FFI-rapport 2010/00015

Recommended application areas for semantic technologies

Bjørn Jervell Hansen, Jonas Halvorsen, Svein Ivar Kristiansen, Rolf Rasmussen,
Marianne Rustad and Geir Sletten

Norwegian Defence Research Establishment (FFI)

12 February 2010

FFI-rapport 2010/00015

1085

P: ISBN 978-82-464-1708-0
E: ISBN 978-82-464-1709-7

Keywords

Semantisk web

Ontologi (databehandling)

Kunnskapsrepresentasjon

Tjenesteorientert arkitektur

Informasjonsintegrasjon

Approved by

Rolf Rasmussen	Project Manager
Eli Winjum	Director of Research
Vidar S. Andersen	Director

English summary

This report describes the final results from the FFI-project Semantini (Semantic Services in the Information Infrastructure). The project has explored semantic technologies in the context of their potential to add value to the information infrastructure, which in turn is a major enabler for network-based defence.

Semantic technologies are information technologies that utilise the meaning (semantics) of the information in a domain of interest in order to contribute to more intelligent, adaptive, and flexible software solutions.

In the report, four subjects considered to be of special interest for the usage of semantic technologies in the military domain are covered in more detail:

- **Reasoning and Rules**, which covers the capability of automatically inferring information on the basis of formal models,
- **Semantic Web Services**, where shortcomings of Web Services, a common way of implementing service-oriented architectures, is mitigated by semantic technologies,
- **Information Integration**, where the potential of semantic technologies regarding integrating information from heterogeneous information sources is explored, and
- **Distributed Information**, where the handling of information on the World Wide Web is related to the expected need to handle distributed information in the information infrastructure.

Additionally, three experiments focusing on promising military use of semantic technologies are presented: One experiment regarding information analysis, and two experiments concerning the use of semantic technologies to add features to Web Services.

As a conclusion of the report, Semantini points to the following application areas as interesting with regards to future use of semantic technologies in the information infrastructure:

- decision support systems, including intelligence analysis solutions
- information integration solutions
- service infrastructures

The value of semantic technologies lies partly in their expected future widespread use. From a larger user community will hopefully follow powerful tools and methods of industrial strength. From the perspective of the Norwegian Armed Forces, insightful awareness and consistent skill-building is a recommendable approach to semantic technologies.

Sammendrag

Denne rapporten beskriver sluttresultatene fra FFI-prosjektet Semantini (Semantiske tjenester i INI). Prosjektet har utforsket semantiske teknologier og potensialet disse teknologiene har til å bidra i informasjonsinfrastrukturen (INI). INI er en viktig muliggjører for nettverksbasert forsvar (NBF).

Semantiske teknologier er informasjonsteknologier som gjør nytte av meningen (semantikken) i informasjonen i et domene for å bidra til mer intelligente, tilpasningsdyktige og fleksible softwareløsninger.

Rapporten går nærmere inn på fire emner som antas å være spesielt interessante med tanke på bruk av semantiske teknologier i det militære domenet:

- **Resonnering og regler**, som dekker egenskapene ved semantiske teknologier som gjør at man kan utlede informasjon på grunnlag av formelle modeller,
- **Semantiske webtjenester**, der svakheter ved webtjenester, en vanlig måte å implementere tjenesteorienterte arkitekturer på, utbedres ved hjelp av semantiske teknologier,
- **Informasjonsintegrasjon**, der bruk av semantiske teknologier for å integrere informasjon fra heterogene informasjonskilder utforskes og
- **Distribuert informasjon**, der informasjonshåndteringen på World Wide Web settes i sammengeng med det forventede behovet av å håndtere distribuert informasjon i INI.

I tillegg presenteres tre eksperimenter som fokuserte på militær bruk av semantiske teknologier: Ett eksperiment med tema informasjonsanalyse og to eksperimenter der semantiske teknologier brukes til å forbedre webtjenester.

Som konklusjon på rapporten, peker Semantini på tre interessante framtidige bruksområder for semantiske teknologier i INI:

- beslutningsstøtte, inkludert etterretning og analyse
- informasjonsintegrasjonsløsninger
- tjenesteinfrastrukturer

Verdien av semantiske teknologier ligger delvis i at de forventes å få en utstrakt utbredelse i fremtiden. Med et stort antall brukere, vil det forhåpentligvis bli utviklet gode verktøy og metoder. Sett ut fra Forsvarets perspektiv, vil vi anbefale å vise oppmerksomhet og sørge for innsikt i dette fagområdet. Det er også viktig å sikre kompetansebygging for å kunne utnytte de fremtidige mulighetene som ligger i semantiske teknologier.

Contents

1 Introduction

This report describes the final results from the FFI-project Semantini (Semantic Services in the Information Infrastructure). The project has explored semantic technologies in the context of their potential to add value to the information infrastructure, which in turn is a major enabler for network-based defence (NBD). This report aims to recommend application areas where this family of technologies can add value to the information systems of the Norwegian Armed Forces.

Semantic technologies are information technologies that utilise the meaning (semantics) of the information in a domain of interest in order to contribute to more intelligent, adaptive, and flexible software solutions.

The project has focused on semantic technologies belonging to the Semantic Web technology stack, see Section 2.5. It should be noted that there exist semantic technologies outside this stack, which may be considered in the cases where the Semantic Web standards are missing, for specific optimisation purposes, etc.

It goes without saying that our primary focus has been the military domain. The border between military and civilian use of these technologies is not very distinct, thus the results given here should also be applicable outside the military domain.

The structure of this document is important for the reader to be aware of, as some parts are more technically focused than others. Section 2 provides some background information that is required for the understanding of the technical part that follows. This technical part consists of four sections that describe and discuss the main technical areas that underpin the overall recommendations: Section 3 about reasoning and rules, Section 4 about semantic web services, Section 5 about information integration and Section 6 about handling distributed information. Each section discusses relevant areas of recommended use and further work.

After this technical part there are two sections dedicated to experimental applications that have been developed in order to explore the potential of semantic technologies: Section 7 describes our experimental decision support demonstrator, while Section 8 describes performed work on how semantic technologies can add value to an infrastructure of dynamic services.

Section 9 considers other interesting areas that may benefit from semantic technologies: Information management, system architecture and other approaches to modelling, NATO RTO efforts to improve semantic interoperability, and some considerations on how to handle unstructured information (text, image, audio etc.).

In Section 10 we discuss our overall recommendations regarding how semantic technologies may be used to add value to the military community, and conclude the report.

2 Background

This section introduces background terms, concepts, and ideas relevant for the report. More specifically, we briefly introduce the topics of knowledge representation and knowledge-based systems, ontologies, semantic technologies, the Semantic Web and the Semantic Web technology stack, as well as the Open World Assumption.

More details on these topics, except the knowledge-related ones, can also be found in Hansen et al. (2007).

2.1 Knowledge Representation and Knowledge-Based Systems

Knowledge representation (KR) involves representing knowledge in the form of structured symbols (typically turning tacit knowledge into explicit knowledge). The term "‘knowledge"’ is a vague concept that most have an intuition of what means, yet is difficult or impossible to define (see Brachman & Levesque (2004, p. 2), Sowa (2000, p.1) and Schreiber et al. (1999, p. 3)). Many have tried, but there is no commonly agreed definition. For the rest of this report, when we refer to knowledge we will refer to the subset of knowledge that can be formally expressed in declarative sentences (i.e. knowledge that can be explicitly described/told to someone). This involves statements of the type "‘John is a Man"’ as well as "‘If someone is a Man, and that someone has at least one Child, then that someone is a Parent"’. The structure of the knowledge is what is of interest, as it allows for computer manipulation in terms of automated reasoning in a way that is consistent with logical human reasoning.

Knowledge-based systems utilise knowledge represented in structured symbols in order to logically derive new conclusions/knowledge from it. In these systems, both the information and the problem-solving knowledge is separated from the application code. General reasoning algorithms, without application-specific knowledge built in (in contrast to classical applications where problem-solving knowledge is built into the algorithms), utilise the domain and inference knowledge in order to solve the problems at hand (generate new information/knowledge). As a result, knowledge-based systems can be seen as acting in a more intelligent manner than traditional software systems where problem-solving is hard-coded.

Typically, knowledge-based systems are utilised in order to automate knowledge-intensive tasks. I.e. tasks that involve utilising knowledge about a domain and involves human reasoning (see Figure 2.1 for knowledge-intensive task types). Experiments have shown that systematic patterns of error frequently occur in human reasoning, even for relatively simple logical operations (i.e. errors in problem solving, see Schreiber et al. (1999, p. 191)). Knowledge-based systems enhance organisational effectiveness by facilitating faster decision-making, increased productivity and increased quality of decision-making (Schreiber et al. 1999, p. 6).

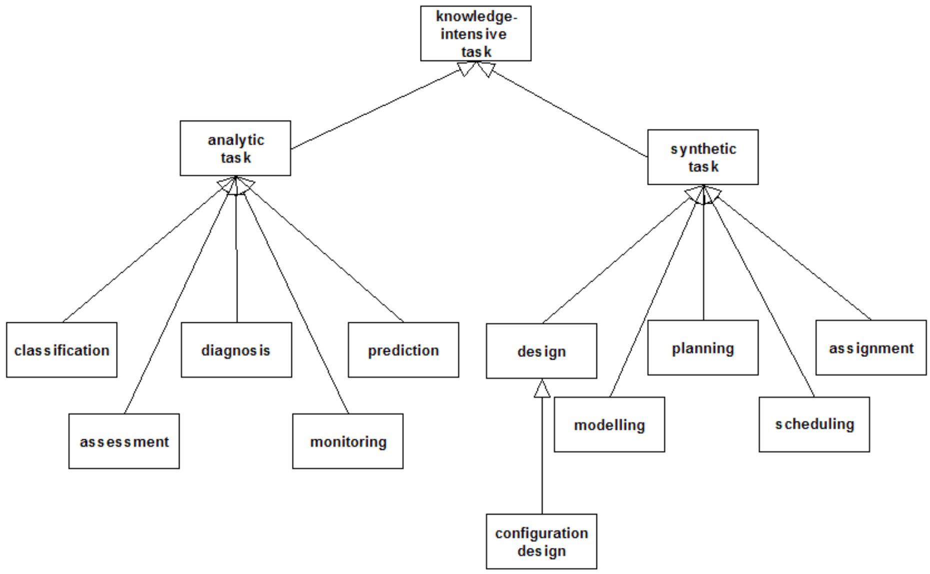


Figure 2.1 Type hierarchy of knowledge-intensive tasks (Schreiber et al. 1999, p. 125)

2.2 Semantic Technologies

Semantic technologies are information technologies utilising formal models that define the vocabulary and problem-solving knowledge of the information domain at hand. This approach is expected to make computers able to perform certain knowledge-intensive tasks and in general contribute to more intelligent, adaptive and flexible software.

Central to semantic technologies are thus formal models, where domain information is explicitly captured and defined. As a result, the semantics (meaning) of the domain knowledge is separated from the data and the application code, and put into the formal models. These models are dynamic, and can be exchanged at runtime. Upon changes to the models the applications change behaviour accordingly. Furthermore, as the models are formal and explicit, they are amenable for computer processing in terms of automatically inferring meaningful conclusions from datasets in accordance to the defined semantics.

Another benefit of semantic technologies is their ability to effectively utilise large amounts of heterogeneous datasets. This, together with the ability to deduce implicit knowledge, provide means to automate certain knowledge-intensive tasks in order to assist human users. Examples of knowledge-intensive tasks include classification, monitoring, prediction, and planning, see Figure 2.1.

2.3 Ontologies

In computer science, an ontology is a formal explicit model of the concepts and relations in a domain. With an ontology, the assumptions in the domain in question are made explicit, making them computer processable. Ontologies also facilitate reuse of domain knowledge.

Ontologies are the core components in any system utilising semantic technologies. Making the necessary ontologies is a modelling task, and represents maybe the biggest challenge in order to make semantic technologies work. Ontologies being formal and computer processable, make them amenable for automated reasoning. Furthermore, ontologies are inter-linkable and well suited to be developed in an incremental fashion.

Ontologies are often arranged into upper ontologies, general domain ontologies, and application ontologies according to their generality.

The upper ontologies define the most general concepts, like for example *PhysicalObject*, *HumanBeing*, and *TemporalObject*. There exist several upper ontologies, the most commonly used being Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) (Library for Applied Ontology 2009), OpenCyc (Cycorp 2009), Suggested Upper Merged Ontology (SUMO) (Niles & Pease 2001), Upper Mapping and Binding Exchange Layer (UMBEL) (Bergman & Giasson 2009), and Basic Formal Ontology (BFO) (Smith & Grenon 2009).

General domain ontologies define concepts in general domains that are common to many applications, examples including the time domain, geography domain, and the C2 domain.

The application ontologies are the most specific ontologies, being designed to serve a specific application or a family of applications. Application ontologies have to be created manually or semi-automatically, but should reuse existing ontologies, both upper ontologies, general domain ontologies, and other application ontologies, when appropriate.

Note that here is no requirement to use all the three types of ontologies when utilising semantic technologies. In particular, it will often be the case that the use of an upper ontology is not necessary.

2.4 The Semantic Web

The Semantic Web is a vision originally developed by Sir Tim Berners-Lee, the inventor of the World Wide Web. The vision was presented to the world in Berners-Lee et al. (2001).

In essence, the vision of the Semantic Web describes an enhancement to the current World Wide Web (WWW), making the contents of the Web accessible to computers as well as to humans.

The Semantic Web is often referred to as representing a shift from todays Web of Documents, where the links are between documents, to a Web of Data (WoD) where the links are between information elements. This is illustrated in Figure 2.2.

Although the Semantic Web vision focused on the World Wide Web, the associated technologies themselves have also shown to be useful in closed internal enterprise systems.

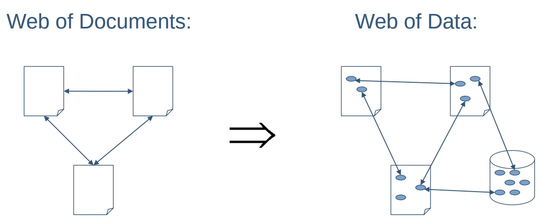


Figure 2.2 From a Web of Documents to a Web of Data

2.5 The Semantic Web Technology Stack

When testing and exploring semantic technologies, the focus of Semantini has been on utilising the recommended specifications developed by the World Wide Web Consortium (W3C) in connection with their effort to realise the Semantic Web. We consider this family of semantic technologies an important toolkit when implementing solutions utilising semantic technologies. Figure 2.3 shows the Semantic Web Layer Cake, which summarises the technologies and standards needed to implement the Semantic Web. In the following, what we consider the most important Semantic Web standards are presented.

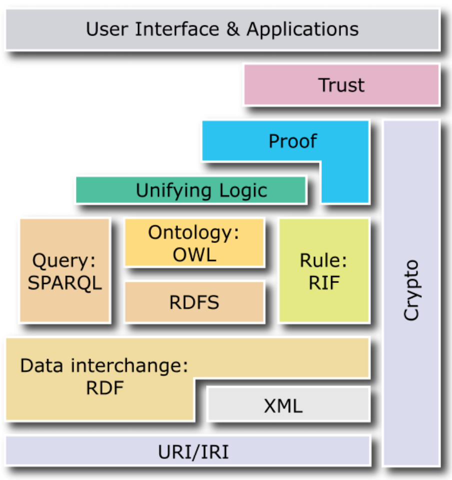


*Figure 2.3 Semantic Web technologies and standards (W3C 2007*b*)*

2.5.1 Uniform Resource Identifier - URI

A Uniform Resource Identifier (URI) is a string providing unique identification for a web resource or a relation between resources. Resources can represent anything, e.g. it can even represent a town or a person.

A URI can be a locator and/or a name:

- A Uniform Resource Locator (URL) is a URI identifying the *location* of a resource and how to access it. E.g. `http://www.ffi.no`

- A Uniform Resource Name (URN) is a URI identifying a resource by *name* in a namespace. It uniquely identifies a resource without having to tell anything about location. However, the namespace is typically a web address where the resource is described.

2.5.2 Resource Description Framework - RDF

The Resource Description Framework (RDF) (W3C 2004*b*) is a formal language for representing structured information in a graph. An information set represented in RDF consists of *triples* - subject-predicate-object tuples. Subjects are information items (identified by a URI), objects can either be an information item or a literal value, while predicates are the relations between the subjects and the objects. A set of triples constitutes a graph, as illustrated in Figure 2.4.

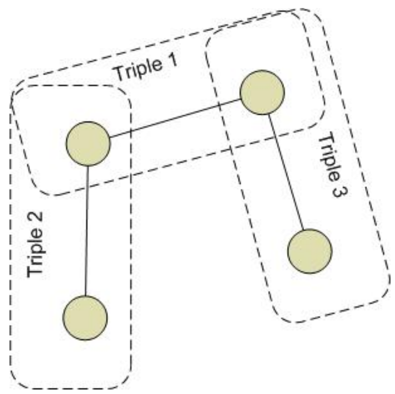


Figure 2.4 An RDF graph consisting of three RDF triples

RDF has several serialisation formats, the most widely used being RDF/XML (W3C 2004*c*), Notation3 (Berners-Lee 2000), and Turtle (Beckett & Berners-Lee 2008).

2.5.3 Web Ontology Language - OWL

The Web Ontology Language (OWL) (W3C 2000) is a formally defined language for representing ontologies on the Web. Furthermore, it is based on description logics (DL) (Nardi & Brachman 2003), a family of logic-based KR formalisms with well-understood computational properties (there exists complete and tractable algorithms). OWL allows for modelling ontologies with definitions of and restrictions on classes, roles, and individuals, and allows the derivation of implicit knowledge through the use of a reasoner. It is a W3C recommended standard with substantial uptake and popularity. Furthermore, it has taken a reasonable balance between expressivity and efficiency with regards to reasoning (favourable scalability properties) (Hitzler et al. 2009, pp. 111-115).

Although, technically, the OWL recommendation specifies two alternative semantics (OWL-Full and OWL-Lite) in addition to OWL-DL (OWL with DL semantics), these two are very rarely used and thus we limit us to exclusively talk about OWL-DL unless specified otherwise [1].

2.5.4 SPARQL Protocol and RDF Query Language - SPARQL

SPARQL Protocol and RDF Query Language (SPARQL) (W3C 2008) is the W3C query language designed to allow querying on RDF graphs, much like SQL is used to query relational databases. Using SPARQL, a user specifies a graph pattern which is matched with the RDF graph in question.

SPARQL includes a capability to specify remote RDF graphs for querying (the FROM clause (W3C 2008, Section 8.2)). This makes SPARQL interesting as a tool to perform federated querying, i.e. the issuing of one query to a number of sources and receiving a single answer.

2.6 The Open World Assumption

Reasoning over OWL ontologies commits to the Open World Assumption (OWA), which means that it is implicitly assumed that a knowledge base may always be incomplete (Hitzler et al. 2009). One example is finding an answer to the question *is Karen a Swedish citizen?*, based on the asserted knowledge that *Karen is a Norwegian citizen*. Reasoning under the Closed World Assumption (CWA) would conclude that Karen is not Swedish. However, under the OWA, a reasoner would not be able to conclude either true or false as there is no knowledge that asserts that a person can not be the citizen of two countries (Karen could for example have dual citizenship).

The open world assumption is by no means an unfamiliar concept in the operational military domain. It is a common feature in military systems that all *available* information is shown, and this does not always mean a *complete* situational picture. If new information is supplied during a military operation, it will be added to the current picture, just like new information on the Semantic Web can be added seamlessly to the existing information graph.

There are methods that can be used to *close the world* in systems where that is needed. By closing the world is meant forcing the system to regard it's information set as a complete. This is common practice in traditional databases, leading to facts not explicitly present in the database considered to be false.

3 Reasoning and Rules

Automated reasoning over formal models is not a new paradigm in computing. This approach, based on deductive, logic-based methods, focus upon users telling the system what it needs to know (i.e. description of domain vocabulary, domain facts, and problem-solving know-how), for so letting

[1] OWL-Lite is a subset of OWL-DL, while OWL-DL is a subset of OWL-Full. However, OWL-Full does not have decidable algorithms, due to it allowing unrestricted use of language elements. OWL-DL restricts usage of certain elements of the OWL language in order to obtain decidability.

the computer find an answer using deductive inference. Expert systems of the 80's and 90's such as Mycin, used by doctors to diagnose illnesses, and XCON, a computer hardware configuration system, showed that systems that reason automatically can be feasible and of real practical use (Brachman & Levesque 2004, pp. 130-132).

The Artificial Intelligence (AI) winter of the late 80's put an abrupt halt in funding and interest in expert systems, as it became clear that it was not possible to match the expectations that were initially promised (Russell & Norvig 2003, p. 24). The main culprits were that of intractability (the decision procedures for even simple logics often fall within the NP-complete family of algorithms, which means that it might take a vast amount of time before an answer is deduced) (Brachman & Levesque 2004, p. 69) and the fundamental issue that first-order logic (FOL) in general is incomplete (no algorithm exists that can deduce all the correct answers that exist.[2]) (Russell & Norvig 2003, p. 302).

Due to recent developments within the field, there is renewed interest in logic-based methods and automated reasoning. This is likely due to a variety of reasons.

First of all, much work has been done on the need to balance expressivity of a formal language versus complexity of reasoning. As a result, subsets of FOL with tractable[3] decision procedures have been defined that are shown to be useful in practice (Russell & Norvig 2003, p. 353). Furthermore, new and optimised algorithms for these subsets have been devised, such as improvements to tableaux algorithms (e.g. hypertableaux) (Motik et al. 2009) and the introduction of instance-based methods (Baumgartner & Thorstensen 2010).

Another aspect that likely has contributed to the renewed interest in logic-based methods is the creation of the World Wide Web. Tim Berners-Lee, the father of the Web, wrote a seminal paper in 2001 that introduced the concept of the Semantic Web, where the data online is to shift from being intended purely for human consumption to that of being intended for computer consumption as well. This new Web, the Semantic Web, differs from the current Web in that it is a Web of data that computers could parse and reason over, automating knowledge based tasks that previously required human processing and action (see also Section 2.4).

The vision of the Semantic Web is based upon established computer science topics such as knowledge representation, ontologies, automated reasoning and intelligent agents, and has during the last years seen substantial growth of interest in academia as well as among large commercial vendors, both civilian (Oracle, IBM, HP) and military (Raytheon/BBN, Lockheed Martin, Northrop Grumman). The three commercial organisations in the military domain (noted above) together comprised the main sponsors of the International Semantic Web Conference (ISWC) in Washington, DC, in 2009 [4], indicating that the field is of high interest in a military setting.

[2] Due to Gödel's incompleteness theorem

[3] Returns an answer within reasonable time.

[4] `http://iswc2009.semanticweb.org/` - ISWC is the largest and most prestigious of the Semantic Web conferences.

3.1 Reasoning over Formal Ontologies

An ontology, in its most basic form, defines objects that exist in the world and the relations between them. In computer science, the term has a more specific meaning in that it refers to a formal model that models (a part of) the world in a way that allows computers to reason over. Furthermore, an ontology is the statement of a logical theory of a domain, defining the axioms that constrain interpretations and usage of objects (Walton 2007, pp. 6-7).

Formal ontology languages are given clearly defined semantics, usually in terms of a well-known logic, in order to dictate allowed interpretations and conclusions that can be drawn. In our case, OWL, the Web Ontology language (see Section 2.5.3), is based on Description Logics which is a family of logics that are syntactic variants of subsets of FOL that have complete and tractable decision procedures (which means that all possible correct inferences are found within reasonable time).

An OWL knowledge base[5] (KB) can conceptually be divided into two:

TBox Terminological knowledge - terms and vocabularies in a domain (i.e. defined classes and properties) and what they mean. Roughly like a schema in databases

ABox Assertional knowledge - assertions about instances (i.e. what types the instances are, what relations they have to others, values). Roughly like the actual instance data in a database table.

$$
\begin{aligned}
\mathsf{Woman} &\equiv \mathsf{Person} \sqcap \mathsf{Female} \\
\mathsf{Man} &\equiv \mathsf{Person} \sqcap \neg\mathsf{Female} \\
\mathsf{Talk} &\equiv \exists \mathsf{topic}.\top \\
\mathsf{Speaker} &\equiv \mathsf{Person} \sqcap \exists \mathsf{gives}.\mathsf{Talk} \\
\mathsf{Participant} &\equiv \mathsf{Person} \sqcap \exists \mathsf{attends}.\mathsf{Talk} \\
\mathsf{BusySpeaker} &\equiv \mathsf{Speaker} \sqcap (\geq 3\, \mathsf{gives}.\mathsf{Talk}) \\
\mathsf{BadSpeaker} &\equiv \mathsf{Speaker} \sqcap \forall \mathsf{gives}.(\forall \mathsf{attends}^{-1}.(\mathsf{Bored} \sqcup \mathsf{Sleeping}))
\end{aligned}
$$

Figure 3.1 TBox: OWL concept definitions (Tessaris et al. 2009, p. 9)

OWL reasoning is mainly focused on reasoning about terminological knowledge. An example of a TBox (with concept descriptions) regarding conference domain knowledge is shown in Figure 3.1. Typical OWL reasoning tasks are consistency checking (checking that the ontology does not contradict itself, e.g. define impossible concepts), calculating class and property hierarchies (sub- and super), class satisfiability (if it is actually possible for the class to have instances, based on the class definition), and inferring and checking class membership.

[5] The collection of all knowledge/sentences the system contains.

OWL is an example of the balance one has to take between expressivity of the formal language versus complexity of reasoning. The more limited the language, the easier it is to reason over. However, languages with low expressivity is generally of limited use as only the simplest of knowledge can be expressed, hence not much interesting reasoning can be done.

3.2 Combination of Ontologies and Rules

OWL ontologies alone are sometimes not enough to properly capture the model of a domain. For example, OWL does not generally allow for arbitrary chains of relations such as the rule *"if x is a Man, and has a brother relation to Y, and Y has a child Z, then X is uncle of Z."* [6]. As a result, increasing focus has been on the combination of ontologies and rules[7] (ONTORULE 2009).

Both DL (which OWL is just a syntactic variant of) and logical rules are logic-based KR formalisms, based on subsets of FOL. However, they differ somewhat in approach to representing structured knowledge. DL is mainly aimed toward representing and reasoning about ontological (terminological) knowledge, while rules are more general in terms of intended usage (any model than can be axiomatised as facts and rules).

Logical rules come from the logic programming community, where the aim is to develop applications in a logical, declarative way (in contrast to the traditional procedural programming method), with a syntax as close as possible to the horn-clause fragment of FOL. The rule-sets that are defined constitute the programs, thus referred to as logic programs (LP) . The horn-like rules of logic programs take the form of IF. . . THEN sentences (e.g. in the form $A \wedge B \rightarrow C$, which reads as *if A and B are true, then C is also true*), and are often interpreted with somewhat different semantics than with FOL semantics in terms of committing to the Closed World Assumption (see Subsection 2.6) due to practical reasons (ONTORULE 2009).

DL and LP rules share notable overlap in what the two formalisms can represent, so that certain parts of a DL ontology can be represented as rules and vice versa. Furthermore, they are both syntactic variants of FOL formalisms, meaning that statements in both formalisms can be rewritten into FOL statements. Yet, there are parts of FOL that can be represented in one of these KR representations and not the other (and vice versa). Additionally, there are large parts of FOL that can not be expressed in either of the two. See Figure 3.2 for illustration.

These two KR representation formalisms complement each other in terms of DL's strength in expressing ontology terms and concepts while rules are good at modelling complicated roles and relations. Typical envisioned usages of logical rules are for constraint checking, query answering, and for generally representing arbitrary chains not possible to capture in DL (which is the main motivation for integrating the two).

As noted, there are certain things that can not be expressed in DL, but can be expressed with rules.

[6]OWL2 property chains address some situations, yet imposes restrictions in terms of disallowing property chain hierarchy as well as referencing data properties and referencing class membership

[7]In the form of horn-clauses

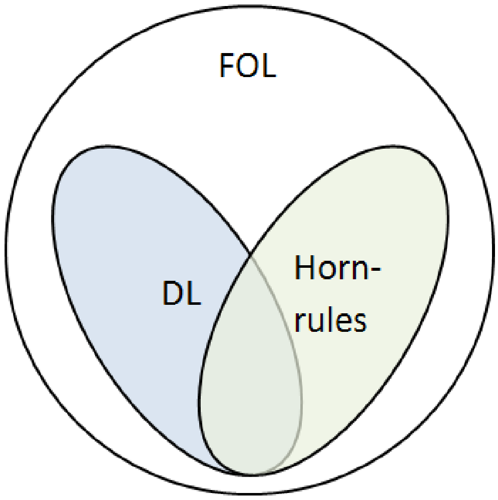


Figure 3.2 Intersection and difference between Ontologies and Rules

Likewise, there are things that can not be expressed in rules that can be expressed in DL. One example is definitions of the pattern *"every father X is parent to some Y"*, which LP rules are not capable of expressing without explicitly naming an instance for Y. Contrary, DL can easily express such statements, and a DL reasoner will be able to infer the existence of individual Y (without a name) and can use the existence of it for further reasoning (also, the individual might later be named as more information is discovered).

Note that there is an issue with naïvely combining these two knowledge representation formalisms in practice. The complicating issue is that they have different semantics in that they take different stances to the Open/Close World Assumption (see Section 2.6). DL commits to the OWA, meaning that the absence of information means that it cannot conclude anything from it, while rules (in general, for efficiency reasons) commits to the CWA meaning that the absence of information can conclude that it is not true. In practice, this means that DL is more conservative in what can be inferred: logical rules takes a common sense reasoning approach, where conclusions can be drawn from incomplete information. DL reasoning incurs that something can be concluded only if it can be fully proven. Due to the potential benefits, there is currently substantial work being done on deciding the best approach to combining these two KR formalisms. Several possible solutions exist, amongst them are controlling the procedural flow between the two worlds by coupling, e.g. first DL-reasoning then feed into LP-reasoning, creating hybrid semantics to fuse the two or creating new alternative semantics for the combined whole (ONTORULE 2009).

Bringing rules into the OWA naïvely and using them in an unrestricted form, can lead to incompleteness as the rules can allow for infinite chains by means of recursion. This is due to the difference in reasoning mechanisms traditionally used for DL and LP. As a result, if one is to utilise current rule-reasoning algorithms in the OWA-setting, then rules must be restricted to prevent uncontrolled recursion e.g. by restricting rule variables to only match named individuals in the KB (this property is referred to as being 'safe'). Although complicated, the combination of ontologies and rules seems to hit a sweet spot of potential in a wide variety of systems, many of them within the military domain, which justifies focus on this area.

3.3 Potentials and Challenges

It is envisioned that declarative approaches that involve reasoning (over ontologies and rules) will make it possible to develop more dynamic systems, reduce the development time and make it easier to verify the final product. From the functionality aspect, declarative techniques will make it possible to develop systems that are more autonomous, acting more intelligently by automating knowledge-intensive tasks previously reserved for humans. This is expected to aid in reducing human error and information overload.

OWL and rules make it possible to formalise domain knowledge in an unambiguous way. In contrast to UML, IDEF-x, etc., OWL has a clear, well-defined semantics that prevents misinterpretation. This is also in contrast to textual descriptions of meaning, which is wildly ambiguous. Furthermore, having a declarative model in the form of OWL+Rules gives you an executable model amenable to automate knowledge-based processes such as terminological (classification: class-hierarchy and individuals) and assertion-based (infer relations due to class descriptions as well as property-chains, rules etc.) inference.

However, in order for these technologies to be successfully utilised, there are a couple of issues (both organisational and technical) that need to be addressed.

The largest threat is probably that of unrealistic expectations. The Semantic Web standards and technologies have heritage from the AI field, which unfortunately comes with baggage in regards to the AI-winter which many people still have in memory. Thus it is already treated with certain skepticism. However, as noted earlier in this chapter, recent developments in technology and theory, the realistic understanding of the field one has today makes it a different case than before.

Furthermore, in a military setting, actions and decisions taken might have severe consequences. Considering the ethical and political factors, not everything is a candidate for automation. Additionally, as noted earlier, not everything can be axiomatised in a proper manner thus cannot be automated. As a result, it is still a need for a human-in-the-loop in critical military systems.

Another non-technical issue is that of developing, updating and governing ontologies and rules. This issue has two aspects. First is that of the availability of modelling experts with know-how about OWL and rule modelling. This modelling paradigm requires a different kind of expertise than that of traditional data modelling, and the availability of experts with required know-how is currently limited. Another aspect, which is important in an organisational setting, is that of governing who is responsible for developing and updating ontologies, best-practices for developing them etc. However, provenance issues are not as important in the setting of the WWW, where there is no strict governing in contrast to in an organisational setting.

Finally, there are still theoretical and technical issues that need to be adequately addressed in order for these technologies to be used in practice. First of all is that of continuing work on the issue of tractability vs. expressivity, obtaining more suitable KR languages and more effective algorithms. Another aspect which has recently received increased focus is non-standard reasoning; that of being able to retract conclusions as new information is received , allowing for defaults that might have exceptions as well as probabilistic reasoning. Addressing these questions will facilitate a range of new possibilities in terms of applications and usage of value in real-life.

3.4 Applications within the Military Domain

There is a vast range of potential application areas for ontologies and rules within the military domain. This technology is of practical use for knowledge-intensive tasks that can be formally represented, e.g. most situations where higher-level information integration and fusion occurs (see Section 5).

Furthermore, we see clear potential use in situation awareness systems, decision support systems, early warning systems, middleware and SOA (semantic interoperability, discovery and orchestration of services, see Section 4), as well as intelligence analysis support systems.

We also see a difference in priority of expressivity vs. tractability depending on what level the system is to work on (strategic/operational/tactical) and what one can expect of hardware resources available. Early warning systems and operational & in-field decision support systems require time-sensitive reasoning over that of increased expressivity, as timing is of great importance. Furthermore, such systems will likely be limited in computational resources which further reduces the potential applicability of complex reasoning at this level.

Contrary to this, for intelligence analysis tools and strategic/tactical decision support systems, timing will likely not be as critical as for the abovementioned systems. Additionally, as these tools will most likely be used at command centres, the hardware resources available are less limited. Thus it is more likely that these systems will prioritise that of increased expressivity in order to be able to capture more of the domain model and perform more advanced reasoning compared to systems at the lower level.

3.4.1 Existing Implementations in the Military Domain

As noted earlier in this section, there are many situations where ontologies and rules can be used in military systems. We now exemplify some possible usages with actual implemented systems.

The MITRE corporation developed in 2007 a prototype C2 system based on ontologies and rules (Stoutenburg et al. 2007). They determined that

> *Increasingly, Command and Control (C2) systems require the ability to respond to rapidly changing environments. C2 systems must be agile, able to integrate new sources*

of information rapidly for enhanced situational awareness and response to real-time events. (Stoutenburg et al. 2007)

Their system aimed at utilising ontologies and rules to address dynamic mission needs. The initial aim of the system was to provide alerts and recommendations to a user. However, during the experimentation they found from experience that ontologies and rules are also very suitable tools for rapid enterprise integration, being able to integrate new heterogeneous data sources *"within hours, instead of weeks or months, using traditional software development methods"* (Stoutenburg et al. 2007). The system was demonstrated at the Joint Expeditionary Force Experiment in 2008.

Another semantic technology prototype in the military domain is described in Baader et al. (2009). The prototype described, called Situational Awareness by Inference and Logic (SAIL), is a generic situation awareness (SA) application framework, based on formal logic and automated reasoning as its core. Their prototype aims to address higher-level information fusion to *"integrate domain specific knowledge and automatically draw conclusions that would otherwise remain hidden or would have to be drawn by a human operator."*(Baader et al. 2009). The developers of the system work in collaboration with Defence Science and Technology Organisation (DSTO) Australia, and a running prototype of it used as a system for SA for the air domain in a NATO scenario is described in the paper. The system addresses data aggregation (perception), semantic analysis (comprehension) and alert generation (projection) (Baader et al. 2009). These three levels are all addressed with declarative techniques.

One of Semantini's latest experiments focused on the development of an intelligence analysis tool that utilise ontologies and rules. The system, called Automated Reasoning Based Intelligence Tool (ARBIT), performed partial rule matching (in order to aid in drilling for relevant data) in addition to standard DL and rule reasoning. The system (and the scenario it was demonstrated in context of) is described further in Section 7.

3.5 Future Research

One should note that the experimental system mentioned in the last section is intended for demonstration purposes only, with made-up data and a limited amount of ontologies and rules. Thus, testing the system in a more realistic setting, with realistic and large amount of data[8], ontologies and rules, would be interesting in order to be able to determine current suitability of the technology for real use in the field. Determining what level of expressivity is needed for different domains/subject areas depending on reasoning needs would be of practical use should the technology be used in real-life systems.

Related to this, it would be interesting to explore the balance of expressivity and reasoning power for different domains/subject areas, and at different levels from the field to the HQ. In a time-sensitivity context, with limited hardware, one could possibly consider using less expressive subsets of OWL

[8] Existing legacy data in databases can easily be exposed in a form suitable for use by semantic technologies.

in combination with incomplete reasoning. Here the theory is that limited, basic inferences can still be useful. Likewise, at a higher level, it would be interesting to increase expressivity of the logics and develop more detailed and powerful models, and reason over these using high-performance and distributed computing. Here, one could explore using highly expressive ontologies and advanced rules as well as looking into introducing uncertainty. We believe that this can automate increasingly complex knowledge-intensive tasks.

Another interesting aspect for further study is that of non-standard reasoning in terms of non-monotonic logics (that of being able to invalidate/retract conclusions), meta-rules & rule hierarchies, as well as default logics (defaults with exceptions e.g. all birds fly, except penguins and ostriches). These extensions should open up a lot more possible application areas in the military domain such as the ability to do "common-sense" reasoning as well as automate even more types of knowledge-intensive tasks.

4 Semantic Web Services

NATO network-enabled capability (NNEC) is a NATO program aiming to transform the effectiveness of the alliance through an alliance-wide networking and information infrastructure (NII).

Interoperability is a main concern when attempting to fully realise NNEC (NC3A 2005). The NNEC vision implies an information infrastructure that supports prioritised access to information, services, and resources from the strategic level, down to the tactical level where communication resources usually are scarce. Web Services (WS) technology has been identified as a key enabling technology for NNEC. Using this technology, all capabilities in a network can be exposed as services that can in principle be discovered and used across heterogeneous networks.

Traditional WS discovery techniques have limitations that makes them ill-suited for the use in a network-centric battlefield. In particular they do not accommodate the distributed nature of military tactical networks, nor do they address dynamic selection and orchestration in a satisfiable way. The use of semantic technologies should be able to address these issues and achieve system interoperability. More specifically, semantic technologies should address the interoperability challenges related to service description and selection.

4.1 SOA and Semantic Web Services

Service-oriented architecture (SOA) is an architectural paradigm enabling heterogeneous systems to cooperate in distributed environments through standard protocols and interfaces. SOA promises a more dynamic and automatic environment where services are viewed as distributed components ready to be discovered, invoked, and possibly combined with other services. A popular technology for implementing SOA is that of WS. Even though traditional WS has many qualities like composability, discoverability, loose coupling, and reusability, there is room for improvement. In the WS setting much of the service activities are preprogrammed and static, based on syntax and human reasoning about services, limiting the promised dynamic and automatic properties of a SOA. Semantic Web Services (SWS) aims to remedy these limitations as semantic technologies enable computers to reason about services, eliminating (some of) the need for human intervention and design time decisions. A layer of explicitly defined meaning is added to the WS descriptions allowing computer reasoning to automate services selection, orchestration and invocation.

4.1.1 Traditional Web Services

WS are based on encapsulating heterogeneous sources in standard languages and interfaces to ensure cross-system interoperability. The encapsulation constitutes a service description that define what the service does, where it can be invoked from and the format of messages used for client-server communication.

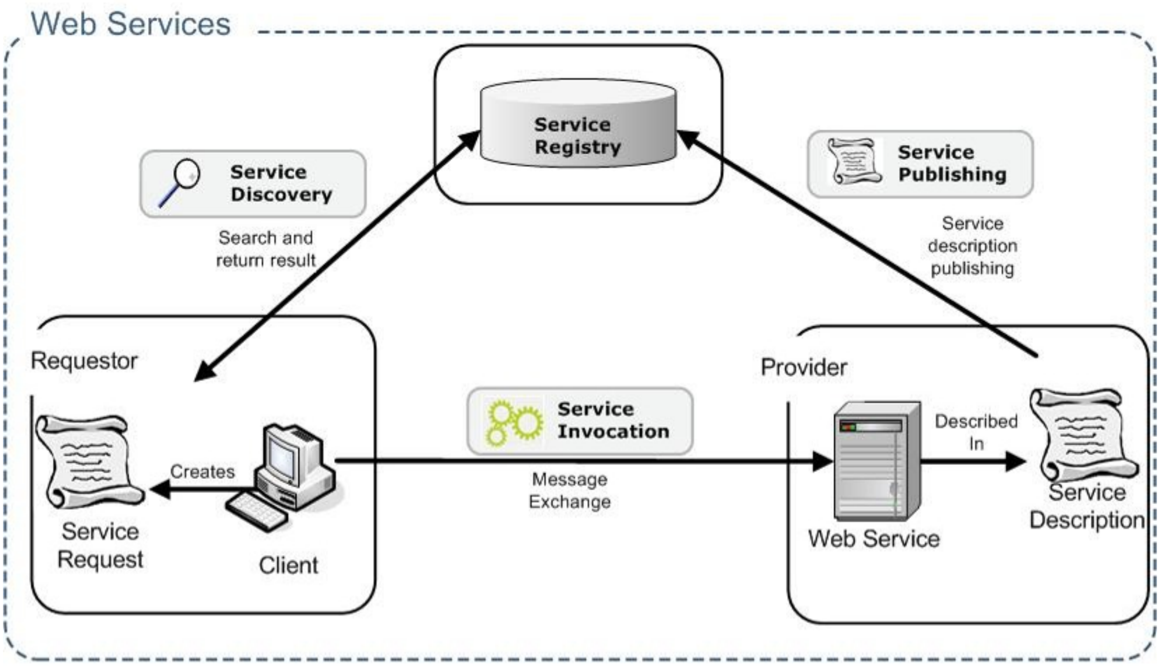


Figure 4.1 Service Oriented Architecture

Figure 4.1 shows the participants in a WS setup as clients, service providers, and registries. The WS are typically described using the WS Definition Language (WSDL) and published in a Universal Description Discovery and Integration (UDDI) service registry. When the wanted service is found, the client uses the service description to interact with the service using the described message format to define SOAP[9] messages.

Cross-system interoperability adds great value to a service environment, but as mentioned there are still limitations when it comes to automatic and dynamic behaviour. When searching for services, the user has to have some notion of how the service is described to be able to find the appropriate

[9] Formerly known as Simple Object Access Protocol

service. The services are described on a syntactic level, and all understanding of what the service does is up for human reasoning. Invocation of services is either client instigated or defined at design time. Orchestration of services is static as it also have to be done at design time. Orchestration is in essence only a composition of services that together constitute one service, and the client has to search for the orchestrated service as if it was a single service.

4.1.2 Semantic Web Services

SWS adds a new layer to the WS in order to enhance dynamic and automatic behaviour in the service environment. SWS is in essence a combination of WS and semantic technologies.

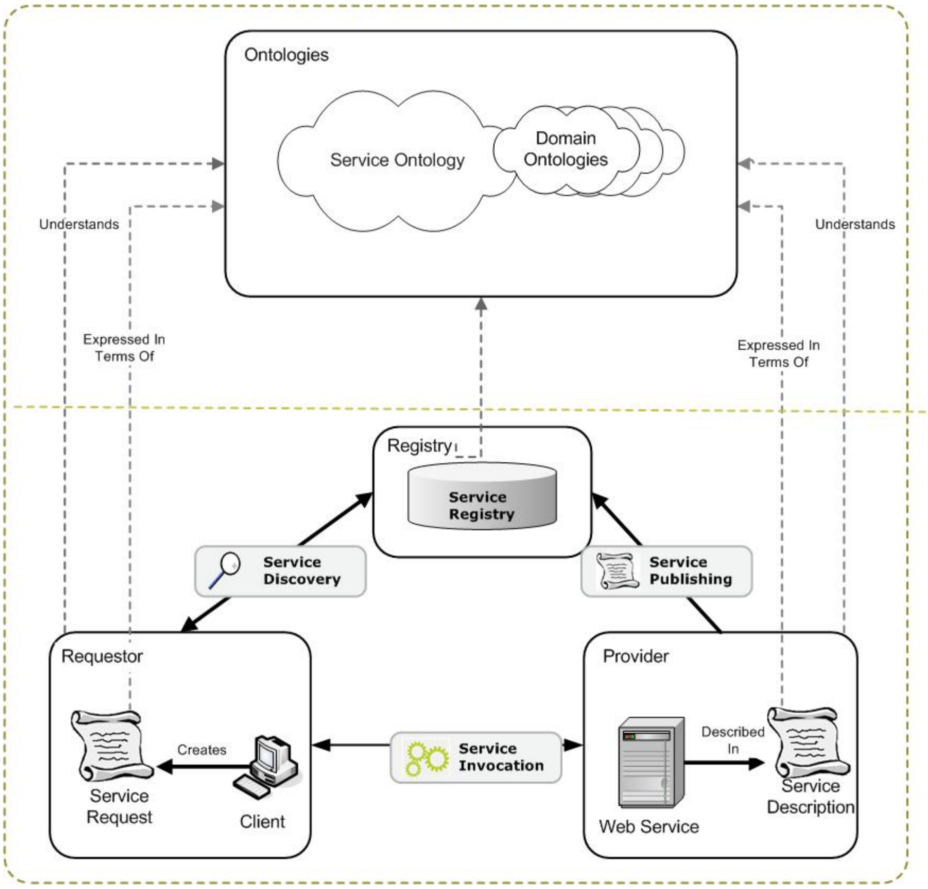


Figure 4.2 Semantic Web Services

The participants in a SWS set-up are the same as the traditional WS, but as Figure 4.2 shows we add a semantic layer that define both what a service is and what domain it covers in the two depicted ontologies. The service ontology defines the properties of a service, while the domain ontologies are used to describe the values of the service properties such as input, output and other service parameters. The combination of service and domain ontologies allows computer reasoning on service capabilities.

By using this semantic description language in the SOA environment, we have a basis for computer reasoning in service selection, orchestration, and invocation. Instead of searching for services syntactically, the client can search for a service capability. In addition, the selection process can find an alternative service if the one originally used has become unavailable. When a service is located, invocation can be performed automatically and orchestration of services can be done on-the-fly. Combining services to fulfil the client needs are based on the individual services capabilities, e.g. if the client searches for a capability and no single service fulfil the client's need, the returned service can in fact be a combination of several services. The promised dynamic and automatic properties following the SOA paradigm, are by this improved when implemented as SWS.

4.1.3 The Semantic Web Services Life Cycle

Figure 4.3 shows the WS life cycle. WS discovery encompasses publishing, identifying, and locating services. In order to achieve this, we need a number of mechanisms present: First, we need a formal description of the service's interface. WS provide such information through WSDLs, which are standardised and describe interoperable interfaces for services and clients. However, WS only support manual selection of services. By adding a semantic service description, we get computer-processable semantics which can provide an automated selection of services. The following describes the steps in a WS's life cycle:

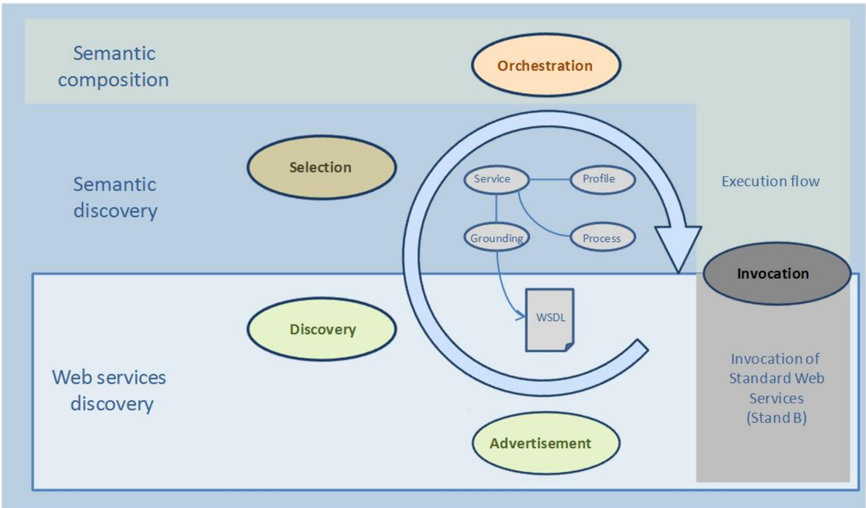


Figure 4.3 The Semantic Web Service life cycle

- We see that the *service description* forms the foundation for this process. WSDLs give the interfaces, and coupled with OWL-S (a service ontology defined using OWL, see Section 4.2.2) we get rich, computer-processable semantics.
- The service descriptions need to be made available to potential clients. This means that once a service has been implemented according to a WSDL, it needs to be published somehow. The service descriptions must be made available through an *advertisement*.

- Once advertisements are disseminated, one can start looking for available services. Querying for advertisements is the process called *discovery*. Basic service descriptions, advertisements, and discovery are what you get from WS technology. The remainder of the steps in the lifecycle require semantic technologies unless you are content with manual selection, orchestration, and human intervention. Manual discovery can be satisfactory when designing a system, but in a dynamic environment it is better to facilitate automated, run-time discovery.

- The discovery process results in a list of available services. Selecting among these can be done manually, or, by using computer-processable semantics the *selection* step can be performed automatically according to selection criteria and the explicit semantics in the service description.

- *Orchestration* means to combine several existing services into a new one. Semantic orchestration can be done at run-time using SWS, where an execution flow can be created automatically. This is in contrast to WS, where you typically only do design-time orchestration using e.g. WS-BPEL. Once an execution flow has been created, either at design-time or run-time, one can start using this composite service.

- *Invocation* is the final step of the lifecycle, where the client binds to the service. In the case of an orchestrated, composite service, the invocation step may include several invocations to different services in succession.

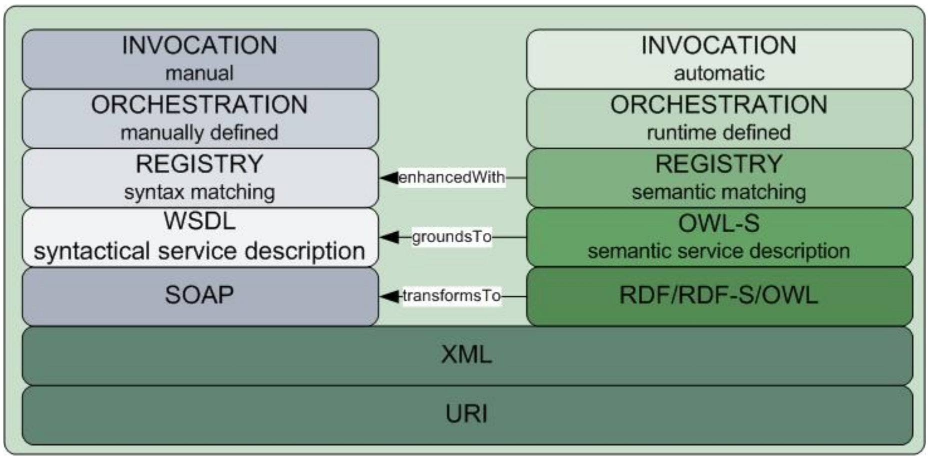


Figure 4.4 Semantic Web Service environment connections to the Web Service environment

The difference between WS and SWS (in this case OWL-S), namely the focus on syntax versus semantics in service descriptions, is shown in Figure 4.4. Note that there is a connection between the two worlds as SWS use several of the standards defined in the WS stack and in fact only adds a new layer to enhance WS. It is important to understand that SWS does not dismiss the vast amount of standards defined in the traditional WS technology stack, but use and expand them. E.g. when a service is located and ready for invocation, the semantic specification of the process model needs to to know the endpoint address of the underlying service as well as the format of the messages to be transported.This is described in the WSDL, and SOAP is still used for transportation.

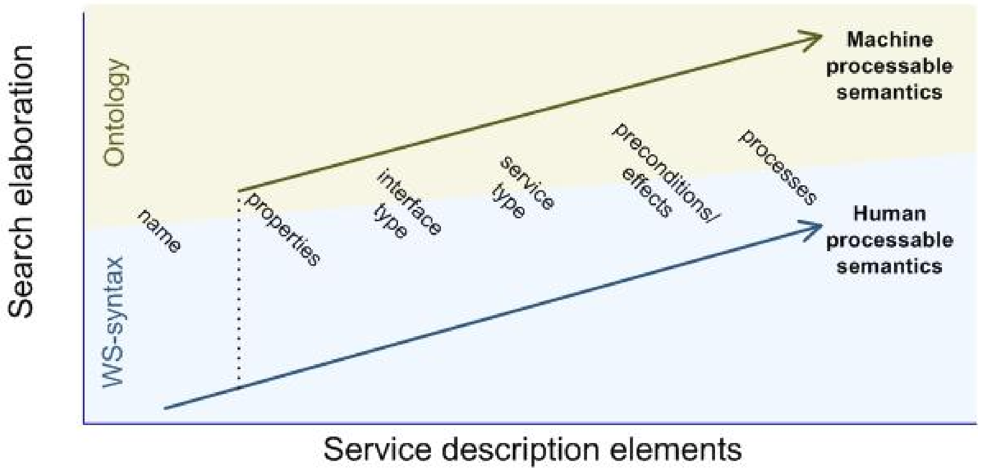


Figure 4.5 Relative description expressiveness for Semantic and non-semantic services

The impact of adding service semantics is portrayed in Figure 4.5 where we show the difference of WS descriptions with and without computer-processable semantics in terms of how they facilitate the tasks necessary for discovery and orchestration. Elaboration of search based on descriptions are divided in "WS-syntax" and "Ontology". The WS-Syntax enables syntactic interoperability, but relies on human interpretation of implicit semantics in terms of ensuring semantic interoperability (e.g. agree on meaning of input/output and functionality of service). The user process the implicit meaning, and as one would expect, increasing number of description elements enable the user to make more elaborate searches for services. An ontological description, on the other hand, includes both the interoperability aspect of the WS-syntax but also enables computer-processable semantics. As with WS-syntax the number of description elements could enable more elaborate searches, but in this case the more elaborate search is computer-processed. In general you see that accumulating service description elements increases the searching possibilities made feasible by the description, but using semantic descriptions elevate it further.

An important thing to note is that making service selection based on name, type, and attributes without explicit semantic description, is dependent on a priori knowledge of the invocation semantics, or protocol, of the service. Further, a service name does not tell the computer anything about the capabilities of a service, which are important to know in the cases where we want to select and invoke services of an unknown type in run-time. Ideally, a client would just need to have an idea of the goal or task to accomplish, and a proper service would be found.

4.2 Standardisation

The elements of WS are standardised by the World Wide Web Consortium (W3C). Standardisation has shown to be important to ensure adoption of technology as well as representing a common ground between systems to ensure interoperability. W3C has a standardisation activity also for semantic technology including SWS based on the elements of the Semantic Web stack (see Section 2.5).

4.2.1 Web Service Standardisation

W3C, Organization for the Advancement of Structured Information Standards (OASIS) and the Web Services Interoperability Organization (WS-I) are organisations handling WS standardisation. There are vast amounts of standards covering a number of areas in a WS environment (InnoQ 2007).

4.2.2 Semantic Web Services Standardisation

Several initiatives have been submitted to W3C as suggested standards for SWS. These include OWL-S (W3C 2004*a*), WSMO (Polleres et al. 2005) and SAWSDL (W3C 2007*a*). In the following, the listed initiatives are described along with their respective strengths and weaknesses. The different approaches are studied in more detail in Rustad & Gagnes (2006).

OWL-S originates from Defense Advanced Research Projects Agency (DARPA) and their DARPA Agent Markup Language (DAML) project. The OWL-S (formerly DAML-S) project was started in 2000 and was proposed to the W3C as a standard. OWL-S use Semantic Web standard technologies (see Section 2.5), and have been driven by commercial interests. The OWL-S approach is a mature initiative that builds on W3C's Semantic Web technologies, enabling dynamic and automatic selection, orchestration, and invocation. Even though this is the most mature initiative there are a limited number of available tools ready for use, and the tools are often immature and incomplete. For our technical try-outs we have in the Semantini project chosen OWL-S as our preferred SWS language. The reasons for choosing OWL-S are maturity of base language (OWL-S is written in OWL), the fact that other focus areas of Semantini use OWL, and the tool-support (even if it is limited).

The *Web Service Modeling Ontology (WSMO)* development was founded by the European Commission, Science Foundation Ireland and by the Vienna city government and developed by both industrial and academic partners. European Semantic Systems initiative (ESSI) WSMO group aligned SWS research projects in order to strengthen European research through world-wide standardisation. In 2005 WSMO was submitted to the W3C for standardisation discussions. The motivation of defining a new SWS solution were areas identified as limitations in both OWL and OWL-S arguing that the logical language used did not support the best possible reasoning about services. WSMO aims to be interoperable with other initiatives, and their own Web Services Modeling Language (WSML) can in some areas map to OWL. But to overcome OWL-S problems, they include language elements not possible to map to OWL-S. We have chosen not to use WSMO for our try-outs as the initiative adds a new ontology language not used in the other Semantini activities, and coordinating efforts in the area is important. Other deciding factor is the complicity of WSMO, and the lack of tool support.

SAWSDL originates from the Meteor-S project at the Large Scale Distributed Information Systems (LSDIS) lab as a successor to WSDL-S. WSDL-S was submitted to W3C in 2004, and shortcomings in the submission resulted in 2006 in a SAWSDL group in W3C expected to produce a recommendation. In 2007, W3C announced the SAWSDL recommendation. Building on standards is important, and as SAWSDL builds on top of the WSDL standard this is an important contribution to the SWS environment. SAWSDL is ontology-language independent and all types of semantic annotations can be added to an extended WSDL description, enabling dynamic service discovery, composition and invocation. However, it depends on other semantic service descriptions in order to facilitate automatic properties enabled by the previous mentioned initiatives.

4.3 Service Registries

The service descriptions need to be stored in a network-accessible framework which allows service providers to advertise them and clients to discover and access them. One traditional method is the use of a registry, which is said to be an authoritative, centrally controlled store of service descriptions. There are several registries available for plain WS, but there are only a few that are somewhat ready for SWS. As this is an implementation aspect in NNEC, this section is devoted to semantic support in registries.

Universal Description, Discovery and Integration (UDDI) (Curbera et al. 2002) is the most frequently used registry for WS. Service providers advertise their services with service descriptions, and clients can find services by name, type, binding and according to a taxonomy. UDDI has third part support for OWL-S-based discovery. Basically, UDDI allows service providers to register their services and service consumers to discover these services both at design-time and run-time. The UDDI registry supports reconfiguration as long as services do not go down unexpectedly. If this happens, advertisements will be in the registry forever because there is no liveness information in the current versions of UDDI.

Another effort in the WS world, also by OASIS, is electronic business XML (ebXML) (Patil & Newcomer 2003); a collection of specifications for conducting business-to-business integration over the Web. EbXML supports more advanced queries than UDDI. Unlike UDDI, the ebXML registry can store vocabularies like XML schemas and ontologies since it also specifies a repository for such items.

WS-Dynamic Discovery, or WS-Discovery for short, is a proposal from several vendors for how to discover nearby WS in ad-hoc networks (Schlimmer et al. 2005). With WS-Discovery, service matching is based mainly on the WSDL port type [10] supported by the service. The port type is described by a namespace URI, and some scope limitation can be done through a simple filter.

[10] WSDL port type: a collection of service name, operations and messages involved in service execution.

WS-Dynamic Discovery does not support discovery based on semantic descriptions.

UDDI, ebXML and WS-Discovery all lack native support for handling SWS descriptions for discovery, invocation, and orchestration. In collaboration with the FFI SOA project we approach the problem by expanding WS-Discovery. The reason for choosing WS-Discovery, is to accommodate the network-centric battlefield where the connectivity can vary while at the same time ensuring interoperability with other systems.

In summary, we can say that several key properties are missing when deploying today's standards for WS discovery in dynamic environments with support for semantic descriptions.

4.4 Future Research

SOA has been identified in the NNEC feasibility study as an enabling technology. Traditional SOA implemented as WS have limitations which use of semantic technology can remedy. Adopting SWS in systems today does not imply that traditional WS has to be replaced. We add an additional layer on top of the already existing SOA enabling computer reasoning about services in selection, orchestration, and invocation.

Experiments performed at CWID-08 and DEMO-2010, see Section 8, proved that SWS could enhance the SOA environment, enabling machine reasoning about services. Running the experiments, a limited selection of SWS possibilities was implemented. There are several areas of interest for future research.

The number of elements describing the individual services were limited in our experiments. There are several other interesting elements when describing services, e.g. preconditions and effects witch describes the world before and after a service is run. This could enable reasoning about what the service actually can do for the user.

SWS efforts in DEMO-2010 was part of a collaboration with the FFI SOA project, where focus was on SWS in MANETs [11]. Quality of service (QoS) parameters constitute a family of description elements that could facilitate better service selection in such an environment. A thought is to use a clients position to select services, based on both network quality and user application requirements.

5 Information Integration

Information integration is a fundamental problem in any environment where several systems need to exchange information. The reason for this is that the said systems are usually not designed to interoperate. In Motro & Anokhin (2006), the information integration problem is defined as *providing a user with the means to (1) perceive a collection of heterogeneous and autonomous information sources as a single source, (2) query it transparently, and (3) receive a single unambiguous answer*.

[11]MANET: Mobile Ad Hoc Network

Heterogeneous here refers to difference in data models, data representations, and interfaces, while autonomous refers to the sources being developed independently of each other, and being maintained by different organisations that may wish to retain control of their sources.

The definition of Motro & Anokhin is the basis for this account on using semantic technologies to handle the information integration problem.

5.1 Solving the Information Integration Problem

The information integration problem can to a certain extent be solved using conventional information technologies. Such solutions are often centred around paradigms like service-oriented architecture (SOA) (Josuttis 2007), extract transform load (Kimball & Caserta 2004), business process management (van der Aalst et al. 2003) or a combination of these. However, these solutions tend to be implementation specific and are often inflexible and costly to setup and maintain, both in terms of man-hours and money (Duke & Richardson 2009). One reason for this is these technologies' inability to represent the semantics of the information to be integrated in an explicit manner separately from the systems. This forces the creation of one-to-one mapping solutions, and the semantics typically being hard-coded in the systems or even kept only in the heads of the system developers.

Both the one-to-one mappings and the hard-coding make these solutions brittle. When the semantics change or a one-to-one mapping breaks, considerable effort is often needed to re-establish the integration solution.

Semantic technologies have the potential to contribute to solutions that are more flexible. There are in particular three aspects that make these technologies interesting as a contributor to solve the information integration problem:

1. The use of a generic, graph-based, information structure
2. The ease of performing federated queries
3. The possibility to align different formats at the semantic level

Information treated using semantic technologies is represented as graphs. This immediately provides a simple integration framework, as integrating two information graphs becomes a trivial case of graph merging. This is illustrated in Figure 5.1, where the graphs $g1$ and $g2$ are merged on a common node.

Federated querying, i.e. issuing one query to a number of sources and receive a single answer, is an important element in the information integration problem, as should be evident from the definition. SPARQL, see Section 2.5.4, has support for these kinds of queries, making it an interesting technology to include in an information integration solution.

By representing the semantics of the information in ontologies, see Section 2.3, semantic technologies open up the possibility to address the difference between the formats of the different systems on

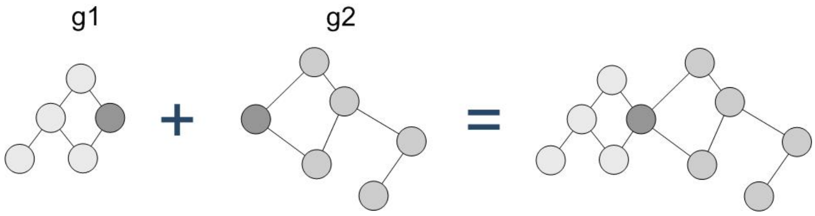


Figure 5.1 Integrating the information graphs g1 and g2 on a common node (Based on (Lacoul 2009, Figure 1))

the semantic level, i.e. create mappings between the ontologies. As an example of how this works, consider Figure 5.2 where information from a logistics system and a C2 system is to be integrated. This example highlights two concepts in the two systems that in fact are related: The C2 system specific concept *ReportingAgent* and the logistic system specific concept *Employee*. However, without more specification it is not possible for a traditional integration solution to treat these two concepts as related. This can be mitigated by the use of ontologies and relating these two application specific concepts to more general concepts: *ReportingAgent* is in this particular case a specialisation of the concept *Soldier*, residing in a general C2 ontology, while *Employee* is a specialisation of *MilitaryEmployee* belonging to a general logistics ontology. Moreover, there exist a relation between these two general concepts stating that a *Soldier* is a *MilitaryEmployee*. By representing this relationship formally, an integration of information from these two systems can use the fact that any *ReportingAgent* and any *Employee* from their respective systems are also a *MilitaryEmployee*, making it possible for instance when queried for all available *MilitaryEmployee*s to return both all *ReportingAgent*s and all *Employee*s.

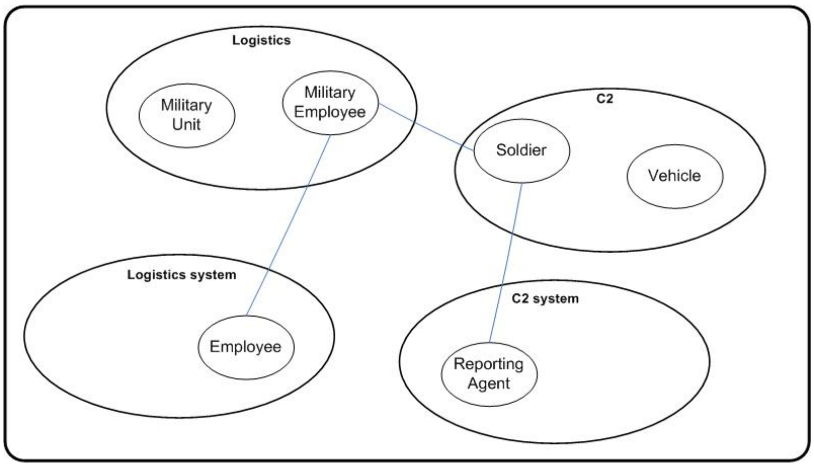


Figure 5.2 Integration of a logistics system and a C2 system

5.2 Elements in a Framework for Information Integration

The use of semantic technologies to solve the information integration problem, requires first and foremost that the information to be fetched and integrated is linked to an ontology. When the semantics is represented in such a way, the linking between the information from the different sources can be performed on the ontologies as exemplified in Section 5.1.

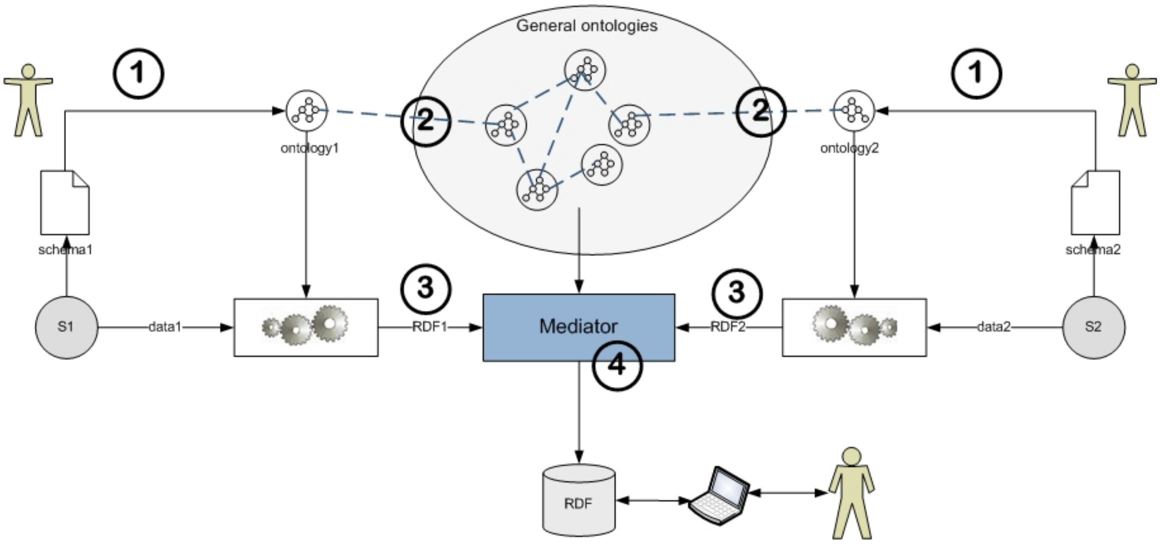


Figure 5.3 Integrating information from the heterogeneous sources S1 and S2

A framework for information integration using semantic technologies should include the following capabilities, as also illustrated with the corresponding numbers in Figure 5.3:

1. The making of ontologies representing the intended semantics of the information being sent out from each system,
2. relating these ontologies, linking them together (ontology matching),
3. transforming the output from the systems to RDF, and
4. utilising the explicit links between the ontologies to link together the information from the systems, including deciding whether any information elements from the different systems represent the same real-world object (entity association).

The elements needed to implement solutions for semantic information integration are thus:

- Ontologies
- Ontology matching methods
- Transformation methods from various formats to RDF
- Entity association methods

In addition to this, automated reasoning (see Section 3) to assist ontology matching and entity association is needed. The reasoning infers the relations necessary to perform these two processes.

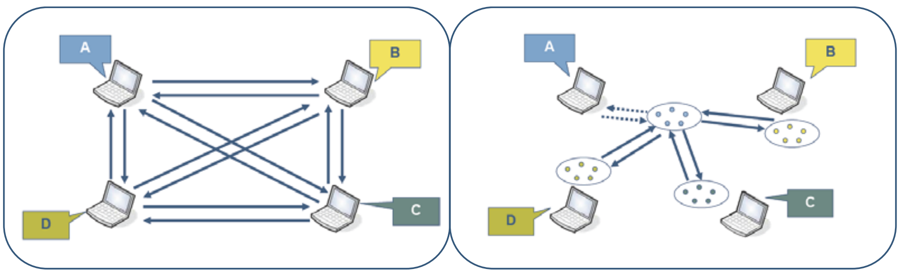
While several of these elements exist and enjoy a reasonable level of maturity, tools and processes needed to link these building blocks together are still lacking and/or experimental. At this point in time, labour-intensive tasks like ontology matching and entity association still need to be performed by, or at least supervised by, human operators. Further, the art of creating ontologies is still not very well explored with a lack of best practices and user-friendly tools. In the following, the above-mentioned elements are further described.

5.2.1 Ontologies for Information Integration

Ontologies, see Section 2.3, are fundamental to any application or solution utilising semantic technologies. Concerning information integration, there are two aspects of ontologies that are particularly important: Ontologies are linkable, making them suited to be developed iteratively, and they make way for a more efficient integration between systems by allowing any mismatch between the systems to be dealt with on the semantic level.

By supporting iterative development, ontologies make it possible to start an integration effort with building a small ontology covering only what is needed to integrate the available systems or information sources. Subsequently, the ontology can be further developed, either by changing the ontology or linking to other ontologies, when further information sources emerge.

The advantage of dealing with the differences between systems using ontologies rather than directly between the systems, is illustrated in Section 5.1. The advantage becomes even more evident, however, when the number of systems to integrate increases. When relying on one-to-one mappings between the systems, the number of necessary mappings increase exponentially in the number of systems. By instead relying on ontologies, it is possible to approach a linear growth in necessary mappings. This is illustrated in Figure 5.4.



(a) (b)

Figure 5.4 Point-to-point mappings (a) vs. information integration based on ontologies (b)

5.2.2 Transformation to RDF

In order for systems based on semantic technology to handle information, the information has to be transformed to RDF. There exist a multitude of such conversion tools, see for example Bergman (2010), and the most commonly used structured formats, like spreadsheets, relational databases, etc., are covered. There also exist solutions to extract RDF from text, like Thomson Reuters' Open Calais (Thomson Reuters 2010).

The topic of converting legacy information into RDF is also partly covered in Section 6.

5.2.3 Ontology Matching

Ontology matching is in Euzenat & Shvaiko (2007, p. 42) defined as *the process of finding relationships or correspondences between entities of different ontologies*. In other words, it is the process of identifying what concepts in the different ontologies are connected, and in what way are they connected (same concept, one concept is a subconcept of the other, etc.).

Although having been studied for several decades, there still hasn't emerged any widespread methodology or best practices regarding ontology matching. There does, however, exist a wide variety of semi-automatic, and often experimental, ontology matching systems. An extensive list is provided in Euzenat & Shvaiko (2007, Chapter 6).

5.2.4 Entity Association

Entity association is the process of identifying what pieces of information from different sources really concern the same real-world entity. This is a fundamental process when doing integration, regardless of whether the sources are based on semantic technologies or not.

The importance of entity association is illustrated in Figure 5.1 where the integration of two information graphs becomes a trivial case of graph merging: The result of a merging of two graphs remains two unconnected graphs unless some common node ties them together or other links are discovered between nodes in the two original graphs. Entity association is thus the process of identifying possible common nodes between the two information graphs.

Entity association in the context of semantic technologies is still a field of research where some experimental methods exist, like for instance the L2R method, that exploits the OWL ontologies of the information in order to decide on association (Saïs et al. 2007), but no widely used method or best practices has yet emerged.

5.2.5 Automated Reasoning

Automated reasoning, already treated in Section 3, has been an important field of study for several decades. There exist several mature reasoners on the market, the most prominent being RACER

(Haarslev & Moller 2001), Fact++ (Tsarkov & Horrocks 2006), KAON2 (Motik & Sattler 2006), and Pellet (Sirin et al. 2007). These reasoners are all generic description logics reasoners, meaning they can perform reasoning on any OWL ontology.

Automated reasoning is the most mature building block in information integration solutions.

5.3 Applications within the Military Domain

From a technological point of view, the core of a network-based defence (NBD) is the information infrastructure (INI), providing the capabilities the users need to solve their respective tasks. The Norwegian INI corresponds to the networking and information infrastructure (NII) of NATO network enabled capability (NNEC) (NC3A 2005).

One of the main tenets of NBD is that military units should have the opportunity to collect relevant information from all available information sources, also the unanticipated ones, to build and maintain a shared situation awareness. In addition to the ability to fetch the relevant information, there is also a factor of timeliness: The military solution to the information integration problem has to take into account that the information in question in several cases is time critical.

A solution to the information integration problem can be seen as a way to support the ability of military units to collect information from unanticipated information sources. Semantic technologies have the potential to address the above-mentioned challenges, provided that the information sources in question adhere to ontologies for their information. These technologies should thus be considered when building information systems to the INI.

5.4 Future Research

The main challenge in employing semantic technologies in solutions for the information integration problem is the lack of tools and processes to tie the different technical building blocks outlined in Section 5.2 together. This should thus be the main focus in further studies.

In parallel with theoretical studies, however, experiments should be conducted both in order to help decide the scope and to verify the results of the studies. Of particular interest should be to decide what is needed in terms of ontologies (types of ontologies, number of ontologies, how to create them, etc.). Further, the topics of ontology matching and entity association should be investigated both through theoretical studies and experiments. Information integration solutions utilising semantic technologies rest heavily on these two areas, and the goal should be to identify methods to conduct these processes and assess their suitability in the military domain.

Experiments are also needed in order to assess how well the technologies behave in an environment of operational military systems handling information of realistic size and complexity.

6 Handling Distributed Information

The current usage of semantic technologies is predominantly linked with the Semantic Web paradigm (Section 2.4). This paradigm and the accompanying conceptual model, Resource Description Framework (RDF) (Section 2.5.2), introduce new ways of thinking with regards to distributed information. It represents a shift from isolated information silos to a structure of fragmented and shared information - *the fragmented information graph*. This force content producers to think differently about how information is modelled and published, and force content users to think differently about how to obtain information. This section will have a closer look at the fragmented information graph as well as the components supporting the handling of distributed information.

6.1 The Fragmented Information Graph

The way information is organised on the Semantic Web, represents a shift from isolated information silos to a WWW-wide fragmented information graph where potential information providers have the means to publish their information on the Internet for anyone to use. On the Semantic Web information is in a uniform format (RDF) and anyone can extend information that is already in a graph - the so-called *AAA* principle: Anyone can say Anything about Any topic (Allemang & Hendler 2008, pp. 7–8).

This means that any information in this giant information graph is potentially used by multiple content consumers. The information thus needs to be modelled not only for a specific purpose or application, but also with re-usability in other unforeseen applications in mind. A combination of reusable generic higher level structures reflecting the generic nature of what is being modelled (upper and general domain ontologies, see Section 2.3) and specific ones to fulfil concrete requirements of an application (application ontologies), is a fundamental principle and a prerequisite for this fragmented information graph.

A concept closely related to the fragmented information graph is *linked data* (LD), often also named *Linked Open Data* (LOD). Coined in Berners-Lee (2006), LD highlights that the Semantic Web is not so much about the data that is being put on the web as it is about making links allowing a person or computer to explore the web of data by following these links. With LD, when you have some of it, you can find other, related data.

In order to build a web of data, four rules, often referred to as the LD principles, were presented:

1. Use URIs as names for things
2. Use HTTP URIs so that people can look up those names
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL)
4. Include links to other URIs, so that they can discover more things

Following these principles, an information provider assures that the information elements are not only unique (by using URIs, LD principle 1), but also that they can be reached using HTTP (LD principle 2) and that they lead to web addresses that provide useful information (LD principle 3). By also applying LD principle 4, the information provider helps the user to discover other, related information by linking to other information elements possibly residing elsewhere on the WWW. This gives a user the possibility to browse the web of data much the same way he can browse the WWW today.

6.2 Information Collection

In order to take advantage of the new opportunities for information handling presented by semantic technologies, the information have to be collected from various sources and transformed to the RDF format.

As suggested in section 6.1, an increasing amount of information is being made available on the WWW offering information on RDF form. It is still the case, however, that the vast majority of sources offering relevant information is not available in this form. They can be expected to retain their original form and function for a long time, and it will be important to find ways to take advantage of the information they have to offer. It is neither realistic nor desirable to expect all applications to convert to RDF internally, and this means that conversion of information upon export will be necessary to make use of this information in systems based on semantic technologies.

In an environment of dynamic and fragmented information, mechanisms to locate the relevant information are of critical importance. A vocabulary addressing this issue is presented in the following. Additionally, how to utilise information from RDF sources (triplestores) and relational databases is outlined.

6.2.1 Finding Information on the Web of Data

One initiative to address the challenge of locating relevant information on the web of data, is the ontology *vocabulary of Interlinked Datasets* (voiD) (Alexander et al. 2009). VoiD aims to provide a vocabulary to bridge information publishers and information consumers, so that consumers can find the right published information for their tasks easily by using the voiD description for a (linked) dataset. With discovery of datasets we mean the identification of datasets given certain attributes, trying to answer the question: Given a set of properties, which available resources match the desired set, which properties do they have, and what are their locations?

A dataset is a collection of information, published and maintained by a single provider, available as RDF, and accessible, for example, through dereferenceable HTTP URIs or a SPARQL endpoint. Dereferenceable HTTP URIs mean that content can be found at the location specified in the given URI.

The main attributes of voiD is allowing an information provider to specify where the information can be found (*sparqlEndpoint*, *uriLookupEndpoint*), what ontologies are being used describing the information (*vocabulary*), and what URI schemes are being used to identify the information elements (*uriRegexPattern*).

6.2.2 Native Triplestores

A triplestore is a purpose-built component for the storage and retrieval of RDF information. Conceptually it resembles a relational database, as information is stored in the triplestore and is retrieved from it using a query language. It is different from a relational database as it is optimised for storage and retrieval of RDF triples.

Some triplestores can store billions of triples (Rohloff et al. 2007). The performance of a particular triplestore can be measured with for instance the Lehigh University Benchmark (LUBM) (Guo et al. 2005).

The open and flexible information structure of RDF has other inherent requirements than traditional databases for storing information. While a relational database stores information with a predefined structure (information needs to fit into the pre-defined database schema constructed with Data Definition Language (DDL)), triplestores need to handle information with little predefined structure, as the structure is defined by the information. It is common, however, to use a relational database for storing triples in a triplestore. The terms *schema-oblivious* and *schema-aware* have been coined to describe the storage architecture of triples in relational databases, where the former refers to a three-column (subject, predicate, object) database table for triple storage, and the latter describes alternative database structures which are optimised and adapted to the RDF content that is stored. The schema-aware alternative is more complex with regards to structure, but can improve query performance (Theoharis et al. 2005).

Both the industry and academia have a large focus on creating triplestores that are easy to use and perform well. As an example, Oracle Corporation, a world-leading enterprise database provider, offers support for triples in its *Oracle Database 11g*.

The requirements of triplestores are much the same as for relational databases in addition to features specific to the Semantic Web paradigm. We consider that key features of a triplestore should include:

- Ability to store triples efficiently
- A SPARQL endpoint that allows local and remote access to the information
- Basic reasoning support
- Traditional database requirements such as performance, scalability, reliability and robustness

6.2.3 RDF from Relational Databases

Legacy relational databases contain valuable information which should, and can, be used by applications based on semantic technologies. This information is typically stored in a proprietary binary format, only computable and available through the database's interface and SQL-queries. The database content can be represented as RDF, however, by applying a wrapper around the database. This software typically has several key functionalities:

- SPARQL query-engine
- HTTP-server functionality
- An information conversion algorithm
- On-the-fly conversion of information

There are several concerns related to conversion of traditional relational database content into RDF. Not all content in a database is relevant outside a specific application's scope. An application may store information on for instance user-history and user interfaces which would be of no meaning or importance to other consumers but the specific application. Also, sensitive information stored in the database needs to be handled in a secure way.

Another critical property of such software, is on-the-fly conversion of information. Legacy databases can contain high volumes of information, and exporting all the information into RDF in one large export is not realistic. It is however not only size concerns that make this unwanted. As information in databases often are dynamic, making a snapshot export of the database is undesirable. One does want the RDF representation of the database content to be up to date.

In our experiments, we have so far focused on two tools that offer relational database content as RDF: D2R Server (Bizer & Cyganiak 2009) and Triplify (Aumuller & Dietzold 2009).

D2R Server is developed by Freie Universität Berlin, and it enables RDF and HTML browsers to navigate the content of the database. Further, it allows applications to query the database using the SPARQL query language. D2R Server supports Oracle, MySQL, PostgreSQL, Microsoft SQL Server and any SQL-92-compatible database. D2R can auto-generate a database-to-RDF mapping file. The user can then either use this mapping directly without modifications, or modify the mapping file manually. A manual edit is necessary to remove any unwanted information from being exposed, e.g. application-specific information.

Triplify, developed by the Agile Knowledge and Semantic Web research group at Universität Leipzig, is another solution for exposing database content to the Internet, where developers can use SQL-queries to extract information from the database. Unlike D2R Server, Triplify does not require a database-to-RDF mapping file. Instead, it harnesses the maturity and flexibility of SQL to extract information from the database. However, a configuration file containing the SQL queries used to extract information from the database and converting the information into suitable RDF elements must be created by the system developer. The current version of Triplify does not offer a SPARQL endpoint, and information has to be accessed REST-style (Fielding 2000).

6.3 Applications within the Military Domain

The fragmented information graph maps well with the ideas underpinning the future information infrastructure (INI) of the Norwegian Armed Forces, which, among other things, will allow military units to share information in a much larger degree than is being done today. Further, the tools presented in this section constitutes promising building blocks with a potential to provide INI with capabilities regarding handling of distributed information when used appropriately. These tools should be relatively easy to introduce into INI as they are constructed with co-existence with legacy systems in mind.

6.4 Future Research

A promising way to build a fragmented information graph is by employing the linked data principles. Further research on the applicability of the fragmented information graph to support the expected increase in information sharing as a result of NBD, should thus include experiments exploring how well suited the linked data principles are in the military domain.

Further, information storage seems to be among the most mature areas of the Semantic Web component tool box. Numerous triplestores and information retrieval engines have been developed and deployed with promising results. Further determination of maturity can only be done with implementations dealing with real information in a real operating environment.

7 Experimental Application 1: Automated Reasoning Based Intelligence Tool

One possible usage for semantic technologies in the military domain is in decision support systems. This includes systems dealing with C2/3/4 , early warning and surveillance, intelligence analysis, etc. In order to explore this application area, we developed ARBIT (introduced in Section 3.4), a demonstration decision support system for the intelligence domain put to work in a fictitious intelligence setting. The system works as an aid for intelligence analysis in terms of acquiring relevant information from unanticipated information sources, and automating knowledge-intensive tasks such as classifying and deducing conclusions from datasets .

7.1 Intelligence Analysis Use Case

ARBIT was developed for, and demonstrated at, the FFI DEMO-2010 event in January 2010. The aim of DEMO-2010 was to show possible usage of future technologies within the context of NNEC, and the technologies shown were based on results from the FFI projects SecSOA, SINETT and Semantini, in cooperation with NC3A. The event revolved around a fictitious scenario involving NATO forces working in a failed state, involving terrorist activities. ARBIT was demonstrated in DEMO-2010 under the story of an intelligence analyst working on an observation report received from the battlefield.

The specific observation report in our intelligence story is regarding the observation of two trucks near a storage building. The intelligence analyst first uses the system to fetch more information about these two trucks.

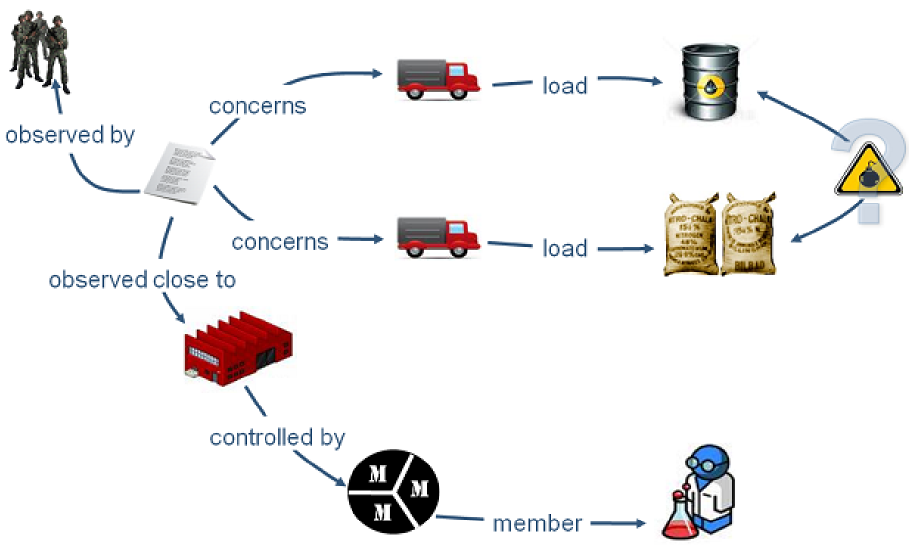


Figure 7.1 Conceptual graph of story

In the background, a federated query is sent out and the information retrieved from the different sources (based on the initial query) is then integrated together as a single answer (in the form of a graph) that followingly is merged with the initial information about the observation (the main graph). The conceptual graph of the information in the story is shown in Figure 7.1. In this case information regarding what the trucks are carrying has been retrieved; ammonium nitrate and fuel oil. The system then uses background knowledge regarding chemistry and bomb making (defined in ontologies and rules) in order to identify that the co-location of these two trucks might indicate a possible bomb as the load that the trucks carry together are major components of an ammonium-nitrate fuel-oil bomb. Furthermore, it notices that if this is a bomb, then it also partially matches a valid time-sensitive target (TST), that according to military doctrine should be taken out immediately. Yet, this still might be coincidental co-location, so the analyst continues to drill down.

The next branch that the analyst focuses on is the storage building. Further drill-down indicates that this building is controlled by a known terrorist organisation. Drilling down on the organisation itself shows that it has a member that has bomb-making expertise. The system now classifies this organisation as a terrorist organisation with bomb-making expertise. Furthermore, this completely satisfies the rule for classifying a specific time-sensitive target, and the analyst is appropriately warned. The warning is then delegated to decision makers in order to decide on appropriate actions.

7.2 Description of System

The ARBIT system provides functionality for graph navigation, information integration, federated querying, ontology- and rule-based reasoning, as well as partial matching of rules. From the analyst's perspective, this should address both fetching relevant information and automatically drawing meaningful conclusions.

First of all, ARBIT provides the ability to drill-down on information (fetching additional information). This is done by performing federated querying and information integration based upon user guidance (point-and-click on nodes in the graph to find and fill in more information). Thus, the systems provides human-guided querying of various unanticipated heterogeneous information sources and returning the results as a single unified answer (information integration, see Section 5 for more details).

In order to facilitate this, the federated querying component of ARBIT utilises voiD descriptions of the information sources (see Section 6.2.1) to determine what sources are relevant. The voiD descriptions are stored in a standard triplestore, named ARC, (ARC-Development 2009) which provides a SPARQL endpoint that ARBIT can use to query against, thus working as a type of service registry. Information resources can be activated and deactivated by registering and de-registering them in the voiD triple store.

For the analysis part, the system provides functionality to import a variety of different expert background knowledge about domains based on user needs. This background knowledge is in terms of ontologies and rules that form executable models of domains. Thus, these models can be run on the dataset and used to infer conclusions from it (e.g. classify objects, infer relations between objects, enforce constraints, etc.). Concrete examples include determining if a dataset indicates a potential bomb and/or a time-sensitive target, classifying an object as a certain type of vehicle, etc.

In addition to executing/firing rules as the conditions are satisfied, the system performs partial rule matching (aka. relaxing the rules) in order to help the analyst drive the data drilling forward. This will hopefully aid the user to reach an answer to the question at hand. E.g. if a rule indicating a time-sensitive target is matched partially, the system will indicate what is missing and indicate where to find more information in order to confirm a suspicion. Figure 7.2 shows ARBIT indicating that a rule is partially matched, where the two dark blue ellipses indicate unmatched conditions (what needs to be found). The tool is semi-automated, as a fully automated approach would result in a combinatorial explosion with a growth in number of sources. Thus humans guide the search space. This technique is common in automated reasoning applications such as mathematical theorem proving, where the search space is unbound (infinite). Early indications show that this approach is suitable for our task as well.

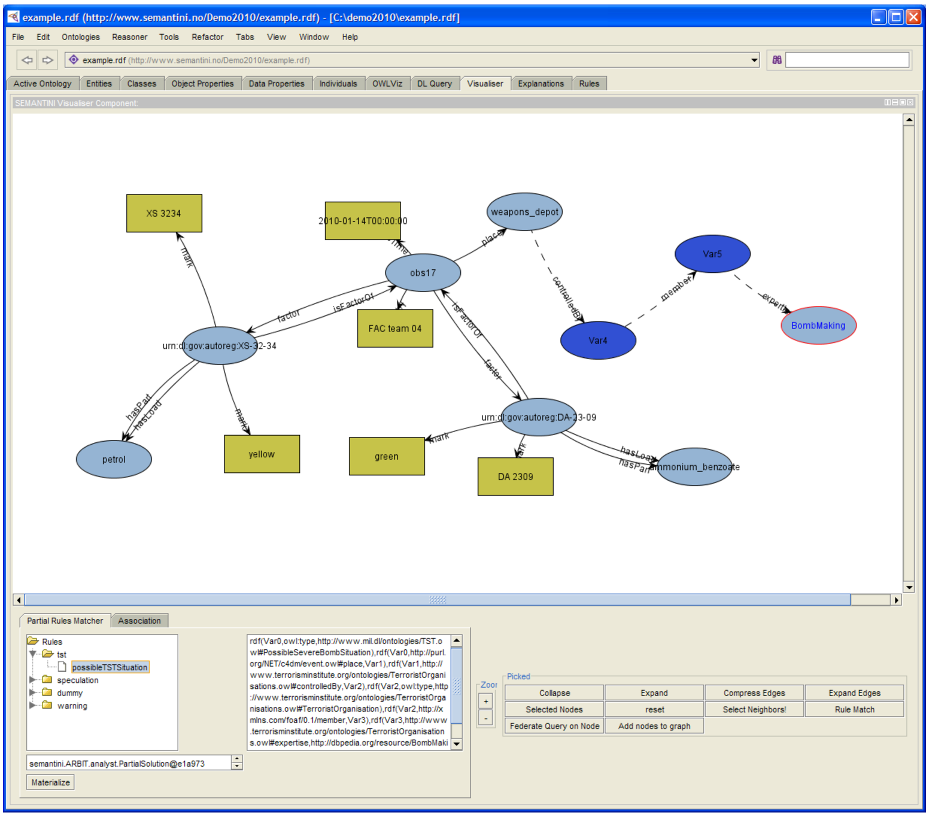


Figure 7.2 Screenshot of ARBIT

8 Experimental Application 2: Applied in an Infrastructure of Dynamic Services

During the Semantini project there have been two experimental implementations to demonstrate and gain experience with Semantic Web services (SWS). At CWID-08, Semantini presented semantic discovery in a traditional service-oriented architecture (SOA) environment. In this experiment, a collection of service providers advertised their services in a registry and a client requesting services searched the registry for selection of services based on semantic descriptions. At DEMO-2010, the focus was on runtime orchestration and automatic service invocation in networks without registry connections. The following sections describe the two experiments in the context of a military operation where a service that delivers surveillance footage of an area is requested.

8.1 CWID-08, Semantic Web Services Discovery

NATO Coalition Warrior Interoperability Demonstration (CWID) is an annual event to support the NATO network enabled capability (NNEC) program . The NNEC slogan "Share to Win" describes the need for interoperability on all levels in the coalition, and SOA has been identified as an enabler for technical interoperability (NC3A 2005). CWID is an arena to share technical interoperability solutions and also to meet with other coalition IT professionals.

Semantini participated in CWID-08 at Lillehammer with a technical experiment of service discovery using a third-party semantic matchmaker on a registry, achieving service discovery and invocation based on semantic descriptions written in OWL-S (Hansen et al. 2008). Not previously known services were described during the experiment, added to the registry, and used to find and invoke the services. Service requests where resolved using direct conceptual matching and subsumption matching (see Section 3), and invocation was performed through a generic Semantini-developed client proving dynamic invocation of the services.

To exemplify the semantic service discovery experiment, we imagine that during a military operation a military decision maker needs information about military units in an area. A deployed team on a reconnaissance mission receives updated information from the headquarters (HQ) guiding them to their destination. To guide them safely through a terrain with enemy activity, the decision maker needs a complete situational picture. In addition to a national back-end system with relevant information, other coalition partners might have additional information of interest to the decision maker. An integrated view of all relevant information is presented using the Semantini demonstrator.

As services come and go, the decision maker searches for currently available information services. Following the Web Service (WS) lifecycle described in Section 4.1.3 and depicted in Figure 4.3, the steps of service discovery and invocation follows:

- The national back-end system delivering surveillance information is described as a WS, and annotated using OWL-S in a semantic *service description*. Relevant information services from other coalition partners are also described using OWL-S. The information services are in this case a NATO Friendly Force Information (NFFI) service and a logistics information system delivering information about friendly logistic activities.
- The service descriptions are *advertised* in a registry residing at the HQ.
- Service advertisements are disseminated, and the military decision maker can *discover* available services.
- After discovering the available services, the military decision maker have to *select* amongst the services, and searches the registry sending a semantically defined request for surveillance information services. (Selection of services were demonstrated with direct match and subsumed match[12]).

[12] Subsumed match: match is done on a direct descended node in the graph

- Found services are *invoked* in sequence by the decision maker and results are shown in an updated view of the area with both national as well as available coalition information about the area. The client program showing the information had no knowledge of the services at design time, but the semantic descriptions enabled invocation and utilisation of the services giving the decision maker a more complete situational picture of the area, helping him guide the reconnaissance team.

The result of this SWS discovery and invocation lifecycle is a combined view of the available surveillance information, as shown in Figure 8.1.

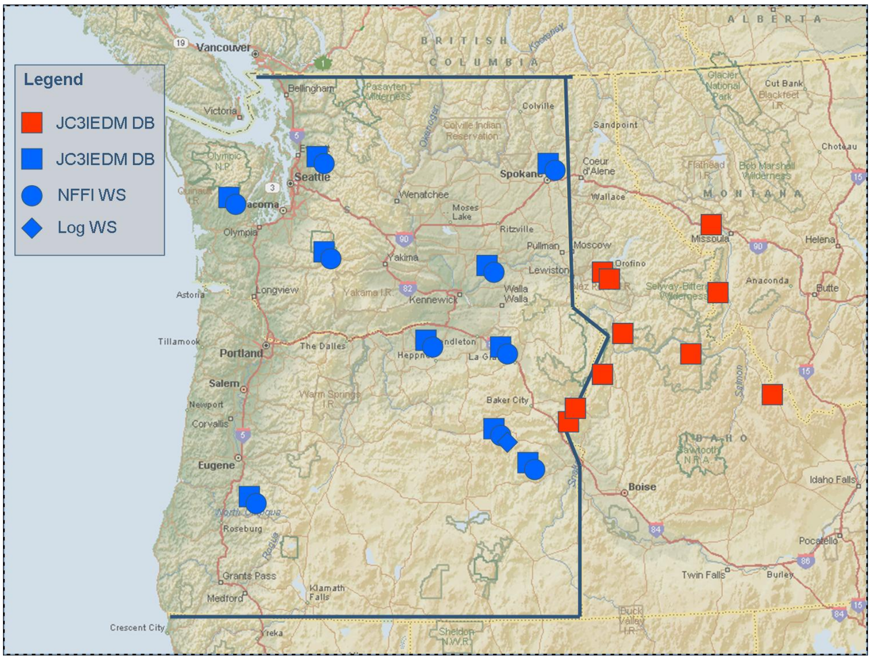


Figure 8.1 A geographical view of the information gathered from available sources

Concluding remarks on the Semantini CWID-08 experiment are that SOA implemented as SWS has great potential value in the service environment, and that the experiment showed that this is possible in a military setting. Valuable results were found annotating external WS within a reasonable amount of time, discovering services using both direct match and subsumed match, and services (not necessarily known at design time) were invoked successfully. Problems encountered during the experiments in regards to SWS are the limited tool support, and that the tools available are immature and incomplete.

8.2 DEMO-2010, Semantic Web Services Orchestration

SOA has shown to be suitable for dynamic networks (Lund et al. 2007). In situations where heterogeneous units come together to perform a joint operation such as an international coordinated military operation, having the ability to quickly set up a functional information sharing capability is essential. There is, however, a number of challenges related to implementing SOA using WS in dynamic networks, one of which is to find services available in the network. Participating nodes can be mobile, and they will be likely to experience both network partitioning and/or loss of connectivity. A unit will need to access locally available services, even when connections to other networks or other partitions of the same network fail. In such a scenario, relying on a centrally located registry for WS can lead to the loss of the discovery capability if the connection to the registry goes down. Units in the network will thus be prevented from finding local services even if these services are still available. However, by utilising WS technology coupled with semantic technologies, we can facilitate automated service selection, orchestration, and invocation.

In collaboration with the FFI SOA project, a Semantically Enabled Service Discovery and Orchestration system for mobile ad-hoc Networks (MANETs) was implemented for demonstration purposes at DEMO-2010, an FFI event showcasing potential use of future technology. The SOA project implemented a novel mechanism, service advertisements in MANETs (SAM) (Johnsen 2009), to distribute the service descriptions. In MANETs, users might experience loss of connection to a registry and also the coming and going of services. With these MANET clients in mind, we did not use a registry for matching but performed service matching and selection at the client's equipment. We demonstrated service selection based on client requests, and orchestration if no single service fulfilled the request. Orchestration resulted in a new service description defining the new combined service. We also demonstrated automatic seamless invocation of the participating services combined in the orchestration.

To exemplify the rapidly changing service environment described above, we imagine that during the military operation described in the previous Section 8.1, a reconnaissance team is moving towards a destination. The reach-back link to HQ goes down and the team has to find another way to move to their destination. The team leader then needs to access a new information source in order to get updated information about other units in the area, helping her to guide the team safely to their destination. As an example, imagine a UAV entering the operations area. This UAV mission may not have been planned when the soldier left the base, thus the soldier may not have the necessary client software to utilise this new service.

Following the WS lifecycle described in Section 4.1.3 and depicted in Figure 4.3, we illustrate how this problem is solved in our demonstration software:

- Traditional WSDL service descriptions and OWL-S *service descriptions* are pre-distributed to the soldier's computer equipment.
- When our soldier receives an *advertisement* from the UAV, the appropriate service descriptions (WSDL and OWL-S) are resolved and added to the local list of available services. The service descriptions are tagged with a time-to-live and removed accordingly if no new liveness information is received.
- Available services are *discovered* and are ready for selection.
- The UAV service is now available for *selection* together with other advertised services. Our

soldier needs surveillance footage of the area. Earlier his client software has used another footage service, but it is no longer available. Our system searches the available services based on expected input (a position, lat/long) and expected output (picture format, jpg).

- Searching the service list there is no simple service matching the request. An *orchestration* process is started, and based on the input and output concepts the system finds that a combination of the UAV picture service and an available picture format conversion service is able to deliver the requested service. The UAV picture service has position as input and picture format gif as output, while the conversion service accepts gif as input and delivers jpg as output. A new OWL-S service description is produced, added to the list of services and used to fulfil the soldiers request.

- The new composite service description is used by the client software when it first *invokes* the UAV picture service and then the conversion service. The client software receives the footage without having to manually run both services, and without human intervention.

Figure 8.2 shows the requested surveillance picture from the UAV, converted from gif to jpg and shown to the team leader.

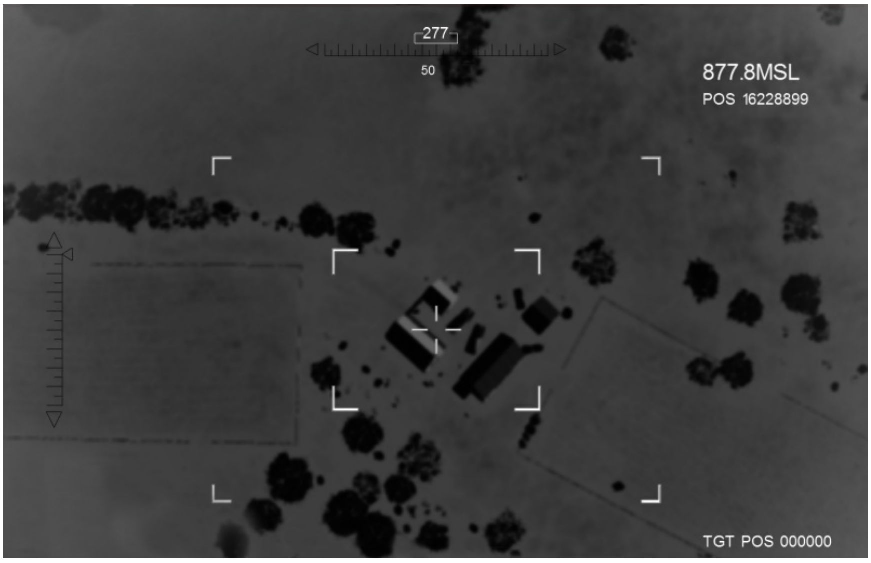


Figure 8.2 Surveillance picture from the requested area

Concluding remarks on the SWS experiment in DEMO-2010 are that SWS have good indications of value in ad-hoc SOA environments where connections to registries and services come and go. The experiment proved that SWS service selection, orchestration, and invocation is possible in this military setting. We demonstrated finding services, orchestrating new services if needed and possible with the resulting new service description, and automatic invocation of intermediary service connections. Problems encountered during the experiment were still the limited tool support, as it was at CWID-08. Some of these tool related problems were rectified at CWID-08, but there is still a shortage of mature tools helping SWS development.

9 Other Potential Application Areas

The main body of this report has discussed use of semantic technology for decision support, information integration, and in a service infrastructure. This section points to some other potential application areas that have not been the focus of any research activities in the Semantini project, but nevertheless deserve to be considered.

9.1 Information Management

One can experience a wide range of different approaches to the discipline called information management (IM). Some focus purely on managing the information content as such, while others also include enabling technologies and consider aspects of how to manage information systems. Providing right information to the right people at the right time, may be viewed as the ultimate goal.

It has been suggested that semantic technologies may contribute to improved IM. Metadata and formal models, flexibility and adaptivity, ability to integrate information from heterogeneous sources: It all fits in as support elements for management of information. The challenge of specifying how, has to be pointed to as recommended future work. An effort of matching IM requirements and enabling technologies would hopefully identify areas for further studies or experimentation.

9.2 System Architecture

NATO Architecture Framework (NAF) is expected to be at the core of Norwegian Armed Forces' activities regarding system architecture the coming years. NAF defines ways of expressing models in a standardised manner, aiming for recognition effects and with a strong urge to reuse modelling artifacts. Models can be exchanged, and there are plans for model repositories. The predominant language used within this discipline is Unified Modelling Language (UML).

The modelling community itself has already pointed to ontologies as the future way of formalising concepts and definitions in an unambiguous way. Again, without stating clearly how (and when), it is fair to believe that elements of semantic technologies can be successfully applied within this area.

9.3 Ontology Use in Traditional Data Modelling

Data models have been an integral part of application development for the last decades, implying that data modelling is a well known activity. Ontology is a fairly new term outside of the academic world. The first question is normally: What is the difference between an ontology and a data model? If we want to underline what is in common, we can say that an ontology is a more formal representation than a data model, but the process of creating an ontology has many similarities to that of creating a data model.

In real life, the biggest difference might be experienced through the different tool sets that are used.

Ontology development is usually done with a completely different set of tools than is normal for data modelling.

Ontologies are considered a more powerful representation than data models, since they are more formal and thus enable increased computer processing. There exist tools that can be used to “lift” traditional data models to ontologies, but it is likely that some additional manual work is required to actually add extra value. It is also likely that the result is not in a form that ontologists are happy with, but that might be a result of the original data modelling philosophy, not the conversion itself.

An example is the Joint Consultation, Command and Control Information Exchange Data Model (JC3IEDM), produced in the Multilateral Interoperability Programme (MIP). The model is created using the Entity-Relationship (ER) paradigm. There also exists an ontology version of the model. This version, however, is a direct conversion from ER, and is not in the same form as it most likely would have been if it had been created as an ontology in the first place. Actually, already by converting it from ER to UML, it is clear that there is a difference in modelling philosophy.

9.4 Semantic Interoperability

Semantini has during its three years of existence taken part in the work of a NATO research group with the same time-span. The group IST-075 ’Semantic Interoperability’ has had an ultimate goal of ’information exchange between computers without loss of meaning’. That is very ambitious, and the first step on the way has become the Semantic Interoperability Logical Framework (SILF), which is described in the final report from the group (Bacchelli et al. 2010). A successor group IST-094 is set up to continue the efforts for another three years. The key to implementation of the proposed framework, is exploiting the properties of semantic technologies.

IST-075 has focused on automated transformations of information according to the more closely defined ontologies of each of the exchange partners. While this work has assumed each partner to be a computerised system, related work have focused on definitions of terms being used in information exchange between humans and systems. The European Union initiative SEMIC (SEMIC 2010) and the Norwegian counterpart SEMICOLON (SEMIColon 2010), are important contributors. They aim to establish compatible sets of concepts and terms to be used within the public sector.

Concepts, terms, and vocabularies are relatively well defined within the military community. The formalised characteristics of military organisations open perspectives of computerised support in order to improve semantic interoperability. Given that it seems impossible to fully avoid misunderstandings between people, the progress towards semantic interoperability might better be labelled as ’improving’ rather than fully ’achieving’. Interoperability is very important, and all improvements are valuable.

9.5 Handling Unstructured Information

The work in Semantini has been restricted to the handling of information that has a structure. Examples are described fields in a database, XML structures, etc. In contrast, the vast majority of existing information volumes is of a kind that gives the human user a very good understanding of the meaning but is meaningless to a computer. Text, images, audio or any kind of sensor input must as a minimum be annotated by metadata before we can speak of any 'meaning' in computer terms.

Computer programs do process text, audio and images. There are specialised software for text pronunciation, automated texting and image recognition. We have so far not considered these specialised programs as belonging to the family of semantic technologies. Neither has Semantini done any work along these lines.

Parsing of unstructured information will often require - or shall we say benefit from - manual input, at least as a quality assurance step. Text will sometimes be ambiguous, image and audio quality has limits, etc. Fully correct interpretations do not always exist.

The point here is that there are necessary steps to be performed before unstructured information achieves the necessary structure and thus can be handled by semantic technologies. Apart from the practical limitations of interpretation, we can think of it as preprocessing. Given the high expectations to the power of semantic technologies, it is important to be aware of the limitations in how to automatically extract meaning from available unstructured information.

10 Recommendations and Conclusion

Semantic technologies provide means to automate more tasks in order to assist a human user. As in all cases regarding automation of previously man-performed tasks, an important step is to identify which of them are suited to be automated. In a military setting, actions and decisions taken might have severe consequences. Considering the ethical and political factors, not every task is a candidate for automation.

This report describes a step on the way towards exploitation of semantic technologies. Further research and development is necessary before semantic technologies can result in applications for operational use in the Norwegian Armed Forces. Nevertheless, it is important to be aware of the potential of these technologies. Future system requirements should reflect these possibilities.

The focus of Semantini has been to explore technical possibilities. The potential benefits are promising, and hopefully the near future will bring improvements in tools and support systems for these technologies. More experience is necessary before cost-benefit aspects of the technical possibilities can be calculated with sufficient precision.

Semantini recommends use of semantic technologies in three application areas:

- decision support systems, including intelligence analysis solutions
- information integration solutions
- service infrastructures

These areas are described in dedicated sections of this report. Descriptions include recommended further research activities within each area. As initial steps, Semantini has set up demonstrators in order to verify thoughts about how to build systems that will exploit the properties of semantic technologies in these areas. Sections 7 and 8 describe experimental solutions showing some of the potential benefits that can be achieved.

Semantic technologies promise more adaptive and flexible software, where system behaviour to a larger extent is driven by models. The abilities of these technologies should be considered in early stages of information systems development. Candidate systems that are expected to benefit strongly from these properties, should in turn be evaluated as potential pilot projects for value-adding use of semantic technologies. Research and development resources from industry should preferably be joined in such efforts.

The value of semantic technologies lies partly in their expected future widespread use. From a larger user community will hopefully follow powerful tools and methods of industrial strength. It is expected to happen, but at this point in time it is difficult to predict when. From the perspective of the Norwegian Armed Forces, insightful awareness and consistent skill-building is a recommendable approach to semantic technologies.

References

Alexander, K., Cyganiak, R., Hausenblas, M. & Zhao, J. (2009), Describing Linked Datasets, *in* C. Bizer, T. Heath, T. Berners-Lee & K. Idehen, eds, 'Proceedings of the WWW2009 Workshop on Linked Data on the Web', Vol. 538 of *CEUR Workshop Proceedings*.

Allemang, D. & Hendler, J. A. (2008), *Semantic web for the working ontologist: modeling in RDF, RDFS and OWL*, Morgan Kaufmann.

ARC-Development (2009), 'Easy RDF and SPARQL for LAMP systems - ARC RDF Classes for PHP', http://arc.semsol.org/home.

Aumuller, S. A. D. & Dietzold, S. (2009), 'Triplify - Expose semantics!', http://triplify.org/Overview.

Baader, F., Bauer, A., Baumgartner, P., Cregan, A., Gabaldon, A., Ji, K., Lee, K., Rajaratnam, D. & Schwitter, R. (2009), A Novel Architecture for Situation Awareness Systems, *in* M. Giese & A. Waaler, eds, 'Proceedings of the 18th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX)', Vol. 5607 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, Heidelberg, pp. 77–92.

Bacchelli, F., Boury-Brisset, A.-C., Isenor, A., Kuehne, S., Reif, B. M., Miles, J., Mojtahedzadeh, V., Poell, R., Rasmussen, R., Uzunali, A. & Wunder, M. (2010), Semantic interoperability – Final Report from RTO IST-075 - TG-034, Technical report, NATO Research and Technology Organization.

Baumgartner, P. & Thorstensen, E. (2010), 'Instance based methods — a brief overview', *KI - Künstliche Intelligenz* . To appear.

Beckett, D. & Berners-Lee, T. (2008), 'Turtle - Terse RDF Triple Language', http://www.w3.org/TeamSubmission/turtle/.

Bergman, M. K. (2010), 'Sweet Tools (Sem Web)', http://www.mkbergman.com/new-version-sweet-tools-sem-web/.

Bergman, M. K. & Giasson, F. (2009), 'Upper Mapping and Binding Exchange Layer (UMBEL)', http://www.umbel.org/index.html.

Berners-Lee, T. (2000), 'Primer: Getting into RDF & Semantic Web using N3', http://www.w3.org/2000/10/swap/Primer.

Berners-Lee, T. (2006), 'Linked data', http://www.w3.org/DesignIssues/LinkedData.html.

Berners-Lee, T., Hendler, J. & Lassila, O. (2001), 'The Semantic Web', *Scientific American* **284**(5), 28 – 37.

Bizer, C. & Cyganiak, R. (2009), 'Publishing relational databases on the semantic web', http://www4.wiwiss.fu-berlin.de/bizer/d2r-server/.

Brachman, R. & Levesque, H. (2004), *Knowledge Representation and Reasoning*, Elsevier/Morgan Kaufmann, Amsterdam.

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N. & Weerawarana, S. (2002), 'Unraveling the Web services Web, An Introduction to SOAP, WSDL, and UDDI', In IEEE Internet Computing, vol. 6, no. 2.

Cycorp (2009), 'OpenCyc', http://www.opencyc.org/.

Duke, A. & Richardson, M. (2009), Semantic Integration for B2B Service Assurance, *in* R. García, ed., 'Semantic Web for Business. Cases and Applications', Information Science Reference, pp. 50 – 64.

Euzenat, J. & Shvaiko, P. (2007), *Ontology matching*, Springer-Verlag, Heidelberg (DE).

Fielding, R. T. (2000), Architectural styles and the design of network-based software architectures, PhD thesis, University of California.

Guo, Y., Pan, Z. & Heflin, J. (2005), 'LUBM: A Benchmark for OWL Knowledge Base Systems', *Journal of Web Semantics* pp. 158 – 182.

Haarslev, V. & Moller, R. (2001), Description of the RACER system and its applications, *in* 'Proceedings International Workshop on Description Logics (DL-2001)', pp. 132–142.

Hansen, B. J., Gagnes, T., Rasmussen, R. E., Rustad, M. & Sletten, G. (2007), Semantic Technologies, FFI-rapport 2007/02461, Norwegian Defence Research Establishment.

Hansen, B. J., Halvorsen, J., Kristiansen, S. I. & Rasmussen, R. E. (2008), Experiment report: "Semantic SOA" – NATO CWID 2008, FFI-rapport 2008/01557, Forsvarets forskningsinstitutt.

Hitzler, P., Krötzsch, M. & Rudolph, S. (2009), *Foundations of Semantic Web Technologies*, Chapman & Hall/CRC, Boca Raton.

InnoQ (2007), 'Web services standards overview', `http://www.infoq.com/news/2007/03/innoq-ws-standards-poster`.

Johnsen, F. T. (2009), 'An nffi-based web service discovery mechanism for tactical networks', Military Communications and Information Systems Conference (MCC 2009), Prague, Czech Republic.

Josuttis, N. M. (2007), *SOA in Practice*, O'Reilly Media.

Kimball, R. & Caserta, J. (2004), *The Data Warehouse ETL Toolkit*, Wiley.

Lacoul, S. (2009), 'Leveraging the Semantic Web for Data Integration', *Cutter IT Journal* **22**(9), 19 – 23.

Library for Applied Ontology (2009), 'Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE)', `http://www.loa-cnr.it/DOLCE.html`.

Lund, K., Eggen, A., Hadzic, D., Hafsøe, T. & Johnsen, F. T. (2007), 'Using web services to realize service oriented architecture in military communication networks', *IEEE Communications Magazine* **45**, 47–52.

Motik, B. & Sattler, U. (2006), A comparison of reasoning techniques for querying large description logic aboxes, *in* 'Proc. of the 13th Int. Conf. on Logic for Programming Artificial Intelligence and Reasoning, Phnom Penh, Cambodia'.

Motik, B., Shearer, R. & Horrocks, I. (2009), 'Hypertableau Reasoning for Description Logics', *Journal of Artificial Intelligence Research* **36**, 165–228.

Motro, A. & Anokhin, P. (2006), 'Fusionplex: resolution of data inconsistencies in the integration of heterogeneous information sources', *Information fusion* **7**(2), 176–196.

Nardi, D. & Brachman, R. J. (2003), *An introduction to Description Logics*, Cambridge University Press, chapter 1, pp. 5 – 44.

NC3A (2005), ‘Nato network enabled capability feasibility study v.2’.

Niles, I. & Pease, A. (2001), Towards a Standard Upper Ontology, *in* C. Welty & B. Smith, eds, ‘Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)’.

ONTORULE (2009), ‘ONTORULE - WP3 Execution and Inference: D3.1 State-of-the-art survey of issues’. Available at `http://ontorule-project.eu/deliverables-and-resources?func=select&id=6`.

Patil, S. & Newcomer, E. (2003), ‘ebxml and web services.’, *IEEE Internet Computing* **16**, 74–82.

Polleres, A., de Bruijn, J., Bussler, C., Domingue, J., Fensel, D., Hepp, M., Keller, U., Kifer, M., König-Ries, B., Kopecky, J., Lara, R., Lausen, H., Oren, E., Roman, D., Scicluna, J. & Stollberg, M. (2005), ‘Web service modeling ontology(wsmo) submission’.

Rohloff, K., Dean, M. & Emmons, I. (2007), An Evaluation of Triple-Store Technologies for Large Data Stores, *in* ‘On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops’, Vol. 4806 of *Lecture Notes in Computer Science*, Springer Berlin/Heidelberg, pp. 1105 – 1114.

Russell, S. & Norvig, P. (2003), *Artificial Intelligence*, Prentice Hall/Pearson Education, Upper Saddle River.

Rustad & Gagnes (2006), A survey of semantic web services, Technical report, FFI.

Saïs, F., Pernelle, N. & Rousset, M.-C. (2007), L2R: a logical method for reference reconciliation, *in* R. C. Holte & A. Howe, eds, ‘Proceedings of the twenty-second AAAI conference on artificial intelligence’, AAAI Press.

Schlimmer, J., Beatty, J., Kakivaya, G., Kemp, D., Kuehnel, T., Lovering, B., Roe, B., John, C. S., Simonnet, G., Walter, D., Weast, J., Yarmosh, Y. & Yendluri, P. (2005), ‘Web services Dynamic Discovery (WS-Discovery)’, OASIS Draft specification, `http://specs.xmlsoap.org/ws/2005/04/discovery/ws-discovery.pdf`.

Schreiber, G., Akkermans, H., Anjewierden, A., Dehoog, R., Shadbolt, N., Vandevelde, W. & Wielinga, B. (1999), *Knowledge Engineering and Management: The CommonKADS Methodology*, The MIT Press.

SEMIC (2010), ‘SEMIC.EU - The Semantic Interoperability Centre Europe’, `http://www.semic.eu`.

SEMIColon (2010), ‘SEMICOLON - Semantisk og organisatorisk interoperabilitet i kommuniserende og samhandlende organisasjoner’, `http://www.semicolon.no`.

Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A. & Katz, Y. (2007), 'Pellet: A practical OWL-DL reasoner', *Journal of Web Semantics* **5**(2), 51 – 53.

Smith, B. & Grenon, P. (2009), 'Basic Formal Ontology (BFO)', `http://www.ifomis.org/bfo`.

Sowa, J. (2000), *Knowledge Representation*, Brooks/Cole, Pacific Grove.

Stoutenburg, S., Obrst, L., Nichols, D., Franklin, P., Samuel, K. & Prausa, M. (2007), Ontologies in OWL for Rapid Enterprise Integration, *in* C. Golbreich, A. Kalyanpur & B. Parsia, eds, 'OWLED', Vol. 258 of *CEUR Workshop Proceedings*, CEUR-WS.org.

Tessaris, S., Franconi, E., Eiter, T., Gutierrez, C., Handschuh, S., Rousset, M.-C. & Schmidt, R. A., eds (2009), *Reasoning Web. Semantic Technologies for Information Systems, 5th International Summer School 2009, Brixen-Bressanone, Italy, August 30 - September 4, 2009, Tutorial Lectures*, Vol. 5689 of *Lecture Notes in Computer Science*, Springer.

Theoharis, Y., Christophides, V. & Karvounarakis, G. (2005), Benchmarking Database Representation of RDF/S Stores, *in* 'International Semantic Web Conference (ISWC) 2005', Vol. 3792 of *Lecture Notes in Computer Science*, Springer Berlin/Heidelberg, pp. 685 – 701.

Thomson Reuters (2010), `http://www.opencalais.com`.

Tsarkov, D. & Horrocks, I. (2006), FaCT++ description logic reasoner: System description, *in* 'Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006)', Vol. 4130 of *Lecture Notes in Artificial Intelligence*, Springer, pp. 292 – 297.

van der Aalst, W., Hofstede, A. & Weske, M. (2003), 'Business process management: A survey', *Lecture Notes in Computer Science* **2678**, 1–12.

W3C (2000), 'OWL 2 Web Ontology Language Overview', `http://www.w3.org/TR/owl2-overview/`. W3C Recommendation.

W3C (2004*a*), 'OWL-S: Semantic Markup for Web Services', `http://www.w3.org/Submission/OWL-S/`. W3C Member Submission.

W3C (2004*b*), 'RDF Primer', `http://www.w3.org/TR/rdf-primer/`. W3C Recommendation.

W3C (2004*c*), 'RDF/XML Syntax Specification', `http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/`.

W3C (2007*a*), 'Semantic Annotations for WSDL and XML Schema', `http://www.w3.org/TR/sawsdl/`. W3C Recommendation.

W3C (2007*b*), 'Semantic Web layer cake', `http://www.w3.org/2007/03/layerCake.png`.

W3C (2008), 'SPARQL Query Language for RDF', http://www.w3.org/TR/rdf-sparql-query/. W3C Recommendation.

Walton, C. (2007), *Agency and the Semantic Web*, Oxford University Press, Oxford Oxfordshire.

Appendix A Abbreviations

AI .... Artificial Intelligence
ARBIT .... Automated Reasoning Based Intelligence Tool
BFO .... Basic Formal Ontology
BPEL .... Business Process Execution Language
C2 .... Command and Control
C3 .... Command, Control, and Communications
C4 .... Command, Control, Communications, and Computers
CWA .... Closed World Assumption
CWID .... Coalition Warrior Interoperability Demonstration
DDL .... Data Definition Language
DL .... Description Logic
DOLCE .... Descriptive Ontology for Linguistic and Cognitive Engineering
DSTO .... Defence Science and Technology Organisation
ebXML .... electronic business XML
ER .... Entity-Relationship
FFI .... Forsvarets forskningsinstitutt (Norwegian Defence Research Establishment
FOL .... First-Order Logic
HQ .... Headquarters
HTTP .... HyperText Transfer Protocol
IDEF-x .... Integration DEFinition, a family of modelling languages
IM .... Information Management
ISWC .... International Semantic Web Conference
JC3IEDM .... Joint Consultation, Command and Control Information Exchange Data Model
KB .... Knowledge Base
KR .... Knowledge Representation
LD .... Linked Data
LOD .... Linked Open Data
LP .... Logic Program
MANET .... Mobile Ad Hoc Network
MIP .... Multilateral Interoperability Programme
NAF .... NATO Architecture Framework
NATO .... North Atlantic Treaty Organization
NBD .... Network-Based Defence
NC3A .... NATO C3 Agency
NFFI .... NATO Friendly Force Information
NII .... Networking and Information Infrastructure
NNEC .... NATO Network Enabled Capability

NP .... Nondeterministic Polynomial time
OASIS .... Organization for the Advancement of Structured Information Standards
OWA .... Open World Assumption
OWL .... Web Ontology Language
OWL-S .... OWL for Services
QoS .... Quality of Service
RDF .... Resource Description Framework
REST .... Representational State Transfer
SAM .... Semantic Advertisements in MANETs
SAWSDL .... Semantic Annotations for WSDL
Semantini .... Semantiske tjenester i INI - Semantic Services in the Information Infrastructure
SILF .... Semantic Interoperability Logical Framework
SOA .... Service-Oriented Architecture
SOAP .... Originally defined as Simple Object Access Protocol
SPARQL .... SPARQL Protocol and RDF Query Language
SQL .... Structured Query Language
SUMO .... Suggested Upper Merged Ontology
SW .... Semantic Web
SWS .... Semantic Web Services
TST .... Time-Sensitive Target
UAV .... Unmanned Aerial Vehicle
UDDI .... Universal Description Discovery and Integration
UMBEL .... Upper Mapping and Binding Exchange Layer
UML .... Unified Modelling Language
URI .... Uniform Resource Identifier
URL .... Uniform Resource Locator
URN .... Uniform Resource Name
voiD .... Vocabulary of Interlinked Datasets
W3C .... World Wide Web Consortium
WoD .... Web of Data
WS .... Web Services
WS-I .... Web Services Interoperability Organization
WSDL .... Web Services Description Language
WSML .... Web Services Modelling Language
WSMO .... Web Service Modelling Ontology
WWW .... World Wide Web
XML .... eXtensible Markup Language