Undefined 1 (2009) 1–5
IOS Press

How Ontologies Benefit Enterprise Applications

Editor(s): Name Surname, University, Country
Solicited review(s): Name Surname, University, Country
Open review(s): Name Surname, University, Country

Daniel Oberle
SAP Research Karlsruhe, Vincenz-Priessnitz-Str. 1, 76131 Karlsruhe, Germany, E-mail: d.oberle@sap.com

Abstract. This paper contributes an argumentation line for how technological features of ontologies lead to benefits for enterprise applications. Although many features are also available in precursory or alternative technologies, we claim that combinations of specific features are uniquely provided by ontologies. A careful elicitation of the available features therefore is a prerequisite for the argumentation line. As a second contribution, this paper reports on several challenges that frequently occur when trying to adopt ontologies in existing enterprise settings. These challenges have to be contrasted with the often overstressed benefits in Semantic Web literature. Together with reports for several SAP Research case studies, this paper channels back experiences to the Semantic Web community. As a third contribution, we give several recommendations for future research directions based on the gathered experiences.

Keywords: Semantics, Semantic Technologies, Ontologies, Reasoning, Enterprise, Industry, Commercial, Application

1. Introduction

The last years have seen great academic interest in the Semantic Web and in semantic technologies and even first industrial products appeared. [4] However, semantic technologies have been criticized for being elusive, especially for use in enterprise applications. [99] Typical questions asked in an enterprise setting are: "*What business problem is semantics solving? What relevance is there between semantics and the real world? What value proposition for the conduct of business is there that is being addressed by semantics?*" [48]

One reason for such sentiments is on the level of terminology. So, the first step in answering such questions is to clarify what we are talking about. As pointed out in [51], the murkiness of the words 'semantics' or 'semantic technologies' adds to the confusion. Therefore, this paper sets its focus on ontologies [40], i.e., a specific semantic technology, in the following.

In the second step, this paper identifies the *technological features* of ontologies (Section 3). Since ontologies draw from several precursory technologies, it is not surprising that many of the individual features are not unique. However, the claim made in this paper is that combinations of technological features are unique compared to alternative technologies. For instance, there exists a plethora of technologies that offer the feature of *conceptual modeling* but only ontologies combine this feature with *Web compliance*, *formality* and *reasoning* possibilities.

Identifying the technological features serves as the prerequisite for explaining how ontologies provide *benefits for enterprise applications* in the third step (Section 4). For example, the combination of features *conceptual modeling*, *reuse*, and *Web compliance*, has led to innovative business scenarios in the context of linked data. This argumentation line represents a contribution since related papers either discuss the benefits of ontologies on the level of application scenarios, such as interoperability or search, or technological benefits only (cf. [23,64,47,98,89]).

The benefits have to be contrasted with *challenges* when adopting ontologies, e.g., modeling costs or training of employees. Accordingly, another contribution of this paper is the identification of several challenges drawn from more than five years of industrial research (Section 5). The challenges are mainly due to the often complex boundary conditions in an enterprise setting, e.g., technical integration into existing technology stacks becomes necessary.

In addition to the challenges, this paper presents experience reports for several SAP Research case studies (Section 6) where ontologies have been applied in products, pilots, and prototypes. The experience reports highlight which technological features, benefits, and challenges did occur in the corresponding case study and which compromises have been taken.

Besides channeling back the experiences to the Semantic Web community, this paper also identifies potential trade-offs, promising research results and contributes recommendations for future research directions (Section 7).

2. Scenario

This section introduces a scenario in the Oil and Gas domain in which ontologies shall be used in corresponding enterprise applications. Subsequent sections draw from this scenario for the sake of a consistent running example.

The declared goal of the Norwegian Oil and Gas industry is the integration of different field data, the validation of information delivered by different sources, and the exchange of information between off-shore and on-shore facilities [96] (cf. Figure 1).

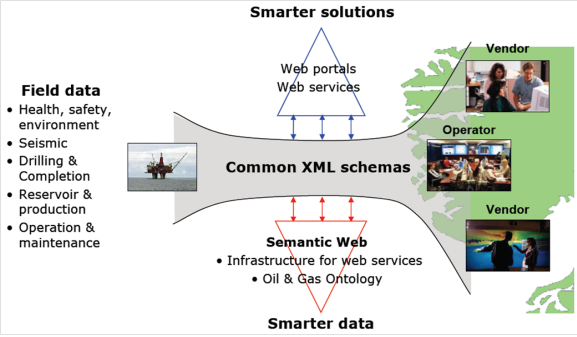

Fig. 1. Ontologies play a major role in the Norwegian Oil & Gas industry's strategy. [1]

This requires that Oil and Gas IT systems, e.g., asset management or facility monitoring, share information according to a reference model, and that they can interpret messages using that model [21]. Such a reference model is given by the ISO 15926 Oil and Gas ontology [54] which is formalized in OWL. The ontology shall serve as a common vocabulary to exchange information between field data and and the onshore participants (operators and vendors). In particular, as pointed out in [1], the offshore industry has huge economic investments in data acquisitions, analysis, visualization, documentation and archiving. Some of the data are in use for decades and it is one of the main assets. Organizational units and IT systems last rarely more than few years. The most stable element in this environment is the terminologies used in the business domains along the value chain. Therefore, ISO 15926 shall capture the terminology in a formal and reusable way.

3. Technological Features of Ontologies

Ontologies inherit from several related and precursory technologies, e.g., conceptual modeling languages, logics, theorem provers, deductive databases, AI, or semantic nets. Therefore, it comes as no surprise that the *individual* technological features of ontologies are not unique but are also offered by related technologies. This makes a comparison with established technologies rather tricky and they are therefore often favored in an enterprise setting.

However, we claim that there are *combinations* of technological features that make ontologies unique compared to related technologies. A careful and complete elicitation of the technological features of ontologies is necessary in order to explain how and why such combinations lead to benefits for enterprise applications in Section 4.[1]

3.1. Conceptual Modeling

Explanation Conceptual modeling is concerned with describing a universe of discourse with the help of a modeling language. According to [42, p. 1], ontologies are a special type of conceptual models based on the following, easy-to-understand modeling constructs:

Classes are collections of instances that have similar properties.

[1] The starting point for the following analysis is [39,44] who introduce communication (here: *Conceptual Modeling* combined with *Formality*), computational inference (here: *Reasoning*) and *Reuse* as features.

Properties represent relations between classes. Classes and properties can be hierarchically organized in taxonomies.

Instances (also called objects or individuals) are concrete members of a class.

Rules & Axioms Rules take the form of *premise* $\rightarrow$ *conclusion* and allow to infer implicit information combined with *Reasoning* (cf. Section 3.8). Axioms allow to capture additional information about classes and properties, e.g., that a relation is transitive or symmetric.

According to [68,46,19], conceptual models better correspond to an end-user's understanding than machine-oriented models. Further, [39] argues that *Conceptual Modeling* paired with *Formality* also facilitates the communication between implemented computational systems and/or humans.

Scenario The most stable element in the Oil and Gas industry is the terminologies. Therefore, capturing this domain in the ISO 15926 ontology provides benefits since it corresponds better to an Oil and Gas expert's conceptualization of the domain.

Related Work There are more established conceptual modeling languages that make use of similar modeling constructs than ontologies as depicted in Table 1.[2]

Table 1
Comparison to established conceptual modeling languages.

	Ontologies	ERM [20]	UML [12]	XSD [28]
Classes	✓	✓	✓	✓
Properties	✓	-	✓	-
Instances	✓	-	(✓)	✓
Rules	✓	-	(✓)	-

3.2. Agile Schema Development

Explanation Databases and (model-driven) software engineering typically start with a conceptual model at design time which is then manually or automatically transformed to an executable model. That means the initial conceptual model is hardly adaptable after the transformation during run time. In contrast, ontology stores allow for agile schema development meaning that classes, properties, rules & axioms, and instances can be managed at run time of an application (also dubbed 'schema last' feature). This is achieved by:

API The basic means for managing classes, properties, rules & axioms as well as instances at run time is an Application Programming Interface (API) provided by an ontology store. The API allows to add, change or delete all constructs during run time of an application.

Evolution Some APIs additionally support automated evolution strategies to unambiguously define the way how changes will be resolved [91]. Ontology changes and their effects can be managed by creating and maintaining different variants of the ontology [70]. This includes reclassification of instances when a class is deleted, for instance.

Dynamic Direct Programming APIs might feature a direct programming model [80], i.e., each object-oriented class or field is an element of the ontology. In dynamically typed languages, the API might also be generated on-the-fly, i.e., changes are reflected in the API at run-time.

Scenario Agile schema development also benefits the Oil and Gas setting, e.g., when new types of equipment have to be captured in the ISO 15926 ontology after it is initially deployed in applications. When a specific class in the taxonomy is deleted, evolution strategies care for its subclasses and for reclassifying all its instances.

Related Work Relational databases require that a schema exists prior to data entry. However, agility is typically given for graph-based data and conceptual modeling languages (cf. [5] for an overview). According to [70], ontology evolution is not the same as database schema evolution because ontologies comprise data (i.e., instances) as well, are more often reused, and are de-centralized by nature. Object-oriented database management systems [53] also support APIs and evolution [15] to a certain extent.

3.3. Direct Interaction

Explanation Through the combination of *Conceptual Modeling* and *Agile Schema Development*, realizing the user-friendly direct interaction with the ontology and is straightforward. An ontology and its instances can be created, changed, and populated by *generic* tools fostered by standardized ontology languages (cf. *Web Compliance*). Direct interaction typically happens via the following UI paradigms:

Graphical Tree or graph-based visualizations of the ontology with editing possibilities (surveyed in [49]).

[2] Note that UML provides instances but in a separate model. Rules & Axioms are also possible but in a separated language (OCL).

Wiki Enhancements of existing Wikis where each page represents an instance with relations to other pages (cf., e.g., Semantic Media Wiki [56]).

Natural language User interfaces that apply controlled natural language, i.e., a constrained part of natural language that can be understood by a human and processed by a computer (e.g., [10]).

Scenario ISO 15926 is engineered collaboratively by special interest groups each of which focusses on a specific sub-domain of Oil and Gas (e.g., drilling and completion or operation and maintenance). Each special interest group is to capture the corresponding sub-domain ontologically and integrate it in the overall ISO 15926. Using the Semantic Media Wiki, it would become possible for the domain experts to collaboratively model and document their sub-domains.

Related Work Direct interaction with relational databases require in-depth knowledge of the relational algebra, e.g., about primary and foreign keys. That means additional abstraction layers have to be established on top to allow a user-friendly interaction. There are other technologies that provide generic tools for direct interaction (e.g., topic maps [90]).

3.4. Reuse

Explanation The declarative nature of ontologies makes them particularly eligible for reuse. That means the effort of conceptually modeling a domain is undertaken once and the resulting ontology can be reused in multiple applications. In addition, ontologies are often attributed as being 'shared,' i.e., users agree on the conceptual model, what potentially increases the level of reuse. According to [85] there are the following levels of agreement:

Individual The ontology is created and used by only one person, e.g., a developer exploits it for reasoning.

Community A *group*, *company* or *sector* agrees on a specific ontology and uses it.

World The ontology is published on the Web (cf. Section 3.6) and can potentially be used by anybody.

Transitions from agreement levels *individual* to *community* or from *community* to *world* require methods for reaching consensus, e.g., [92].

Scenario In the Oil and Gas setting, the ISO 15926 ontology shall be used for several purposes and in different applications. Thus, reuse is one of the inherent features sought by using an ontology. The level of agreement would be *community*, since the ontology shall be used by the Oil & Gas industry.

Related Work Although it is not their initial motivation, technologies such as conceptual database design (with, e.g., ERM [20]), UML [12], Model-Driven Engineering with its platform-independent models [50], or XML Schemas, can achieve reuse as well. In practice, shared conceptualizations are often represented by such languages. Examples are the bulk of SOA reference models such as SOA-RM [60] or SOA-RAF [2] given in UML.

3.5. Best Practices

Explanation Ontologies targeted at reuse, e.g., [73, 7], are often built by ontological analysis. That means classes and properties are related to predefined categories grounded in philosophical studies. What is typically gained is an increased understanding of one's own ontology and a cleaner design. There exist the following best practices:

Foundational Ontologies to prevent modeling from scratch by giving a well-engineered starting point for one's own ontology. The use of such an ontology prompts an ontology engineer to sharpen his/her notion of the concepts to be modeled (cf. DOLCE [37] as an example).

Ontology Design Patterns save modeling efforts by providing templates for ever re-occurring modeling needs. [78]

Quality Criteria The OntoClean [41] approach provides explicit *quality criteria* for building an ontologically correct taxonomy.

Scenario ISO 15926 Part 2 specifies an ontology for long-term data integration, access and exchange to support the evolution of data through time. It can be regarded a foundational ontology in the sense that it is very specific about its ontological assumptions.[9]

Related Work To the best of our knowledge, the idea of having foundational ontologies has not occurred in other disciplines. Note, however, that foundational ontologies can be applied to other conceptual modeling languages as well, e.g., UML class models. There are examples for quality criteria, e.g., normalization in

relational database schemas [25]. There is a bulk of related approaches to ontology design patterns (e.g., software engineering design patterns [36]).

3.6. Web Compliance

Explanation Up until the late 90s there were several proprietary and incompatible ontology languages and corresponding tool suites. The recommendations of the W3C Semantic Web Activity provide Web Compliance for ontologies and allow:

Publication RDF(S) [16] and OWL [62] allow to publish ontologies on the Web. That means every element of an ontology is identified by a URI, XML serializations are available, and ontologies can be distributed, modularized and referenced across the Web. If URIs are used as identifiers for resources and, at the same time, as pointers to additional information on those resources, one speaks of the Web of *Linked Data* [11].

Querying SPARQL [79] may be used as a unified querying language across the different ontologies. SPARQL resembles SQL in syntax but pays tribute to the fact that data is distributed in the Web.

Annotation There are W3C recommendations for annotating existing Web resources with ontologies. Those are SA-WSDL [30] and SA-REST [38] for annotating Web service descriptions and RDFa [3] for annotating HTML pages.

Scenario One reason for formalizing ISO 15926 in OWL, besides its original language EXPRESS, is certainly because OWL is a W3C recommendation and many tools for editing, storage, and reasoning exist.

Related Work Web compliance is also given for the whole set of recommendations that revolve around XML. The differences between XML Schema [28] and RDF(S) are elusive as pointed out in [24].

3.7. Formality

Explanation Typically, ontology languages such as OWL are based on formal logics with model-theoretic semantics. On the one hand, this provides for unambiguous meaning of modeling constructs. As an example, consider the formal semantics of the 'subclass-of' modeling construct. The model theory underlying OWL basically maps the modeling construct to the subset operator in set theory. Contrast this with the incomplete, ambiguous, and informal specification of UML [82]. On the other hand, the model theory is a prerequisite for having a proof theory, and, thus, *Reasoning* becomes possible as another feature. There are three major families of ontology languages [74]:[3]

Logic Programming Languages in this family are geared to deal efficiently with larger sets of instances and the reasoning tasks of *rules* and *instance retrieval* (e.g., F-Logic [52]).

Description Logics Languages in this family are strict subsets of first-order logics geared at the reasoning tasks of *subsumption checks*, *consistency checks*, and *instance classification* (cf. Section 3.8), e.g., OWL-DL [62].

First-Order Logics and higher-order logics allow for greater expressiveness but are typically not decidable, i.e., a sound and complete reasoner cannot be built.

Scenario Since one of the goals of ISO 15926 is to capture the the terminology, formality benefits the endeavor by countering ambiguity. Being a prerequisite for reasoning, formality also helps the goal postulated in [1], which is to enable autonomous solutions by applying reasoning on the basis of ISO 15926.

Related Work Figure 2 shows that formality actually a continuum [95]. At the one extreme, there are purely informal, i.e., natural, languages to specify an 'ontology.' However, in our definition, speaking of an ontology is only valid in the rightmost category of 'logical languages.' In between are languages such as ERM, that do provide a mathematical definition, yet lack a model-theoretic formal semantics as a prerequisite for reasoning.

3.8. Reasoning

Explanation Any logical deduction on the basis of a formal ontology is defined as reasoning in this paper. Different reasoning services are offered by inference engines (also called reasoners) depending on the ontology language.

Subsumption Checking infers implicit super- and subclass relations between classes. Being applied mutually to every possible pair of classes, subsumption checking makes it possible to classify the class hierarchy (*automatic classification*).

[3] Most languages existed prior to being used as an ontology language and stem from related disciplines such as mathematical logics, deductive databases or artificial intelligence.

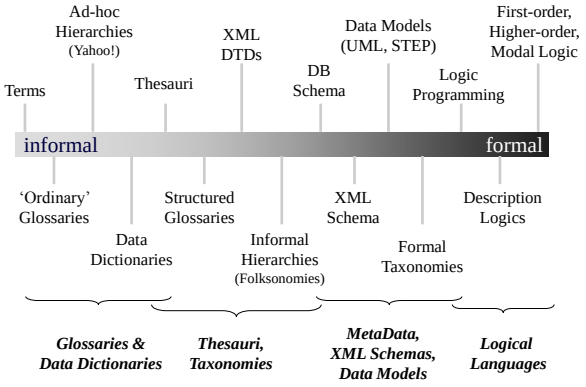

Fig. 2. The continuum of formality. Adapted version from [95].

Consistency Checking examines whether an ontology contains contradictions.

Instance Classification checks whether a specific instance is a member of a given class.

Instance Retrieval means querying for instances including the consideration of rules and axioms.[4]

Scenario Rules can be applied to declaratively capture and infer knowledge on the basis of ISO 15926 in production optimization.

Related Work The reasoning services of logic-programming-based inference engines overlap with Rete-based business rules management systems [34]. However, the latter are less formal with respect to the continuum depicted in Figure 2 and are not paired with a conceptual modeling language.

4. Benefits for Enterprise Applications

The previous section identified individual technological features of ontologies whose combination can benefit enterprise applications. Figure 3 represents a generalization, i.e., several pilots, prototypes and products were analyzed to learn which technological features are *typically* applied to achieve benefits for enterprise applications.[5]

The benefits are ordered by maturity, i.e., established benefits are explained first. The latter ones are still in a state of research. Benefits 2 - 4 are related to user, operation and engineering costs, respectively. Benefit 4 pays tribute to the fact that several new business scenarios have been enabled by ontologies & reasoning. Concrete pilots, prototypes, and products may achieve one or more benefits at the same time.

4.1. Innovative Business Scenarios

Explanation There is evidence that ontologies facilitate hitherto impossible or hard to achieve business scenarios. A business scenario describes future business circumstances based on past and present trends, uncertainties, and assumptions.

Technological Features The crucial technological features in order to reach this benefit are *Reuse* and *Web Compliance*. The latter allows information to be published and queried on the Web. As soon as the level of *Reuse* of an ontology is *community* or even *world*, the ontology and instances act as a global database with a common schema according to the principles of Linked Data. This, paired with the technological feature of *Conceptual Modeling*, i.e., classes, properties, instances and rules in one coherent, easy-to-understand model, opens up hitherto impossible business scenarios as the examples in the following paragraph show.

Examples The first example concerns the British Broadcasting Corporation (BBC). The BBC publishes large amounts of content online, such as text, audio, and video about programmes or music. *Web Compliance* (Linked Data) is used to provide cross-domain navigation and machine-readable representations enabling richer applications on top of BBC's data. The system gives BBC the flexibility and a maintainability benefit: the web site becomes BBC's API. The RDF representations of these web identifiers allow third party developers to use BBC's data to build and monetize new applications. *Reuse* is required for this innovative business scenario and provided by the BBC Programme ontology. [55]

The second example is the Yahoo SearchMonkey [65] which enables a business model for improved result presentation by specialized search engines. In this case *Web Compliance*, especially the feature of *Annotation*, allows Yahoo to harvest RDFa-enriched Web pages and to store the extracted RDF triples in its index. In turn, third party developers can build their own search engine with a development kit allowing access to Yahoo's search index. *Reuse* is required to enable this scenario and given by an ontology prescribed by the third party developer.

[4] This means checking the entailment of ground facts (instances) in terms of logic.

[5] This does not rule out that further technological features could be leveraged in principle.

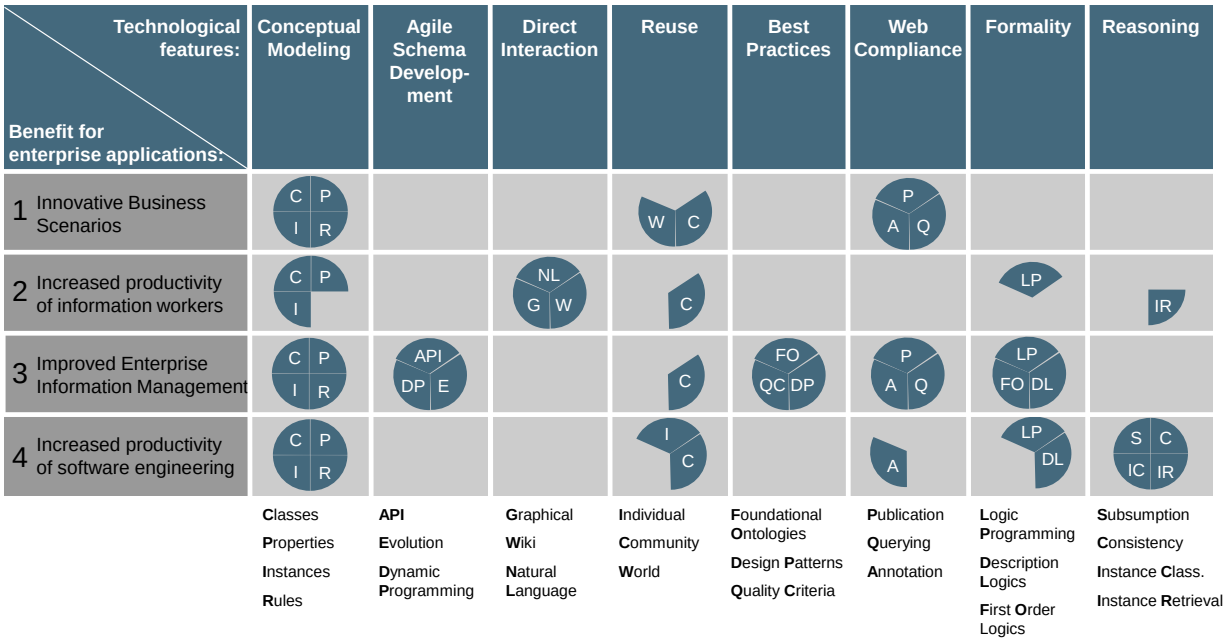

Fig. 3. Technological features of ontologies lead to benefits for enterprise applications.

4.2. Increased Productivity of Information Workers

Explanation The productivity of an information worker can be increased by enabling more efficient access to required information. Information workers elaborate on knowledge-intensive problems that influence company decisions, priorities and strategies. Information workers may also be found in the Oil and Gas setting, e.g., an operator in an onshore control room. According to [76], ontologies can be used for making access to useful information more efficient by improving:

Visualization Capabilities Ontologies are used for improving the appearance of the user interface, i.e., the way information is presented to the user, by *Information Clustering*, *Text Generation*, and *Adaptation of UI Appearance* [76].

Interaction Possibilities The user has to interact with an enterprise application by entering data, selecting items, issuing commands, etc. Ontologies can improve *Browsing*, *Input Assistance*, *Providing Help*, and *UI Integration* [76].

Technological Features The starting point for obtaining this benefit is the integration of all relevant information in a way that is easily consumable by the information worker (*Conceptual Modeling*). This typically includes classes and taxonomies, properties, and instances (rules and axioms are of less importance in this case). The second crucial technological feature is *Direct Interaction*, i.e., the straightforward, user-friendly way of viewing and editing the ontology and instances is possible for the information worker. It is typical for most approaches that they require *Reuse* on the level of *community*, since a group of information workers cooperate on specific tasks. Some approaches in this category also exploit the *Reasoning* capabilities of *Logic Programming* languages.

Examples A good example is SAP's FindGrid application further discussed in Section 6.1. Another example is ontology-enhanced business intelligence such as presented in [87]. Here, an ontology captures the end user's domain in an intuitive way. This could be ISO 15926 for capturing the Oil and Gas terminology. The operator of an onshore control room interacts with the business intelligence solution via the ontology which can also be personalized and changed for her/his purposes. The ontology shields the end user from IT systems' intricacies by means of mapping legacy data structures to the ontology for intuitive interaction. This is achieved by *Instance Retrieval* applying *Rules*.

4.3. Improved Enterprise Information Management

Explanation The goal of Enterprise Information Management (EIM) is to find solutions for optimal use of information within organizations, for instance to sup-

port decision-making processes or day-to-day operations that require the availability of knowledge. EIM tries to overcome traditional IT-related barriers to managing information on an enterprise level.

Technological Features *Conceptual Modeling* is often used for representing an integrated and easy-to-understand view of enterprise information. Surveys that discuss this benefit can be found in [97,69]. In many cases, clarity in representing and storing enterprise information is a must — semantic ambiguities, let alone conceptual inconsistencies, are likely to cause misunderstandings. To ensure sustainable modeling, building the ontology with *Best Practices*, such as *Foundational Ontologies* (often formalized in *First-Order logic*), *Ontology Design Patterns* and *Quality Criteria*, can help to capture enterprise information with particular high quality. Also, modeling a domain once and its enterprise-wide *Reuse* is one of the goals sought by EIM.

Agile Schema Development is of importance because it allows to access and evolve the ontology programmatically over time according to the enterprise's representation needs. Ontology *APIs* with *Evolution* possibilities and *Dynamic Direct Programming* allow to flexibly cope with inevitable schema changes in enterprise settings.

Web Compliance allows for standardized *Publication*, *Annotation* and *Querying* of enterprise information with corresponding tooling. This avoids information silos and enables seamless integration with information on the Web (in particular exploiting available linked data). Finally, *Formality* counters ambiguities of enterprise information.

Examples Disaster management is an example where *Conceptual Modeling* with *Best Practices* can help to improve enterprise information management. In [6,7], we explain that with the many organizations involved in the disaster, crossing regional or even national borders, information exchange is crucial but cumbersome due to differing vocabularies and representations both at human language and IT level. Carefully designed ontologies can be instrumental in addressing this problem, by providing a reference model for humans, and with that the basis for enterprise information management at the IT level. We devise an ontology stack covering the description of damages (caused by the disaster), resources (available to organizations fighting the disaster), and their connection (e.g., which resources are relevant for which damage). To ensure sustainable modeling, we follow the guiding principles of the foundational ontology DOLCE.

In our running example, it is one purpose of ISO 15926 to facilitate and enable information integration across IT systems. The declared goal of the Norwegian Oil and Gas industry is to enable information integration based on ISO 15926 to support, e.g., the integration of different data sources and sensors, the validation of information delivered by different sources, and the exchange of information within and between companies via Web services [96].

4.4. Increased Productivity of Software Engineering

Explanation While the three benefits so far are already evident in commercially used applications and products, a prominent research area of ontologies has been software engineering, cf. [84], where an often sought benefit is the increase of productivity. The research community tried to increase the productivity mainly in three ways. The first way is *Full Automation* of Web service discovery and composition. Activities in this realm are frequently dubbed *Semantic Web Services* [63]. Second, ontologies are also applied to more established software engineering in order to achieve *Cost and Time Reduction*. Third, *Quality Improvements* are often sought in software engineering.

Technological Features *Full Automation* or *Semantic Web Services* first leverage *Conceptual Modeling* to capture additional information for services, e.g., the meaning of inputs and outputs. This information is then annotated to a Web service description (*Web Compliance*), e.g., by SA-WSDL. Semantic Web Services approaches typically rely on the *Formality* of *Description Logics* or *Logic Programming* languages in order to exploit *Reasoning* capabilities for automating tasks such as service discovery or composition.

Approaches that target *Cost and Time Reduction* or *Quality Improvements* typically capture relevant, software-related information by *Conceptual Modeling*. Here, *Rules* are often exploited in order to infer additional knowledge. Note that the level of *Reuse* is often low since the ontology is mainly used within an application for exploiting *Reasoning* functionality. Therefore, as in the case of *Semantic Web Services*, approaches typically rely on the *Formality* of either *Logic Programming* or *Description Logic* languages.

Examples An example for *Full Automation* in the realm of Semantic Web Services is [59]. They use a *Description Logic* ontology as well as *Subsumption*

Reasoning for Web service discovery. Both Web service descriptions and search requests are represented as classes with properties and axioms. *Subsumption Reasoning* then checks whether the search request is subsumable under, i.e., a subclass of, an existing Web service description. If this is the case, both match and the corresponding Web service can fulfill the request.

An example for *Cost and Time Reduction* is [71]. Here, the developer or administrator is supported in typical middleware management tasks, such as ensuring compatibility of library licenses. In order to achieve this, scattered information about software components and accompanying information is integrated by *Conceptual Modeling*. The corresponding ontology is formalized in a *Logic Programming* based language including simple *Rules & Axioms*. The reasoning task of *Instance Retrieval* is used to obtain information, e.g., a pair of libraries with incompatible licenses.

In our running example, building Oil and Gas applications promises to take less time, require less cost, or feature improved quality — independent of whether the application actually uses ISO 15926. The basic idea of Semantic Web Services is also foreseen in the Oil and Gas setting since an architecture for Web services using ISO 15926 is explicitly mentioned.

5. Challenges of Adopting Ontologies

Sections 3 and 4 focussed purely on the positive side of ontologies. However, the benefits have to be contrasted with challenges when adopting ontologies. The challenges presented in this section were identified during more than five years of industrial research and are mainly due to the often complex boundary conditions in an enterprise setting.

Some of the challenges also apply to the introduction of other technologies. However, the severity of the challenges increases for ontologies when moving to the right on the axis depicted in Figure 4.

5.1. Cost-Benefit Ratio

The first challenge is to arrive at a positive overall cost-benefit ratio for each use case of ontologies in an enterprise setting. On the one hand, ontologies promise benefits for enterprise applications as discussed in Section 4. On the other hand, the costs, e.g., caused by technical integration or modeling, have to be summed up and contrasted to the envisioned benefits. Here, the term *Total Cost of Ownership (TCO)* [33] comes into play which is a financial estimate. The purpose of TCO is to help enterprise managers determine direct and indirect costs of a product or system. One way of formalizing the TCO in a software product is shown below

$$TCO \sim TCO\ drivers \times \sharp of\ stacks\ in\ landscape \times \sharp of\ technologies$$

where a *TCO driver* might be any administrative, operational or user specific action. More specifically, such a driver could be the required acquisition of ontology experts, training of existing employees, technical integration costs, modeling costs, maintenance, technology buy-in or redevelopment costs. Each technology introduces different tools, UIs, and system behaviors which all end up in a higher effort to install, learn, run, and maintain software. The *stacks in the landscape* is best explained by an example: with complete ABAP and Java support, the SAP landscape would feature two technology stacks. If ontology editors, stores, or reasoners are to be integrated in both stacks, the TCO will approximately rise by the factor of 2. The *number of technologies* equals 1 if ontology language, reasoner, editor, and store are to be integrated once. If different reasoners, editors, or stores are required, the number will rise accordingly.

Coming back to our running example, an Oil company might be forced to integrate ISO 15926 in the existing IT landscape in order to be interoperable and compliant with partners. This use case affects the information worker and might require the integration of a suitable ontology store and API in its existing IT landscape. However, the Oil company might also feature a large IT department where custom software is developed in-house. In order to increase the productivity of software engineering, the company fancies to establish an ontology-supported environment for its developers. This might require the integration of additional — and potentially different — ontology store, reasoner and API. Therefore, the $\sharp$*of technologies* would equal 2 because the Oil company has to integrate different ontology stores and APIs for two different use cases. If the technology stacks of the IT landscape and the development environment differ, the $\sharp$*of stacks in landscape* would also equal 2.

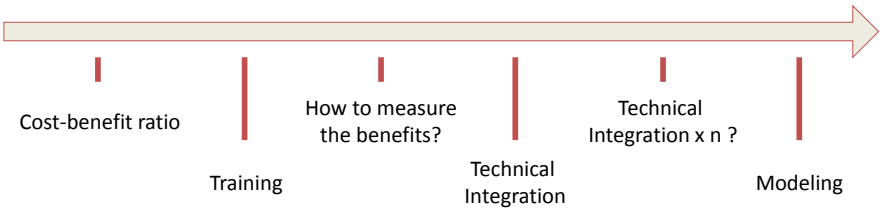

Fig. 4. Overview of challenges.

5.2. Training

Two of the TCO drivers mentioned in Section 5.1 are the *training of existing employees* and the *acquisition of ontology experts*. Both concern the challenge of training and are further discussed below.

Concerning the *training of existing employees*, their expertise typically lies in one specific technology. Even if they are willing to adopt and familiarize with a new technology or paradigm, educating them requires training costs. Usually, the training costs are very high and managers are not willing to expend them unless there is a compelling business case. Here, the intricacies of ontology languages and reasoning techniques, especially description-logic-based, are noteworthy since they are particularly hard to understand for a non logician. [32]

In the case of existing information workers, they, too, have to be trained if they are confronted with ontology interpretation or editing. As an example, even if the Oil company applies a potentially intuitive tool such as the Semantic Media Wiki, basic knowledge about the structure of an ontology is required to efficiently use the tool.

Acquiring ontology experts is also problematic since the technology is still rather new and not established in the industry. Contrast this with the wide-spread nature of relational databases. If there is no convincing business case, an enterprise might decide to realize the use case with conventional technologies, i.e., technologies where there is expertise readily available in the company.

5.3. How to Measure the Benefits?

Calculating the TCO of an additional technology represents an estimate and depends on probabilistic variables. The calculation and determination of the benefits is even less concrete. Therefore, the third challenge lies in measuring the benefits, i.e., finding measures for proving that an ontology-based application is beneficial compared to another solution.

The ontology and Semantic Web community has been struggling to evaluate their contributions accordingly. Indeed, one hardly finds scientific methods or measures to prove the benefits. Other communities share similar struggles, however. It is equally hard to measure the benefits for knowledge management applications, conceptual modeling techniques in general, or object-oriented programming languages compared to procedural languages.

Probably even harder than a sound scientific evaluation is coming up with a business case. A *business case* captures the rationale for initiating a project or task and is the basis of an enterprise manager's decision for investment. The logic of the business case is that, whenever resources or effort are consumed, they should be in support of a specific business need. As an example, consider the exemplary business need of an Oil company to increase the productivity of its in-house IT department with custom software development by 5% thus saving several thousand Euros per year. A proper business case must answer to how this can be achieved

- under consideration of the cost-benefit ratio
- including a deployment plan of resources
- defining quantifiable success criteria
- concerning the business capabilities and impact
- specifying the required investment
- including a project plan

5.4. Technical Integration

The fourth challenge concerns embedding of ontology editors, stores, and reasoners in the existing landscape of the enterprise. As an example, consider an Oil company that wants to incorporate ISO 15926 along a specific ontology store and API in its existing IT landscape. Adaptors have to be written to legacy code and databases, versions of programming languages might have to be switched, specific software components might have to be replaced because of license incompatibilities, etc. The challenge typically increases with

the size of the legacy code and size of the enterprise's portfolio. The following issues have to be addressed:

Build or buy? If unavailable, the enterprise is prompted to buy the technology or to consider redevelopment due to immaturity of existing technology, license incompatibilities, or strategic investments. In the case of buying, the challenges continue below. A redevelopment is only possible if the required expertise is at hand (cf. Section 5.2).

Maturity level of tools Most use cases of ontologies so far emerged in the realm of scientific contributions with simple prototypes and lacking maturity, documentation, and maintenance. Often such projects fail when the original developer or community is not available. This might stop the whole project or lead to redevelopment.

Enterprise scale performance The scientific contributions are often only tested against toy examples and not thoroughly run through a set of real-world test cases with enterprise-scale amounts of data. Depending on the ontology language, enterprise scale performance might not even be theoretically possible due to the complexity of the language.

How to handle ontologies in the transport system? Applications for large enterprises typically feature a transport system for dealing with updates [45]. When mission critical software is to be updated, a test system is set up which can be transported automatically to a productive system. So far it remains unclear how to deal with ontologies in such software transport systems.

5.5. Technical Integration $\times n$?

Suppose several use cases of ontologies in software engineering are adopted by an enterprise. Each use case might require a different ontology language possibly based on a different logic. That means different use cases might require different editors, stores, and reasoners. In this case, the challenge of technical integration might have to be faced $\times n$. The situation is further complicated when the different use cases are geared at different beneficiaries. Some might benefit the developer and some might benefit the end user, eventually. That means editor, reasoner, or store might have to be integrated both in the developer run time or the application run time environment. Both environments might be completely different.

Coming back to the example of Section 5.1, the Oil company is prompted to apply ontologies for two different use cases. Therefore, the hope is to minimize n ideally to 1. That means, the same ontology language, editor, store and reasoner can be used for both use cases.

5.6. Modeling

5.6.1. Target Audience

The audience for most ontology editing tools is experts in *Conceptual Modeling* with the required knowledge in the *Formality* of ontology languages. However, this is not the audience who is expected to create ontologies. Often, subject matter experts of domains are the most suitable people for authoring ontologies. They usually have no technical expertise in complex and deep computer science subjects. [26,8]

This phenomenon is also apparent in the Oil & Gas scenario where special interest groups have to model their corresponding domain of expertise (drilling and completion, operations and maintenance, or production and reservoir). However, the group members are typically no ontology experts.

5.6.2. Upfront Modeling Costs

Ontologies are particularly often attributed to be afflicted with large upfront modeling cost. Although true, this holds for conceptual modeling languages in general and also for modeling database schemas [31,88]. [86] present a specific cost model for engineering ontologies and identify the following cost drivers:

Product-related cost drivers account for the impact of the characteristics of the ontology on the overall costs. Examples are the domain analysis complexity, the conceptualization complexity, or the documentation needs.

Personnel-related cost drivers emphasize the role of team experience, ability and continuity with respect to the effort required by an ontology development project.

Project-related cost drivers related to the characteristics of the overall ontology development project and their impact on the total costs (e.g., level of support and automation provided by ontology engineering software).

5.6.3. Use Case Dependency

Indeed, if several use cases are to be applied, the hope is that modeling a domain once and capturing it in an ontology will suffice for all use cases. However, different use cases typically have different represen-

tation needs. Below, we show two typical phenomena that cause this challenge.[6]

Regarding the first phenomenon, consider the domain of Oil and Gas as a simple example. When modeling this domain, there might be the need to represent information about equipment (such as pumps and valves) and at which production plant it is deployed. For one use case it might be enough to represent equipment and plant as classes and to establish a simple relation between them to capture the deployment (cf. Figure 5 top). However, another use case might have to capture information about the deployment itself, e.g., its duration. Therefore, a simple relation between equipment and plant is not sufficient to meet this requirement since decidable ontology languages typically do not allow attaching attributes and properties to other properties. The deployment relation has to be reified to a separate class (cf. Figure 5 bottom).

Fig. 5. Different use cases might require different representations of the same domain of discourse.

There is a second phenomenon that leads to different representations of the same domain. For example, the modeling applied by [61] to the domain of Web services is geared towards reasoning. That means modeling decisions are taken in a way such that subsumption checking can be applied to eventually discover Web services. In contrast, [73] models the same domain but for the purpose of establishing a reference ontology. Here, modeling decisions are influenced by *Best Practices* including ontology quality criteria such as OntoClean [41] leading to a different ontology.

According to our experience, ontologies are often not reused and built without consensus geared to a specific use case, e.g., for purely exploiting *Reasoning* features and transparent to the user.

[6]Note that the phenomena are not idiosyncratic to ontology languages but are also known from other conceptual modeling languages or database schema design (normalization theory [25]).

6. Case Studies

The following section highlights experience reports for two SAP Research case studies where ontologies have been applied and also comes back to one particular realization of the Oil & Gas scenario. The experience reports highlight which technological features, benefits, and challenges did occur and which compromises have been taken.

The first case study, code-named FindGrid, is a success story and indeed the first product of SAP that applies ontologies. The second case study, code-named xCarrier, has been successfully prototyped and demonstrated to board members with positive feedback but proved to be too disruptive. The third case study is in fact a negative example where the challenges outweighed the sought after benefits.

6.1. FindGrid

Problem Knowledge in enterprises is not easily accessible since information is hidden in disparate locations or in the heads of experts. There is typically no way to capture and reuse know-how. Established solutions, such as search engines, enterprise content management systems, case management systems or collaboration spaces only solve parts of the issue.

Instead, enterprise memory should be built via a collaborative, self-learning environment that enables knowledge capture across people, teams and the whole enterprise. Information workers should be enabled to activate, consolidate and summarize artifacts such as folders, bookmarks, tags, notes, or pictures, thus creating, harmonizing and sustaining the enterprise memory.

Relevant Technological Features FindGrid provides automated tool support for all of the above. In essence, the goal is to increase the productivity of information workers of SAP's customers by accelerated speed to knowledge and new ideas. FindGrid is a solution that supports business research activities, such as market or product research. A single interface allows a product manager to search for information, organize findings, and collaborate with others.

As can be seen in Figure 6, the solution relies on a *Semantic Integration Layer*. This layer manages the *Enterprise Memory* upon which *Semantic Functions*, such as search or auto-completion, are realized.

The following technological features are exploited: FindGrid relies on *Agile Schema Development* since it

requires the 'schema last' feature to cope with building the enterprise memory via a collaborative, self-learning environment. The schema is captured via *Conceptual Modeling*, in this particular case via an informal ontology (i.e., a semantic net [57]). The semantic net captures information about cases, folders, associated terms, etc. The level of *Reuse* is on *Community* since all users implicitly share its conceptualization via FindGrid.

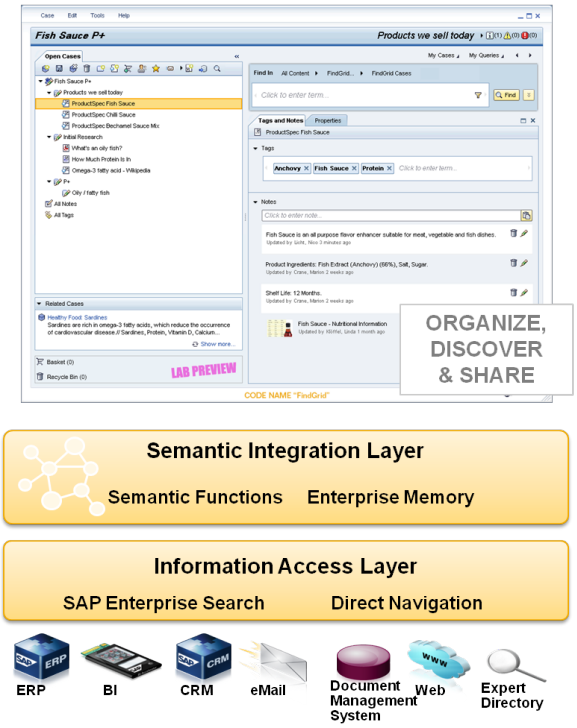

Fig. 6. FindGrid layered architecture.

Relevant Challenges When FindGrid was initiated, SAP did not own the required ontology store, so a decision had to be made with respect to *Build or buy?* (cf. challenge *Technical Integration*). For the first version of FindGrid, the decision was taken to license an existing ontology store with complementary services for realizing the *Semantic Functions*.

Another major challenge concerned the *Cost-Benefit Ratio*, especially the TCO. In the first version of FindGrid, the ontology store has been integrated as 'black box,' i.e., unavailable to other applications that require such a store as well. The rationale behind this decision was that developing enabling technology should not hinder the development of the actual solution. Besides, this decision avoided the cost of integrating the store in the comprehensive SAP technology platform in the first place and allowed faster prototyping.

Trade off However, SAP intends to develop further applications that leverage the semantic integration layer. The 'black box' integration of the ontology store and functionality in the first version of FindGrid would have to be duplicated in each of the further applications. Instead, the semantic integration layer has to become an integral part of SAP's comprehensive technology platform. The semantic integration layer will thus be incrementally factored out of FindGrid and redeveloped on the basis of SAP's in-memory graph database system (HANA and AIS [29,13]).

6.2. xCarrier

Problem This case study concerns the need of SAP's customers to integrate multiple carrier services for different destinations and goods to decrease transportation costs. As an example, consider a manufacturing company that needs to ship its goods via the services of carriers such as UPS, DHL, or FedEx depending on dimension, weight, or destination. An internal SAP prototype, called *xCarrier* [83], addressed B2B integration on the technical level by introducing *XCE* (xCarrier Enterprise) and *XCC* (xCarrier Connect) on top of the shipper and carrier backends, respectively (cf. Figure 7). XCC provides normative Web service operations for rate-lookup or tracking which are invoked by its counterpart XCE. Although a significant benefit to the one-off manual integration, this internal SAP prototype still did not address the following problems:

1. How to describe carrier service products?
2. How to find suitable carrier services?
3. How to automatically select the appropriate carrier service products?

Relevant Technological features In order to address the first problem, *Conceptual Modeling* has been applied to describe carrier service products via an OWL-DL ontology. Each carrier service product is represented as a class with ship from and to properties to locations. Further classes capture information about shipping items such as parcels, containers or documents and many more. The second problem was addressed by introducing another software component, viz., *XCD* (xCarrier Discovery), which allows carriers to publish their service descriptions via XCC and shippers to find suitable carrier service products. A *Carrier Manager* was developed which applies the benefit

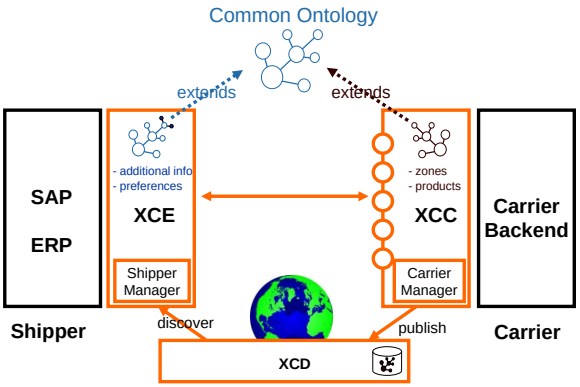

Fig. 7. xCarrier setup. Circles in XCC (xCarrier Connect) represent Web service operation interfaces.

of *Direct Interaction* allowing the definition of carrier service products. The solution to the third problem is based on [35], where we reduce selection and discovery to the *Formality* of *Description Logics* and subsequent *Subsumption Checking*. [35] also allows to define preferences in the ontology to specify which carrier service product is to be used depending on a specific shipment request. This is done graphically in the *Shipper Manager*. All this increased the productivity of software engineering since selection, discovery and preferences could be reduced to existing reasoning services.

Web Compliance improves enterprise information management since SAP could prescribe a *Common Ontology* about carrier service products and publish it on `www.sap.com`, for instance. Carriers could then specialize the *Common Ontology* for their specific representation needs and publish on their domain. Specific carrier service description can still be maintained in XCD but reference their corresponding ontologies. Our selection and discovery approach still works because of the open-world assumption of OWL-DL. WSDL descriptions of XCC can apply *Annotation* by applying SA-WSDL in order to link to their ontology. Similar holds for the XML messages that are payload of actual SOAP messages during runtime. Shippers may also specialize the *Commmon Ontology* to represent additional information required for specifying their preferences.

Relevant Challenges One major challenge of this case study has been *Training*. The case study relies on the approach in [35] which makes use of intricate description logic capabilities. Since this could be understood only by the authors but not by the remaining software engineering team, the hurdle was too high. Another challenge proved to be *Enterprise scale performance* (the reasoner has to process up to 20.000 automatic classifications per hour) and the *Maturity level of tools* (no commercially proven description logic reasoner has been available at that time). Another challenge was to convince decision makers of a positive *Cost-benefit Ratio* compared to a solution with established technology. Also here, the *Build or buy?* challenge was evident and led to many problems on the side of decision makers.

Trade off The dilemma between technological features and challenges was solved by not productizing xCarrier. Apart from the challenges there were other reasons, e.g., lack of adoption of the carriers and organizational obstacles. In other words, the proposed solution was too disruptive at that time.

6.3. Oil & Gas

Problem The Oil & Gas scenario in Section 2 requires a middleware solution for composing existing applications and information repositories (such as rotating equipment monitoring, facility monitoring, engineering systems, asset management) to new applications (such as drilling program planning, production optimization, equipment fault detection, or plan turnaround). Such composite applications depend on information stemming from several existing IT systems, and, thus, benefit from semantically unambiguous information exchange provided by ISO 15926.

One commercially available middleware solution is given by [22] which however existed prior to the introduction of the ISO 15926 ontology. The middleware provides an enterprise service bus exchanging messages via a set of existing standards in Oil & Gas. This set of standards is proprietarily represented via a conceptual model, i.e., [77], which is given in UML, however. The problem was to incorporate support for ISO 15926 in this existing environment.

Relevant Technological features The main technological feature sought in this particular case is *Reuse* — the middleware has to pay tribute to the fact that ISO 15926 is standardized, and, thus, needs to support the standard.

Relevant Challenges The main challenges as indicated above has been the *Technical Integration* and *Cost-Benefit Ratio*. The middleware solution, including its UML-based conceptual model, existed prior to the task of integrating ISO 15926, so immense integra-

tion costs would have to be justified in order to support the ontology.

Trade off The solution chosen was to remodel and adapt parts of ISO 15926 in a self-owned modeling environment, viz., [58], which required manual effort and eventually led to the deviation of the standard. The deviation was required to allow for scalability of storage [66].

7. Recommendations

This paper delineates the technological features of ontologies and explains why these features are relevant to achieve specific business benefits. In addition, this paper identifies a set of challenges when trying to adopt ontologies in a given enterprise setting. The following discussion elicits potential trade-offs and recommends future research directions for each challenge.

Cost-Benefit Ratio A recent trend to lower TCO is to use semantic technologies as a service (cf., [27]). According to this idea, users and developers can exploit ontologies, e.g., an ontology store, without the need for an own infrastructure, benefitting from the elastic scalability, flexibility, ease of deployment and maintenance in the cloud. In this context, cloud services present a viable alternative to in-house solutions promising significant reduction of cost and ease of use.

Training More research should be conducted in better integration with the software engineering community and their wide-spread and mature tools. A good example is OWL2Ecore [81] which enables software engineers to develop applications on the basis of an ontology in their familiar environment, viz., Eclipse. This avoids that software engineers have to acquaint with ontologies.[7]

Another proposed remedy is to include the topic of ontologies in standard curricula in computer science courses. Some recommendations for how curricula should look like are given in [67]. Also, the Semantic Web community should target publishing their research results in the 'benefitting community.' A good example is [72], where ontologies are used to facilitate management of application servers. Correspondingly, the contribution has been positioned in the middleware community.

How to measure the benefits? A cost model for ontology engineering published in [86] has been proposed in the Semantic Web community. Together with first efforts of measuring and comparing cost and benefits, e.g., in [18], this is an early but promising research direction which should be strengthened.

With respect to defining a business case, the communiqué in [94] is a promising endeavor in that it assists in making the case for the use of ontology by providing concrete application examples, value metrics, and advocacy strategies. The communiqué targets ontology technology evangelists who already get the value but want to overcome the blank stares they get when trying to explain it to people who do not.

Technical Integration When ontologies are applied in enterprises, they almost always have to be integrated in existing IT landscapes with several boundary conditions. One stream of research, viz., [75], addresses this challenge of *technical integration* by introducing a reference architecture for using ontologies in existing enterprise settings in a non-intrusive way.

Technical Integration $\times n$? The hope is that technical integration is not required on a 'per use case' basis but that software components such as ontology editor, stores, and APIs can be factored out much like what happened with database management systems. Endeavors, such as the NeOn API and plug-ins [93], have been a step in this direction.

Modeling One established area of countering the challenge of *Upfront Modeling Costs* is ontology learning. Although initial learning results require manual adaptations, the approach helps to bootstrap ontologies as shown in [43], for instance. Another idea is to only model what is initially required and then exploit the technological feature of *Agile Schema Development* to evolve the schema at run time.

[14] is a way to address *Use Case Dependency* by contextualizing ontologies. Specific parts of the ontology can be kept local, when they do not represent a shared view but only a party's view of a domain.

Acknowledgements My appreciation to Kunal Verma, the initial editor, for valuable feedback shaping the final version as well as the initial reviewers Rama Akkiraju and an anonymous reviewer. Further acknowledgments go to Heiko Paulheim, Boris Motik, and Frithjof Dau for open reviews as well as Simon Schei-

[7] This is not to be confused with exploiting the plethora of existing UML editors for ontology engineering, e.g., [17], what serves primarily the Semantic Web community.

der, Michael Uschold, and Leo Obrst for official reviews. Many thanks also to Nico Licht from the FindGrid team and my colleague Andreas Friesen.

References

[1] Integrated operations and the Oil & Gas ontology. Technical report, The Norwegian Oil Industry Association (OLF), POSC Caesar Association (PCA), 2008.

[2] Reference architecture foundation for service oriented architecture 1.0. Committee Draft 2, OASIS, Oct 2009. http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/soa-ra-cd-02.pdf.

[3] B. Adida, M. Birbeck, S. McCarron, and S. Pemberton. RDFa in XHTML: Syntax and processing. Recommendation, W3C, Oct 2008. http://www.w3.org/TR/rdfa-syntax/.

[4] J. Angele. Ontoprise: Semantic web technologies at business. In C. Pautasso, C. Bussler, M. Walliser, S. Brantschen, M. Calisti, and T. Hempfling, editors, *Emerging Web Services Technology*, Whitestein Series in Software Agent Technologies and Autonomic Computing, pages 1–2. Birkhäuser Basel, 2007.

[5] R. Angles and C. Gutierrez. Survey of graph database models. *ACM Comput. Surv.*, 40:1:1–1:39, February 2008.

[6] G. Babitski, S. Bergweiler, O. Grebner, D. Oberle, H. Paulheim, and F. Probst. SoKNOS — using semantic technologies in disaster management software. In G. Antoniou, M. Grobelnik, E. P. B. Simperl, B. Parsia, D. Plexousakis, P. D. Leenheer, and J. Z. Pan, editors, *The Semanic Web: Research and Applications - 8th Extended Semantic Web Conference, ESWC 2011, Heraklion, Crete, Greece, May 29 - June 2, 2011, Proceedings, Part II*, volume 6644 of *Lecture Notes in Computer Science*, pages 183–197. Springer, 2011.

[7] G. Babitski, F. Probst, J. Hoffmann, and D. Oberle. Ontology design for information integration in disaster management. In S. Fischer, E. Maehle, and R. Reischuk, editors, *Informatik 2009: Im Focus das Leben, Beiträge der 39. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 28.9.-2.10.2009, Lübeck, Proceedings*, volume 154 of *LNI*, pages 3120–3134. GI, 2009.

[8] E. G. Barriocanal, M.-Á. Sicilia, and S. S. Alonso. Usability evaluation of ontology editors. *Knowledge Organization*, 32(1):1–9, 2005.

[9] R. Batres, M. West, D. Leal, D. Price, K. Masaki, Y. Shimada, T. Fuchino, and Y. Naka. An upper ontology based on iso 15926. *Computers & Chemical Engineering*, 31(5-6):519–534, 2007.

[10] A. Bernstein and E. Kaufmann. Gino - a guided input natural language ontology editor. In I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *The Semantic Web - ISWC 2006, 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006, Proceedings*, volume 4273 of *Lecture Notes in Computer Science*, pages 144–157. Springer, 2006.

[11] C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.

[12] G. Booch, J. E. Rumbaugh, and I. Jacobson. The unified modeling language user guide. *J. Database Manag.*, 10(4):51–52, 1999.

[13] C. Bornhövd, R. Kubis, W. Lehner, H. Voigt, and H. Werner. Flexible information management, exploration and analysis in sap hana. In M. Helfert, C. Francalanci, and J. Filipe, editors, *DATA 2012 - Proceedings of the International Conference on Data Technologies and Applications, Rome, Italy, 25-27 July, 2012*, pages 15–28. SciTePress, 2012.

[14] P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt. Contextualizing ontologies. *J. Web Sem.*, 1(4):325–343, 2004.

[15] P. Brèche. Advanced principles for changing schemas of object databases. In P. Constantopoulos, J. Mylopoulos, and Y. Vassiliou, editors, *Advances Information System Engineering, 8th International Conference, CAiSE'96, Heraklion, Crete, Greece, May 20-24, 1996, Proceedings*, volume 1080 of *Lecture Notes in Computer Science*, pages 476–495. Springer, 1996.

[16] D. Brickley and R. Guha. RDF vocabulary description language 1.0: RDF Schema. Recommendation, W3C, Feb 2004. http://www.w3.org/TR/rdf-schema/.

[17] S. Brockmans, R. Volz, A. Eberhart, and P. Löffler. Visual modeling of OWL DL ontologies using UML. In S. A. McIlraith, D. Plexousakis, and F. van Harmelen, editors, *The Semantic Web - ISWC 2004: Third International Semantic Web Conference,Hiroshima, Japan, November 7-11, 2004. Proceedings*, volume 3298 of *Lecture Notes in Computer Science*, pages 198–213. Springer, 2004.

[18] T. Bürger and E. P. B. Simperl. Measuring the benefits of ontologies. In R. Meersman, Z. Tari, and P. Herrero, editors, *On the Move to Meaningful Internet Systems: OTM 2008 Workshops, OTM Confederated International Workshops and Posters, ADI, AWeSoMe, COMBEK, EI2N, IWSSA, MONET, OnToContent + QSI, ORM, PerSys, RDDS, SEMELS, and SWWS 2008, Monterrey, Mexico, November 9-14, 2008. Proceedings*, volume 5333 of *Lecture Notes in Computer Science*, pages 584–594. Springer, 2008.

[19] H. C. Chan, K. K. Wei, and K. Siau. Conceptual level versus logical level user-database interaction. In J. I. DeGross, I. Benbasat, G. DeSanctis, and C. M. Beath, editors, *Proceedings of the International Conference on Information Systems, ICIS 1991, December 16-18, 1991, New York, NY, USA*, pages 29–40. Association for Information Systems, 1991.

[20] P. P. Chen. The entity-relationship model - toward a unified view of data. *ACM Trans. Database Syst.*, 1(1):9–36, 1976.

[21] F. Chum. Semantic technologies at the ecosystem level. In P. Horowitz, editor, *Semantic Web in the Enterprise*, volume Spring of *Technology Forecast*. PwC, 2009.

[22] R. Credle, V. Akibola, V. Karna, D. Panneerselvam, R. Pillai, and S. Prasad. Discovering the business value patterns of chemical and petroleum integrated information framework. Red Book SG24-7735-00, IBM, August 2009.

[23] F. Dau. Semantic technologies for enterprises. In S. Andrews, S. Polovina, R. Hill, and B. Akhgar, editors, *Conceptual Structures for Discovering Knowledge - 19th International Conference on Conceptual Structures, ICCS 2011, Derby, UK, July 25-29, 2011. Proceedings*, volume 6828 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2011.

[24] S. Decker, S. Melnik, F. van Harmelen, D. Fensel, M. C. A. Klein, J. Broekstra, M. Erdmann, and I. Horrocks. The semantic web: The roles of XML and RDF. *IEEE Internet Computing*, 4(5):63–74, 2000.

[25] C. Delobel. Normalization and hierarchical dependencies in the relational data model. *ACM Trans. Database Syst.*,

3(3):201–222, 1978.

[26] A. J. Duineveld, R. Stoter, M. R. Kenepa, and V. R. Benjamins. Wondertools? a comparative study of ontological engineering tools. *International Journal of Human-Computer Studies*, 52(6):1111–1133, 2000.

[27] A. Eberhart, P. Haase, D. Oberle, and V. Zacharias. Semantic technologies and cloud computing. In D. Fensel, editor, *Foundations for the Web of Information and Services*, pages 239–251. Springer, 2011.

[28] D. C. Fallside and P. Walmsley. XML Schema part 0: Primer second edition. Recommendation, W3C, Oct 2004. `http://www.w3.org/TR/xmlschema-0/`.

[29] F. Färber, S. K. Cha, J. Primsch, C. Bornhövd, S. Sigg, and W. Lehner. SAP HANA database: data management for modern business applications. *SIGMOD Rec.*, 40(4):45–51, Jan. 2012.

[30] J. Farrell and H. Lausen. Semantic annotations for WSDL and XML Schema. Recommendation, W3C, Aug 2007. `http://www.w3.org/TR/sawsdl/`.

[31] A. Felfernig. Effort estimation for knowledge-based configuration systems. In F. Maurer and G. Ruhe, editors, *Proceedings of the Sixteenth International Conference on Software Engineering & Knowledge Engineering (SEKE'2004), Banff, Alberta, Canada, June 20-24, 2004*, pages 148–154, 2004.

[32] D. Fensel, M.-C. Rousset, and S. Decker. Workshop on comparing description and frame logics. *Data Knowl. Eng.*, 25(3):347–352, 1998.

[33] B. Ferrin and R. E. Plank. Total cost of ownership models: An exploratory study. *Journal of Supply Chain Management*, 38(3), 2002.

[34] C. Forgy. Rete: A fast algorithm for the many patterns/many objects match problem. *Artif. Intell.*, 19(1):17–37, 1982.

[35] A. Friesen and K. Namiri. Towards semantic service selection for B2B integration. In N. Koch and L. Olsina, editors, *Workshop Proceedings of the 6th International Conference on Web Engineering, ICWE 2006, Palo Alto, California, USA, July 11-14, 2006*, volume 155 of *ACM International Conference Proceeding Series*, page 17. ACM, 2006.

[36] E. Gamma, R. Helm, R. E. Johnson, and J. M. Vlissides. Design patterns: Abstraction and reuse of object-oriented design. In O. Nierstrasz, editor, *ECOOP'93 - Object-Oriented Programming, 7th European Conference, Kaiserslautern, Germany, July 26-30, 1993, Proceedings*, volume 707 of *Lecture Notes in Computer Science*, pages 406–431. Springer, 1993.

[37] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, and L. Schneider. Sweetening ontologies with DOLCE. In A. Gómez-Pérez and V. R. Benjamins, editors, *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings*, volume 2473 of *Lecture Notes in Computer Science*, pages 166–181. Springer, 2002.

[38] K. Gomadam, A. Ranabahu, and A. Sheth. SA-REST: Semantic annotation of web resources. Member submission, W3C, Apr 2010. `http://www.w3.org/Submission/SA-REST/`.

[39] M. Grüninger and J. Lee. Ontology applications and design - introduction. *Commun. ACM*, 45(2):39–41, 2002.

[40] N. Guarino, D. Oberle, and S. Staab. What is an ontology? In S. Staab and R. Studer, editors, *Handbook on Ontologies*, Handbooks on Information Systems. Springer, 2nd edition, 2009.

[41] N. Guarino and C. A. Welty. Evaluating ontological decisions with OntoClean. *Commun. ACM*, 45(2):61–65, 2002.

[42] G. Guizzardi and T. A. Halpin. Ontological foundations for conceptual modelling. *Applied Ontology*, 3(1-2):1–12, 2008.

[43] H. Guo, A. Ivan, R. Akkiraju, and R. Goodwin. Learning ontologies to improve the quality of automatic web service matching. In *2007 IEEE International Conference on Web Services (ICWS 2007), July 9-13, 2007, Salt Lake City, Utah, USA*, pages 118–125. IEEE Computer Society, 2007.

[44] M. Hepp. Ontologies: State of the art, business potential, and grand challenges. In M. Hepp, P. D. Leenheer, A. de Moor, and Y. Sure, editors, *Ontology Management, Semantic Web, Semantic Web Services, and Business Applications*, volume 7 of *Semantic Web And Beyond Computing for Human Experience*, pages 3–22. Springer, 2008.

[45] J. A. Hernandez, J. Keogh, and F. Martinez. *SAP R/3 Handbook*, chapter 6, pages 215–270. McGraw-Hill/Osborne, third edition, 2006.

[46] A. R. Hevner, S. T. March, J. Park, and S. Ram. Design science in information systems research. *MIS Quarterly*, 28(1):75–105, 2004.

[47] P. Horowitz, editor. *Semantic Web in the Enterprise*, volume Spring of *Technology Forecast*. PwC, 2009.

[48] B. Inmon. The semantics mystery. BeyeNETWORK, Article no. 4605, `http://www.b-eye-network.com/print/4605`, Jun 2007.

[49] A. Katifori, C. Halatsis, G. Lepouras, C. Vassilakis, and E. Giannopoulou. Ontology visualization methods — a survey. *ACM Comput. Surv.*, 39(4), Nov. 2007.

[50] S. Kent. Model driven engineering. In M. Butler, L. Petre, and K. Sere, editors, *Integrated Formal Methods*, volume 2335 of *Lecture Notes in Computer Science*, pages 286–298. Springer Berlin / Heidelberg, 2002.

[51] N. D. Kho. Semantics in practice. EContent — Digital Content Strategies & Resources, `http://www.econtentmag.com/?ArticleID=70970`, Nov 2010.

[52] M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *J. ACM*, 42(4):741–843, 1995.

[53] W. Kim. Object-oriented databases: Definition and research directions. *IEEE Transactions on Knowledge and Data Engineering*, 2:327–341, 1990.

[54] J. W. Kluewer, M. G. Skjæveland, and M. Valen-Sendstad. ISO 15926 templates and the semantic web. In *W3C Workshop on Semantic Web in Oil & Gas Industry, Houston, TX, USA, 9–10 December, 2008*, 2008.

[55] G. Kobilarov, T. Scott, Y. Raimond, S. Oliver, C. Sizemore, M. Smethurst, C. Bizer, and R. Lee. Media meets semantic web - how the BBC uses DBpedia and linked data to make connections. In L. Aroyo, P. Traverso, F. Ciravegna, P. Cimiano, T. Heath, E. Hyvönen, R. Mizoguchi, E. Oren, M. Sabou, and E. P. B. Simperl, editors, *The Semantic Web: Research and Applications, 6th European Semantic Web Conference, ESWC 2009, Heraklion, Crete, Greece, May 31-June 4, 2009, Proceedings*, volume 5554 of *Lecture Notes in Computer Science*, pages 723–737. Springer, 2009.

[56] M. Krötzsch, D. Vrandecic, and M. Völkel. Semantic MediaWiki. In I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *The Semantic Web - ISWC 2006, 5th International Semantic Web*

Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006, Proceedings, volume 4273 of *Lecture Notes in Computer Science*, pages 935–942. Springer, 2006.

[57] F. Lehmann. Semantic networks. *Computers & Mathematics with Applications*, 23(2):1–50, Mar 1992.

[58] D. Leroux, M. Nally, and K. Hussey. Rational software architect: A tool for domain-specific modeling. *IBM Systems Journal*, 45(3):555–568, 2006.

[59] L. Li and I. Horrocks. A Software Framework for Matchmaking Based on Semantic Web Technology. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003, Budapest, Hungary, 20-24 May 2003*, pages 331–339. ACM, 2003.

[60] C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, and R. Metz. Reference model for service oriented architecture 1.0. Oasis standard, OASIS, Oct 2006.

[61] D. L. Martin, M. H. Burstein, D. V. McDermott, S. A. McIlraith, M. Paolucci, K. P. Sycara, D. L. McGuinness, E. Sirin, and N. Srinivasan. Bringing semantics to web services with OWL-S. In C. L. Williamson, M. E. Zurko, P. F. Patel-Schneider, and P. J. Shenoy, editors, *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 243–277. ACM, 2007.

[62] D. L. McGuinness and F. van Harmelen. OWL Web Ontology Language Overview. Recommendation, W3C, Feb 2004. `http://www.w3.org/TR/owl-features/`.

[63] S. A. McIlraith, T. C. Son, and H. Zeng. Semantic web services. *IEEE Intelligent Systems*, 16(2):46–53, 2001.

[64] T. Menzies. Cost benefits of ontologies. *Intelligence*, 10(3):26–32, 1999.

[65] P. Mika. Year of the monkey: Lessons from the first year of SearchMonkey. In S. Fischer, E. Maehle, and R. Reischuk, editors, *Informatik 2009: Im Focus das Leben, Beiträge der 39. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 28.9.-2.10.2009, Lübeck, Proceedings*, volume 154 of *LNI*, page 387. GI, 2009.

[66] F. Myren, U. Pletat, and J. W. Kluewer. Model driven integration architecture for IO G2 information – reference semantic model alignment to ISO 15926. Semantic Days 2009, 18 - 20 May, Clarion Hotel Stavanger, Session 6: Semantic technology for IO Generation 2, 2009.

[67] F. Neuhaus, E. Florescu, A. Galton, M. Grüninger, N. Guarino, L. Obrst, A. Sanchez, A. Vizedom, P. Yim, and B. Smith. Creating the ontologists of the future. *Applied Ontology*, 6(1):91–98, 2011.

[68] D. Norman. Cognitive engineering. In D. Norman and S. Daper, editors, *User Centered System Design*, pages 31–61. Lawrence Erlbaum Associates, Hillsdale, New Jersey, USA, 1986.

[69] N. F. Noy. Semantic integration: a survey of ontology-based approaches. *SIGMOD Rec.*, 33:65–70, December 2004.

[70] N. F. Noy and M. C. A. Klein. Ontology evolution: Not the same as schema evolution. *Knowl. Inf. Syst.*, 6(4):428–440, 2004.

[71] D. Oberle. *Semantic Management of Middleware*, volume 1 of *Semantic Web And Beyond Computing for Human Experience*. Springer, 2006.

[72] D. Oberle, A. Eberhart, S. Staab, and R. Volz. Developing and Managing Software Components in an Ontology-based Application Server. In H.-A. Jacobsen, editor, *Middleware 2004, ACM/IFIP/USENIX 5th International Middleware Conference, Toronto, Ontario, Canada*, volume 3231 of *LNCS*, pages 459–478. Springer, 2004.

[73] D. Oberle, S. Lamparter, S. Grimm, D. Vrandecic, S. Staab, and A. Gangemi. Towards ontologies for formalizing modularization and communication in large software systems. *Applied Ontology*, 1(2):163–202, 2006.

[74] D. Oberle, S. Staab, and R. Volz. Three dimensions of knowledge representation in wonderweb. *Künstliche Intelligenz*, 1:31–35, 2005.

[75] H. Paulheim, D. Oberle, R. Plendl, and F. Probst. An architecture for information exchange based on reference models. In *Software Language Engineering - Fourth International Conference, SLE 2011, Praga, Portugal, July 3-4, 2011*, Lecture Notes in Computer Science. Springer, 2011.

[76] H. Paulheim and F. Probst. Ontology-enhanced user interfaces: A survey. *Int. J. Semantic Web Inf. Syst.*, 6(2):36–59, 2010.

[77] U. Pletat and V. Narayan. Towards an upper ontology for representing oil & gas enterprises. In *W3C Workshop on Semantic Web in Oil & Gas Industry, Houston, TX, USA, 9–10 December, 2008*, 2008.

[78] V. Presutti and A. Gangemi. Content ontology design patterns as practical building blocks for web ontologies. In Q. Li, S. Spaccapietra, E. S. K. Yu, and A. Olivé, editors, *Conceptual Modeling - ER 2008, 27th International Conference on Conceptual Modeling, Barcelona, Spain, October 20-24, 2008. Proceedings*, volume 5231 of *Lecture Notes in Computer Science*, pages 128–141. Springer, 2008.

[79] E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF. Recommendation, W3C, Jan 2008. `http://www.w3.org/TR/rdf-sparql-query/`.

[80] C. Puleston, B. Parsia, J. Cunningham, and A. L. Rector. Integrating object-oriented and ontological representations: A case study in Java and OWL. In A. P. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. W. Finin, and K. Thirunarayan, editors, *The Semantic Web - ISWC 2008, 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008. Proceedings*, volume 5318 of *Lecture Notes in Computer Science*, pages 130–145. Springer, 2008.

[81] T. Rahmani, D. Oberle, and M. Dahms. An adjustable transformation from OWL to Ecore. In D. C. Petriu, N. Rouquette, and Ø. Haugen, editors, *Model Driven Engineering Languages and Systems - 13th International Conference, MODELS 2010, Oslo, Norway, October 3-8, 2010, Proceedings, Part II*, volume 6395 of *Lecture Notes in Computer Science*, pages 243–257. Springer, 2010.

[82] G. Reggio, M. Cerioli, and E. Astesiano. Towards a rigorous semantics of UML supporting its multiview approach. In H. Hussmann, editor, *Fundamental Approaches to Software Engineering*, volume 2029 of *Lecture Notes in Computer Science*, pages 171–186. Springer Berlin / Heidelberg, 2001.

[83] H. Roggenkemper, R. Ehret, and A. Tönne. Capture accurate solution requirements the first time with exploratory modeling (xM): How to nearly eliminate post-go-live application design failures. *SAP Professional*, 10(1), Jan 2008.

[84] F. Ruiz and J. R. Hilera. Using ontologies in software engineering and technology. In C. Calero, F. Ruiz, and M. Piattini, editors, *Ontologies for Software Engineering and Software Technology*, pages 49–102. Springer Berlin Heidelberg, 2006.

[85] S. Schaffert, A. Gruber, and R. Westenthaler. A semantic wiki for collaborative knowledge formation. In *Proceedings of SE-*

MANTICS 2005, Vienna, Austria, 2006. Trauner Verlag.

[86] E. P. B. Simperl, C. Tempich, and Y. Sure. A cost estimation model for ontology engineering. In I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, editors, *The Semantic Web - ISWC 2006, 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006, Proceedings*, volume 4273 of *Lecture Notes in Computer Science*, pages 625–639. Springer, 2006.

[87] M. Spahn, J. Kleb, S. Grimm, and S. Scheidl. Supporting business intelligence by providing ontology-based end-user information self-service. In *Proceedings of the First International Workshop on Ontology-supported Business Intelligence OBI '08*, pages 10:1–10:12. ACM, 2008.

[88] P. Spyns, R. Meersman, and M. Jarrar. Data modelling versus ontology engineering. *SIGMOD Record*, 31(4):12–17, 2002.

[89] A. Stark, M. Schroll, and J. Hafkesbrink. Die Zukunft des Semantic Web. Think!innowise Trend Report, 2009.

[90] K. Steiner, W. Eßmayr, and R. Wagner. Topic maps - an enabling technology for knowledge management. In A. M. Tjoa and R. Wagner, editors, *12th International Workshop on Database and Expert Systems Applications (DEXA 2001), 3-7 September 2001, Munich, Germany*, pages 472–476. IEEE Computer Society, 2001.

[91] L. Stojanovic, A. Maedche, B. Motik, and N. Stojanovic. User-driven ontology evolution management. In A. Gómez-Pérez and V. R. Benjamins, editors, *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings*, volume 2473 of *Lecture Notes in Computer Science*, pages 285–300. Springer, 2002.

[92] C. Tempich, H. S. Pinto, Y. Sure, and S. Staab. An argumentation ontology for distributed, loosely-controlled and evolving engineering processes of ontologies (DILIGENT). In A. Gómez-Pérez and J. Euzenat, editors, *The Semantic Web: Research and Applications, Second European Semantic Web Conference, ESWC 2005, Heraklion, Crete, Greece, May 29 - June 1, 2005, Proceedings*, volume 3532 of *Lecture Notes in Computer Science*. Springer.

[93] T. Tran, P. Haase, H. Lewen, Ó. Muñoz-García, A. Gómez-Pérez, and R. Studer. Lifecycle-support in architectures for ontology-based information systems. In K. Aberer, K.-S. Choi, N. F. Noy, D. Allemang, K.-I. Lee, L. J. B. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber, and P. Cudré-Mauroux, editors, *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 508–522. Springer, 2007.

[94] M. Uschold, J. Bateman, M. Bennett, R. Brooks, M. Davis, A. Dima, M. Gruninger, N. Guarino, E. Lucier, L. Obrst, S. Ray, T. Schneider, J. Sowa, R. Sriram, M. West, and P. Yim. Making the case for ontology. *Appl. Ontol.*, 6(4):377–385, Dec. 2011.

[95] M. Uschold and M. Gruninger. Ontologies and semantics for seamless connectivity. *SIGMOD Rec.*, 33:58–64, Dec 2004.

[96] F. Verhelst, F. Myren, P. Rylandsholm, I. Svensson, A. Waaler, T. Skramstad, J. Ornæs, B. Tvedt, and J. Høydal. Digital Platform for the Next Generation IO: A Prerequisite for the High North. In *SPE Intelligent Energy Conference and Exhibition*, 2010.

[97] H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner. Ontology-based integration of information - a survey of existing approaches. In *Proceedings of IJCAI-01 Workshop: Ontologies and Information Sharing, Seattle, WA, 2001*, pages 108–117, 2001.

[98] D. West. What semantic technology means to application development professionals. Forrester Research Report, Oct 2009.

[99] A. White. Semantic web moving ever close to the 'semantic enterprise'? Gartner Blog Network, `http://blogs.gartner.com/andrew_white/2009/04/30`, Apr 2009.