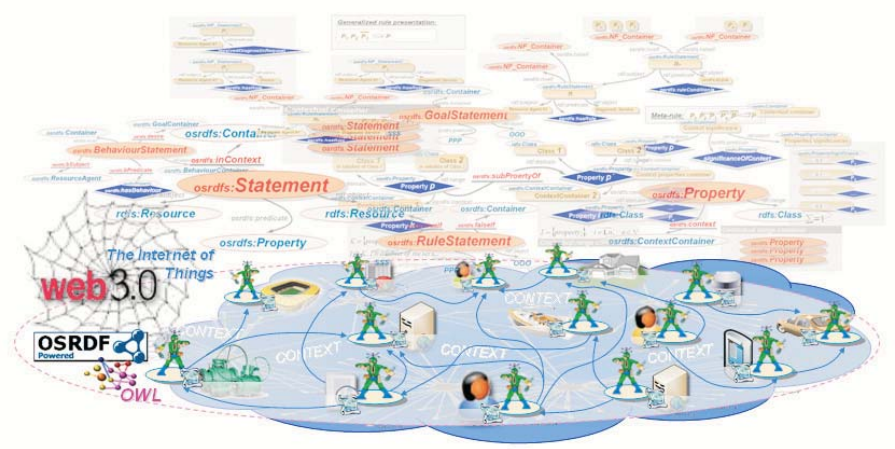
JYVÄSKYLÄ STUDIES IN COMPUTING
97

Oleksiy Khriyenko

Adaptive Semantic Web based Environment for Web Resources



JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 97

Oleksiy Khriyenko

Adaptive Semantic Web based Environment for Web Resources

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi Agora-rakennuksessa (Ag Aud. 2)
joulukuun 13. päivänä 2008 kello 14.

Academic dissertation to be publicly discussed, by permission of
the Faculty of Information Technology of the University of Jyväskylä,
in the building Agora, Ag Aud. 2, on December 13, 2008 at 2 p.m.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2008

Adaptive Semantic Web based Environment for Web Resources

JYVÄSKYLÄ STUDIES IN COMPUTING 97

Oleksiy Khriyenko

Adaptive Semantic Web based Environment for Web Resources

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2008

Editors
Timo Männikkö
Department of Computer Science and Information Systems
Pekka Olsbo, Marja-Leena Tynkkynen
Publishing Unit, University Library of Jyväskylä

Cover Picture by Oleksiy Khriyenko

URN:ISBN:978-951-39-3446-0
ISBN 978-951-39-3446-0 (PDF)

ISBN 978-951-39-3410-1 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2008

ABSTRACT

Khriyenko, Oleksiy
Adaptive Semantic Web based Environment for Web Resources
Jyväskylä: University of Jyväskylä, 2008, 193 p.
(Jyväskylä Studies in Computing, ISSN 1456-5390; 97)
ISBN 978-951-39-3446-0 (PDF), 978-951-39-3410-1 (nid.)
Finnish Summary
Diss.

We are entering era of ubiquitous computing and communication. In the new Internet of Things interactions occur, not only between humans and applications, but also between applications of various kinds, applications and equipment with embedded software or any other logical or physical entities. To manage this heterogeneous and dynamic Internet of Things we definitely need explicit semantics, even more than the traditional Web – for automatic discovery and interoperability among heterogeneous resources, for inference on implicit data, and also to facilitate the behavioral coordination of the components of complex physical-digital systems.

The "resources" to be annotated semantically are no more limited to documents, web-pages and services in the Web. Variety of resources/things (devises, machines, services, human-experts, processes, organizations and realworld objects) will be connected to the IT systems. This not only increases the amount of resources, but also introduces new classes of properties to be described. Resource of the new Web is a *proactive* goal-driven *dynamic* entity, which reacts autonomously on changes within its external environment or within itself. As a consequence of resources' dynamism and proactiveness, the environment itself becomes more dynamic. Such dynamism and proactiveness require *context-awareness* from the system, as more and more statements and behaviours become context-dependent.

The existing solutions are not sufficient in the new situation. The basic tool of Semantic Web, Resource Description Framework lacks semantics to describe new dynamic and proactive resources. On the other hand, languages developed for Agent behaviour modelling, unfortunately, follow a set of various different standards and lack a common standard and semantics.

In this work we show that it is possible to enrich RDF to enable description of proactive and dynamic resources and to present context-sensitive information. Our elaborated formalism (OSRDF) enables to represent rules, plans and behaviours within the same data representation model. At the same time it provides an approach to adapting such new environments to be natural for humans.

Keywords: Semantic Web, context-sensitive metadata description, contextual extension of RDF, resource adaptation, resource proactivity, context-aware GUI

Author's address Oleksiy Khriyenko
Department of Mathematical Information Technology
University of Jyväskylä
P.O.Box 35
FIN-40014 Jyväskylä, Finland
oleksiy.khriyenko@jyu.fi

Supervisors Prof. Dr. Vagan Terziyan
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Prof. Dr. Timo Tiihonen
Department of Mathematical Information Technology
University of Jyväskylä
Finland

Reviewers Prof. Tatiana Gavrilova
Graduate School of Management
St.Petersburg State University
Russia

Prof. Marie Duzi
Department of Computer Science
Ostrava Technikal University
Czech Republic

Opponent Dr. Evgeny Osipov
Department of Systems and Interaction
Luleå University of of Technology
Sweden

ACNOWLEDGEMENTS

First of all I am grateful to my parents Olena and Volodymyr, and my loved sister Nataliya for having raised and encouraged me to enable me to reach this point in my life, for their unconditional support, their love and belief in me.

I would like to thank the Department of Mathematical Information Technology of the University of Jyväskylä, where I have had the fortune to conduct my research since 2004. In addition I would also like to acknowledge some earlier period. I can remember, my parents and grandparents cultivated my interest in exact sciences and logic. Then, teachers and administration in the schools that I attended did their best to create a favorable environment for further developing of my skills. I appreciate their efforts very much, and I am really grateful to them. In 1998 when I was enrolling in the Kharkiv National University of Radio Electronics in Ukraine I already knew that I would like to innovate, to create something new for the benefit of mankind. That is why I chose the Intelligent Decision Support Systems line, which was more science-oriented. I wish to express my gratitude to the professors and doctoral students who managed that IDSS line and immersed me into the world of scientific research. Four years later I found an opportunity to continue my Masters studies and get a Doctoral degree abroad at the University of Jyväskylä in Finland. For this, I would like to thank Dr. Olena Kaykova, Prof. Vagan Terziyan and Prof. Timo Tiihonen, people who made this possible by organizing a student exchange programme between the University of Jyväskylä and Kharkiv National University of Radioelectronics, Ukraine. I am especially grateful to Timo Tiihonen for his helpful support to the members of this exchange program and, particularly, to our Industrial Ontologies Group.

I thank my supervisors Professors Vagan Terziyan and Timo Tiihonen for our creative discussions, for their unwavering guidance, careful criticism and friendly attitude.

I would like to thank all the co-authors of the papers related to this work: Prof. Vagan Terziyan, Oleksandr Kononenko, Andriy Zharko, Olena Kaykova, Anton Naumenko and Dmytro Kovtun. I am very grateful to them and to other members of the Industrial Ontologies Group (Sergiy Nikitin, Yaroslav Tsaruk, Artem Katasonov, Oleksiy Loboda, Kostyantyn Zagaynov, Andriy Taranov, Olga Nagula, Olga Klochko and Jari Marttinen), with whom I have been working in different projects during this period, for their fruitful cooperation.

During the doctoral research I participated in several research projects, and I highly appreciate the financial support I received from them. Most of this research was conducted as part of the SmartResource ("Proactive Self-Maintained Resources in Semantic Web") project in Agora Center (University of Jyväskylä, Finland). It was financially supported by the National Technology Agency of Finland (TEKES) and an industrial consortium of cooperating companies consisting of ABB, Metso Automation, TeliaSonera, TietoEnator and Jyväskylä Science Park. Among others contributing towards the research there

are “InBCT: Innovations in Business, Communications and Technology” (a TEKES project in Agora Center at the University of Jyväskylä), “IdeaMentoring: Refining research ideas to the new business opportunities" and “IdeaMentoring II” (Nokia projects supported by the Jyväskylä Science Park), and “UBIWARE: Smart Semantic Middleware for Ubiquitous Computing" (a Tekes research project supported by the industrial consortium of ABB, Fingrid, Metso Automation, Metso Shared Services, Inno-W and Hansa Ecura). Equally, I highly appreciate the financial support from the Graduate School in Computing and Mathematical Sciences (COMAS) and the support given to me by its director Pekka Neittaanmäki, thanks to which I was able to concentrate more on the scientific part of the research and on finalizing this dissertation. Additionally, a grant from Research and Training Foundation of TeliaSonera Finland Oyj in 2004 has allowed me to elaborate some ideas and publish a journal paper, which is included as one of the case examples in this thesis.

I would like to thank the reviewers of this dissertation, Prof. Tatiana Gavrilova and Prof. Marie Duzi, for their proficient and very useful comments in their review of this thesis, and the opponent, Dr. Evgeny Osipov. My gratitude goes also to Steve Legrand for correcting the language of the manuscript.

Last, but not least, I wish to thank all my friends for their support and friendship.

Jyväskylä
November 2008

Oleksiy Khriyenko

ABBREVIATIONS

ACL	Agent Communication Language
AML	Agent Modeling Language
API	Application Programming Interface
APL	Agent Programming Language
BDI	Beliefs, Desires and Intentions
BPEL4WS	Business Process Execution Language for Web Services
BRML	Business Rule Markup Language
CDF	Context-sensitive Description Framework
CHI	Computer-Human Interaction
C-OWL	Context Web Ontology Language
CORBA	Common Object Request Broker Architecture
DCOM	Distributed Component Object Model
DLs	Description Logics
ebXML	Electronic Business using eXtensible Markup Language
fDLs	Fuzzy Description Logics
FIPA	Foundation for Intelligent Physical Agents
GAF	General Adaptation Framework
GNF	General Networking Framework
GPF	General Proactivity Framework
GPRS	General Packet Radio Service
GPS	Global Positioning System
GUI	Graphical User Interface
GUN	Global Understanding eNvironment
HRAP	Human-Resource AdaPter
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
ICT	Information and communication technology
IOG	Industrial Ontologies Group
JADE	Java Agent DEvelopment Framework
MAS	Multi-Agent System
MDA	Model-driven architecture
MMS	Multimedia Messaging Service
MRDF	Mobile RDF
OOP	Object Oriented Programming
OSRDF	OntoSmartResource Description Framework
OSRDFS	OSRDF Schema
OWL	Web Ontology Language
OWL-S	Semantic Markup for Web Services
RDF	Resource Description Framework
RDQL	RDF Data Query Language

RDSMS	Resource Description Semantic Maintenance System
RFID	Radio-Frequency IDentification
RG/BDF	Resource Goal/Behaviour Description Framework
RG/BDFS	RGBDF Schema
RMI	Remote Method Invocation
RS/CDF	Resource State/Condition Description Framework
RS/CDFS	RSCDF Schema
RuleML	Rule Markup Language
S-APL	Semantic Agent Programming Language
SemaSM	Semantically enhanced Smart Messaging
SMS	Short Message Service
SMM	Smart Message Manager
SQL	Structured Query Language
SSA	Search Assistant/Facilitator
SWRL	Semantic Web Rule Language
TCP	Transmission Control Protocol
TIL	Transparent Intensional Logic
UI	User Interface
UML	Unified Modeling Language
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WSMO	Web Service Modeling Ontology
WSDL	Web Service Definition Language
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language
XSL	stylesheet language for XML
XSLT	XSL transformation

LIST OF FIGURES

CONTENTS

CHAPTER 1

INTRODUCTION

1 PROBLEM STATEMENT AND RESEARCH QUESTIONS

The Web is changing rapidly. We are about to enter a new era of ubiquitous computing and communication that will radically transform our corporate, community, and personal spheres. Tomorrow's world of Ubiquitous Pervasive Computing and Internet of Things (see FIGURE 1) is a technological revolution that represents the future of computing and communications and its development depends on technical innovations in a number of important fields. Development of them enables new forms of communication between people and things, and between things themselves. It adds a new dimension to the world of information and communication technologies (ICTs): from anytime, any place connectivity for anyone, we will now have connectivity for anything (ITU, 2005).

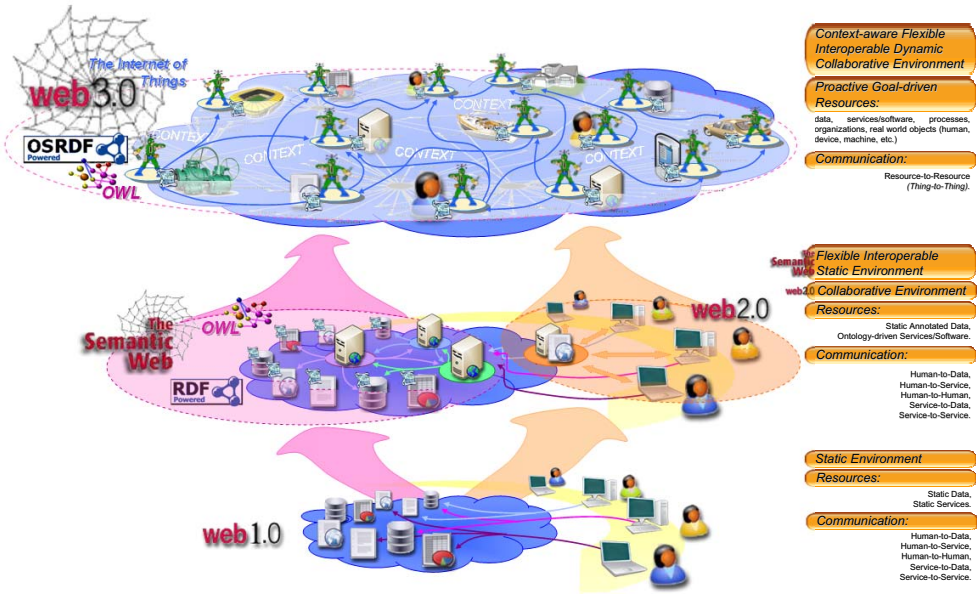


FIGURE 1 Evolution of the Web

In the development of Semantic Web tehnology research activities have been mainly focused on machine to machine interaction, on the development of the infrastructure that allows computers "understand" each other. Semantic Web is a place where machines can read Web pages much as we humans read them, a place where search engines and software agents can better explore the Net and find what we're looking for.

Unfortunately the question of human interaction with such intelligent environment of computers has been left without proper research. Further, focus has been shifted to the problems of human in the Web and appears as Web 2.0 social network. Web 2.0 is the buzzword that resumes the main changes that come from the maturity of Internet and the tools that give today more freedom to users in order to create and collaborate in Internet. Web 2.0 came to describe almost any site, service, or technology that promoted sharing and collaboration right down to the Net's grass roots. But still, with the growing amount of human-procesible information, we face a need of smart software to search and filter this information.

When we are talking about Internet of Things, we have to consider an integration of these two approaches, we have to combine Semantic Web and Web 2.0 technologies and come up with third generation of the Web (see FIGURE 1). In the new Web, the meaning of "resource" is not limited to the documents, web-pages and services in the Web. Now, a much larger number of resources/things (devises, machines, services, human/experts, processes, organizations and real world objects) can get connected to the IT systems. This increases not only the amount of resources, but also introduces new classes of properties for the resources. We defined some of these properties based on elaborated Global Understanding eNvironment (GUN) concept (see FIGURE 2) (Kaykova et al., 2005a) and prototyping of the real industrial cases within the SmartResource project[1] ("Proactive Self-Maintained Resources in Semantic Web").

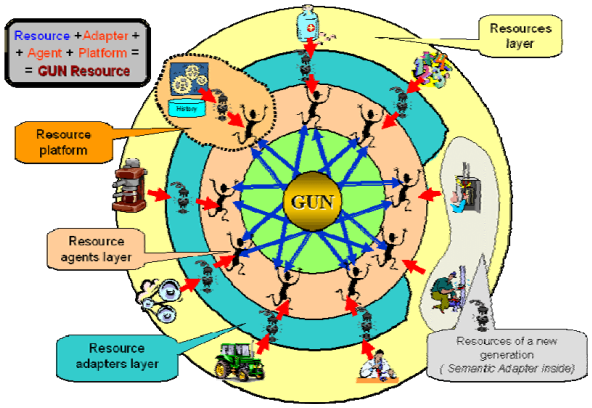


FIGURE 2 Global Understanding eNvironment concept

[1] SmartResource project (2004-2007) - http://www.cs.jyu.fi/ai/OntoGroup/SmartResource_details.htm

A resource of new Web is a *proactive* goal-driven *dynamic* entity that adequately and proactively reacts on changes within its external environment or within itself. As a consequence of resource dynamism and proactiveness, environment itself becomes more dynamic. From one side, resource's behaviour results to changes in the resource itself and to changes in the environment. In return, changes in the environment influence on the resources. Thus, such dynamism and proactiveness bring *context-awareness* to the system. It is equally important to describe the conditions of validity (context) than statements and behaviors themselves. Each statement is true only in certain context that should be described. We will name as "context" the collection of those conditions that can be observed (statements about subject resource, other resources, environment) and influence on the validity of a statement on given resource in one way or another. Thus, context is a collection of statements. As existing resource description approaches are not able to describe context sensitive information in practical way, the concepts and structures needed for processing the contextual information are the main goal of this work.

Technology advances in many different dimensions: approaches, languages, tools, and etc. In our case, a catalyst of development is a *Resource.* Extension of the meaning for "resource" and appearance of new properties and features have effect on the evolution in those dimensions. It becomes imposible to meet new demands with existing set of languages and tools, and we need to elaborate a new one or extend the existing, if the semantics has been changed. We can visualize the Resource as a multidimensional volume. Thus balanced extension of the concept necessitates development in several directions/dimensions (see FIGURE 3).

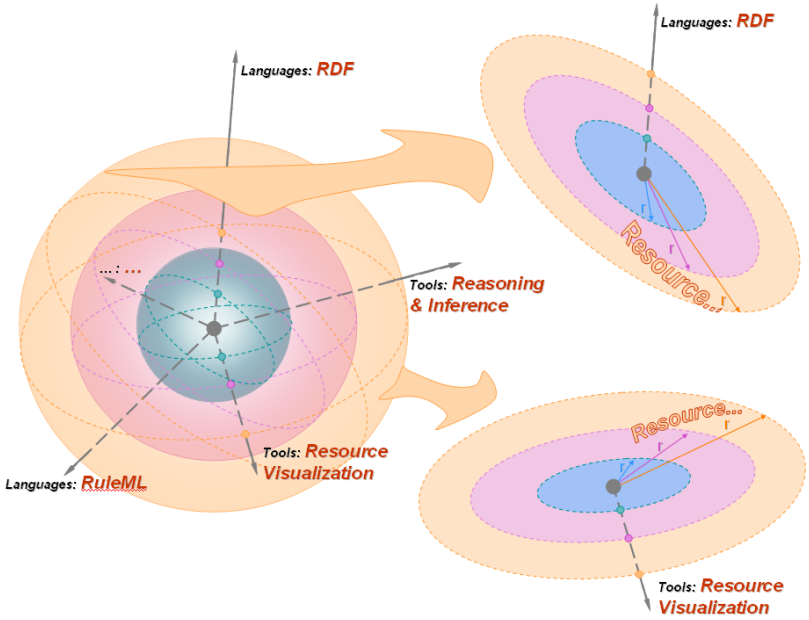


FIGURE 3 Resource – catalyst of evolutional development

According to the semantic layered cake (Berners-Lee et. al., 2001) and (Berners-Lee, 2006), we have layers that are supported by sets of languages: Metadata, Ontologies, Rules, Plans and Behaviours, Policies, Configurations, etc. Appearance of new languages causes development of tools that support them and manipulate with data, tools, which enabling: complete and compact storage of information; suitable advanced querying, reasoning and inference; automation, interoperability, sharing and integration; proactivity and self-management; effective resource visualization; etc. Now, when we start to consider machines and devices, technological and business processes, humans, communities and organizations as resources of the Web, we have changed the semantics of *Resource* notion and have to change, add or extend (if possible) existing languages and tools to meet the demands of new resources.

Integration of heterogeneous resources (applications and data sources) into an interoperable system is one of the most relevant challenges for many knowledge-based corporations nowadays. Semantic technologies are viewed today as a key technology to resolve the problems of interoperability and integration within the heterogeneous world of ubiquitously interconnected objects and systems. But still, aspects such as context and proactivity of these resources and systems are quite in demand nowadays and should be considered more comprehensively.

When we came with the idea to extend the meaning of the "resource" notion, first of all, we faced a problem of extension of a resource description language. In this research work we mainly concentrated our focus on extension of Resource Description Framework (RDF)[2] that presents Metadata Layer in Semantic Web Cake. RDF is a W3C standard for resource description purposes in Semantic Web. That is why we took it as a basis for our extension. Further, when we started to consider the behaviour description issues, we decide to elaborate a common formalism for both purposes, based on the same well known and widely used standard. We enriched this framework to enable description of proactive and dynamic resources and present context-sensitive information, to enable rule representation, plans and behaviour description with the same data representation model. As we started to consider new Web resources from Semantic Web point of view, it has been natural to study how well RDF can be extended to be applied also for behavior description of dynamic resources. The reverse process, extending some existing agent programming language with comprehensive semantic features, would in our opinion be more challenging.

According to the theory of Semantic Web, the main features of it are semantic search/quering (Data quering), inferencing of implicit data and *data visualization* (issue that has been usually ignored). To cover a whole scope of problems in new-generation Semantic Web based integration environments, we also address the problem of Human-resource adaptation via context-sensitive resource visualization approach.

---

[2] Resource Description Framework - http://www.w3.org/RDF/

The main objectives of this research and expected research results are: to determine an integration infrastructure for distributed heterogeneous resources and to determine the main resource challenges in integrative and adaptive Semantic Web based environments; to consider the context-sensitivity and proactiveness of the resources and to provide a detailed and comprehensive design of OntoSmartResource Description Framework (OSRDF) for such enriched resource description; consider a human representation in such an environment and provide the ways of human adaptation to the environment. Finally, some use cases of such adaptive integration environments will be provided.

In order to achieve all this, the following research questions were set up:

An integrated infrastructure for distributed heterogeneous resources:

- How to provide interoperability for heterogeneous resources in the web?

Smart Resources of the Semantic Web: the notion and the features:

- What is the role of a context for the resource state/condition description and knowledge representation in an intelligent integration environment?
- Regarding static and dynamic resources, what do we mean by resource proactivity and how can it be appended to a resource?
- What is meant by an abstract resource in a Semantic Web based environment?
- How can a human be represented in such an environment? What is the role of a human, and what are the ways of human adaptation to the environment?

A framework for resource description in a Smart Semantic Web based environment:

- How to enrich the existing resource description framework with context sensitiveness and resource proactiveness?

Use cases for Smart Semantic Web based environments:

- What are posible application areas for the Semantic Web based environments and what are the benefits these environments bring to today's and future information systems?

This work and the above questions comprise the author's contribution to the larger vision of Global Understanding eNvironment (General Adaptation Framework) that includes challenges like Peer-to-Peer inter-resource communication; managing and integrating distributed histories; security; self-management, configuration and integration; flexible semantic human interface; etc. (Terziyan and Zharko, 2003, Kaykova et. al., 2007, Naumenko, 2005, Nikitin et. al., 2007, Katasonov and Terziyan, 2007, Naumenko, 2007, Khriyenko, 2007d).

2 BACKGROUND AND RELATED WORK

As was mentioned before, interactions in the interconnected world of computers occur not only between humans and applications, but also between applications of various kinds, applications and equipment, low-level software units or any other logical or physical entities. In resource integration (in common case), we have to deal, in many respects, with heterogeneous resources. To enable knowledge exchange and resource integration, ontology provides a common language at a human and a machine level. Ontologies are the key technology used to describe the semantics of information exchange. They provide a shared and common understanding of a domain that can be communicated across people and application systems, and thus facilitate knowledge sharing and reuse. The underlying technology that enables these main desired features is Semantic Web (Ankolekar et al., 2002, Paolucci et. al., 2002). Semantic Web uses ontologies to create a comprehensive environment in which intelligent agents (software applications) can access annotated resources, communicate and perform collaborative activities. With the reference to the Web evalution, Internet of Things definitely needs explicit semantics, even more than traditional Web – for automatic discovery and interoperability among heterogeneous resources (things), inferencing of implicit date, and also to facilitate the behavioral coordination of the components of complex physical-digital systems. Semantic technologies are viewed today as a key technology to resolve the problems of interoperability and integration within the heterogeneous world of ubiquitously interconnected objects and systems. Semantic technologies are a qualitatively stronger approach to interoperability than contemporary standards based approaches.

At the same time, the vision of autonomic computing emphasizes that the run-time self-manageability of a complex system requires its components to be, to a certain degree autonomous themselves. We envision that the software agent technologies will play an important part in building such complex systems. Agent based approach to software engineering is also considered to be facilitating the design of complex systems. When it comes to developing complex, distributed software based systems, the agent based approach is

advocated (Jennings, 2001). From the implementation point of view, agents are the next step in the evolution of software engineering approaches and programming languages, the step following the trend towards increasing degrees of localization and encapsulation in the basic building blocks of the programming models (Jennings, 2000).

To achieve the vision of ubiquitous knowledge, the next generation of integration systems will utilize different methods and techniques. These include Semantic Web and Web Services, Agent Technologies, Mobility (Curbera et al., 2002, Clabby, 2002), and WebServices (Ankolekar et al., 2002, Paolucci et. al., 2002, FIPA, 2001).

2.1 Semantic Web technology

The Semantic Web (Semantic Web, 2001) is a logical evolution of the existing Web. It is based on a common conceptual data model of a great generality that allows both humans and machines to work with interrelated, but disjoint, information as if it were a single global database. Semantic Web aims to promote the existing Web to a qualitatively new and higher level, utilizing machine-processable metadata associated with Web resources. The next generation of intelligent applications will be capable of utilizing such resource descriptions and perform resource discovery and integration based on semantics. The Semantic Web approach is to develop, on top of the Web, a global environment with interoperable heterogeneous applications, web services, data repositories, humans, etc. Currently, time consuming search and navigation tasks have to be performed by the user and there is no easy way to automate that. Databases and catalog information are generally hidden behind HTML tabular representation or some digest page (@Semantics). The way in which the software applications and web-services are made available is also complex. This makes it tedious, often impossible, to integrate heterogeneous decentralized resources for the user. The Semantic Web aims to automate information discovery and integration. Computer programs (applications, services, agents) will be able to find and navigate today's Web resources autonomously.

Since artificial intelligence in its classical philosophical sense does not yet equal human intelligence, people need to provide a technique for machines and software which would enable some kind of "understanding" of the meaning of available information. In other words, we need to provide some machine-understandable knowledge base for the software in order to allow a more intelligent automated treatment, by that software, of the resources available in the global network. The article (Berners-Lee et al., 2001), which originally introduced the notion of Semantic Web and presented it as a research area and as the direction of Internet's future growth, appeared in Scientific American in 2001, co-authored by Tim Berners-Lee, James Hendler and Ora Lassila. The

original ideas of Berners-Lee et al. (Berners-Lee et al., 2001) were later revised by Shadbolt et al. (Shadbolt et al., 2006) with respect to the current situation. The Semantic Web will allow us to use more automated functions on the Web (such as reasoning, information and service discovery, and autonomous agents), easing the work of humans. The Semantic web will also pave the way for true device independence and customization of information content for consumers (Lassila, 2002).

World Wide Web Consortium (W3C, 2001) is developing different techniques, guidelines, software tools, specifications, etc. to lead the web to its full potential. There is a lot of academic and industrial research going on in the area of Ontologies and the Semantic Web. Many scientific achievements have already been based on a variety of Semantic Web language specifications, which are defined by W3C and other entities. Aleready there is a group of specifications for different purposes, allowing different levels of formality and semantics. Currently, the W3C consortium offers a set of language specifications, which allow structural representation of available knowledge. Some of the specifications are less formal; some include a number of restrictions and additional vocabularies allowing a representation of conceptual models of different knowledge spaces. Below a brief overview of the specifications that are the most relevant to this work:

- XML (Extensible Markup Language) (XML, 1998) provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents.
- XML Schema is a language for restricting the structure of XML documents. It also extends XML with datatypes.
- RDF (Resource Description Framework) is a framework for representing information on the web. It has an abstract flexible XML based syntax that reflects a simple graph-based data model, and formal semantics with a rigorously defined notion of entailment providing a basis for well-founded deductions in RDF data (Klyne and Carroll, 2004). The RDF consists of the RDF data model and vocabulary definition (RDF schema). A triple based simple data model of RDF is easy for applications to process and manipulate. It also is designed to be used as a base for other, more restricted ontology languages. The motivation behind the development of RDF includes the following:
    - Web Metadata - semantic annotation about web resources and the systems that use them.
    - Open information models for applications – a common language for information modeling would allow applications to share and exchange information.
    - Machine processable information - the data is processed outside the environment where it has been created.
    - Applications working together – a common language for information description allows applications to combine data from several applications to arrive at new information.
    - Automated processing of web information by software agents.

- OWL (Web Ontology Language) is designed for use by applications that need to process the content of information instead of just presenting information to humans. OWL facilitates greater machine interpretability of web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics. OWL has three increasingly expressive sublanguages: OWL Lite, OWL DL (OWL Description Logic), and OWL Full (OWL, 2004).
    - OWL Lite supports users who primarily need a classification hierarchy and simple constraints. For example, while it supports cardinality constraints, it only permits cardinality values of 0 or 1. It should be simpler to provide tool support for OWL Lite than to its more expressive relatives. OWL Lite provides a quick migration path for thesauri and other taxonomies. Owl Lite also has a lower formal complexity than OWL DL.
    - OWL DL supports those users who want the maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). OWL DL includes all OWL language constructs, but these can be used only under certain (23 of them) restrictions (for example, while a class may be a subclass of many classes, a class cannot be an instance of another class). OWL DL is so named due to its correspondence with description logics, a field of research that has studied the logics that form the formal foundation of OWL.
    - OWL Full is meant for users who want maximum expressiveness and syntactic freedom of RDF with no computational guarantees. For example, in OWL Full a class can be treated simultaneously as a collection of individuals and as an individual in its own right. OWL Full allows an ontology to augment the meaning of the pre-defined (RDF or OWL) vocabulary.

The languages presented above allow representation of knowledge spaces with weaker or stronger formal semantics. There is another group of languages designed for information extraction from the available data models represented by the above languages or their extensions. The area of these query languages is still under extensive development. There are quite a many of them, developed by different research groups, and used for querying XML, RDF, and OWL documents. Some of them have been submitted to W3C, e.g. Algae (A-RDF-QL, 2006) Buchingae (Buchingae, 2005), RuleML (The Rule Markup Language) (RMi, 2006), RDQL (A Query Language for RDF) (RDQL, 2004) among others. All theses languages define a certain syntax for constructing queries to structured documents that are mainly in the RDF format. It is obvious, that, in the long run, there should not be so many query syntaxes in use, especially keeping in mind the aim of global interoperability.

The SPARQL query language is currently under work in W3C. It consists of the syntax and semantics for asking and answering queries against RDF graphs. SPARQL contains capabilities for querying by triple patterns,

conjunctions, disjunctions, and optional patterns. It also supports constraining queries by source RDF graph and extensible value testing. The results of SPARQL queries can be ordered, limited and offset in number, and presented in several different formats (SPARQL, 2006b). Recently, W3C has been actively developing the SPARQL group of specifications, which include SPARQL Query language (SPARQL, 2006b), SPARQL Protocol (SPARQL, 2006a) and SPARQL Query XML Results Format (SPARQL, 2006c).

XPath language's primary purpose is to address parts of an XML document. It also provides basic facilities for manipulation of strings, numbers and booleans. XPath uses a compact, non-XML syntax to facilitate the use of XPath within URIs and XML attribute values. XPath operates on the abstract, logical structure of an XML document, rather than on its surface syntax. XPath gets its name from its use of a path notation, as in URLs, for navigating through the hierarchical structure of an XML document (XPath, 1999).

XQuery is another language under work by W3C. It is designed to be a language in which queries are concise and easily understood. It is also flexible enough to query a broad spectrum of XML information sources, including both databases and documents. XQuery Version 1.0 is an extension of XPath Version 2.0 (XQuery, 2006).

Unfortunately, the current state-of-arts is far from being ideal. New researches aimed to improve and inreach existing standards appear. According to the work of Marie Duzi and Anneli Heimburger (Duzi and Heimburger, 2006), the RDF approach based languages originally did not have a model theoretic semantics, which led to many discrepancies. The RDF syntax consists of the so-called triples (subject, predicate and object), where only binary predicates are allowed. This causes serious problems concerning compatibility with more expressive languages. They argue that in the Semantic Web we need a rich language with transparent semantics, in order to build up metadata on the conceptual level of the Semantic Web architecture. They propose a powerful logical tool of Transparent Intensional Logic (TIL), which provides a logico-semantic framework for a fine-grained knowledge representation and conceptual analysis, where formal knowledge specification is semantically transparent and comprehensible, with all the semantically salient features explicitly present.

Systems and tools for managing metadata repositories of RDF triples already exist. However, storing triples without being able to track back to their original source (producer of the statement) or denote the condition under which it was true is not sufficient for many applications. Especially in RDF, which provides possibility for everybody to say anything about everything, it is mandatory for the users to know the context of the given information (source, time, place and any other contextual identifier). For us, who are dealing with solutions for real industrial applications, it plays an important role. In the absence of this essential data, contradictive statements collected from a variety of sources can occur in RDF repositories, and users are not able to determine which ones they can trust. One possibility for making the RDF model more reliable in modeling context information is to use the RDF reified statements

(statements about statements, possible in RDF syntax). But this solution is not practical (MacGregor and Ko, 2003).

Bouquet et al.'s extension of the OWL language, C-OWL (Bouquet et al., 2004), has been defined to represent contextual ontologies where a context is a concrete domain viewed from the description logic perspective. In this work, ontology is contextualized when its contents are kept local and mapped with the contents of other ontologies via explicit mappings using bridge rules. These rules represent the following relations: equivalent to, more general than, less general than, compatible and incompatible. C-OWL allows a user to define an ontologies alignment where it is inappropriate to define a global shared ontology. However, the limited expressiveness of C-OWL fails to address the contextual differences found in most practical settings, as it will be shown later.

Another context representation related approach is elaborated and supported by the OpenRDF community. Sesame 2.0 also supports the notion of *context*, which we can think of as a way to group sets of statements together through a single group identifier (this identifier can be a blank node or a URI). According to Aduna (Aduna, 2007), a very typical way to use context is tracking *provenance* of the statements in a repository, that is, finding out which file these statements originate from. For example, consider an application where you add RDF data from different files to a repository, and then one of those files is updated. You would have to alter the data of that single file in the repository. In order to do that you would need a way to figure out which statements needed to be altered. But still, in our vision, this way of considering the notion of context is quite limited.

2.2 Software Agent technology

There are a huge number of academic and industrial initiatives world-wide related to agent oriented analysis. To organize these efforts, a special Co-ordination Action for Agent Based Computing, AgentLink III[3], funded by the European Commission's 6th Framework Program, was launched on the 1st of January, 2004. The AgentLink III initiative has registered more then 100 projects and even more software products based on the agent approach. Core technologies of several commercial organizations utilize different agent paradigms. For example, Whitestein Technologies[4] and Agent Oriented Software Pty Ltd[5] have provided advanced software agent technologies, products, solutions, and services for selected application domains and industries since 1999. The agent based approach has been tried in a research of industrial automation systems domain (Pirttioja et al., 2004, Seilonen, 2003).

---

3 http://www.agentlink.org/

4 http://www.whitestein.com/

5 http://www.agent-software.com/

Modeling of multi-agent systems and behaviour of concrete agents in them has been one of the most significant topics in various domains. Model-driven approach to design of agent behaviours emerged a long time ago and initially was based on UML modeling (Chella et al., 2004, Torres da Silva et al., 2004). Later this approach was extended to a level of meta-modeling (Djuric et al., 2004). As one of the mature UML-based methodologies for modeling multi-agent systems, Agent Modeling Language can be mentioned (Cervenka et al., 2005). Currently, Agent Programming Language (APL) used in commercial software projects is supported by CASE tools, and the first version of its specification has been presented to the public for its further development. One of the fundamental formal theories about behaviour in multi-agent systems (Dastani et al., 2004a) is being developed and lectured in Free University of Amsterdam[6]. Researchers have contributed various methodologies for designing multi-agent systems (MAS), including Gaia (Wooldridge et al., 2000), TROPOS (Bresciani et al., 2004), and OMNI (Vazquez-Salceda et al., 2005).

All the above efforts have elaborated the conceptual base of agent behavioural modeling and motivated its further development. There have been attempts even to elaborate a conceptual convergence of an agent layer and Web Service Architecture (Zhao et al., 2004). However, the academic efforts lack concrete details concerning methodology of modeling or have managed very preliminary prototype implementations only, as in, e.g., the Agent Academy project (Laleci et al., 2004).

Recently, the ontology-driven approach has been gaining in popularity as an alternative to the Model-driven one. It has several advantages:

- Possibility of reasoning on a level of a single model and inter-model relationships and mappings, supporting meta-model level as well.
- Flexibility support for tools (e.g., XSLT transformations) based on ontology during an evolution of the ontological model (see the analysis of evolution of classes and properties and its impact on tools in (Naumenko et al., 2005)).
- More flexible modeling framework based on a graph (Mazzocchi, 2004).

DERI is among the research centres that are very close to implementing really powerful prototypes of ontology-driven modeling for web services and multi-agent systems. Significant efforts for development of agent goal-behaviour frameworks based on the WSMO standard (ontology-driven) have been conducted by a research group from DERI according to their vision of Semantic Web (Stollberg, 2004).

Another stream of research, on individual agents, has contributed, e.g., with the well known BDI architecture, and introduced agent-oriented programming (Shoham, 1993) along with several agent programming languages such as AGENT-0 (Shoham, 1993), AgentSpeak(L) (Rao, 1996), 3APL (Dastani et al., 2004b) and ALPHA (Collier et al., 2005). All of those languages are declarative and based on the first-order logic of n-ary predicates.

---

[6] http://www.vu.nl/

As a possible option for implementing of an agents behaviour engine are Horn-like rules. W3C standardization efforts aimed at this direction have recently resulted in a family of standards: RuleML[7], SWRL[8] (Semantic Web Rule Language Combining OWL and RuleML), FOL RuleML[9] (First-Order-Logic RuleML), SWRL FOL[10] (SWRL extension to First-Order Logic). All these standards are tightly related to previous research carried out by IBM alphaWorks Labs in the development of CommonRule[11] and BRML (Business Rule Markup Language). The initiative within CommonRule was aimed at a development of a framework for specification of executable business rules by non-programmer business domain experts. The final result represents a reusable technology of business rules and rule based intelligent agents embodied as an extensible Java library. Industrial Standards, related to modeling and automation of business behaviour, are currently concentrated around BPEL4WS[12] and ebXML[13].

From the technological side, there are reliable options to form a basis for implementing frameworks for modeling behaviours in multi-agent systems: JADE-Jess-Protégé[14] and Aglets SDK[15]. In the JADE implementation several Java upper classes have been provided (JavaLIB), and this has promoted the use of the JADE platform in the implementation of tools for modeling complex agent behaviours. The JADE framework has been extended by a BDI infrastructure within the Jadex[16] project (Braubach et al., 2004), and its behavioural model was extended by Hewlett Packard Lab in their HP SmartAgent initiative (Griss et al., 2002).

---

7 http://www.ruleml.org/
8 http://www.daml.org/2003/11/swrl/
9 http://www.daml.org/2004/11/fol/folruleml
10 http://www.daml.org/2004/11/fol/
11 http://www.research.ibm.com/rules/commonrules-overview.html
12 http://www-128.ibm.com/developerworks/library/specification/ws-bpel/
13 http://www.ebxml.org/
14 http://jade.tilab.com/doc/examples/JadeJessProtege.html
15 http://www.trl.ibm.com/aglets/
16 http://vsis-www.informatik.uni-hamburg.de/projects/jadex/

3 THESIS OUTLINE

The Semantic Web and Agent technology open a possibility for the development of a new generation environment for dynamic proactive goal-driven heterogeneous resources. It will be an environment where resource adaptation level is elaborated based on Semantic Web technology and which allows interoperability between heterogeneous resources via a common ontology and common understanding of things. It will be supplied with Resource Agents (used to obtain new features) and resources will become more proactive in order to achieve the desired goals. This environment will enable building of a huge number of various industrial processes, business and human interaction models.

The projected development entails many challenges. This doctoral dissertation is aimed at addressing a number of selected problems of a challenging smart resource integration environment and elaborating extended resource description framework OSRDF with explicit context specification (where context is considered as any context, including conditions and properties of the resources and environment, role-based behavioural context, visualization context and etc.). To meet the demands of context-sensitive OntoEnvironment, we need to elaborate common formalism to allow OSRDF (additionally to resources description) to describe the proactiveness of a goal-driven resource via resource behaviour description.

The research of this thesis has been conducted mainly during the SmartResource project of "Industrial Ontologies Group"[17] (IOG) and partially motivated by real needs of of industrial companies (industrial partners of the projects). At the same time, the fact that research has been done under the real projects, has influenced the whole research process and problem statement. The goal of the project was to find solutions to resolve the stated problems and to validate them based on certain prototype. This means that means research and development is done on certan level of detalization, but covers all the involved elements. The project amed to develop a working environment. Thus, research

17 Industrial Ontologies Group (IOG) - http://www.cs.jyu.fi/ai/OntoGroup

has been done in several associated directions at the same time to show a real value of the research results. Conceptual-Analytical research (Järvinen, 2004) method has been used to motivate and produce OntoEnvironment and SmartResource abstract architectures, OSRDF framework and Human-Resource adaptation strategies, and real prototype development has been used to prove the proposed approach.

The dissertation is structured to logically provide a presentation of the Adaptive Semantic Web based Environment for Web Resources and the complementary work stages. The thesis consists of 4 logically and functionally separated Chapters:

- *CHAPTER 1 "Introduction and Research Questions":* The chapter describes the problem domain and the work and effort related to it as a background for the research presented in this thesis. The research process and methods are presented in this chapter. We then formulate our main research problem by establishing the specific research goals to be achieved, determining a solution methodology, identifying the main contributions, and previewing the research results to be expected.

- *CHAPTER 2 "OntoEnvironment: an Integration Infrastructure for Distributed Heterogeneous Resources":* In this chapter we present a vision of a modern resource integration environment regarding the further evolution of the Web. We also define a new generation resource with its features that is required by the environment infrastructure and user needs. Finally we present some solutions for the challenging problems in the process of this new resource description and integration within the environment.

  Section 1 "An intelligent infrastructure for distributed heterogeneous resources": This section describes an integration environment, defines the features of distributed heterogeneous resources that are needed for their adaptation and provides a Semantic Web based approach for building adaptation environment. (Khriyenko et al., 2004).

  Section 2 "OntoSmartResource – a smart resource of the Semantic Web": In this section we define a SmartResource – a resource of a new generation for semantically enabled intelligent integration environments, highlighting the new features and challenges (context-awareness, proactiveness, and goal-driveness) of such resources. (Khriyenko and Terziyan, 2004), (Kaykova et al., 2005c), (Kaykova et al., 2005b), (Khriyenko, 2007a), (Khriyenko, 2007d), and (Khriyenko, 2007b).

A *smart resource* is a proactive goal-driven dynamic resource, which sufficiently and proactively reacts on changes within its external environment or within itself. With resource dynamics and proactiveness, environment itself becomes more dynamic, more and more statements and behaviours become context-dependent and cannot be considered as absolute truths. At the same

time, a human being is an intelligent resource, which can be useful for other resources (for other humans, devices and software applications) as a service (an expert in a specific domain) or an information source. That is why we also consider a human as a potential smart resource, which can be semantically discovered in the Web, queried and used by both any resource of virtual world (application, service, and agent) and real world resources (humans, smart-devices, etc). Appearance of new resource features leads us to elaboration of new description/representation technics and new tools development.

Regarding the context-sensitive resource description, when we talk about context, intuitively, we think of a set of facts in which something exits or occurs. This idea is not reflected by the approaches described above in background section. Our intention is to apply the theory about context (Guha, 1995, McCarthy, J. and Buvac, S., 1997), for information semantics modeling and combine it with ontology and resource description. In our opinion this is a more appropriate way to take advantage of both approaches' strengths in complex domains. In the theory defined in (Guha, 1995, McCarthy, J. and Buvac, S., 1997), axioms and statements *p* are only true in a context *c*. In information semantics modeling area we would state that a context is a set of facts in which a concept interpretation is true. Defining context as a set of facts instead of a label allows us to manipulate it in a flexible way. It will be shown later as a basis of OntoSmartResource Description Framework (OSRDF).

As we can see, there are a lot of different languages elaborated for Agent behaviour modelling. They follow a set of various different standards and do not support RDF. In our opinion, it would be beneficial to extend the resource description framework and elaborate common formalism to allow it (additionally to resources description) describes resource behaviour, especially since behaviour is hinged upon surrounding information and behaviour rules also can be considered as resources.

The code in APL (Agent Programming Language) is, roughly speaking, a text. However in complex systems, a description of a role may need to include a huge number of rules and also a great number of beliefs representing the knowledge needed for playing the role (Katasonov and Terziyan, 2007). Also, in a case where the code is accessed by agents that are not going to enact the role, it is likely that they may wish to receive only the relevant part of it, not the whole thing. Therefore, a more efficient, e.g., a database-centric, solution is probably required. When an APL code is provided by an organization to an agent, or shared between agents, mutual understanding of the meaning of the code is obviously required. While using first-order logic as the basis for an APL assures understanding of the semantics of the rules, the meaning of the predicates used in those rules still needs to be consistently understood by all the parties involved. On the other hand, we are unaware of tools allowing unambiguous description of the precise semantics of n-ary predicates. As a solution to these two issues, we see creating an APL based on W3C's Resource Description Framework. RDF uses binary predicates only, i.e., triples (nary predicates can be represented nevertheless, of course, using several approaches). For RDF, tools are available for efficient database storage and

querying, and also for explicit description of semantics, e.g., using OWL. Our proposition for such an RDF based APL is a part of OntoSmartResource Description Framework (OSRDF) that deals with the Agent State/Condition and behaviour description.

Section 3 "OntoSmartResource Description Framework (OSRDF): This section contains the main part of the contribution. It is an extension of the Resource Description Framework which provides the tools for SmartResource context-sensitivity (context-awareness) and proactiveness description. (Khriyenko, 2006), (Kaykova et al., 2005c), and (Khriyenko, 2007a).

To show the value of the benefits of the Semantic Web based environments we will finalize the thesis with several use cases from the OntoEnvironment.

- *CHAPTER 3 "OntoEnvironment Use Cases":* Here we present four use cases of OntoEnvironment where collaboration/interaction of enhanced SmartResources solves a certain domain-specific problem. The chapter is based on the materials of the following referred published papers: (Kaykova et al., 2007), (Kaykova et al., 2005a), (Khriyenko, 2007b), (Khriyenko, 2007c), and (Khriyenko, 2005).

  Section 1 "Knowledge transfer from an expert to an artificial intelligence (Automated Industrial Maintenance)": Section presents the case of environment for the knowledge transfer from an expert to an artificial intelligent system and further automated industrial maintenance process performing.

  Section 2 "Environment for intelligent visualization of integrated information": This section describes open environment for semantically enhanced context-dependent multidimensional resource visualization, which enhances information search and browsing processes.

  Section 3 "4i Multimedia: semantically enhanced multimedia browsing": Semantically enhanced and context-based browsing through multimedia resource instances is presented in this section as one of the application area of semantically enhanced 4i (FOR EYE) technology.

  Section 4 "SemaSM: semantically enhanced Smart Messaging": This section presents one more case of integration environment. This is a Smart Messaging - messaging with semantically enhanced content and semantic-based user interfaces.

- *CHAPTER 4 "Conclusions and Further Research":* In this chapter we summarize the research of this thesis and answer the research questions. Finally we discuss some problems left unsolved and highlight the directions of further research.

CHAPTER 2

ONTOENVIRONMENT: AN INTEGRATION INFRASTRUCTURE FOR DISTRIBUTED HETEROGENEOUS RESOURCES

1 AN INTEGRATION INFRASTRUCTURE FOR DISTRIBUTED HETEROGENEOUS RESOURCES

In this section we overview an approach to heterogeneous resources integration based on the OntoShell concept and a Semantic Web enabled integration environment (OntoEnvironment). This idea owes its origin to the OntoServ.Net concept (IOG, 2003) developed by IOG.

1.1 OntoShell approach to the integration of heterogeneous resources

At the current stage of ICT development, there is a diversity of heterogeneous systems, applications, standards of data representation and ways of interaction. All those systems have been tailored for particular tasks and goals. The world is heterogeneous, and modern industry is looking for fast global solutions related to knowledge management, enterprise application integration, electronic commerce, asset management, etc. However, in spite of advancements in data processing and data acquisition it is still difficult to automatically process and exchange data between heterogeneous systems. Various industrial standards, which have been created and implemented by different consortia, appear to be insufficient for growing interoperability demands. Taking into account the great variety of possible types of information resources, data formats and ways of data accessing and acquisition, an integration of such resources into a unified environment is an important development challenge.

To enable autonomous integration of heterogeneous resources over the Web, we have to provide a common language for the interactions of those resources. We need to describe them in a common way based on a common ontology. Basically, the integration tasks can be solved by adaptation of data from heterogeneous formats to some commonly accepted and semantically enriched format, i.e., by adaptation of heterogeneous applications and data, originally represented according to a different standard, to common standard.

To resolve this problem we propose OntoShell as a suitable approach. An OntoShell is a software shell, which can be used to make a resource semantically enabled. It is configured for a concrete resource based on the ontology of its domain. The OntoShell represents a resource and carries an ontology based description. It plays the role of a mediator, which provides interoperability between the resources and the worlds of other OntoShells (other resources), whenever they have common interaction mechanisms and common language (FIGURE 4). Depending on the resource domain, an ontology based annotation must comprise not only a resource description (inputs, outputs, parameters), but it must deal also with many other aspects, which concern resource goals, intentions, interaction aspects, etc.

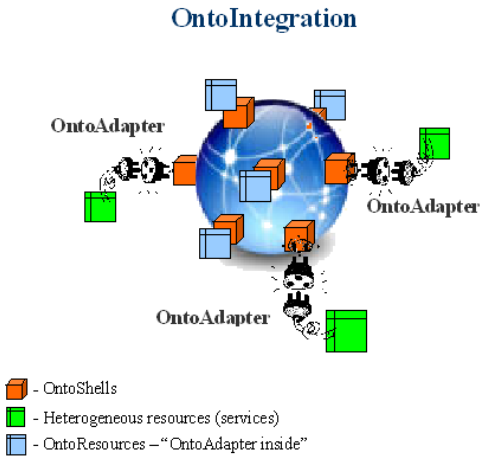


FIGURE 4 Environment-mediator

One of the important OntoShell parts is OntoAdapter for resources. When we develop a service based on the OntoShell approach (i.e., when we support an interaction interface with OntoShell), we need to adapt our service on the semantic level via the visual interface of the OntoShell. On the other hand, if we need to transform an existing resource to a semantically enabled one, then we have to develop certain mechanisms for accessing that resource. Since the resources are developed according to different standards for both content (WSDL, C/C++ DLL, Java classes or applications, SQL Server, DCOM, CORBA, etc.) and transport protocols (TCP, HTTP, RMI, etc.) we need to design and develop resource (services) transformation modules (OntoAdapters) for the semantic (semantic description of a resource content), content (programming interface) and transport levels. Depending on the resource description, construction blocks will then fill OntoShell (FIGURE 5).

OntoAdapters are ontology based modules supplied with both interaction interfaces for the OntoShells and the concrete class of the resources. For example, there are many services, databases, smart devices (software interfaces for them), humans, etc. "Ontology based" means that we have to create all the ontologies of these resources and their domains in advance. Ontologies'

building phase includes development of upper ontology and development of ontologies themselves, which include data about the resources and environment domains. Concrete data will be annotated (marked up) in terms of upper ontology and common ontology. Here, the ontology provides a basis for a well-understood "common language" to be used between system elements.

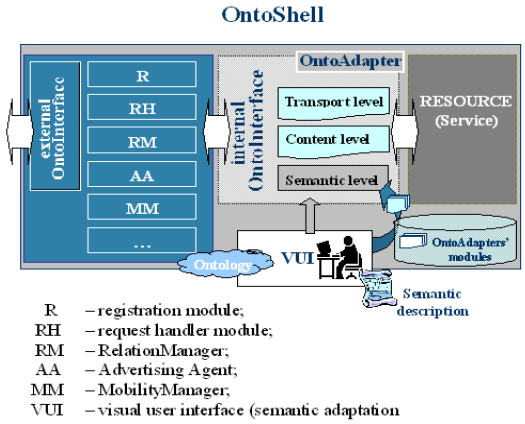


FIGURE 5 Structural schema of the OntoShell

The OntoAdapter approach has been elaborated and utilized in the SmartResource project and, further, has been enriched for the development of Smart Semantic Middleware for Ubiquitous Computing in the ongoing UBIWARE (2007-2010) research project[18].

1.2 OntoEnvironment for Semantic Web enabled resources

Regarding distributed resource integration, it is time to consider the architecture of an ontology based distributed integration environment for Semantic Web Resources, which is based on the OntoShell concept (OntoEnvironment).

1.2.1 Environment architecture

OntoShell is the main structural component of the OntoEnvironment. As mentioned earlier, OntoShell is based on a mechanism which includes an ontological description and provides interoperability for resources. As the end result, we have an environment with many OntoShells, which can interact with each other via a common language. But that is not enough; these OntoShells need also interaction, advertising and registration mechanisms, mobility, etc.

18 UBIWARE project (2007-2010) - http://www.cs.jyu.fi/ai/OntoGroup/UBIWARE_details.htm

Subsequently, an OntoEnvironment consists of an organized set of OntoShell enabled elements (services) (FIGURE 6), such as:

- OntoAdapter for the resources;
- OntoShellContainer;
- OntoMeetingPlatform;
- OntoMobilityService.

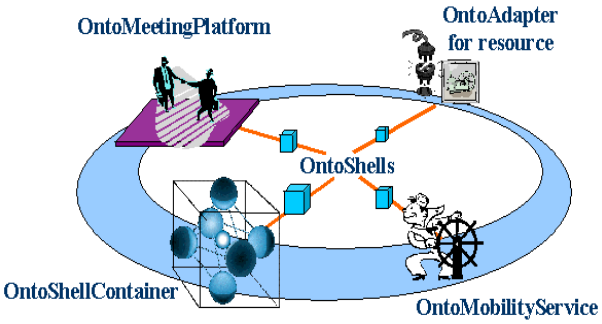


FIGURE 6 Elements of OntoEnvironment

Thus, a modular approach is employed for constructing a universal resource integration system based on OntoShells. It assumes that resources can be nested to an arbitrary number of levels via such shells in order to model multilevel cluster architecture. In the OntoEnvironment, services can be organized into a cluster (OntoShellContainer), which represents services wrapped within OntoShells. In order to be able to share their information, OntoShellContainers must also be integrated into a higher level network like a resource. Each of the elements of the OntoEnvironment can be connected to several others. Finally, the integrated elements will form a decentralized environment of resources – a Peer-to-Peer network. In such a context, the OntoShellContainers become representatives of local/nested resources at an appropriate level of the network. Resource clusters will reduce the cost of searching resources. Such consolidation into clusters may be organized according to various principles, such as:

- Location in a concrete server;
- Membership in a concrete domain;
- One-target federation of the resources (services);
- Geographical location (e.g., in cases, where a human is a resource, or a resource is a movable device, for example).

1.2.2 Hybrid interaction model

In a centralized interaction model, each OntoShell has a mechanism for registration to a shell which represents a cluster formed by an aggregate of OntoShells. Thus, the whole interaction (e.g., requests for searching of necessary resources and advertising oneself) are realized via a "mother shell", i.e.,

OntoShellContainer. A special demountable (adapter) module for OntoShell representation in role of the OntoShellContainer is needed for the purpose. Such a demountable module has to be configurable in a detailed way (especially in the realization of a business model). It has to be made responsible for observation of registration agreements, quality of provided search service, etc. One may think of OntoShellContainer as a mediation platform driven by various goals of resource alliances.

We may now consider the two main reasons for cluster organization:

- A cluster is organized in order to decrease useless traffic while searching a resource. In this case, the cluster is organized hierarchically and uses a "class-subclass" type relation based on a resource ontology. The "mother shell" might register those elements which are members of its subclasses. An example of such clusterization is shown in FIGURE 7. Since the organization of such clusters is carried out spontaneously and the shells at some level may not register in the "mother shell", we are not talking about a shell management architecture that is centralized in this case.

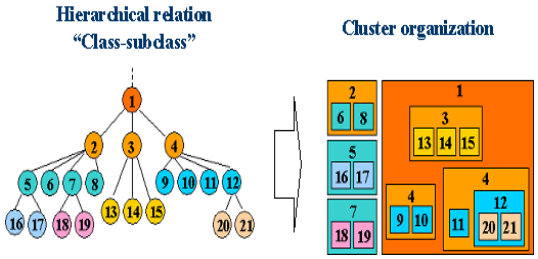


FIGURE 7 "Class-subclass" clusterization model

- A cluster is organized to behave based on a community goal of a closed set of functioning resources (components), which compose it. The cluster can be used to cover a concrete domain with a set of different resources without having a relation to the same class (for example, a maintenance platform with a set of services such as a main maintenance service, a device alarm service, a set of classifiers, etc.). In this case, the "mother shell", which represents some cluster, provides a search and interaction organization for the registered resources. However a mother shell cannot always represent all of its elements the same way as a (sub)class in the hierarchical model, because we are not assuming that an aggregation of heterogeneous components covers a separate class. The organization of such a heterogeneous cluster is shown in FIGURE 8.

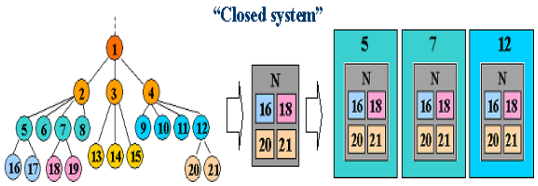


FIGURE 8 "Closed system" clusterization model

Because of the impossibility of nesting all the hierarchical clusters which would cover all the levels of a "class-subclass" type ontology, we cannot provide a guaranteed resource search via the "mother shells". Also, search within a cluster-tree (formed at some level) provides both a centralized top down search approach and a non effective bottom-up search approach at the same time.

For resolving these two problems the OntoEnvironment introduces an additional possibility of interaction between elements without a "mother shell". This can be considered as a P2P interaction model. The main challenge here is own "record book" keeping by each OntoShell. This "record book" has to contain a list of useful resources. In that way each shell (resource) can use its own "record book" directly. Replenishment and modification of a resource's "record book" is executed during the establishment of interaction with other resources.

Let us consider some variants for resource search in the hybrid interaction model:

- Interaction organization via OntoShellContainer ("mother shell")
- Records exchange during interaction between resources.
- Using OntoMeetingPlatforms – places where shells (more precisely, shells' Advertising Agents) can meet each other and exchange their "record books" (fill them in).
- Using special search services.

In each record's exchange a negotiation mechanism may be used.

OntoMeetingPlatform is a service which enables shells' publicity agents (PublicityAgents) to meet each other and exchange records in their "record books". This service may be placed into OntoShell or may be elaborated as a service of a new generation in OntoEnvironment, supplied with the same interaction interface as OntoShells. Such OntoMeetingPlatforms may be attached to some class of the service classification tree in the ontology and cover some specific resource domain. Such relation to the concrete domain may be fixed in the OntoMeetingPlatform annotation (description) and used by OntoShells' PublicityAgents.

Since the number of records will increase very fast, we must supplement the OntoShell structure with a management block for the "record book" – RelationManager. To do this, we insert two additional elements into the OntoShell for management of relations. These are the RelationManager and PublicityAgent blocks. These blocks have to be configurable. RelationManager is made responsible for the rectification of the "record book" as dictated by the number of useless and useful records. PublicityAgent is made responsible for visiting necessary OntoMeetingPlatforms, for negotiating with other agents for an exchange of the records, etc.

1.2.3 Mobility

In a distributed environment for resources the necessity of resource mobility emerges in a number of cases. In other words, there is sometimes a need to move a resource with its necessary "equipment" from one machine (computing system) to another one. The realization of such a movement is a duty of a special service (OntoMobilityService), which will provide mobility in OntoEnvironment. The party (player) requiring resource mobility has to supply its computing system with such specific service.

To be a "player" within a mobile environment, the elements of the OntoEnvironment have to be supplied with a MobilityManager module. This module has to be configured in conformity with a policy system (concerning mobility). A resource can be configured in both ways, either as a movement initiator or as an available resource to be moved. All resources of the mobile environment which support OntoMobilityService and thus mobility, have to provide the necessary data for this service, such as location, final point of destination, residence time, etc.

1.2.4 Business model

Considering implementation issues of a distributed integration environment based on the OntoShell approach, we have to also think of the related business environment. In such an environment service providers are interested in frequent use of their services; that is why service advertising and search play an important role. Also, within such business environment some mediation elements, which provide necessary services for players, have to be embedded.

OntoShell. At the very beginning of its appearance an OntoShell needs to advertise its resource. For the realization of this goal we may consider two ways: registration in a "mother shell" and delegating the responsibility for advertising duties to it; or self-advertising during its life cycle while visiting OntoMeetingPlatforms. In case of needing to interact with some resource (which is not available in the "record book"), an OntoShell has to use its search process via the "mother shell" or other special search services. An alternative solution is to stay on an OntoMeetingPlatform in order to meet the necessary resources or find a reference to them. During the establishment of a link with an environment element for records (from "record book") exchange or registration in a cluster, some negotiation mechanism is used. Thus, various aspects of behaviour have to be configured in advance via a software visual interface module. Such configuration plays an important role especially in a business environment, where "service costs money".

OntoMeetingPlatform. We may consider two ways of providing OntoMeetingPlatforms: if provided in a centralized way, they will be advertised in one central point; if provided without centralization, they will need to advertise themselves in the same way as OntoShells. In a general case,

OntoMeetingPlatform, as a resource in OntoShell, plays its/OntoShell's role. It may register in a cluster, visit another OntoMeetingPlatform, use search services, etc.

OntoShellContainer. OntoShellContainer provides a mechanism with a more complicated behaviour especially in a business environment, where it plays the role of a commercial mediation element. Loose configuration of such element may result to a negative profit. From the moment an OntoShellContainer emerges it needs to advertise itself in the same way as an OntoShell does. When in the role of "mother shell" the OntoShellContainer has two main goals:

- Advertising its "daughter shells" via self-advertising.
- Supplying a search mechanism.

Registration in a cluster allows OntoShell to share its "record book" within whole OntoShellContainer for advertising purposes. This information allows execution of a more effective search and removal of useless ascent (bottom-up mode) in a cluster-tree during search, which has been described in Section 1.2.2. For a further refresh of the OntoShell's "record book", the OntoShell may proceed sharing it within OntoShellContainer ("mother shell"). Depending on the number of new records (references) the OntoShellContainer updates its profile used for advertising the whole cluster. There is a competition between "daughter shells" to get more queries from a mother shell based on advanced personal profile. At the same time, there is a competition between OntoShellContainers based on the updated community profile.

In our business model we may highlight the set of following "players":

A – provider of OntoShells, OntoShellContainers and OntoMeetingPlatform, OntoMobilityService;
B – OntoAdapter blocks developers;
C – Owner of an OntoShell with resource;
D – Owner of an OntoShellContainer;
E – Owner of an OntoMeetingPlatform;
F – Owner of some search service.

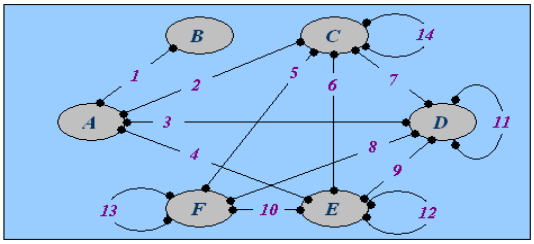


FIGURE 9 Inter-players interaction

FIGURE 9 shows the business relations between players:

1 – Player "A" is a customer of player "B" for adaptation modules development (OntoAdapter's modules);

2 – Player "A" supplies OntoShell with necessary adaptation modules and OntoMobilityService (in case of need) to player "C" for inclusion of its resource into OntoEnvironment;

3 – Player "A" supplies OntoShellContainer and OntoMobilityService (in case of need) to player "D" for cluster organization;

4 – Player "A" supplies OntoMeetingPlatform and OntoMobilityService (in case of need) to player "E";

5 – Player "C" pays player "F" in case there is a need to search a necessary resource;

6 – Player "C" pays player "F" in case there is a need to find someone or refresh the "record book" during the stay on an OntoMeetingPlatform;

7 –An OntoShell registers itself in an OntoShellContainer based on some agreements and advertises itself for further discovery. Additionally the OntoShellContainer provides a search service for registered OntoShells. Player "C" pays player "D" for that search service;

8 – In a manner similar to that in 5, an OntoShellContainer may need to search some resource to guarantee a high-level quality of its services (increasing its competitiveness in that way). In case of search services use, player "D" pays player "F". At the same time player "F" plays the role of player "C" and may have a need to register in the OntoShellContainer (case 7), then player "F" pays player "D";

9 – Player "D" pays player "E" for the use of an OntoMeetingPlatform by an OntoShellContainer. OntoMeetingPlatform is a service which needs to advertise itself. In that case, the OntoMeetingPlatform may be registered in an OntoShellContainer;

10 – Player "F" pays player "E" for the use of the OntoMeetingPlatform with a goal to supplement the resource database of the search service. On the other hand, the OntoMeetingPlatform may use the search service to find a necessary resource (another OntoMeetingPlatform, or OntoShellContainer). In that case, player "E" plays the role of player "C" and pays player "F" (case 5);

11 - In a manner similar to that of OntoShell, an OntoShellContainer may register itself within other OntoShellContainer for advertising and additionally for search via a "mother shell". In that case, player "D" pays player "D" for that search service.

12–An OntoMeetingPlatform may visit another necessary OntoMeetingPlatform in case there is a need to advertise itself for concrete resources. Then player "E" pays to another player "E".

13 – Player "F" plays the role of player "C" in case there is a need to use a search service with a goal to supplement its resource database and increase its quality. Then this player "F" pays to another player "F".

14 – If we consider a real business environment, we have commercial services, which require payment for their service. In this case player "C" pays to another player "C".

2 ONTOSMARTRESOURCE – A SMART RESOURCE OF THE SEMANTIC WEB

This section shows us a vision of a Smart Resource in the future Web. Here we consider resources of different nature (real and virtual world resources, human, processes and etc.); dynamism, context-awareness and proactiveness of the goal-driven resources, as the main features for theresource of next-generation Web. Concerning the resource managing issues, the concept of intelligent resource visualization is presented as basis for resource search/browsing.

Currently, Semantic Web Activity and semantic technology applications are focused on domains of shared and reusable Web content (Davies et al., 2002). Intelligent Web Services (Fensel, 2003) and correspondent ontologies develop most rapidly there. However, for the industrial adoption of Semantic Web technology these efforts do not seem to be enough. The problem is the initial orientation of semantic technology development towards World Wide Web digital resources. As a result, other industrial domain resources have not been adequately taken into account; devices, processes and even humans have been left out of consideration.

The traditional Semantic Web point of view considers "machine" as a set of applications, web-services, agents, etc. Nowadays "machines" can also be thought of as embedded computational entities, such as, e.g., intelligent parts of field devices (FieldSence, 2002). We should integrate the objects of the real world into Semantic Web based Environment. Of course, the main object of the real world has always been the human, and must thus be a resource (and not just a mere user) of the global semantic enabled environment.

What is a *smart resource*? Traditionally resources have been considered either as active or passive. A *passive resource* just provides access to itself. An *active resource* additionally can use other resources to support its functional aim (the final goal). If a resource changes its content (changes itself) or its state is changed by the environment during the execution, then such a resource is said to have become a *dynamic resource* (in relation to its content). As an example.

one might consider a website which automatically refreshes its information by accessing and using information from other sites; or some Web service based on a neural network classifier, which incrementally learns from external learning samples; or an industrial field device (which also can be considered as a Web resource adapted to the Web environment via an intelligent software component (e.g. *semantic wrapper*), which performs a predictive maintenance function, activation of the maintenance activities, etc.) and the parameters of which can be changed; etc. In these cases it might be necessary to examine some of the change related actions, such as: notification of other resources, which depend on the changed resource's content; new advertising to reflect some new characteristics that might have emerged; activation of the resource maintenance processes; etc. These actions are important for resources (systems) which depend on information from other resources. There are systems, such as companies' human resource management systems and other systems or groupings (research groups, clubs, teams) which are concerned with people or their activities; all kinds of clusters (the formation of which is based on some common features of the components), which totally depend on a change in a component; statistical systems (customers clusterization) and customization systems, especially for marketing environment. Sometimes a resource content modification (change) brings with it also a change in the resource's semantics (semantic description of a resource). In that case, the most important action would be to automatically change the resource's semantic description (*Semantic Maintenance of a resource*). A *smart resource* is, thus, a proactive goal-driven dynamic resource, which sufficiently and proactively reacts on changes within its external environment or within itself.

With resource dynamics and proactiveness, environment itself becomes more dynamic. With this, more and more statements and behaviours become context-dependent and cannot be considered as absolute truths. Thus, this fact gives rise to a new problem – the inability of the existing resource description approaches to describe context-sensitive information.

2.1 Multilayered Context-Sensitive Resource Description

Context is a collection of relevant conditions and surrounding influences that make a situation unique and comprehensible. The human cognitive and perceptual systems are designed to identify and use context automatically as we go about our daily lives (Degler and Battle, 2000).

There is now considerable interest in "context-aware computing" – in computational systems that can sense and respond to aspects of the settings in which they are used. However, a considerable amount of confusion surrounds the notion of "context" – what it means, what it includes, and what role it plays in interactive systems. For Dourish (Dourish, 2004), context is a form of information. It is something that can be known (and hence encoded and

represented much as other information is encoded and represented in software systems).

What is "context"? There are quite a many different points of view regarding "context". Software systems are representational, so a concern with context naturally leads to a question of how context can be encoded and represented. For instance, Schilit and Theimer (Schilit and Theimer, 1994) define context as "location and the identity of nearby people and objects." Ryan et al (Ryan et al., 1997) define context as "location, identity, environment, and time. In one of the more extensive investigations of context-based computing, Dey et al. (Dey et al., 2001) define context as "any information that can be used to characterize the situation of entities" and elaborate it as "typically the location, identity, and state of people, groups, and computational and physical objects." One of the broadest definitions is one of the earliest; Schilit et al. (Schilit et al., 1994) observe: "Context encompasses more than just the user's location, because other things of interest are also mobile and changing. Context includes lighting, noise level, network connectivity, communication costs, communication bandwidth, and even the social situation; e.g., whether you are with your manager or with a co-worker." Context is a collection of conditions and influences (statements about subject resource, other resources, environment that influence on the resource in one way or another). They are contextual statements.

If we consider a location-sensitive services area, we might come up with another question: What is the difference between location-based and location-aware services? The whole world can be divided into three parts (see FIGURE 10). The first part contains substances that directly influence the result of a service and are directly relevant as input parameters. In other words, they are parameters that should be given to the service; otherwise it can not provide a result. The second part is formed of substances that improve the service result depending on the situation, but without which the service can also produce a result (even if it is not the optimal result). And the third part is formed of irrelevant substances, which have no influence on the result of the service.

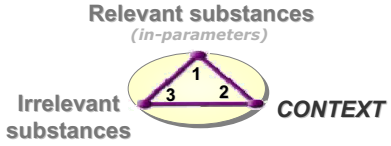


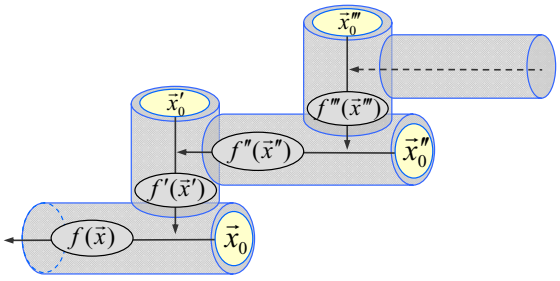
FIGURE 10 Substance relevance

Let us consider an example of a service that provides information about restaurants in a certain city. The input parameters of this service might include the meal preferences of the user, diet, etc. This service may be location-aware and provide, for the user, information about the nearest restaurant that fits the user's preferences. This can be done via a GPS add-on and adding the user location data to the search algorithm. In this case we are improving the service result by utilizing user context information. And even if no location data is

available, the service returns some result. Such a service can also be location based if the goal of the service is to provide information about the nearest restaurant. The user location data here plays a role of the necessary input parameter for the service, without which the service can not return any result.

When we consider the relation between baseness and awareness and the context-sensitive service area, we meet new challenges. What is a context-based service? What is the first part of the substances (see FIGURE 10), if a context plays a role of the second (contextual) part? Is location data in location-based services also a context? And if we generalize the relation between all the substances in the world, we can say that everything somehow influences everything else in one way or another.

Thus, generalizing to a sufficient degree, context can mean anything that in one way or another affect the subject statement. Let's consider a subject statement as some function $f(\vec{x})$, where $\vec{x}$ is a set of parameters (other statements) that affect the function. In general it is possible to say that our world is filled by substances that directly or indirectly or in no way influence a certain object (initial substance). Thus, it is possible to state that $\vec{x}$ is a complex set and can be divided to $\vec{x}_0$ (a set of the parameters that directly affect the function) and $\vec{x}'$ (a set of the indirectly affecting parameters). But, what does "indirect influence" mean? It means that $\vec{x}'$ influences something that then directly influences our initial $f(\vec{x})$. Thus we have $f(\vec{x}_0, f'(\vec{x}'))$. Many experts call $\vec{x}'$ the context for function $f(\vec{x})$ and $\vec{x}_0$ the set of the first-hand input parameters. But if we generalize this approach and try to define $\vec{x}'$ more exactly, then we can state that $\vec{x}'$ is a complex vector also. It contains $\vec{x}'_0$ (substances that directly influence $f'(\vec{x}')$) and $\vec{x}''$ (substances of indirect influence for the $f'(\vec{x}')$ function). Again we have a function like $f'(\vec{x}'_0, f''(\vec{x}''))$. Thus we can define a concept as a derivative of a context. Now we can refer to the whole vector $\vec{x}$ as a "context for the result substance", the context order (level of the context) being where a context of the 0th order is a vector of directly influencing substances ($\vec{x}_0$). FIGURE 11 shows us how a certain context can be divided to a set of ordered contexts ($\vec{x}_0$ - 0th order, $\vec{x}'_0$ - 1st order, $\vec{x}''_0$ - 2nd order, $\vec{x}'''_0$ - 3rd order contexts).

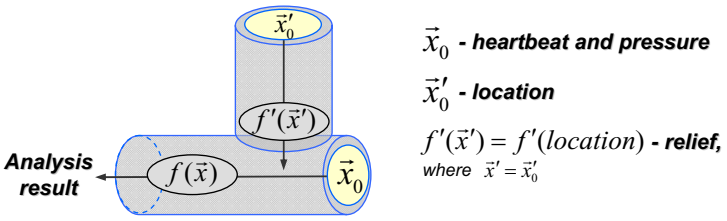


$$f(\vec{x}) = f(\vec{x}_0, f'(\vec{x}'_0, f''(\vec{x}''_0, f'''(\vec{x}'''_0, \ldots)))) \sim f(\vec{x}_0, \vec{x}'_0, \vec{x}''_0, \vec{x}'''_0, \ldots)$$

FIGURE 11 Ordered context definition

There is one more addition for the previous substance dividing from FIGURE 10. Context does not just play the role of an additional parameter that enhances the final result of the process or helps to get a necessary input parameter for the function, but it can also affect the choice of the function for process performance. In this case $f'(\vec{x}')$ may play a role of a meta-rule that, as an outcome, has a certain function $f(\vec{x})$ from a set of functions to be used. According to Dourish (Dourish, 2004), activity happens "within" a context that describes the features of the environment within which the activity takes place but which are separate from the activity itself. That is why context and activity are separable, but context arises from activity and can be actively produced, maintained and enacted in the course of that activity.

Thus, we do not divide substances to two or three parts. We just define the order of their influence (relevance) on a certain other substance. Let us consider an example to illuminate this approach. Consider a device that measures a sportsman's (for example, a runner or a cyclist) heartbeat and blood pressure and keeps track of his/her location (via a GPS) throughout the exercise. Further, based on this data, the sportsman can analyze his training in order to improve his results in future. Is this analysis location based or location aware? Actually there is no direct dependence on location as such; rather, information about the relief of the terrain is needed. So, although we can say that this is a relief/terrain based analysis, we nevertheless feel that location somehow influences and has some level of relevance to our analysis. This is true, of course: location is a necessary inpu parameter for a location-based service that provides relief information based on location. We see that, regarding the above classification (FIGURE 10), location is neither a direct input parameter nor a contextual input parameter, but it somehow influences the final result. That is why the ordered context approach is a suitable approach for context-sensitive description. Regarding this approach, the location in the last example plays a role of the 1st order context (see FIGURE 12).



$$f(\vec{x}) = f(\vec{x}_0, f'(\vec{x}')) = f(heartbeat\ \&\ pressure, relief) \sim f(heartbeat\ \&\ pressure, location)$$

FIGURE 12 Ordered context definition example

2.2 Resource Behaviour

According to the idea which was born as part of the OntoEnvironment concept (Terziyan and Khriyenko, 2004, Khriyenko et al., 2004), the resource management agent shell (called OntoShell) can be extended with *ResourceBehaviourAgent,* a proactive goal/behaviour interpretation module**.**

ResourceBehaviourAgent is a resource management agent with programmable goals (behaviour). It interprets assigned behaviour rules and performs actions corresponding to them. Such rules may be "IF=>THEN" rules (in simple case), where the conditions and actions are described according to a common ontology. This ontology must contain the main terms of the conditions and actions describing all types of the resources. ResourceBehaviourAgent must be supplied with a deducing (derivation) mechanism based on the assigned rules. This mechanism provides a way to simplify and combine the rules for effective deduction. Also, such mechanism can reveal contradictory (conflicting) rules. The use of feedback coupling after the actions plays a very important role, because this information about the consequences of the decisions made allows correction of behaviour rules. Such a mechanism is very important especially in case of automated replenishment and modification of the rules set during an experience (behaviour rules) exchange between the resources. Such possibility to learn during the resource's life is an additional advantage of the smart resources.

Concerning the organization (formation) of the ResourceBehaviourAgent, we need to have a suitable user interface to assign the behaviour rules. Such an interface must be made optimally convenient for a human for assigning rules with the help of necessary ontologies. For providing the required convenience, such user interface must be dynamic and must adapt to each concrete type of resource, condition and action. In this case, we have to develop not only an ontology for behaviour description (conditions and actions), but also an ontology for the description of the terms presented by the "behaviour" ontology.

2.2.1 Resource Goal and Behaviour Description

Autonomous systems must be automatic and, in addition, they must have a capacity to form and adapt their behaviour while operating in the environment. Traditional AI systems and most robots are automatic but not autonomous - they are not fully independent from the control provided by their designers. Autonomous systems are independent and able to self-control. As it is argued in this work, to do this, they must be motivated.

In Agent Environment (as well as in the real world) the base for any interaction is the behaviour of each individual. Further, integration of these individual behaviours may form the behaviour of Agent Alliance. In real world

almost all behaviours (actions) are goal-driven, but some of them are not. With software agents in mind we are focused just on goal-driven behaviour. What is a goal-driven behaviour? It means following a set of rules which are aimed for achieving of certain goal. A goal is a statement about the environment which is not yet true (does not exist as a true statement in the description of the environment), and an agent aims to act so that statement becomes true (fact statement). As a result, we have a trio: a *behaviour* which is driven by a certain *goal* which can be achieved by performing certain actions that follow a set of behavioural *rules*. However, even having a rule base, which enables an agent to achieve a goal, still extra information (environmental facts) is needed. This is because each rule has to have a sufficient condition. In our case a sufficient condition is a presence of input data for the action being performed. Apart from the sufficient condition we should take into account also a necessary condition: a presence of a goal along with a certain context (set of facts of the environment) for performing the goal. Not all goals assume execution of unambiguous rule(s). Some goals can be represented by an aggregation of more specific goals.

Referring to the trios that were discussed above, each agent should have an initial set of those trios (regulated by their initial role). These trios represent the expertise and experience of the agent. As in the real world, agents can exchange their expertise (rules for execution of actions that depend on the goals and on software modules for execution) here too. A wide spectrum of those trios being available, an agent can automatically divide up the goals (that cannot be achieved because of lack of information) into subgoals and to create a chain of nested trios.

One more thing from the modeling paradigm that can be applied to an agent is an agent role. Agent role means an aggregate of goals corresponding to a specific purpose of the agent. Individual role does not assume a fixed set of activities, the set of the goals can be different even for the same role depending on the context. This approach to the goal and behaviour description brings a possibility for an agent to be more autonomous. Through utilization of this approach the agent can change its role, the set of the goals corresponding to its purpose depending on the condition of the environment. In other words, the agent can change its behaviour based on a context.

The approach of the resource goal and behaviour description assumes that all the goals, roles, descriptions and templates of behavioural rules can be found in an ontology. The templates of behavioural rules are described in a general way and can be applied to any particular agent. Such a description requires utilization of a handy and flexible description schema (RG/BDFS-Lite), which will be presented later. The architecture of Agent Platform is represented in FIGURE 13.

On its own platform the agent has a resource history (encoded, e.g., in the RS/CDF contextual extension of RDF (Kaykova et al., 2005b)), where it stores all the statements about resource states, conditions and actions that have been performed by the resource agent and other contextual information that can be useful (statements about the environment of the resource).

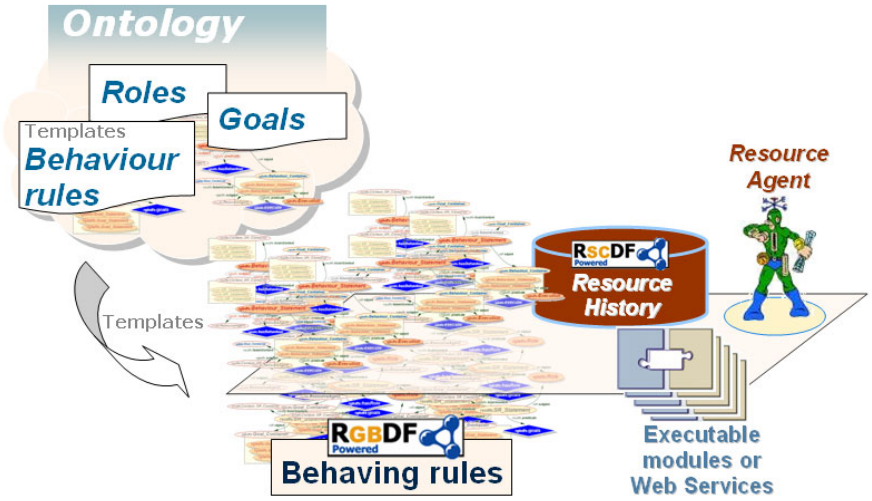


FIGURE 13 Architecture of SmartResource platform

Some executable modules (code) that the agent must perform can also be located there as an output of its behavioural rule chain. Otherwise the agent has to utilize external web services. The agent has to interact with an ontology server to be able to download a necessary role, goal description or behavioural template whenever the need arises.

Behavioural template represents a rule for behaviour in RDF serialization. The template is represented by a behavioural statement osrdfs:Behaviour_Statement (it will be described in Section 3) and contains a necessary condition (goal) and a sufficient condition (condition of the environment) as the contexts of rule execution and a set of the executive descriptions (specification of the executable modules that should be invoked) as an output of the rule.

We can divide the process of the resource goal and behaviour annotation into several stages. The first is the goal instance definition stage that assumes a creation (process of describing) of a statement towards which an agent should strive. This goal can be specified directly by an expert or via a specification of the agent goal. Based on this goal description, the corresponding appropriative behavior template(s) have to be found in the ontology, downloaded and transformed to behavioural instance(s) on the resource platform. After this the needed executable modules (if they are not located at the resource platform) also can be downloaded. As a final stage of the goal/behaviour annotation process an expert has to specify (add/modify) the context of the behaviour. Now the platform contains the behavioural rule(s) in an RDF/XML serialization format that can be used by the agent engine. This engine follows the behavioural rules until the goal is achieved.

2.2.2 SmartResource Agent Architecture

The main feature of the SmartResource Platform is process performance via Resource Agents' communication. All Platform Agents are designed in a uynified way to provide interoperability and a common approach. General SmartResource Agent Architecture is represented in FIGURE 14. The main functionality of the agent is based on execution of a behaviour (set of behaviour rules) that corresponds to an assigned role. The behaviour description is an RG/BDF based script, which can be loaded from the ontology of Roles. Based on the RG/BDF script role and on RS/CDF based beliefs descriptions, the agent executes reusable atomic behaviours, i.e., actions (executable modules). A performed action results in a change of Environment State. In other words, this performance modifies agent beliefs. Such atomic behaviours can be downloaded from a remote pool of atomic behaviours on demand, but the basic set of them and the frequently used actions can be placed locally on the agent platform. The basic set of actions that were used in the current prototype of SmartResource Platform is shown in FIGURE 14.

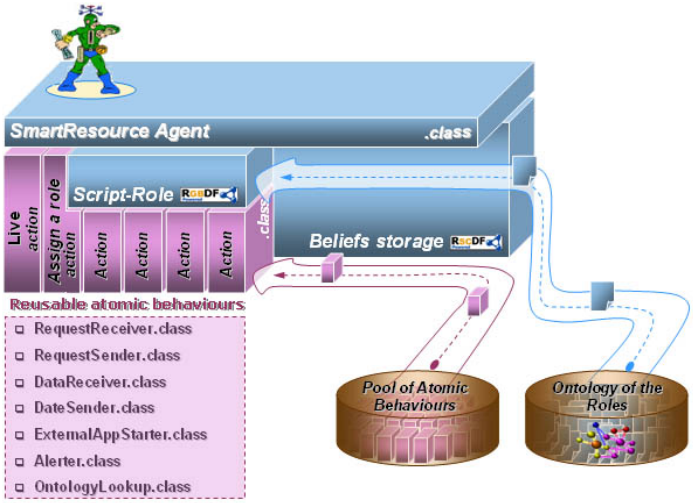


FIGURE 14 Architecture of SmartResource Agent Shell

2.3 Semantic Maintenance of a Resource

As mentioned before, resource semantics (resource role and purpose) can change when the resource content is modified. One should change the semantic description of the resource accordingly. *Resource Description Semantic Maintenance System (RDSMS)* is a system (a set) of mobile or static services, which automatically make (create, modify) a new semantic description based on a changed resource content, parameters, preferences, actions, etc (see FIGURE 15).

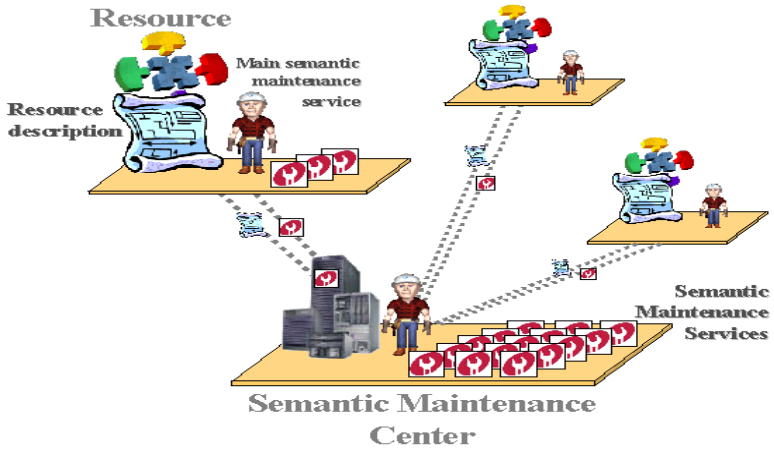


FIGURE 15 Resource Description Semantic Maintenance System

A large portion of available digital information is written as text documents in the form of web pages, reports, papers, descriptions, emails, etc. Extracting the knowledge of interest from such documents from multiple sources in a timely fashion is therefore crucial. A set of such services offered by *Resource Description Semantic Maintenance System* may vary depending on the class (type) of the resource and may be composed automatically to create more complex services whenever necessary.

Let us consider some research article as a sample of a resource. The semantic maintenance system can use the related component service as the domain qualifier of the article content, as a word or page counter, or as an additional information qualifier (authors, contact addresses, references), etc. Such services should be developed in accordance with a set of resource characteristics. Interaction between these components and the resource will be organized through the main semantic maintenance service, which uses the set of available internal services or finds external ones for making the required descriptions. Such an alliance between a resource and the Resource Description Semantic Maintenance System can be organized to behave in a manner that is based on a community goal of a closed set of functioning resources (components) which compose it (Khriyenko et al., 2004). The system can be organized for both private (individual) and shared use by a number of the resources belonging to some class (type).

2.4 Process as a Smart Resource in the Semantic Web

In General Networking Framework (GNF) ontological modeling of business processes integrates the component behavioural models of various business actors (i.e., the agents representing smart resources in the web). This integration

then constitutes the behavioural model of an agent responsible for the "alliance" of the components. This means that such a "corporate" agent will monitor the behaviours of the proactive components against the constraints provided by the integration scenario. The model is naturally recursive. This means that the corporate agent can be a component in a more complex business process and will itself be monitored by an agent from a higher level of hierarchy. The agent hierarchy can be considered as a possible mapping to the part-of ontological hierarchy of the domain resources.

Before discussing process integration issues, let us answer the question: What does a process mean in Global Understanding eNvironment (GUN) (Kaykova et al., 2005a)? According to the first axiom (see FIGURE 16) of GUN, Process is a resource similar to other resources in GUN (Device, Service and Human/Expert), but does not belong to the world of physical resources. As all GUN resources, Process has its own properties that describe its state, history, sub processes, and whether it belongs to an upper process (super process). Thus, following the principles of a GUN resource, each Process is enhanced with an agent that serves the process as a resource and actually can implement it as a behaviour engine. Each process is a sequence of actions (osrdfs:Execution, see the next chapter) that results in the achievement of the final goal. So, each agent per se is a process. In this case Agent Behaviour plays the role of a sequence of actions and the final result is represented by the Agent Goal.

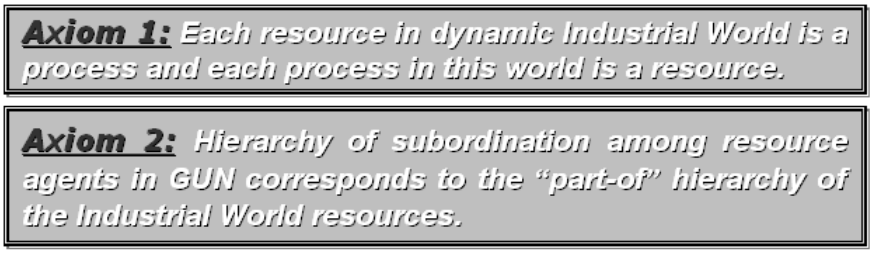


FIGURE 16 The axioms of GUN

Each GUN resource can theoretically be involved with several processes and appropriate commitments and activities, which can be either supplementary or contradictory. This means that the resource forms a part of several more complex resources and its role within each of these resources might be different.

There are some models for upper process organization. To start with, let us consider an executable module as a set of atomic non-configurable actions. This is to say, the choreography of a subject resource by its agent via action performing is a non-configurable atomic leaf-process. In this case, agents behave according to a certain plan – or, in other words, a planned set of behaviours. Such simple processes can be organized in alliances. The main function of a Process-Agent is the orchestration of a set of subprocesses. Following this approach, architectures of arbitrary nested processes can be

built. In these architectures, leaf-processes are physical world Resource-Agents (Device-Agent, Service-Agent and Human/Expert-Agent).

One aim of process (upper process) creation is to organize the cooperative work of subprocesses to improve their individual performance. Each agent should be supplied with a behaviour planner module that generates a plan for a behaviour performance without any conflicts. In this particular case, the Process-Agent should utilize the behaviour planner to build a plan for the cooperative work of subprocesses and to set constraints on their own plans. Another aim of process creation is to utilize other processes to realize another, separate group goal. In this case, the achievement of the subprocesses' goals depends on the commitments and contracts between all parties. Thus, the Agent-owner of this group goal plays two roles: the role of a subprocess as another of the subprocesses in this Process (but with a difference – it just has a goal and does not have atomic behaviour) and the role of the Process-Agent that performs the orchestration of the subprocesses. If we separate these two roles, we come up with the first model where we have a blank subprocess (it has just a goal and it does not have any atomic behaviour) among subprocesses, and where the achievement of the group goal takes the highest priority. FIGURE 17 shows us two process models (described above) and generalized model where the Process-Agent replans the subprocesses' behaviours according to the goal achievement priorities of those subprocesses.

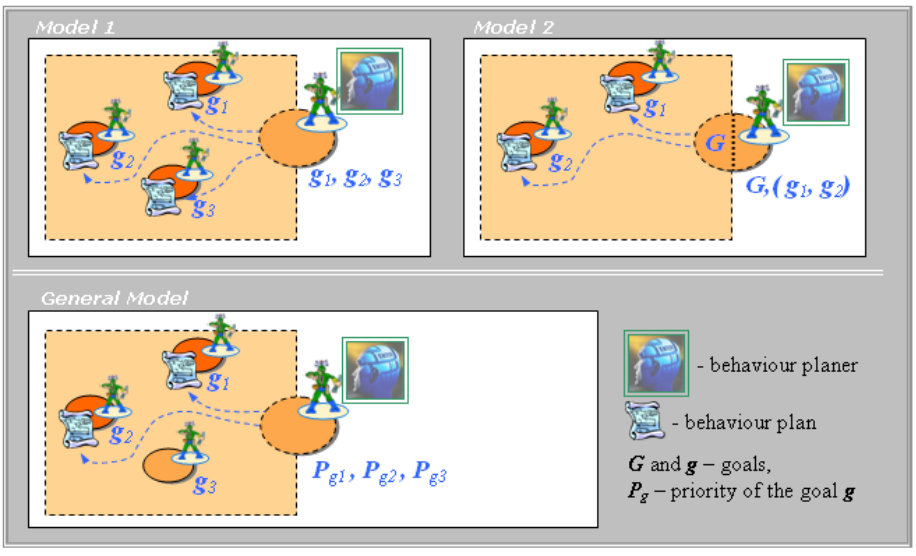


FIGURE 17 Process coordination models

Nobody can guarantee the stability of environmental data if the data space is shared among several processes. Depending on the changes the behaviour might need replanning. The optimal way to reduce the amount of replanning is to collect all the processes, sharing the same data space, under one upper process, if possible.

Generally, all behaviours are represented by a set of rules that operate with the classes of resources (not the concrete instances). But during behaviour processing by the Behaviour Engine all the rules are bounded with concrete instances. After such bindings, conflict situations may occur. If two processes use different instance spaces (spaces of facts, desires, etc.), then there might be no conflicts. But, if they share the same instance space, they can block each others' process performance by changing the shared information space. Actually, while those Resource-Agents are living separately (i.e., they are not members of some bigger processes), these conflicts of performance do not matter, and each of the Resource-Agents can concentrate wholly on the achievement of its own goals. But when those two processes are members of another bigger upper process, the duty of the Process-Agent is to resolve the conflicts via setting the constraints for the behaviours of its members to reach its own goal and the goals of the members (if it has been stated in the process contract). The initial behavior of the Process-Agent contains actions such as:

- Collec all the behaviours of the process members and conversion of those behaviours to a set of rules;
- Apply an algorithm to build a sequence of actions (performance plan) for an optimal achievement of the final goal and the intermediate goals (if necessary) based on the behaviour rules of the subprocesses;
- Set the constraints on the behaviours of the members for conflict situations (when several rules, resulting in a different Environment State, might be applied). In other words, we need to define and provide meta-behaviour rules for the subprocesses.

Such constraints (for process behaviour-rules) change the behaviour of the Resource-Agent and restrict the degrees of freedom for the agent. Actually, with its degrees of freedom, a subprocess sacrifices something to the upper process when it becomes a part of it. This does not always negatively affect the subprocess's goal achievement, often the opposite is true – the result can, in fact, be a speedup for the goal achievement.

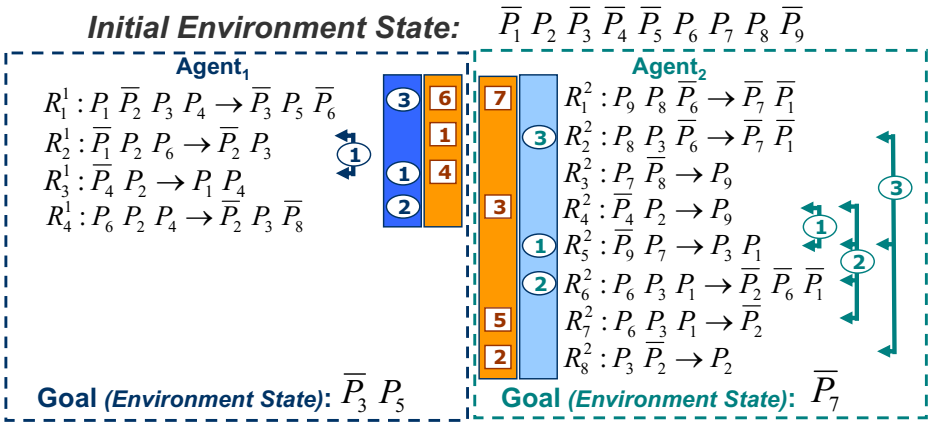


FIGURE 18 Separate Agents' behaviours

Let us consider an example that explains some issues related to process integration. For the easiest rule representation we will use the Production Model of knowledge representation. Where P and $\overline{P}$ are certain statement and negation of it, and R and $\overline{R}$ are active (ready to be executed) and passive (blocked from the execution) rules that show how set of statements (state) results in another state. However we should remember that an agent operates with the RG/BDF behaviours, not with the rules. RG/BDF behaviour is a subclass of RG/BDF rule and has Execution (Executable module) in the right part. In turn, each Execution (action performance) brings certain changes to the Environment State. In FIGURE 18 we can see two sets of rules that are behaviours of two separate agents ($Agent_1$ and $Agent_2$). Each of them has its own goal substate for the Environment (a subset of the Environment statements) and shares the common State of Environment. The numbers in the circles show the order of rules according to the possibility to apply the rule for current state. Each rule is set to optimally achieve the corresponding goal. Also from the arrows you can see the rules that can be applied at the same time for the current State of Environment. The numbers in the squares came from the next figure FIGURE 19 to show the difference between anisochronous behaviours of the $Agent_1$ and $Agent_2$, and synchronized behaviours of them under Upper-process Agent guidance.

In FIGURE 19 we can see the rule set of a process that is an upper process for the previous two processes ($Agent_1$ and $Agent_2$). This rule set is a combination of the subprocesses' rules. The figure shows the final goal of this upper process and the order of rule application. This order is also shown in FIGURE 19 (numbers in squares). And now we can see that the order of rules in the subprocesses is different from that in the upper-process. This is because the upper process is aimed to resolve the conflicts between the subprocesses and to organize their cooperative work.

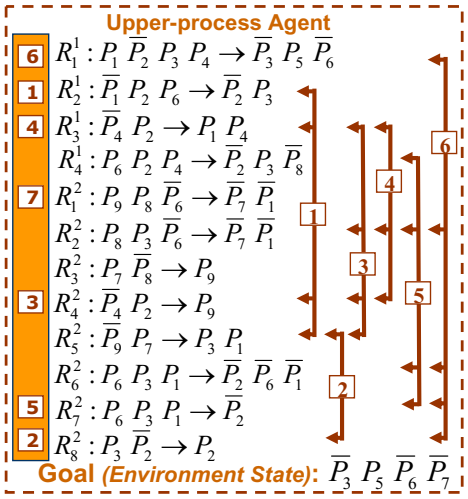


FIGURE 19 Upper-process Agent behaviour

For example, the rule order $\{R_3^1, R_4^1, R_1^1\}$ of Agent1 is the optimal one to reach the goal. Rule R_3^1 will be applied, even though on the first step any one of the rules $\{R_2^1, R_3^1\}$ could be applied. However, applying rule R_2^1 would stop the process. This is because P_1 and P_4 could not be achieved any longer until some other process updated (changes) the Environment State with P_1 and P_4. But for the upper process, which is aimed to achieve its own goal and the subgoals (goals of the sub processes), rule R_2^1 should be applied first, because the rule order $\{R_2^1, R_8^2, R_4^2, R_3^1, R_7^2, R_1^1, R_1^2\}$ is the optimal order for conflict resolution and for the achievement of all the goals. Also rule R_4^1 should not be applied in any case, because that would stop the second process (Agent2). Statement P_8 would not be achieved by this process.

Taking into account all the above, the main functionality of the upper process is to define the rule constraints for subprocesses in order to realize their orchestration. This brings us to the meta-rules for agents behaviours. FIGURE 20 shows us meta-rule enhanced Agent Behaviours (Process1 and Process2). Now the rule order of the upper process is determined by the constraints of the subprocesses' behaviour rules. But, as we mentioned before, an agent operates with the RG/BDF Behaviours, not with the rules. In this case, meta-rule enhancement means that the agent's behaviour rules set extends with additional behaviours that play the role of a meta-rule and alter the behaviour rules' conditions.

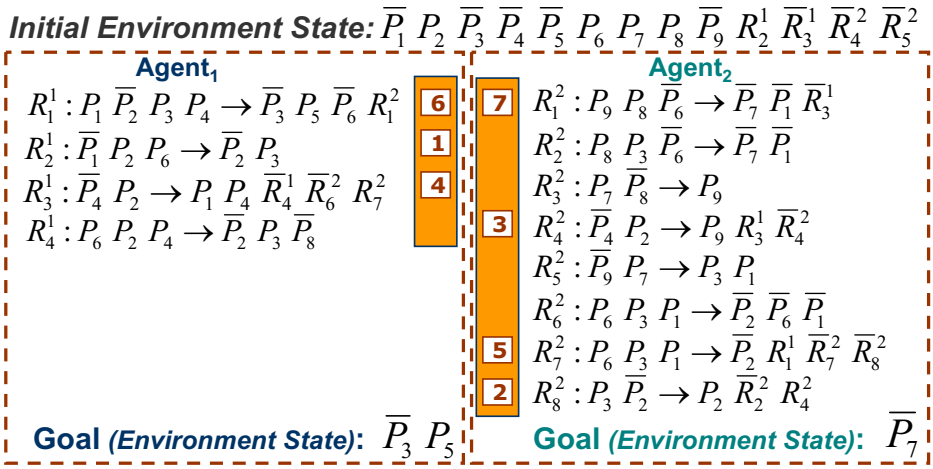


FIGURE 20 Coordinated Agents' behaviours

2.5 Human as a Smart Resource in the Semantic Web

Computer-Human Interaction (CHI, also abbreviated as HCI, for human-computer interaction) is the study of the ways that people use computers. It also refers to the very intense practice of making computers easier for people to use. However, computers can use people also, and the question arises how to make it easier for a computer to use people. A human in such a context is considered an "Intelligent Web-Service", which can be "invoked" by an external application and utilized, e.g., as an intelligent "query-answering machine". For this it is necessary to consider a special type of interface, i.e., a computer-human interface vs. human-computer one. Such an interface should be able to visualize, to a human, an application query in a way he/she can understand it. After the query is answered by the human, it should be coded back to an application understandable format. Such services can be considered as resources of the Semantic Web. These human-services would need to be taken into account and might create some problems when utilizing existing approaches (e.g. OWL-S (OWL-S, 2003) for annotating Web-Services based on ontology. Also, concerns related, e.g., to web service discovery and integration might obtain special features when human resources are taken into consideration.

Formerly, the human was just a user of other resources in the Web, and was not regarded as a resource in Semantic Web itself. However, as a matter of fact, humans are very active and dynamic resources. According to a modern vision of Semantic Web Environment, humans can semantically discover and utilize not just an electronic document via a browser, but also they can discover and access any smart (connected via sensors) resource from the real world. At the same time, the human is an intelligent object (resource), which can be useful for other resources (other humans or even software applications) as a service (expert in specific domain) or an information source. That is why we think reasonable to consider a human as a resource (web-service or agent), which can be semantically discovered in the Web, queried and used as any resource of the virtual world (application, service, and agent) or as any resource of the real world (humans, smart-devices, etc). There are many emerging industrial applications, which depend on such resource. This makes sense in new automated industrial environments where, guided by an embedded intelligent system, a smart device performs maintenance activities utilizing domain-oriented maintenance services and human-experts. Humans can be motivated to become a part of the web services, especially in business environments where they can get money or other benefits through their own knowledge and utilization of their capabilities. Of course, were this the case, these humans should be certified as a web-service and should be responsible for their decisions and activities.

2.5.1 Human Adaptation

Human Adaptation here means adaptation to both human-software and software-human directions. A human component must be connected to the appropriate Web service environment via a software interface (HumanOntoAdapter), which should be able to react and take into account all the changes/modifications of a resource (human) and the state of the environment (other resources). In other words, human adaptation should take into account the context of a situation/scene. Human interface must have a flexible intelligence for automatic registration of a change and, on the other hand, an unobtrusive request interface to a human it represents. Interaction between the HumanOntoAdapter and a human must be organized in an understandable form for the human and partly in a way of semantic conversation (context-sensitive resource visualization, semantically rich natural language requests and answers, etc.). A more detailed description of the context-sensitive resource visualization and human adaptation approaches will be provided in the next section (Section 2.6) and the second section of chapter 3.

In such Semantic Web based intelligent environments a human being can play various roles. He/She can be a user of the environment (environment platform administrator), be presented as a service that provides knowledge to other resources (expert in a certain area/domain) and as a device to manipulate as well. Considering a human as a resource that can be manipulated by other resources, integrating such human representative into an automated resource environment opens up a new market for the producers and providers of personal mobile devices. Connecting a real world resource (human) with its representative from virtual world becomes physically possible. Why is a physical contact so important? A human being can change him- or herself in any moment of his/her life. This also applies to his/her preferences when buying, ordering, or renting something, or during any other activities. That is why it is very important that a personal assistant takes part in these activities. Of course, a human can change information related to him/her via other points of access: when changing the information on a website, or when filling some form, etc. In cases like this, automated synchronization of all these *access objects* would play a very important role in combining and storing of HumanResource information. In other words, information related to a particular human should be collected via any personal devices, sensors, and applications which are used by that human in his/her real life.

2.5.2 Ontology personalization

A user interface for a human presents some problems especially in interactions with any personal human component representing the human in the integrated OntoEnvironment. Problems might be encountered with any other service equipped with a human interface, or with tools that deal with ontologies. How

to construct an ontology to enable interoperability among humans in a (comfortable) way that an ontology designed to provide interoperability between applications does? There are some special considerations to be taken into account here. There are also many fundamental differences between people:

- Linguistic and cultural differences.
- Different notions concerning one (same) entity.
- Different perceptions of the incoming information.
- Etc.

This problem can be found to occur also in business environments, in the e-commerce domain. The problem emerges, because the internal information representation of each enterprise differs from the others, and we cannot force these players to use a common (standardized) language for the management of their own information.

One solution for this problem is *Ontology Personalization*. *Ontology Personalization* means development of a support mechanism for an ontology with two sides. Each player will be able to create a personal ontology; in other words he/she can describe each object from their common ontology (or an often-used part of it) in a desired way (language, terms, notion, etc.) Ontology Personalization would be a human adaptation on the *semantic layer*. Thus, we would have a two-sided ontology. One of the sides would have a common ontology, which could be used by any semantically enabled resources (elements of the OntoEnvironment). The other side would have a personal ontology in a form understandable to a concrete player. For this we would need some mechanism for ontology interpretation (translation) for both sides of the two-sided ontology. The goal of such *OntologyInterpreter* is two-way interpretation of the the input and output of the human user interface (FIGURE 21).

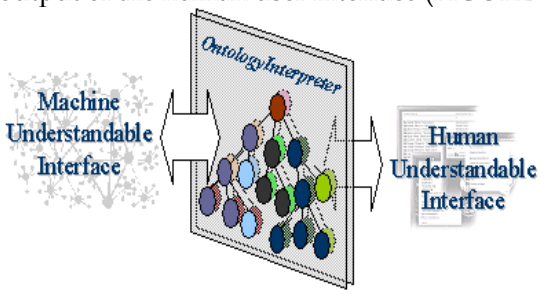


FIGURE 21 OntologyInterpreter – ontology personalization module

There are many different aspects of the ontology personalization:

- *Language and terms difference.* Each player has an own language and may have own names for the terms (slang). In this case, Ontology Personalization mechanism allows an interaction between a player (smart software or human) and OntoShell (adapter for a software resource or a human interface) in the existing personal language of the resource. This would resolve the problem of the heterogeneity of the players' languages.

Personalization on this level can be realized by utilizing, e.g., a vocabulary of synonyms.

- *Meanings difference of the same entity.* The same entity can have different meanings for the different players depending on the domain of the player's activity. It does not mean that the player never uses some meaning of a notion. It means that the player uses the notion with an appropriate distribution of probabilities of its meanings. Consider an example, where an entity, "Driver", has a set of the meanings: a name of a movie, a special software for a device, a car driver, a special equipment for golf, and a game (race emulator) (FIGURE 22). On this personalization layer, a multiform vocabulary with usage features and a mechanism for context definition plays the main role.
- *Temporal expansion of concepts.* Sometimes a player may use a persistent long-term notion or concept, which defines not just some object or entity, but also a set of its properties and relations (ties) with other entities. For example, the notion of planning a trip includes ordering tickets (kind of transportation, convenience, price range, etc.), finding an apartment (also a number of properties is specified), etc. Such set of the notion's attributes may be used by the player to serve as a personalized notion. Personalization can also be used for improving semantic search: a simple search query is assumed to be semantically enhanced when the Ontology Personalization mechanism is used with it.
- *Personalized representation of information.* Ontology personalization helps in generating a personalized view about the information for each player. A player can get and set information in a way best suitable for (him/her). This information representation ontology is the basis for a personalized and dynamic visual interface for a human. It uses a personal user semantic profile in its task. Some work related to personal information representation issues has been done in awarded by TeliaSonera Finland Oyj[19] research (Khriyenko, 2005) (see Section 5.4).

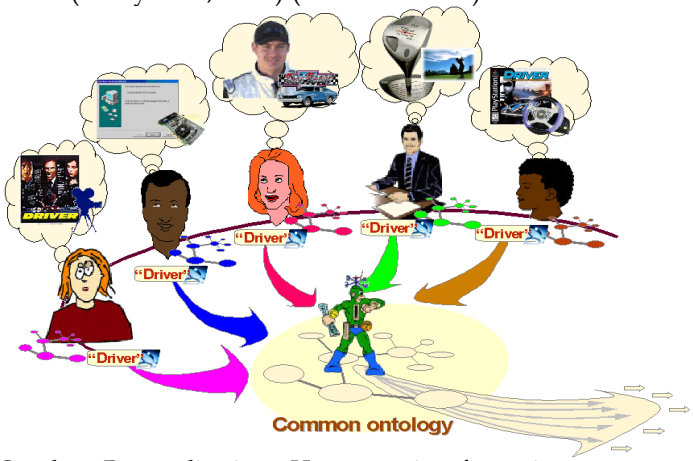


FIGURE 22 Ontology Personalization – Heterogeneity of meanings

[19] http://www.teliasonera.com/

Such ontology personalization mechanism must closely interact with the player's profiling system. The player's profiling system will provide access to the personal ontology regardless of the way it is connected to the OntoEnvironment (wired or wireless connection).

The discussion about ontology personalization implies that *OntologyInterpreter* is a complicated aggregate module for the human adaptation mechanism (HumanOntoAdapter), and must be supplied with an intelligent and human-friendly tool for personal ontology creation and training.

It is not obligatory to create a complete personal ontology if a player uses just that part of it which concerns a specific domain of his/her activities, or when storage space is restricted, e.g., in a personal mobile device. For overrunning the available part of ontology, an ontology swap-in mechanism (OntologyCaching) can be used (FIGURE 23).

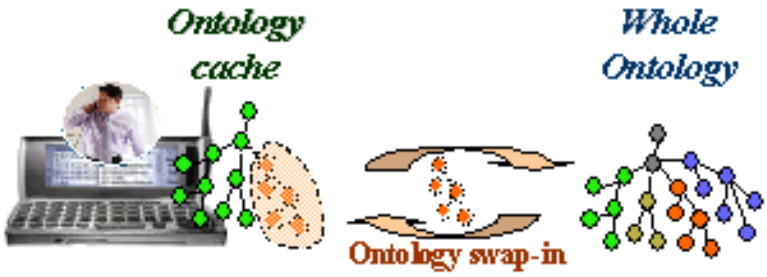


FIGURE 23 Ontology Caching – swap-in mechanism

The development of dynamic and flexible user interfaces plays a very important role for ontology personalization. Such interfaces have to provide the means to create, change and learn a personal ontology. The goal is to provide a more suitable way to manage a personal ontology, and each application must be supplied with such an interface.

The "Language and terms difference" aspect and "personalized representation of information" may be managed by creating and changing the personal ontology. However, aspects such as "a variety of the same concept meanings" and "temporal expansion of concepts" of an ontology personalization require a special learning mechanism based on stored information related to a human player. Ontology personalization learning helps human interfaces to become intelligent, enabling human presence in an OntoEnvironment more easily and with more comfort.

2.6 Intelligent Resource Visualization

According to Nixon (Nixon, 2006), as the current trends develop we expect to experience a future Web which will be media rich, highly interactive and user oriented. The value of this Web will lie not only in the massive amount of

information that will be stored within it, but in the ability of Web technologies to organize, interpret and bring this information to the user. A graphical user interface will be one of the important parts in these processes. Media presentation is a key challenge for the emerging media-rich Web platforms. Emerging technological trends strongly suggest that it is time to initiate a new stage in multidimensional resource visualization (visualization of resource properties, contexts of inter-resource communication and interaction) and a new stage in semantic metadata based visual browsing across resources.

2.6.1 Motivation for intelligent resource visualization

One of the reasons for intelligent resource visualization springs from a necessity to enhance resource search and browsing processes. According to Marcos et al. (Marcos et al., 2005), there are a number of important criticisms that can be made of the Classical Model of information search. The model does not adequately distinguish between the needs of a user and what the user must do in terms of querying to satisfy those needs. Very often, users may not know how to create a good search query, even when using natural language terms. Often the first query attempt, rather than yielding useful results to select from, begs a question about what is there to be searched over. The second important criticism related to the Classical Model is that any knowledge generated during the process of query formulation is not used later on in the sequence of search process steps to influence the filtering step, to present the search results, or to select the results. Finally, the Classical Model provides an essentially context-free process. There is no proper way in which knowledge of the task context and situation and user profile can have influence on the information search process.

To address these criticisms, the WIDE Model of information retrieval (Marcos et al., 2005) treats the general task of finding information as a kind of design task, and not as search specification and results selection tasks. Information retrieval, when understood as a design task, first recognizes the difference between users' stated needs and formation of well-specified requirements, and then properly supports the incremental development of complete and consistent requirements as well as re-use of the knowledge generated in this (sub)process. Thus it effectively supports the subsequent steps in the process that concludes in a useful set of search results. There are several projects that aim to somehow enhance the Classical Model of information retrieval. For example, the problem of search query uncertainty afflicted the ”Semantic Facilitators for Web Information Retrieval”[20] project of Industrial Ontologies Group. The main idea behind that project was that Semantic Search Assistant/Facilitator (SSA) using ontologically defined knowledge (WordNet[21]) about words from Google search requests, allows the user to specify the right

---

20 SemanticFacilitator -www.cs.jyu.fi/ai/OntoGroup/SemanticFacilitator.htm

21 WordNet - http://wordnet.princeton.edu

meaning of the words from a set made thus available. Further, based on the description of a selected word meaning, SSA uses embedded support for advanced Google search query features in order to construct more efficient queries from the formal textual description of searched information (Kaykova et al., 2004). This shows that a lot of work is going on in the area of the classical information retrieval model. New, useful features are being continuously added to enhance the model. Even more, nowadays there are many efforts aimed at the creation of a fully ontology based semantic query and search mechanisms, where search query is created based on ontological concepts' specification. But this is a complicated and challenging task. It is not evident how to detect the user needs and how to provide only relevant ontological concepts for the user during query specification.

The growing interest to Wiki-based systems makes it likely that in the near feature we will have a huge mass of information to deal with. However, being unintelligible for machines/software, and very difficult to search/browse for a human, this information will be quite useless. Today, the interest of the experts is starting to turn towards Web 3.0, which term that has been coined to describe the latest stage in the evolution of Web usage and interaction. The concept includes transforming the Web into a database, making content accessible by multiple non-browser applications, leveraging of artificial intelligence technologies and the Semantic web, and three-dimensional interaction and collaboration (Web3.0). According to Wikipedia, Web 3.0 is the final step in the decomposition of monolithic Web Pages into Presentation (HTML and XHTML), Logic (Web Services APIs), and Data (Data Models). It is a trinity that moves the data from the web pages to web data. Its emergence simplifies the development and deployment of Data Model driven composite applications that provide easy, transparent and organized access to "the world's data, information, and knowledge". Web 3.0 thus promises to be much more useful than its predecessor, Web 2.0, and to render today's search engines more or less obsolete. To fit to the vision of Web 3.0, Wiki-system content can be annotated (using Semantic MediaWiki) and knowledge of a given subject (or domain) can be applied to build intelligence into Wiki. Semantics can thus be added to the Wiki content, but search and browsing of that annotated information are treated as non-reinforceable issues.

Thus, when we consider the vision of GUN (where all the resources of the virtual and the real world are connected and interoperate with each other) and the huge amount of information involved, it is clear that we have to elaborate new visualization techniques that simplify the search and browsing processes by reducing the number of queries via context-dependent resource visualization. We need somehow to visualize the resource properties, the various relations between resources and the inter-resource communication process. And even more, we should make this visualization more context-dependent, to be able to represent information in a handy and adequate way, to achieve the required plasticity for UIs (Thevenin and Coutaz, 1999). Thus, the main focus in GUI development will be on the resource visualization aspects.

We are left with the challenging task of *semantically enhanced, context-dependent, and multidimensional resource visualization*.

2.6.2 Visualization of a resource

Several information visualization techniques have been developed in the past few years due to the need of representing and analyzing a huge amount of data generated by several applications or made available through the World Wide Web. In the beginning, information systems had to deal with data visualization, or with data format representation, to be more precise. Depending on a data format - be it text, image or video - a graphical user interface presented that data in a certain way. Soon, visualization of object part-of relations, in a form of a tree, became a useful step in the development. Later on, with the semantic definition of the objects, there was a need to represent an ontology concept tree and semantic graph (Yuxin et al., 2005). Recent expectations about the Web development strongly depend on the success of Semantic Web technology. But it is going to be just a small step to the semantics and ontology representation.

Information visualization aims to provide compact graphical presentations and user interfaces for large numbers of items that are to be interactively manipulated. Information visualization is the study of how to effectively present information visually. A lot of work in this field focuses on creating innovative graphical displays for complicated datasets. It is becoming evident now that we cannot separate the visual aspects of both data representation and graphical interface from the interaction mechanisms that help the user to browse and query a data set through its visual representation.

The problem with manipulating a huge amount of information is the complexity of search query specification and provisioning of the relevant links for content browsing. The idea of intelligent resource visualization is to simplify the search and browsing processes via associative resource visualization. Multidimensional associative resource visualization means visualization of a resource depending on a context, via association with various aspects of the resource (relations with other resources, domains, areas of interest, etc.). Sometimes we cannot specify exactly what we are looking for, but we feel that it is somehow related to a certain matter, a certain situation, a certain context. Such visualization can give us a hint, turn us to the right direction, show us related objects and provide links to them. In other words, visualization will utilize context-based filtering and enrichment of the visualized scene with the relevant links.

All the resources have a set of properties, and if we consider that all the resources are parts of the world and all of them are related and somehow linked to each other, then we have a very huge amount of resource properties. It seems it is impossible to elaborate a handy query system that will operate with all these properties. Thus, the context-based approach is a great solution for resolving this problem. In a certain sense, context refers to extraction of the

resources, some of their properties, and relations that are relevant to a certain situation, action or other aspect of the resource (see FIGURE 24). Applying context-dependent visualization we reduce the number of the "steps" on a path leading to the final destination. Thus, a context can be a basis for a specific visualization view of a resource and other resources related to it. For example, a "part-of" relation (if it concerns a physical relation of resources) can be visualized as a 3D model of nested or somehow connected resources; or as a 2D model, if the third dimension is not valuable (for example when presenting the resources on a map, if they are part-of the world). On the other hand, a "part-of" relation of a resource can be abstract and may have no physical contact with another resource. A resource can be a part of some (business) process. In that case, there is no physical contact between the parts, and such relation should be represented in a different way (for example as a relation graph).

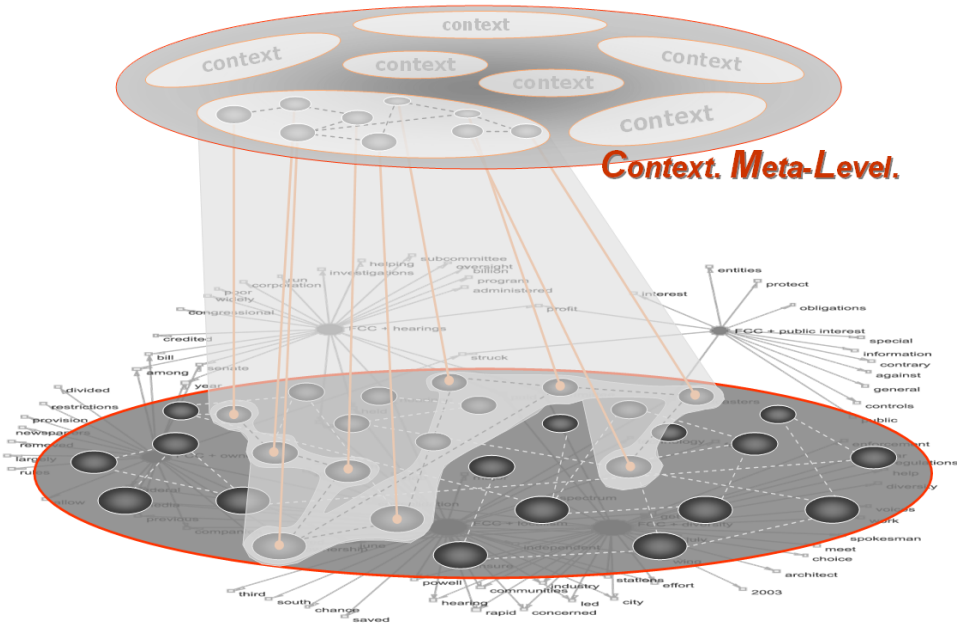


FIGURE 24 Context. Meta-level extraction

To show the different dimensions of a resource, let us consider a "person" resource (FIGURE 25). The visualization of a "person" in the context of healthcare and the condition of one's organs can be shown as a human body diagram (with a view of the organs). This would allow an expert/doctor to check the organs' condition (based on its visualized properties), the influence of the organs to each other and the body systems; to switch the view to the internal view of the necessary organ and to manipulate the related information. At the same time, a "person" resource in a context of healthcare and the location of a healthcare organization can be visualized in a form of a map which highlights the location, whether it is an organization which belongs to the person's employer or just the nearest organization to the person's location.

Another visualization dimension of a "person" resource can be occupation/profession. It can be realized by visualizing a working area/place with the relevant work-related links: duties, area of interests, professional resources, contacts, etc.

For example, if the "person" is a goalkeeper of a football team, then its visualization in a context of profession can be realized in a form of a team on a football field. Then we have an access to the other team members (know their roles in the team), have a link to the stadium (the "stadium" resource, which provides facilities for training) and to the home "team" resource. Another context for a "person" resource visualization can be a family relation of a person. In this case, visualization can be performed in a form of a genealogical (family) tree. Several other examples of context-dependent resource visualization can be found in the figure below (FIGURE 25).

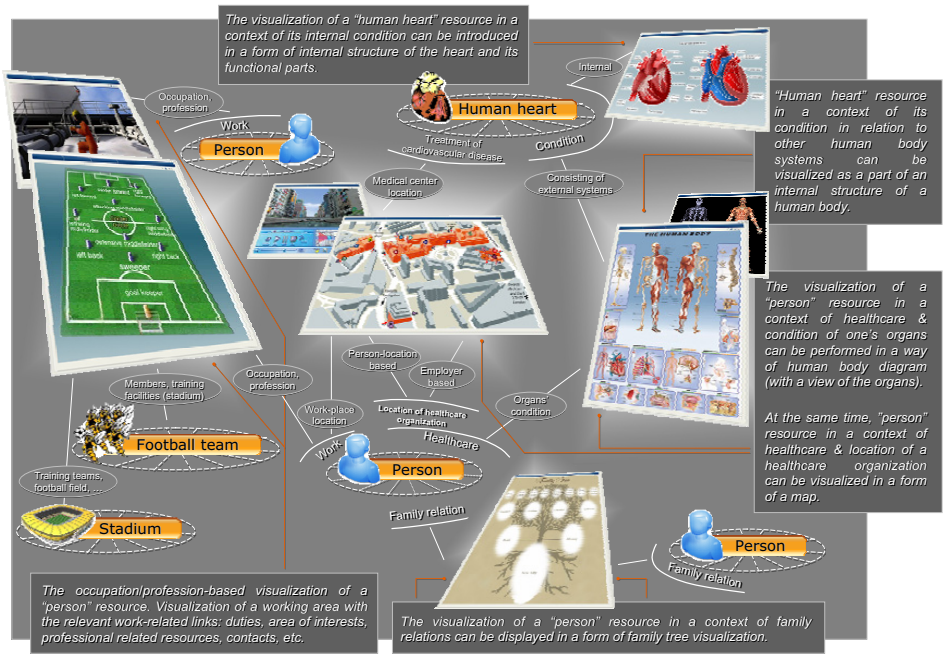


FIGURE 25 Resource visualization

Depending on a context, a human/expert needs information (information related to subject resource) to be visualized in certain way. This gives us one of the requirements for visual interfaces – ability to represent information in line with the chosen contextual property of a resource or the contextual situation. Such interface would allow the user to simply choose a context for the data representation, and would even support cases of multiple contextual property selection for complex views and filtering purposes. Choosing the appropriate representation(s) is challenging, and research is needed to evaluate and compare different approaches. Also, different visualization modules are specialized and present resources in certain domains. Such domains can target a

restricted as well as a wide set of resources. For example, one of the restricted domains can be anatomy and the processes between the parts of the human body. In this case, a visualization module can present the information based on a 3D human body visualization and should be able to visualize the properties from the anatomy domain. This task becomes more challenging in case of a wide target domain. For example, we might need to represent information via a spatial resource location view and cover all of the resources, and the interface should somehow visualize all the properties. Thus, the next requirement for a visual interface is to be able to visualize the properties of the resources that belong to the domain ontology in question.

2.6.3 Utilization of Intelligent Resource Visualization in next-generation systems

There are already some developed domain-oriented software applications, which try to visualize domain specific data in a way suitable for humans (graphics software from SmartDraw®[22], concept-browser Conzilla[23] and Human Semantic Web browser Conzilla2, Google Maps, etc.). But still, the technologies are developed for specific standalone domain-oriented applications. And when we face a real need in an open unlimited collaboration environment (Web 3.0), we will have to develop many more visualization tools and modules to visualize various resource properties, contexts, situations and associations in order to provide a flexible and handy Human-Machine interaction interface. Thus, semantic-based, context-dependent , and multidimensional resource visualization approach can form a basis for the development of such an interface.

It is well known that the 3rd dimension provides new ways for organizing, presenting and interacting with content. Ideally 3D will make computer interaction easier, more intuitive and more natural. 3D is also very useful for presenting great amounts of data in easily understood 3D visualizations or graphs. That is why Web3D Consortium[24] aims to promote open standards for Real-Time 3D Communication. Octaga AS[25], a producer of world class real-time 3D software products, creates outstanding content for real-time 3D, while Media Machines Inc.[26] is a leading provider of 3D virtual worlds software and services for the Web. Some analysts remain divided over whether 3D on the Web will be of much interest to a general audience. At the same time, marketers have reported success with 3D retail environments in which products can be rotated and manipulated in three dimensions. Other recent growth areas include multi-user games, e-learning applications, data

---

22 SmartDraw® - www.smartdraw.com

23 Conzilla concept-browser – www.conzilla.org

24 Web3D Consortium - www.web3d.org

25 Octaga AS - www.octaga.com

26 Media Machines Inc. - http://william.mediamachines.com

visualization and warehousing, and collaborative design and engineering. The resource visualization approach presented can be utilized in the development of Web 3D applications to enhance them with more natural and associated resource visualization and context-dependent browsing technique.

Intelligent Resource Visualization plays an important role in 4i (FOR EYE) technology (see Section 2.1 in Chapter 3). That technology can be considered as a valuable extension of the text based Semantic MediaWiki. It forms the basis for context-based Visual Semantic MediaWiki, a new generation resource collaboration environment that follows the vision of Web 3.0.

Now, when the human is becoming a very dynamic and proactive resource of a large integration environment with a huge amount of different heterogeneous data, it is quite necessary to provide a technology and tools for easy and handy human information access and manipulation. Semantically enhanced, context-dependent, and multidimensional resource visualization provides an opportunity to create an intelligent visual interface that presents relevant information in a more suitable and personalized form for the user. The proposed resource visualization approach can be utilized in various visual systems and especially in next-generation human-centric open environments for resource collaboration with enhanced semantic and context-based visual resource browsing.

3 ONTOSMARTRESOURCE DESCRIPTION FRAMEWORK (OSRDF)

Section contains the main contributeion of the thesis and presents OntoSmartResource Description Framefork OSRDF - logical extension to the RDF. We present a new context-oriented vision about statement and a property representation and approach for role based resource proactivity description. Thus, OntoSmartResource Description Framework is a common framework that allows context-sensitive description of the states, conditions, goals and behaviours of Smart Resources in a new generation of the Web.

The amount of data within the World Wide Web is increasing continuously. It is becoming more and more difficult to retrieve relevant information by using the current search engines that are based on pattern matching. The Semantic Web approach pretends to solve the problem by annotating resources and enabling semantic search engines. The key issue is the ability of machines to "understand" the content of resources not only at the syntactic but also at the semantic level. To standardize such annotations, Resource Description Framework is used by the W3C consortium as a framework for managing metadata on the Web and as a basis for other Semantic Web languages, technologies and tools. The emergent RDF is expected to enable metadata interoperability across different communities and applications by supporting common conventions on metadata syntax, structure, and semantics. RDF data can be regarded as a set of atomic sentences, each having a subject, a predicate and an object. These sentences are also called RDF statements or triples. Systems and tools for managing metadata repositories of RDF triples already exist.

3.1 Context-Sensitive Metadata Description

Storing triples without being able to track back their original source (producer of the statement) or denote the condition under which they are true is not sufficient for many applications. Especially in RDF, which provides a possibility for everybody to say anything about everything, it is mandatory for the users to know the context of the given information (source, time, place and any other contextual identifier). Context is a form of information that can be known and hence encoded and represented much as other information is encoded and represented in software systems. In the absence of this essential data, contradictory statements collected from a variety of sources can occur in RDF repositories, and users are not able to determine which ones they can trust. One way for making the RDF model more reliable in modeling context information is to use RDF reified statements (statements about statements are possible in the RDF syntax). MacGregor and Ko (MacGregor and Ko, 2003) point out that this solution is not practical. The main reasons are that a) it results in a proliferation of triples, b) it is difficult to read, c) it is difficult to write queries to extract relevant material, and finally d) it is much more difficult to handle and therefore less efficient.

Another problematic issue is how to give a reasonable definition for a context that is useful within RDF. There are quite a few definitions to choose from. For example, Joseph et al.'s (Joseph et al., 2000) definition for context reads: "The part of a text or statement that surrounds a particular word or passage and determines its meaning. The circumstances in which an event occurs; a setting." However encompassing this explanation may be, there is still no clear and universally accepted definition for context in the area of knowledge base systems. Jansen (Jansen, 1993) presents an overview of existing interpretations of the term *context* in the area of knowledge base systems. Cyc technology[27] resolved some of the issues concerning context description in knowledge base. Cyc Microtheory that was used in the task is an abstract informational thing that represents a context in Cyc.

We have at least three different situations where the term context is used in RDF. First, there is the context given by the surrounding graph, i.e., an internal context. The way how to handle and interpret this internal context is discussed by Hayes and McBride (Hayes and McBride, 2004). In the second situation, we talk about an external context, such as source information, time of creation, name of the author and others, which normally are not included in the RDF model itself, though they could be. Finally, there is the context used to identify triples for a clear and easier handling of sets of triples, e.g., to merge/unmerge graphs (since this identification is not coming from inside the RDF model) (Tolle and Wleklinski, 2004).

---

27 The world's largest and most complete general knowledge base and commonsense reasoning engine (www.opencyc.org).

A common argument against quads (the fourth "context" component that can be added to the existing RDF "subject-object-predicate" triple) goes "We have triples; now you want quads, where is it going to stop? Quintuples? Sextuples?" The answer of MacGregor and Ko (MacGregor and Ko, 2003) is that quadruples is all you need (their well-educated guess). Some RDF systems (Jena's RDB model, for example) internally implement a quad structure that adds a model column to the subject/predicate/object columns. This allows mapping the model to a set of statements. It might seem that adding contexts to their quads would turn the quads into quintuples. Although one could add a fifth context column, a better solution is to convert the models column to a context column and adopt the convention that each context belongs to exactly one model. That way, we have quads, and we can also directly map each statement to a model through its associated context. We agree with the argumentation of MacGregor and Ko (MacGregor and Ko, 2003) and also think that the triples-plus model architecture can be converted to a quad architecture with no significant increase in storage requirements.

There is a case for a logical extension of RDF to Context sensitive Description Framework. RDF is a base for higher levels of computational semantics (for example, OWL), and that is why we decided to make the extension on a lowest level. The aim here was to extend the resource description language, not the ontology representation. Even reification would not help us in this, because we had to consider the context exactly as it is when associating it with a subject statement. Even more, here we are dealing with a restricted range of contextual properties (predicates of contextual statements of a subject statement) for subject statement's predicate. In this case, we think that a quadruple for a statement representation is the best solution.

3.1.1 New vision of a statement and a property representation

In our vision all properties have some sense in a certain context, which should be specified by the context tolerance range. Thus we have a need to define a contextual range for a property, which plays the role of a statement predicate. Such approach to the property definition gives us a new vision of statement representation. Each statement may be true or false in relation to the different conditions of an environment. In this case we consider the context of a statement as a set of other statements that describe a certain condition – the state of an environment. Such descriptions of environmental properties may contain also the source of the statement descriptions, and thus provide opportunities to manage trust in distributed systems. Each contextual statement itself may also have its own context (i.e., we have a nested context). A nested context allows vertical in-depth reasoning based on context-sensitive descriptions. It is obvious that using a triplet-based model for a statement-in-context description is not a good idea, and therefore we use quadruples for modeling, the fourth component being a container of contextual statements.

Not wanting to contradict much with the existing standards, we have elaborated a contextual extension of *statement* in RDF (Khriyenko and Terziyan, 2006). OntoSmartResource Description Framework (OSRDF) is a logical extension of RDF and is meant to model the context dependence of world properties. It allows us to take two significant steps in the resource description approach. We logically go from a duplet (domain-range) vision of a property description to a triplet description (domain-range-context), and from a triple representation of a statement to a quadruple representation (statement in a context of other statements).

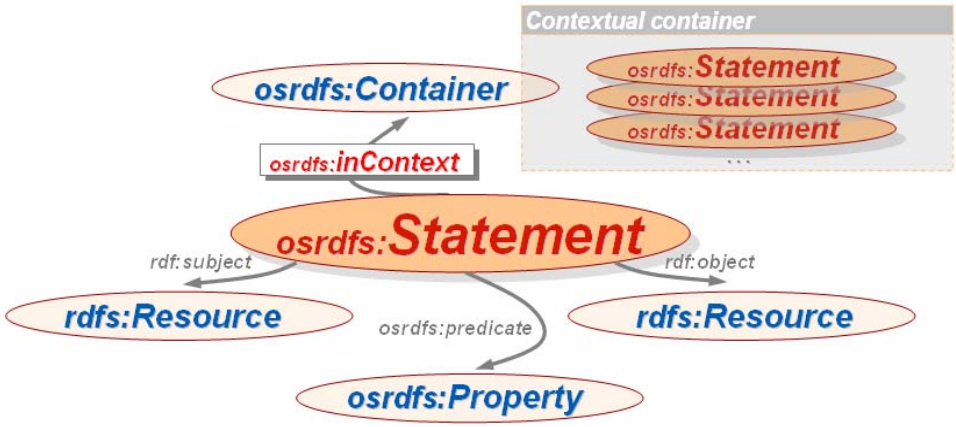


FIGURE 26 A quadruple vision of statement

An OSRDF quadruple, as defined here (see FIGURE 26), contains four components: a subject, which is an RDF URI reference or a blank node; a predicate, which is an RDF URI reference; an object, which is an RDF URI reference, a literal or a blank node; and a contextual container (context), which is an RDF URI reference or a blank node. An OSRDF quadruple is conventionally written in the following order: subject, predicate, object, contextual container. A predicate is also known as the property of the quadruple. With a purpose to define an OSRDF quadruple we have inherited the rdf:Statement class and have added the additional osrdfs:inContext property. An OSRDF statement - *osrdfs:Statement* is a statement made by a token of an OSRDF quadruple. The subject of an OSRDF statement is an instance of rdfs:Resource identified by the subject of the quadruple. The predicate of an OSRDF statement is an instance of *osrdfs:Property* identified by the predicate of the quadruple. The object of an OSRDF statement is an instance of rdfs:Resource identified by the object of the quadruple. The context of an OSRDF statement is an instance of *osrdfs:Container* identified by the contextual container of the quadruple. The *osrdfs:inContext* property has the osrdfs:Statement and osrdfs:Container classes as the domain and range respectively, the osrdfs:Container class being an inherited class from the

rdfs:Container and restricted with a content. The instances of the osrdfs:Container class may contain only instances of the osrdfs:Statement class, which plays a role of a statement context. FIGURE 26 shows a quadruple approach to statement representation.

As mentioned in the previous section, sometimes we have to describe statements in an ordered context. In order to describe a context in such format for a subject statement, OSRDF-Schema has been extended with the *osrdfs:inOrderedContext* property that refers subject statement to an instance of the *osrdfs:OCC_Container* class. osrdfs:OCC_Container (a subclass of the rdf:Container class) is a class of containers that contain a set of different order contexts. Each of the context in this container is represented by another container – an instance of the *osrdfs:OrderContextContainer* class that defines $f(\vec{x})$ of a different context order. The members of the OSRDF OrderContextContainer container are contextual statements that play a role of input parameters of direct influence for this concrete level (order) of a context. osrdfs:OrderContextContainer is a subclass of the osrdfs:Container class, and is extended with the *osrdfs:relatedRule* property that refers to a certain *osrdfs:RuleStatement* instance and with the *osrdfs:contextOrder* property that makes it possible to define the order of a context. FIGURE 27 shows two different ways to represent context.

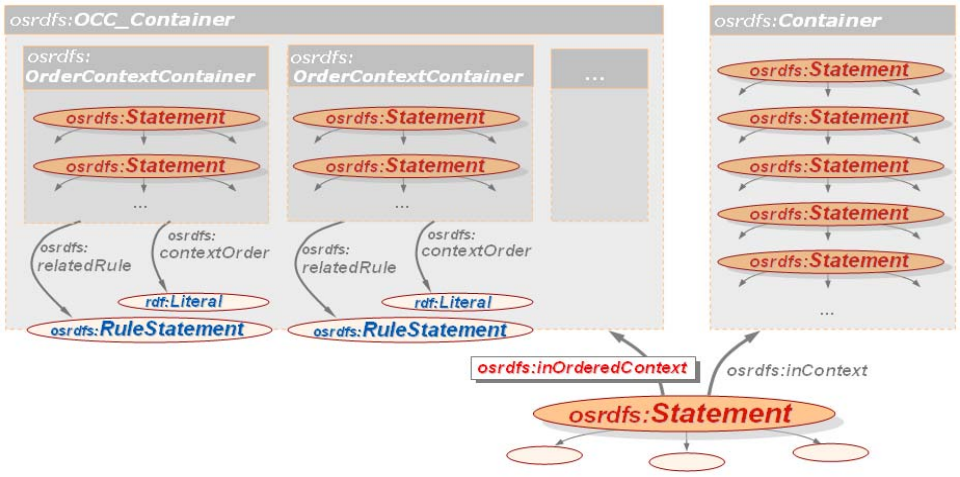


FIGURE 27 Two ways to describe context sensitive statements

Now we can describe any statement with a binding to a context. Such a context-dependent representation of a statement entails a specification of the contextual container content range according to a quadruple predicate. Thus we come to the necessity of taking one more logical step in the resource description approach and go to a triple vision of a property.

Following the first extension step we extend the existing rdf:Property, which is described by rdf:domain and rdf:range, with an *osrdfs:context* description (exactly with a "context tolerance range" definition). As the RDF

Concepts and Abstract Syntax specification (Klyne and Carroll, 2004) describes the concept of an RDF property, we describe the concept of an OSRDF property as a context-dependent relation between the subject resources and the object resources. An OSRDF triple property representation contains three components: a domain, which refers to a domain class; a range, which refers to a range class; and a context, which refers to a set of contextual properties (context range). Dourish (Dourish, 2004) also describes context as a delineable substance and states that, for some set of applications or application requirements, we can define what counts as the context of activities that the application supports and can do so in advance. FIGURE 28 shows a new triple vision of a property. As a rdf:domain property defines a restricted area (rdfs:Class) of the subject property domain and a rdf:range property sets a subject property range (rdfs:Class), the osrdfs:context property defines a vector of the properties (osrdfs:ContextContainer) that plays the role of a subject property context.

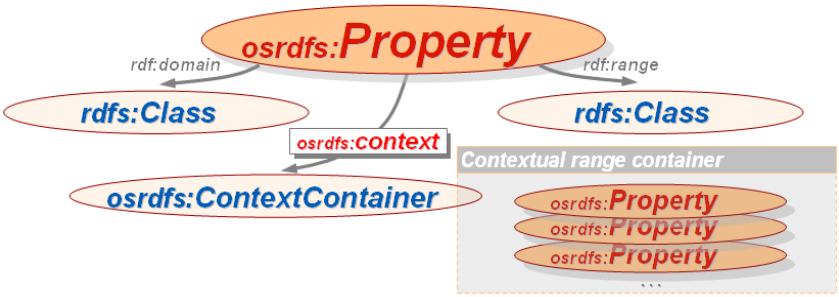


FIGURE 28 A triple vision of property

The class *osrdfs:ContextContainer* is a subclass of the rdfs:Container in a general case. It contains a set of the osrdfs:Property instances. They restrict the number of the properties that can be used as objects of the *osrdfs:predicate* property in a contextual statement description. In other words, this container specifies a range (a set of the object properties) for the osrdfs:predicate properties of the statements in the contextual container of the subject Statement (FIGURE 29).

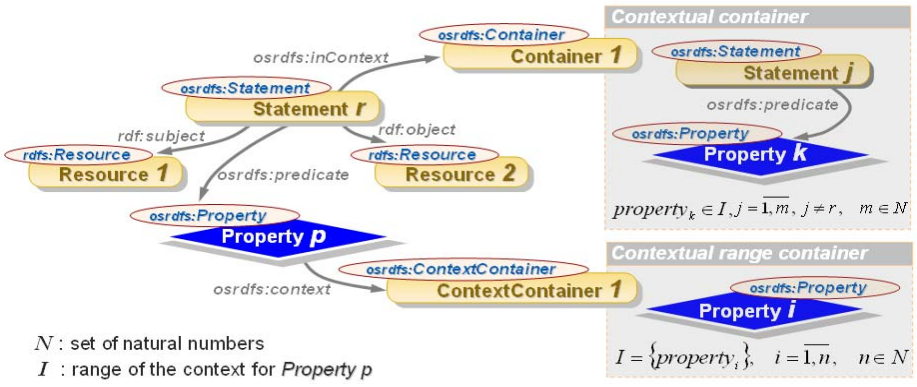


FIGURE 29 Context tolerance range definition

Due to the new vision of resource description, it is reasonable to redefine the concept of a sub-property. The *osrdfs:subPropertyOf* property may be used to indicate that one property is a sub-property of another one. If property P is a sub-property of property P', then all the resource triplets (subject resource, object resource, and inContext container) that are related by P are also related by P'. The term super-property is often used as the inverse of sub-property. If property P' is a super-property of property P, then all the resource triplets that are related by P will also be related by P'.

Three rules correspond to the sub-property definition. In the same way as in the RDF specification, the domain and range classes of a sub-property should be the same classes or subclasses of the super-property domain and range classes. Additionally, the sub-property context (a vector of the properties) should be covered by the context of the super-property. It means that each element of the subject property context vector (a property) should be a sub-property of some super-property context vector element or a new property (not presented in a super-property context vector) (FIGURE 30).

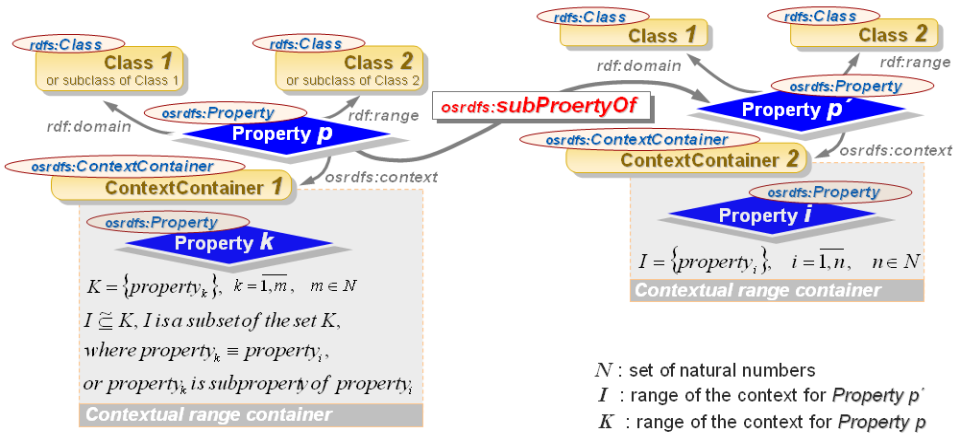


FIGURE 30 Definition of subproperty concept

Let's consider some example from an industrial domain. There are two devices, D#1 and D#2, which are instances of the *osrdfs:Device* class. D#2 is an atomic resource and a part of D#1. Additionally D#1 is a part of E#1, which is an instance of the exmpl:Environment class. The osrdfs:Device and exmpl:Environment classes are subclasses of the *osrdfs:OntoSmartResource* class. A hierarchy of relational properties is shown in FIGURE 31. The figure shows a simple hierarchy of measurement and condition properties. The specifics of the measurement properties are based on the partOf relation of the resources. The measurement of an atomic resource is a physical measurement. That is why the context of the statement that describes a physical measurement is a statement about a specific partOf relation of the subject statement.

A logical measurement has a slightly different meaning. A logical measurement of a "mother" resource is based on the "daughter" resource

condition (which is based on an own measurement). In this case the context of the statement that describes the logical measurement is the statements about the partOf relation of the subject statement (but not the atomic relation) and the condition that forms the basis for this measurement. The context for the statement about the resource condition is the statement about the subject resource measurement. We consider the values of the measurements and condition as the instances of the exmpl:QuantityValue class of the values: QV#1, QV#2, QV#3.

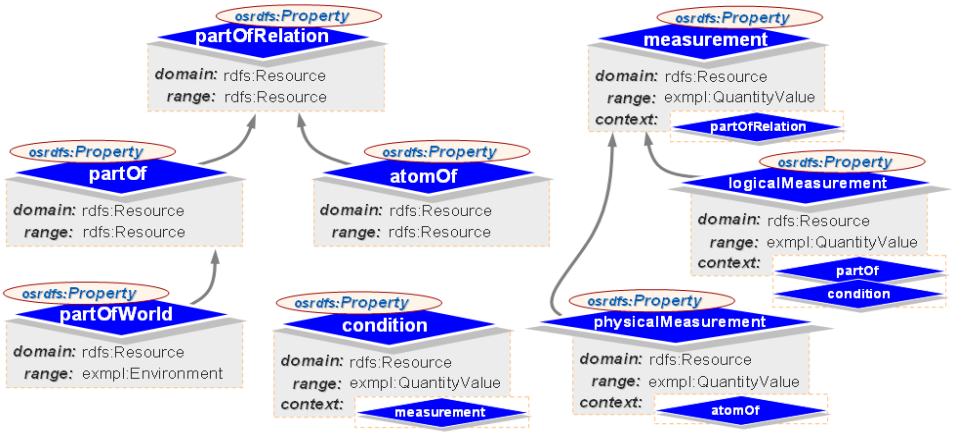


FIGURE 31 Sub-property hierarchy

Based on this set of the properties, we can describe the partOf relation between the resources, the measurements and condition with all the necessary context relations according to OSRDF Schema. FIGURE 32 illustrates the description.

From FIGURE 32 we see that the statement about the logical measurement of D#1 is true in context of other two statements: the statement that D#1 is part of E#1 (which totally fits to the context restriction for the logicalMeasurement property) and the second statement about the condition of the daughter resource D#2 (that D#2 has condition QV#3), which is the basis for the value of this logicalMeasurement. Here we have a nested context, because the condition property itself has its own context and statement about the D#2 condition, which is true in context of the statement about the D#2 physical measurement. Such nesting can be performed until reaching the atomic statement, which has its property with undefined context.

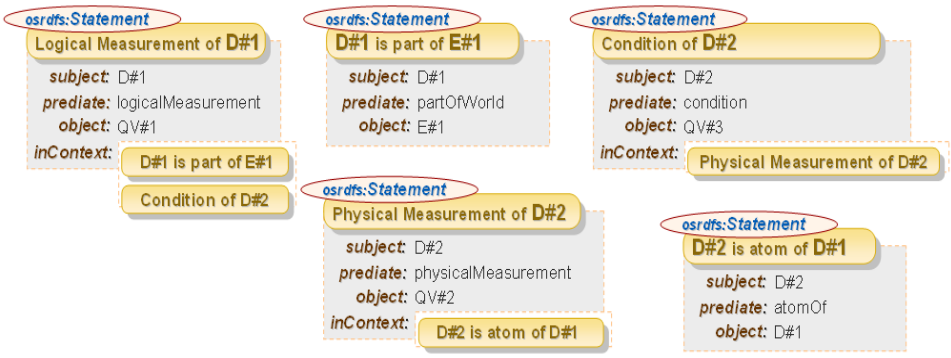


FIGURE 32 Context-sensitive description

3.1.2 Context probabilistic model

As was mentioned earlier, each statement is in a certain context, i.e., in a set of contextual statements. The number of contexts (contextual containers) is not limited by any one context. A statement can have a set of contexts, but we can not expect that each of these contexts makes it 100% true. It seems reasonable to define a probability of a statement to be true in each possible context for this statement. With this aim, we have extended the number of properties of the osrdfs:Container and osrdfs:OCC_Container classes with the *osrdfs:contextProbability* property (FIGURE 33). Now we can specify a probability value (between 0 and 1) for each contextual container of the subject statement. This gives us a possibility to build a probabilistic model on a top of this and to enable probabilistic reasoning based on it. However, even after extending a context-dependent resource description with a probability value, we still need one more element. Since the "significances" (relevancies) of the contextual properties might differ from one other and the significance of the property depends on a certain context, we have to model these as well. And it will be possible to define the probabilistic significance of contextual properties via utilization of the same OSRDF approach.

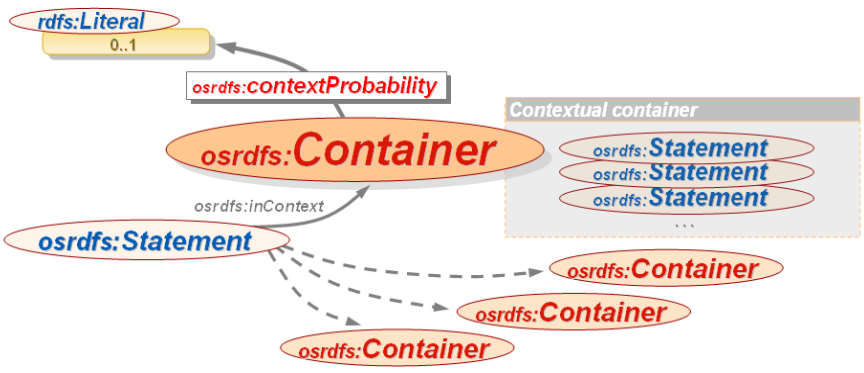


FIGURE 33 Probability of statement context

First we should define the *osrdfs:PropertySignificance* class. An instance of this class sets the significance of the subject property via the *osrdfs:subjectProperty* (refers to the subject property) and *osrdfs:pSignificance* (with a value between 0 and 1). Then it is necessary to define a property to be able to specify the significance of the subject property context. The *osrdfs:significanceOfContext* points to an *osrdfs:PropSignContainer* class instance – a container of osrdfs:PropertySignificance instances (contextual properties with correspondent significances for the subject property [osrdfs:Property instance]). And finally we can create a statement that defines the significance of the contextual properties dependent upon a certain context (FIGURE 34).

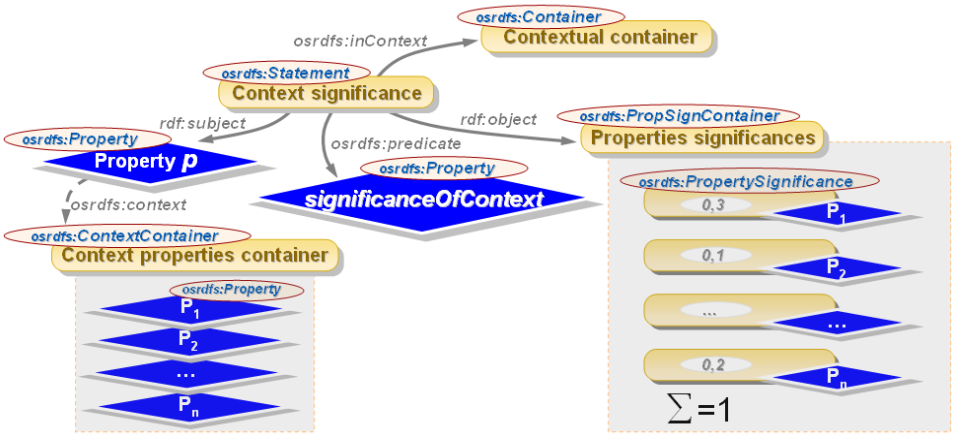


FIGURE 34 Context dependent significance of contextual properties

In many cases probabilistic models for contextual reasoning might not be enough and should be enhanced by fuzzy models. From that point of view there is still room for improving the context description part of OSRDF based on related work in Fuzzy Description Logic. Sicilia and García-Barriocanal (Sicilia and García-Barriocanal, 2004) whose work is based on the earlier research of Straccia (Straccia, 2001), propose Fuzzy Description Logics (fDLs) as an extension to conventional Description Logics (DLs) (Tresp and Molitor, 1998) to handle uncertainty and imprecision in a numerical way. DLs, as a logical reconstruction of frame-based representation languages aim to provide simple declarative semantics to capture the meaning of the features of structured knowledge representation. A pragmatic way is being sought here to extend current tools and interfaces for DL–based applications with a fuzzy processing layer to deal with some specific scenarios of uncertainty handling associated to ontologies.

3.2 Resource Proactivity Description

Modeling of multi-agent systems and the behaviour of concrete agents in them have been significant topics in various domains. The model-driven approach to

the design of agent behaviours emerged quite some time ago and initially was based on UML modeling (Chella et al., 2004, Torres da Silva et al., 2004). Later this approach was extended to a level of meta-modeling (Djuric et al., 2004). As one of the mature UML-based methodologies for modeling multi-agent systems, Agent Modeling Language (AML) deserves a special mention (Cervenka et al., 2005). Currently, AML is used in commercial software projects, is supported by CASE tools, and, in the near future, the first version of its specification will be presented to the public for its further development. One of the fundamental formal theories about behaviour in multi-agent systems (Dastani et al., 2004a) has been developed and lectured at the Free University of Amsterdam[28].

In behaviour annotation of a resource (agent) and the resource proactivity performance stages, we face two challenges. First, we need a handy and intelligent user interface for resource behaviour (rules) and the resource mental states specification. And second, we need an engine to run these rules and to perform the actions.

Rather than making life complex, applying new technologies should make life (business processes, etc.) easy, flexible and scalable. These new technologies should be simple and attractive for users intending to utilize such multi-agent systems.

Based on such platform of interactive, proactive resources, many different processes can be modeled. They can be both complex processes (business processes, enterprise integration, distributed maintenance, distributed diagnostic and learning, supply chain management, etc.) and more primitive ones (personal agents' interaction, home devices' interaction, etc.). The modeling of each process demands specific domain knowledge from the system user - beginning from the expert of a big corporation and ending with a housewife. But in both cases the user interface (the entire module for interaction with the user) should provide a handy and intelligent functionality for the system.

3.2.1 Rule Representation Model

Another part of OSRDF, which is oriented on rule, resource goal and behaviour description, was elaborated during the second project year as Resource Goal and Behaviour Description Framework (RG/BDF). In continuation to the idea of Context-sensitive Description Framework (CDF) (Khriyenko and Terziyan, 2006), such approach has been applied to the context sensitive resource goal and behaviour description. RG/BDFS-Lite is an upper schema for the description of resource goal and behaviour. It is based on the CDF schema and extends it together with Resource State and Condition Description Framework Schema (RS/CDFS). One of the main features of the CDF is its ability to describe context-dependent facts (fact-statements) about resources. At the same

[28] http://www.vu.nl/

time RG/BDF brings a new (additional) vision to resource description. It is a description of a resource mental state. If we consider an agent (software agent) as a resource in GUN, then we have to consider and model its believes, desires, intentions, etc. Now we can model not just the statements that describe the facts, but also goals-statements that describe a desirable state of environment from the viewpoint of a resource (agent), other resources' states, etc. (Kaykova et al., 2005c) provide more information regarding RG/BDF and give examples of a CDF concept implementation.

The main challenge in goal and behaviour description is how to deal with the different semantic meanings of a statement. In RDF and in the CDF extension we had to deal with a resource, its state and condition description. In other words, we described some facts about subject resource. But, now, when we need to describe rules via a probable resource condition and goals description, we face the necessity to somehow represent a statement which is not a fact. That is, a goal is not a statement of a fact, it is a statement that a resource wants to achieve.

Fact Statement

The *Fact Statement* describes the facts of the entire system (environment), including the states of the resources and their sub-histories. The osrdfs:Statement (enhanced with a context extension) fits this purpose very well. But to distinguish fact statements from unsubstantial (non-fact) statements we extend the set of OSRDF Statements with the *osrdfs:F_Statement* class (a subclass of the osrdfs:Statement class). To clarify this, an example is presented. Device #1 has some state description (its parameters' values) at a certain time (time of the Environment) (see FIGURE 35).

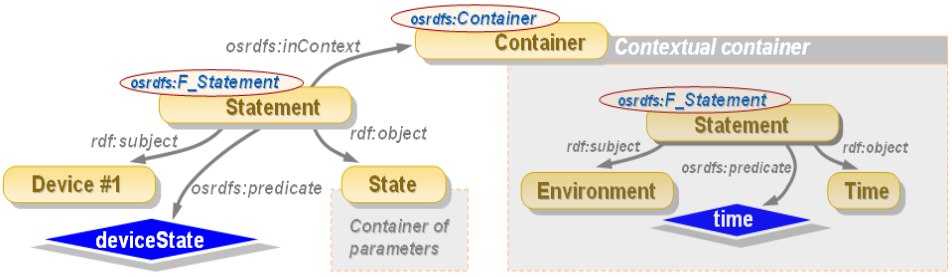


FIGURE 35 Fact Statement

Non-Fact Statement (Mental Statement)

The *Non-Fact Statement* allows the possibility to describe not just a history as facts, but also to describe the mental states of a resource (according to BDI [Beliefs-Desires-Intensions] Model). To describe a semantically new statement, which is an unsubstantial statement, we define the *osrdfs:NF_Statement* class (a subclass of the osrdfs:Statement class). Here a triple <SSS-PPP-OOO> describes some statement which is not a fact and absent in the history. As a subclass of osrdfs:NF_Statement we have defined *osrdfs:GoalStatement* - a class of the

resource (agent) goal instances, which should be achieved by the agent (see FIGURE 36). This statement also can be described with a certain context.

Such statement becomes TRUE just when the same fact-statement (osrdfs:F_Statement) appears in the resource (or environment) history. Otherwise it takes on the value of FALSE. We also have to define a container for mental statements – *osrdfs:NF_Container* as a subclass of the osrdfs:Container class.

FIGURE 36 Goal Statement

FIGURE 37 shows one of the NF_Statements of a SmartResource (its agent). The goal is to have sent a diagnostic request as a result of a request sending action. The statement object is not defined, because, in this particular case, it does not matter what the diagnostic request is (it can be any diagnostic request).

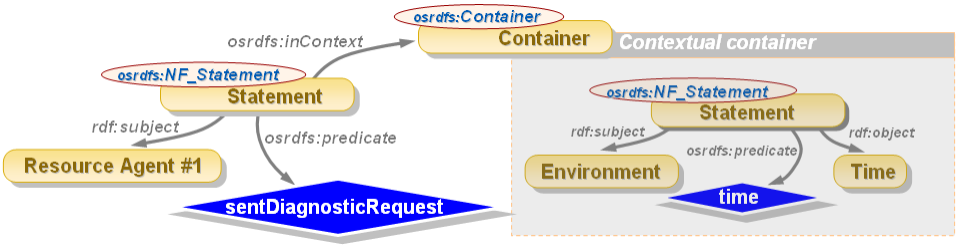


FIGURE 37 Non-Fact (Mental) Statement

Now we need to represent the container of goal statements, which define the goals. For this purpose we have defined *osrdfs:GoalContainer* as a class of the goal container instances. Such container plays a role of context (via the *osrdfs:desire* property) for the behaviour statement until the goal is achieved, and that is why it is a direct subclass of osrdfs:NF_Container.

As was mentioned previously, a goal may be divided to a set of sub goals. Thus the goal container also plays the role of a goal set, the members of which are subgoals of a complex goal. To define a set of subgoals for a complex goal there is the *osrdfs:subGoal* property (see FIGURE 38). The domain and range for this property are the osrdfs:GoalStatement and osrdfs:GoalContainer classes respectively.

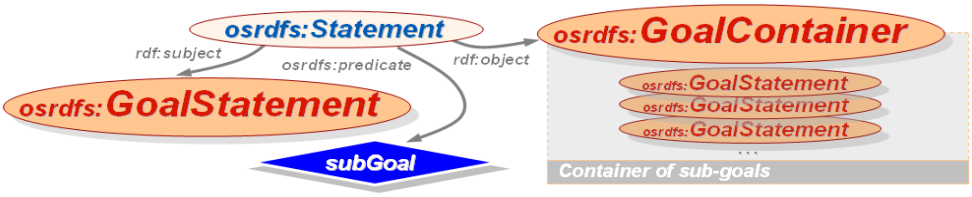


FIGURE 38 OSRDF Goal

Rule Statement

Another challenge for OSRDF is rule description. *Rule Statement* can be referred to as a Non-Fact Statements, as a statement allows one to describe the rules of environment modification (changes in the resources' states and descriptions). With this statement we can define a statement's truth dependence on other statements (the Non-Fact Statements), which dictate the necessary and sufficient conditions. Thus, the Rule Engine, which follows these rules, can modify (add, delete, etc.) the content of the History (which contains the Fact Statements), the Non-Fact Data (which contains the Non-Fact Statements) Storages, and Rule/Behaviour sets (see also Smart Resource Platform Architecture) in the operational memory of the engines. *osrdfs:RuleStatement* is a class of rule instances, and a subclass of the osrdfs:NF_Statement. Rule description brings the slightly different semantics of the fourth part of the OSRDF quadruple context for rule statement definition. The osrdfs:inContext property allows describing a context in which a subject statement has occurred. Now, however, when we describe a rule, we should describe the condition for the rule performance through the context of the rule statement. Thus, the context plays a role of a condition for rule performance with "IF->THEN" or "IF NOT->THEN" meaning. In order to describe rule context, two additional properties have been defined. The *osrdfs:trueIf* and *osrdfs:falseIf* properties are instances of the rdf:Property and sub-properties of the osrdfs:inContext property. A rule statement will be true if the statements, which are contained in osrdfs:Container via the osrdfs:trueIf property are true, and the statements, which are contained in osrdfs:Container via the osrdfs:falseIf property are false (see FIGURE 39). In case of resource behaviour description, such properties play the role of a trigger, which switches on and off the behaviour performance (the performance of a behaviour, which is described via the behaviour container).

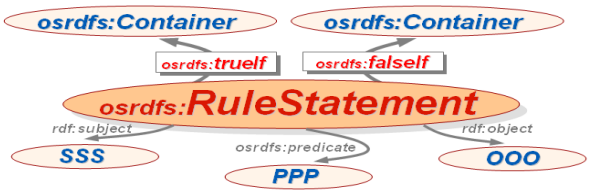


FIGURE 39 Rule Statement

With the OSRDF Rule Statement it is possible to describe a rule in an "AND-OR-NOT" notation that can be easily presented in an "AND-NOT" notation (see FIGURE 40). Here the osrdfs:trueIf property plays the role of the logic "AND" operation and the osrdfs:falseIf the "AND-NOT" operation. The NF_Container (connected via the osrdfs:trueIf property) is a collection of productions combined via the "AND" operation, while the NF_Container (connected via the osrdfs:falseIf property) is a collection of productions combined via the "OR" operation. In the event of multiple usage of the osrdfs:trueIf and osrdfs:falseIf properties, they are combined via the "OR" operation (see FIGURE 40).

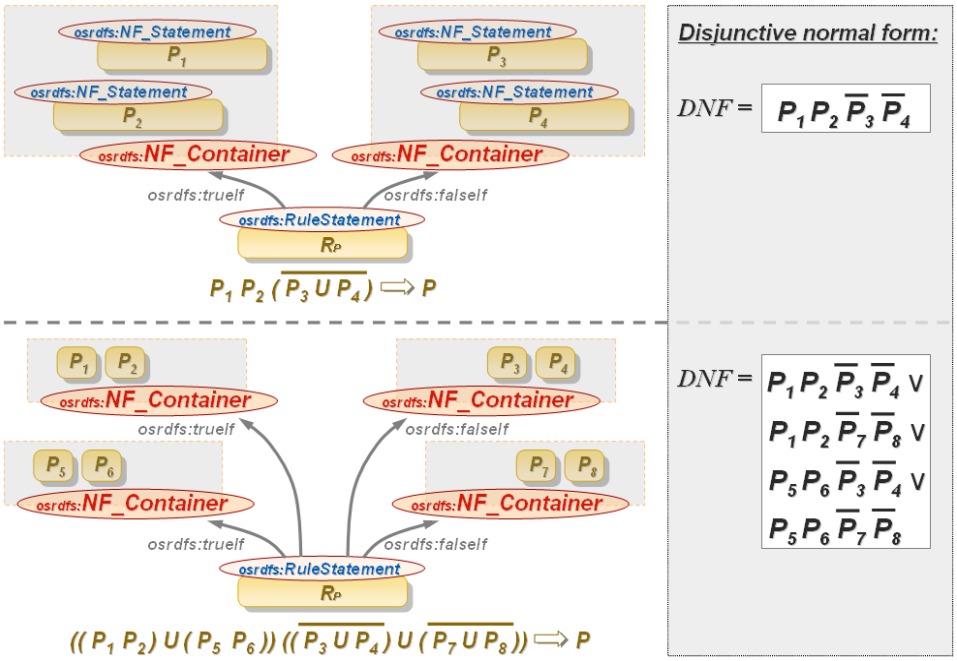


FIGURE 40 Rule Logic Framework

Sometimes, a device can play not just the role of Device (the object of diagnostics), but also that of a diagnostic service. During a long performance period, the device can collect diagnoses of its states, and in future provide diagnostics based on this labeled history. One of Rule Statements can be, for example, a rule that ResourceAgent should play a role of the diagnostic service if it has received a diagnostic request, has played the role of Device, and has not played a role of the diagnostic Service already (FIGURE 41). If the preconditions will state that this Rule Statement is TRUE, then the correspondent Fact Statement (that the ResourceAgent plays a role of the diagnostic Service) will be added to the Environmental Storage.

Such a Rule Statement can be used for a meta-rules definition as well. We can describe the state of a rule via the *osrdfs:ruleConditionIs* property, where the values for the rule state are restricted by the osrdfs:Active and osrdfs:Passive

values (instances of the *osrdfs:RuleCondition* class). Thus, the NF Statement (which defines the state of a rule) can be activated/deactivated via the Rule Statement (FIGURE 42) and can play the role of a sufficient condition for the subject rule. The meta-rule definition provides the possibility to define the context for the rules and behaviours.

The method of meta-rule execution constantly depends on the Engine realization and can be done in different ways. When all the rules are located in the Engine Operational Rule Memory (Storage) and contain NF Statements about their states (active or passive) as a sufficient condition, the Operational Rule Memory will have a huge number of Rule Statements bringing a decrease in the rule engine performance. When the platform has another rule engine for meta-rules tracking, all the rules can be stored in some Rule Storage and will be added to the correspondent Engine Operational Rule Memory just when they are active (and removed from it otherwise).

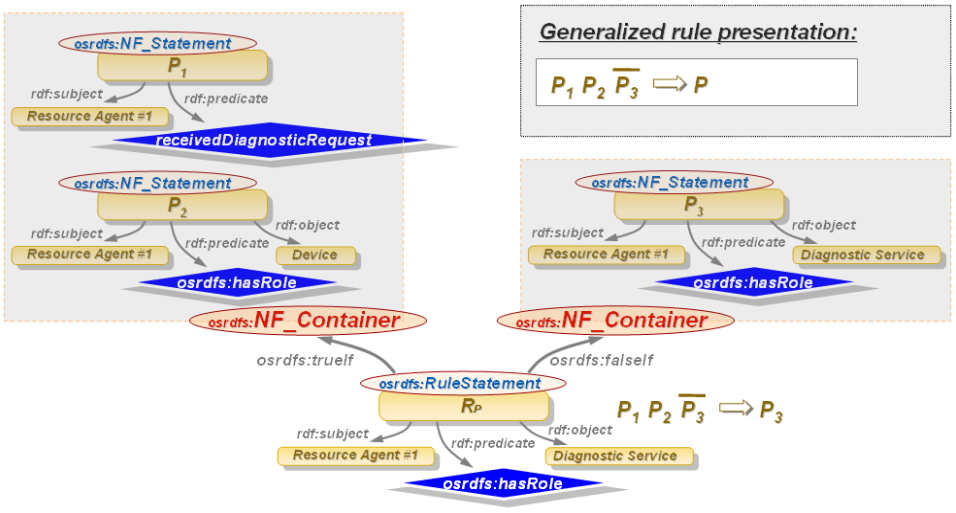


FIGURE 41 OSRDF Rule

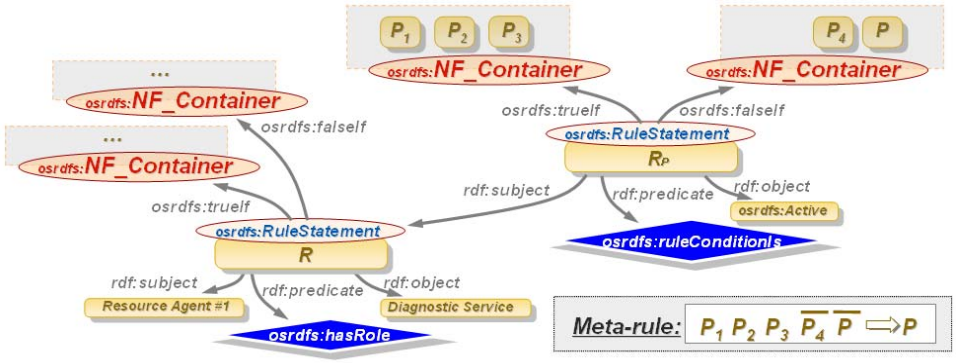


FIGURE 42 Meta-Rule definition

Behaviour Statement

The *Behaviour Statement* is also a statement for rule definition. It describes a rule of behaviour performance (fragmentation to more simple behaviours via the *osrdfs:hasBehaviour* property or the performance of concrete executable modules/action via the *osrdfs:execution* property). In case of a goal driven behaviour rule description, we have defined the *osrdfs:BehaviourStatement* class as a direct subclass of osrdfs:RuleStatement with extended properties. The *osrdfs:ResourceAgent* class plays the role of the subject *(osrdfs:bSubject)* range. The range of the statement's predicate (*osrdfs:bPredicate*) is restricted by the *osrdfs:B_Property* class (a subclass of the osrdfs:Property class). An object of the behaviour statement can be represented by the *osrdfs:BehaviourContainer* container of nested behaviour statements (if root behaviour is complex) or atomic execution (an instance of the *osrdfs:Execution* class). Also we have defined the *osrdfs:desire* property (a sub-property of osrdfs:falseIf). It creates a link to the goal container, which contains the goal statement(s) (because behaving has a certain sense when a goal is not achieved). If the presence of a Goal is a necessary condition for the behaviour, then context statements (condition of the environment) is a sufficient condition (which is represented by a contextual container via the osrdfs:trueIf property) (see FIGURE 43).

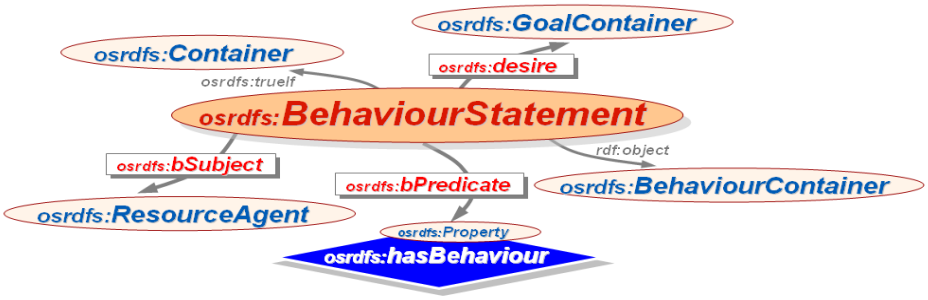


FIGURE 43 Behaviour Statement

In analogy with the Rule Statement, activation of the Behaviour Statement depends on the Environmental and Resource Mental (Non-Fact) States. Let us consider a behaviour (FIGURE 44), which is aimed at sending a diagnostic request after the device encounters an alarm situation and has received an alarm statement. This circumstance results in the activation of some set of sub-behaviours or in the execution of the correspondent executable module. Here the statement, "device has an alarm", plays the role of a sufficient condition for the behaviour performance (via the *osrdfs:trueIf* property, as part of the IF-THEN rule). Meanwhile, the statement, which indicates that "ResourceAgent has sent a diagnostic request", plays the role of a necessary condition (via the *osrdfs:desire* property, which is a part of the IF NOT-THEN rule). Again, the rule engine (Behaviour Rule Engine), which follows the rules and performs the actions, can modify (add, delete and etc.) the content of the Rule sets; Environmental and Resource Mental (Non-Fact) States Storages (see FIGURE

46). As a result of the behaviour (described in FIGURE 44), the goal will be reached, and a Fact Statement, which states that that "ResourceAgent has sent a diagnostic request", will be added to the Environmental Storage.

osrdfs:BehaviourContainer is a class of the behaviour container instances. The main role of a behaviour container is to collect the nested behaviours for a complex behaviour (represented by a behaviour statement) (see FIGURE 45).

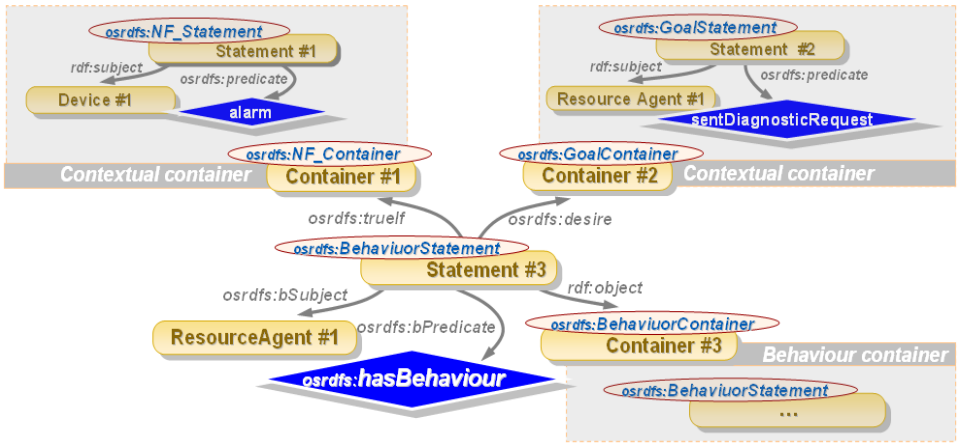


FIGURE 44 OSRDF Behaviour

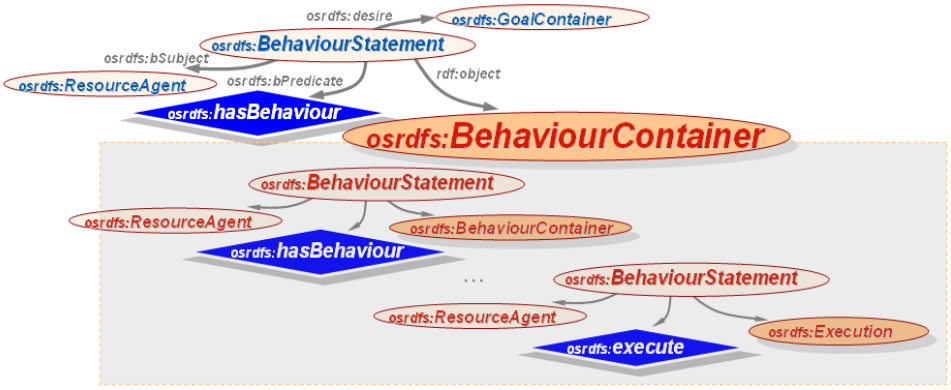


FIGURE 45 OSRDF Complex/Nested Behaviour

Simple behaviour that means the performance of a certain action (execution of certain method, code…) can be represented by an instance of the *osrdfs:Execution* class referred to via the *osrdfs:execute* property (instance of the osrdfs:B_Property class) (see FIGURE 43). An instance of the osrdfs:Execution class describes the exact method (code, service and etc.), inputs, outputs and other features of the execution entry. To define a complex behaviour (which means performing a set of nested behaviours) for an agent, OSRDF-Schema has

the *osrdfs:hasBehaviour* property (an instance of the osrdfs:B_Property class). This property defines a set of behaviour statements via a behaviour container for the resource agent (see FIGURE 45).

Another important part of behaviour structuring is the agent role (see FIGURE 46). The *osrdfs:hasRole* property defines a role (*osrdfs:Role*) for the resource agent in a certain context. Another property that is related to the agent role is *osrdfs:goals*, which defines a goal or a set of the goals, corresponding to the subject role, via a goal container. As it was mentioned previously, a resource agent may have different roles and a set of the goals can be different even for the same role. To allow the definition of a context for them, these two properties are instances of the osrdfs:Property class.

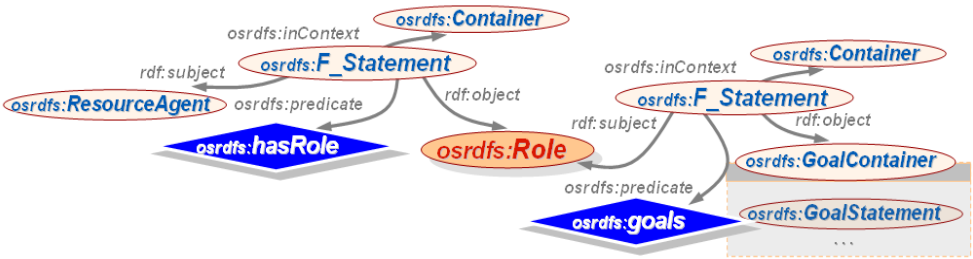


FIGURE 46 RGBDF Role

Regarding the process coordination approach (considered in the previous chapter), the upper-process Agent should provide additional behaviour-rules (meta-rules) with the necessary RG/BDF Executions (atomic executable modules - Actions) that perform a behaviour-rules condition switching. It makes sense to define *osrdfs:RuleConditionSetter* as a subclass of osrdfs:Execution and supply this class with two properties: *osrdfs:subjectRule* and *osrdfs:subjectRuleCondition* (see Appendix A). Thus, an Action (atomic executable module), that changes the rule condition, gets as an input a certain instance of the osrdfs:RuleConditionSetter class and references to the subject rule (its condition should be set) and the condition value itself. As a result of such an Action performance, the correspondent Fact Statement about rule condition will be added to the Active Data Space.

The RG/BDF behaviour-rule description approach fits very well the constraints definition by adding a restriction behaviour statement. FIGURE 47 shows an example of the RG/BDF representation of behaviour-rule R_3^1 before and after the constraints adding.

The presented Resource Goal Behaviour Description part of OSRDF is fully compliant with the BDI (Belief-Desire-Intention) model that is well-known in the scientific world of Multi-Agent Systems (Rao and Georgeff, 1995). By now, quite a few research results is available in the domain of agents based on the BDI model.

Recent results report significant development in modeling frameworks for BDI agents (Mascardi et al., 2004) and different logic programming languages,

such as AgentSpeak (Rao, 1996). The BDI model has been actively developed towards cooperative behaviour of agents (Ancona and Mascardi, 2004), particularly aiming it at exchanging executive plans (Ancona et al., 2004, Ancona and Mascardi, 2004). FIGURE 48 explains the parallel between the BDI and RG/BDF models.

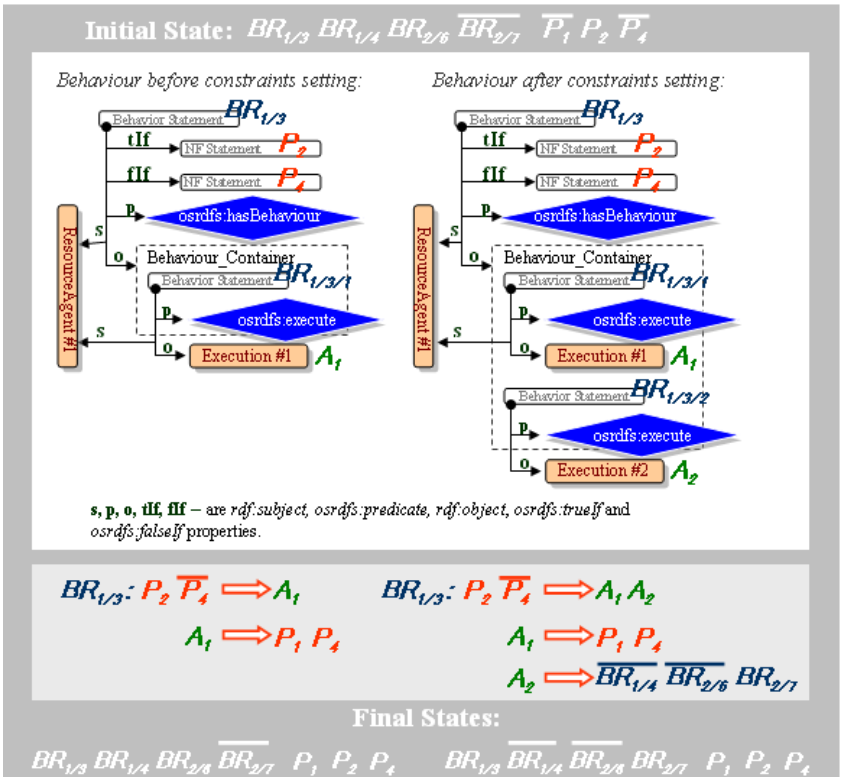


FIGURE 47 An example of RG/BDF representation of behaviour-rule

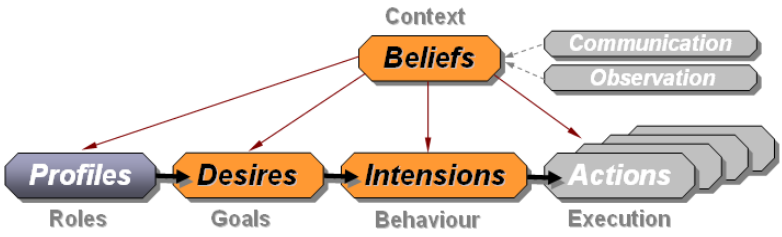


FIGURE 48 BDI: Underlying Model for OSRDF

FIGURE 49 presents the architecture of the proactive layer of the Smart Resource Platform. The structure contains four storages: the Environmental (containing the Fact Statements) one, the Resource Non-Fact States (containing the Non-Fact Statements and Rule Statements) Storages, a storage where

ontology and all the instances (Resources such as Devices, Services, Human Experts, Agents, etc.) are located, and a storage of the programmable executable modules. In reality, the storage of the Fact Statements is presented by two storages: Operational Memory and Long Term History. The Operational Memory contains updated information relevant to the performance data. For example, if a statement that the ResourceAgent plays some new role goes to the Operational History, then the statement about the previous role should be removed to the Long Term History, or the irrelevant alarm statements should be removed. Such a filter should not allow a contradiction occur within the operational data.

There are also two engines, the Rule Engine and the Behaviour Engine, which iteratively check the rules, perform them and run actors (modules). As shown in the example of the Rule Statement description, the Rule Engine primarily generates (changes) the context for resource behaviour. The outcome of the Rule Engine is an update of the history data that can affect the rules' performances, and an update of a rule set in the Operational (Working) Memories of the engines. The outcome of the Behaviour Engine is additionally extended by the running of the executable modules or web services.

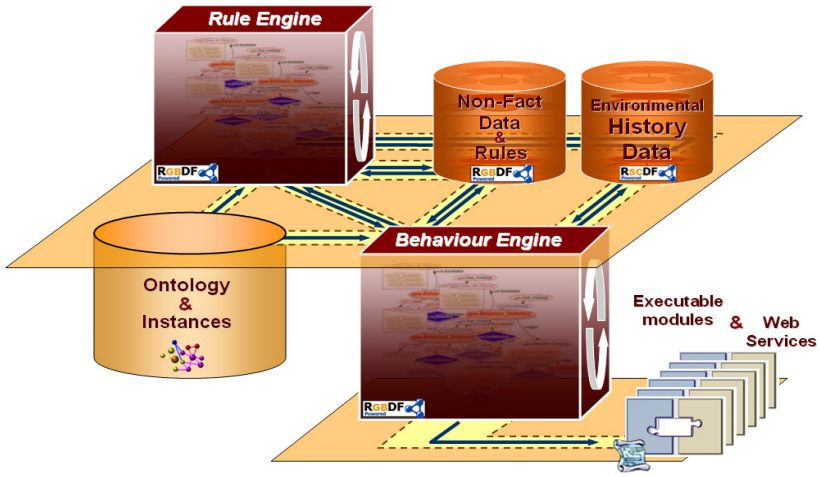


FIGURE 49 The Proactivity Layer architecture

If we try to generalize this approach, we can see that even the mediation platform, which plays the role of host for the resources (devices, services, etc.), can be considered a smart resource with its own proactivity layer. By utilizing such a centralized approach, we can create alliances of resources with internal "rules of the game". On the other hand, cooperation between resources can be realized via P2P connections. But in both of these architectures of resources, the interaction agents, i.e., Resource Agents, can access each other, retrieve the necessary data, and have an influence on the behaviours of other resources. However, everything depends on the security access permissions between the resources.

From the system usability point of view, all of these complex models of element interaction and the functionality of the engines should be hidden from the end-user via an interface that is handy and intelligent, i.e., SmartInterface. The main information that a user should specify during a ResourceAgent setting is the goal (a Goal Statement that describes the ResourceAgent's aim). At the same time, the user should specify the input data (not necessarily any existing fact from the History [if there are no statements yet], but a template – a statement without an object). During all of these manipulations, the interface should provide the user all the available information from the ontology and data, which is stored on the platform: a list of instances, a list of intellectually filtered properties, etc. Thus, if there are semantic profiles of accessible executable modules and web services (with semantically annotated inputs and outputs), then the "behaviour modeling module" on the platform will generate the behaviour rules automatically (and will try to build a process execution). Otherwise, we will need to specify the semantic profile for all of the available executable modules on the platform and for the web services that will be used. If there is no executable module or web service which can exactly satisfy the goal, then the goal can be divided to a set of sub-goals based on the correspondent information in the ontology or in the iterative process of automatic sub-goals generation (i.e., the required inputs for modules that can reach the goal, but where the necessary inputs are not provided). Thus, the goal will be reached by using a set of interactive executable modules. Again, if such a platform applies for a new infrastructure, it creates the need to define not just the rules of the Agent behaviours, but also the rules of the Environment (whole system) influence (the rules of behaviour context). These are the actions the user should execute in the worst case, when he/she adapts the platform for a specific purpose (specific domain). But there is also an easier way to configure the ResourceAgent, if it used for a widespread, widely used, and known process. It is based on the Agent Role specification only, and implies that the ontology contains all the relations between the agent roles and the goals with the corresponding behaviour rules. But, as you can see, both of these ways (and especially the second one) assume that a lot of hard work has been done by the ontology engineers beforehand, and that the ontology contains enough information (knowledge) to allow the SmartInterface to demand as little as possible from the user.

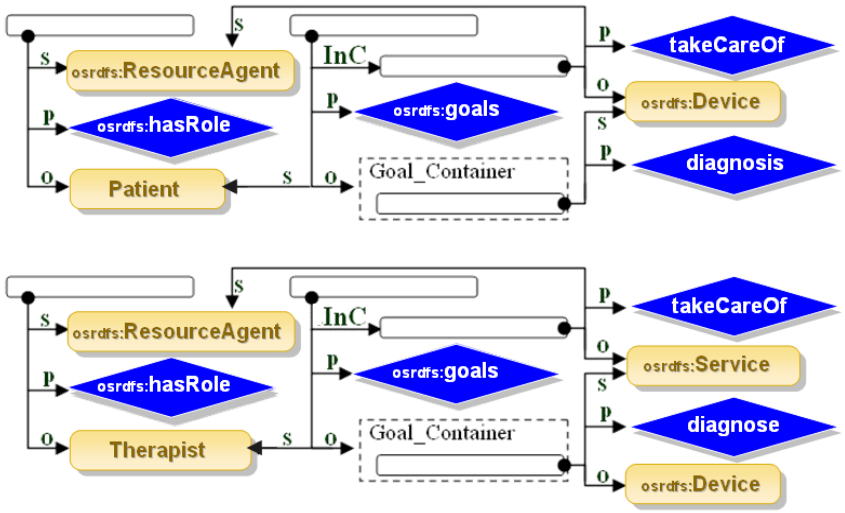
3.2.2 Agent Behaviour Case

Let us consider the case of agent behaviour. It is a simple case of device diagnostic by some web service. Actually we have two smart resources – these are resources (field device and web service) with the agents which take care of them.

An agent, who represents a field device, plays a role of a patient that wants to take care (know its condition/diagnosis) of itself in case if certain

alarm occurs. Thus, the goal of this agent is to get a diagnosis statement from the diagnostic unit (in our case the diagnostic web service) based on the sub-history of device states if an alarm statement appears. It is a complex goal and contains nested subgoals. The agent should send a diagnostic request to the web service: initially. this requires collecting the set of device states and retrieving the appropriate web service to get the corresponding response with a diagnosis statement from the service. On the other hand, we have a web service agent, which plays a role of a therapist (diagnostic unit). The goal of this agent is to base the diagnosis on device state sub-histories. This is a complex goal, which assumes receiving a diagnostic request, making a diagnosis and sending a response back to the field device agent.

As was mentioned before, the ontology contains the templates of roles, goals and behaviours. FIGURE 50 represents two role templates and the corresponding goals templates.



s, p, o, InC – are *rdf:subject*, *osrdfs:predicate*, *rdf:object* and *osrdfs:inContext* properties.

FIGURE 50 Role and correspondent Goal templates

Later on we will concentrate on the example of the first agent, which takes care of a field device. As you can see, the agent, in a role of a patient, has its goal set for getting a diagnostic statement about a certain device, the context being that the agent takes care of this device. But this goal is not atomic: it has a set of nested sub goals. The next figure shows how nested subgoals can be described in an ontology (FIGURE 51). In a similar way, the template of agent behaviour can be described via osrdfs:BehaviourStatement. Such statement ties a certain execution statement (which defines the execution module for a certain action through the rdf:object property via osrdfs:BehaviourContainer) to a goal (described through the osrdfs:desire property) and a context (which specifies condition when the action should be performed through the osrdfs:trueIf property). It can happen that performance gets stuck without some statement. Then, this statement automatically becomes a goal statement (sub-goal) for the

resource to continue performance and achieve the final goal, and appropriate behavior can be retrieved from ontology.

Let's consider the process of goal and behaviour specification for an agent. At the initial stage an expert managing the connection of the agent to a resource (field device) should specify, from the ontology, a certain role or a goal or even a set of them for the current agent. If that certain role is specified, then a set of correspondent goals or a single goal may be retrieved automatically from the ontology (from a set of goals, correspondent to specified role). Then the appropriate behaviour templates, corresponding to the agent goals, can also be retrieved from the ontology. After all the necessary templates are collected the correspondent instances of them (with a linking to the concrete instances of the resource agent, resources and etc.) should be placed to the agent storage on the resource platform. Depending on the complexity of the goals, a nested hierarchy of the agent behaviour rules will be composed automatically by the engine of the agent shell (see the example in FIGURE 52).

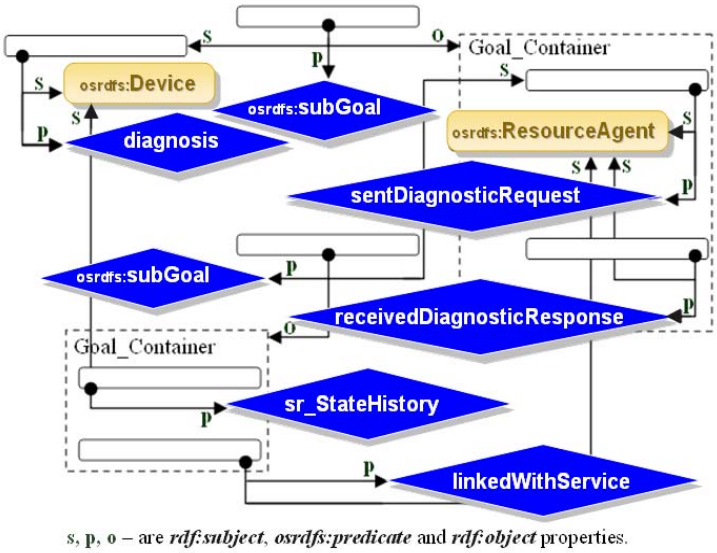
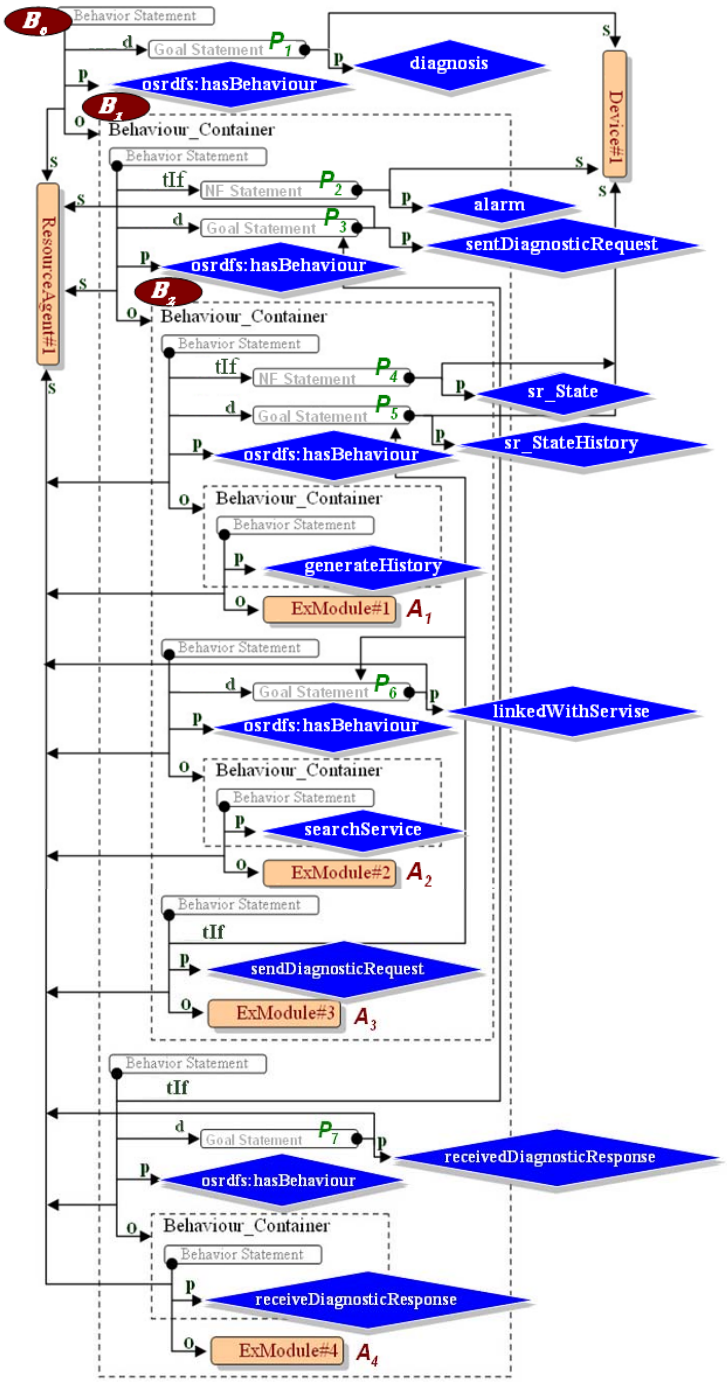


FIGURE 51 Nested Goal representation

Now, when the rules of the agent behaviour are specified, it is the time to run the agent engine to behave. The working space (storage) of the SmartResource Platform (a combination of a device and the agent which takes care of it) should be divided into two parts, i.e., a temporal storage and a long term storage. Initially, all information (all statements that concern the resources' states and conditions) is saved to the temporal storage and plays the role of the behaviour context and input data for execution modules. As was mentioned before, the goal statements (referred to via the osrdfs:desire property) play the role of a trigger to run a certain behaviour rule. If there is no statement in the temporal storage similar to the goal statement, then the agent engine performs a rule.



s,p,o,tIf,d – are *rdf:subject*, *osrdfs:predicate*, *rdf:object*, *osrdfs:trueIf* and *osrdfs:desire* properties.

FIGURE 52 Nested hierarchy of agent behaviour rules

Let's consider our example. The agent exhibits root behaviour as long as the statement about device diagnosis does not exist in the temporal storage. Otherwise the agent engine skips the behaviour rule and goes to the next one on the same level of nesting. Each level may contain both types of behaviour statements: complex behaviour statements and atomic execution statements (which specifies an execution module via an instance of the osrdfs:Execution class). Actually these execution modules generate (add to the temporal storage statement), which is required by the goal of the behaviour rule. For example, execution module, which is described by the instance *ap:ExModule#3,* generates a statement which states that an agent (ap:ResourceAgent#1) has sent a diagnostic request to a certain diagnostic service. Thus, the agent has achieved one of the subgoals. But two of the subgoals (i.e., collecting a history of the device states and retrieving a suitable service for diagnostic) had been achieved before, because they were needed for performing execution module #3. Once the agent has achieved the upper goal, the statement of the achieved goal is removed to the long term storage and kept in the history of the SmartResource. At the same time all the statements of the achieved nested goals should be removed also. Some of the contextual statements having played the role of input data should also be removed (for example the statement about an alarm, which plays the role of a context for sending a diagnostic request, and the statements about device states, which were used for diagnostics).

3.2.3 Resource Agent Behaviour (Rule) Engine performance

Let us consider our previously described example and try to follow a platform execution. The case is a simple device diagnostic performed by a web service. In actuality, we have two smart conventional resources (a field device and a web service) supplied with the agents that maintain them.

The agent, which represents a field device, plays the role of a patient that takes some measures towards self-care (will know its own condition/diagnosis) when a certain alarm takes place. Thus, the goal of this agent is to get a statement about a diagnosis from a diagnostic unit (in our case a diagnostic web service) based on the sub-history of device states. It is a complex goal and contains nested subgoals. The agent has to send a diagnostic request to the web service, which requires, initially, collecting of a set of the device states and a search of the appropriate web service. After the request has been sent, the agent must get a corresponding response with a statement about the diagnosis from the web service. On the other hand, we may have a web service agent that plays the role of a therapist (diagnostic unit). The goal of this agent is to diagnose the condition based on the sub-histories of the device states. It is a complex goal, which assumes receiving a diagnostic request, diagnosing the problem, and sending a response back to the field device agent. The nested hierarchy of the agent behavioural rules is presented in FIGURE 52.

These rules can be formalized via *productions* (Production System). Production systems are very useful tools for modeling behaviour. They can model cognitive processes such as reasoning, but the way they do this is generally regarded to be different from the way reasoning occurs in humans. That is not to say that these types of models are not widely used today. It is agreed, however, that the method underlying these models is different from the processes of the brain or mind. Production systems have three main components (Turner and Roeck, 1988): a Rule Base, a Working Memory, and an Inference Engine or Interpreter.

Let us define all of the Non-Fact Statements as predicates [P_1, P_2, P_3, P_4, P_5, P_6, and P_7], the sets of sub-behaviours via the rule collections [B_0, B_2, B_3], and the executable modules via the actions [A_1, A_2, A_3, A_4] (see FIGURE 52).

P_1 – Device has a diagnosis (state about diagnosis);

P_2 – Device has some alarm statement;

P_3 – ResourceAgent has sent a diagnostic request;

P_4 – Device has at least one state description;

P_5 – Devise has formed a sub-history of the states;

P_6 – ResourceAgent has linked with an appropriate web service;

P_7 – ResourceAgent has received a diagnostic response.

Now we can define a simple Production System as a set of the rules according to our example (see FIGURE 52). Sub-behaviours are presented by a subset of the rules and are performed by an engine as a separate thread. FIGURE 53 shows the Production System of a ResourceAgent (which represents a field device in the device diagnostics case) behaviour.

Actions A_1, A_2, A_3, A_4 (FIGURE 53) are internal actions, which are performed by the subject ResourceAgent and affect the Fact Statements (through their appearance in the History Storage), which in their turn affects the Non-Fact Statements' truth. At the same time, we have actions A_N *and* A_K as external actions. They are actions of other ResourceAgents, which also affect the Non-Fact Statements through operations with the Fact Statements.

A_1 – A history is generated based on the existent device states. As a result, a new Fact Statement that the "Device has a sub-history of the states" will be generated and located in the Operational Memory. It makes the correspondent Non-Fact Statement TRUE (P_5 is TRUE). At the same time it removes statements regarding the device states (which are collected within the history) from the Operational Memory (then P_4 is FALSE);

A_2 – A search of the relevant diagnostic Service. At the end of the action, the Fact Statement that the "ResourceAgent is linked with a diagnostic Service" will be added to the Operational Memory. It makes P_6 TRUE;

A_3 – A diagnostic request is sending an action. This action adds the Fact Statement that the "ResourceAgent has sent a diagnostic request" (it makes P_3 TRUE), removes the Statement that the "Device has a sub-history" and the Statement about an alarm state from the Operational Memory, because the sent request has operated on these issues (it makes P_5 and P_2 - FALSE). Finally, this action removes the statement about the linking to the Service (P_6 is FALSE, and it can be an irrelevant link for the next aim);

A_4 – Receiving of a diagnostic response. It adds a Fact Statement about the Device diagnosis and makes P_1 TRUE;

A_N – An external action, which is performed by another ResourceAgent, adds a new Fact Statement about the alarm situation to the Operational Memory (that makes P_2 TRUE);

A_K – An external action, which generates new Fact Statements about the Device states. These statements are located in the Operational Memory and it makes P_4 TRUE.

In some cases, it is better to utilize other execution engines that are perhaps better suited or more used for a specific domain. For example, in the case of the process performed by the web services, it makes sense to use BPEL Workflow Engine (BPEL, 2004). In this case we need to enhance the Platform with a transformation module that transforms the OSRDF behaviour description to a BPEL description of the process. The advantage of the OSRDF process representation is that the web services can be described through semantic profiles instead of by an exact web service description. Thus, we take a step from individual WS binding to Semantic Scenarios Specification. It allows us to select the suitable web services from the available set, and then to make a transformation to the BPEL scenario. Such an approach brings the possibility to share and utilize knowledge about the process without depending on the available services (FIGURE 54).

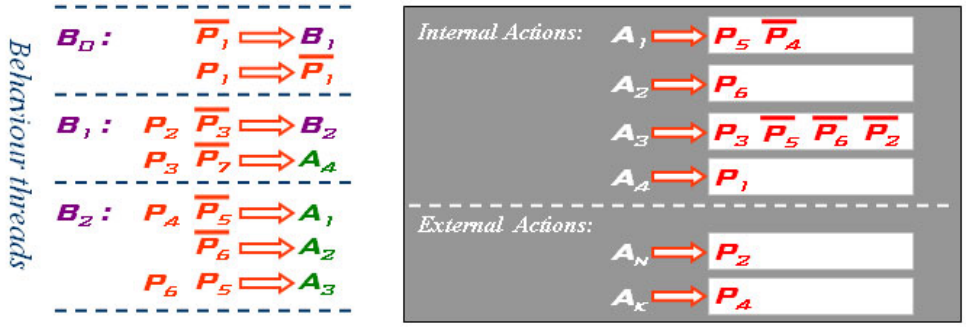


FIGURE 53 Production System of ResourceAgent behaviour

Domain ontology provides a common shared understanding of the domain representation. Each web service is semantically annotated according to this ontology by means of a WS Profile. Business Processes are modeled in an implementation-independent way (e.g., without hard binding to concrete service implementations) and can be stored in OSRDF or other suitable process modeling language which allows for the decoupling of the business process logic from a concrete activity implementation. The business process in this case represents the logic of the semantic data flow between semantically described service profiles. Real world web services can be considered as instances of the corresponding web service profiles (e.g., objects of a class in OOP). The flow enactment can be done dynamically by selecting a Semantic Scenario Specification and automatically transforming it to a ready-to-execute BPEL file.

The transformation procedure lies in the selection of instances of the appropriate web service profiles involved in a particular scenario (in MDA terms: Platform Independent to Platform Specific Model transformation).

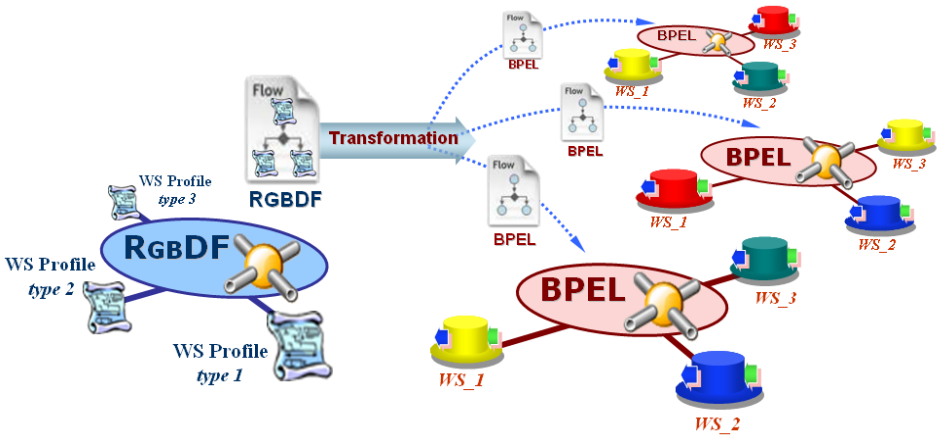


FIGURE 54 Resource independent process knowledge sharing

3.2.4 Related work

There have been some noteworthy activities in the Agent Behaviour area. One of these is an initiative of the France Telecom Research & Development (FTR&D). They provided JADE Semantics Add-on as a framework based upon JADE (JADE, 2004), to interpret the meaning of exchanged speech acts, according to their formal semantics as specified by FIPA-ACL; to make agents more flexible, in order to better interact in open environments; and to simplify the coding of JADE agents. This new add-on aims to benefit better from the semantic dimension of the FIPA-ACL language. FIGURE 55 shows a FTR&D's vision of a semantic agent.

That figure has something in common with the above described approach. "Agent knowledge" (FIGURE 55) can be compared with the History Data and the Non-Fact Data Storages (FIGURE 49). The "Create sense" module checks the rules, just like the Rule Engine (FIGURE 49), and, depending on a result, the "Consumes sense" module updates the agent knowledge and runs the correspondent Agent Behaviour. But our approach utilizes a standardized (RDF based) data representation that allows the resource agent to operate and be used in cooperation with other heterogeneous resources, i.e., it enables knowledge sharing and reuse.

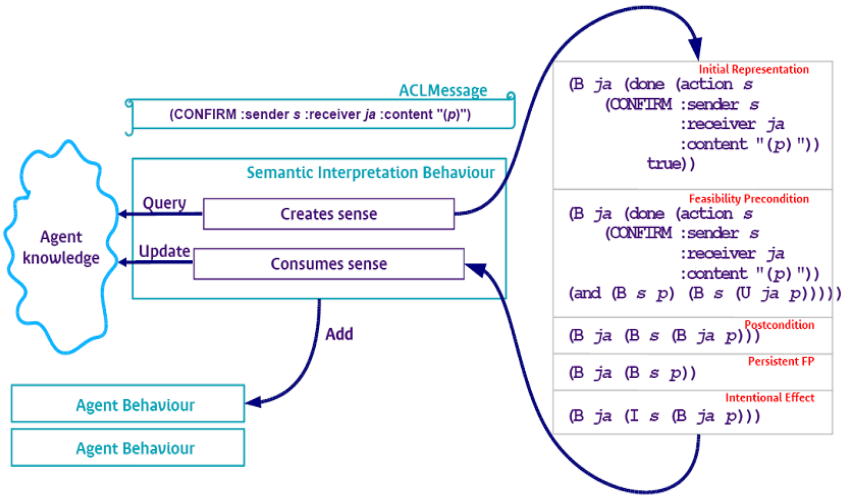


FIGURE 55 JADE Semantic Agent (JSA)

3.3 Implementation

Part of OSRDF that concerns context sensitive resource description was elaborated as Context Description Framework during the first year of the "SmartResource Project" and successfully applied to a dynamic and context-sensitive industrial data description. During the second project year the Resource Proactivity part of OSRDF was designed and used as a basis for the Platform-Agent's behaviour/desire/intension description.

The main objective of the Industrial Ontologies Group is to contribute to fast adoption of Semantic Web and related technologies to local and global industries. It includes research and development aimed to design a Global Understanding Environment as the next generation of Web-based platforms by making heterogeneous resources (files, documents, services, devices, business processes, systems, organizations, human experts, etc.) web-accessible, proactive and cooperative in a sense that they will be able to automatically plan their own behaviour, monitor and correct their own state, communicate and negotiate among themselves depending on their role in a business process, utilize remote experts, Web-services, software agents and various Web applications. Three fundamentals of such platform are Interoperability, Automation and Integration. Interoperability in GUN requires utilization of Semantic Web standards, RDF-based metadata and ontologies and semantic adapters for the resources. Automation in GUN requires proactivity of resources based on applying the agent technologies. Integration in GUN requires ontology-based business process modeling and integration and multi-agent technologies for coordination of business processes over resources. For more details about GUN, see (Terziyan, 2003, Terziyan, 2004, Kaikova et al., 2004).

The SmartResource project in its research and development efforts analyzes Global Understanding Environment decomposing it into three frameworks:

- *General Adaptation Framework (GAF),* for Interoperability (1st project year - 2004). GAF provides a framework to describe domain resources (declarative knowledge). It includes Resource State/Condition Description Framework (RS/CDF), an appropriate RS/CDF-based domain ontology, an appropriate RS/CDF Engine and a family of so-called "Semantic Adapters for Resource" to provide an opportunity to transform data from a variety of possible resource data representation standards and formats to RS/CDF and back.
- *General Proactivity Framework (GPF),* for Automation (2nd project year - 2005). GPF provides a framework to describe individual behaviours (procedural knowledge). It includes Resource Goal/Behaviour Description Framework (RG/BDF), an appropriate RG/BDF-based domain ontology, an appropriate RG/BDF engine and a family of "Semantic Adapters for Behaviour" to provide an opportunity to transform data from a variety of possible behaviour representation standards and formats to RG/BDF and back.
- *General Networking Framework (GNF),* for Integration (3rd project year - 2006).

RS/CDF is an extension of RDF and introduces the upper ontology for describing maintenance-oriented characteristics of resources, which include states and correspondent conditions, dynamics of state changes that happen, target condition of the resources and historical data about previous states.

Further, OSRDF has been used as a basis for Semantic Agent Programming Language (S-APL) (Katasonov and Terziyan, 2007) in the UBIWARE project. The project aims at a new generation middleware platform UBIWARE which will allow creation of self-managed complex systems, in particular industrial ones, consisting of distributed, heterogeneous, shared and reusable components of various nature, e.g., smart machines and devices, sensors, actuators, RFIDs, web-services, software components and applications, humans along with their interfaces, and others. Such middleware will enable different components to automatically discover each other and to configure a system with a complex functionality based on the atomic functionalities of the components. The main distinctive features of the platform are externalization of behaviour prescriptions, i.e., agents access them from organizational repositories, and utilization of the RDF-based Semantic Agent Programming Language, instead of common Prolog-like languages. In defining/referring to beliefs and goals, S-APL does not use the RDF syntax, but has them as literals of the form "subject predicate object".

CHAPTER 3

ONTOENVIRONMENT USE CASES

1 KNOWLEDGE TRANSFER FROM AN EXPERT TO AN ARTIFICIAL INTELLIGENCE (AUTOMATED INDUSTRIAL MAINTENANCE)

Nowadays, the lack of new qualified personnel is a big problem for almost all the companies and organizations (especially in an industry). Experts change their place of work or retire on a pension and take away their knowledge with them. Thus, we have the challenging task to transfer the tacit knowledge from experts to artificial intelligence that later would be able to replace some of the experts.

To resolve this problem was one of the SmartResource project aims. The main research objective of the project was to provide tools and solutions to make heterogeneous industrial resources (files, documents, services, devices, processes, systems, human experts, etc.) web-accessible, proactive and cooperative in a sense that they would be able to analyze their state independently from other systems or to order such analysis from remote experts or Web-services in order to be aware of their own condition and to plan their behaviour towards effective and predictive maintenance (see FIGURE 56). The outcome of this project actually is a prototype of OntoEnvironment. All the scenarios of the project are based on the OntoEnvironment approach including context-sensitive resource annotation, role-based goal-driven resource proactivity and coordination.

A scenario of the tacit knowledge transfer from an expert to a service is presented in FIGURE 57. It consists of three different types of resources (industrial devices, Web-services for intelligent diagnostics and human experts). The basic idea of this scenario is that the device is self-monitoring and in case of some fault or alarm initiates request to the expert for human diagnostics or to the Web-service for automated diagnostics. Let the Web-service in this case be some intelligent tool, which is based on Neural Network diagnostics and which should have been trained on some training set of already diagnosed samples prior to making diagnostics. Consider the human expert as the best but expensive and not always available source of diagnostics for data

from device sensors. We can split the whole scenario to three scenes. During Scene 1, the agent responsible for the device plays the role of a "patient", which means that it monitors

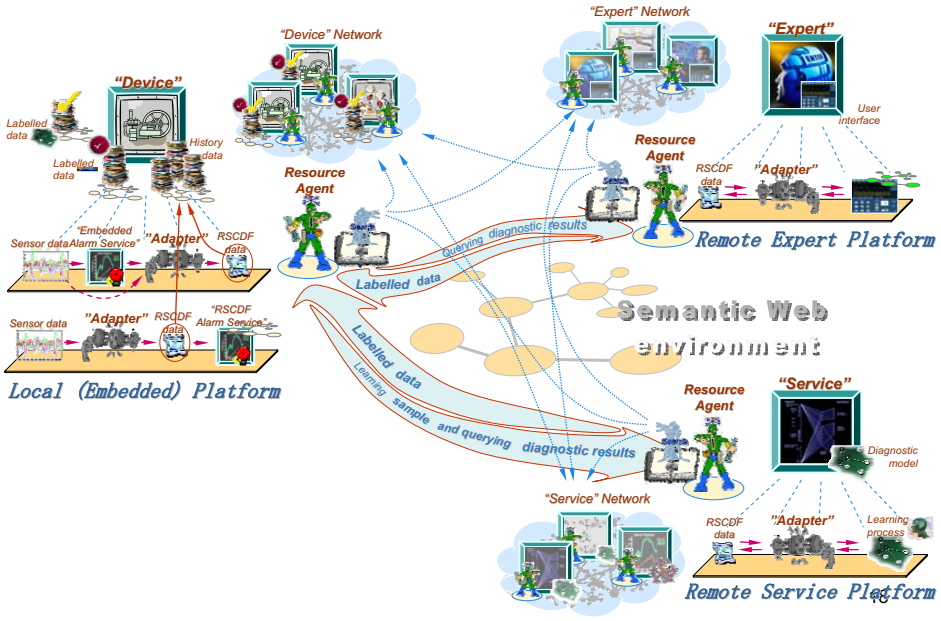


FIGURE 56 Maintenance Networking Environment

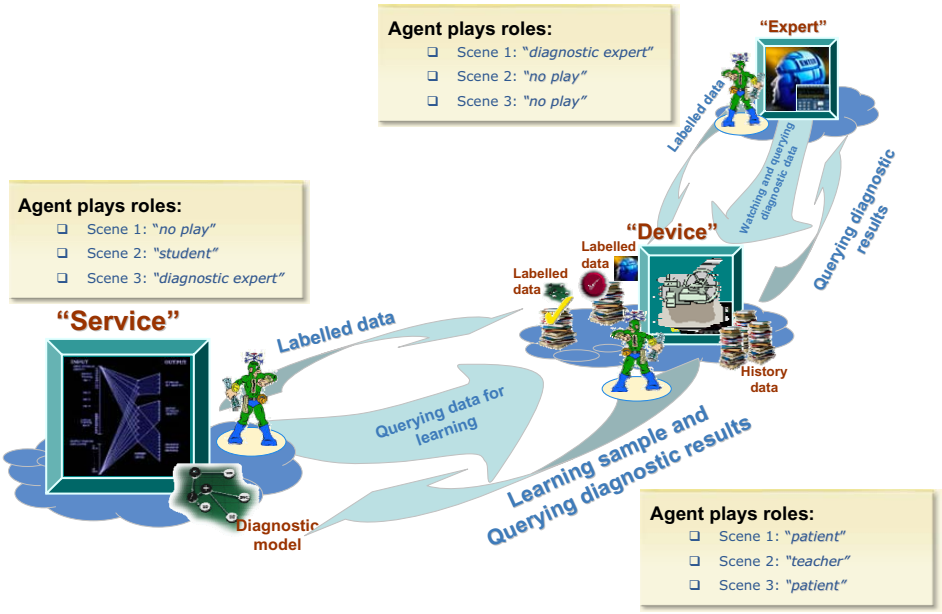


FIGURE 57 Knowledge transfer from an expert to a service

its own parameters via some sensors and in case of any alarm sends these parameters to a human expert for the diagnostics. In this scene the agent of the expert plays the role of a "diagnostic expert", which is responsible to reply by naming a concrete diagnosis based on the requests from the device agent. The agent of the Web service in this scene is passive. During Scene 2 of the scenario, after the device agent has collected enough cases of its own diagnoses it can change the role from a "patient" to a "teacher", which means that it can provide training samples to the Web-service. Accordingly, the agent of the web-service will be taking the role of a "student", i.e., one who will learn based on a sample set and will produce some neural network for future diagnostics. The agent of the expert will not play any active role anymore. During Scene 3, after the Web-service has learned and is able to make diagnostics automatically, its agent is taking the role of a "diagnostic expert" and the agent of the device can take the role of a "patient" back, because now it can address all its diagnostic requests to the Web-Service. The above scenario shows that the roles (i.e. appropriate behaviours) of the agents can be chosen and changed depending on the current context of the situation, and this means that each agent should be able to download from some shared place the description of a new role, whenever needed. As more advanced resource maintenance scenarious, consider the sample scenarious in FIGURE 58 and FIGURE 59. Assume that some industrial GUN resource (device) on a local GUN platform has collected labeled data (e.g., faulty state descriptions of personal history labeled by experts with appropriate diagnoses). The following procedure has been widely used in a machine learning field known as learning ensembles of classifiers. The local platform agent divides the data into two subsets: training sample set and test sample set. The training set is given to several external services so that they are able to adjust their models or learn the new diagnostic model specifically for that device from scratch. External services can in principle support different learning algorithms (e.g. Neural, Fuzzy, Bayesian, Genetic, etc.). After all contacted services have reported that they have learned what they should, the device provides them another part of the labeled data, i.e. the training set with the hidden actual diagnoses. Services are requested to provide the diagnoses for given cases. The device agent compares the received outcomes from the services with known (actual) diagnoses and calculates for each service the performance value (e.g. percentage of correctly classified cases from the test set) and considers these values as personalized ranks of services, which determine the trust of the "device" towards appropriate service providers. The ranks can be used in future to diagnose new states of the device either by selecting the best ranked service from the available ones or by integrating the outcomes from several services as weighted (based on ranks) voting among them. This approach should increase diagnostic performance in comparison to randomly selected service or simple averaging of outcomes from several services. The second scenario shows us the "one service – many devices" interaction model. A service can build one diagnostic model for each device, or a common model for a class of them. The service is then able to serve a new device that does not have an appropriate history yet.

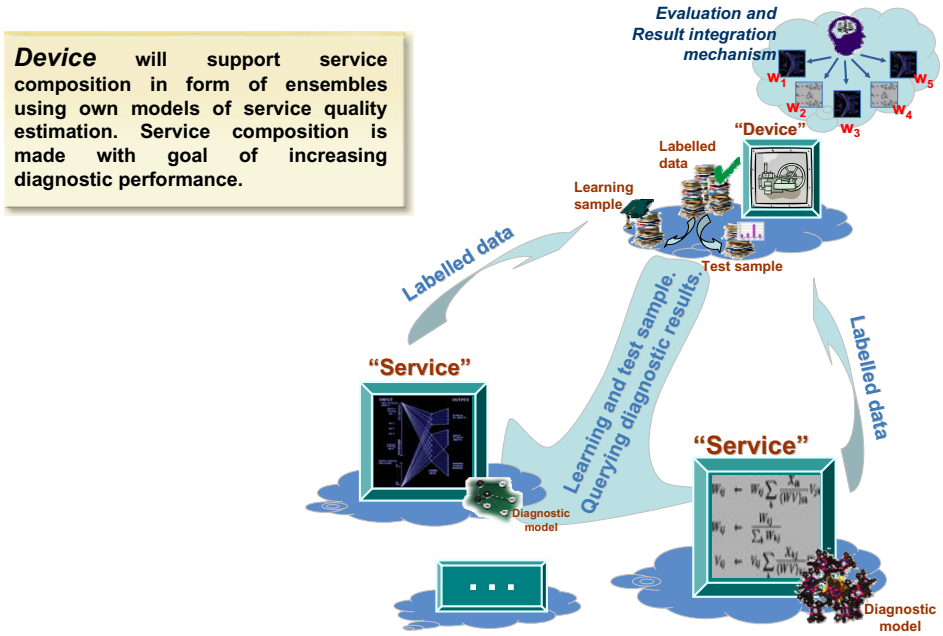


FIGURE 58 One device – many services

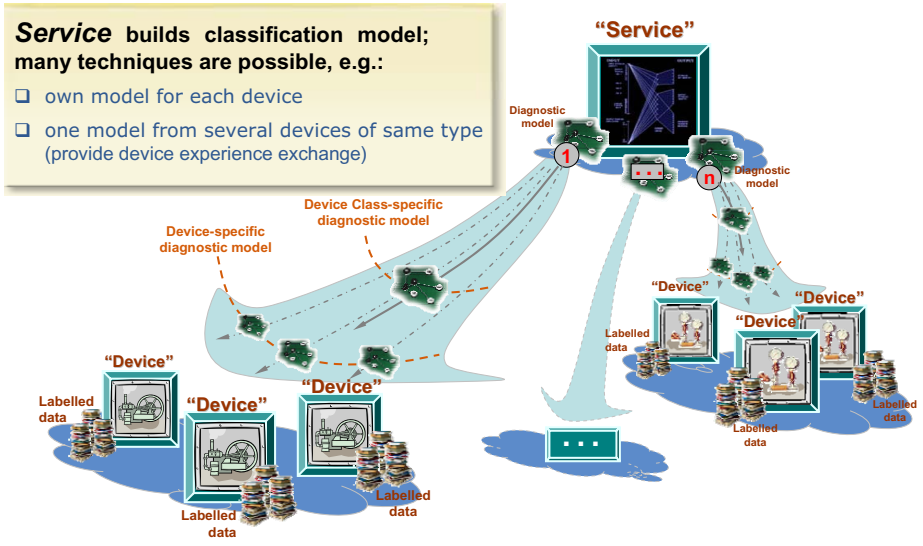


FIGURE 59 One service – many devices

2 ENVIRONMENT FOR INTELLIGENT VISUALIZATION OF INTEGRATED INFORMATION

2.1 4i (FOR EYE) technology: Intelligent Interface for Integrated Information.

Now it has become evident that we cannot separate visual aspects of both data representation and graphical interface from interaction mechanisms that help a user to browse and query a data set through its visual representation. Following the GUN-Resource centric approach, let us consider user interfaces for context-based resource access and contextually related information retrieving. The challenging task is to create a visual interface that provides integrated information from a variety of information providers in a context-dependent way.

Regarding the Intelligent Resource Visualization approach (presented in Section 3.6 of Chapter 1), depending on a context, a human/expert needs information (information related to subject resource) to be visualized in certain way. At the same time, we need an interface for access to a resource also in a context depended way. This gives us one of the requirements for visual interfaces – an ability to represent information regarding the chosen contextual property of the resource. Such interface should allow the user to simply choose a context for data representation, and should even support cases of multiple contextual property selection for complex views and filtering purposes. Such requirements can be met by MetaProviders - sui generis portals of GUN-Resources with a specific visualization view. The name, MetaProvider, is due to the fact that it provides an access and presents other resources, which in turn are providers of their own information (data). All GUN-Resources have a certain own location (physical and digital). But it does not mean that they should have just one way to get an access to it. MetaProvider is an interface-mediator that makes it possible to mark/select a resource (object) on its

interface and provide a link to the original resource location. In other words, it allows resource registration for further access to its data. At the same time, any resource can be registered on a variety of different MetaProviders in different views. The main feature of the MetaProviders is that each party that takes care of some GUN-Resource registers the resource itself. It causes fast filling in of information accessible through a MetaProvider. And each user/resource in one moment has an access to the related information portion of others. But such interoperability brings a new requirement for the MetaProviders and users. They should share a common ontology that is interoperable on a semantic level. In addition to semantic interoperability, GUN-Resources are proactive/goal-driven resources and supplied with a Resource Agent for resource-to-resource (R2R)/ agent-to-agent (A2A) communication.

4i (FOR EYE) is an ensemble of GUN Resource Platform Intelligent GUI Shell (smart middleware for context dependent use and combination of a variety of different MetaProviders depending on user needs) and MetaProviders, visualization modules/platforms that provide context-dependent filtered representation of resource data and integration on two levels (information/data integration of the resources to be visualized and integration of resource representation views with a handy resource browsing). Context-awareness and intelligence of such interface brings with them a new feature that enables the user to get not just raw data, but required integrated information based on a specified context. GUI Shell allows user dynamic switching between MetaProviders for more suitable information representation depending on a context or resource nature. MetaProvider plays four main roles:

- Context-aware resource visualization module that presents information regarding a specified context in a more suitable and personalized form (Intelligent Resource Visualization approach) for the user;
- Interface for integrated information visualization with intelligent context-aware filtering mechanism to present only relevant information, avoiding a glut of unnecessary information;
- Visual Resource Platform that allows resource registration, search, access and modification of needed information/data in a space of registered resources;
- Mediator that facilitates resource to resource (R2R) communication.

Such switching and filtering process is kind of visual semantic browsing based on semantic description of the context and resource properties.

FIGURE 60 shows us the principle of such smart ensemble work. There are three GUN-Resources (power line, forest and weather) that are registered on a set of MetaProviders. Let us consider the GUN-Resource that presents a power line as the main initiator of a visualization process. An intelligent GUI, as a part of the GUN Platform, provides an opportunity for the user to initiate a context-based search process that returns the appropriate MetaProvider or a set of them. Search process can be performed via a centralized or a decentralized system of MetaProviders' registration. Depending on a contextual property, the Intelligent GUI Shell provides an access to the filtered set of the retrieved MetaProviders. Then the user can register the resource (if it is not registered

yet) or/and get the related resource integrated information on the MetaProvider interface. It is not necessary that a search result should contain just one instance of a fit class of the MetaProviders. This is an open environment and there can be a set of various realizations of MetaProviders from different producers. Also, GUI Shell allows dynamic switching between MetaProviders for more suitable information representation, depending on a context (a set of contextual resource properties). On the other hand, MetaProvider provides an API to specify an information filtering context – the context for visualization of appropriate resources and their necessary properties. In this scenario the user asks the MetaProvider to show the resources in a certain area around the subject resource (power line) in the context of physical damage and relevant to these resource properties.

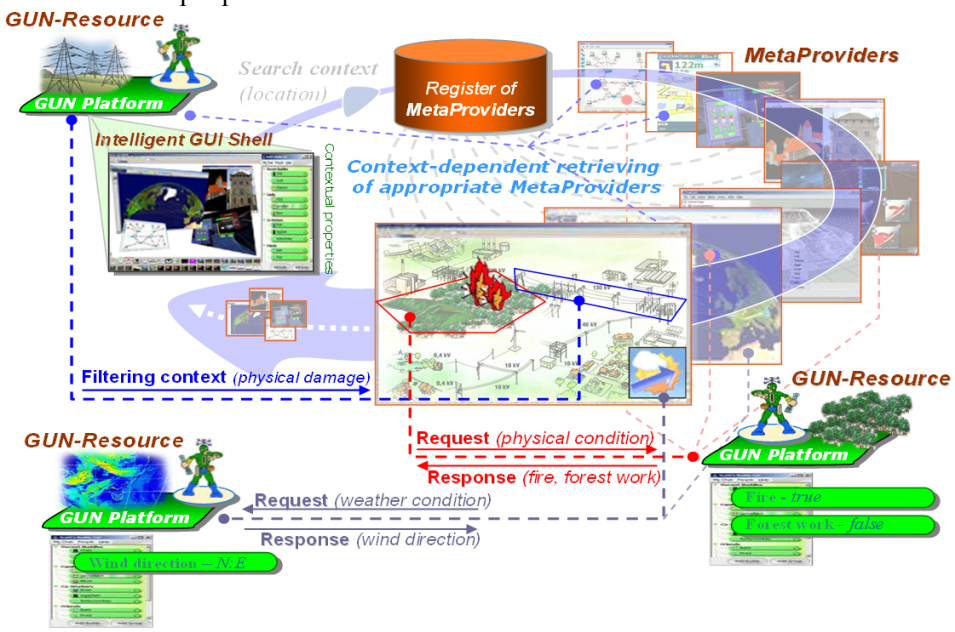


FIGURE 60 Intelligent Interface for Integrated Information (4i technology)

Thus, the physical conditions of other two resources (forest and weather that are shown in the figure) have been requested and the values of the corresponding properties are shown on the interface. Now, when an expert has recognized an alarm situation, he/she needs, for example, to change the architecture of the electrical chain (electricity supply) and for this can easily switch to another MetaProvider with a more appropriate internal view of the power line architecture. Another valuable benefit of such smart ensemble architecture is the possibility to perform autonomous agent-based resource communication via the MetaProvider's and GUI Shell APIs. It is an open environment for MetaProviders and it is a good base for different business models that can be built on it. Thus, to make each player an interoperable part of the open environment, each has to be supplied with the API and each should

be semantically adapted to understand the requests and to provide understandable response.

Now, when a human becomes a very dynamic and proactive resource of a large integration environment with a huge amount of different heterogeneous data, it is quite necessary to provide a technology and tools for easy and handy human information access and manipulation. Semantic-based context-dependent multidimensional resource visualization provides an opportunity to create an intelligent visual interface that presents relevant information in a more suitable and personalized form for the user. Context-awareness and intelligence that such interface brings allows the user to get not just raw data, but required information based on a specified context.

Now it is clear that when unlimited interoperability and collaboration demand data and information sharing, we need more open semantic-based applications that are able to interoperate and collaborate with each other. Ability of the system to perform semantically enhanced resource search/browsing based on resource semantic description brings a valuable benefit for the today's Web and for the Web of the future with unlimited amount of resources. Subscribing to an opinion of Nixon (Nixon, 2006), bridging the gap between the emerging folksonomies of Web 2.0 and the formal semantics of the Semantic Web, ontologies would benefit the Semantic Web community, which would be able to leverage the content and knowledge that Web 2.0 is already generating from its users and making available over standardized APIs. The proposed technology allows creation of a human-centric open environment for resource collaboration with an enhanced semantic and context-based visual resource browsing. 4i (FOR EYE) technology can be considered as a valuable extension of the text-based Semantic MediaWiki to Context-based Visual Semantic MediaWiki, a new generation of resource collaboration environments that follow the vision of Web 3.0. This is a good basis for the different business, production, maintenance, healthcare, social process models creation and multimedia content management.

2.2 OntoEnvironment for 4i (FOR EYE) and 4i (FOR EYE) for the OntoEnvironment.

As was previously mentioned, 4i (FOR EYE) is an open environment for various realizations of resource visualization modules and is a perfect subject to be developed on the base of OntoEnvironment. All the players of the intelligent integrated information representation process (Intelligent GUI Shell and various MetaProviders) can be considered as smart resources (OntoResource) provided and supported by different parties (FIGURE 60). According to the OntoEnvironment approach, all these players are supplied with a ResourcePlatform in order to make them proactive goal-driven context-sensitive resources adapted to GUN. This makes the GUI Shell and

MetaProviders more compatible with other resources, i.e., with the customers of the visualization service and those resources that should be requested by this service and visualized. At the same time, the goal-drivenness and proactivity of the MetaProviders enable on-the-fly resource visualization service enhancements and help in building new business models.

On the other hand, context-aware intelligent resource visualization (see Section 3.6 from Chapter 1) is one of the significant features of the OntoEnvironment. The intelligent GUI Shell is a part of the ResourcePlatform, and the 4i (FOR EYE) technology itself is a part of the OntoEnvironment approach. Thus, the 4i (FOR EYE) technology is used by the OntoEnvironment to perform intelligent resource visualization and, at the same time, the 4i (FOR EYE) technology is based on the OntoEnvironment approach.

Humans are important resources and can play several distinct roles: a resource under care, a service provider, a user, and an administrator. Obviously, they need some graphical interfaces to interact with the rest of the system. The same person can play several roles, switch between them depending on the context, and, as a result, require different interfaces at different times. OntoEnvironment is a large integration environment with potentially huge amounts of heterogeneous data. Therefore, there is a need for tools facilitating information access and manipulation by humans. A semantic context-aware multimodal visualization approach would provide an opportunity for creating smart visual interfaces able to present relevant information in a more suitable and more personalized form.

In the UBIWARE project we consider a human interface as a special case of a resource adapter (Human-Resource AdaPter - HRAP). We believe, however, that it is unreasonable to embed all the data acquisition, filtering and visualization logic into such an adapter. Instead, external services and applications should be effectively utilized. Therefore, the intelligence of a smart interface (HRAP) is a result of collaboration between multiple agents: the human's agent, the agents representing the resources of interest (those to be monitored or/and controlled and requesting a human), and the agents of various visualization services – MetaProviders via an agent of HRAP (see FIGURE 61).

Based on the 4i (FOR EYE) technology, an infrastructure will be embedded into UBIWARE enabling effective realization of the following system functions:

- visualization of data provided by a service in response to a request;
- search, retrieval and visualization of data required by a human expert;
- access to contextual information, and its visualization;
- visualization of resource registration, configuration, and security policy establishment processes;
- resource discovery via MetaProviders (because they act as thematic portals).

The first prototype of GUI-Shell and MetaProvider for "MemberOf"-visualization has been elaborated during the fist yaer of UBIWARE project. FIGURE 62 presents a screen shot of the 4I (FOR EYE) browser in case of IOG research group (resource) visualization in context of personnel/staff.

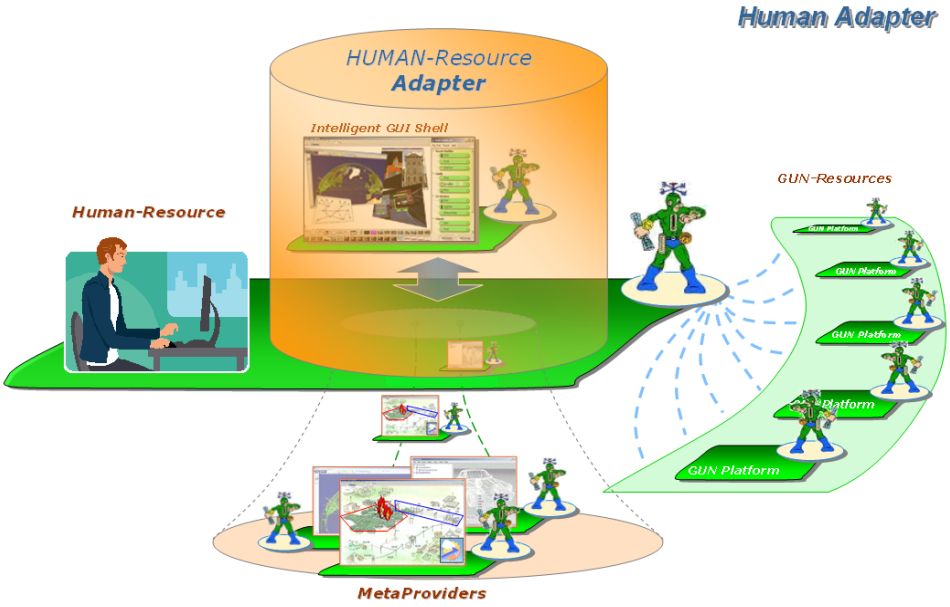
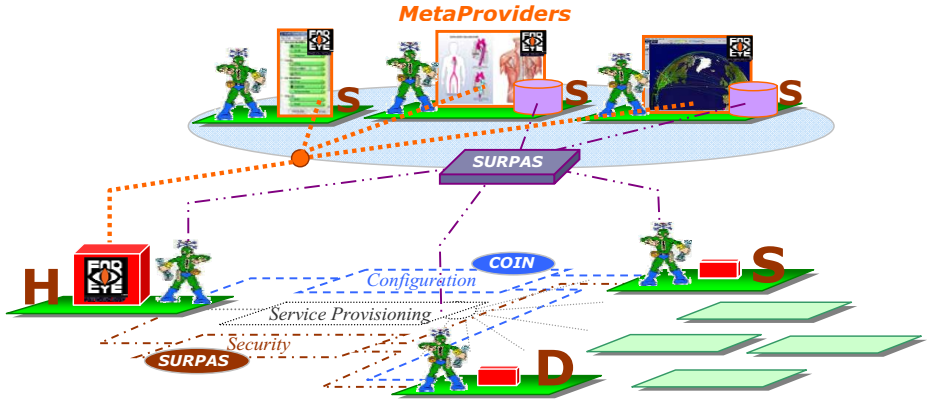


FIGURE 61 Human-Resource Adapter based on 4i (FOR EYE) technology

FIGURE 62 SmartInterface screen shot

This approach plays a significant role in UBIWARE – it connects human eyes and human hands to it. Approach will enable UBIWARE to provide flexible and context-aware interfaces to humans participating in the activities of a UBIWARE based system (OntoEnvironment). Assuming that the system will have some human administrators, there is a need for a GUI through which the administrators will be able to manually configure the system, which also includes defining the security policies. Therefore, 4i (FOR EYE) must allow for that, and special MetaProviders have to be developed for creating interfaces to perform such administrative functions. As shown in FIGURE 63, the research on smart interfaces will affect and will be affected by the research and results from other UBIWARE project work packages, such as WP3: SURPAS "Smart Ubiquitous Resource Privacy and Security" and WP4:COIN "Self-Management, Configurability and Integration".



H, D, S – human, device and service respectively.
– resource configuration process.
– resource security policy and privacy establishment process.
– automated agent-based resource search process.
– visualization modules utilization process.
– resource adapter.

FIGURE 63 Processes related to UBIWARE Platform and 4i (FOR EYE)

3 4I MULTIMEDIA: SEMANTICALLY ENHANCED MULTIMEDIA BROWSING

3.1 Resource semantic track

The challenge of enabling computer systems to make better use of Web data by making that data machine-processable has been taken up by the Semantic Web effort, which proposes formal knowledge structures to represent concepts and their relations in a domain. These structures are known as ontologies and the World Wide Web Consortium has recommended two standards: the simpler, RDF, and the more expressive one, OWL,.

A number of vocabularies that deal, at some level, with multimedia content currently exist (Geurts et al., 2005): MPEG-7, Dublin Core Element Set, VRA, Media Streams, Art and Architecture Thesaurus (AAT), MIME, CSS, Composite Capabilities/Preference Profiles (CC/PP), PREMO, Modality Theory, Web Content Accessibility Guidelines. Of course, it is very important to develop an appropriate format for semantic annotation of multimedia content. But, on the other hand, it is more natural to find a way to build full semantics into the digital formats of multimedia (image, video, audio). Nowadays, production houses shoot high-quality video in a digital format; organizations that hold multimedia content (such as TV channels, film archives, museums, and libraries) digitize analog material and use digital formats. Maybe it is the time to bring together all the digital media formats. For that Semantic Track can be used. A semantic track will contain not just the content structure, but full semantic content annotation including content structure, concepts, objects, actions and etc.

The discussions around multimedia strongly suggest that humans are the main customers of multimedia services and end-users of multimedia content. With a sustainable multimedia content growing, the human/user needs new intelligent techniques for multimedia content browsing, search, and retrieval and for adapted representation. At the same time, the stated goal of the Semantic Web initiative is to enable machine understanding of web resources.

However, it is not at all evident that such machine-readable semantic information will be clear and effective for human interpretation. Hence, in order to effectively harness the powers of the semantic web, it needs a "conceptual interface" (Naeve, 2005), that is more comprehensible for humans. Such conceptual interface can improve multimedia content retrieving process and together with a well elaborated Semantic Track of the multimedia resources, can provide a unique basement for semantically enhanced across multimedia contents browsing.

The sub-symbolic abstraction level covers the raw multimedia information represented in well-known formats for video, image, audio, text, metadata, and etc. These are typically binary formats, optimized for compression and streaming delivery. They are not well suited for further processing that uses, for example, the internal structure or other specific features of the media stream. A structural (symbolic) layer on top of the binary media stream provides this information. The standards that operate in this middle layer for the representation of multimedia document descriptions are: Dublin Core, MPEG-7, Visual Resource Association, and so on. The problem with this structural approach is that the semantics of the information encoded in XML are only specified within each standard's framework. MPEG-7 was not built specifically for web applications and thus does not facilitate embedding links to other resources and interoperability between them. A possible solution to resolve the interoperability conflict is to add a third layer (the logical abstraction level) that provides the semantics for the middle one, actually defining the mappings between the structured information sources and the domain's formal knowledge representation based on semantically enriched languages (RDF and OWL).

RDF based languages and technologies provided by the W3C community are well suited to the formal, semantic descriptions of the terms in a multimedia document's annotation. A combination of the existing standards seems to be the most promising path for multimedia document description in the near future. For these reasons, the W3C started Multimedia Annotation with the Semantic Web MM Task Force[29] (later continued with the Multimedia Semantics Incubator Group at the closing of the Semantic Web Best Practices and Deployment Working Group that guided some aspects of these activities). The task force operated within the framework of the W3C Semantic Web Activity group[30]. One goal was to provide guidelines for using Semantic Web languages and technologies to create, store, manipulate, interchange, and process image metadata. Another was to study interoperability issues between multimedia annotation standardization and RDF- and OWL-based approaches. Hopefully, this effort will provide a unified framework of good practices for constructing interoperable multimedia annotations.

Research towards multimedia content and content description bounding has been going for the last several years. Commonwealth Scientific and

---

29 http://www.w3.org/2001/sw/BestPractices/MM/

30 http://www.w3.org/2001/sw/

Industrial Research Organization have developed an open source family of technologies, ANNODEX (Pfeiffer et al., 2003), for embedding annotations and hyperlinks directly within digital audio and video files. Such embedding allows the combined resource to be treated as like any web document which has content and content description bound into one. Also, the idea of utilizing a media semantic track has been elaborated in one of my research works (Khriyenko, 2005), which concerns some of the issues of multimedia smart messaging in an environment of limited devices.

Semantic annotation of multimedia content is performed by using appropriate domain specific ontologies that model the multimedia content domain. Ontologies typically represent concepts by linguistic terms. However, also multimedia ontologies that assign multimedia objects to concepts can be created. At the same time with semantic content metadata annotation, we should provide a basis for multimedia content features to be presented in semantic annotation also. This would include annotation of the concepts such as people (artist, owner, restorer, author, producer, etc.), art objects and representations (painting, sculptures, films, digital representations, etc.), events and activities, places, methods and techniques. This would create the conditions for better automatic annotation of multimedia content. Later on, we try to specify the features of the multimedia content that can be detected and presented in a Semantic Track.

Bertini et al. (Bertini et al., 2005) present a list of systems of automatic semantic annotation, most of them in the application domain of sports video. Among these, there is an approach where MPEG motion vectors, playfield shape and player's position have been used with the Hidden Markov Models to detect soccer highlights. Another approach has been aimed to detect the principal soccer highlights, such as shot on goal, placed kick, forward launch and turnover, from a few visual cues. Also, the ball trajectory also has been used in order to detect the main actions like touching and passing and to compute the ball possession by each team; a Kalman filter has been used to check whether a detected trajectory can be recognized as a ball trajectory. But, in all these approaches a model based event classification has not been associated with any ontology-based representation of the domain. However, although linguistic terms are appropriate to distinguish between event and object categories, they are inadequate when they must describe specific patterns of events or video entities. In this case, high level concepts, expressed through linguistic terms, and pattern specifications represented instead through visual concepts, can be organized into new extended ontologies, that will be referred to as pictorially enriched ontologies. Ontologies can be extended to multimedia enriched ontologies where concepts that cannot be expressed in linguistic terms are represented by prototypes/patterns of different media, like video, audio, etc.

Also, there are audio features that can be used to characterize sound signals and classify instrument samples. The CUICADO project (Peeters, 2003), provided a set of 72 audio features, and research has shown that some of the features are more important in capturing the signal characteristics. These

features are temporal shape, temporal feature, energy features, special shape features, harmonic features, perceptual features and MPEG-7 Low Level Audio Descriptors (spectral flatness and crest factors). It is interesting to see then how many are the multimedia-specific features and properties that can enrich the Semantic Track of multimedia resources.

3.2 Across multimedia content semantically enhanced browsing

We have to consider another developing trend on the Web – a growth in multimedia content. Technological progress has meant that we have never had access to so much media content as now. Future challenges for the Web will include the meaningful organization of this huge amount of online media content as well as its meaningful delivery to the user. However, the present state of the art of media and Web technologies prevents a richer integration.

Multimedia semantic browsing, as a sub-class of general resources browsing, is a complex process that combines various sub-processes. This process can be based on the presented 4i (FOR EYE) technology. FIGURE 64 shows us an example of an across multimedia contents semantic browsing architecture. A GUI-Shell in the center left of the figure is presented as a combination of the tools that take part in the process: a multimedia content player, a Semantic Track visualization component, a concept browser and a semantic search query builder/creator.

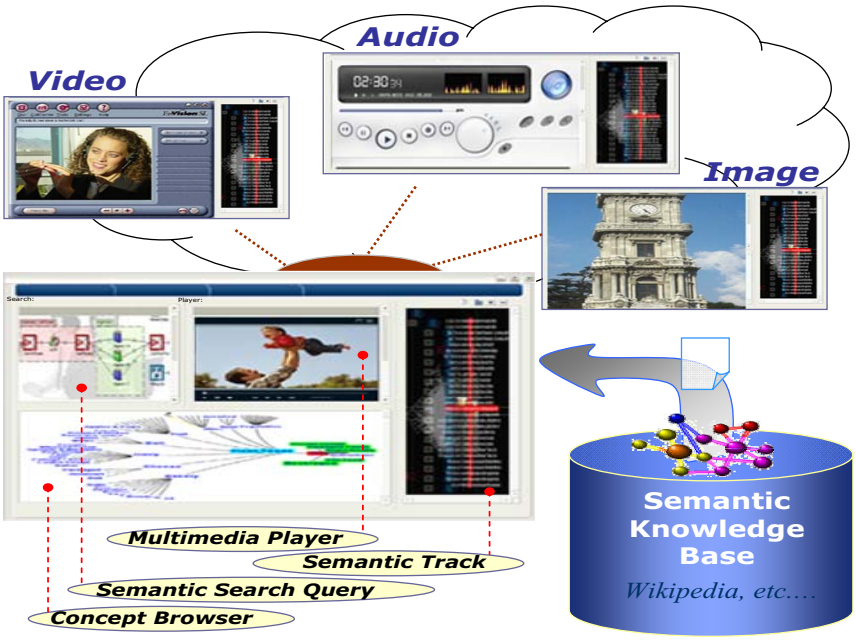


FIGURE 64 Semantically enhanced though multimedia content browsing

Let us consider an example where the user is watching an episode of a movie with some song (soundtrack) at the background. The user likes this song/melody and would like to find more songs of its author (or an even more complex goal – find songs similar to the initial one). To achieve the goal, the user should browse the Semantic Track of this video instance, which contains a structure of a video file, its objects, actions, soundtracks, etc., and find a reference to the searched song. Then, utilizing a concept browsing tool, which is connected to a remote ontology, the user can specify a semantic query for the needed multimedia resource (in our case - a song). Such query specification can be considered as a creation/construction of a resource semantic pattern (virtual nested resource with specified properties). As a result of the search process, the appropriate audio resource will be returned and even the lyrics of the song can be displayed based on its' Semantic Track.

But this was just a simple case of a semantic search/browsing process. A Multimedia Resource Semantic Track usually contains just a structure of content and descriptions of multimedia content specific features (see the previous section). And because of this, very often we can not specify a direct link between the contents of two Semantic Tracks for different resources. The "glue" for these two semantic annotations is to be found in Semantic Knowledge Bases (for example in a semantically-enhanced Wikipedia or in different ontologies). It shows its usefulness in the next example. Let's suppose we are looking for an image of the house of the first wife of some actor from a movie that we are watching. First, we stop the movie on a scene where this actor is presented and, based on the Semantic Track, find a link to this person. Then we browse a semantic knowledge base via the concept browser and find a link to his first wife and her house. After a semantic search query generation we get the searched image on the browser.

At the same time, the approach of instance based search via MetaProviders can be beneficially utilized in multimedia content searching/browsing. Let us consider a case where we would like to see other houses, which are located near the house of the mentioned wife. We can use some MetaProvider – Wikimapia kind of service, which provides an access to the registered resources by showing them on a map. If the image is registered on this service/platform, then we easily can find other registered images in the same area (location), especially if the final visualization will be filtered in a context that the searched resource is an image of a house.

According to the GUN approach, all the parts of searching/browsing process presented in GUI-Shell can be developed as separate functional modules (resource) and can be chosen by the user to allow personalization of the browsing interface. In this particular use case of the OntoEnvironment with resources of the real world (people, objects and etc.) we come across new semantically-enhanced media-file resources. As was mentioned, these resources contain not just their internal structure in their Semantic Tracks, but also links to other resources. Thus, to be competitive in the open market of the media resources and to be highly ranked by usage, resources should be self-maintained and have up-to-date links in the Semantic Track at all times. Here

we see the necessity of proactive behaviour by the resources. Supplied with an agent-based GUN Platform, behaviour of the resource can be configured in a way that allows a resource to communicate with other resources and change/update its own Semantic Track in real time (see FIGURE 65).

FIGURE 65 Semantically enhanced multimedia resource infrastructure

4 SEMASM: SEMANTICALLY ENHANCED SMART MESSAGING

Today the Multimedia Messaging concept facilitates new styles of communication that respond to the needs of the Mobile World, where business and personal lifestyles are changing and evolving very fast. Communication is at the heart of people's lives. At the same time, the recent trends connected with information integration demand solid technology to provide interoperability between heterogeneous, interacting applications. Applications of the Semantic Web recently have presented many interesting opportunities in this domain. In this paper a Semantically Enhanced Messaging approach is presented and the possibilities are analyzed for applying this approach in the development of next-generation Semantic Web enabled ICT products in the communication domain. Future mobile users are expected to be interested in semantic messaging and new applications built on top of semantically enhanced multimedia content. Semantic messaging provides interoperability between heterogeneous message based applications, decreases user expenses for data transmission and provides a more advanced tool to access distributed information. Semantically enhanced Smart Messaging (SemaSM) is designed to be easy-to-use, fun, and an attractive service. For other players (message content providers, service providers, access device producers and application providers) it offers a future-proof, evolutionary migration path and thus a road to profitable business.

4.1 Semantically enhanced Smart Message Framework

The mobile phone is infused with a rich new meaning. Now, when mobile phones have become a part of our way of life, it is impossible to find even a single user of a mobile terminal who has not used SMS (Short Message Service) messaging, the pioneering format in wireless data exchange technology that has become so popular. And the number of MMS (Multimedia Messaging Service)

users also is increasing each day. MMS is one of the most recent developments in mobile messaging. Just as the traditional short message service, multimedia messaging provides automatic and immediate delivery of personal messages. Unlike the SMS however, MMS is a next-generation messaging protocol that allows mobile device users to incorporate audio, video, images and other rich content into traditional text messages, transforming them into personalized visual and audio messages. MMS offers an exciting functionality and full multimedia content to subscribers and an array of business opportunities to operators and service providers (IBM_CH). MMS technology offers more than just a broadening of message content. With MMS, it is not only possible to send your multimedia messages from one phone to another, but also from phone to email, and vice versa. This feature dramatically increases the possibilities of mobile communication, both for private and corporate use. Multimedia messaging reshapes the landscape of mobile communication, making it more personal, more versatile, and more expressive than ever before.

The recent trends connected with information integration demand solid technology to provide interoperability between heterogeneous, interacting applications. Applications of the Semantic Web, in recent researches, have shown many useful features for this domain. Companies developing software can contribute a lot in the integration processes which provide interoperability for legacy systems that now have to be integrated. This can be done through the development of software that has, on the one hand, software-specific interface and. On the other, standardized interface for data access. Original data, in order to be available to external applications, needs to be presented in some format which those applications can process. To present things in a commonly understandable form is one of the goals of the Semantic Web technology. Interoperability issues can be overcome if software-developer companies utilize the potential of this technology. Software systems can be adapted to produce data using a common “language” provided by the Semantic Web approach.

This work introduces an extension of the MMS content to semantically enhanced message content. The second chapter of this paper describes the semantically enhanced Smart Messaging Framework and its main aspects within this approach. The third chapter is dedicated to the issues concerning a supportive interface for semantic messaging. The last content part of the paper describes a business opportunity within the Semantic Messaging approach.

We can see the user's growing interest in information exchange taking hold. The types of information that can be sent are not limited only to a text, video or audio message or images. Unfortunately, there is a lot of information which can be managed and understood just by human. An exchange of information which can be processed not just by human, but also by software applications would be a very important innovation for the growing communication domain. It would allow users to manipulate applications and say what they want in an easy way via a message, and it would, at the same time, allow software understand human needs. There is a lot of information people want to exchange. For example there are schedules of activities (especially for managing cooperative work or organizing group activities),

appointments, various settings of mobile terminal's properties (such as a profile), multimedia data, and others. Another important aspect is that the users need to exchange various information via easy, effortless mechanism of sending (creation) and receiving such Smart Messages.

In this case, we have a need to develop an application for processing complex messages which contain many types of information. On the one hand, such software (Smart Message Manager (SMM) agent for supporting smart message exchange) has to support a user interface capable of accessing data stored on a terminal. On the other hand, it has to access other software applications and data storage on a mobile terminal. This supporting application can be developed and imbedded in their product by mobile terminal producers. To be able to interact, exchange and understand information, these products must support a common ontology 'language'. Information resources exchanged via a Smart Message also must be described (annotated) via a common ontology processable by the application. For such information resource annotation, the mobile terminal has to be supplied by a semantic annotation mechanism which allows the user to create it.

There are additional benefits from data annotation to software development even if there is no need to deliver information outside the original computing system: special formats of data exchange between applications are needed no more, since a common standard based on ontology will have been put in place. Software can be developed in a modular, scalable manner and with support for this standard (ontology). Such commitment to a shared (upper) ontology will result in software compatibility. As new types of information appear, the ontology can be extended to reflect the changes demanded and continue being used as a standard. Extensibility of ontology is its inherent feature.

4.1.1 Interoperability between heterogeneous mobile devices (applications)

The common ontology approach provides interoperability between products of different producers. The idea is based on the resource adaptation approach, i.e., adaptation of any resources to Global Understanding eNvironment. Such adaptation brings a transformation of the specific data representation formats to a common transaction format. To accomplish this transformation we need to develop bidirectional transformation modules (adapters) serialized via the context sensitive extension of the RDF standard (via Context Description Framework, the lite version of the CDF-Schema) with a support for the corresponding ontologies.

Let us consider, for instance, a calendar entry. The calendar model contains three calendar entries (meeting, memo and anniversary), each of which has a set of facets (properties). FIGURE 66 shows the properties belonging to the calendar entries.

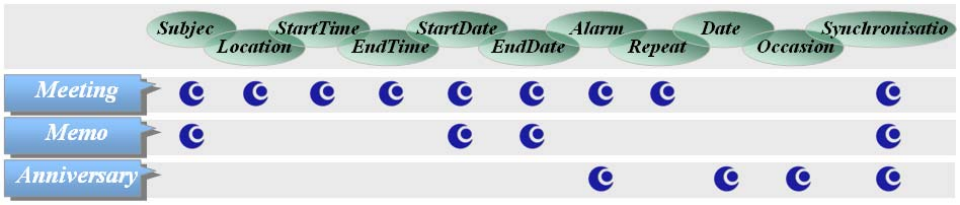


FIGURE 66 Calendar model (property belonging to the calendar entries)

A simple hierarchy of the properties (which allows describing the calendar instance independently from a phone model) represents also a simple mobile calendar ontology. In addition to the property hierarchy, the ontology contains a specification of the property values. The referred prototype of the phone calendar ontology is elaborated and can be found in Appendix B.

In the property hierarchy we can emphasize two main super properties: *calendarEntry* (describes that subject device (phone) has some calendar entry in the form of a statements container, which represent the entry's properties) and *calendarProperty* (describes the abstract calendar entry property). The *meetingEntry*, *memoEntry*, and *anniversaryEntry* properties are subproperties of the *calendarEntry* property and represent the corresponding calendar entries. And *subject*, *location*, *startTime*, *endTime*, *startDate*, *endDate*, *alarm*, *repeat*, *synchronization*, *occasion* and *date* are subproperties of *calendarProperty* and correspond to the properties of the calendar entries. This property ontology is elaborated based on the CDF-Schema in the context sensitive description framework and includes the context sensitive relations between the properties. The hierarchy of the property values is also represented in the ontology and is headed by *CalPropValue* superclass. An example of the mobile calendar entry instance is shown in FIGURE 67. A full calendar entry description can be found in Appendix C.

Concerning message transformation, there are two ways to organize it: transformation can be decentralized (local transformation on the client side) or centralized (on the message service side) (FIGURE 68a and 68b). Concerning the decentralised architecture the agent with an adapter should be located on the client (mobile device) side. This can be a challenge, as we have a limited memory and storage at our disposal (especially if we have the ontology also at the mobile end in order to decrease the amount of remote accesses and thus save money). The advantage here is that we do not need to touch the message service at all. On the other hand, concerning the centralized architecture, we would have no changes on the client side, and we should locate all the adapters with the agents on the message service side. But whether the transformations be centralized or decentralized we would not pose any challenges for the end user. All these architectures and transactions would be transparent for him/her.

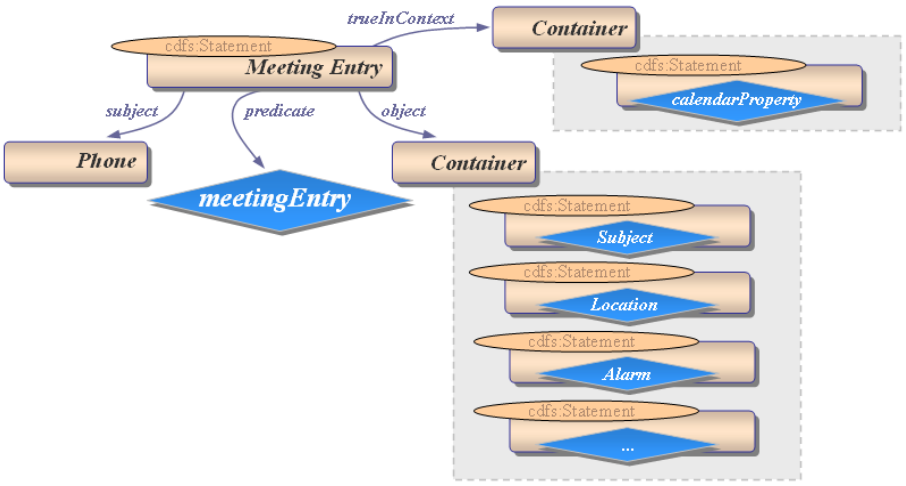


FIGURE 67 Calendar entry instance description

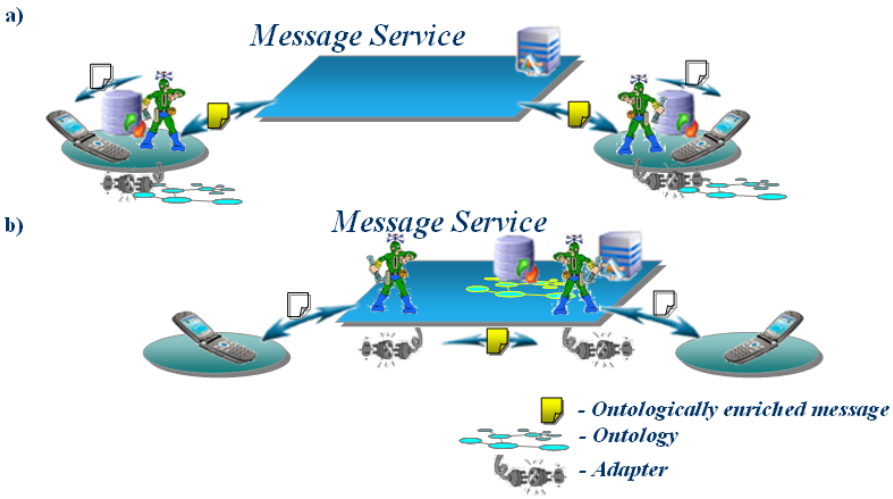


FIGURE 68 (a) Decentralized transformation architecture. (b) Centralized transformation architecture

4.1.2 Semantically enhanced multimedia data exchange

Some significant benefits provided by metadata exchange include a novel way to access multimedia data from the external world, access to the user dependent information and decreasing expenses for a multimedia data exchange. Such approach also provides the means to satiate the message with multimedia content in case of deficiency of multimedia data on the sender side. This feature

can be realized via a multimedia content semantic annotation exchange. Data duplication (e.g. sending of a data, which already exists on the recipient side) can thus be prevented and storage space saved. Such approach can, in fact, be utilized on sending a multimedia enriched message while using data which semantically fits to the content of the message but is located on the recipient side,

The data handled here can be divided into different types: original (concrete) data of a sender which can not be replaced by anything else; similar data which fits to a semantic annotation of the original data; and recipient dependent data which can be found via a personal description of the object. The hall schema of interaction is shown in FIGURE 69.

FIGURE 69 Interaction model

There are tree players in the interaction: a sender, a message service and a recipient. Each of them is represented by an agent and a storage space. Additionally, the sender and the recipient can have the representatives on the service side with a storage place or without it. The storage places can contain different types of data according to the figure. Consider a set of these use cases:

- original (concrete) sender's data transmission;
- abstract semantically described data transmission;
- recipient dependent data transmission.

Original sender's data transmission. There is a case when sender wants to send exact data (data that is not similar nor semantically the same). To send strict data does not make a sense, as it does not save any money for the sender. But if the sender is keeping, for example, a LifeBlog (storage of multimedia data on the service side), there are two possible models:

The sender keeps a copy of all data, which is stored on the mobile device, on the service side. Then he/she can create a message in an ordinary way, but the sent message will contain just a semantic annotation of the original data and will be enriched with the real multimedia data on the service side (FIGURE 70a).

The sender shifts data to the service side, but keeps the semantic annotations of the original data that require fewer resources on a mobile phone. Thus while

creating a message, he/she would use the data description module to describe the content of the message by semantic annotations. As a result of this, the original data will be added to the message on the service side, and the recipient will get a multimedia enriched message (FIGURE 70b).

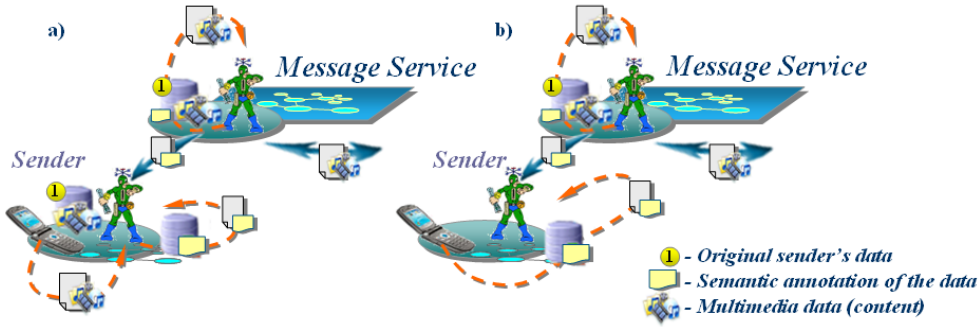


FIGURE 70 Sender's original data transmission: (a) Duplication and (b) Shifting the data to the service side

The first type of transmission necessitates a mechanism of automatic message content generation in the form of semantic data annotation to enhance the original data content. The second method needs a user friendly and handy interface for message creation via semantic annotations on the contained objects. Both of them imply a resource (data) annotation presence, which can be provided with data or be created by the user itself in a way of the common annotation approach when utilizing a common ontology.

Abstract semantically described data transmission. In this type of a messaging the sender does not direct his attempts to creating concrete data, but mainly wants to send some representative instance of the described data class. Let's consider, for instance, some situation, where the sender wants to congratulate a recipient having a birthday and would like to present a bunch of flowers. This could be accomplished by any image of a bunch of flowers or a video clip showing a presentation of flowers. The sender should just make a semantic annotation about the message content: the type of data (image or video), data content, and the kinds of flowers (if it is necessary). When a message with semantic annotations of the content objects arrives in the message service, the appropriate data may be found in the sender storage on the service side and in a large service shared multimedia data storage as well. If there is information (on the recipient service side) that the appropriate data is located on the recipient's mobile phone, the semantic message can be media enriched on the recipient side. That takes some load off the network traffic (FIGURE 71).

The sender's and the recipient's agents on the service side may provide not just information about available media content (data), but also a lot of other kind of information; for example, they can provide information about the preferences of the user they represent (kinds of flowers he/she likes, for

example, if such information has been provided by user). We can thus create message content that is not dependent directly on the recipient but on the recipient context.

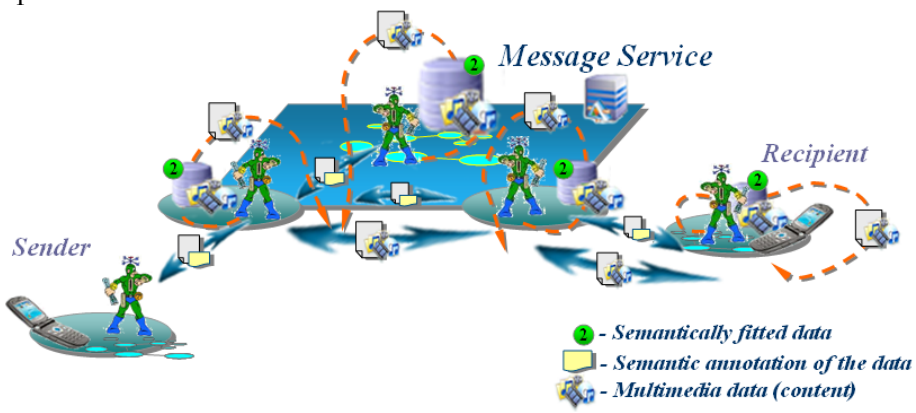


FIGURE 71 Multimedia enriching of the semantically annotated message content

Recipient dependent data transmission. This is the most interesting case, where the sender does not know or does not have specific information (data) concerning the recipient. For instance, let's assume the sender would like to congratulate his old friend for his son's birth. They have not seen each other for a long time and the sender does not know the name of the recipient's son, because he received the information concerning the birth from another friend. It would be preferable if the message had an image of the recipient's wife with a baby on her hands and the real names in the text instead of: "Dear Jon, I would like to congratulate you and Mary for the birth of your son". In this case there might be no other way to create the message content than by adding the semantic annotations about the unknown data and expecting an existence of the necessary data on the recipient side (FIGURE 72). To be sure that the recipient would get the complete message, we must specify a textual analog for that annotated data in case the specified objects were not found (such as the names in our example).

FIGURE 72 Message content enriched with a recipient related data

Undoubtedly the ideas touched upon above would bring improvements on the existing messaging framework. At least our dependence on varying multimedia data offering would be lessened. But when we consider the effect, on the senders' expenditure, of huge amounts of multimedia data transferred, then the benefit from this is not so evident. Certainly, if the content of a message covers more than one MMS message (when transferring via an existing MMS technology), then one MMS message would be enough for a message with a semantically described content. But if there is not much information that the sender wants to send, then the content can be located in a single MMS message. How we can improve the situation? Leaving aside GPRS technology utilization, the solution is a semantic message transfer via SMS messages. Keeping in mind the specific nature of the mobile environment, we would do well by elaborating a semantic description code minimization method. It might be a translator of a usual semantic description to a description in a more compact form. In other words, we need to elaborate a new RDF based language for a compact resource description (ontology and annotations creation) - Compact RDF or Mobile RDF (MRDF). Via such translation modules on the sender's and recipient's sides content can be transformed two-directional for a transfer (MRDF based content) and for processing by already existing tools in the RDF standard format.

4.2 Supportive interface for semantic messaging performance

At first, let's have an example of a message we will refer to in this section. Suppose the sender congratulates his friend Jon for the birth of his son and invites his family to the sender's new house. The content of the resulting message is shown in FIGURE 73. Initially, this message had another/different content, because the sender did not have some data and knowledge during the creation of the message. Just let us assume that Jon's son was born three weeks earlier and the sender does not know his name; imagine that the sender does not have a picture of Jon's family (with his wife and little son), but feels it would be cool to insert this picture to the message. Here we can see two relevant and coordinated features in the message: an image that somehow represents the time, and a map which shows the way to get to the sender's house.

We can consider these two images as a new kind of multimedia data – dynamic (on-the-fly) multimedia data, because those images did not exist in the mobile device that initiated the transfer. Of course they exist there in principle, but it makes more sense if such data is created on-the-fly by specific services. In this case there is a service which creates an image with a certain time representation form (clock) based on input time value, and a service which creates an image of the map with a path between the input start and destination points.

FIGURE 73 Result message content

In addition to the usual text content we have now in our message:

- *semantically fit multimedia data*: a message mood representation in the form of an image with a smile, i.e., an emotion icon (an emoticon, either textual or graphical);
- *recipient dependent data*: the name of the recipient's son. Such information can be found on the recipient side and it should be semantically annotated for purposes of this kind;
- *recipient dependent data*: a picture of the whole recipient's family (Jon, his wife and sun);
- *semantically fit / recipient dependent data*: multimedia time representation in the form of "image on-the-fly" (perhaps with a recipient's personalized view);
- *sender's original data*: an image of the sender's new house;
- *semantically fit / sender dependent / recipient dependent data*: the map with a path between the sender's and the recipient's houses.

For this additional data we must create semantic annotations during message content creation. We need such annotation for totally sender dependent (original) data not only when the real data is located on the sender's service side but also when that data is located on the sender's mobile device. Of course, in the latter case data can be found and inserted by the sender manually, but in such a case the semantic annotation should be added to the message automatically. This may facilitate the appropriate data retrieval (if the semantic search method is assisted).

4.2.1 Semantically annotated message content creation

The base for semantic annotation creation lies in the appropriate ontologies of the domain. Semantic description technology for semantic message content creation deals basically with the same issues as the content data semantic annotation technology. There is a difference though, that is, there is a description for the required resource (abstract data) as a request for semantic matching, but it is not an existing resource description. Thus the ontologies for resource (data) semantic annotation form also a base for a semantic content creation interface.

There is a set of ontologies that is required for semantic multimedia message content annotation. The first one of these ontologies is the message content object ontology. The set of content objects is larger than the set of multimedia objects, which we can insert to MMS message (image, audio and video clips). There may already be a couple of additional objects on the list: "image on-the-fly", and unknown information (text). The ontologies for object content description are significant. For example, there are ontologies such as: content type ontology (portrait, icon, logo, image, movie, animation, news, clip); object ontology (objects which can be presented on an image or video); ontology of persons (as a part of object ontology); location ontology (places); action ontology (for video clips in particular); and others. We will not dig deeply into all these ontologies, because there is a lot of related work under way or already completed. We will concentrate more on the semantic multimedia message content ontology.

As discussed in the previous chapter, three types of data are defined (the sender's original data, semantically fit data, and recipient dependent data) and should be defined in the message content ontology also. A simple example (prototype) of the semantic multimedia message content ontology can be found in Appendix D.

Now when we have an ontology for semantic message content annotation, we should specify the message format. Below we present one of the possible structures of a message. Such structure is represented by two parts (containers): data and metadata message tracks. The data track contains a plain text with the inserts of the references to the semantic data description located in the semantic (metadata) track, which is invisible for the user. To provide as complete as possible message content to the recipient, even in the absence of the appropriate data, such insert should be supplied to allow specification of a text analog which can then be shown to the recipient instead of the unavailable data. We can define any format for the insert's syntactic representation, but let us here follow the format which is shown in the message example (FIGURE 74), where the first part plays a role of a reference to the metadata and second part after the separator ("~") is the text analog.

Data track

$$Object_01$$ Hi, Jon!!! I congratulate you and Marry with $$Object_02/name~"your son"$$ birth. $$Object_03$$ I and Kati invite you for a diner next Sunday at $$Object_04~"7 pm"$$ to our new house $$Object_05$$ The address: Mooney st. 45/2 $$Object_06$$

Semantic track

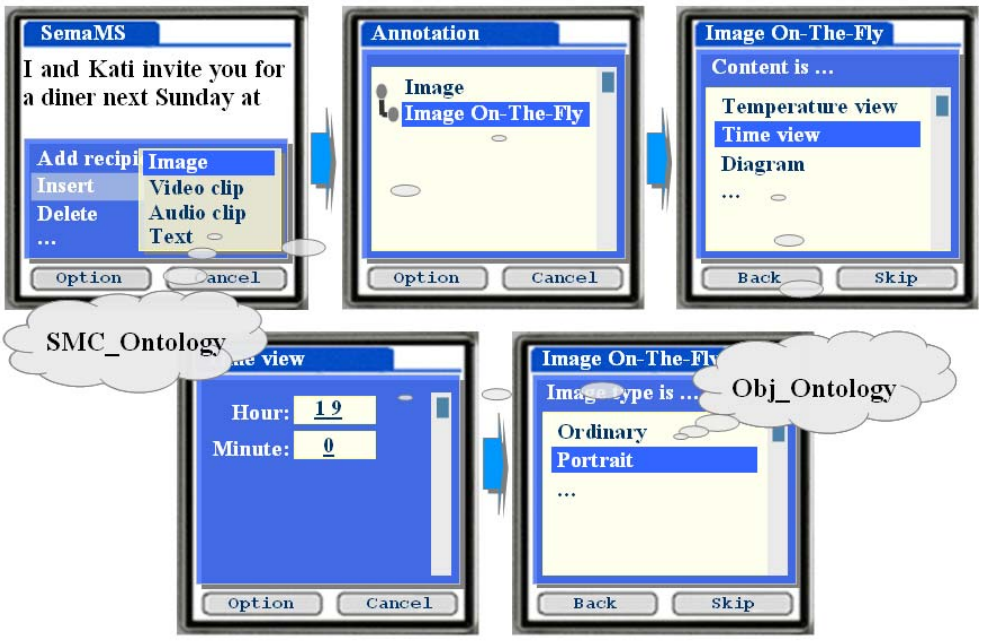
```
<?xml version='1.0' ?>
<!DOCTYPE rdf:RDF [
      <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
      <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
      <!ENTITY smc 'http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#'>
      <!ENTITY obj 'http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#'>     ]>
<rdf:RDF            xmlns:rdf="&rdf;"
                                    xmlns:rdfs="&rdfs;"
                                    xmlns:smc="&smc;"
                                    xmlns:obj="&obj;"       >

<smc:Image rdf:about="#Object_01">
   <smc:img_Type rdf:resource="&obj;Portrait"/>
   <smc:img_Content rdf:resource="#Res_01"/>
</smc:Image>
<smc:Person rdf:about="#Object_02">
   <smc:father rdf:resource="&smc;Recipient"/>
</smc:Person>
<smc:Image rdf:about="#Object_03">
   <smc:img_Type rdf:resource="&obj;Ordinary"/>
   <smc:img_Content rdf:resource="#Res_02"/>
   <smc:img_Content rdf:resource="#Object_02"/>
   <smc:img_Content rdf:resource="&smc;Recipient"/>
</smc:Imaget>
<smc:Image_OnTheFly rdf:about="#Object_04">
   <smc:img_Type rdf:resource="&obj;Portrait"/>
   <smc:img_Content rdf:resource="#Res_03"/>
</smc:Image_OnTheFly>
<smc:Image rdf:about="#Object_05">
   <smc:img_Type rdf:resource="&obj;Portrait"/>
   <smc:img_Content rdf:resource="#Res_04"/>
</smc:Image>
<smc:Image_OnTheFly rdf:about="#Object_06">
   <smc:img_Type rdf:resource="&obj;Portrait"/>
   <smc:img_Content rdf:resource="#Res_05"/>
</smc:Image_OnTheFly>

<obj:Smile rdf:about="#Res_01">
   <obj:mood rdf:resource="&obj;Excellent"/>
</obj:Smile>
<smc:Person rdf:about="#Res_02">
   <smc:name>Marry</smc:name>
   <smc:husband rdf:resource="&smc;Recipient"/>
</smc:Person>
<obj:TimeView rdf:about="#Res_03">
   <obj:hour>19</obj:hour>
   <obj:minute>0</obj:minute>
</obj:TimeView>
<obj:House rdf:about="#Res_04">
   <obj:owner rdf:resource="&smc;Sender"/>
</obj:House>
<obj:MapWithPath rdf:about="#Res_05">
   <obj:start_point rdf:resource="#Res_06"/>
   <obj:destination_point rdf:resource="#Res_07"/>
</obj:MapWithPath>
<obj:PostAddress rdf:about="#Res_06">
   <obj:resident rdf:resource="&smc;Recipient"/>
</obj:PostAddress>
<obj:PostAddress rdf:about="#Res_07">
   <obj:resident rdf:resource="&smc;Sender"/>
</obj:PostAddress>

</rdf:RDF>
```

FIGURE 74 Semantically enhanced message

A Semantic Track contains an RDF document with the references to the necessary ontologies and schemas (RDF, RDFS, CDFS). In the current example (FIGURE 74) the Semantic Track links two simple prototypes of ontologies for a message Data Track content objects' description. SMC_Ontology (Appendix D) contains the hierarchy of the classes and properties for a message content description. Second Obj_Ontology (Appendix E) specifies the objects for content annotation. These ontologies can be reached by referred links. In order to define the recipient and the sender of the current message, SMC_Ontology contains two instances of the smc:Person class (smc:Recipient and smc:Sender). In addition to the classes which describe the message multimedia objects, SMC_Ontology contains the smc:Text class for semantic annotation of textual

information. For instance, “Object_02” describes the name of the recipient's son in the current example. But, since “Object_02” is an instance of the class smc:Person, we should specify a property (“Object_02/name”) which returns a text value of the semantically matched resource.

As was mentioned, ontology is a base for user annotation interface. It means that an interface should be adaptive. We consider adaptation not just from the user personalization point of view, but from the ontology personalization point of view also. It should be a universal interface capable of utilizing the connected ontology. Such interface should be realized in a handy and user friendly form, and provide as simple as possible way for the user to annotate data. Ontology visualization is a big challenge especially for mobile terminals with a small screen size. The illustration in FIGURE 75 shows one of the possible user interfaces for semantic description of a multimedia object (“image on-the-fly” with time representation) from our message example.



Semantic Track

```
<smc:Image_OnTheFly rdf:about="#Object_04">
  <smc:img_Type rdf:resource="&obj;Portrait"/>
  <smc:img_Content rdf:resource="#Res_03"/>
</smc:Image_OnTheFly>
<obj:TimeView rdf:about="#Res_03">
  <obj:hour>19</obj:hour>
  <obj:minute>0</obj:minute>
</obj:TimeView>
```

FIGURE 75 Message content object annotation

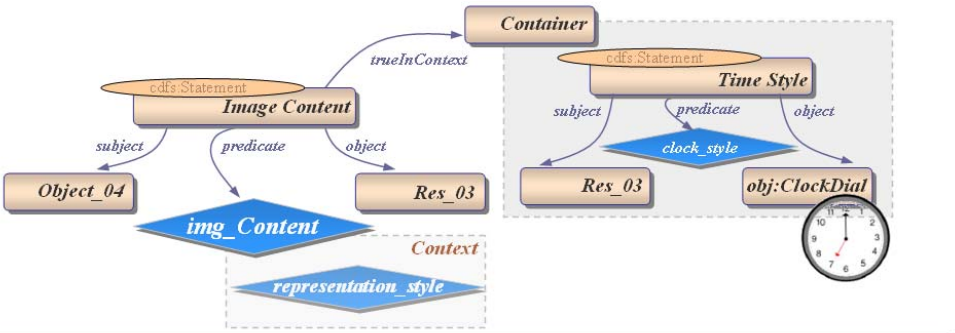
At the same time as the user creates the message Data Track, the Semantic Track is automatically created by a message creation application. In the current message creation interface there is a list representation of the ontology nodes of the same level and a supervisor questionnaire form presentation of the properties. Of course there will be trouble surfing through the lists of nodes in a huge ontology. To make it easier, a search bar may be used for fast search of an appropriate node. But, to allow such approach each ontology node (class) should be supplied with a set of synonyms. Another feature of the user personalization approach that could be utilized would order the list in the way which would emphasize the nodes used more often by the user. Such techniques can also be applied for properties representation. There are many approaches to make ontology surfing more natural and easier for heterogeneous users. One of such approaches called Ontology Personalization could be utilized.

4.2.2 Semantic personalization of a message content representation

Some of the personalization issues were considered already in the previous sub-chapter. There were interface personalization issues which concern ontology representation and surfing-through methods. But there is one more personalization aspect which concerns content representation. In our example we have two object representations referred to as "Image On-The-Fly". One of them is a time representation object (an image, which visualizes time in a "Clock Dial" style). There are many different visualization styles for time, temperature, and other measures, which can be presented by many various kinds of diagrams. What style would be the best for a user having personal preferences from the visualization point of view? Being able to specify the style of representation for the content and to describe the personal user content representation preferences makes a lot of sense.

Regarding the context concept of a context dependent resource description, as elaborated in the Context Description Framework, object representation style can be considered as a context of an object contained in the message. Simultaneously with a preferences specification in the user profile, the services (creators of the On-The-Fly Images) also should provide a specification of the representation attributes (styles) in the user's own profile. This is helpful when searching for an appropriate service. FIGURE 76 shows a message object description statement with a presentation style in the context.

Such additional specification of the object can be added not just by the sender during a message creation, but also by an agent on the service side, after which the corresponding recipient's information will be retrieved.




```
<cdfs:Statement rdf:about="#iStatement_01">
        <rdfs:label>Image Content</rdfs:label>
        <rdfs:subject rdf:resource="#Object_04"/>
        <cdfs:predicate rdf:resource="&smc;img_Content"/>
        <rdfs:object rdf:resource="#Res_03"/>
        <cdfs:trueInContext rdf:resource="#iContContainer_01"/>
</cdfs:Statement>
<cdfs:Container rdf:about="#iContContainer_01">
        <cdfs:member rdf:resource="#iStatement_02"/>
</cdfs:Container>
<cdfs:Statement rdf:about="#iStatement_02">
        <rdfs:subject rdf:resource="#Res_03"/>
        <cdfs:predicate rdf:resource="&smc;clock_style"/>
        <rdfs:object rdf:resource="&obj;ClockDial"/>
</cdfs:Statement>
```

FIGURE 76 Object representation style as a context of the object

4.3 Business opportunity within Semantic Messaging

Multimedia Messaging Service is a killer messaging service over 2.5/3rd generation networks, and will bring revolutionary changes to enterprises and consumers. There are a lot of activities that are aimed at improving MMS utilization. According to IBM China Research Laboratory (IBM_CH), which is devoted to the development on the MMS platform to help enterprises to deliver fantastic MMS based services, MMS will make a significant impact on our personal and working lives. The mobile community has great hopes that MMS will prove to be the next important development and will play a key role in halting the decline of ARPU (Average Revenue Per User). Mobile network operators are keen to introduce MMS as the flagship service on 2.5 or 3rd generation networks. Research analysts forecast that by the end of 2007 all new handsets sold in Western Europe will be equipped for MMS.

At the same time, the Semantically Enhanced Messaging approach can be applied for the development of next-generation Semantic Web enabled ICT products in the communication domain. There is already a fast growing trend of activities, which promote and show the benefit derivable from content semantic annotation. Users are expected to be interested in semantic messaging

and in new applications built on top of semantically enhanced multimedia content. Semantic messaging decreases user expenses for data transmission and opens a new possibility to access the world of information (data). New business opportunities will be opened up for multimedia content providers (multimedia creators, TV channels, etc.) as well as for service providers (internet and mobile operators), media access device producers (digital device producers) and of course for application providers (as usual when a new technology appears).

For consumers, Semantically enhanced Smart Messaging (SemaSM) as an intersection of these two fast growing approaches, delivers easy-to-use fun and utility. For other players it offers a future-proof, evolutionary migration path to profitable business (FIGURE 77).

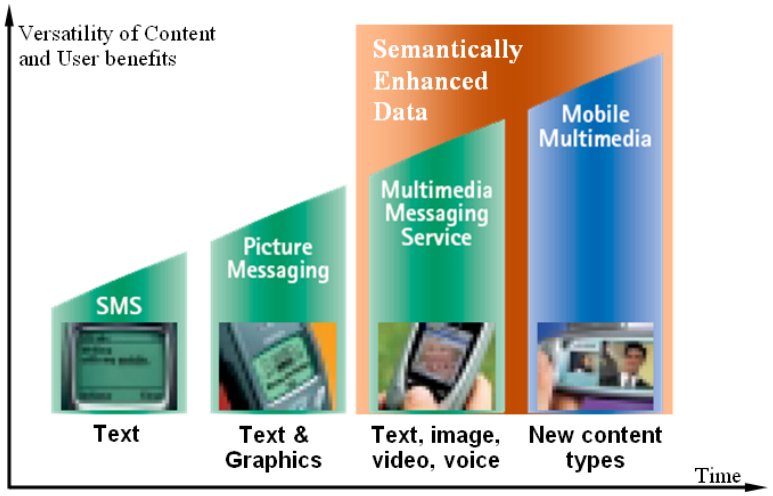


FIGURE 77 Messaging migration path

CHAPTER 4

CONCLUSIONS AND FURTHER RESEARCH

1 CONTRIBUTIONS AND ANSWERS TO THE RESEARCH QUESTIONS

At the beginning of the thesis, in the introductory chapter, we highlighted several questions that define the direction of the present research. These were divided into four subsets:

- An integrated infrastructure for distributed heterogeneous resources.
- Smart Resource of the Semantic Web, its notion and features.
- A framework for resource description in a Semantic Web based environment of Smart Resources.
- Use cases in the Smart Semantic Web based environments.

This section provides the description of the results of this thesis by answering these research questions.

Today's world is overcrowded with information, which is decentralized and non-shared (i.e. not available) for a wide community of users who might need this information. The Semantic Web approach based on creation of and using common ontologies seems to be the appropriate solution for integration and sharing useful information, knowledge, services and in a general sense – Web resources.

Resources and services are heterogeneous and require adaptation via a common ontology. As one of the ways to provide interoperability for heterogeneous resources in the Web, we considered adaptation of all the resources to a certain common Semantic Web based environment. We came up with the OntoShell adaptation approach and OntoEnvironment for Semantic Web-enabled resources (see Chapter 2, Section 1). Employing these technologies, all resources (already existing and being developed) can be transformed to semantically enabled resources for their integration in an environment, which supports mobility of the elements to enable effective integration of distributed resources. Such environment provides integration within an enterprise, as well as with its trading partners, suppliers, and customers, by offering the latest technology and open standards. This integration solution provides an opportunity to create a cost-effective, extended enterprise and get more returns on information assets from the existing ICT investments.

There is quite a lot of ongoing research in this direction, and that is why this particular question was not the main one in this thesis. Nevertheless it became the starting point for the main research. The main part of the research was to define the Smart Resource of the Semantic Web based environment, to highlight new challenging resource features, and to provide solutions for representation, description and utilization of them.

The new generation of the Web is gradually overtaking the previous one. Today's evolution leads us to the Ubiquitous (Pervasive) Computing and Internet of Things (see FIGURE 1). We cannot restrict resources to only virtual world resources, we should provide solutions for real world resources and make them a valuable part of the Web. Any object from the real world supplied with an Agent (a software agent who takes care of the user's own resource) and adapted to the Semantic level becomes a resource for the new generation Web.

While answering the next subset of the questions we came up with the idea of smart resource. In this work we have extended the usual set of Web resources to a new generation of the enhanced smart resources (*OntoSmartResources*). We considered following aspects as: context-awareness/sensitiveness of a resource, role-based and goal-driven behavior of a resource, dynamism and proactiveness of it. *OntoSmartResource* is a proactive goal-driven dynamic resource, which adequately and proactively reacts on changes within its external environment or within itself. Resource proactiveness is provided by a responsible Agent (behaviour mechanism) with an assigned role and programmable behaviour. It makes resources to become goal-driven and self-controlled entities of the environment (see Chapter 2, Section 2.2). At the same time with resource proactiveness, environment itself becomes more dynamic. In dynamic environment all the information (statements and behaviours) become context-dependent, that is why all the resource descriptions and resource-to-resource communications become context-sensitive. Unfortunately, existing resource description approaches cannot to provide proper tool for context-sensitive information describtion. As a solution for this problem we present the multilayered context-sensitive resource description approach (see Chapter 2, Section 2.1). We also considered a Resource Description Semantic Maintenance System for the OntoSmartResource. A support for a resource description semantic maintenance system opens up a new avenue for the dynamic resources to be proactive and do (semantic) maintenance of their own (metadata) content when needed.

Formerly, human was just a user of other resources in the Web, and was not regarded as a resource for the Semantic Web itself. However, human is very active and dynamic resource. A human being is an intelligent resource and can be useful for other resources (for other humans or even software applications) as a service (an expert in a specific domain) or an information source. That is why we consider a human as a potential resource, which can be semantically discovered in the Web, queried and used both by any resource of virtual world (application, service, and agent) and by real world resources (humans, smart-devices, etc). We involved a human as a resource (user/player and service) of

the global integrated Semantic Web based environment and considered the aspects related to the human representation and adaptation mechanism. We paid particular interest to emphasising the role of an ontology personalization mechanism in simplifying interactions between the players of the OntoEnvironment and to increase their collaborative performance. With respect to the human adaptation we proposed 4i FOR EYE technology – approach of intelligent information integration with context-sensitive multidimensional resource visualization. The benefit of involving humans into automated resource integration environment is evident in the emerging market for producers and providers of new applications for personal mobile devices. Connecting a real world resource (human) with its representative from a virtual world becomes much easier. Such aspects of Human-resource integration to a Semantic Web based environment in the forms of Human-resource adaptation, ontology personalization and intelligent resource visualization have been studied and presented here also (Chapter 2, Section 2.5 and Section 2.6). A use case of an environment for intelligent visualization of integrated information was presented as a solution for Human adaptation process (Chapter 3, Section 2). The first prototype of resource visualization browser has been developed recently during the first year of UBIWARE project.

Additionally, while considering the resources of the new generation Web (objects from the virtual and from the real world), we considered also such an abstract resource as a Process. In our opinion, it is a resource similar to other resources (Device, Service and Human/Expert), but does not belong to the world of physical resources. As all resources, Process has its own properties that describe the process's state, history, sub-processes and membership in an upper-process (super-process). Each process is a sequence of actions that results in the achievement of the final goal. Thus, each process is enhanced with an Agent that serves that process as a resource and actually realizes it as a behaviour engine (see Chapter 2, Section 2.4).

Recent expectations regarding the new generation of the Web strongly depend upon the success of the Semantic Web technology. The Resource Description Framework is a basis for an explicit, machine-readable representation of semantics of various Web resources and enables a framework for interoperability of future Semantic Web based applications. However previous research indicates that RDF is not suitable for describing highly dynamic and context-sensitive resources (e.g. industrial devices, processes, etc.). Therefore an appropriate extension of the existing RDF is necessary.

Here we come to the major contribution of the thesis and answer the question:

- *How to enrich the existing resource description framework with context sensitiveness and resource proactiviness?*

The whole of the third section of Chapter 2 is devoted to answering this question. To meet the demands of context-sensitive OntoEnvironment, we extended the resource description framework and elaborated common formalism to allow it (additionally to resources description) describes resource behaviour, especially since behaviour is hinged upon surrounding information

and behaviour rules also can be considered as resources. There we define the OntoSmartResource Description Framework that contains two significant parts as a logical extension of RDF for context sensitive resource description and for goal-driven behaviour (rule) based resource description. We present a new vision about statement and a property representation. We considered the contextual value as a container of RDF statements and add context dependability to the resource description via the "InContext" property (as forth component to the basic RDF triple "subject-predicate-object") and allow multilayered description of a resource. At the same time with the extension of the RDF Statement to a quadruple statement representation, we have extended the RDF Property description to a triple-based property description with the osrdfs:context property, which defines a context tolerance range for the subject property. Such an "InContext Statement" approach was elaborated during the first year of the "Smart Resource Project" and successfully applied to a dynamic and context-sensitive industrial data description during the project prototype development. We lay the foundation of the context probabilistic model description, proposed in the RDF-based rule representation model and approach for role based resource proactivity description. A probabilistic component that has been added to the model allows for not only describing the multilevel contextual dependence, but also presumes the possibility for Bayesian reasoning within the RDF model.

As discussed above, the ontology-driven approach in modeling agent behaviour is anticipated to become a powerful solution providing more benefits than the conventional model-driven approaches. RG/BDF that presents the resource goal and behaviour description part of OSRDF was designed during the second stage of the SmartResource project (Proactivity Stage). The Proactivity Layer of the Smart Resource Platform and the components of the platform that are based on the successful beneficial extensions of RDF have been presented in detail. This approach provides a semantically enhanced way for the rule and meta-rule definition. The ontology-driven approach toward modeling agent behaviour as a context-sensitive dynamic change of standardized and reusable roles, goals and actions, is anticipated to become a powerful solution which will provide some benefits compared to conventional model-driven approaches. The case of the Rule Engine execution, based on the Production System approach and possible utilization techniques of other execution engines, was presented. RG/BDF part of OSRDF has been taken as a basis for RDF-based Semantic Agent Programming Language (S-APL) that has been elaborated during the first year of UBIWARE project and became the internal language of UBIWARE platform. The most beneficial issue in the usage of the standardized data representations is the chance it gives us to operate and work cooperatively with other heterogeneous resources; it provides the opportunity for knowledge sharing and reuse. FIGURE 78 shows us three main dimentions of RDF extension in OSRDF. Thus, OntoSmartResource Description Framework is a common framework that allows context-sensitive description of the states, conditions, goals and behaviours of Smart Resources in a new generation of the Web.

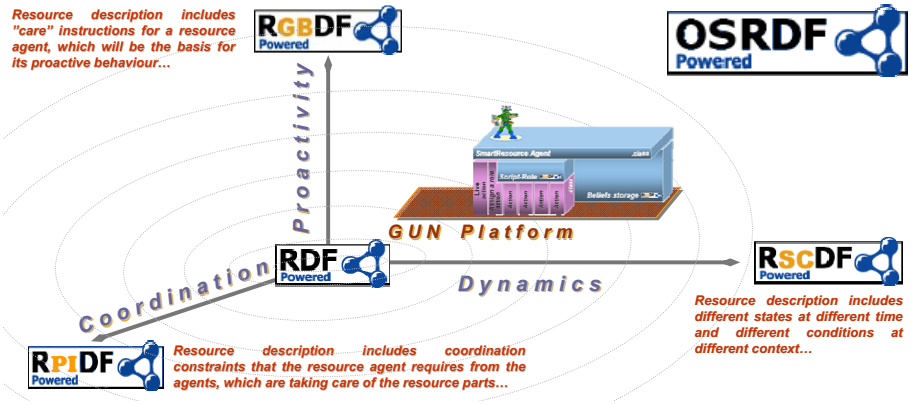


FIGURE 78 OSRDF – three-dimentional extension of RDF

To show the value of the benefits of the Semantic Web based environments we have finalized the thesis with several use cases from the OntoEnvironment. Among these there are cases of automated industrial maintenance with knowledge transfer from an expert to an artificial intelligence, an environment for intelligent visualization of integrated information, semantically enhanced multimedia browsing and semantically enhanced smart messaging.

2 LIMITATIONS AND FURTHER RESEARCH

2.1 Limitations

In this section we are going to consider some limitations and weaker sides of the presented approach. In spite of the fact that the proposed OSRDF itself can be considered as a quite reliable framework, because its elaboration is based on standardized RDF and follows its basic principles, we still need to design and develop software tools for storing, querying, reasoning and manipulating data in this format. Some attempts to store, query and utilize quadruple statement representation were made in the first year prototype of the SmartResource project. In this prototype, the fact that OSRDF is based on RDF allowed us to use existing RDF supportive tools. RDF-based Semantic Agent Programming Language (S-APL), elaborated during the first year of UBIWARE project, springs from RGBDF part and supports the main idea. But we still have to elaborate a complete set of tools to show the potential and benefits of the present framework. Thus, after successful testing of the tools and framework in practice, we may consider the possibility to standardize this enriched framework.

Clearly, in this work we concentrated on an extension of RDF. Quite naturally the question "Why not OWL?" will come up. Obviously, OWL facilitates greater machine interpretability of web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing an additional vocabulary along with a formal semantics. But in this work, we have extended not just the vocabulary, but also the semantics of the RDF statement and property. We started from RDF as a basis for other description languages. Application of this approach towards OWL can be considered as the next logical step in the continuation of this work.

Regarding Human-Resource adaptation and particularly, the basis for the proposed 4i (FOR EYE) technology - Context-sensitive Multidimensional Resource Visualization - we again emphasize that this approach requires context ontology that will provide knowledge about visualization contexts for

the resources related to the context resource properties. Such ontology was not a part of the goal of this thesis and is not presented in this work, but certain requirements for such ontology development have been presented. Also, initial prototype of the SmartInterface based of this technology is developed during the first year of UBIWARE project.

2.2 Further research

We can identify several directions for the continuation of our work. Since the present research is a theoretical solution (in general) with partial implementation, we are positioning our further work as a practical development to prove and apply the achieved results.

Regarding the context-sensitive resources description framework as a logical continuation of the approach, we are considering an elaboration of the context-dependent query language and quadruple storing. We are going to elaborate a new conceptual model for a resource descriptions storage. Another significant challenge will be utilising nested and probabilistic context for advanced reasoning based on the CDF model.

Nowadays, research towards the semantic based knowledge and heterogeneous application integration is very popular in the ICT domain. And it is also the main part of the research and development of IOG in the recently started UBIWARE project. But, in our opinion, issues related to Human-resource adaptation are becoming more and more important and challenging. That is why we think to aim our further research at the development of a context-sensitive Human-resource adaptation framework, elaboration of Human-resource related domain ontology and advanced multidimensional resource visualization techniques. The first steps towards this aim have been taken and further research will be continued in the UBIWARE project (workpackage WP5:" Smart Interfaces: Context-aware GUI for Integrated Data (4i technology)").

REFERENCES

Aduna, B., (2007). User Guide for Sesame 2.0-beta6. OpenRDF.org - a community site to support the development of Sesame. URL: http://www.openrdf.org/doc/sesame2/2.0-beta6/users/userguide.html.

Ancona, D. and Mascardi, V., (2004). Coo-BDI: Extending the BDI Model with Cooperativity, In Proc. of the First Declarative Agent Languages and Technologies Workshop, J. A. Leite, A. Omicini, L. Sterling and P. Torroni (Eds.), Springer LNAI 2990, 2004, pp. 109-134.

Ancona, D., Mascardi, V., Hübner, J. F. and Bordini R.H., (2004). Coo-AgentSpeak: Cooperation in AgentSpeak through Plan Exchange, In Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2004), N. R. Jennings and C. Sierra and L. Sonenberg and M. Tambe (Eds.), pp. 698-705, ACM press, 2004.

Ankolekar, A., Burstein, M., Hobbs, J.R., Lassila, O., Martin, D.L., McDermott, D., McIlraith, S.A., Narayanan, S., Paolucci, M., Payne, T.R. and Sycara, K., (2002). DAML-S: Web Service Description for the Semantic Web, URL: http://www-2.cs.cmu.edu/ ~terryp/Pubs/ ISWC2002-DAMLS.pdf.

A-RDF-QL, (2006). Algae RDF Query Language, http://www.w3.org/2004/05/06-Algae (visited 31.10.2006).

Berner-Lee, T., (2006). Putting the Web back in Semantic Web. In: Keynote Presentation, 4th International Semantic Web Conference, Available at: http://www.w3.org/2005/Talks/1110-iswc-tbl/ Accessed on May 22, 2006.

Berners-Lee, T., Hendler, J. and Lassila, O., (2001). The Semantic Web. Scientific American 284(5), 34-43.

Bertini, M., Cucchiara, R., Bimbo, A. and Torniai, C., (2005). Ontologies Enriched with Visual Information for Video Annotation, In: *Multimedia and the Semantic Web workshop as part of the 2nd European Semantic Web Conference (ESWC2005-MSW),* Heraklion, Crete, May-June, 2005.

Bouquet, P., Giunchiglia, F., van Harmelen, F., Serafini, L. and Stuckenschmidt, H., (2004). Contextualizing Ontologies. In: Web Semantics: Science, Services and Agents on the World Wide Web, 1: 4, 321–419.

BPEL, (2004). Business Process Execution Language for Web Services, http://www-128.ibm.com/developerworks/library/specification/ws-bpel/.

Braubach, L., Pokahr, A. and Lamersdorf, W., (2004). Jadex: A Short Overview, in: Main Conference Net.ObjectDays 2004, AgentExpo.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J., (2004). Tropos: An agentoriented software development methodology. Autonomous Agents and Multi-Agent Systems 8(3), 203–236.

Buchingae, (2005). Buchingae - A Rule Language for The Web, avail able at http://mknows.etri.re.kr/mknowswikidata/BossamRuleEngine/attachments/buchingae-rule-language.html, 2005 (visited 31.10.2006).

Cervenka, R., Trencansky, I., Calisti, M. and Greenwood, D., (2005). AML: Agent Modelling Language. Toward Industry-Grade Agent-Based Modelling. Springer LNCS, Vol. 3382/2005: Agent-Oriented Software Engineering V: 5th International Workshop, NY, USA.

Chella, A., Cossentino, A. and Sabatucci, L., (2004). Tools and patterns in designing multi-agent systems with passi. In 6th WSEAS Int.Conf. on Telecommunications and Informatics (TELE-INFO 04), Cancun, Mexico, May 2004.

Clabby, J., (2002). Web Services Executive Summary, URL: http://www-106.ibm.com/ developerworks/webservices/library/ws-gotcha/?dwzone= webservices.

Collier, R., Ross, R., O'Hare, G., (2005). Realising reusable agent behaviours with ALPHA. In: Eymann, T., Kl¨ugl, F., Lamersdorf,W., Klusch, M., Huhns,M.N. (eds.)MATES 2005. LNCS (LNAI), vol. 3550, pp. 210–215. Springer, Heidelberg.

Curbera, F., Dufler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S., (2002). Unraveling the Web Services Web: An introduction to SOAP, WSDL and UDDI, Internet computing.

Dastani, M., Jonker, C.M., and Treur, J., (2004a). A Requirement Specification Language for Configuration Dynamics of Multi-Agent Systems. Intern.l Journal of Intelligent Systems, vol. 19, 2004, pp. 277-300.

Dastani, M., van Riemsdijk, B., Dignum, F., Meyer, J.J., (2004b) A programming language for cognitive agents: Goal directed 3APL. In: Dastani, M., Dix, J., El Fallah-Seghrouchni, A. (eds.) PROMAS 2003. LNCS (LNAI), vol. 3067, pp. 111–130. Springer, Heidelberg.

Davies, J., Fensel, D., Harmelen, F. and (eds.), (2002). Towards the Semantic Web, Wiley.

Degler, D. and Battle, L., (2000). Knowledge Management in Pursuit of Performance: the Challenge of Context. Performance Improvement (EPSS Special Edition). ISPI, 39(6), July 2000. As viewed online, www.ipgems.com/writing/kmcontext.htm.

Dey, A., Abowd, G. and Salber, D., (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Human-Computer Interaction, 16(2-4).

Djuric, D., Gaševic, D. and Damjanovic, V., (2004). AIR A Platform for Intelligent Systems. In Proceedings of the 1st IFIP Conf. on AI Applications and Innovations, Toulouse, France, August 2004.

Dourish, P., (2004). What we talk about when we talk about context. Personal Ubiquitous Comput. 8, 1 (Feb. 2004), 19-30. DOI= http://dx.doi.org/10.1007/s00779-003-0253-8.

Duží, M. and Heimburger, A., (2006). Web Ontology Languages: Theory and practice, will they ever meet?. In Information Modelling and Knowledge Bases XVII. Ed. Y. Kiyoki, J. Hanno, H. Jaakkola, H. Kangassalo, Amsterdam: IOS Press , 2006, Vol. XVII, 20-37, ISBN 1-58603-591-6.

Fensel, D., (2003). Semantic Web Services: A Communication Infrastructure for eWork and eCommerce. In: Proceedings of the ICWE 2003, Oviedo, Spain, LNCS 2722, Springer.

FieldSence, (2002). FieldSence – Intelligent Products for Field Systems Lifecycle Management, Metso Automation Web Cite, URL: http://www.metsoautomation.com.

FIPA, (2001). FIPA Interaction Protocol Library Specification Specification, FIPA00025. URL: http://www.fipa.org/specs/fipa00025/.

Geurts, J., Ossenbruggen, J. and Hardman, L., (2005). Requirements for practical multimedia annotation, In: *Multimedia and the Semantic Web workshop as part of the 2nd European Semantic Web Conference (ESWC2005-MSW),* Heraklion, Crete, May-June, 2005.

Griss, M., Fonseca, S., Cowan, D. and Kessler, D., (2002). SmartAgent: Extending the JADE agent behaviour model, HPL 2002-18, January 2002.

Guha, R., (1995). Contexts: A Formalization and Some Applications. PhD Thesis. Computer Science Dep., Standford University.

Hayes, P. and McBride, B., (2004). RDF Semantics, W3C Recommendation.

IBM_CH. IBM China Research Laboratory, URL:http://www.research.ibm.com/beijing/updates/mms.html.

IOG, (2003). Industrial Ontologies Group "Semantic Web Enabled Network of Maintenance Services for Smart Devices", Tekes project proposal, Agora Center, University of Jyväskylä, URL: http://www.cs.jyu.fi/ai/ Metso_Maintenance.ppt.

iPlanet, (2002). iPlanet Application Server Enterprise Connector for CICS, URL: http://docs-pdf.sun.com/806-5504/806-5504.pdf.

ITU, 2005. "ITU Internet Reports 2005: The Internet of Things. Executive Summary", International Telecommunication Union (ITU). Geneva, November 2005. URL: http://www.itu.int/osg/spu/publications/internetofthings/InternetofThings_summary.pdf.

JADE, 2004. Java Agent DEvelopment Framework, http://jade.tilab.com/.

Jansen, B., (1993). Context: A real problem for large and shareable knowledge bases, Building/ Sharing Very Large Knowledge Bases (KBKS'93), Tokyo.

JavaLIB. Specification of Java library for programming agent behaviours, http://jade.tilab.com/doc/api/jade/core/behaviours/Behaviour.html.

Jennings, N., (2000). On agent-based software engineering. Artificial Intelligence 117(2), 277–296.

Jennings, N., (2001). An agent-based approach for building complex software systems. Communications of the ACM 44, 4 (2001) 35–41.

Joseph, P. et al., (2000). The American Heritage Dictionary of the English Language, Pickett, Fourth Edition 2000, Houghton Mifflin Company.

Järvinen, P., (2004). On research methods, Tampere:Opinpajan Kirja.

Kaikova, H., Khriyenko, O., Kononenko, O., Terziyan, V. and Zharko, A., (2004). Proactive Self-Maintained Resources in Semantic Web, Eastern-European Journal of Enterprise Technologies, Vol. 2, No. 1, ISSN: 1729-3774, Kharkov, Ukraine, pp. 4-16.

Katasonov, A. and Terziyan, V., (2007). SmartResource Platform and Semantic Agent Programming Language (S-APL), In: P. Petta et al. (Eds.), Proceedings of the 5-th German Conference on Multi-Agent System Technologies (MATES'07), 24-26 September, 2007, Leipzig, Germany, Springer, LNAI 4687 pp. 25-36.

Kaykova, O., Khriyenko, O., Klochko, O., Kononenko, O., Taranov, A., Terziyan, V. and Zharko, V., (2004). Semantic Search Facilitator: Concept and Current State of Development*, In: InBCT Tekes Project Report, Chapter 3.1.3 : "Industrial Ontologies and Semantic Web"*, Agora Center, University of Jyväskylä, January – May 2004. URL: http://www.cs.jyu.fi/ai/OntoGroup/InBCT_May_2004.html.

Kaykova, O., Khriyenko, O., Kovtun, D., Naumenko, A., Terziyan, V. and Zharko, A., (2005a). General Adaptation Framework: Enabling Interoperability for Industrial Web Resources, In: International Journal on Semantic Web and Information Systems, Vol. 1, No. 3, 2005, pp. 30-62.

Kaykova, O., Khriyenko, O., Naumenko, A., Terziyan, V. and Zharko A., (2005b). RSCDF: A Dynamic and Context Sensitive Metadata Description Framework for Industrial Resources, In: Eastern-European Journal of Enterprise Technologies, Vol.3, No.3, June, ISSN: 1729-3774, 30 pp.

Kaykova, O., Khriyenko, O., Terziyan, V. and Zharko A., (2005c). RGBDF: Resource Goal and Behaviour Description Framework, In: M. Bramer and V. Terziyan (Eds.): Industrial Applications of Semantic Web, Proceedings of the 1-st International IFIP/WG 12.5 Working Conference IASW-2005, August 25-27, Jyvaskyla, Finland, Springer, IFIP, Vol.188, pp. 83-99.

Kaykova, O., Khriyenko, O., Kovtun, D., Naumenko, A., Terziyan, V., Zharko, A., (2007). Challenges of General Adaptation Framework for Industrial Semantic Web. In: Amit Sheth and Miltiadis Lytras (eds.), Semantic Web-Based Information Systems: State-of-the-Art Applications, CyberTech Publishing, pp. 61-97 (Chapter III).

Khriyenko, O., Kononenko, O. and Terziyan, V., (2004). OntoEnvironment: An Integration Infrastructure for Distributed Heterogeneous Resources. In: IASTED International Conference on Parallel and Distributed Computing and Networks (PDCN 2004), Innsbruck, Austria.

Khriyenko, O. and Terziyan, V., (2004). OntoSmartResource: An Industrial Resource Generation in Semantic Web. In: Proceedings of the 2nd IEEE International Conference on Industrial Informatics (INDIN'04), Berlin, Germany.

Khriyenko, O., (2005). SemaSM: Semantically enhanced Smart Message. In: Eastern-European Journal of Enterprise Technologies,Vol. 1, No. 13, 2005, ISSN: 1729-3774.

Khriyenko, O. and Terziyan, V., (2006). A Framework for Context-Sensitive Metadata Description. In: International Journal of Metadata, Semantics and Ontologies, Inderscience Publishers, ISSN 1744-2621, Vol. 1, No. 2, pp. 154-164.

Khriyenko, O., (2007a). Coordination of the Distributed Proactive Smart Resource. In: Proceedings of the IASTED International Conference on

Parallel and Distributed Computing and Networks (PDCN 2007) as part of the 25th IASTED International Multi-Conference on APPLIED INFORMATICS, Innsbruck, 7pp.

Khriyenko, O., (2007b). 4I (FOR EYE) Technology: Intelligent Interface for Integrated Information. In: 9th International Conference on Enterprise Information Systems (ICEIS-2007), Funchal, Madeira – Portugal.

Khriyenko, O., (2007c). 4I (FOR EYE) Multimedia: Intelligent semantically enhanced and context-aware multimedia browsing. In: Proceedings of the International Conference on Signal Processing and Multimedia Applications (SIGMAP-2007), Barcelona, Spain, 7pp.

Khriyenko, O., (2007d). Context-sensitive Multidimensional Resource Visualization. In: Proceedings of the 7th IASTED International Conference on Visualization, Imaging, and Image Processing (VIIP 2007), Palma de Mallorca, Spain, 7pp.

Klyne, G. and Carroll, J.J., (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, 10 February 2004, http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/. Latest version available at http://www.w3.org/TR/rdf-concepts/.

Laleci, G., Kabak, Y., Dogac, A., Cingil, I., Kirbas, S., Yildiz, A., Sinir, S., Ozdikis, O. and Ozturk, O., (2004). A Platform for Agent Behaviour Design and Multi Agent Orchestration. AOSE 2004: pp. 205-220.

Lassila, O., (2002). Semantic Web. Nokia Mobile Internet Technical Architecture, Vol. 1. Technologies and Standardization, Part 3.3. Finland: Edita Publishing, 127-135.

MacGregor, R. and Ko, I.Y., (2003). Representing Contextualized Data using Semantic Web Tools, In Proceedings of the 1st International Workshop on Practical and Scalable Semantic Systems, ISWC 2003, Sanibal Island, Florida, USA.

Marcos, G., Smithers, G., Jiménez, G., Posada, G., Stork, G., Pianciamore, M., Castro, M., Marca, M., Mauri, M., Selvini, M., Sevilmis, N., Thelen, N. and Zechino, V., (2005). A Semantic Web based approach to multimedia retrieval, *In: Fourth International Workshop on Content-Based Multimedia Indexing (CBMI 2005)*, Riga, Latvia, 21-23 June, 2005.

Mascardi, V., Martelli, M., Zini, F. and Degl'Innocenti R., (2004). CaseLP: a Prototyping Environment for Heterogeneous Multi-Agent Systems. Journal of Autonomous Agents and Multi-Agent Systems, 2004.

Mazzocchi S., (2004). It's All about Graphs, White Paper, February 11, 2004, available at: http://www.betaversion.org/~stefano/linotype/news/46.

McCarthy, J. and Buvac, S., (1997). Formalizing Context (Expanded Notes), Computer Natural Language.

Naeve, A., (2005). The Human Semantic Web: shifting from knowledge push to knowledge pull, In: *International Conference on Computational Science (ICCS 2005)*, Atlanta, USA, 22-25 May, 2005.

Naumenko, A., Nikitin, S., Terziyan, V. and Zharko A., (2005). Strategic Industrial Alliances in Paper Industry: XML- vs. Ontology-Based

Integration Platforms, In: The Learning Organization, An International Journal, Emerald Publishers, Vol. 12., No. 5.

Naumenko, A., (2007). Semantics-Based Access Control in Business Networks, PhD. Thesis, Department of Mathematical Information Technology, University of Jyväskylä, Vol. 78, June 2007.

Nikitin, S., Terziyan, V. and Pyotsia, J., (2007). Data Integration Solution for Paper Industry - A Semantic Storing, Browsing and Annotation Mechanism for Online Fault Data, In: Proceedings of the 4th International Conference on Informatics in Control, Automation and Robotics (ICINCO), May 9-12, 2007, Angers, France, INSTICC Press, ISBN: 978-972-8865-87-0, pp. 191-194.

Nixon, L., (2006). Multimedia, Web 2.0 and the Semantic Web: a strategy for synergy, In: *First International Workshop on Semantic Web Annotations for Multimedia (SWAMM), as a part of the 15th World Wide Web Conference*, Edinburgh, Scotland, 22-26 May, 2006.

OWL, (2004). Web Ontology Language Overview. W3C Recommendation, 10 February 2004, available at http://www.w3.org/TR/owl-features (visited 04.12.2006).

OWL-S, (2003). OWL-S: Semantic Markup for Web-Services. The OWL Services Coalition, URL: http://www.daml.org/services/owl-s/1.0/owl-s.html.

Paolucci, M., Kawamura, T., Payne, T. R., Sycara, K., (2002). Importing the Semantic Web in UDDI. URL:http://www-2.cs.cmu.edu/~softagents/papers /Essw.pdf.

Peeters, G., (2003). A large set of audio features for sound description (similarity and classification) in the CUIDADO project, In: *CUIDADO project report*, 2003. URL: http://www.ircam.fr/anasyn/peeters/ ARTICLES/Peeters_2003_cuidadoaudiofeatures.pdf.

Pfeiffer, S., Parker, C. and Pang, A., (2003). The Annodex annotation and indexing format for time continuous bits reams, *Version 2.0 (work in progress).* http://www.ietf.org/internet-drafts/draft-pfeiffer-annodex-01.txt, December 2003.

Pirttioja, T., Seilonen, I., Appelqvist, P., Halme, A. and Koskinen, K., (2004). Agent-based architecture for information handling in automation systems, 6th IFIP International Conference on Information Technology for Balanced Automation Systems in Manufacturing and Services (BASYS 2004), Vienna, Austria.

Rao, A.S. and Georgeff, M.P., (1995). BDI Agents: From Theory to Practice, in Proc. of the 1st International Conference on Multi-Agent Systems (ICMAS-95), San Francisco, USA, June, 1995.

Rao, A.S., (1996). AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language. In Proc. MAAMAW-96. Springer Verlag, LNAI 1038, pp. 42–55.

RDQL, (2004). RDQL - A Query Language for RDF. W3C Member Submission, 9 January 2004, available at http://www.w3.org/Submission/RDQL (visited 31.10.06).

RMi, (2006). The Rule Markup Initiative, http://www.ruleml.org (visited 31.10.2006).

Ryan, N., Pascoe, J., and Morse, D., (1997). Enhanced Reality Fieldwork: The context-aware archeological assistant. In Gaffney, Leusen, and Exxon (eds), Computer Applications in Archeology.

Schilit, B., Adams, N. and Want, R., (1994). Context-Aware Computing Applications. Proc. IEEE Workshop on Mobile Computing Systems and Applications (Santa Cruz, CA). New York: IEEE.

Schilit, B. and Theimer, M., (1994). Disseminating active map information to mobile hosts. IEEE Network, 8(5), 22–32.

Seilonen, I., Koskinen, K., Pirttioja, T., Appelqvist, P. and Halme, A., (2003). Agent-based approach to enhanced flexibility in process automation systems, 3rd International Symposium on Open Control Systems, Helsinki, Finland, September 9-10, 2003.

Semantic Web, (2001). URL: http://www.w3.org/2001/sw/.

Shadbolt, N., Hall, W. and Berners-Lee, T., (2006). The Semantic Web Revisited. IEEE Intelligent Systems 21(3), 96-101.

Shoham, Y. (1993). Agent-oriented programming. Artificial Intelligence 60(1), 51–92.

Sicilia, M.A. and García-Barriocanal, E., (2004). Notes on Extending Terminological Software Frameworks with Fuzziness through Aspect Oriented Design. WSEAS Transactions on Systems 3(3), pp. 1083-1089.

SPARQL, (2006a). SPARQL Protocol for RDF. W3C Candidate Recommendation, 6 April 2006, available at http://www.w3.org/TR/rdf-sparql-protocol (visited 01.12.2006).

SPARQL, (2006b). SPARQL Query Language for RDF. W3C Working Draft, 4 October 2006, available at http://www.w3.org/TR/rdf-sparql-query (visited 01.12.2006).

SPARQL, (2006c). SPARQL Query Results XML Format. W3C Candidate Recommendation, 6 April 2006, http://www.w3.org/TR/rdf-sparql-XMLres (visited 01.12.2006).

Stollberg, M., Keller, U., Zugmann, P. and Herzog, R., (2004). Semantic Web Fred - Agent Cooperation on the Semantic Web, 3rd International Semantic Web Conference, Hiroshima, Japan, 7 - 11 November 2004.

Straccia, U., (2001). Reasoning within fuzzy description logics. \emph{J. Artificial Intelligence Research}, {\bf 14}: 137--166.

Terziyan, V., (2003). Semantic Web Services for Smart Devices in a “Global Understanding Environment”, In: R. Meersman and Z. Tari (eds), On the Move to Meaningful Internet Systems 2003: OTM 2003 Workshops, Lecture Notes in Computer Science, Vol. 2889, Springer-Verlag, pp.279-291.

Terziyan, V. and Zharko, A., (2003). Semantic Web and Peer-to-Peer: Integration and Interoperability in Industry, In: International Journal of Computers, Systems and Signals, IAAMSAD, ISSN 1608-5655, Vol. 4, No. 2, 2003, pp. 33-46.

Terziyan, V., (2004). Semantic Web Services for Smart Devices Based on Mobile Agents, In: J. Marx Gomez (ed.), Intelligent Mobile Agents in Peer-to-Peer Networks: Proceedings of the Forth International ICSC Symposium on Engineering Intelligent Systems (EIS-2004), Madeira, Portugal, February 29 – March 2, ICSC Academic Press, Canada, ISBN: 3-906454-35-5, pp. 10-16.

Terziyan, V. and Khriyenko, O., (2004). Mobile Agent-Based Web Service Components in Semantic Web. In: Eastern-European Journal of Enterprise Technologies, Vol. 2, No. 2.

Thevenin, D. and Coutaz, J., (1999). Plasticity of User Interfaces: Framework and Research Agenda, *In Proceedings of Interact'99*, vol. 1, Edinburgh: IFIP, IOS Press, 1999, pp. 110-117.

Torres da Silva, V., Choren, R. and Pereira de Lucena, C.J., (2004). A UML Based Approach for Modelling and Implementing Multi-Agent Systems. AAMAS 2004: 914-921.

Tolle, K. and Wleklinski, F., (2004). Trust and context using the RDF-Source related Storage System (RDF-S3) and easy RQL (eRQL), URL:http://www.eworks.de/research/2004/ 07/RDFS3AndErql/RDF-S3_and_eRQL.pdf.

Tresp, C.B. end Molitor, R., (1998). A Description Logic for Vague Knowledge. In: \emph{Proceedings of the 13th biennial European Conference on Artificial Intelligence} ({ECAI}'98), J.~Wiley and Sons, Brighton, UK 361-365.

Turner, R. and Roeck, A., (1988). Understanding Cognitive Science, ed. Michael F. McTear, (University of Ulster at Jordanstown) Chichester: Ellis Horwood, 1988, 264 pp. (Ellis Horwood series in cognitive science) ISBN 0-7458-0161-7 (hb).

Vazquez-Salceda, J., Dignum, V., Dignum, F., (2005). Organizing multiagent systems. Autonomous Agents and Multi-Agent Systems 11(3), 307–360.

W3C, (2001). W3C – World Wide Web Consortium, http://www.w3.org/2001/sw, (visited 02.12.2006).

Web3.0. In: Wikipedia - a multilingual, web-based, free content encyclopedia. URL: http://en.wikipedia.org/wiki/Web_3.0.

WebServices. URL: http://www.webservices.org.

Wooldridge, M., Jennings, N., Kinny, D., (2000). The Gaia methodology for agent-oriented analysis and design. Autonomous Agents and Multi-Agent Systems 3(3), 285–312.

XML, (1998). Extensible Markup Language (XML) 1.0. W3C Recommendation 10 February 1998, available at http://www.w3.org/TR/1998/REC-xml-19980210 (visited 01.12.2006).

XPath, (1999). XML Path Language (XPath). Version 1.0. W3C Recommendation, 16 November 1999, available at http://www.w3.org/TR/xpath (visited 02.12.2006).

XQuery, (2006). XQuery 1.0: An XML Query Language. W3C Proposed Recommendation, 1 November 2006, available at http://www.w3.org/TR/xquery (visited 02.12.2006).

Yuxin, M., Zhaohui, M., Zhao, X., Huajun, C. and Yumeng, Y., (2005). Interactive Semantic-Based Visualization Environment for Traditional Chinese Medicine Information. *In: Web Technologies Research and Development - APWeb 2005, 7th Asia-Pacific Web Conference*, Shanghai, China, March 29 - April 1, 2005, Springer, volume 3399/2005, ISBN 978-3-540-25207-8, pp. 950-959.

Zhao, L., Mehandjiev, N. and Macaulay, L., (2004). Agent Roles and patterns for supporting Dynamic behaviour of Web Service Applications. In: Proceedings of AAMAS04 Workshop on Web Services and Agent-Based Engineering (WSABE).

@Semantics S.R.L. URL: http://www.asemantics.com/.

APPENDICES

APPENDIX A

OSRDF-S:

OSRDF Container	
osrdfs:Container	The osrdfs:Container class is the class of OSRDF Statement containers (which contain just instances of osrdfs:Statement class). It is an instance of rdfs:Class and a subclass of rdfs:Container.
osrdfs:ContextContainer	The osrdfs:ContextContainer class is the class of OSRDF Property containers (which contain just instances of osrdfs:Property class). It is an instance of rdfs:Class and a subclass of rdfs:Container.
osrdfs:OCC_Container	The osrdfs:OOC_Container class is the class of OSRDF Ordered Context containers. Container contains the instances of osrdfs:OrderContextContainer class. It is an instance of rdfs:Class and a subclass of rdfs:Container.
osrdfs:PropSignContainer	The osrdfs:PropSignContainer class is the class of OSRDF Property Significance containers (which contain just instances of osrdfs:PropertySignificance class). It is an instance of rdfs:Class and a subclass of rdfs:Container.
osrdfs:OrderContextContainer	The osrdfs:OrderContextContainer class is the class of OSRDF contextual statement containers (which contain instances of osrdfs:Statement class). It is an instance of rdfs:Class and a subclass of osrdfs:Container.
osrdfs:NF_Container	The osrdfs:NF_Container class is the class of OSRDF Non-Fact Statement containers (which contain just instances of osrdfs:NF_Statemant class). It is an instance of rdfs:Class and a subclass of osrdfs:Container.
osrdfs:GoalContainer	The osrdfs:GoalContainer class is the class of OSRDF Goal containers (which contain just instances of osrdfs:GoalStatement class). It is an instance of rdfs:Class and a subclass of osrdfs:NF_Container.
osrdfs:BehaviourContainer	The osrdfs:BehaviourContainer class is the class of OSRDF

	Behaviour containers (which contain just instances of osrdfs:BehaviourStatement class). It is an instance of rdfs:Class and a subclass of osrdfs:NF_Container.
osrdfs:member	osrdfs:member is an instance of rdf:Property and a subproperty of rdfs:member property, it is used to state the member of a OSRDF Statement container. A triple of the form: *C osrdfs:member S* states that C is an instance of osrdfs:Container and that the member of C is S. The rdfs:domain of osrdfs:member is osrdfs:Container. The rdfs:range of osrdfs:member is osrdfs:Statement.
osrdfs:nfMember	osrdfs:nfMember is an instance of rdf:Property and a subproperty of osrdfs:member property, it is used to state the member of a OSRDF Non-Fact Statement container. A triple of the form: *C osrdfs:nfMember S* states that C is an instance of osrdfs:NF_Container and that the member of C is S. The rdfs:domain of osrdfs:nfMember is osrdfs:NF_Container. The rdfs:range of osrdfs:nfMember is osrdfs:NF_Statement.
osrdfs:gMember	osrdfs:gMember is an instance of rdf:Property and a subproperty of osrdfs:nfMember property, it is used to state the member of a OSRDF Goal Statement container. A triple of the form: *C osrdfs:gMember S* states that C is an instance of osrdfs:GoalContainer and that the member of C is S. The rdfs:domain of osrdfs:gMember is osrdfs:GoalContainer. The rdfs:range of osrdfs:gMember is osrdfs:GoalStatement.
osrdfs:bMember	osrdfs:bMember is an instance of rdf:Property and a subproperty of osrdfs:nfMember property, it is used to state the member of a OSRDF Behaviour Statement container. A triple of the form: *C osrdfs:bMember S* states that C is an instance of osrdfs:BehaviourContainer and that the member of C is S. The rdfs:domain of osrdfs:bMember is osrdfs:BehaviourContainer. The rdfs:range of osrdfs:bMember is osrdfs:BehaviourStatement.
osrdfs:o3cMember	osrdfs:occMember is an instance of rdf:Property and a subproperty of rdfs:member property, it is used to state the member of a OSRDF Ordered Context container. A triple of the form: *C osrdfs:occMember C1* states that C is an instance of osrdfs:OCC_Container and that the member of C is C1. The rdfs:domain of osrdfs:occMember is osrdfs:OCC_Container. The rdfs:range of osrdfs:occMember is osrdfs:OrderContextContainer.

osrdfs:occMember	The osrdfs:occMember is an instance of rdf:Property and a subproperty of rdfs:member property, it is used to state the member of a OSRDF contextual statement container. A triple of the form: *C osrdfs:occMember S* states that C is an instance of osrdfs:OrderContextContainer and that the member of C is S. The rdfs:domain of osrdfs:occMember is osrdfs:OrderContextContainer. The rdfs:range of osrdfs:occMember is osrdfs:Statement.
osrdfs:cMember	osrdfs:cMember is an instance of rdf:Property, it is used to state the member of OSRDF Property container. A triple of the form: *C osrdfs:cMember P* states that C is an instance of osrdfs:ContextContainer and that the member of C is P. The rdfs:domain of osrdfs:cMember is osrdfs:ContextContainer. The rdfs:range of osrdfs:cMember is osrdfs:Property.
osrdfs:pscMember	osrdfs:pscMember is an instance of rdf:Property and a subproperty of rdfs:member property, it is used to state the member of a OSRDF Property Significance container. A triple of the form: *C osrdfs:pscMember Cl* states that C is an instance of osrdfs:PropSignContainer and that the member of C is Cl. The rdfs:domain of osrdfs:pscMember is osrdfs:PropSignContainer. The rdfs:range of osrdfs:pscMember is osrdfs:PropertySignificance.
osrdfs:contextProbability	osrdfs:contextProbability is an instance of rdf:Property, it is used to state the probability of the context (subject statement container). A triple of the form: *C osrdfs:contextProbability L* states that C is an instance of osrdfs:Container or osrdfs:OCC_Container and that the member of C is L. The rdfs:domain of osrdfs:contextProbability is osrdfs:Container and osrdfs:OCC_Container. The rdfs:range of osrdfs:contextProbability is rdfs:Literal.
osrdfs:relatedRule	osrdfs:relatedRule is an instance of rdf:Property, it is used to state the contextual rule as a rule that uses contextual statements located in OSRDF contextual statement container (OrderContextContainer). A triple of the form: *C osrdfs:relatedRule S* states that C is an instance of osrdfs:OrderContextContainer and that the rule, which uses the members of the container as in-parameters is S. The rdfs:domain of osrdfs:relatedRule is osrdfs:OrderContextContainer. The rdfs:range of osrdfs:relatedRule is osrdfs:RuleStatement.

osrdfs:contextOrder	osrdfs:contextOrder is an instance of rdf:Property, it is used to state the order of a context represented by OSRDF contextual statement container (OrderContextContainer). A triple of the form: *C osrdfs:contextOrder L* states that C is an instance of osrdfs:OrderContextContainer and that the value of the context order is L. The rdfs:domain of osrdfs:contextOrder is osrdfs:OrderContextContainer. The rdfs:range of osrdfs:contextOrder is rdfs:Literal.
OSRDF Statement	
osrdfs:Statement	osrdfs:Statement is an instance of rdfs:Class and subclass of rdf:Statement. It is intended to represent the class of OSRDF statements. osrdfs:Statement belongs to the domain of the properties osrdfs:predicate, rdf:subject, rdf:object, osrdfs:inContext and osrdfs:inOrderedContext. Different individual osrdfs:Statement instances may have the same values for their osrdfs:predicate, rdf:subject, rdf:object, osrdfs:inContext and osrdfs:inOrderedContext properties.
osrdfs:F_Statement	osrdfs:F_Statement is an instance of rdfs:Class and subclass of osrdfs:Statement. It is intended to represent the class of OSRDF fact statements.
osrdfs:NF_Statement	osrdfs:NF_Statement is an instance of rdfs:Class and subclass of osrdf:Statement. It is intended to represent the class of OSRDF non-fact statements.
osrdfs:GoalStatement	osrdfs:GoalStatement is an instance of rdfs:Class and subclass of osrdf:NF_Statement. It is intended to represent the class of OSRDF goal statements.
osrdfs:RuleStatement	osrdfs:RuleStatement is an instance of rdfs:Class and subclass of osrdf:NF_Statement. It is intended to represent the class of OSRDF rule statements. osrdfs:RuleStatement belongs to the domain of the properties osrdfs:predicate, rdf:subject, rdf:object, osrdfs:trueIf and osrdfs:falseIf.
osrdfs:BehaviourStatement	osrdfs:BehaviourStatement is an instance of rdfs:Class and subclass of osrdf:RuleStatement. It is intended to represent the class of OSRDF behaviour statements. osrdfs:BehaviourStatement belongs to the domain of the properties osrdfs:bPredicate, osrdfs:bSubject, rdf:object, osrdfs:trueIf and osrdfs:falseIf.
rdf:subject	rdf:subject is an instance of rdf:Property that is used to state the subject of a statement. A triple of the form: S rdf:subject R states that S is an instance of cdfs:Statement and that the subject of S is R. The rdfs:domain of rdf:subject is rdf:Statement (and

	osrdfs:Statement accordingly). The rdfs:range of rdf:subject is rdfs:Resource.
osrdfs:bSubject	osrdfs:bSubject is an instance of rdf:Property and subproperty of rdf:subject that is used to state the subject of a behaviour statement. A triple of the form: S osrdfs:bSubject RA states that S is an instance of osrdfs:BehaviourStatement and that the subject of S is RA. The rdfs:domain of osrdfs:bSubject is osrdfs:BehaviourStatement. The rdfs:range of osrdfs:bSubject is osrdfs:ResourceAgent.
osrdfs:predicate	osrdfs:predicate is an instance of rdf:Property and subproperty of rdf:predicate that is used to state the predicate of a statement. A triple of the form: S osrdfs:predicate P states that S is an instance of osrdfs:Statement, that P is an instance of osrdfs:Property and that the predicate of S is P. The rdfs:domain of osrdfs:predicate is osrdfs:Statement and the rdfs:range is osrdfs:Property.
osrdfs:bPredicate	osrdfs:bPredicate is an instance of rdf:Property and subproperty of osrdfs:predicate that is used to state the predicate of a behaviour statement. A triple of the form: S osrdfs:bPredicate P states that S is an instance of osrdfs:BehaviourStatement, that P is an instance of osrdfs:Property and that the predicate of S is P. The rdfs:domain of osrdfs:bPredicate is osrdfs:BehaviourStatement and the rdfs:range is osrdfs:Property.
rdf:object	rdf:object is an instance of rdf:Property that is used to state the object of a statement. A triple of the form: S rdf:object O states that S is an instance of osrdfs:Statement and that the object of S is O. The rdfs:domain of rdf:object is rdf:Statement (and osrdfs:Statement accordingly). The rdfs:range of rdf:object is rdfs:Resource.
osrdfs:inContext	osrdfs:inContext is an instance of rdf:Property that is used to state the non-ordered context (contextual container) of a statement. A triple of the form: S osrdfs:inContext C states that S is an instance of osrdfs:Statement, and that the context of S is C. The rdfs:domain of osrdfs:inContext is osrdfs:Statement and the rdfs:range is osrdfs:Container.

osrdfs:inOrderedContext	osrdfs:inOrderedContext is an instance of rdf:Property that is used to state the ordered context (contextual container) of a statement. A triple of the form: S osrdfs:inOrderedContext C states that S is an instance of osrdfs:Statement, and that the context of S is C. The rdfs:domain of osrdfs:inOrderedContext is osrdfs:Statement and the rdfs:range is osrdfs:OCC_Container.
osrdfs:trueIf	osrdfs:trueIf is an instance of rdf:Property and subproperty of osrdfs:inContext that is used to state the rule condition (contextual container) of a rule statement. A triple of the form: S osrdfs:trueIf C states that S is an instance of osrdfs:RuleStatement, and that the context of S is C. The rdfs:domain of osrdfs:trueIf is osrdfs:RuleStatement and the rdfs:range is osrdfs:NF_Container.
osrdfs:falseIf	osrdfs:falseIf is an instance of rdf:Property and subproperty of osrdfs:inContext that is used to state the rule condition (contextual container) of a rule statement. A triple of the form: S osrdfs:falseIf C states that S is an instance of osrdfs:RuleStatement, and that the context of S is C. The rdfs:domain of osrdfs:falseIf is osrdfs:RuleStatement and the rdfs:range is osrdfs:NF_Container.
osrdfs:desire	osrdfs:desire is an instance of rdf:Property and subproperty of osrdfs:falseIf that is used to state the rule condition (goal container as a desire of Resource Agent) of a behaviour statement. A triple of the form: S osrdfs:desire C states that S is an instance of osrdfs:BhaviourStatement, and that the context of S is C. The rdfs:domain of osrdfs:desire is osrdfs:BehaviourStatement and the rdfs:range is osrdfs:GoalContainer.
OSRDF Property	
osrdfs:Property	osrdfs:Property is the class of OSRDF properties. osrdfs:Property an instance of rdfs:Class and subclass of rdf:Property.
osrdfs:B_Property	osrdfs:B_Property is the class of OSRDF behaviour properties. osrdfs:B_Property an instance of rdfs:Class and subclass of osrdfs:Property.
osrdfs:PropertySignificance	osrdfs:PropertySignificance is the class of OSRDF properties significances. osrdfs:PropertySignificance an instance of rdfs:Class.

rdfs:range	rdfs:range is an instance of rdf:Property that is used to state that the values of a property are instances of one or more classes. The triple of the form: P rdfs:range C states that P is an instance of the class cdfs:Property, that C is an instance of the class rdfs:Class and that the resources denoted by the objects of quadruples whose predicate is P are instances of the class C. Whenever P has more than one rdfs:range property, then the resources denoted by the objects of quadruples with predicate P are instances of all the classes stated by the rdfs:range properties. The rdfs:range property can be applied to itself. The rdfs:range of rdfs:range is the class rdfs:Class. This states that any resource that is the value of an rdfs:range property is an instance of rdfs:Class. The rdfs:range property is applied to properties. This can be represented in RDF using the rdfs:domain property. The rdfs:domain of rdfs:range is the class rdf:Property. This states that any resource with an rdfs:range property is an instance of rdf:Property or subproperty of it (osrdfs:Property as an instance).
rdfs:domain	rdfs:domain is an instance of rdf:Property that is used to state that any resource that has a given property is an instance of one or more classes. A triple of the form: P rdfs:domain C states that P is an instance of the class cdfs:Property, that C is an instance of the class rdfs:Class and that the resources denoted by the subjects of quadruples whose predicate is P are instances of the class C. Where a property P has more than one rdfs:domain property, then the resources denoted by subjects of quadruples with predicate P are instances of all the classes stated by the rdfs:domain properties. The rdfs:domain property may be applied to itself. The rdfs:domain of rdfs:domain is the class rdf:Property. This states that any resource with an rdfs:domain property is an instance of rdf:Property or subproperty of it (osrdfs:Property as an instance).The rdfs:range of rdfs:domain is the class rdfs:Class. This states that any resource that is the value of an rdfs:domain property is an instance of rdfs:Class.
osrdfs:context	osrdfs:context is an instance of rdf:Property that is used to state that any property that has a given property has a restriction of a context tolerance range in the form of a contextual properties set. A triple of the form: P osrdfs:context C states that P is an instance of the class osrdfs:Property, that C is an instance of the class osrdfs:ContextContainer and that the resources denoted by the statement context of quadruples

	whose predicate is P are instances of the class C. The rdfs:domain of osrdfs:context is osrdfs:Property. The rdfs:range of osrdfs:context is osrdfs:ContextContainer.
osrdfs:subPropertyOf	The property osrdfs:subPropertyOf is an instance of rdf:Property and subproperty of rdfs:subPropertyOf that is used to state that all resources related by one OSRDF property are also related by another one. A triple of the form: P1 osrdfs:subPropertyOf P2 states that P1 is an instance of osrdfs:Property, P2 is an instance of osrdfs:Property and P1 is a subproperty of P2. The osrdfs:subPropertyOf property is transitive. The rdfs:domain of osrdfs:subPropertyOf is osrdfs:Property. The rdfs:range of osrdfs:subPropertyOf is osrdfs:Property.
osrdfs:subjectProperty	osrdfs:subjectProperty is an instance of rdf:Property, it is used to state the subject property of the property significance object. A triple of the form: C osrdfs:subjectProperty P states that C is an instance of the class osrdfs:PropertySignificance, that P (instance of the class osrdfs:Property) is a subject property for C. The rdfs:domain of osrdfs:subjectProperty is osrdfs:PropertySignificance. The rdfs:range of osrdfs:subjectProperty is osrdfs:Property.
osrdfs:pSignificance	osrdfs:pSignificance an instance of rdf:Property, it is used to state the significance value (between 0 and 1) of the subject property of the property significance object. A triple of the form: C osrdfs:pSignificance L states that C is an instance of the class osrdfs:PropertySignificance, that L (instance of the class rdfs:Literal) is a significance value for subject property of C. The rdfs:domain of osrdfs:pSignificance is osrdfs:PropertySignificance. The rdfs:range of osrdfs:pSignificance is osrdfs:Literal.
Other OSRDF Classes	
osrdfs:OntoSmartResource	osrdfs:OntoSmartResource is an instance of rdfs:Class and subclass of rdfs:Resource. It is intended to represent the class of OSRDF resources.
osrdfs:Device	osrdfs:Device is an instance of rdfs:Class and subclass of osrdfs:OntoSmartResource. It is intended to represent the class of OSRDF devices.
osrdfs:Service	osrdfs:Service is an instance of rdfs:Class and subclass of osrdfs:OntoSmartResource. It is intended to represent the class of OSRDF services.
osrdfs:Human	osrdfs:Human is an instance of rdfs:Class and subclass of

	osrdfs:OntoSmartResource. It is intended to represent the class of OSRDF humans (experts).
osrdfs:Message	osrdfs:Message is an instance of rdfs:Class and subclass of osrdfs:OntoSmartResource. It is intended to represent the class of OSRDF messages.
osrdfs:ResourceAgent	osrdfs:ResourceAgent is an instance of rdfs:Class and subclass of osrdfs:OntoSmartResource. It is intended to represent the class of OSRDF resource agents.
osrdfs:Execution	osrdfs:Execution is an instance of rdfs:Class and subclass of rdfs:Resource. It is intended to represent the class of OSRDF executable modules.
osrdfs:RuleConditionSetter	osrdfs:RuleConditionSetter is an instance of rdfs:Class and subclass of osrdfs:Execution. It is intended to represent the class of OSRDF executable module that plays role of meta-rule performance and change the condition of the Rule Statement.
osrdfs:Role	osrdfs:Role is an instance of rdfs:Class and subclass of rdfs:Resource. It is intended to represent the class of OSRDF resource agent's roles.
osrdfs:RuleCondition	osrdfs:RuleCondition is an instance of rdfs:Class and subclass of rdfs:Resource. It is intended to represent the class of OSRDF rule conditions.
osrdfs:Active	osrdfs:Active is an instance of osrdfs:RuleCondition. It is intended to represent the "active" rule condition.
osrdfs:Passive	osrdfs:Pasive is an instance of osrdfs:RuleCondition. It is intended to represent the "passive" rule condition.
Other OSRDF Property instances	
osrdfs:significanceOfContext	osrdfs:significanceOfContext an instance of osrdfs:Property, it is used to state the significances of the contextual properties for subject property in certain context. A quadruple of the form: P osrdfs:significanceOfContext C1 C2 states that P is an instance of the class osrdfs:Property, that C1 (instance of the class osrdfs:PropSignContainer) is a significance of the context for P in the context of the contextual properties, which are collected in C2 (instance of osrdfs:ContextContainer). C2 is empty (it means that any property can be contextual for osrdfs:significanceOfContext property). The rdfs:domain of osrdfs:significanceOfContext is osrdfs:PropertySignificance. The rdfs:range of osrdfs:significanceOfContext is osrdfs:Property. The osrdfs:context of osrdfs:significanceOfContext is osrdfs:ContextContainer.
osrdfs:hasBehaviour	osrdfs:hasBehaviour an instance of osrdfs:Property, it is used

	to state the behaviour of the ResourceAgent as a container of the behaviour statements. A quadruple of the form: SR osrdfs:hasBehaviour C1 C2 states that SR is an instance of the class osrdfs:ResourceAgent, that C (instance of the class osrdfs:BehaviourContainer) is a behaviour of SR in context of C2 (instance of osrdfs:ContextContainer). C2 is empty (it means that any property can be contextual for osrdfs:hasBehaviour property). The rdfs:domain of osrdfs:hasBehaviour is osrdfs:ResourceAgent. The rdfs:range of osrdfs:hasBehaviour is osrdfs:BehaviourContainer. The osrdfs:context of osrdfs:hasBehaviour is osrdfs:ContextContainer.
osrdfs:execute	osrdfs:execute an instance of osrdfs:Property, it is used to state the executable module for action performance. A quadruple of the form: SR osrdfs:execute E C states that SR is an instance of the class osrdfs:ResourceAgent, that E (instance of the class osrdfs:Execution) is a execution module for RS action in the context of C (instance of osrdfs:ContextContainer). C is empty (it means that any property can be contextual for osrdfs:execute property). The rdfs:domain of osrdfs:execute is osrdfs:ResourceAgent. The rdfs:range of osrdfs:execute is osrdfs:Execution. The osrdfs:context of osrdfs:execute is osrdfs:ContextContainer.
osrdfs:hasRole	osrdfs:hasRole an instance of rdf:Property, it is used to state a role for resource agent. A quadruple of the form: SR osrdfs:hasRole R C states that SR is an instance of the class osrdfs:ResourceAgent and that R (instance of the class osrdfs:Role) is a role of the SR in the context of C (instance of osrdfs:ContextContainer). C is empty (it means that any property can be contextual for osrdfs:hasRole property). The rdfs:domain of osrdfs:hasRole is osrdfs:ResourceAgent. The rdfs:range of osrdfs:hasRole is osrdfs:Role. The osrdfs:context of osrdfs:hasRole is osrdfs:ContextContainer.
osrdfs:goals	osrdfs:goals an instance of rdf:Property, it is used to state a goal for resource agent role. A quadruple of the form: R osrdfs:goals C1 C2 states that R is an instance of the class osrdfs:Role and that C1 (instance of the class osrdfs:GoalContainer) is a set of goals (GoalStatements) of the role R in the context of C2 (instance of osrdfs:ContextContainer). C2 is empty (it means that any property can be contextual for osrdfs:goals property). The rdfs:domain of osrdfs:goals is osrdfs:Role. The rdfs:range of osrdfs:goals is osrdfs:GoalContainer. The osrdfs:context of

	osrdfs:goals is osrdfs:ContextContainer.
osrdfs:subGoal	osrdfs:subGoal an instance of rdf:Property, it is used to state a sub-goal for subject goal. A quadruple of the form: S osrdfs:hasRole C1 C2 states that S is an instance of the class osrdfs:GoalStatement and that C1 (instance of the class osrdfs:GoalContainer) is a set of the goals (GoalStatements) that play role of sub-goal for S in the context of C2 (instance of osrdfs:ContextContainer). C2 is empty (it means that any property can be contextual for osrdfs:subGoal property). The rdfs:domain of osrdfs:subGoal is osrdfs:GoalStatement. The rdfs:range of osrdfs:subGoal is osrdfs:GoalContainer. The osrdfs:context of osrdfs:subGoal is osrdfs:ContextContainer.
osrdfs:ruleConditionIs	osrdfs:ruleConditionIs an instance of rdf:Property, it is used to state a condition of a rule (RuleStatement) that can be active and passive (osrdfs:Active and osrdfs:Passive). A triple of the form: S osrdfs:ruleConditionIs RC states that S is an instance of the class osrdfs:RuleStatement, that RC (instance of the class osrdfs:RuleCondition) is a value of the condition for S rule. The rdfs:domain of osrdfs:ruleConditionIs is osrdfs:RuleStatement. The rdfs:range of osrdfs:ruleConditionIs is osrdfs:RuleCondition.
osrdfs:subjectRule	osrdfs:subjectRule an instance of rdf:Property, it is used to state a rule (RuleStatement) that should get new condition (active or passive). A triple of the form: E osrdfs:subjectRule R states that E is an instance of the class osrdfs:RuleConditionSetter, that R (instance of the class osrdfs:RuleStatemant) is a subject rule for E. The rdfs:domain of osrdfs:subjectRule is osrdfs:RuleConditionSetter. The rdfs:range of osrdfs:subjectRule is osrdfs:RuleStatement.
osrdfs:subjectRuleCondition	osrdfs:subjectRuleCondition an instance of rdf:Property, it is used to state a condition of a RuleConditionSetter subject rule (RuleStatement) that can be active and passive (osrdfs:Active and osrdfs:Passive). A triple of the form: E osrdfs:subjectRuleCondition RC states that E is an instance of the class osrdfs:RuleConditionSetter, that RC (instance of the class osrdfs:RuleCondition) is a value of the condition for E subject rule. The rdfs:domain of osrdfs:subjectRuleCondition is osrdfs:RuleConditionSetter. The rdfs:range of osrdfs:subjectRuleCondition is osrdfs:RuleCondition.

APPENDIX B

Prototype ontology for mobile phone calendar description:

```
<?xml version='1.0' ?>

<!DOCTYPE rdf:RDF [
      <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
      <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
      <!ENTITY cdfs 'http://www.cc.jyu.fi/~olkhriye/csdf/0.1/cdfs#'>
      <!ENTITY                                                    ontoCalendar
'http://www.cc.jyu.fi/~olkhriye/SemaSM/MobileCalendar/ontologies/CalendarOntology
#'>
]>

<rdf:RDF
   xmlns:rdf="&rdf;"
   xmlns:rdfs="&rdfs;"
   xmlns:cdfs="&cdfs;"
   xmlns:ontoCalendar="&ontoCalendar;"
>

<rdfs:Class rdf:about="&ontoCalendar;Phone">
         <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
         <rdfs:comment>Class of the phones</rdfs:comment>
         <rdfs:label>Phone</rdfs:label>
</rdfs:Class>

<ontoCalendar:CalPropValue rdf:about="&ontoCalendar;CalPropValue">
         <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
         <rdfs:comment>Supperclass of the calendar properies values</rdfs:comment>
         <rdfs:label>CalPropValue</rdfs:label>
         <rdfs:subClassOf rdf:resource="&rdfs;Class"/>
</ontoCalendar:CalPropValue>

<ontoCalendar:SynchronisationValue rdf:about="&ontoCalendar;SynchronisationValue">
         <rdfs:isDefinedBy
```

```
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Class of the calendar synchronisation propery
values</rdfs:comment>
        <rdfs:label>SynchronisationValue</rdfs:label>
        <rdfs:subClassOf rdf:resource="&ontoCalendar;CalPropValue"/>
</ontoCalendar:SynchronisationValue>

<ontoCalendar:SynchronisationValue rdf:about="&ontoCalendar;Private">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the synchronisation propery values</rdfs:comment>
        <rdfs:label>Private</rdfs:label>
</ontoCalendar:SynchronisationValue>

<ontoCalendar:SynchronisationValue rdf:about="&ontoCalendar;Public">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the synchronisation propery values</rdfs:comment>
        <rdfs:label>Public</rdfs:label>
</ontoCalendar:SynchronisationValue>

<ontoCalendar:SynchronisationValue rdf:about="&ontoCalendar;None">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the synchronisation propery values</rdfs:comment>
        <rdfs:label>None</rdfs:label>
</ontoCalendar:SynchronisationValue>

<ontoCalendar:AlarmValue rdf:about="&ontoCalendar;AlarmValue">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Class of the calendar alarm propery values</rdfs:comment>
        <rdfs:label>AlarmValue</rdfs:label>
        <rdfs:subClassOf rdf:resource="&ontoCalendar;CalPropValue"/>
</ontoCalendar:AlarmValue>

<ontoCalendar:AlarmValue rdf:about="&ontoCalendar;On">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the alarm propery values</rdfs:comment>
        <rdfs:label>On</rdfs:label>
</ontoCalendar:AlarmValue>

<ontoCalendar:AlarmValue rdf:about="&ontoCalendar;Off">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the alarm propery values</rdfs:comment>
        <rdfs:label>Off</rdfs:label>
```

```
</ontoCalendar:AlarmValue>

<ontoCalendar:TimeValue rdf:about="&ontoCalendar;TimeValue">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Class of the calendar time propery values</rdfs:comment>
        <rdfs:label>TimeValue</rdfs:label>
        <rdfs:subClassOf rdf:resource="&ontoCalendar;CalPropValue"/>
</ontoCalendar:TimeValue>

<ontoCalendar:DateValue rdf:about="&ontoCalendar;DateValue">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Class of the calendar date propery values</rdfs:comment>
        <rdfs:label>DateValue</rdfs:label>
        <rdfs:subClassOf rdf:resource="&ontoCalendar;CalPropValue"/>
</ontoCalendar:DateValue>

<ontoCalendar:RepeatValue rdf:about="&ontoCalendar;RepeatValue">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Class of the calendar repeat propery values</rdfs:comment>
        <rdfs:label>RepeatValue</rdfs:label>
        <rdfs:subClassOf rdf:resource="&ontoCalendar;CalPropValue"/>
</ontoCalendar:RepeatValue>

<ontoCalendar:RepeatValue rdf:about="&ontoCalendar;Not_repeated">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the repeat propery values</rdfs:comment>
        <rdfs:label>Not_repeated</rdfs:label>
</ontoCalendar:RepeatValue>

<ontoCalendar:RepeatValue rdf:about="&ontoCalendar;Daily">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the repeat propery values</rdfs:comment>
        <rdfs:label>Daily</rdfs:label>
</ontoCalendar:RepeatValue>

<ontoCalendar:RepeatValue rdf:about="&ontoCalendar;Weekly">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the repeat propery values</rdfs:comment>
        <rdfs:label>Weekly</rdfs:label>
</ontoCalendar:RepeatValue>

<ontoCalendar:RepeatValue rdf:about="&ontoCalendar;Fortnightly">
        <rdfs:isDefinedBy
```

```
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the repeat propery values</rdfs:comment>
        <rdfs:label>Fortnightly</rdfs:label>
</ontoCalendar:RepeatValue>

<ontoCalendar:RepeatValue rdf:about="&ontoCalendar;Monthly">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the repeat propery values</rdfs:comment>
        <rdfs:label>Monthly</rdfs:label>
</ontoCalendar:RepeatValue>

<ontoCalendar:RepeatValue rdf:about="&ontoCalendar;Yearly">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Instance of the repeat propery values</rdfs:comment>
        <rdfs:label>Yearly</rdfs:label>
</ontoCalendar:RepeatValue>

<cdfs:Property rdf:about="&ontoCalendar;calendarProperty">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Supperproperty of the calendar properties</rdfs:comment>
        <rdfs:label>calendarProperty</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&rdfs;Resource"/>
        <rdfs:context rdf:resource="&ontoCalendar;Calend"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;subject">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>subject</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <rdfs:context rdf:resource="&ontoCalendar;Meet_Memo"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;start_date">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>start_date</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&ontoCalendar;DateValue"/>
        <rdfs:context rdf:resource="&ontoCalendar;Meet_Memo"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>
```

```
<cdfs:Property rdf:about="&ontoCalendar;end_date">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>end_date</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&ontoCalendar;DateValue"/>
        <rdfs:context rdf:resource="&ontoCalendar;Meet_Memo"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;start_time">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>start_time</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&ontoCalendar;TimeValue"/>
        <rdfs:context rdf:resource="&ontoCalendar;Meet"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;end_time">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>end_time</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&ontoCalendar;TimeValue"/>
        <rdfs:context rdf:resource="&ontoCalendar;Meet"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;location">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>location</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <rdfs:context rdf:resource="&ontoCalendar;Meet"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;repeat">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>repeat</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&ontoCalendar;RepeatValue"/>
        <rdfs:context rdf:resource="&ontoCalendar;Meet"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>
```

```
<cdfs:Property rdf:about="&ontoCalendar;alarm">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>alarm</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&ontoCalendar;AlarmValue"/>
        <rdfs:context rdf:resource="&ontoCalendar;Meet_Anniver"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;synchronization">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>synhronization</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&ontoCalendar;SynhronizationValue"/>
        <rdfs:context rdf:resource="&ontoCalendar;Meet_Memo_Anniver"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;occasion">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>occasion</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
        <rdfs:context rdf:resource="&ontoCalendar;Anniver"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;date">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>date</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&ontoCalendar;DateValue"/>
        <rdfs:context rdf:resource="&ontoCalendar;Anniver"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;calendarEntry">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:comment>Supperproperty of the calendar entries</rdfs:comment>
        <rdfs:label>calendarEntry</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&cdfs;Container"/>
        <rdfs:context rdf:resource="&ontoCalendar;CalendarProp"/>
</cdfs:Property>
```

```
<cdfs:Property rdf:about="&ontoCalendar;meetingEntry">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>meetingEntry</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&cdfs;Container"/>
        <rdfs:context rdf:resource="&ontoCalendar;CalendarProp"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarEntry"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;memoEntry">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>memoEntry</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&cdfs;Container"/>
        <rdfs:context rdf:resource="&ontoCalendar;CalendarProp"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarEntry"/>
</cdfs:Property>

<cdfs:Property rdf:about="&ontoCalendar;anniversaryEntry">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>anniversaryEntry</rdfs:label>
        <rdfs:domain rdf:resource="&ontoCalendar;Phone"/>
        <rdfs:range rdf:resource="&cdfs;Container"/>
        <rdfs:context rdf:resource="&ontoCalendar;CalendarProp"/>
        <cdfs:subPropertyOf rdf:resource="&ontoCalendar;calendarEntry"/>
</cdfs:Property>

<cdfs:ContextContainer rdf:about="&ontoCalendar;Calend">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>Calend</rdfs:label>
        <cdfs:cMember rdf:resource="&ontoCalendar;calendarEntry"/>
</cdfs:ContextContainer>

<cdfs:ContextContainer rdf:about="&ontoCalendar;Meet_Memo">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>Meet_Memo</rdfs:label>
        <cdfs:cMember rdf:resource="&ontoCalendar;meetingEntry"/>
        <cdfs:cMember rdf:resource="&ontoCalendar;memoEntry"/>
</cdfs:ContextContainer>

<cdfs:ContextContainer rdf:about="&ontoCalendar;Meet_Memo_Anniver">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>Meet_Memo_Anniver</rdfs:label>
```

```
        <cdfs:cMember rdf:resource="&ontoCalendar;meetingEntry"/>
        <cdfs:cMember rdf:resource="&ontoCalendar;memoEntry"/>
        <cdfs:cMember rdf:resource="&ontoCalendar;anniversaryEntry"/>
</cdfs:ContextContainer>

<cdfs:ContextContainer rdf:about="&ontoCalendar;Meet_Anniver">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>Meet_Anniver</rdfs:label>
        <cdfs:cMember rdf:resource="&ontoCalendar;meetingEntry"/>
        <cdfs:cMember rdf:resource="&ontoCalendar;anniversaryEntry"/>
</cdfs:ContextContainer>

<cdfs:ContextContainer rdf:about="&ontoCalendar;Meet">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>Meet</rdfs:label>
        <cdfs:cMember rdf:resource="&ontoCalendar;meetingEntry"/>
</cdfs:ContextContainer>

<cdfs:ContextContainer rdf:about="&ontoCalendar;Anniver">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>Anniver</rdfs:label>
        <cdfs:cMember rdf:resource="&ontoCalendar;anniversaryEntry"/>
</cdfs:ContextContainer>

<cdfs:ContextContainer rdf:about="&ontoCalendar;CalendarProp">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/ontologies/ontoCalendar#
"/>
        <rdfs:label>CalendarProp</rdfs:label>
        <cdfs:cMember rdf:resource="&ontoCalendar;calendarProperty"/>
</cdfs:ContextContainer>

</rdf:RDF>
```

APPENDIX C

The calendar entry description:

```
<?xml version='1.0' ?>
<!DOCTYPE rdf:RDF [
      <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
      <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
      <!ENTITY cdfs 'http://www.cc.jyu.fi/~olkhriye/csdf/0.1/cdfs#'>
          <!ENTITY ontoCalendar
'http://www.cc.jyu.fi/~olkhriye/SemaSM/MobileCalendar/ontologies/CalendarOntology
#'>
          <!ENTITY calInst
'http://www.cc.jyu.fi/~olkhriye/SemaSM/MobileCalendar/instances/CalendarEntryInsta
nce#'>  ]>
<rdf:RDF
   xmlns:rdf="&rdf;"
   xmlns:rdfs="&rdfs;"
   xmlns:cdfs="&cdfs;"
   xmlns:ontoCalendar="&ontoCalendar;"
   xmlns:calInst="&calInst;"              >

<ontoCalendar:Phone rdf:about="&calInst;iPhone_1">
          <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
          <rdfs:comment>Instance of the ontoCalendar:Phone class</rdfs:comment>
          <rdfs:label>iPhone_1</rdfs:label>
</ontoCalendar:Phone>

<cdfs:Statement rdf:about="&calInst;iStatement_Meeting_1">
          <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
          <rdfs:comment>Instance of the meeting statement</rdfs:comment>
          <rdfs:label>iStatement_Meeting_1</rdfs:label>
          <rdfs:subject rdf:resource="&calInst;iPhone_1"/>
          <cdfs:predicate rdf:resource="&ontoCalendar;meetingEntry"/>
          <rdfs:object rdf:resource="&calInst;iContainer_Meeting_1"/>
          <cdfs:trueInContext rdf:resource="&calInst;iContContainer_Meeting_1"/>
</cdfs:Statement>
```

```
<cdfs:Container rdf:about="&calInst;iContainer_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:label>iContainer_Meeting_1</rdfs:label>
        <cdfs:member rdf:resource="&calInst;iStatement1_Meeting_1"/>
        <cdfs:member rdf:resource="&calInst;iStatement2_Meeting_1"/>
        <cdfs:member rdf:resource="&calInst;iStatement3_Meeting_1"/>
        <cdfs:member rdf:resource="&calInst;iStatement4_Meeting_1"/>
        <cdfs:member rdf:resource="&calInst;iStatement5_Meeting_1"/>
</cdfs:ContextContainer>

<cdfs:ContextContainer rdf:about="&calInst;iContContainer_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:label>iContContainer_Meeting_1</rdfs:label>
        <cdfs:member rdf:resource="&calInst;iContStatement_Meeting_1"/>
</cdfs:ContextContainer>

<cdfs:Statement rdf:about="&calInst;iContStatement_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:comment>Instance of the contextual statement</rdfs:comment>
        <rdfs:label>iContStatement_Meeting_1</rdfs:label>
        <rdfs:subject rdf:resource="&calInst;iPhone_1"/>
        <cdfs:predicate rdf:resource="&ontoCalendar;calendarProperty"/>
        <cdfs:trueInContext rdf:resource="&calInst;iContContainer_CalProp_1"/>
</cdfs:Statement>

<cdfs:ContextContainer rdf:about="&calInst;iContContainer_CalProp_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:label>iContContainer_CalProp_1</rdfs:label>
        <cdfs:member rdf:resource="&calInst;iStatement_Meeting_1"/>
</cdfs:ContextContainer>

<cdfs:Statement rdf:about="&calInst;iStatement1_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:comment>Instance of the calendar property statement</rdfs:comment>
        <rdfs:label>iStatement1_Meeting_1</rdfs:label>
        <rdfs:subject rdf:resource="&calInst;iPhone_1"/>
        <cdfs:predicate rdf:resource="&ontoCalendar;subject"/>
        <rdfs:object>Project Meeting</rdfs:object>
        <cdfs:trueInContext
rdf:resource="&calInst;iContContainer_Statement1_Meeting_1"/>
</cdfs:Statement>

<cdfs:ContextContainer rdf:about="&calInst;iContContainer_Statement1_Meeting_1">
        <rdfs:isDefinedBy
```

```
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:label>iContContainer_Statement1_Meeting_1</rdfs:label>
        <cdfs:member rdf:resource="&calInst;iStatement_Meeting_1"/>
</cdfs:ContextContainer>

<cdfs:Statement rdf:about="&calInst;iStatement2_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:comment>Instance of the calendar property statement</rdfs:comment>
        <rdfs:label>iStatement2_Meeting_1</rdfs:label>
        <rdfs:subject rdf:resource="&calInst;iPhone_1"/>
        <cdfs:predicate rdf:resource="&ontoCalendar;location"/>
        <rdfs:object>University. Meeting room.</rdfs:object>
        <cdfs:trueInContext
rdf:resource="&calInst;iContContainer_Statement2_Meeting_1"/>
</cdfs:Statement>

<cdfs:ContextContainer rdf:about="&calInst;iContContainer_Statement2_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:label>iContContainer_Statement2_Meeting_1</rdfs:label>
        <cdfs:member rdf:resource="&calInst;iStatement_Meeting_1"/>
</cdfs:ContextContainer>

<cdfs:Statement rdf:about="&calInst;iStatement3_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:comment>Instance of the calendar property statement</rdfs:comment>
        <rdfs:label>iStatement3_Meeting_1</rdfs:label>
        <rdfs:subject rdf:resource="&calInst;iPhone_1"/>
        <cdfs:predicate rdf:resource="&ontoCalendar;alarm"/>
        <rdfs:object rdf:resource="&ontoCalendar;On"/>
        <cdfs:trueInContext
rdf:resource="&calInst;iContContainer_Statement3_Meeting_1"/>
</cdfs:Statement>

<cdfs:ContextContainer rdf:about="&calInst;iContContainer_Statement3_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:label>iContContainer_Statement3_Meeting_1</rdfs:label>
        <cdfs:member rdf:resource="&calInst;iStatement_Meeting_1"/>
</cdfs:ContextContainer>

<cdfs:Statement rdf:about="&calInst;iStatement4_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:comment>Instance of the calendar property statement</rdfs:comment>
        <rdfs:label>iStatement4_Meeting_1</rdfs:label>
        <rdfs:subject rdf:resource="&calInst;iPhone_1"/>
```

```
        <cdfs:predicate rdf:resource="&ontoCalendar;repeat"/>
        <rdfs:object rdf:resource="&ontoCalendar;Daily"/>
        <cdfs:trueInContext
rdf:resource="&calInst;iContContainer_Statement4_Meeting_1"/>
</cdfs:Statement>

<cdfs:ContextContainer rdf:about="&calInst;iContContainer_Statement4_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:label>iContContainer_Statement4_Meeting_1</rdfs:label>
        <cdfs:member rdf:resource="&calInst;iStatement_Meeting_1"/>
</cdfs:ContextContainer>

<cdfs:Statement rdf:about="&calInst;iStatement5_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:comment>Instance of the calendar property statement</rdfs:comment>
        <rdfs:label>iStatement5_Meeting_1</rdfs:label>
        <rdfs:subject rdf:resource="&calInst;iPhone_1"/>
        <cdfs:predicate rdf:resource="&ontoCalendar;synchronization"/>
        <rdfs:object rdf:resource="&ontoCalendar;Private"/>
        <cdfs:trueInContext
rdf:resource="&calInst;iContContainer_Statement5_Meeting_1"/>
</cdfs:Statement>

<cdfs:ContextContainer rdf:about="&calInst;iContContainer_Statement5_Meeting_1">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/MobileCalendar/instances/CalendarEntryIn
stance#"/>
        <rdfs:label>iContContainer_Statement5_Meeting_1</rdfs:label>
        <cdfs:member rdf:resource="&calInst;iStatement_Meeting_1"/>
</cdfs:ContextContainer>

</rdf:RDF>
```

APPENDIX D

Semantic Multimedia message content ontology:

```
<?xml version='1.0' ?>

<!DOCTYPE rdf:RDF [
      <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
      <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
            <!ENTITY cdfs 'http://www.cc.jyu.fi/~olkhriye/csdf/0.1/cdfs#'>
            <!ENTITY smc
'http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#'>
            <!ENTITY obj
'http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#'>
]>

<rdf:RDF
   xmlns:rdf="&rdf;"
   xmlns:rdfs="&rdfs;"
   xmlns:cdfs="&cdfs;"
   xmlns:smc="&smc;"
   xmlns:obj="&obj;"
>

<rdfs:Class rdf:about="#Image">
          <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
          <rdfs:comment>Class of the images</rdfs:comment>
          <rdfs:label>Image</rdfs:label>
</rdfs:Class>

<rdfs:Class rdf:about="#Text">
          <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
          <rdfs:comment>Class of the texts</rdfs:comment>
          <rdfs:label>Text</rdfs:label>
</rdfs:Class>

<rdfs:Class rdf:about="#Image_OnTheFly">
          <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
```

```
        <rdfs:comment>Class of the "on-the-fly" images</rdfs:comment>
        <rdfs:label>Image_OnTheFly</rdfs:label>
        <rdfs:subClassOf rdf:resource="#Image"/>
</rdfs:Class>

<rdfs:Class rdf:about="#Video">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
        <rdfs:comment>Class of the video clips</rdfs:comment>
        <rdfs:label>Video</rdfs:label>
</rdfs:Class>

<rdfs:Class rdf:about="#Audio">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
        <rdfs:comment>Class of the audio clips</rdfs:comment>
        <rdfs:label>Audio</rdfs:label>
</rdfs:Class>

<rdfs:Class rdf:about="#Person">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
        <rdfs:comment>Class of the persons</rdfs:comment>
        <rdfs:label>Person</rdfs:label>
        <rdfs:subClassOf rdf:resource="&obj;Object"/>
        <rdfs:subClassOf rdf:resource="#Text"/>
</rdfs:Class>

<cdfs:Property rdf:about="#img_Type">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
        <rdfs:label>img_Type</rdfs:label>
        <rdfs:domain rdf:resource="#Image"/>
        <rdfs:range rdf:resource="&obj;ImageType"/>
</cdfs:Property>

<cdfs:Property rdf:about="#img_Content">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
        <rdfs:label>img_Content</rdfs:label>
        <rdfs:domain rdf:resource="#Image"/>
        <rdfs:range rdf:resource="&obj;Object"/>
        <rdfs:context rdf:resource="#iImgContent_Context"/>
</cdfs:Property>

<cdfs:ContextContainer rdf:about="#iImgContent_Context">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
        <rdfs:label>Context container for img_Content property</rdfs:label>
        <cdfs:cMember rdf:resource="&obj;representation_style"/>
</cdfs:ContextContainer>

<cdfs:Property rdf:about="#name">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
```

```
        <rdfs:label>name</rdfs:label>
        <rdfs:domain rdf:resource="#Person"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</cdfs:Property>

<cdfs:Property rdf:about="#husband">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
        <rdfs:label>husband</rdfs:label>
        <rdfs:domain rdf:resource="#Person"/>
        <rdfs:range rdf:resource="#Person"/>
</cdfs:Property>

<cdfs:Property rdf:about="#father">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#"/>
        <rdfs:label>father</rdfs:label>
        <rdfs:domain rdf:resource="#Person"/>
        <rdfs:range rdf:resource="#Person"/>
</cdfs:Property>

<smc:Person rdf:about="#Recipient"/>
<smc:Person rdf:about="#Sender"/>

</rdf:RDF>
```

APPENDIX E

Object ontology for message content objects description:

```
<?xml version='1.0' ?>

<!DOCTYPE rdf:RDF [
      <!ENTITY rdf 'http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
      <!ENTITY rdfs 'http://www.w3.org/2000/01/rdf-schema#'>
            <!ENTITY cdfs 'http://www.cc.jyu.fi/~olkhriye/csdf/0.1/cdfs#'>
            <!ENTITY smc
'http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/SMC_Ontology#'>
            <!ENTITY obj
'http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#'>
]>

<rdf:RDF
   xmlns:rdf="&rdf;"
   xmlns:rdfs="&rdfs;"
   xmlns:cdfs="&cdfs;"
   xmlns:smc="&smc;"
   xmlns:obj="&obj;"
>

<rdfs:Class rdf:about="#Object">
          <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
          <rdfs:comment>Upper-Class of the objects</rdfs:comment>
          <rdfs:label>Object</rdfs:label>
</rdfs:Class>

<rdfs:Class rdf:about="#Smile">
          <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
          <rdfs:comment>Class of the smiles</rdfs:comment>
          <rdfs:label>Smile</rdfs:label>
          <rdfs:subClassOf rdf:resource="#Object"/>
</rdfs:Class>

<rdfs:Class rdf:about="#TimeView">
          <rdfs:isDefinedBy
```

```
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:comment>Class of the time views</rdfs:comment>
        <rdfs:label>TimeView</rdfs:label>
        <rdfs:subClassOf rdf:resource="#Object"/>
</rdfs:Class>

<rdfs:Class rdf:about="#TemperatureView">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:comment>Class of the temperature views</rdfs:comment>
        <rdfs:label>TemperatureView</rdfs:label>
        <rdfs:subClassOf rdf:resource="#Object"/>
</rdfs:Class>

<rdfs:Class rdf:about="#Diagram">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:comment>Class of the diagrams</rdfs:comment>
        <rdfs:label>Diagram</rdfs:label>
        <rdfs:subClassOf rdf:resource="#Object"/>
</rdfs:Class>

<rdfs:Class rdf:about="#House">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:comment>Class of the houses</rdfs:comment>
        <rdfs:label>House</rdfs:label>
        <rdfs:subClassOf rdf:resource="#Object"/>
</rdfs:Class>

<rdfs:Class rdf:about="#MapWithPath">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:comment>Class of the maps with path</rdfs:comment>
        <rdfs:label>MapWithPath</rdfs:label>
        <rdfs:subClassOf rdf:resource="#Object"/>
</rdfs:Class>

<rdfs:Class rdf:about="#ImageType">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:comment>Class of the image types</rdfs:comment>
        <rdfs:label>ImageType</rdfs:label>
</rdfs:Class>

<rdfs:Class rdf:about="#MoodType">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:comment>Class of the mood types</rdfs:comment>
        <rdfs:label>MoodType</rdfs:label>
</rdfs:Class>

<rdfs:Class rdf:about="#ViewStyle">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
```

```
        <rdfs:comment>Class of the view styles</rdfs:comment>
        <rdfs:label>ViewStyle</rdfs:label>
</rdfs:Class>

<rdfs:Class rdf:about="#TimeViewStyle">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:comment>Class of the clock view styles</rdfs:comment>
        <rdfs:label>ClockViewStyle</rdfs:label>
        <rdfs:subClassOf rdf:resource="#ViewStyle"/>
</rdfs:Class>

<rdfs:Class rdf:about="#PostAddress">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:comment>Class of the post addresses</rdfs:comment>
        <rdfs:label>PostAddress</rdfs:label>
</rdfs:Class>

<cdfs:Property rdf:about="#mood">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:label>mood</rdfs:label>
        <rdfs:domain rdf:resource="#Smile"/>
        <rdfs:range rdf:resource="#MoodType"/>
</cdfs:Property>

<cdfs:Property rdf:about="#representation_style">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:label>representation_style</rdfs:label>
        <rdfs:domain rdf:resource="#Object"/>
        <rdfs:range rdf:resource="#ViewStyle"/>
</cdfs:Property>

<cdfs:Property rdf:about="#clock_style">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:label>clock_style</rdfs:label>
        <rdfs:domain rdf:resource="#TimeView"/>
        <rdfs:range rdf:resource="#TimeViewStyle"/>
        <cdfs:subPropertyOf rdf:resource="#representation_style"/>
</cdfs:Property>

<cdfs:Property rdf:about="#hour">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:label>hour</rdfs:label>
        <rdfs:domain rdf:resource="#TimeView"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</cdfs:Property>

<cdfs:Property rdf:about="#minute">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
```

```
        <rdfs:label>minute</rdfs:label>
        <rdfs:domain rdf:resource="#TimeView"/>
        <rdfs:range rdf:resource="&rdfs;Literal"/>
</cdfs:Property>

<cdfs:Property rdf:about="#owner">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:label>owner</rdfs:label>
        <rdfs:domain rdf:resource="#House"/>
        <rdfs:range rdf:resource="&smc;Person"/>
</cdfs:Property>

<cdfs:Property rdf:about="#start_point">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:label>start_point</rdfs:label>
        <rdfs:domain rdf:resource="#MapWithPath"/>
        <rdfs:range rdf:resource="#PostAddress"/>
</cdfs:Property>

<cdfs:Property rdf:about="#destination_point">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:label>destination_point</rdfs:label>
        <rdfs:domain rdf:resource="#MapWithPath"/>
        <rdfs:range rdf:resource="#PostAddress"/>
</cdfs:Property>

<cdfs:Property rdf:about="#resident">
        <rdfs:isDefinedBy
rdf:resource="http://www.cc.jyu.fi/~olkhriye/SemaSM/ontologies/Obj_Ontology#"/>
        <rdfs:label>resident</rdfs:label>
        <rdfs:domain rdf:resource="#PostAddress"/>
        <rdfs:range rdf:resource="&smc;Person"/>
</cdfs:Property>

<obj:ImageType rdf:about="#Portrait"/>
<obj:ImageType rdf:about="#Ordinary"/>
<obj:MoodType rdf:about="#Excellent"/>
<obj:MoodType rdf:about="#Sad"/>
<obj:TimeViewStyle rdf:about="#ClockDial"/>
<obj:TimeViewStyle rdf:about="#Electronic"/>

</rdf:RDF>
```

YHTEENVETO (FINNISH SUMMARY)

Semanttiseen verkkoon perustuva adaptiivinen ympäristö verkkoresursseille

Tulevaisuudessa tietotekniikka ja tietoverkot ovat läsnä kaikkialla. Uudessa ”kaiken internetissä” tietojärjestelmät kommunikoivat paitsi käyttäjien kanssa, myös toisten sovellusten, instrumentoitujen laitteiden ym. kanssa. Uuden heterogeenisen ja dynaamisen verkkoympäristön hallinta edellyttää sen resurssien eksplisiittistä semanttista kuvausta, jotta eri resurssit voidaan löytää ja yhteen sovittaa automaattisesti, jotta järjestelmien tiedoista voidaan tehdä päätelmiä ja jotta monimutkaisen kokonaisuuden komponenttien käyttäytymistä voidaan ohjata nykyistä helpommin.

Semanttisesti kuvattavat ”resurssit” eivät rajoitu dokumentteihin, verkkosivuihin ja verkkopalveluihin vaan myös erilaiset laitteistot, palvelut, asiantuntijat, organisaatiot, ym. liittyvät tulevaisuudessa suoraan tietojärjestelmiin. Tämä ei ainoastaan lisää kuvattavien resurssien määrää vaan tuo mukanaan uusia kuvattavia ominaisuuksia. Uuden internetin resurssi on proaktiivinen tavoiteohjattu dynaaminen kokonaisuus, joka reagoi itsenäisesti muutoksiin ympäristössään ja omassa itsessään. Tämän seurauksena myös toimintaympäristöstä tulee aiempaa dynaamisempi. Dynaamisuus ja proaktiivisuus edellyttävät järjestelmältä kontekstitietoisuutta, koska yhä suurempi osa tietosisällöistä ja järjestelmän käyttäytymisestä on kontekstiriippuvaa.

Nykyiset lähestymistavat eivät riitä uudessa tilanteessa. Semanttisen verkon perustyökalu RDF (Resource Description Framework) ei sisällä semantiikkaa dynaamisten ja proaktiivisten resurssien kuvaamiseen. Toisaalta dynaamisten ohjelmistoagenttien käyttäytymistä mallittavilta kieliltä puutuvat yhteiset standardit ja yhteinen semantiikka.

Tässä työssä osoitamme, että on mahdollista laajentaa RDF kuvaamaan myös proaktiivisia dynaamisia resursseja ja esittämään kontekstiriippuvaa informaatiota. Konstruoimamme formalismi (OSRDF) mahdollistaa mm sääntöjen, suunnitelmien ja käyttäytymisten kuvaamisen samassa tietomallissa näitä ohjaavan tiedon (datan) kanssa. Samalla se helpottaa ihmisille luontevien käyttöympäristöjen rakentamista uuteen internetympäristöön.