chapter answers the following research question:

RQ2: How can we support domain experts contributing to the engineering of industrial ontologies?

5.1 Requirements

The development of TurtleEditor was originally driven by the requirements of the *MobiVoc consortium*[2]. The aim of the MobiVoc consortium is to support the mobility of people through the mobility of data by developing a comprehensive ontology for all aspects of mobility ranging from map data to points of interest to gas stations, electric charging points, and traffic management. The consortium's stakeholders include car manufacturers, researchers, IT companies, and public administrations, sending representatives with different backgrounds to the working groups.

While teaching the Turtle RDF syntax [80] to domain experts proved feasible in this project, avoidable faulty commits led us to identify the following requirements for an editing tool, with the aim to improve the quality of the stakeholder contributions:

Syntax Checking: The editor should tolerate syntax errors, but it should detect, highlight and report them in a comprehensible way.

Auto Completion: Terms of widely used ontology languages, such as RDF, RDFS and OWL, should be suggested to the user as auto completions.

[1] [78] is a joint work with Alexandra Similea, a previous MSc student at the University of Bonn. In this paper, my contributions include the overall problem definition, motivation, minor contributions to the development and a general supervision of her research which resulted in this publication and her MSc thesis at the University of Bonn.
[69] is a joint work with Lavdim Halilaj, a research engineer at Bosch. In this paper, I contributed to the problem definition, defining the requirements, reviewing the related work, and to the development of *VoCol*.

[2] `http://www.mobivoc.org`

Syntax Highlighting: Syntactic constructs should be highlighted to increase the readability and improve the comprehensibility of the content.

Loading and Saving: The user should be able to load ontology files from a central repository, and to save and merge any changes on the ontologies back into the repository after editing.

Query Service: A SPARQL query interface should enable the execution of test queries to validate the functionality of the ontology.

User feedback on the first version of TurtleEditor led to additional requirements. The main feature request was to add a visual perspective to the editing. The lack of visual editors for RDF-based ontology development led us to develop our own approach. We defined the following requirements to keep the editing experience as user-friendly as possible:

Visual Editing: The visual interface should provide basic editing operations, such as creating, editing, and removing nodes (subjects or objects) and edges (predicates) in the RDF graph.

Hybrid Editing: The user should be able to choose between working in 1) the visual editor, 2) the text editor, or 3) a combined visual and text editor. The different editing views should be linked, i.e., the focus in the text editor should follow the user selections in the visual editor.

Synchronization: Any changes made in either the text or visual editor should be automatically synchronized with the other editor at runtime. When syntactical errors are introduced in the text view, the visual view should freeze until the errors have been corrected.

Graph Filtering: To unclutter the visualization of large and dense RDF graphs, the filtering of certain types of nodes and edges should be supported.

Graph Clustering: When the visualization of larger RDF graphs becomes unmanageable, nodes should be grouped together in order to retain readability. This implies finding an appropriate clustering metric as well as taking into account topological aspects of the graph.

5.2 Architecture

A key principle of our implementation of TurtleEditor is to reuse existing mature, broadly accepted components. Thanks to more and more repository hosting platforms providing RESTful interfaces, it is feasible to build customized web frontends for their functionality, as explained in this section. Similarly, thanks to the increasing number of available mature JavaScript libraries, our client-side development can focus on the actual Turtle editing, as explained in Section 5.3.

TurtleEditor is designed to be compatible with repository hosting platforms such as GitHub. Alternative platforms (e.g., BitBucket or GitLab) also provide REST APIs and a similar set of features identified as useful for ontology development. While we aim at supporting various platforms, we focused on GitHub in a first step because of its popularity and since many ontology projects are already hosted on GitHub.

TurtleEditor aims at providing an ontology-aware frontend on top of repository hosting platforms, without forcing people to use it. Users can always clone (i.e., download) the repository and work locally with their preferred ontology editor, or even resort to basic text editors provided by the repository hosting platform itself.

While the goal is to improve the quality of contributions, we nevertheless still allow commits with errors in them. Following the *wiki* principle [81], we prefer faulty contributions being added and fixed by

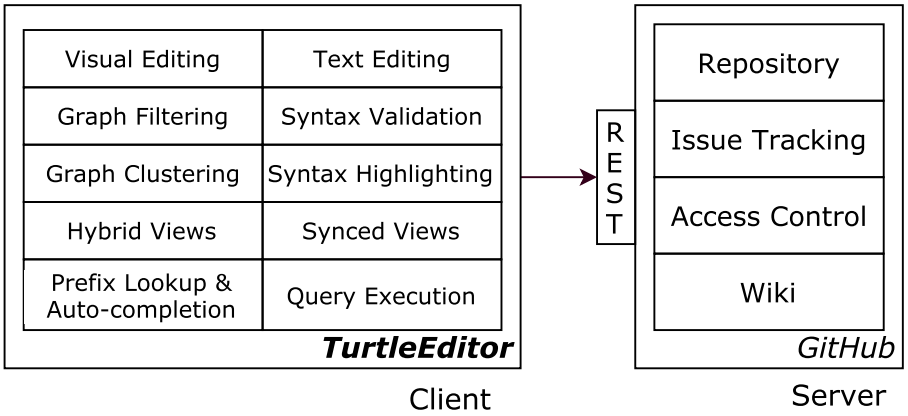


Figure 5.1: Proposed client-server architecture for the *TurtleEditor*. While the server offers certain project-management-specific and security requirements, the majority of identified requirements for the system are to be implemented on client-side for direct user-feedback, without the need of communicating with the server.

others over useful contributions not being made at all. In what follows, we first present the high-level architecture of TurtleEditor and then describe the implemented features step by step.

Figure 6.3 shows the architecture design of TurtleEditor, which is a client-side application. The developed RDF-based ontology is saved to a user-defined Git repository.

The textual modules include the *text editor*, *syntax validation* for live feedback on edits being made, and the *syntax highlighting* to increase readability. *Auto-completion* proposes class and property names for well-known ontologies and a *query execution* view to test the ontology using SPARQL queries.

The graphical modules include the *visual editor*, the two-way *code synchronization* between the different editing views, a graph *visualization*, which includes various functionalities for displaying RDF graphs in a meaningful way, a *prefix lookup* service, to search for existing ontologies with their classes and properties, and a *clustering* module, which reduces the size of the graph visualization.

5.3 Implementation

In the following, we describe how each requirement was implemented step by step. The entire implementation is based on multiple mature open-source JavaScript libraries and our own developments. The decision for a web application is motivated by the goal to enable domain experts to participate immediately without the need for any software installation on their computers.[3]

Text Editor For the browser-based text editor, we used the *CodeMirror*[4] JavaScript library as a basis. Started in 2007, CodeMirror focuses on extensibility and offers a rich programming API. Besides being able to handle large texts, it is able to fulfill two of our requirements: syntax highlighting and auto completion. Syntax highlighting is supported natively for more than 100 programming languages, Turtle being one of them.

Auto Completion CodeMirror supports auto completion by internally defining namespaces as *hints*. Thus, we defined the `rdfs` terms, which are checked once the auto completion is triggered. As an example, we defined the keyboard shortcut `Ctrl+Space` as the trigger event. Once the prefix `rdfs:` has been typed and this event is fired, TurtleEditor internally checks for the `rdfs` namespace to suggest

[3] A demo, a video, and the source code of TurtleEditor are available at `http://editor.visualdataweb.org`.
[4] `https://codemirror.net`

SPARQL editor:

```
PREFIX mv: <http://purl.org/net/mobivoc/>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
PREFIX schema: <http://schema.org>

SELECT ?fuelname ?price ?stationName
WHERE
{
  ?station a mv:FillingStation .
  ?station rdfs:label ?stationName .
  ?station mv:hasOffer ?fuel .
  ?fuel rdfs:label ?fuelname .
  ?fuel mv:hasPrice ?price .
 }

```

Run Query

fuelname	price	stationName
Aral Super 95	1,57	Aral Tankstelle
Aral Super 95 E10	1,59	Aral Tankstelle
Aral Super Plus 98	1,63	Aral Tankstelle
Aral Super Plus 92	1,60	Aral Tankstelle
Shell Fuel Save 95	1,55	Shell Tankstelle
Shell Fuel Save E10	1,58	Shell Tankstelle
Shell Diesel Fuel Save	1,32	Shell Tankstelle

Figure 5.2: Web-based user interface for testing the execution of SPARQL queries. The upper part allows the user to specify the SPARQL query with direct feedback in case of syntactical errors. The query can be tested by pressing the blue *Run Query* button with the results display in the table at the bottom.

matching terms.

Prefix Lookup To help users reuse already existing class and property definitions, we added a look up service to Linked Open Vocabularies (LOV)[5]. LOV is a repository for Semantic Web ontologies and vocabularies. Besides its own user interface to look terms up, it also provides multiple APIs to build tools on top of LOV[6]. We use these APIs to support prefix completion and term completion.

Syntax Validation Another major requirement is the validation of the ontology. The *N3.js*[7] JavaScript library supports client-side RDF manipulation. We implemented the parsing feature like in common IDEs: Each time CodeMirror detects a change by the author, the entire text source is parsed automatically by N3.js for syntax errors. If an error is found, the faulty code line is highlighted with a red background, and a red dot is added next to the line number. A tool tip providing error descriptions is being presented whenever the user hovers the mouse over the dot. This gives the user the chance to fix faulty lines and also to gain a better understanding of the Turtle language. Nevertheless, as mentioned in our key principles, syntax checking is only a soft measure and does not prevent users from committing these changes to the repository.

Query Service To support querying the edited ontology, we used the `rdfstore-js` library[8]. rdfstore-js currently supports SPARQL 1.0 and most of SPARQL 1.1 Update. SPARQL 1.1 Query support is currently being added to the library. To clearly separate the UIs for editing and querying the ontology, we added a separate window, which displays the results accordingly. For this, we reused the

[5] `http://lov.okfn.org/dataset/lov/`
[6] `http://lov.okfn.org/dataset/lov/api`
[7] `https://github.com/RubenVerborgh/N3.js`
[8] `https://github.com/antoniogarrote/rdfstore-js`

CodeMirror code editor to allow for editing and executing SPARQL queries. The results of the query are presented using an HTML table (see Figure 5.2 for an example).

Repository Communication The communication to the repository is realized by using the REST interface provided by the GitHub repository. This way, the user can specify any given GitHub repository project for TurtleEditor to access from. All files of the repository are filtered depending on the file format ('.ttl') to ignore non-ontology sources such as *readme* files. Thus, projects in which the ontology consists of multiple modules can be easily edited by switching between files (using a drop-down list).

Access Control Since most projects hosted on GitHub are open source, it is not always necessary to log in before checking out the ontology files. However, committing changes to the repository, as well as reading files from a private project (non-accessible to the public), requires a GitHub account. Instead of logging in with credentials, we also provide the option to use a generated personal access token to authenticate with the GitHub REST API and thus avoid sharing one's actual credentials with the third party that hosts TurtleEditor. Furthermore, one can select a specific branch to support non-linear ontology development projects. By default, the 'master' branch is selected.

Visual Editor As the elements defined in RDF are basically interrelated entities, the apparent structure that can be employed for their visual representation is a graph. A well-documented and mature JavaScript library that supports manipulating such a structure is *vis.js*[9]. The initial drawing of the graph expects an object containing the arrays of nodes and edges, and optionally, a set of configurations for the available modules. The *nodes* and *edges* modules offer settings like: id, shape, color, label, title, etc. One example where we made use of these options is visually separating URI nodes from string literals, by using different colors and shapes. Editing in the visual view is realized by a set of buttons (e.g., "Add Node", "Edit Node", see Figure 5.3). Edges between two nodes can be added in a drag-and-drop manner by the user.

While the *vis.js* physics engine has proved to be useful when it comes to the intuitive interaction with the graph, it requires a lot of browser resources and leads to performance issues. This goes as far as creating a lag in the client and freezing the entire browser for some seconds. To address this problem, we integrated the possibility of deactivating the *physics* module by offering a "freeze" option. This also provides users with the option to manually rearrange the graph layout and drag the nodes at their positions of choice.

Synchronization This module is based on the idea of having an underlying model that connects the two views. The model consists of an array of triples, where a triple is a JavaScript object with three fields: subject, predicate, and object. The value of each field is a string representing the URI of the corresponding entity.

We detect textual changes using the *CodeMirror* library. Change detection is handled by triggering an event every time the editor content is modified. The propagation of these changes towards the underlying model relies on the parsing functionality offered by the *N3.js* library. With each change event, the Turtle code is parsed, and, if no syntax errors are detected, the valid triples are returned by the library to be processed. This array of triples is then compared with the previous model version. When differences are detected, the underlying model and the visual view both are updated. The change detection in the visual view is handled by the *vis.js* library, which triggers an event with each modification made to the structure of the visualized graph.

To process changes in the visual view accordingly, the array of triples need to be translated into Turtle code. For this, we again employ the functionalities of the *N3.js* library, which features a *Writer* object that serializes one or more triples (given as an array) into an RDF file, where the default format is Turtle.

Hybrid View The textual and visual views are realized as separate tabs. In addition, we added the

[9] `http://visjs.org`

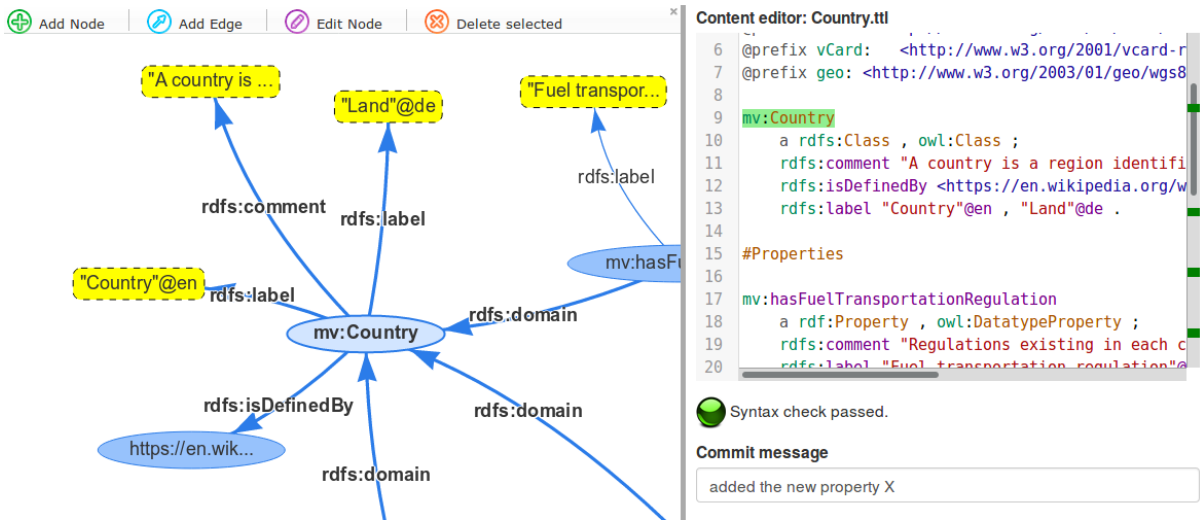


Figure 5.3: Web-based user interface for editing and visualizing ontologies. Changes in either the visual view (left) or the text-based view (right) result in an update on the other side. Changes in the ontology can be pushed to the server by specifying a commit message at the bottom right.

split (or hybrid) view option to allow users to supervise changes in both views in parallel. In the split view, another functionality is available: *term highlighting* – when the user clicks on a node in the visual graph, its corresponding representation in the text view is marked with a light-green background (see Figure 5.3). Moreover, the highlights are also available on the scroll bar, so that the user can easily navigate between multiple occurrences of a term found in the Turtle code. This feature is implemented using the *CodeMirror* library with its *matchesonscrollbar.js*[10] add-on.

Graph filtering Another feature of the visualization module is hiding basic graph structures to avoid unnecessary *clutter*. Figure 5.3 is showing literals as rounded yellow rectangles, classes or links as blue ellipses, and properties as blue arrows. *rdf:Property*, *owl:Class*, *rdfs:Literal* triple definitions are thus filtered since their meaning is already conveyed through their visual representation. The user can activate or deactivate this feature by flagging the "Hide defaults" checkbox.

Clustering *vis.js* provides support for clustering by offering several methods for grouping nodes together based on selected properties. Out of these, our clustering strategy is to identify *outliers*, where an outlier represents a node with exactly one edge, i.e., one neighboring node. This strategy can be employed iteratively, as long as new outliers are found after each clustering step.

Furthermore, the user can revert the clustering on demand by clicking on clustered elements to set hidden graph elements free. To cluster and uncluster the entire RDF graph step by step, corresponding buttons are provided in the control bar of TurtleEditor.

5.4 Evaluation

We evaluated the TurtleEditor in a qualitative user study and with browser performance tests.

[10] `https://codemirror.net/addon/search/matchesonscrollbar.js`

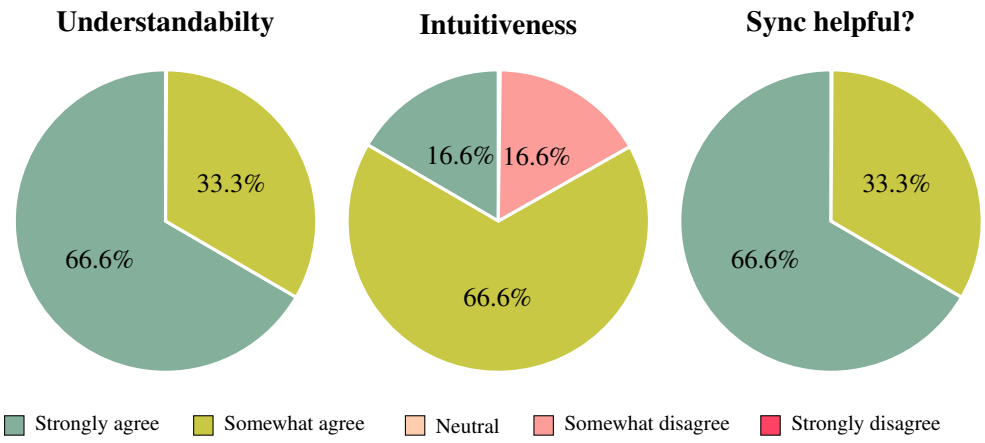


Figure 5.4: Feedback on our user evaluation of the *TurtleEditor*

5.4.1 Qualitative Evaluation

We assembled six experienced developers and asked them to edit an ontology modeled in RDFS[11] plus simple OWL constructs such as `owl:Class` and `owl:DatatypeProperty`. Tasks assigned to the test users included adding and removing classes and properties to evaluate the usability of the different modules and features. All participants successfully accomplished the tasks in a time proportional to their experience level, 15 minutes on average. Afterwards, they gave feedback in a questionnaire covering the following statements, as summarized in Figure 5.4.

Understandability *The graph visualization represents the information in an understandable way.*

Intuitiveness *The graphical editor is intuitive to use.*

Sync helpful? *The synchronization helps with understanding the code representation in Turtle.*

The general feedback was positive, only one participant was unsatisfied with the editor's intuitiveness. The main reason for this was the method employed for drawing the edges. In fact, clicking and holding the left mouse button to draw a new edge from one node to another appeared as an unexpected method to five of the participants. A possible explanation could be the influence of previous experience with other graphical tools that used a different mechanism for realizing this functionality, which points out the need for implementing a standardized method. This and other difficulties encountered while testing the editor led to the following suggestions for improvement for the visual editor:

1. more intuitive edge drawing, such as consecutively clicking the nodes that need to be linked together;
2. possibility to specify the type of a newly added node through a drop-down menu with available types;
3. search functionality within the graph;
4. auto completion when adding new labels;

[11] `https://github.com/vocol/mobivoc/blob/master/Country.ttl`

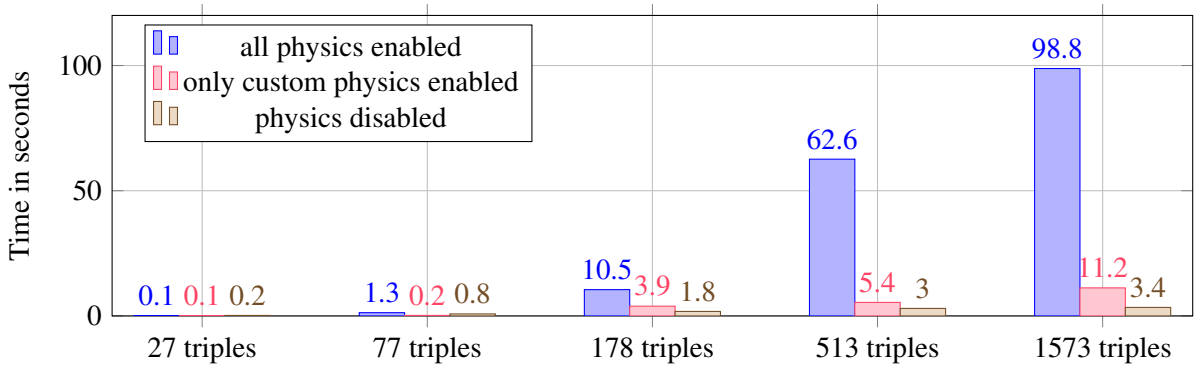


Figure 5.5: Load time performance of the *TurtleEditor* in a web browser

5. filtering by different kinds of nodes (e.g. hiding all literals);

6. direct editing of node and edge labels by mouse interaction instead of using buttons (that is, when an element is clicked, its label becomes editable);

7. copy-paste functionality for graph elements.

5.4.2 Quantitative Evaluation

Figure 5.5 depicts the times required for loading five different RDF-based ontologies with the Google Chrome web browser (version 54). The experiment was executed on a laptop with an i5 CPU with two physical cores (2.66 GHz each), 4 GB main memory and a 64-bit OS. The tested RDF graphs ranged from 27 triples, to 77 triples, 178 triples, 513 triples, and 1573 triples. While performing the time evaluation, we noticed that much computational power is consumed by the graph physics, which uses a iterative method to calculate the forces exerted between nodes. As a result, we loaded the RDF graphs considering the following scenarios: 1) all physics enabled (blue bar), 2) only custom physics enabled (red bar), and 3) physics completely disabled, such that no forces are calculated among the nodes (green bar). The custom physics brings certain modifications to the way the edges are drawn, concerning smoothing, robustness, and length.

The third scenario yields the best results regarding time performance, but the resulting graph is almost unreadable as many nodes are overlapping due to lack of repulsion forces. Therefore, we decided to have the physics implicitly enabled and provide the possibility of disabling them after the graph is loaded.

As observed during development, the graph load performance is not optimal and requires more work. As a result of this experiment, graphs are not loaded by default in the visual view, only in case the user actively opens it.

5.5 Integration into the Authoring Platform VoCol

TurtleEditor is integrated into the collaborative ontology authoring platform named *VoCol*. VoCol supports various front-end views to better coordinate ontology engineering projects. The following features are included:

- Syntax validation
- Human-friendly version (HTML)

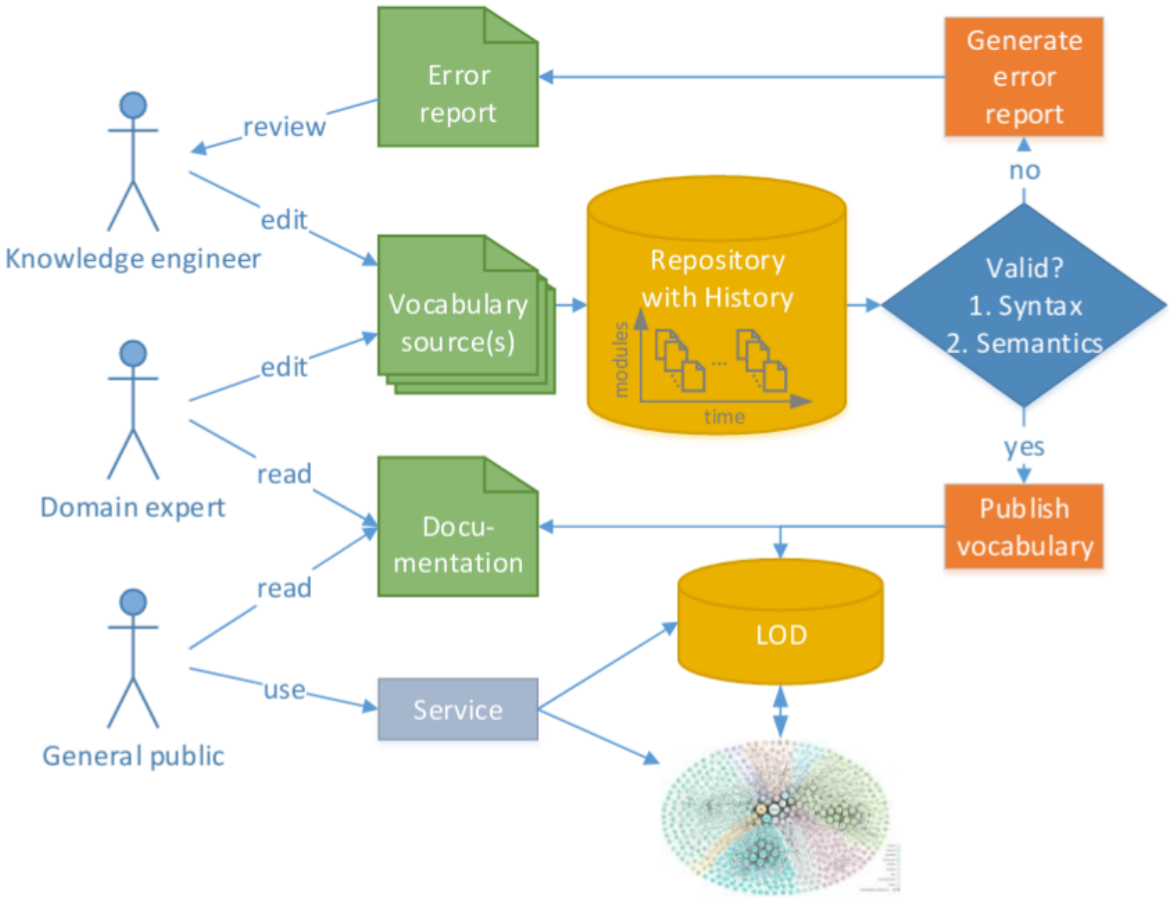


Figure 5.6: Architecture of the collaborative ontology authoring environment *VoCol*

- Machine-interpretable version (RDF)
- Graphical visualization
- Evolution tracking
- Querying service

Figure 5.6 depicts the architecture of VoCol and presents the workflow. The ontology is hosted in a Git repository, from there changes are pulled and committed. Syntax validation is offered based on single ontology source files on the front-end via the TurtleEditor, and for all ontology files at back-end for provide a full syntax validation report. For the entire ontology, a human-friendly version in form of a searchable web page is generated and the environment provides a machine-interpretable version in RDF. A graph visualization is also offered by integrating WebVOWL[12] and a SPARQL endpoint including a US is provided to support direct evaluation queries on the ontology.

Figure 5.7 depicts a screenshot of the TurtleEditor integrated into VoCol. The view presents the only *lightweight* option for non-technical contributions to participate in the development of the ontology. The alternative would be to contribute with changes directly via the version control system Git.

[12] `http://www.visualdataweb.de/webvowl/`

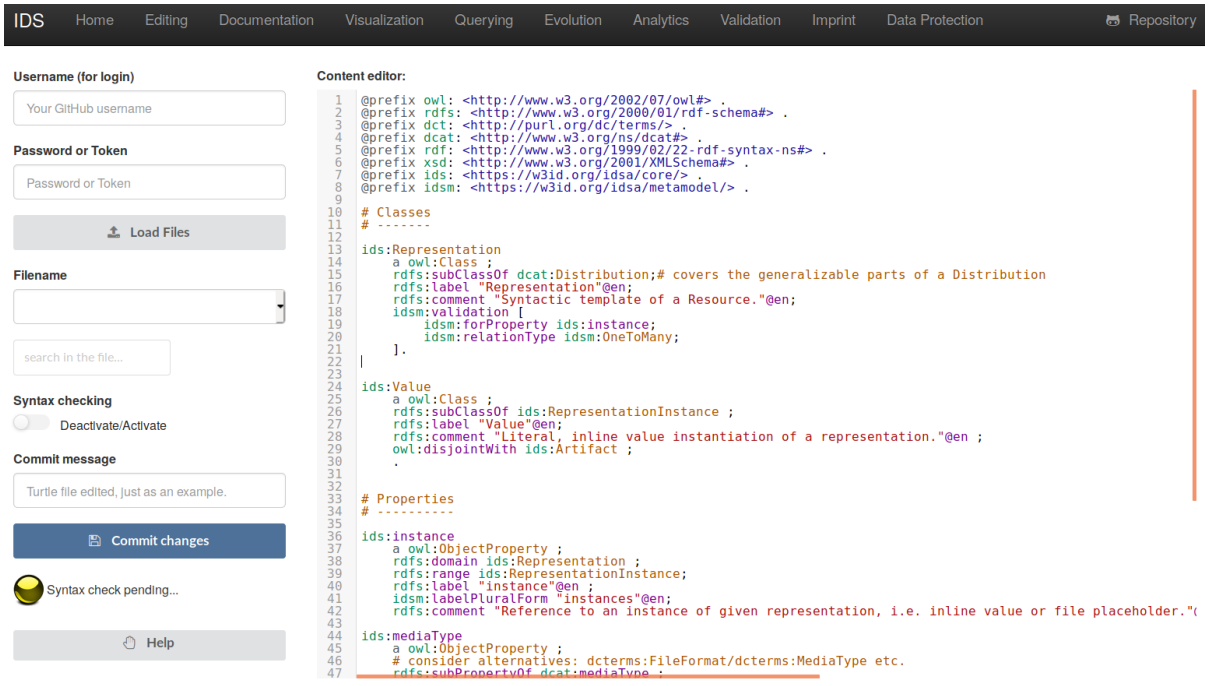


Figure 5.7: Web front-end of TurtleEditor integrated into the *VoCol* ontology authoring environment.

5.6 Concluding Remarks

In this chapter, we presented TurtleEditor, a web-based RDF editor for distributed ontology development. While the implementation of the text editor was rather straight-forward, adding a visual editor raised a couple of issues which needed to be addressed. We discussed these issues and explained how we solved them. The entire code base is open-source[13] and free to be used and extended by any interested party.

Future work includes improving the discovered performance issues and usability of certain components of the visual editor. The latest version of TurtleEditor will soon be integrated into the larger VoCol [69] environment, which further facilitates collaborative ontology development by additional functionality. We also plan to add a default tutorial for domain experts to learn step by step ontology engineering in the browser. Furthermore, the syntax validation in the text editor could be optimized to validate only new changes instead of the entire source code.

By having synchronized textual and visual editors integrated in one tool, we aim at lowering the barrier for domain experts with limited knowledge of RDF to participate in ontology development. Synchronized editors can also provide a new perspective to ontology engineers or to people who are learning RDF.

[13] `https://github.com/EIS-Bonn/TurtleEditor`

CHAPTER 6

Industrial Case Studies

This chapter is dedicated in exploring industrial use cases where semantics were applied. The pressing question is to evaluate where semantic technologies can make a difference compared to existing technologies in place. The content of this chapter is based on the publications [68, 82–84] and new research results as an active scientist in the International Data Spaces (IDS)[1] research project.

This chapter answers the following research question:

RQ3: How can we leverage semantics in industrial use cases?

6.1 An RDF-based Information Model for a Manufacturing Company

This section is dedicated to explore how the recently proposed Ontology Based Data Access (ODBA) data integration method may be used to integrate relational data into an industrial information model, or, into a Knowledge Graph for an enterprise to exploit. The content of this chapter is based on the publication [82].

6.1.1 Motivating Scenario

The information modeling project involved employees from different departments and hierarchical levels of the manufacturing company, external consultants and a third party IT provider. The company itself realized that their IT infrastructure has reached a level of complexity making difficult to manage and effectively use their existing systems and data. While adding new sensors to production lines is straightforward, using the sensor data effectively to improve the production process and decision-making can be cumbersome. The need to share production data with clients led them to evaluate the fitness of semantic technologies. For example, the production of bearing tools is fairly quality-driven depending on the customer specifications. Sharing the production details in a more processable format (compared to non-machine-comprehensible formats) aroused interest. Further goals were to gain a *bigger picture of the company's assets* (physical and non-physical) and to capture as much expert knowledge as possible.

[1] https://www.internationaldataspaces.org/

Use Cases

The concrete use cases are based upon a machine newly introduced into the production lines of the company, a so-called *machine tool*. This is a machine that requires the mounting of tools to assemble specific metal or rigid products. Compared to older generations, the new machine features more than 100 embedded sensors that monitor the production.

Tool Management Possible tools to be mounted into the machine are cutters, drillers or polishers. A tool usually consists of multiple parts. The number of parts depends on the manufacturer of the tool, which is not necessarily the same as the manufacturer of the machine. Mounting tools into a machine is a time-consuming task for the machine operator. Uncertain variables of the tools, such as location, availability and utilization rate, play a major role in the efficiency of a work shift and of a machine in particular. The production of certain goods may wear a tool out quickly, thus decreasing its overall lifetime and forcing the machine operator to stop the machine and replace it with a new tool. Reducing the idle time for remounting the machine by clearly describing its configuration, location and weariness was therefore one concrete goal to be addressed by the information model.

Energy Consumption Producing goods with the machine tool is an energy-intensive process. Before we started the information modeling project, only the energy costs per factory were known. Sensors were added to track the energy consumption per machine and processed work order. For the cost calculation, data from the added sensors and the work orders, which resides in different data sources, needs to be linked and jointly queried. Therefore, integrating this data to be able to retrieve the information at run-time was another concrete goal addressed by the information modeling project.

Data Sources

Three types of data were of particular interest in the project: i) Sensor Data (SD), ii) the Bill of Materials (BOM), and iii) data from the Manufacturing Execution System (MES). The SD comprises sensor measurements of the machine tool. These measurements record parameters needed for the continuous monitoring of the machine, such as energy, power, temperature, force, vibration, etc. The MES contains information about work orders, shifts, material numbers, etc. The machine produces assets based on the work order details, which provide the necessary information for the production of a given asset. The BOM contains information about the general structure of the company, such as work centers, work units, associated production processes, as well as information related to the work orders and the materials needed for a specific production.

6.1.2 Realizing the RDF-based Information Model

The information model aims at a holistic description of the company, its assets and information sources. The core of the model is based on a factory ontology we developed in a previous project [68], which describes real world objects from the factory domain, including factories, employees, machines, their locations and relations to each other, etc. In addition, the information model comprises the mappings between ontologies that represent the data sources (i.e., SD, BOM, MES) and their corresponding schemes.

Development Methodology

Our development methodology was based on the approach proposed by Uschold et al. [70]. We first defined the purpose and scope of the information model; then, we captured the domain knowledge, conceptualized and formalized the ontologies and aligned them with existing ontologies, vocabularies

and standards. Finally, we created the mappings between the data sources and ontological entities. In line with best practices, we followed an iterative and incremental development process, i.e., with an increased understanding of the domain, the information model was continuously improved.

All artifacts were hosted and maintained by VoCol [69], a collaborative vocabulary development environment which we adapted for the purpose of this project. VoCol supports the requirements of the stakeholders: i) version-control of the ontology; ii) online and offline editing; and iii) support for different ontology editors (by generating a unique serialization before changes are merged to avoid false-positive conflicts [85]). In addition, it offers different web-based views on the ontology, including a human-readable documentation, a visualization and charts generated from queries applied to the ontology and instance data. These views are designated to ease the collaboration of domain experts in the development process, i.e., enabling them to participate without having to set up and maintain a proper infrastructure themselves.

Purpose and Scope

The information model comprises i) a formal description of the physical assets of the company, ii) mappings to database schemas of existing production systems, and iii) a formalization of domain-related knowledge of experienced employees about certain tasks and processes within the company.

The heart of the information model represents the aforementioned *machine tool*, including its sensor data, usage processes and human interaction. Therefore, the majority of concepts are defined by their relation to this machine.

The scope is set by the motivating examples *energy consumption* and *tool management* introduced in Section 6.1.1. Nevertheless, the management considered it also *nice to have* to gain a clearer picture of all assets of the company. For example: What local knowledge exists in the factory? What kind of data exists for which machine? Where is that data? Who has access to it? Discussions on fully automated order-driven production sites are ongoing. The management hopes for this to be supported by the information model, and we aim to provide the basis for that goal.

Capturing Domain Knowledge

We captured the domain knowledge in different ways:

1. The company provided descriptive material of the domain, including maps of factories, descriptions of machines and work orders, information about processes, sensor data and tool knowledge. The types of input material ranged from formatted and unformatted text documents to spreadsheets and SQL dumps.

2. An on-site demonstration of the machine within the factory was given during the project kick-off, including a discussion of further contextual information missing in the material. In subsequent meetings, open questions were clarified and concrete use cases for the information model were discussed.

3. We reviewed relevant existing ontologies and industry standards, intending to build on available domain conceptualizations and formalizations.

4. We created customized document templates to enable easy participation of domain experts by collecting input on the ontology classes and properties in a structured way. We collected names and descriptions of all properties having a given class as their domain in one table with one row per property; additional details about the domains and ranges of properties were collected in a separate table. These documents were handed over to the domain experts to be reviewed and completed.

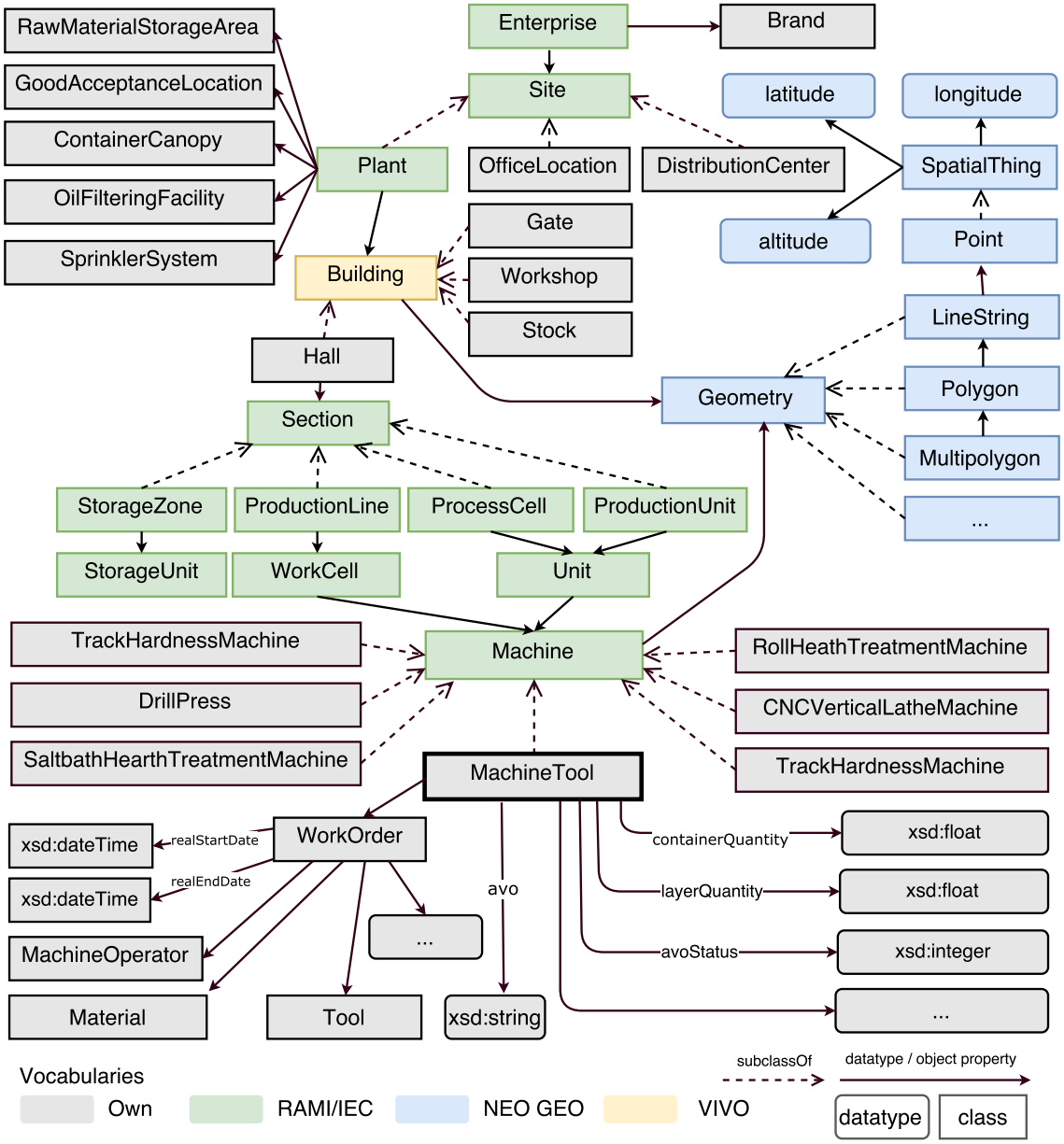


Figure 6.1: Core concepts of the developed ontology to represent the organization of the manufacturing company.

5. We trained IT-affine employees of the company on modeling ontologies using editors such as Protégé, TopBraid Composer and the Turtle editor integrated into the VoCol environment.[2]

Conceptualizing and Formalizing

Figure 6.1 shows the core concepts of the developed information model. The colors group the different subdomains and reused vocabularies. Since `Machine`(s) are the main assets of the manufacturing company, they have been used as a starting point for creating the ontology. Each machine contains a geo-location (property with range `Geometry`) and is part of a certain `Section`, which

[2] `http://protege.stanford.edu/`, `http://www.topquadrant.com/tools/modeling-topbraid-composer-standard-edition/`, `https://github.com/vocol/vocol`

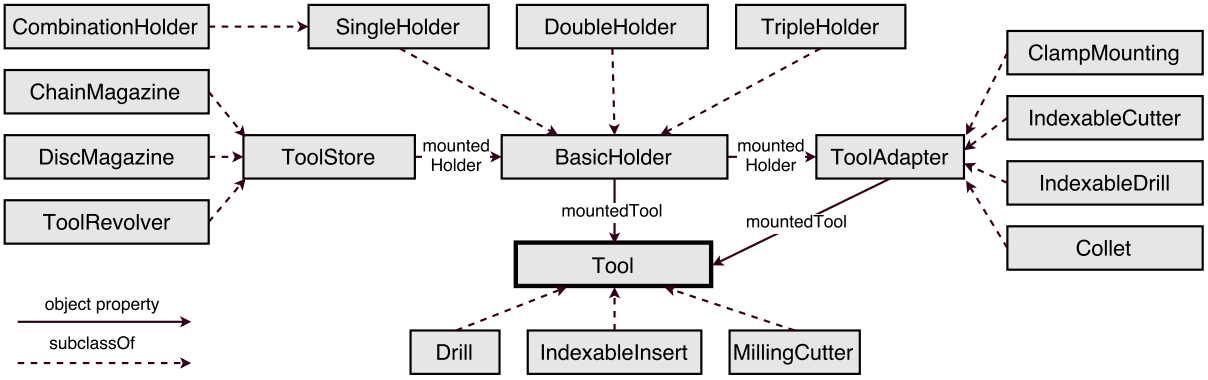


Figure 6.2: Ontological zoom into the *Tool* concepts.

is in a certain `Hall`. Each `Hall` can contain multiple sections and also has a geo-location (inherited from `Building`). `Plant`, `OfficeLocation` and `DistributionCenter` are different types of `Site`(s), each serving a specific purpose. The `MachineTool` comprises domain-related properties to describe its `AVO` (operation status). Next, it is connected with `WorkOrder`(s) to be processed. Each `WorkOrder` defines the required `Material` and `Tool`(s), as well as which machine should be used by which operator to execute a particular task.

Figure 6.2 provides a more detailed view on the `Tool`-related concepts. Machines have different interfaces, called `ToolStore`(s). Tool stores can be equipped with different `BasicHolder`(s).There exist three kinds of basic holders: `SingleHolder`(s), `DoubleHolder`(s) and `TripleHolder`(s). The name indicates the number of tools a holder can be assembled with. A `CombinationHolder` is a special kind of `SingleHolder` that can only be combined with a specific tool. The majority of holders are to be combined with a `ToolAdapter`, on which the actual `Tool` is mounted. Certain tools can be mounted directly onto the holder. Tools are the parts that wear out over time and need to be replaced. As with machines, their geo-location is defined, such that their position can be shown on a map. The tool ontology part reflects the configuration options for tools of multiple tool manufacturers.

In total, the developed ontologies comprise of 148 classes, 4662 instances, 89 object and 207 datatype properties. We focused on the description of the core concepts here that are needed to understand this work; a description of all ontology concepts would be out of the scope of this chapter.

Aligning with Existing Ontologies & Standards

The developed information model consists of concepts from existing vocabularies and industrial standards that we formalized during the project. In particular, concepts from the VIVO (`vivo:Building`), NeoGeo (`ngeo:Geometry`), FOAF (`foaf:Person`) and Semantic Sensor Network `ssn:Sensor`) vocabularies and ontologies are reused.[3]

Furthermore, we aligned the ontologies of the information model to RDF vocabularies of the industry standards RAMI [86, 87] and IEC 62264 [88] that we developed.[4] RAMI is a reference model aiming at structuring the interrelations between IT, manufacturing and product life cycles, whereas IEC 62264 defines hierarchical levels within a company. RAMI includes the IEC concepts and adds the “connected world” as an additional level on top, which aligns well with the basic idea and motivation of this work.

[3] `http://vivoweb.org/`, `http://geovocab.org/doc/neogeo.html`, `http://xmlns.com/foaf/spec/`, `https://www.w3.org/TR/vocab-ssn/`

[4] `https://w3id.org/i40/rami/`, `https://w3id.org/i40/iec/62264`

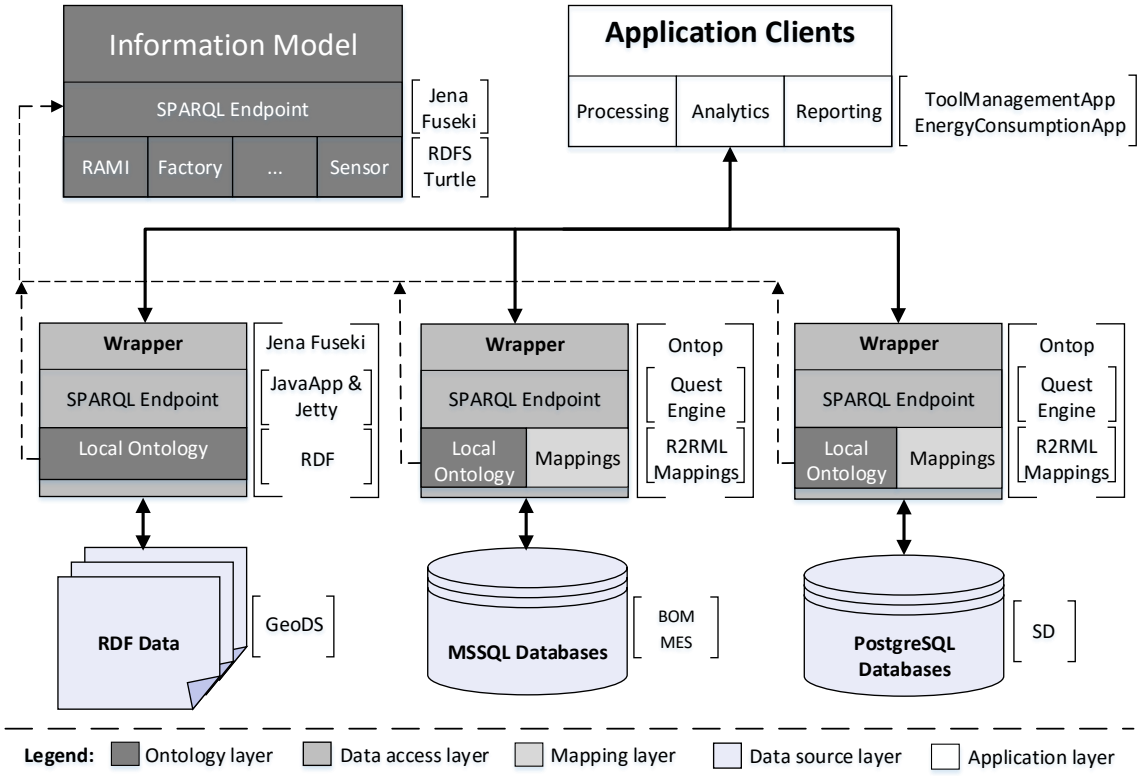


Figure 6.3: The proposed architecture: Web-based user interface clients access the information model to run their respective analytics. The information model comprises of the formalized ontology, RDF data compliant to the ontology as well as the description of relational data in different database management systems. Access to the different knowledge bases (Triplestore and RDBMS) are given via respective wrappers.

6.1.3 Architecture and Implementation

With the objective to provide a uniform interface for accessing heterogeneous distributed data sources, we designed and implemented the architecture illustrated in Figure 6.3. It is extensible and able to accommodate additional components for accessing other types of data sources as well as supporting federated query engines. The architecture distinguishes the following four main layers, some of which are orthogonally located across different components:

The **ontology layer** consists of several ontologies that have been created to conceptualize a unified view of the data. Wache et al. distinguish three main approaches of using ontologies to explicitly describe data sources [89]: i) *Global Ontology Approach*—all data sources are described in an integrated, global ontology; ii) *Multiple Ontology Approach*—separate local ontologies represent the respective data sources, and mappings between them are established; the iii) *Hybrid Ontology Approach*—a combination of the two previous approaches with the aim to overcome the drawbacks of maintaining a global shared ontology and mappings between local ontologies.

We followed the third approach, which enables new data sources to be added easily, avoiding the need for modifying the mappings or the shared ontology. Accordingly, our ontologies are organized in two groups: i) a shared ontology to represent the highest level of abstraction of concepts and mappings with external ontologies; and ii) local ontologies representing the schemas of the respective data sources. This

makes our architecture quite flexible with respect to the addition of diverse types of data sources [90].

The **data access layer** consists of various wrappers acting as bridges between client applications and heterogeneous data sources. It receives user requests in the form of SPARQL queries, which are translated into the query languages of the respective data sources, and returns the results after query execution. Accessing relational databases is realized using the *Ontology-Based Data Access* (OBDA) paradigm, where ontologies are used as a conceptualization of the unified view of the data, and mappings to connect the defined ontology with the data sources [91]. In particular, the Ontop [92] framework is used to access the data sources, i.e., the BOM, MES, and SD data, which exposes the relational databases as virtual RDF graphs, thus eliminating the requirement to materialize data into RDF triples. Additionally, Jena Fuseki is used as in-memory triple store for loading GeoDS, since the information about the geo-locations of the machines are less than 20,000 RDF triples.

The **mapping layer** deals with the mappings between the data stored in the data sources and the local ontologies. For the definition of the mappings, we used *R2RML*[5], the W3C standard RDB-to-RDF mapping language. As a result, it is possible to view and access the existing relational databases in the RDF data model.

The **data source layer** comprises the external data sources, i.e. databases and RDF datasets as described in Section 6.1.1. Due to the high dynamicity and the great amount of incoming data, the data sources are replicated and synchronized periodically. As a result, any performance and safety degradation of the production systems is avoided. Additional types of data sources can be easily integrated in the overall architecture by defining local ontologies, mappings with the global ontology and data sources as well as choosing an appropriate wrapper.

The **application layer** contains client applications that benefit from the unified access interface to the heterogeneous data sources. These applications can be machine agents or human interface applications able to query, explore and produce human-friendly presentations.

6.1.4 Application to the Use Cases

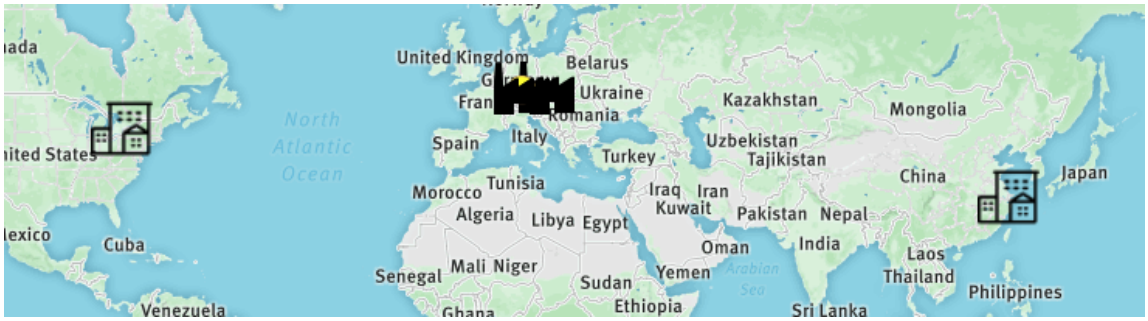
We applied the developed information model to the use cases introduced in Section 6.1.1 to demonstrate the possibilities resulting from semantically integrated data access.

Tool Management

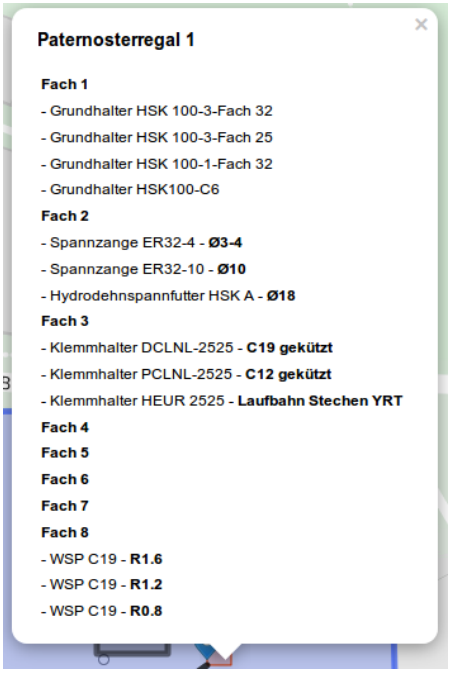
Figure 6.4 displays different views on the assets of the company. On a world map (see Figure 6.4(a)), the sites of the company are highlighted based on their geo-location given in the information model. By zooming in, the different locations can be investigated w.r.t. their functionality, address, on-site buildings up to the level of machines, etc. By clicking on the objects on the map, static and live production data is displayed. As an example, Figure 6.4(b) shows all tools stored in a certain paternoster system, grouped by drawer. Figure 6.4(c) provides an example of a machine with its properties: production name, current status, self-visualization, mounted basic holder, tool with its diameter, etc. Further, it contains links to existing external analytical web pages. A “Determine Tool Availability” function is offered for locating the tools to be assembled in the closest paternoster storage system based on the location of the machines.

Each time a certain view is opened, a SPARQL query is executed to retrieve the required data in the information model. Geo-locations are drawn to the world map view using the Leaflet JavaScript library[6].

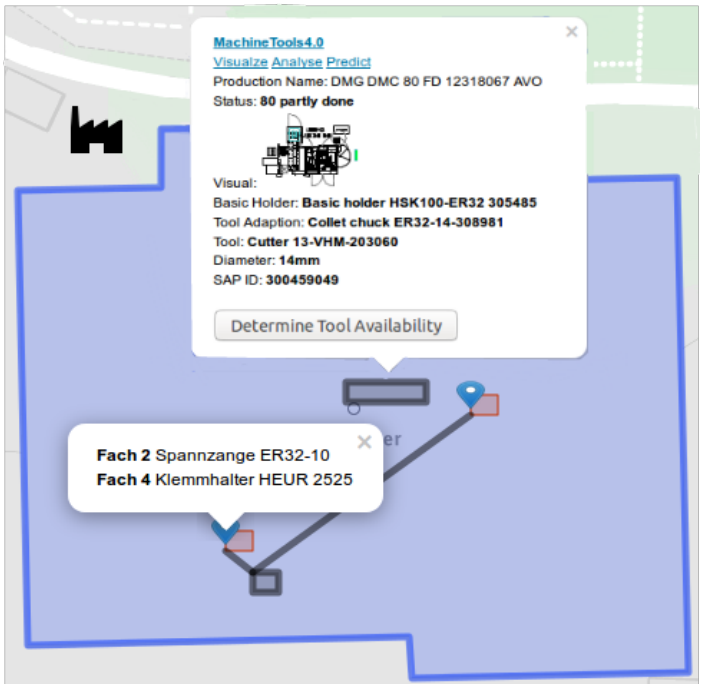
[5] `http://www.w3.org/TR/r2rml/`
[6] `http://leafletjs.com/`



(a) Global view of the company sites on a world map.



(b) Paternoster view on the factory level.



(c) Factory view with certain machines selected.

Figure 6.4: Web-based user interface for displaying geographical data. The worldmap is interactive and allows the user to explore additional information by selecting certain elements and to further run certain functions by clicking on for example the *Determine Tool Availability* button.

```
1  @prefix rr: <http://www.w3.org/ns/r2rml#> .
2  @prefix im: <http://iais.fraunhofer.de/vocabs/infomodel#> .
3  <WorkOrderMap> a rr:TriplesMap ;
4      rr:logicalTable [ rr:tableName "WorkOrders" ];
5      rr:subjectMap   [ rr:template "http://.../infomodel/WorkOrder/{WorkOrderId}";
6      rr:class im:WorkOrder];
7      rr:predicateObjectMap
8        [rr:predicate im:workOrderId;     rr:objectMap [rr:column "WorkdOrderId"]],
9        [rr:predicate im:beginWorkOrder; rr:objectMap [rr:column "BeginWorkOrder"]],
10       [rr:predicate im:targetAmount;    rr:objectMap [rr:column "TargetAmount"]],
11       [rr:predicate im:totalExecTime;   rr:objectMap [rr:column "TotalExecTime"]],
12       [rr:predicate im:matDesc;         rr:objectMap [rr:column "MatDesc"]];
13       ...
```

Listing 6.1: Example of an R2RML mapping for work orders.

Energy Consumption

Information about the energy, power or temperature are critical for the company to forecast the production process, expenses and maintenance. In the second use case, we asked the following question: what is the *energy consumption* of a given machine for a given day for a particular *work order*? To answer this question, data from sources introduced in Section 6.1.1 (SD, MES, BOM) needs to be taken into account. Since the SD lacks work order definitions, we used time intervals to access the required energy stream data. Next, we linked the work order IDs in the BOM and MES databases. Listing 6.1 displays an excerpt of the R2RML mappings for the relational database table `WorkOrders`. Among others, it includes its material number, total execution time and target production amount.

Based on these mappings, we defined two queries: The first one retrieves information about work orders (cf. Listing 6.2); the second one retrieves the energy consumption values for a work order in a specific time interval (cf. Listing 6.3).

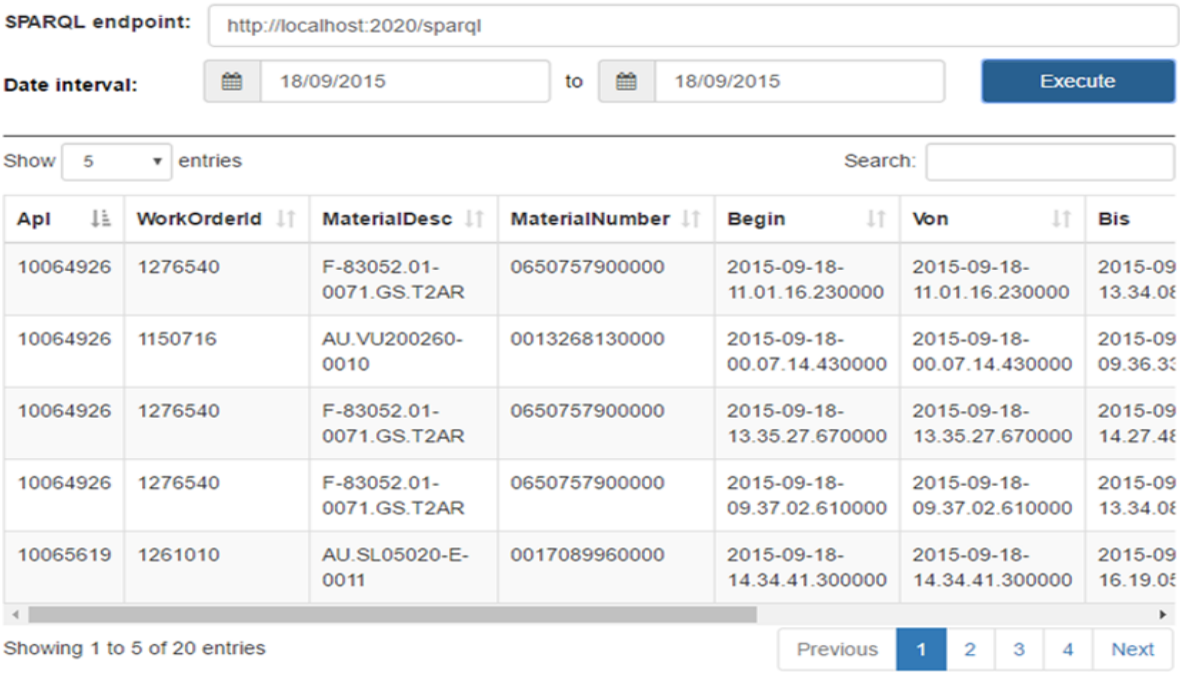
```
@prefix im: <http://iais.fraunhofer.de/vocabs/infomodel#>

SELECT DISTINCT  ?workOrderId ?materialDesc ?materialNumber
  ?beginTime ?dateFrom ?dateTo ?totalExecTime ?targetAmount
WHERE {
  ?o a im:workOrder;                  im:workOrderId ?workOrderId ;
     im:beginWorkOrder ?beginTime;    im:dateFrom ?dateFrom;
     im:dateTo ?dateTo;               im:targetAmount ?targetAmount;
     im:totalExecTime ?totalExecTime; im:matNr ?materialNumber;
     im:matDesc ?materialDesc.
FILTER (?dateFrom >= dateFrom && ?dateTo <= dateTo)}
```

Listing 6.2: SPARQL query to retrieve information about work orders within a specified timespan.

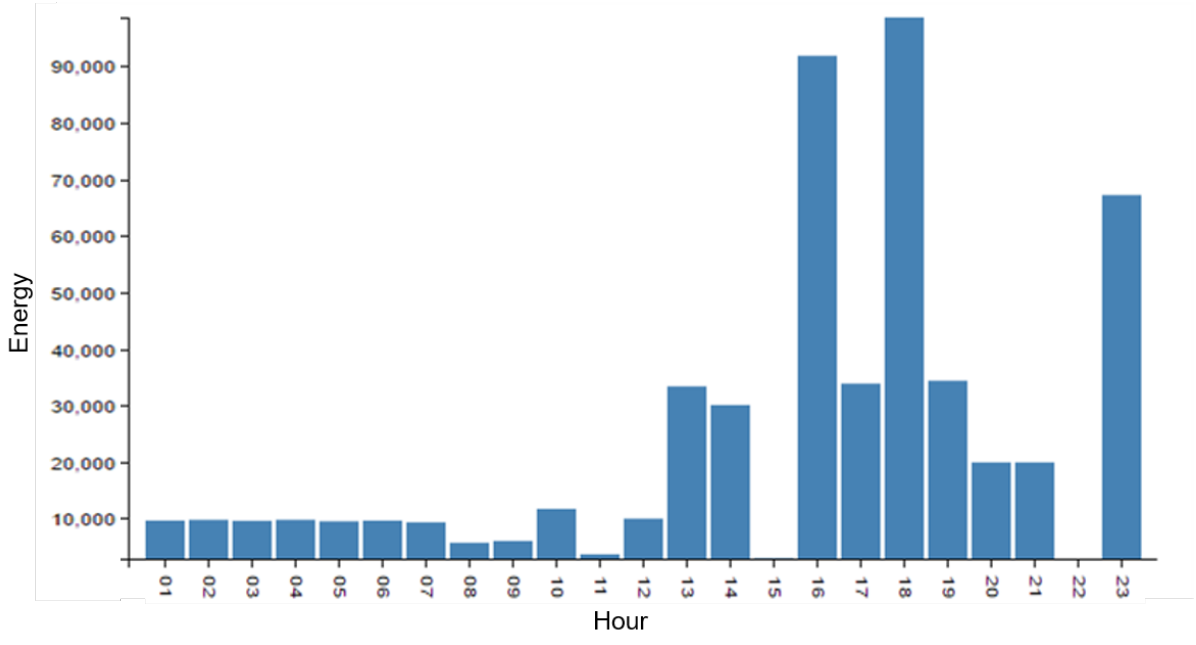
```
@prefix im: <http://iais.fraunhofer.de/vocabs/infomodel#>

SELECT ?hour ((?latest - ?earliest) AS ?measurementByHour)
{
  { SELECT ?hour (MIN(?measurement) AS ?earliest)
    WHERE
    {
      ?machineSensor im:energyTime ?time;
                     im:energyValue ?measurement.
    } GROUP BY (HOURS(?time) AS ?hour)
  }
  { SELECT ?hour (MAX(?measurement) AS ?latest)
    WHERE
    {
        ?machineSensor im:energyTime ?time;
                       im:energyValue ?measurement.
    } GROUP BY (HOURS(?time) AS ?hour)
  }
  FILTER (?hour>= timeFrom && ?hour <= timeTo) }
  ORDER BY ?hour
```

Listing 6.3: SPARQL query to retrieve the energy consumption for each machine per hour.

This allowed us to integrate the information from the three data sources using the information model and SPARQL queries. Figure 6.5(a) depicts the integrated information of work orders for a given machine, and Figure 6.5(b) shows the energy consumption per hour for that machine for a given day. Overall the performance of the implemented solution was satisfactory, i.e., time to retrieve the information for energy consumption of particular work order was less than five seconds.



(a) Web-based user interface to display work order data for a given machine within a specified timespan.



(b) Web-based user interface to display the energy consumption of a given work order within of a specified day for each hour.

Information Model Governance

Introducing new technologies is often a challenge for companies. The introduction has to be well-aligned with the organizational structure of the company to balance the added value produced for the information model to the business and the maintenance costs of the technology. Thus, in parallel with the introduction of the information model, we defined a procedure to support the *governance* of information to ensure the maintenance of the model and uniform decision-making processes.

Since the core of the information model is a network of ontologies and vocabularies with a clear hierarchical and modular structure, there are boards of experts assigned to each part, which are responsible for its maintenance. Decisions cover, for instance, new terms to be included or existing ones to be removed, external vocabularies to be reused and aligned, and the continuous alignment with industry standards implementing the Industry 4.0 vision, e.g., RAMI, IEC, ISO. Additionally, we provided concrete guidelines for maintaining the information model along with the use of VoCol, for example[7]

- detailed documentation of all terms defined in the vocabulary by `skos:prefLabel`, `skos:altLabel`, and `skos:definition`;
- multilingual definition of labels, i.e., in English and German;
- definition of `rdfs:domain` and `rdfs:range` for all properties;
- inclusion of provenance metadata, licenses, attributions, etc.

6.1.5 Evaluation and Lessons Learned

To gain feedback from the stakeholders involved in the information modeling project, we designed a questionnaire and sent it to the stakeholders, asking for anonymous feedback. Table 6.1 lists the questions and results of the questionnaire. We were interested in how the stakeholders evaluate the developed information model and semantic technologies in general, based on the experience they gained in the project.

6.1.6 Stakeholder Feedback

Five employees of the manufacturing company (three IT experts, one analyst, and one consultant) who were actively involved in the project answered the questionnaire. The results varied across the stakeholders: While some regarded the information model and future potential of semantic technologies as promising, others remained skeptical about its impact within the company. Question 6 asking for the expectations towards semantic technologies (cf. Table 6.1) was answered by nearly all as an "enabler for autonomous systems" and by one as a "potential technology to reduce the number of interfaces". One stakeholder praised the "integration and adaption" capabilities of semantic technologies. Question 7 asking for the biggest bottleneck yielded the following subjective answers: "lack of standardized upper ontologies", "lack of field-proven commercial products", "lack of support for M2M communication standards", "skepticism of the existing IT personnel". While the stakeholders find the advantages of semantic technologies appealing, the lack of ready-to-use business solutions, industrial ontologies and available IT personnel is halting their efforts to move forward. As a result of the project, the company is actively seeking IT personnel with a background in semantic technologies.

[7] These are based on the W3C "Data on the Web Best Practices" Recommendation, `https://www.w3.org/TR/dwbp/`

Question (with Likert scale of 1 to 5, 1 = not at all, 5 = very much; M = mean value, SD = standard deviation)	**M**	**SD**
1. Did the developed RDF-based information model meet your expectations?	2.4	0.9
2. Do you think investing in semantic technologies can result in a fast ROI?	3.0	1.4
3. Do you consider semantic technologies fit for usage in the manufacturing domain?	3.6	0.9
4. Are you satisfied with the software for semantic technologies available on the market?	2.8	1.3
5. Is it easy to hire personnel with knowledge in semantic technologies?	1.8	0.4
Free-text questions:		
6. What do you expect from semantic technologies in manufacturing contexts?		
7. What is the biggest bottleneck in using semantic technologies in manufacturing contexts?		

Table 6.1: Questions of the questionnaire and answers of the stakeholders

Lessons Learned

Technology awareness within the company After all, the majority of the stakeholders were enthusiastic and committed to developing an integrated information model and applications on top of it. Nevertheless, reservations on the fitness of the technology and methodology existed from the start. A few stakeholders preferred a bottom-up approach of first gathering and generating internally an overview of the existing schemas and models before involving external parties (such as our research institute). However, the management preferred an *outside view* and put a focus on quick results. Instead of spending time on finding an agreement on how to proceed, speed was the major driving force. Thus, they preferred to try out a (for them) "new" technology and methodology, which does not yet have the reputation of strong industrial maturity.

Perceived maturity of semantic technologies While semantic technologies are already widely used in some domains (e.g., life sciences, e-commerce or cultural heritage), there is a lack of success stories, technology readiness and show-case applications in most industrial areas. With regard to smaller and innovative products, the penetration of semantic technologies is still relatively small. A typical question when pitching semantic technologies within companies is "Who else in our domain is using them already?". Therefore, it is important to point to successful business projects, even if details on them are usually rare.

Lack of semantic web professionals on the job market Enabling the employees of the manufacturer to extend the information model by themselves is crucial for the success of the project. Consequently, it is necessary to teach selected stakeholders the relevant concepts and semantic technologies. Hiring new staff experienced with semantic technologies is not necessarily an easy alternative. Compared to relational data management and XML technologies, there is still a gap between the supply of skilled semantic technology staff and the demand of the market.[8]

Importance of information model governance Of major importance for the company is a clear governance concept around the information model, answering questions such as who or which department is allowed to access, modify and delete parts of the information model. An RDF-based information model has advantages in this regard: i) it enables people across all sites of the company to obtain a holistic

[8] For the related field of data science, the European Data Science Academy has conducted extensive studies highlighting such a skill/demand gap all over Europe; cf. Deliverables D1.2 and D1.4 ("Study Evaluation Report 1/2") downloadable from `http://edsa-project.eu`.

view of company data; ii) current data source schemes are enriched with further semantic information, enabling the creation of mappings between similar concepts; and ii) developers can follow a defined and documented process for further evolving and maintaining the information model.

Building on top of existing systems Accessing data from the existing infrastructure as a virtual RDF graph was an important requirement of the manufacturing company. It avoids the costs of materializing the data into RDF triples and maintaining them redundantly in a triple store, and at the same time, benefits from mature mechanisms for querying, transaction processing, security, etc. of the relational database systems. Three different data access strategies were considered:

DB in Dumps Relational data to be analyzed is dumped in an isolated place away from the production systems, as not to affect their safety and performance. This strategy is used in cases where the amount of data is small and most likely to be static or updated very rarely.

DB in Replication All data is replicated, allowing direct access from both production systems and new analytic platforms. This solution was considered in cases where data changes frequently and the amount of data is relatively high. It requires allocation of additional resources to achieve a "real-time" synchronization and to avoid performance degradation of the systems in production. We used this strategy to implement our solution, since it allows accessing the data sources as a virtual RDF graph and benefit from the maturity of relational database systems.

DB in Production The strategy of accessing data in real-time systems does not require allocating additional resources, such as investment in new hardware or software. Since this strategy exposes a high risk for performance degradation of the real-time systems, whereas sensitive information requires high availability and not providing it on time can have hazardous consequences, we did not apply it in our scenario.

6.1.7 Concluding Remarks

We have presented a case study on realizing an RDF-based information model for a global manufacturing company using semantic technologies. The information model is centered around machine data and describes all relevant assets, concepts and relations in a structured way, making use of existing as well as specifically developed ontologies. Furthermore, it contains a set of RML mappings that link different data sources required for integrated data access and SPARQL querying. Finally, it is aligned to relevant industry standards, such as *RAMI* [86] and *IEC 62264* [88], to additionally foster data exchange and semantic interoperability. We described the used methodology to develop the information model, its technical implementation and reported on the gained results and feedback. Additionally, we reflected on the lessons learned from the case study.

As for the enterprise, a high-level ontology is under development to extend the existing information model with our guidance. Its goal is to describe entire business units, their processes and assets for the entire organization.

The use of data-centric approaches in engineering, manufacturing and production is currently a widely discussed topic (cf. the related initiatives concerning Industry 4.0, the Industrial Internet or Smart Manufacturing). The challenges and complexity of data integration are perceived as a major bottleneck for the comprehensive digitization and automation in these domains. A key issue is to efficiently and effectively integrate data from different sources to ease the management of individual factories and production processes up to complete companies. The presented information model is envisioned to serve as a crystallization point and semantic reference in this context.

Future work concerns the continuous translation of relevant industry concepts and standards into RDF as well as their integration and alignment with existing ontologies and vocabularies. In addition, further ontologies for different industry domains need to be developed to enable data integration and semantic interoperability within and between companies. There is also a lack of related business processes and governance models, as it was shown by the case study.

From a technical point of view, further support for ontology-based data access would be needed to achieve the envisioned scalability. While we had good experiences with Ontop as an OBDA framework, complete coverage of SPARQL 1.1 is not yet given and challenging to achieve [92]. Further, R2RML supports only RDB-to-RDF mappings, while other types of data sources are not covered. Although there exist first proposals to extend R2RML beyond relational databases (e.g., RML[9]), tool support for these extensions is limited and not yet on the same maturity level as for R2RML.

6.2 Automating Factories with Semantic Technologies

This section is dedicated to explore an ontology and an integration infrastructure to obtain a holistic view of the status of a factory from different perspectives. The content of this chapter is based on the publication [68].

6.2.1 Motivation

A key motivation of our Semantic Factory model is to establish a holistic and integrated view on an enterprise in order to reduce the overall complexity and improve decision making. This includes the workforce, business processes, machines, shift plans, supply chains, etc. While a lot of this information is already captured by different IT systems, it is rarely accessible in a combined way without investing significant manual effort. Thus, the goal of this work is to make all data that is currently stored in various systems available in a unified model to support users with different roles in decision making.

An example is a factory planner who requires diverse information about order plans, workforce availability and machine maintenance dates. Another example is a machinist who needs to know which tools are to be mounted into which machine, where these tools are located, where the material is stored and what quality control standards are required during the production process. A controller, on the other hand, wants to keep track of the productivity of a factory and get an overview of certain Key Performance Indicators (KPIs). These comprise, in particular, information on the production time and effort required by each machine for each product, such as employee effort and energy consumption data.

To provide all those stakeholders with the information needed to perform their tasks and optimize decision making, we aim at semantically describing as many assets of a factory as possible, taking into account information from different manufacturing systems.

6.2.2 Requirements

We elicited the following requirements in the context of a research project for a global manufacturing company. The company's objective to gain a better picture of its assets (e.g. machines and factories) led us to develop an ontology that serves as the core element in the overall architecture.

From descriptions of the assets provided by the company and from interviewing domain experts, we gained an overview on typical tasks, processes and problems of each stakeholder. The interviews

[9] http://rml.io/spec.html

took place at the company site in multiple meetings, where the company's current IT infrastructure was described in detail. That way, we gathered requirements for the Semantic Factory model step by step:

Semantic Multi-Modality. The types of data found in a factory context are diverse. Hence, the representation of various information, including attribute trees, relational, sensor, tabular, graph and entity data, must be supported.

Multi-Dimensionality. Information along several dimensions must be represented and captured, such as:

- *Business processes:* Temporal views on diverse business activities are required to judge the success of an enterprise.
- *Spatial hierarchies:* The exploration of assets from a geographical perspective must be possible to increase the findability of said assets and related information.
- *Lifecycle:* Product and business lifecycles must be represented to support strategy management and business innovation.

Multi-Granularity. Views on different levels of detail must be provided:

- *Components:* Instant access to sensor and component data must be enabled to support possible intervention measures.
- *Factories:* To decrease the complexity of factories, master and operational data needs to be accessible in a singular view.
- *Organization:* A big picture of all business units is needed, including their hierarchies and responsibilities.

Traceability and Integration. Data and information is currently spread across various systems, such as manufacturing execution systems, quality assurance systems, enterprise resource planning systems, etc. It is important to integrate all relevant information from these systems, while maintaining the systems' record-keeping character. When integrating information from these systems, the provenance of the data must be preserved, and changes to information in the source systems must be reflected in the integrated views, wherever possible in real time.

6.2.3 Architecture

Based on the requirements presented in Subsection 6.2.2, we designed an architecture for a Semantic Factory application (see Figure 6.6). Its core is a factory ontology, which describes real world objects, such as employees, machines and factories, locations of assets and their relations with each other. Operational data, such as information about work orders, machine sensor and process data, is also covered by the factory ontology; in practice, this data is dynamically mapped from the respective databases to the ontology.

The ontology is made available through an RDF triple store. Different applications can execute queries on the data, which is expressed as RDF or available in relational databases. Finally, for geospatial data, an external map provider service is used for drawing, among others, factories on a world map.

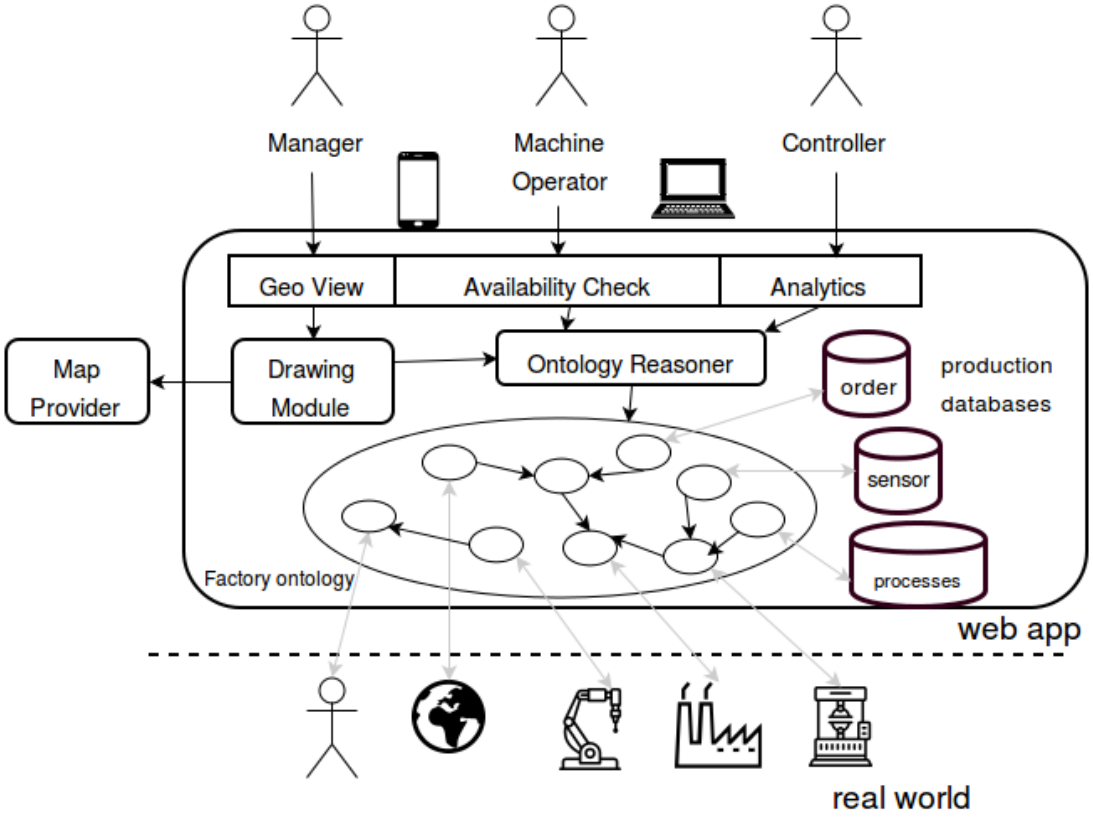


Figure 6.6: Proposed architecture for the *Semantic Factory*. The factory ontology (middle) represents things in the real world (bottom) including links to the respective databases (right) which holds the data needed for certain analytical tasks (top). An external *drawing module* for displaying information on a map as well as an *ontology reasoner* are included to support certain analytical tasks.

6.2.4 Ontology

The following subsections explain how we developed the factory ontology following the methodology proposed by Uschold et al. [70]. We first defined the purpose and scope of the ontology; then, we captured step-by-step the domain knowledge, conceptualized and formalized the ontology and aligned it with existing ontologies. Finally, we evaluated the ontology by measuring the performance of certain queries for different sized datasets (see Subsection 6.2.7).

Purpose and Scope

The purpose of the ontology is to provide a holistic view of an enterprise. This is realized by implementing the requirements specified in Subsection 6.2.2. The intended users are different stakeholders of an enterprise, such as managers, machine operators and controllers. Each of them needs different information to effectively and efficiently perform the corresponding tasks and duties. Thus, the ontology enables viewing the factory from different perspectives to support each class of stakeholders in their decision making.

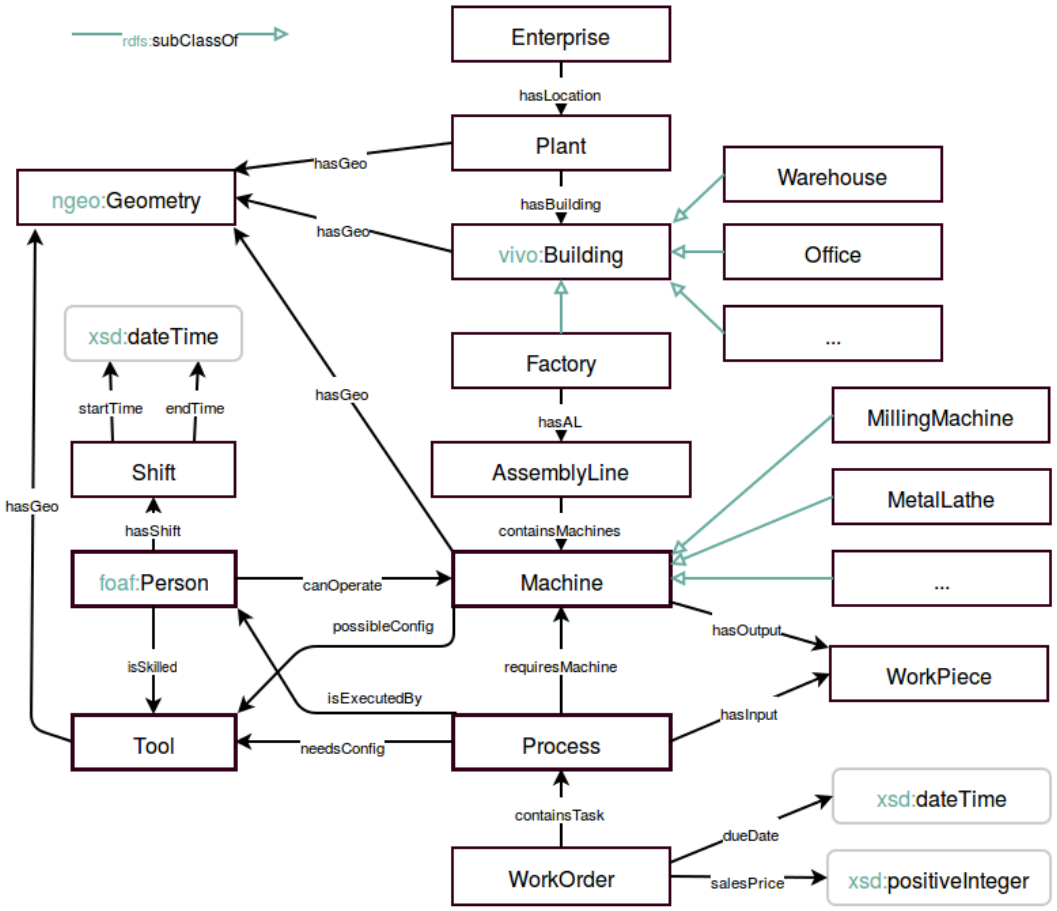


Figure 6.7: Core concepts of the *factory ontology*

Capturing Domain Knowledge

We captured the domain knowledge in three ways:

1. The company provided us with descriptive material of the domain, including maps of factories, descriptions of machines and work orders, process information, sensor data and tool knowledge. The types of input material ranged from formatted and unformatted text documents to spreadsheets and SQL dumps.

2. A live demonstration of a particular machine execution was given, including a discussion of further contextual information which was missing in the material. In subsequent meetings, open questions were clarified and concrete use cases of the ontology were discussed.

3. We reviewed relevant existing ontologies with the intention to build upon available conceptualizations and formalizations of domain knowledge.

Conceptualizing and Formalizing

The resulting ontology comprises 86 classes, 73 object properties and 142 datatype properties. Since it has been designed to support an industrial project with sensitive business logic descriptions, not the entire ontology could be made publicly available. However, the part of the ontology that can be published is accessible via a permanent URL[10]. In the following, we describe the ontology, starting from the high-level organizational layer to the low-level machine and sensor layers. The core concepts of the ontology are depicted in Figure 6.7.

In any concrete scenario, the class `Enterprise` is instantiated to represent the organization to which all further resources belong. Possible instances may be "Volkswagen", "General Electric" or "Samsung". Inter-organizational supply chains could involve multiple enterprises. Different production locations of an enterprise are described using the class `Plant`. A plant can comprise one or many `Building`s, such as an office building, a factory building or a warehouse building. Typically, we can assume that each building serves a single function, though other configurations can also be represented, as OWL ontologies support multi-membership. Therefore, the subclasses of `Building` are not defined to be disjoint in the ontology.

Each `Factory` building may have one or multiple `AssemblyLine`s. An assembly line usually consists of a sequence of multiple machines. The classes `MillingMachine` and `MetalLathe` are examples of subclasses of the abstract class `Machine`. Machines can be configured to use certain `Tool`s to produce specific work pieces. As an example, a milling machine may be equipped with different lathes or end mills of varying granularity. `WorkPiece`s represent everything which is an output of a machine. Once a work piece reaches its final stage of production, it becomes a product and is ready for shipping. Plants, factories and machines may have a representation of their geographical location (`ngeo:Geometry`[11]), which can, for instance, be used by front-end applications to display them on a map. A clear description of the entire capital of an enterprise supports the controllers and managers to keep track of utilized and unutilized assets.

As a next step, we describe the part of the ontology that represents the everyday operation of a factory. Employees are instances of the class `foaf:Person`. Each employee is qualified to operate specific machines (`canOperate`) and skilled to use certain tools (`isSkilled`). The class `WorkOrder` describes orders driven by customers. Each such order contains one or more `Process`es, which need to be executed to fully complete the order. Typical processes may be the configuration of a machine, the execution of a machine, quality control, etc. Each order has a due date and a sales price. The properties `requiresMachine`, `hasInput`, `needsConfig`, `isExecutedBy` are used together with the `Process` class. For example, they define which `Machine` is required for that process together with the needed configuration (tools assembled) and the input `Material`. Finally, it is described which employees have the skills use certain tools (`isSkilled`). All these details are required by the machine operator in the event of reconfiguring the machine for a specific order.

Aligning with Existing Ontologies

The ontology includes concepts and properties from well-known ontologies. The *Semantic Sensor Ontology*[12] provides us with a rich description of sensors, their measurements, devices and related concepts. The workforce and employees are described based on definitions by the *Friend of a friend* (FOAF)[13]

[10] `https://w3id.org/i40/smo/`
[11] Prefixes are defined according to `http://prefix.cc`.
[12] `https://www.w3.org/2005/Incubator/ssn/ssnx/ssn`
[13] `http://xmlns.com/foaf/spec/`

ontology. Coordinates of factories and machines are based on the latitude and longitude definitions of the *W3C Geo Vocabulary*[14]. Finally, we reused geometrical concepts, such as the representation of polygons, from the *NeoGeo Geometry Ontology*[15].

6.2.5 Implementation

We developed a software system that implements the presented Semantic Factory architecture and ontology and applied it to industry data. In this section, we first describe the front-end and back-end implementation. Then, we illustrate its usage by presenting various SPARQL queries that retrieve information for different production management tasks.

Front-end

The front-end is realized as a web application to facilitate access from different devices. The decision is motivated by the diversity of IT systems, platforms and devices usually deployed in a factory. The application uses the web framework *AngularJS*[16], which follows the model-view-controller design pattern to separate logic from representation. We created multiple views to address the collected requirements:

The *map view* (Figure 6.8(a)) projects all instances (e.g. buildings, machines) with a geographical representation on a map by making use of the map provider *MapBox API*[17]. This API offers map tiles based on the open geographical database *OpenStreetMap*[18]. The projection itself is realized using the *leaflet.js*[19] JavaScript library. Coordinates in the factory ontology are represented using the *NeoGeo Geometry Ontology*[20] concepts and properties translated into `leaflet` geographical objects to be drawn on the map.

Further information, such as the person currently operating a machine, which order is executed or the status of a machine, is provided in the *machine view* that can be opened from the map view (see Figure 6.8(b)) or independently by the machine operator. Besides static information, the pop-up contains also links to operational views and services. For example, the machine operator can follow the order link to retrieve additional information of that order. Furthermore, based on the tools required for the next machine operation, the "Find available tools" functionality points to the respective geographical location of the tools in the factory. The links *Visualize*, *Analyze* and *Predict* point to external pages that provide additional graphical content about the machine, machine usage indicators and prediction dates when machine parts are worn out and need to be replaced.

Back-end

The back-end consists of a Python web server[21] that supports REST API calls from the front-end. Each request triggers the generation of SPARQL queries executed either on the factory ontology or the production databases. To provide access to the ontology, the Python library *rdflib*[22] is used.

[14] `https://www.w3.org/2003/01/geo/wgs84_pos`
[15] `http://geovocab.org/geometry.html`
[16] `https://angularjs.org`
[17] `https://www.mapbox.com/developers/`
[18] `http://www.openstreetmap.org`
[19] `http://www.leafletjs.com`
[20] `http://geovocab.org/geometry.html`
[21] `http://flask.pocoo.org`
[22] `https://github.com/RDFLib/rdflib`

(a) Map View (b) Machine View

Figure 6.8: Web-based user interface for displaying information of factory sites and machines on a world map.

Access to the relational production databases is realized using the D2RQ[23] system. D2RQ provides a generator for creating an RDF mapping file for the database tables and columns, thus preserving the schema of the relational database. Using the mapping file, incoming SPARQL queries are translated ad-hoc into SQL queries and are executed on the respective database. Thus, D2RQ acts as a gateway between the web server and relational databases.

Once the data is obtained, it is returned in JSON data format[24] and processed by the respective front-end controller.

6.2.6 Factory Queries

In the following, we provide a set of SPARQL queries that demonstrate the usage of the factory ontology.

Order Feasibility Check.

Listing 6.4 shows a query that determines if certain machines in the factory are free to use or already scheduled for other production plans. Each factory work order contains a list of tasks to be completed by a different machine. Thus, each needed machine is checked for its availability. Only if all machines are available, the query returns a positive answer such that the work order can be started. The query always checks the current state of the factory.

```
1  ASK
2  {    # get tasks
3       ?order a :WorkOrder .
4       ?order :requiredMachines ?machineList .
5       ?machineList rdfs:member ?machine.
6
7       # check if the needed machines are free
8       EXISTS { ?machine :isFree false } .
9  }
```

Listing 6.4: SPARQL query for checking if machines are available to execute a specific order.

[23] http://d2rq.org
[24] http://json.org

Retrieve Geographical Coordinates.

Listing 6.5 shows a query to retrieve the machines, their names and coordinates. The outline of a machine is conceived as a polygon, represented as an `rdf:List` of geographical points, each with latitude and longitude, which is linked to the machine using the `ngeo:posList` datatype property. This information is returned to the front-end to be projected on the world map.

```
SELECT ?machine ?label
            (GROUP_CONCAT( ?lat ; separator=";") AS ?lats)
            (GROUP_CONCAT(?long ; separator=";") AS ?longs)
WHERE {
    ?machine rdfs:label ?label .
    ?machine ngeo:posList/rdf:rest*/rdf:first ?point .
    ?point geo:lat ?lat .
    ?point geo:long ?long .
} GROUP BY ?machine ?label
```

Listing 6.5: SPARQL query to retrieve geographical coordinates of machines.

Suitable Assembly Lines.

Listing 6.6 shows a query to find suitable assembly lines with regard to their sequence. Assembly lines that contain more machines than required but fulfill the correct order are still considered suitable. Thus, certain stations may be skipped within an assembly line.

Suppose, for example, that the machines `2` and `4` are required for an order. Suitable assembly lines include those having the following sequences of machines: `2,4` or `1,2,3,4,5`. Sequences such as `4,2` or `4,3,2` are considered non-suitable.

The query itself works as follows: First, assembly line candidates are filtered (`MINUS`) based on whether they contain the required machines in the work order. Second, of those assembly lines, the position of the needed machines is calculated. This is achieved by preparing the needed order (`?reqSequence`) and then retrieving the position of each machine in that order (`?machineSeq`). Third, these sequences are concatenated into strings and, finally, it is checked by a regular expression if the sequence is increasing.[25]

```
SELECT ?assemblyLine {
  ?assemblyLine :machineList ?lists .

  # filter all assembly lines with the wrong order
  FILTER REGEX(?seq,"^0*1*2*3*4*5*6*7*8*9*$")

  # concatenate order of the lists into a string
  {SELECT ?lists (GROUP_CONCAT(?machineSeq; separator="")
                AS ?seq)

   #Machine Sequence, _sorted_ by required order Sequence
  {SELECT ?lists ?machineSeq {
    {SELECT ?lists ?machineInstance ?machines
    (STRAFTER(STR(?memberProp), "_") AS ?machineSeq)
           {?lists rdfs:member ?machineInstance .
            ?lists ?memberProp ?machineInstance .
            ?machineInstance a ?machines . }}

   # get required Sequence of the Work Order
   {SELECT ?machines (STRAFTER(STR(?prop), "_")
                      AS ?requiredSeq) {
```

[25] The query is limited to sequences of up to 9 machines but may be extended.

```
22          :sampleOrder :requiredMachines ?orderList .
23          ?orderList   ?prop             ?machines .
24          FILTER (STRSTARTS(STR(?prop), STR(rdf:_)))}
25          ORDER BY ?requiredSeq}
26
27        # Identify Assembly Lines Candidates
28      {SELECT ?lists ?assembly
29              {?assembly d:machineList ?lists.}}
30       MINUS
31      {SELECT ?lists {
32              ?lists       a                  rdf:Seq .
33              ?workOrder   :requiredMachines  ?neededMachineList .
34              ?neededMachineList  rdfs:member ?machineType .
35        FILTER NOT EXISTS{?lists (rdfs:member/a) ?machineType .}}}
36       } ORDER BY ?lists ?requiredSeq
37    } GROUP BY ?lists }}
```

Listing 6.6: SPARQL query to identify a suitable assembly line needed for a specific order.

6.2.7 Performcen Evaluation

We evaluated our factory ontology and software application by testing the performance of the SPARQL queries introduced in Subsection 6.2.6. These queries were chosen due to their representativeness in an industrial setting. For that, we prepared multiple datasets, consisting of 10K, 100K, 1M, 2M and 5M triples. The datasets contain generated test data based on our factory ontology, such as order information, workforce details, assembly lines, etc.

The queries were executed using the *ARQ* SPARQL processor version 2.13.0[26]. The machine we used for the experiment contains 8GB of RAM, 256GB SSD and an Intel i7-3537U CPU with 2.00GHz.

Figure 6.9 depicts the results of the performance evaluation. While the growth for the "Retrieve machine coordinates" query of Listing 6.5 is linear, a response time of 25 seconds in larger datasets is not satisfactory for a front-end application. Thus, large datasets should be split to keep the execution time end-user friendly.

Similarly, as in the previous query, the execution time for the "Order Feasibility Check" query of Listing 6.4 grows linearly. While the overall performance is slightly better, one should nevertheless keep machine status data in an isolated dataset.

Finally, for the large "Suitable Assembly Lines" query of Listing 6.6, it is rather surprising that the execution time is quite similar to the short previous queries. As before, with a linear growth, the performance evolution becomes predictable and it is recommended to split instance data depending on certain services.

Overall, ontology-centered web applications are feasible with a satisfactory performance. As a common rule of thumb, the acceptable response time for complex operations is less than 10 seconds in order to keep the user's attention on the task [93]. Thus, certain crucial instance data should be kept in different triple stores to stay below that threshold.

However, the `d2rq` system for accessing relational databases reveals performance issues. First experiments using `ontop`[27] to query databases seem to be a promising alternative.

[26] `https://jena.apache.org/documentation/query/`
[27] `http://ontop.inf.unibz.it`

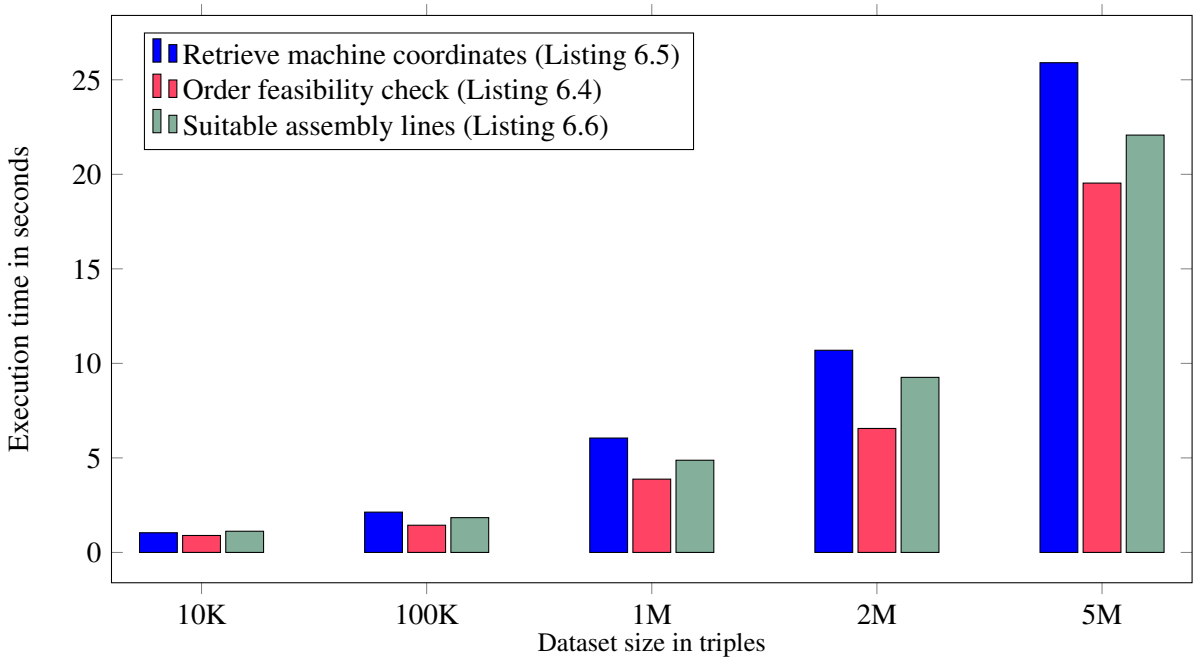


Figure 6.9: Query execution performance of the proposed SPARQL queries for the *Semantic Factory*.

6.2.8 Concluding Remarks

The use of data-centric approaches in engineering, manufacturing and production is currently a widely discussed topic (cf. Industry 4.0, smart manufacturing or cyber-physical systems initiatives). The complexity of data integration in general is perceived to be one of the major bottlenecks of the field. A key issue in engineering, manufacturing and production is to be able to efficiently and effectively manage factories.

This paper described an ontology and an integration infrastructure to obtain a holistic view of the status of a factory from different perspectives. We see the work presented in this paper as a first step towards establishing an ontology-based integration approach for manufacturing, which is centered around a common information model, but at the same time supports the management of data in a decentralized manner in the existing systems of record. The integration follows a loosely-coupled architecture, where the decentralized data sources are mapped on demand to the factory ontology. The factory ontology is not supposed to be a fixed, monolithic schema, but rather a flexible, evolving and interlinked knowledge fabric. For this purpose, we have developed the collaborative vocabulary development methodology and support environment VoCol [94].

We see a number of directions for future work. In particular, the Semantic Factory approach could be expanded from single factories to an integration approach covering the entire enterprise as well as supply networks (e.g. based on the SCOR model [65]) involving a large number of organizations. Another promising direction of future work is the exploitation of the integrated data for advanced analytics and forecasting [95].

6.3 Semantic Integration in the International Data Spaces

This section is dedicated to explore the implementation of a data space: The *International Data Spaces*, a research initiative funded by the German government. This chapter describes how the initiative is organized, what results have emerged and to present and explain the core components in detail. The content of this chapter is based on new research and of the publications [83, 84][28]

6.3.1 Motivation

IDS

The International Data Spaces (IDS, formerly: Industrial Data Space) started as a research project in 2015 by the German government. As of 2019, the IDS contains ~100x members from academia and private organizations. The mission of the IDS is to create "a secure data space that supports enterprises of different industries and different sizes in the autonomous management of data."[29].

The academic institutions are mostly represented by Fraunhofer institutes which focus on applied research and the private organizations by multiple globally operating industrial enterprises.

6.3.2 Discussion of the IDS Architecture

The architecture of the IDS is described in the *Reference Architecture Model* [15] document published by the initiative. The document puts the entire vision into perspective and provides a more detailed discussion on its realization. That is, any implementation needs to be compliant with the specifications described in that document. It should be noted that the reference architecture model is refined nearly each year and is in its third version in the year 2019.

IDS Connector

The *IDS Connector* acts as the core component in the IDS. To quote the definition in the *Reference Architecture Model* [15]:

> The Connector Architecture uses application container management technology to ensure an isolated and secure environment for individual data services. A data service matches a system which offers an API to store, access or process data.

Any organization who wants to participate in the IDS needs to set up an IDS Connector to be able to subscribe and publish data. That is, the IDS Connector represents the gateway from an organization to the outside via an API of their choice. The IDS purposely decided to not commit themselves to a specific API protocol to allow its members to make their own choice based on their requirements. However, the chosen API protocol needs to be formally described and published in a so called *self-description* so that other participants in the IDS can deduce how to subscribe to that IDS Connector. How such a self-description looks like will be explained in the following sections.

[28] [84] is a joint work with Jaroslav Pullmann, a researcher at the Fraunhofer institute for Intelligent Analysis and Information Systems (IAIS). In this paper, I contributed to the development of the general use case, the creation of the *Steel Grade Vocabulary*, the service metadata description of the *IDS Connector* and the visual for describing the interaction of the components.
[83] is a joint work with Glykeria Alvanou, a previous MSc student at the University of Bonn. In this paper, my contributions include the overall problem definition, motivation, minor contributions to the development of the *MTConnect* with regard to semantic validation and the general supervision of her research which resulted in this publication and her MSc thesis at the University of Bonn.

[29] `https://www.fraunhofer.de/en/research/lighthouse-projects-fraunhofer-initiatives/industrial-data-space.html`

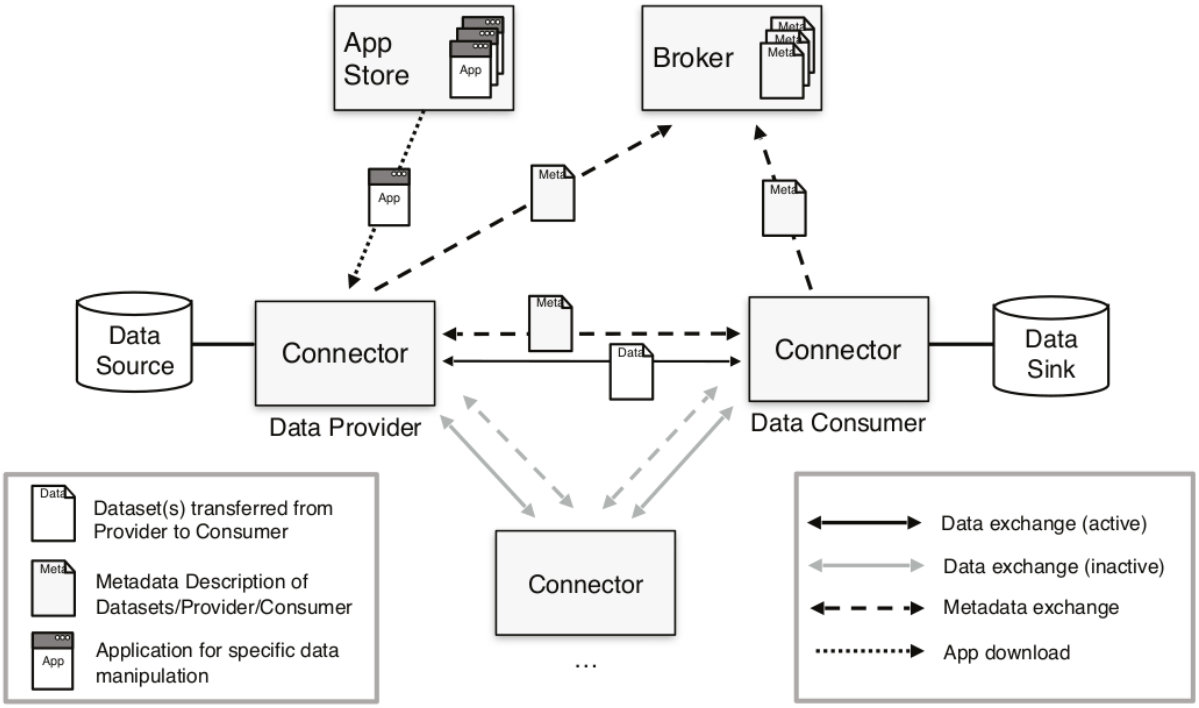


Figure 6.10: Overview of the interaction of the components in the *IDS* [15]

As described in the definition, the IDS Connector can further be employed with certain technology to provide a secure environment based on the requirements of the organization. Also, *IDS Apps* container can be deployed inside the IDS Connector to support transforming and preparing data to be ingested into legacy-systems or to run for example data quality operations.

Together, all IDS Connectors form the decentral network branded *International Data Spaces*.

IDS Information Model

The *IDS Information Model* (formerly called IDS Vocabulary) is an RDFS vocabulary and serves as a metamodel within the IDS. It provides the basis for the creation of machine-interpretable *self-descriptions* for the IDS Connector, datasets to be offered in the IDS, security profiles and contracts.

As an example: To find different IDS connectors, each connector needs to provide a self-description of what is being offered at this endpoint. The self-description needs to contain the details of the technical API implementation, including the parameters and what is to be expected if the API call was successful or failed. Furthermore, the self-description needs to contain the security profile which needs to be adhered to in case a consuming application is being built to call this API. Finally, the dataset itself should be described such that it becomes easy for the consumer to understand the structure and meaning, so that is becomes easier to make use of the data. Optionally, the self-description also contains details about a possible contract which needs to be set up in order to retrieve data from that endpoint, including a possible price.

As of 2019, the vocabulary source code is actively developed on the Git version control hosting Service *GitHub*[30] and provides currently a human-friendly version on
`http://ids.semantic-interoperability.org`[31]
but is expected to move in the future move to

[30] `https://github.com/IndustrialDataSpace/InformationModel`
[31] `http://ids.semantic-interoperability.org`

https://schema.industrialdataspace.org/[32].

IDS Broker

The IDS Broker functions as a catalog for the IDS connector metadata description. An IDS Broker may represent all IDS Connectors, or only a subnetwork. An example of a subnetwork may represent a *Medical Data Space* or a *Mobility Data Space*. If the configuration of an IDS Connector changes (e.g. different API protocol), then the IDS Broker catalog entry needs to be updated. The update is supposed to be triggered by the IDS Connector. Technically, the IDS Broker needs to have the machine-interpretable metadata description (RDF file) of the IDS Connector which is conform to the *Information Model*. The IDS Broker is supposed to offer a web user interface, such as an HTML page, which can be browsed by the participants to find useful datasets. Based on the rich metadata of the self-descriptions, various filters can be applied in the search. As an example, one may only search for data being offered to be used under certain legal conditions.

Further IDS components

The *IDS App Store* offers container applications to be deployed inside *IDS Connectors*. The *IDS Vocabulary Provider* offers vocabularies which are supposed to describe the semantics of the data in a way that the other organizations can better consume/integrate them into their systems. The *IDS Clearing House* provides services to guarantee that organizations fulfill their obligations within the IDS. These components and possible future additions are supposed to provide helpful services for the participants of the IDS.

6.3.3 Implementation of an IDS Use Case: Steel Industry

In order to show the applicability of our *IDS Information Model*, we provide a use case within the steel industry focusing on two specific aspects: 1) the usage of domain-specific vocabularies, and 2) the need for machine-interpretable interfaces. Machine-interpretability means that machines can autonomously evaluate the data based on a strong semantic definition of a particular interface and the data it provides.

Domain Vocabularies

Figure 6.11 depicts an example with two interoperability problems: Using a Stock Information endpoint, a client requests the amount of certain steel available in stock of a steel producer. Here, the client uses a different measurement unit (*pound* vs. *kilogram*) and a British Standard (BS4360) to describe the needed steel grade, while the steel producer uses the European Standard (EN10025). To solve these interoperability problems, we use open domain vocabularies to gain a shared understanding (via a "published and well-defined language") of the data in order to provide means for performing the necessary conversions.

Our *Steel Grade Vocabulary* unifies the general concept of steel grades and existing standards, such as the British standard `bs4360_40B` and the European standard `en10025_235JR`. Furthermore, the vocabulary also describes additional chemical and mechanical properties, such as the minimum and maximum amount of phosphor (`:maxPhosphor "9"`). Information about these standards and the defined properties is publicly available. However, it has not yet been explicitly homogenized in such domain vocabularies using RDF and Linked Data technologies.

The second interoperability problem of diverging measurement units is already solved by existing vocabularies. For instance, the QUDT Vocabulary [96] contains transformation formulas for different units of measurement, such as the `qudt:conversionMultiplier` (0.453595) for the conversion of pound into kilogram. Thus, by reusing the existing vocabulary, we ensure that others can *understand*

[32] https://schema.industrialdataspace.org/

the data and at the same time save the costs for developing a new model. For the sake of clarity, we do not use the QUDT vocabulary literally in our example, but use a simplified version to concisely express its intention.

Machine-interpretable interfaces

The aim of the IDS Vocabulary is to semantically describe the interfaces and datasets used in the Industrial Data Space.

```
1
2  ex:stockInformation
3    idsv:label "Bestandsauskunft für Stahl"@de ;
4    a idsv:PassiveDataSource ;
5    idsv:service ex:stockQueryService ;
6    idsv:protocolBinding ex:stockQueryServiceProtocols .
7
8  ex:stockQueryService
9    a idsv:DataService ;
10   idsv:operation ex:getStock .
11
12 ex:getStock
13   a idsv:RetrieveOperation ;
14   idsv:label "Stahlbestand abfragen"@de ;
15   idsv:input ex:grade ;
16   idsv:input ex:amount ;
17   idsv:input ex:unit ;
18   idsv:output ex:amount.
19
20 ex:grade
21   a idsv:Parameter ;
22   idsv:label "Stahlgüte"@de ;
23   idsv:name "grade" ;
24   idsv:semanticType sgv:SteelGrade ;
25   idsv:dataType xsd:anyURI.
26
27 ex:amount
28   a idsv:Parameter ;
29   idsv:label "Menge"@de ;
30   idsv:name "amount";
31   idsv:dataType xsd:double.
32
33 ex:unit
34   a idsv:Parameter ;
35   idsv:label "Gewichtseinheit"@de ;
36   idsv:name "unit" ;
37   idsv:semanticType qudt:MassUnit ;
38   idsv:dataType xsd:anyURI.
```

Listing 6.7: Machine-interpretable self-description for an IDS Connector

Listing 6.7 describes an example of an IDSV-conform IDS Connector metadata description. The IDS Connector offers for interested parties the availability of steel. Since it is a read-only IDS Connector, it is defined as a *idsv:PassiveDataSource* from the perspective of the IDSV. The technical operation to retrieve the availability is defined as *ex:getStock*. Three input parameters are available: The *ex:grade*, the *ex:amount* and the *ex:unit*. As a response, the IDS Connector will return the *ex:amount* the organization is able to fulfill.

The *ex:grade idsv:semanticType sgv:SteelGrade* triple defines to which classification the steel belongs.

The `ex:StockInformation` endpoint is intended to respond to client requests (class `idsv:PassiveDataSource`) providing the mandatory input parameters of `ex:grade` (a URI identifying a

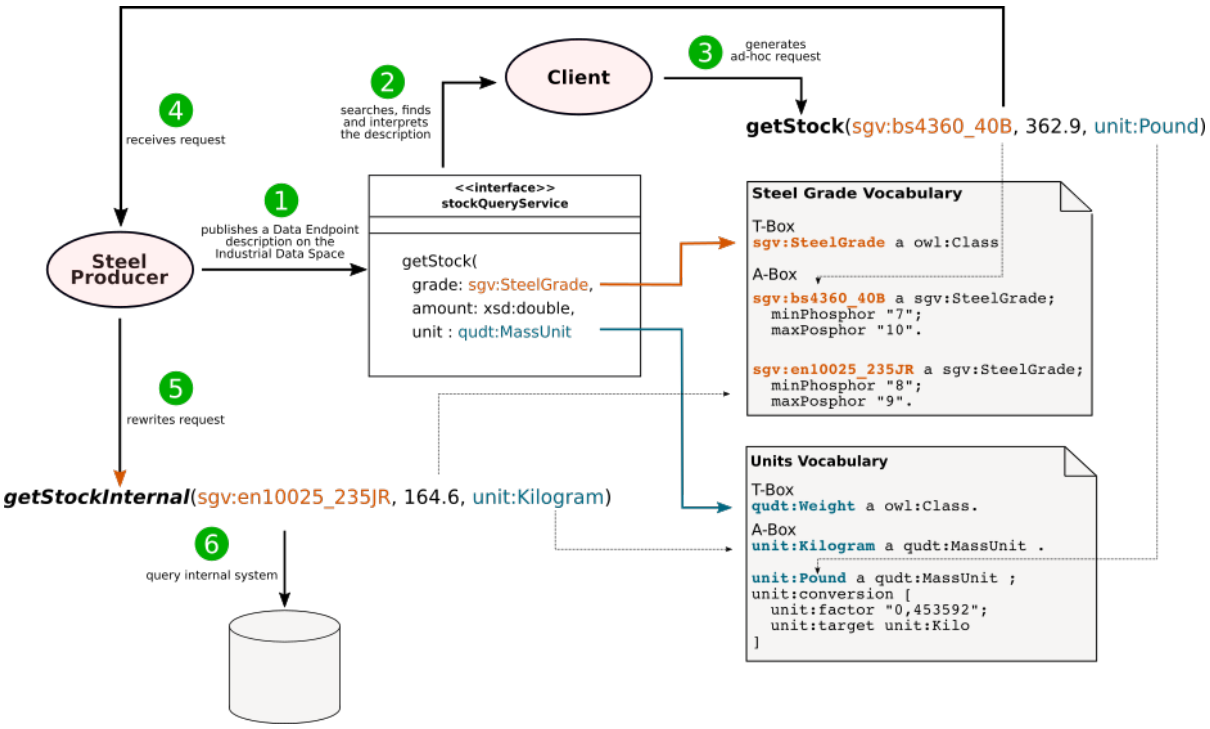


Figure 6.11: Workflow visualization for the interaction of components in the IDS including the metadata descriptions.

`sgv:SteelGrade` instance), `ex:amount` (of type `xsd:double`) and `ex:unit` (a URI identifying a `qudt:MassUnit` instance). The `ex:amount` parameter is reused to carry the result of the `idsv:RetrieveOperation`, the amount of steel effectively available.

Interoperability Workflow

The corresponding workflow is shown in Figure 6.11. A steel manufacturer (Data Provider) announces its stock information endpoint on the Industrial Data Space, indicating domain vocabularies that describe the terminology used for the parameters (steel grade and units). Based on this description, a Data Consumer can formulate a request referencing RDF individuals that represent, e.g., a certain steel grade such as `svg:bs4360_40B` and, in a similar way, uniquely named units of measurement (`unit:Pound`). The request is sent to the Data Provider which looks up the original values in domain vocabularies and infers their equivalents in order to rewrite the request. The rewritten request contains normalized values in accordance with the internal back-end system's requirements regarding the steel grade and weight. Compatible steel grades are found by mapping the physical and chemical properties, weight unit conversion is done based on the unit vocabulary's conversion factor.

It should be noted that interface methods that are published on the Industrial Data Space are not required to use RDF resources (URIs) as parameter values. However, it is recommended to follow the approach outlined in the example as it fosters a continuous development and transition to a data space with self-describing interfaces and data.

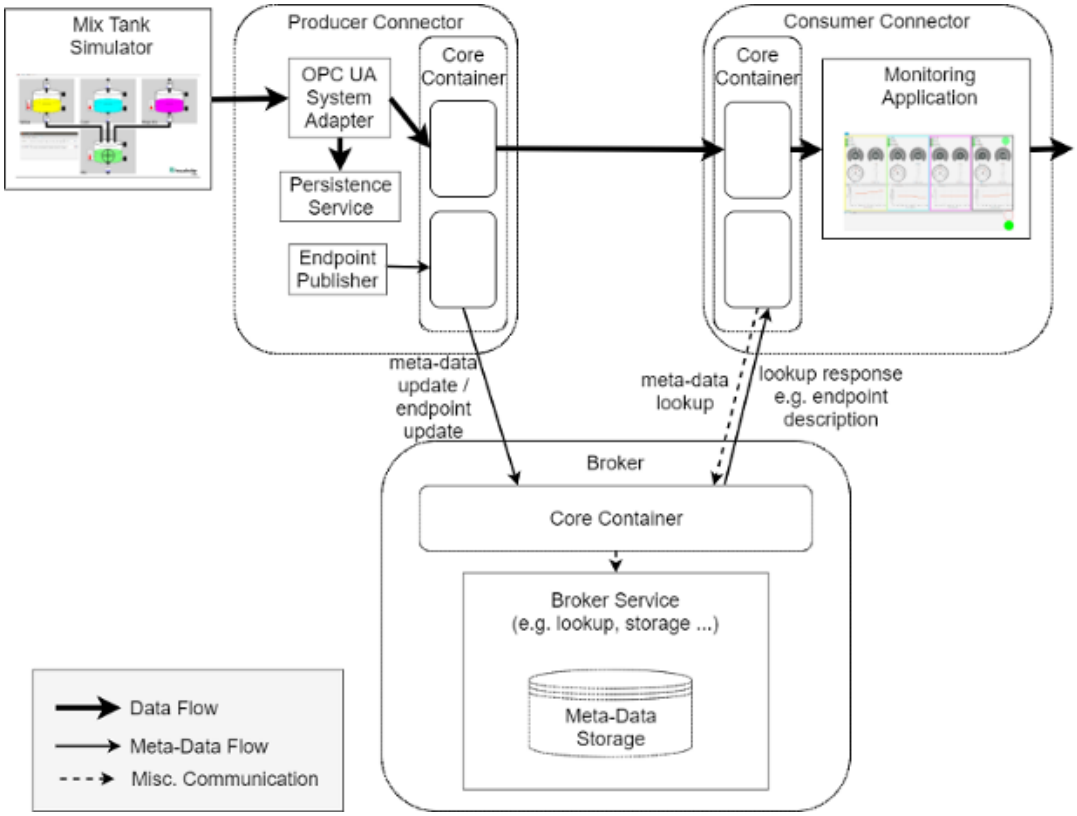


Figure 6.12: Architecture overview on the IDS production use case.

6.3.4 Implementation of an IDS Use Case: Production Data

The aim of the use case is to exchange sensor data between two organizations using the IDS Connector. The IDS Connector is the interface of an organization to the IDS ecosystem. The IDS Broker acts as a metadata hub. It informs and keeps track of all registered IDS Connectors. Among others, this includes detailed metadata about the organizations, data format, transport protocols, data usage rights and contract conditions. Furthermore, this metadata includes a machine-interpretable description in RDF of the operations, bindings and parameters, including the expected results.

Figure 6.12 provides an overview of the data and metadata flows in the IDS ecosystem on the example of the production use case: First, the organization acting as *Data Producer* registers its Connector and metadata at the *Broker*. Second, the organization acting as *Data Consumer* looks up suitable data offerings at the Broker via its Connector using SPARQL queries. Once found, the Data Consumer (semi-)automatically signs a contract with the Data Producer and establishes a connection between their Connectors. The data flow works as follows in this use case: The OPC UA sensor data is retrieved from the source system *Mix Tank Simulator* into the Producer Connector. From there, the data can be retrieved by the Consumer Connector and loaded into the *Monitoring Application* for analytical and decision making purposes.

By having all necessary information as metadata in the Broker, participants in the IDS can quickly judge whether the contract conditions fit their business strategy, whether a connection to a particular interface protocol is easily set up and whether the structure of the data can be easily injected into their core systems.

```
# Protocol Binding
opcua:sensorHttpBinding
    a idsv:HttpBinding ;
    idsv:location <http://opcua-ids-connector:8080/> ;
    idsv:operationBinding opcua:sensorHttpBindingGet.


# GET Operation
opcua:sensorHttpBindingGet
    a idsv:OperationProtocolBinding ;
    idsv:method http:Get;
    idsv:path "sensors/{sensorId}" ;
    idsv:operation opcua:getSensorDesc ;
    idsv:parameterBinding opcua:sensorIdInParamBindingHttp ;
    idsv:parameterBinding opcua:sensorDescOutParamBindingHttp .

# Input Parameter
opcua:sensorIdInParamBindingHttp
    a idsv:ParameterProtocolBinding ;
    idsv:operationParameter opcua:sensorIdParameter  ;
    idsv:protocolParameter opcua:sensorIdUrl ;
    .

# Description of the parameter
opcua:sensorIdUrl
    a http:URITemplate ;
    idsv:host "opcua-ids-connector" ;
    idsv:port "8080" ;
    idsv:path "sensors/{sensorId}" ;
    # query details if they exist
    .

# Return Parameter
opcua:sensorDescOutParamBindingHttp
    a idsv:ParameterProtocolBinding ;
    idsv:operationParameter opcua:sensorDescParameter  ;
    idsv:protocolParameter [
        a http:EntityBody
    ] .
```

Listing 6.8: Description of an IDS connector interface compliant to the IDS Information Model.

Figure 6.13 depicts an IDS App which provides a user interface to monitor the simulated live tank data (temperature, flow). The data is exchange between the two IDS connectors and highlights on the consumer side if certain thresholds in the tank data is reacher.

6.3.5 IDS Domain Vocabulary for Machine Components

In the context of shop floor equipment and software applications, MTConnect [97] is a recently proposed standard to provide the basis for data exchange between different shop floor equipment and software applications. MTConnect defines specific data patterns to facilitate healthcare monitoring of machine tools. Thus, it provides the basis for predictive maintenance to reduce the possibly premature exchange of expensive machine parts or to prevent entire machine outages due to broken parts based on sensor data. The MTConnect standard is published as a document[33] (PDF) and consists of an implementation in a machine-readable format (XML). This section describes the implementation of MTConnect as machine-interpretable ontology (OWL) to be used in the IDS. Furthermore, we defined multiple machine healthcare evaluation queries (SPARQL) to lay the groundwork for standardized machine evaluation

[33] http://www.mtconnect.org/standard-documents

Figure 6.13: User interface for monitoring the simulated live tank data (temperature, flow).

methods. The ontology is tested using a qualitative evaluation by checking its semantic correctness, by creating sample data and by running multiple sample queries.

Implementation

The MTConnect ontology is developed manually based on the provided documentation of the MTConnect standard as well as existing data published conforming to MTConnect. The ontology is expressed in the Turtle syntax due to its compact form.

Methodology The building of our ontology follows the top-down approach [98] as well as the one proposed by Uschold et al. [70], which contains the following steps:

- **Scope.** The developed ontology fully conforms to the MTConnect standard (Version 1.3.1). No further concepts are added or missing, which are not part of the standard.
- **Capture of the Ontology.** The MTConnect official specification[34] [97] and the acquired XML data[35] are the sources on which the developed ontology is based.
- **Reuse of Existing Ontologies.** Concepts and properties from the Semantic Sensor Network Ontology (*ssn*)[36], the Smart Appliances REFerence (*saref*) ontology [37], the Simple Event Model Ontology (*sem*)[38] as well as the recommended best practises on *Simple part-whole relations*[39].

[34] http://www.mtconnect.org/standard-documents/
[35] https://smstestbed.nist.gov/vds/
[36] https://www.w3.org/2005/Incubator/ssn/ssnx/ssn
[37] https://w3id.org/saref#
[38] http://semanticweb.cs.vu.nl/2009/11/sem/
[39] https://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/

- **Ontology Documentation** The ontology is available on GitHub[40] together with a human-readable presentation (HTML) of the ontology.

Ontology Development We started the development of the ontology based on the core concept of the standard: *Device*. A device is defined as a single piece of equipment e.g. a machine or any set of elements that operate together to perform a function. The different parts that form a device are described by using the concept *Component* (Listing 6.9) which contained the subclasses *Axe*, *Sensor*, *Controller*, *System*, *Actuator*, *Stock*, *Interface* and *Door*. Furthermore, each of those subclasses contain more subclasses. For example, the class *Axe* consists of the subclasses *LinearAxe* and *RotaryAxe*, which are types of axes. The class *Path* is the subclass of the *Controller*. This hierarchical structure is preserved in our ontology.

```
1  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
2  @prefix  owl: <http://www.w3.org/2002/07/owl#> .
3  @prefix  rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
4  @prefix  xsd: <http://www.w3.org/2001/XMLSchema#> .
5  @prefix  mto: <https://raw.githubusercontent.com/vocol/MTConnect/master/MTConnect.ttl> .
6
7  mto:Component
8      rdf:type     owl:Class;
9      rdfs:label   "Component"^^xsd:string;
10     rdfs:comment "Defines the structure of the physical or
11     logical parts of a device that provide more precise
12     definition for the structure of the device."@en.
```

Listing 6.9: Example class definition in the *MTConnect ontology*

The measurement values of the sensors are defined by MTConnect using so called *data items*. Therefore, we defined the class *Dataitem* which contains subclasses such as *Temperature*, *Availability*, *Condition*, *Angle* and *Pressure* to represent these numeric or non-numeric values that are related to the machine status. The object property `ssn:hasPart` is used to connect the different device parts and `ssn:hasSubSystem` to define the respective sub components. The datatype property `saref:has_Name` is used to define the device name.

The datatype property *sem:hasTimeStamp* defines the exact date and time that an event occurred and the *part:partOf* describes the relationship between different parts of a device.

Figure 6.14 provides an overview of the MTConnect ontology main concepts.

Evaluation

We evaluated the ontology using a qualitative evaluation. We tested the semantic correctness of the ontology, we created sample instances based on live data to ensure its completeness and finally, we developed and tested multiple SPARQL queries to proof that the ontology is able to return the expected results in certain machine health and structural queries. While executing the queries, no performance issues were observed.

Semantic validation We used the *OOPS!* (OntOlogy Pitfall Scanner!)[41] for inspecting gaps in the context of the created ontology. As a result, no errors or warnings were raised in this test.

[40] `https://github.com/vocol/MTConnect`
[41] `http://oops.linkeddata.es/`

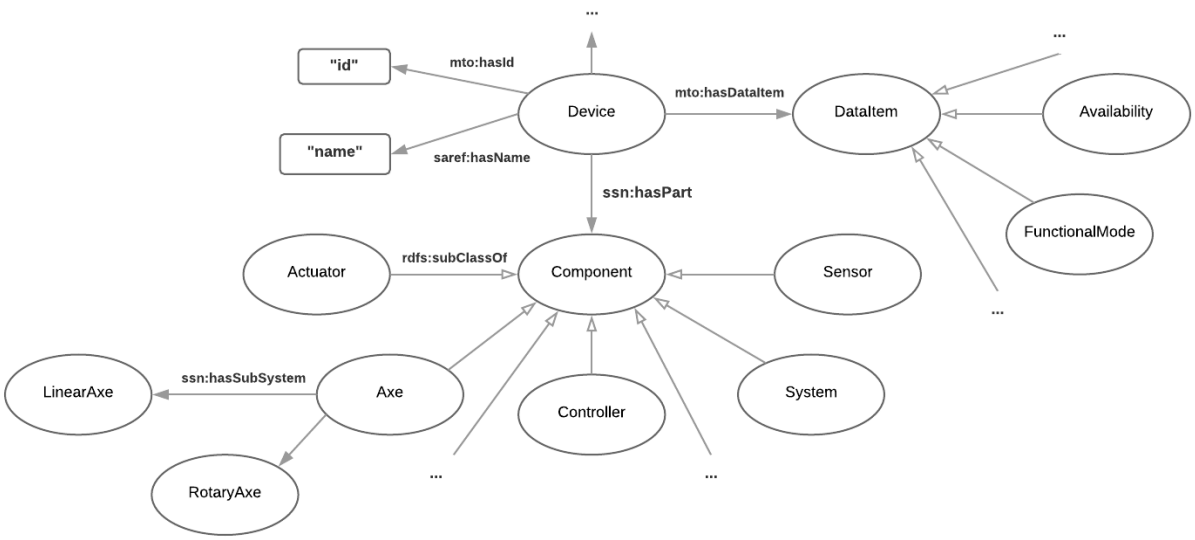


Figure 6.14: Visual of the core machine part concepts in the *MTConnect* ontology.

Sample Instances We further created sample instances based on live steam data published by National Institute of Standards and Technology (NIST)[42] to the validate the completeness our ontology. The format of the live stream data is XML and available via the Smart Manufacturing Systems endpoint[43]). Due to the poor experience with automated methods of transforming XML schema to an OWL ontology, we decided to create a few sample instances manually.

```
mto:Agie01
  a ssn:Device, owl:NamedIndividual;
  rdfs:label "Agie01";
  saref:hasName "GFAgie01";
  saref:hasDescription  "Agie Mikron HPM600U - GF Agie Charmilles HPM600U";
  saref:hasManufacturer "Agie Charmilles";
  saref:hasModel "HPM600U";
  mto:hasCreationtime "2018-02-15T13:05:28Z"^^xsd:dateTime;
  mto:hasUuid "mtc_adapter001";
  rdfs:isDefinedBy mto: .
```

Listing 6.10: Example device instance data based on the *MTConnect* ontology.

As a result, six samples corresponding to different machines are created based on our developed MTConnect ontology. Listing 6.10 depicts an example class instance expressed in Turtle.

SPARQL queries In order to check the health status of the machines and their responses, we examined the condition values that are introduced by the MTConnect documentation. The provided minimum and maximum danger values indicate the operational status of the machines and notify for possible abnormal machine behaviors. Listing 6.11 displays a query to retrieve all the devices which have a *WARNING* condition and returns the condition message.

```
SELECT ?device ?message
```

[42] https://www.nist.gov/
[43] https://smstestbed.nist.gov/vds/

```
2  WHERE {  ?device     ssn:hasPart             ?component.
3           ?component  mto:hasDataItem         ?dataitem.
4           ?dataitem   mto:hasCondition        "WARNING";
5                       sem:hasTimeStamp        ?timestamp;
6                       mto:hasConditionMessage ?message. }
```

Listing 6.11: SPARQL query for measing the device condition

With the use of SPARQL queries it is also possible to observe the structural design of a machine. For example, Listing 6.12 shows all the components and subcomponents that are part of a device.

```
1  SELECT *
2  WHERE { ?device    ssn:hasPart      ?component.
3          ?component ssn:hasSubSystem ?subcomponent.}
4  ORDER BY ASC(?device)
```

Listing 6.12: SPARQL query for retrieving the structural design of a device.

We created in total 12 SPARQL queries related to the MTConnect ontology which are all available on the GitHub repository. All queries are executed in the GraphDB[44] environment and were executed within 0.5 seconds such that no performance issues were observed.

6.3.6 Concluding Remarks

This section provides an explanation of the IDS vision, the IDS architecture, and describes in detail multiple contributions to realize the IDS. In total, the IDS has gained a lot of traction as it can be observed in the rise of members participating in the IDS research project. As of June 2019, various components (IDS Connectors, IDS Broker, Information Model) are under active development and the IDS architecture is constantly refined. Furthermore, various verticalization projects are currently ongoing which further adjust the IDS architecture to their own needs. Examples include the *Medical Data Space*[45], the *Materials Data Space*[46] and the *Mobility Data Space*[47].

In general, the IDS seems to raise awareness that data can be a valuable asset on which new business models can be built. More and more organizations see a value in providing high quality meta data such that one can better integrate external data inside their own infrastructure.

[44] https://ontotext.com/products/graphdb/

[45] https://www.medical-data-space.fraunhofer.de/en.html

[46] https://www.fraunhofer-materials-data-space.de/

[47] https://www.bmvi.de/SharedDocs/DE/Artikel/DG/mfund-projekte/vorstudie-verknuepfung-des-mdm-mit-mds-mdmd-mds.html

CHAPTER 7

Conclusion

In this thesis, we study how semantic technologies can be used to better describe and integrate data for managing Supply Chains and production lines in a factory. We motivated and defined our research questions in Chapter 1, provided the necessary context of the domain in Chapter 2 and presented and discussed the related work in Chapter 3.

This chapter reflects the research questions based on the three contribution chapters: Chapter 4, Chapter 5 and Chapter 6. Finally, we provide closing remarks on the entire thesis and explore opportunities for future work.

7.1 Research Question Analysis

In the following, we revisit the three research questions defined in Section 1.2 and reflect on our results:

> **RQ1:** How can the Linked Data paradigm be used to represent industrial data to improve Supply Chain Management?

In Chapter 4, we described the formalization of *SCORVoc*, a vocabulary based on the ≈1000 pages *SCOR* standard document, which, according the standardization organization *APICS*, claims to be the "most widely accepted framework for evaluating and comparing supply chain activities and performance"[1]. Besides defining the concepts and their properties, we formalized standard SPARQL queries for the KPIs defined by SCOR. These SPARQL queries filled a gap in academia since they had been, to the best of our knowledge, not formalized before. SCORVoc itself fills another gap, since it contains all processes of the standard, including their metrics, which, again, to the best of our knowledge, were beyond the scope of the related work. The vocabulary is further published on the version control hosting platform GitHub[2] and mirrors the 11th version (2012) of the standard, while other attempts reflected the older version, those, missing new processes, metrics and KPIs. In a recent research project with a manufacturing enterprise, we managed to quickly update SCORVoc with the recently introduced 12th (2017) version of SCOR. This shows that the vocabulary can form the basis for organizations, which have the following aim: (1) Conform their own Supply Chain processes to a global standard and thus (2) allow for the execution of standardized KPIs, such that the KPI results of other organizations along the Supply Chain become comparable. Thus, SCORVoc allows organizations to identify strong and weak processes and as a result enables them to collaborative improve the inter-organizational Supply Chain.

[1] http://www.apics.org/about/overview/about-apics-scc
[2] https://github.com/vocol/scor

RQ2: How can we support domain experts contributing to the engineering of industrial ontologies?

Chapter 5 describes TurtleEditor, an application to support the development of ontologies for industrial ontologies. Various methodologies and tools have been proposed by academia to support building ontologies, for teams and individuals. However, the proposed methods and tools still lack features we consider critical for the successful execution of industrial ontology engineering projects. Based on research projects that cover a broad scope of industrial production and logistics scenarios, we collected requirements to enable domain experts to contribute to the development of ontologies. The central requirement identified is to support contributions via web browsers without the need of installing a particular software. Further requirements including offering various views (graph visualization, source-code) based on the experience of the domain experts and providing guidance while contributing, such as syntax highlighting and validation or auto-completion. Our users feedback and experiments show that TurtleEditor helped domain experts to contribute to the development of industrial ontologies, compared to the feature-rich but considered overwhelming alternative ontology editor software by domain experts.

RQ3: How can we leverage semantics in industrial use cases?

Chapter 6 provides multiple use cases for the application of semantic technologies in the industry. One section focuses on its application within an organization, one within a factory, and one within the *International Data Spaces* initiative. Overall, ontologies were especially useful to capture a domain, where knowledge was widespread and partly implicit. In particular, the description of DB schemas with ontologies, to match identical concepts or properties across databases, seems to be a prime candidate for leveraging semantics in the industry. Furthermore, by being able to provide context by linking to documents, web pages or people who hold knowledge, applications were able to provide a view on data which was before hidden across various data silos. The IDS use case describes where semantic technologies can be leveraged to allow the search of datasets, to describe their structure and interpretation and how this particular data can be accesses. In general, we consider this chapter adds valuable insights on various use cases to the scare corpus of publications reporting on semantics being applied in industrial use cases. It seems that by using semantic technologies, a project team is forced by the nature of the technology to thoroughly discuss the knowledge about the domain, the data which exist, where it is, who knows what etc., which seems to be an advantage already by itself.

7.2 Closing Remarks and Future Work

While this thesis presents multiple contributions in applying semantic technologies in industrial use cases, our experience in working with various organizations is that it is nevertheless rather difficult for enterprises to use them. We present three possible future scientific directions which could facilitate the adoption of semantic technologies for organizations outside of academia:

1. **Improving the toolset around semantic technologies for enterprises**
   Our research suggests that for organizations outside of academia, deploying semantic technologies can be quite hard. The existing toolset seems to be sparse, especially for organizations who prefer commercial tools over available open source tools due their need for technical support. Tools for managing ontologies or linked data are in our experience even for individuals who have a Computer Science degree difficult to use. We believe that the lack of more user-friendly semantic web tools is the biggest bottleneck for the adoption of semantic technologies in the industry. More research

into the requirements of enterprises for tools may help in reducing the perceived complexity of semantic technologies and increase their adoption in industrial use cases.

2. **Methods to better describe and deploy ontologies**
   To maximize the usage of ontologies, it is critical to describe them in a way that individuals outside of the own domain have a chance to understand them. Currently, it can be quite an investment of time to fully grasp the scope of other ontologies. The description should be honest about the scope of the ontology and not contain too generic or ambiguous terms. Ontologies often lack context about themselves: Why was it developed? Which standard does it conform to? Where was the ontology developed? Where is it used? Which are the core classes? Which parts of a domain does it not cover? It would be valuable to research what can help a visitor of an ontology web page to quickly grasp if the ontology is useful or not to them.

3. **Methods for open, technology-oriented business standardization**
   Many standardization activities result in the creation of documents instead of executable models. By focusing on documents, there is the chance of a faulty technical implementation or no adoption at all, since it can become too costly. The open source tools developed in the Semantic Web community to generate a human-friendly version (web pages) for ontologies might be useful for standardization activities to develop the model and the document hand in hand. Furthermore, by publishing standards online on platforms such as GitHub, one opens the door for possible valuable feedback by the open source community.

Bibliography

[1] A. Gandomi and M. Haider, *Beyond the hype: Big data concepts, methods, and analytics*, International Journal of Information Management **35**.2 (2015) 137 (cit. on p. 1).

[2] S. Lee, *A.40 Computational Modeling Algorithms and Cyberinfrastructure*, tech. rep., Jet Propulsion Laboratory, National Aeronautics and Space Administration (NASA), 2011 (cit. on p. 2).

[3] T. Berners-Lee, J. Hendler and O. Lassila, *The semantic web*, Scientific american **284**.5 (2001) 34 (cit. on pp. 2, 16).

[4] C. Bizer, T. Heath and T. Berners-Lee, "Linked data: The story so far", *Semantic services, interoperability and web applications: emerging concepts*, IGI Global, 2011 205 (cit. on p. 2).

[5] H. Kagermann, J. Helbig, A. Hellinger and W. Wahlster, *Recommendations for implementing the strategic initiative INDUSTRIE 4.0: Securing the future of German manufacturing industry; final report of the Industrie 4.0 Working Group*, Forschungsunion, 2013 (cit. on p. 2).

[6] W. Bauer and P. Horváth, *Industrie 4.0-Volkswirtschaftliches Potenzial für Deutschland*, Controlling **27**.8-9 (2015) 515 (cit. on p. 2).

[7] Supply Chain Council, *Supply chain operations reference model (SCOR)*, version 11, 2012 (cit. on pp. 4, 10, 11, 32, 44).

[8] R. R. Lummus and R. J. Vokurka, *Defining supply chain management: a historical perspective and practical guidelines*, Industrial Management & Data Systems **99**.1 (1999) 11 (cit. on p. 9).

[9] F. Galasso, Y. Ducq, M. Lauras, D. Gourc and M. Camara, *A method to select a successful interoperability solution through a simulation approach*, Journal of Intelligent Manufacturing (2014) 1 (cit. on p. 9).

[10] B. Otto and M. Ofner, "Towards a Process Reference Model for Information Supply Chain Management.", *ECIS*, 2010 75 (cit. on p. 9).

[11] M. Brettel, N. Friederichsen, M. Keller and M. Rosenberg, *How virtualization, decentralization and network building change the manufacturing landscape: An Industry 4.0 Perspective*, International Journal of Science, Engineering and Technology 8 (1), 37 **44** (2014) (cit. on p. 9).

[12] M. Hermann, T. Pentek and B. Otto, "Design Principles for Industrie 4.0 Scenarios", *49th Hawaii Int. Conf. on System Sciences (HICSS)*, IEEE, 2016 3928 (cit. on p. 10).

[13] S.-W. Lin, B. Miller, J. Durand, R. Joshi, P. Didier, A. Chigani, R. Torenbeek, D. Duggal, R. Martin, G. Bleakley et al., *Industrial internet reference architecture*, Industrial Internet Consortium (IIC), Tech. Rep (2015) (cit. on p. 13).

[14] S. Lin, B. Miller, J. Durand, G. Bleakley, A. Chigani, R. Martin, B. Murphy and M. Crawford, *The industrial internet of things volume G1: reference architecture*, Industrial Internet Consortium (2017) 10 (cit. on p. 13).

[15] B. Otto, S. Steinbuß, A. Teuscher and e. Lohmann Steffen, *IDS Reference Architecture Model, Version 3.0*, (2019) 118 (cit. on pp. 13, 14, 80, 81).

[16] I. V. C. Initiative, *Connected Industries Open Framework for Manufacturing - Basic Specification of System Requirement*, (2019) 70 (cit. on pp. 14, 15).

[17] A. of Industrial Internet, *Industrial Internet Architecture*, (2016) 28 (cit. on pp. 14, 15).

[18] G. Klyne and J. J. Carroll, *Resource description framework (RDF): Concepts and abstract syntax*, (2006) (cit. on p. 17).

[19] D. Brickley, R. V. Guha and B. McBride, *RDF Schema 1.1*, W3C recommendation **25** (2014) 2004 (cit. on p. 17).

[20] D. L. McGuinness, F. Van Harmelen et al., *OWL web ontology language overview*, W3C recommendation **10**.10 (2004) 2004 (cit. on p. 17).

[21] M. Dürst and M. Suignard, *Internationalized resource identifiers (IRIs)*, tech. rep., RFC 3987, January, 2005 (cit. on p. 17).

[22] D. Beckett, T. Berners-Lee, E. Prud'hommeaux and G. Carothers, *RDF 1.1 Turtle*, World Wide Web Consortium (2014) (cit. on pp. 18, 31).

[23] T. R. Gruber, *A translation approach to portable ontology specifications*, Knowledge acquisition **5**.2 (1993) 199 (cit. on p. 20).

[24] Y. Ye, D. Yang, Z. Jiang and L. Tong, *An ontology-based architecture for implementing semantic integration of supply chain management*, International Journal of Computer Integrated Manufacturing **21**.1 (2008) 1 (cit. on p. 23).

[25] M. Fayez, L. Rabelo and M. Mollaghasemi, "Ontologies for supply chain simulation modeling", *37th conference on Winter simulation*, 2005 (cit. on p. 23).

[26] J. Leukel and S. Kirn, "A supply chain management approach to logistics ontologies in information systems", *Business Information Systems*, Springer, 2008 (cit. on p. 23).

[27] O. Sakka, P.-A. Millet and V. Botta-Genoulaz, *An ontological approach for strategic alignment: a supply chain operations reference case study*, International Journal of Computer Integrated Manufacturing **24**.March 2015 (2011), ISSN: 0951-192X (cit. on pp. 23, 24).

[28] M. Zdravković, H. Panetto, M. Trajanović and A. Aubry, *An approach for formalising the supply chain operations*, Enterprise Information Systems **5**.4 (2011) (cit. on pp. 23, 24).

[29] Y. Lu, H. Panetto, Y. Ni and X. Gu, *Ontology alignment for networked enterprise information system interoperability in supply chain environment*, International Journal of Computer Integrated Manufacturing **26**.1-2 (2013) 140 (cit. on pp. 23, 24).

[30] M. Zdravković, M. Trajanović, H. Panetto, A. Aubry and M. Lezoche, "Ontology-based supply chain process configuration", *34th International Conference on Production Engineering, ICPE 2011*, 2011 (cit. on p. 24).

[31] 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 (2012) (cit. on p. 24).

[32] T. Grubic and I. S. Fan, *Integrating process and ontology for supply chain modelling*, International Journal of Computer Integrated Manufacturing **24**.March 2015 (2011) 847, ISSN: 0951-192X (cit. on p. 24).

[33] R. Palma, O. Corcho, A. Gómez-Pérez and P. Haase, *A holistic approach to collaborative ontology development based on change management*, Journal of Web Semantics **9**.3 (2011) (cit. on p. 24).

[34] T. Tudorache, J. Vendetti and N. F. Noy, "Web-Protege: A Lightweight OWL Ontology Editor for the Web.", *OWLED*, vol. 432, Citeseer, 2008 (cit. on p. 24).

[35] T. Tudorache, N. Noy, J. Vendetti and T. Redmond, *Collaborative ontology development with protege*, tech. rep., Technical report, Standford University, Tech. Rep, 2007 (cit. on p. 25).

[36] A. Stellato, S. Rajbhandari, A. Turbati, M. Fiorelli, C. Caracciolo, T. Lorenzetti, J. Keizer and M. T. Pazienza, "VocBench: A Web Application for Collaborative Development of Multilingual Thesauri", *ESWC '15*, Springer, 2015 38, URL: http://dblp.uni-trier.de/db/conf/esws/eswc2015.html#StellatoRTFCLKP15 (cit. on p. 25).

[37] V. Zacharias and S. Braun, "SOBOLEO – Social Bookmarking and Lighweight Engineering of Ontologies", *CKC Workshop at WWW '07*, 2007, URL: http://www2007.org/workshops/paper_41.pdf (cit. on p. 25).

[38] T. Schandl and A. Blumauer, "PoolParty: SKOS thesaurus management utilizing linked data", *ESWC '10*, Springer, 2010 421 (cit. on p. 25).

[39] C. Ghidini, M. Rospocher and L. Serafini, "MoKi: a Wiki-Based Conceptual Modeling Tool.", *ISWC '10 Posters and Demos*, CEUR-WS:658, 2010, URL: http://dblp.uni-trier.de/db/conf/semweb/pd2010.html#GhidiniRS10 (cit. on p. 25).

[40] S. Auer, S. Dietzold and T. Riechert, "OntoWiki–a tool for social, semantic collaboration", *International Semantic Web Conference*, Springer, 2006 736 (cit. on p. 25).

[41] G. Fu and W. Rao, *An introduction of Knoodl*, 2015, URL: http://tinman.cs.gsu.edu/~raj/8711/sp11/presentations/knoodlReport.pdf (cit. on p. 25).

[42] L. Halilaj, I. Grangel-González, G. Coskun, S. Lohmann and S. Auer, *Git4Voc: Collaborative Vocabulary Development Based on Git*, Int. J. Semantic Computing **10**.2 (2016) 167 (cit. on p. 25).

[43] R. Falco, A. Gangemi, S. Peroni, D. Shotton and F. Vitali, "Modelling OWL Ontologies with Graffoo", *ESWC 2014 Satellite Events*, Springer, 2014 320 (cit. on p. 25).

[44] B. Otto, S. Lohmann, S. Auer, G. Brost, J. Cirullies, A. Eitel, T. Ernst, C. an Haas, M. Huber, C. Jung, J. Jürjens, C. Lange, C. Mader, N. Menz, R. Nagel, H. Pettenpohl, J. Pullmann, C. Quix, J. Schon, D. Schulz, J. Schütte, M. Spiekermann and S. Wenzel, *Reference Architecture Model for the Industrial Data Space*, tech. rep., Fraunhofer, 2017, URL: http://www.industrialdataspace.org/publications/industrial-data-space-reference-architecture-model-2017/ (cit. on p. 26).

[45] M. Dumontier, C. Wu, A. Zaveri, K. Jagodnik, R. Verborgh, R. Terryn, P. Avillach and G. Korodi, *smartAPI Specification*, URL: https://websmartapi.github.io/smartapi_specification/ (cit. on p. 27).

[46] R. Chinnici, J.-J. Moreau, A. Ryman and S. Weerawarana, *Web services description language (wsdl) version 2.0 part 1: Core language*, W3C recommendation **26**.1 (2007) 19 (cit. on p. 27).

[47] The Open University. Knowledge Media Institute, *iServe's Data Model*, 2015, URL: http://kmi.github.io/iserve/latest/data-model.html (cit. on p. 27).

[48] F. Maali, J. Erickson and P. Archer, *Data Catalog Vocabulary (DCAT)*, URL: https://www.w3.org/TR/vocab-dcat/ (cit. on p. 27).

[49] W. Terkaj and M. Urgo, "Virtual factory data model to support performance evaluation of production systems", *Proceedings of the Workshop on Ontology and Semantic Web for Manufacturing (OSEMA '12)*, vol. 886, CEUR-WS, 2012 24 (cit. on p. 27).

[50] R.-S. Chen and M. A. Tu, *Development of an agent-based system for manufacturing control and coordination with ontology and RFID technology*, Expert systems with applications **36**.4 (2009) 7581 (cit. on p. 27).

[51] R. Want, *An introduction to RFID technology*, IEEE Pervasive Computing **5**.1 (2006) 25 (cit. on p. 27).

[52] C. Büscher, H. Voet, M. Krunke, P. Burggräf, T. Meisen and S. Jeschke, *Semantic information modelling for factory planning projects*, Procedia CIRP **41** (2016) 478 (cit. on pp. 27, 28).

[53] K.-Y. Kim, D. G. Manley and H. Yang, *Ontology-based assembly design and information sharing for collaborative product development*, Computer-Aided Design **38**.12 (2006) 1233 (cit. on p. 28).

[54] F. Ameri and L. Patil, *Digital manufacturing market: a semantic web-based framework for agile supply chain deployment*, Journal of Intelligent Manufacturing **23**.5 (2012) 1817 (cit. on p. 28).

[55] D. Zuehlke, *SmartFactory—Towards a factory-of-things*, Annual Reviews in Control **34**.1 (2010) 129 (cit. on p. 28).

[56] E. Kharlamov, S. Brandt, E. Jimenez-Ruiz, Y. Kotidis, S. Lamparter, T. Mailis, C. Neuenstadt, Ö. Özçep, C. Pinkel, C. Svingos et al., "Ontology-based integration of streaming and static relational data with optique", *Proceedings of the 2016 International Conference on Management of Data*, ACM, 2016 2109 (cit. on p. 28).

[57] E. Kharlamov, T. Mailis, G. Mehdi, C. Neuenstadt, Ö. Özçep, M. Roshchin, N. Solomakhina, A. Soylu, C. Svingos, S. Brandt et al., *Semantic access to streaming and static data at Siemens*, Web Semantics (2017) (cit. on p. 28).

[58] E. Kharlamov, D. Hovland, E. Jiménez-Ruiz, D. Lanti, H. Lie, C. Pinkel, M. Rezk, M. G. Skjæveland, E. Thorstensen, G. Xiao et al., "Ontology based access to exploration data at Statoil", *ISWC*, 2015 (cit. on p. 28).

[59] N. Rychtyckyj, V. Raman, B. Sankaranarayanan, P. S. Kumar and D. Khemani, "Ontology re-engineering: a case study from the automotive industry", *AAAI*, 2016 (cit. on p. 28).

[60] D. Ostrowski, N. Rychtyckyj, P. MacNeille and M. Kim, "Integration of big data using semantic web technologies", *ICSC*, 2016 (cit. on p. 28).

[61] M. Kim, S.-T. Wang, D. Ostrowski, N. Rychtyckyj and P. Macneille, *Technology Outlook: Federated Ontologies and Industrial Applications*, Semantic Computing **10**.01 (2016) (cit. on p. 28).

[62] W. Greenly, C. Sandeman-Craik, Y. Otero and J. Streit, *Case Study: Contextual Search for Volkswagen and the Automotive Industry*, 2011, URL: https://www.w3.org/2001/sw/sweo/public/UseCases/Volkswagen/ (cit. on p. 28).

[63] J. Golebiowska, R. Dieng-Kuntz, O. Corby and D. Mousseau, "Building and exploiting ontologies for an automobile project memory", *K-CAP*, 2001 (cit. on p. 28).

[64] M. Rodriguez-Muro, R. Kontchakov and M. Zakharyaschev, "Ontology-based data access: Ontop of databases", *ISWC*, 2013 (cit. on p. 28).

[65] N. Petersen, I. Grangel-González, G. Coskun, S. Auer, M. Frommhold, S. Tramp, M. Lefrançois and A. Zimmermann, "SCORVoc: Vocabulary-Based Information Integration and Exchange in Supply Networks", *Proceedings of the 10th International Conference on Semantic Computing; Laguna Hills, California; February 3-5 2016*, IEEE, 2016 132 (cit. on pp. 31, 42, 79).

[66] S. Tramp, R. N. Piris, T. Ermilov, N. Petersen, M. Frommhold and S. Auer, "Distributed Linked Data Business Communication Networks: The LUCID Endpoint", *The Semantic Web: ESWC 2015 Satellite Events*, Springer, 2015 154 (cit. on p. 31).

[67] M. Frommhold, N. Arndt, S. Tramp and N. Petersen, "Publish and Subscribe for RDF in Enterprise Value Networks", *WWW2016 Workshop: Linked Data on the Web (LDOW2016)*, Springer, 2016 (cit. on p. 31).

[68] N. Petersen, M. Galkin, C. Lange, S. Lohmann and S. Auer, "Monitoring and Automating Factories Using Semantic Models", *JIST*, 2016 (cit. on pp. 31, 57, 58, 70).

[69] L. Halilaj, N. Petersen, I. Grangel-González, C. Lange, S. Auer, G. Coskun and S. Lohmann, "VoCol: An Integrated Environment to Support Version-Controlled Vocabulary Development", *Knowledge Engineering and Knowledge Management - 20th International Conference, EKAW 2016, Bologna, Italy, November 19-23, 2016, Proceedings*, 2016 303, URL: http://dx.doi.org/10.1007/978-3-319-49004-5_20 (cit. on pp. 31, 47, 56, 59).

[70] M. Uschold and M. Gruninger, *Ontologies: Principles, methods and applications*, Knowledge engineering review **11**.2 (1996) 93 (cit. on pp. 31, 58, 72, 87).

[71] S. Harris, A. Seaborne and E. Prud'hommeaux, *SPARQL 1.1 query language*, W3C Recommendation **21** (2013) (cit. on p. 34).

[72] H. L. Lee, V. Padmanabhan and S. Whang, *Information distortion in a supply chain: the bullwhip effect*, Management science **50**.12_supplement (2004) 1875 (cit. on p. 39).

[73] D. Hardt, *The OAuth 2.0 authorization framework*, (2012) (cit. on p. 42).

[74] B. Fitzpatrick, B. Slatkin, M. Atkins and J. Genestoux, *PubSubHubbub Core 0.4. Working draft, PubSubHubbub W3C Community Group (2013)*, tech. rep. (cit. on p. 43).

[75] J. Unbehauen, M. Frommhold and M. Martin, "Enforcing scalable authorization on SPARQL queries", *SEMANTiCS*, 2016 (cit. on p. 43).

[76] M. Hermann, T. Pentek and B. Otto, *Design principles for Industrie 4.0 scenarios: a literature review*, Technische Universität Dortmund, Dortmund (2015) (cit. on p. 45).

[77] N. Petersen, G. Coskun and C. Lange, "TurtleEditor: An Ontology-Aware Web-Editor for Collaborative Ontology Development", *2016 IEEE Tenth International Conference on Semantic Computing (ICSC)*, 10th International Conference on Semantic Computing, ICSC (Laguna Hills, CA, USA, 3rd–5th Feb. 2016), ed. by T. Li, A. Scherp, D. Ostrowski and W. Wang, 2016 183, URL: http://dx.doi.org/10.5281/zenodo.35499 (cit. on p. 47).

[78] A. Similea, N. Petersen, C. Lange and S. Lohmann, "TurtleEditor 2.0: A Synchronized Graphical and Text Editor for RDF Vocabularies", *11th IEEE International Conference on Semantic Computing, ICSC 2017, San Diego, CA, USA, January 30 - February 1, 2017*, 11th International Conference on Semantic Computing, ICSC (San Diego, CA, USA, 30th Jan.–1st Feb. 2017), ed. by D. D'Auria, J. Liu and G. Pilato, IEEE Computer Society, 2017 286 (cit. on p. 47).

[79] N. Petersen, A. Similea, C. Lange and S. Lohmann, *TurtleEditor: A Web-based RDF Editor to Support Distributed Ontology Development on Repository Hosting Platforms*, International Journal on Semantic Computing **11**.3 (2017): *Special Issue – Best Papers from the Workshops Co-Located with the 11th IEEE International Conference on Semantic Computing (ICSC 2017)* 311, ed. by S. Bansal, C. Chaves Gattaz, R. Mertens, F. Persia, G. Pilato and G. Zhang, ISSN: 1793-351X (cit. on p. 47).

[80] D. Beckett, T. Berners-Lee, E. Prud'hommeaux and G. Carothers, *RDF 1.1 Turtle, Terse RDF Triple Language*, W3C Recommendation, World Wide Web Consortium (W3C), 2014, URL: http://www.w3.org/TR/2014/REC-turtle-20140225/ (cit. on p. 47).

[81] B. Leuf and W. Cunningham, *The Wiki Way: Collaboration and Sharing on the Internet*, Addison-Wesley Professional, 2001, ISBN: 0-201-71499-X (cit. on p. 48).

[82] N. Petersen, L. Halilaj, I. Grangel-González, S. Lohmann, C. Lange and S. Auer, "Realizing an RDF-Based Information Model for a Manufacturing Company – A Case Study", *The Semantic Web, Part II*, 16th International Semantic Web Conference, ISWC (Vienna, Austria, 21st–25th Oct. 2017), ed. by C. d'Amato, M. Fernandez, V. Tamma, F. Lecue, P. Cudré-Mauroux, J. Sequeda, C. Lange and J. Heflin, Lecture Notes in Computer Science 10588, Heidelberg: Springer Verlag, 2017 350, ISBN: 978-3-319-68203-7 (cit. on p. 57).

[83] G. Alvanou, I. Lytra and N. Petersen, "An MTConnect Ontology for Semantic Industrial Machine Sensor Analytics", *Proceedings of the Workshop on Semantic Web of Things for Industry 4.0 (SWeTI), In conjunction with the 15th Extended Semantic Web Conference (ESWC 2018)*, 2018 (cit. on pp. 57, 80).

[84] J. Pullmann, C. Mader, N. Petersen and S. Lohmann, "Ontology-based Information Modelling in the Industrial Data Space", *22st IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2017, Limassol, Cyprus, September 12-15, 2017*, IEEE, 2017 (cit. on pp. 57, 80).

[85] L. Halilaj, I. Grangel-González, M.-E. Vidal, S. Lohmann and S. Auer, "Proactive Prevention of False-Positive Conflicts in Distributed Ontology Development", *KEOD*, 2016 (cit. on p. 59).

[86] Adolphs, P., et al., *Reference Architecture Model Industrie 4.0 (RAMI4.0)*, Status Report, ZVEI and VDI, 2015 (cit. on pp. 61, 69).

[87] I. Grangel-González, L. Halilaj, S. Auer, S. Lohmann, C. Lange and D. Collarana, "An RDF-based approach for implementing industry 4.0 components with Administration Shells", *ETFA*, 2016 (cit. on p. 61).

[88] *IEC 62264-1: Enterprise-Control System Integration Part 1: Models and Terminology*, Standard, IEC, 2013 (cit. on pp. 61, 69).

[89] H. Wache, T. Voegele, T. Visser, H. Stuckenschmidt, H. Schuster, G. Neumann and S. Huebner, "Ontology-based integration of information - a survey of existing approaches", *IJCAI-01 Workshop: Ontologies and Information*, 2001 (cit. on p. 62).

[90] L. Bellatreche and G. Pierra, "OntoAPI: An Ontology-based Data Integration Approach by an a Priori Articulation of Ontologies", *DEXA*, 2007 (cit. on p. 63).

[91] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini and R. Rosati, *Linking Data to Ontologies*, J. Data Semantics **10** (2008) 133 (cit. on p. 63).

[92] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro and G. Xiao, *Ontop: Answering SPARQL Queries over Relational Databases*, Semantic Web **8**.3 (2017) 471 (cit. on pp. 63, 70).

[93] J. Nielsen, *Response times: The 3 important limits*, Usability Engineering (1993) (cit. on p. 78).

[94] L. Halilaj, I. Grangel-González, G. Coskun and S. Auer, "Git4Voc: Git-Based Versioning for Collaborative Vocabulary Development", *10th International Conference on Semantic Computing (ICSC '16)*, IEEE, 2016 285 (cit. on p. 79).

[95] N. Petersen, C. Lange, S. Auer, M. Frommhold and S. Tramp, "Towards Federated, Semantics-Based Supply Chain Analytics", *19th International Conference on Business Information Systems (BIS '16)*, Springer, 2016 436 (cit. on p. 79).

[96] J. E. Masters and R. Hodgson, *QUDT Dimensions Vocabulary Version 1.1*, 2010, URL: http://www.linkedmodel.org/doc/qudt-vocab-dimensions/1.1/ (cit. on p. 82).

[97] W. Sobel, *MTConnect standard. Part 1—overview and protocol*, Standard—MTConnect. http://www. mtconnect. org/standard. Accessed **10** (2015) (cit. on pp. 86, 87).

[98] N. F. Noy and D. L. Mcguinness, *Ontology Development 101: A Guide to Creating Your First Ontology*, tech. rep., 2001 (cit. on p. 87).

APPENDIX A

List of Publications

- *Journal Articles*:

  1. **Niklas Petersen**, Alexandra Similea, Christoph Lange, Steffen Lohmann. *TurtleEditor: A Web-based RDF Editor for Supported Distributed Ontology Development on Repository Hosting Platforms* In International Journal on Semantic Computing, Vol. 11, No. 03, 311–323, 2017;

- *Conference Papers*:

  2. **Niklas Petersen**, Lavdim Halilaj, Irlán Grangel-González, Steffen Lohmann, Christoph Lange, Sören Auer. *Realizing an RDF-based Information Model for a Manufacturing Company* In Proceedings of the 16th International Semantic Web Conference (ISWC '17), 350–366, Springer;
  3. **Niklas Petersen**, Irlán Grangel-González, Gökhan Coskun, Sören Auer, Marvin Frommhold, Sebastian Tramp, Maxime Lefrançois, Antoine Zimmermann. *SCORVoc: Vocabulary-Based Information Integration and Exchange in Supply Networks* In Proceedings of the Tenth International Conference on Semantic Computing (ICSC '16), 132–139, IEEE;
  4. **Niklas Petersen**, Christoph Lange, Sören Auer, Marvin Frommhold, Sebastian Tramp. *Towards Federated, Semantic-based Supply Chain Analytics* In Lecture Notes in Business Information Processing (BIS '16), 436–447, Springer;
  5. **Niklas Petersen**, Michael Galkin, Christoph Lange, Steffen Lohmann, Sören Auer. *Monitoring Automating Factories Using Semantic Models* In Lecture Notes in Computer Science (LNCS, volume 10055; Joint International Semantic Web Technology Conference (JIST) '16), 315–330, Springer;
  6. **Niklas Petersen**, Gökhan Coskun, Christoph Lange. *TurtleEditor: An ontology-aware web-editor for collaborative ontology development* In Proceedings of the Tenth International Conference on Semantic Computing (ICSC '16), 183–186, IEEE;
  7. Alexandria Similea, **Niklas Petersen**, Christoph Lange, Steffen Lohmann. *TurtleEditor 2.0: A Synchronized Graphical and Text Editor for RDF Vocabularies* In Proceedings of the 11th International Conference on Semantic Computing (ICSC '17), 286–289, IEEE;
  8. Marvin Frommhold, Rubén Navarro Piris, Natanael Arndt, Sebastian Tramp, **Niklas Petersen**, Michael Martin. *Towards Versioning of Arbitrary RDF Data* In Proceedings of the 12th International Conference on Semantic Systems (SEMANTiCS '16), 33–44, ACM;

9. Lavdim Halilaj, **Niklas Petersen**, Irlán Grangel-González, Christoph Lange, Sören Auer, Gökhan Coskun, Steffen Lohmann. *VoCol: An Integrated Environment to Support Version-Controlled Vocabulary Development* In Knowledge Engineering and Knowledge Management of the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW '16), 303–319, Springer;
10. Jaroslav Pullmann, Christian Mader, **Niklas Petersen**, Steffen Lohmann. *Ontology-based Information Modelling in the Industrial Data Space* In Proceedings of the 22nd International Conference on Emerging Technologies and Factory Automation (ETFA '17), 1–8, IEEE;
11. Sebastian Pretzsch, Holger Drees, Christoph Lange, Christian Mader, **Niklas Petersen**, Lutz Rittershaus. *Mobility Data Space – An Open Decentral Ecosystem for Mobility Data* In 25th ITS World Congress;

- *Workshop Papers*:

12. Sebastian Tramp, Rubén Navarro Piris, Timofey Ermilov, **Niklas Petersen**, Marvin Frommhold, Sören Auer. *Distributed Linked Data Business Communication Networks: The LUCID Endpoint* In Proceedings of The Semantic Web: ESWC 2015 Satellite Events (ESWC '15), 154–158, Springer;
13. Marvin Frommhold, Natanael Arndt, Sebastian Tramp, **Niklas Petersen**. *Publish and Subscribe for RDF in Enterprise Value Networks* In Proceedings of the Workshop on Linked Data on the Web (LDOW '16), Co-located with the 25th International World Wide Web Conference (WWW '16);
14. Glykeria Alvanou, **Niklas Petersen**. *An MTConnect Ontology for Semantic Industrial Machine Sensor Analytics* Submitted to the Workshop on Semantic Web of Things for Industry 4.0 (SWeTI'18), Co-located with the 15th Extended Semantic Web Conference (ESWC '18)

List of Figures

List of Tables