JYVÄSKYLÄ STUDIES IN COMPUTING
73

Nataliya Kohvakko

Context Modeling and Utilization in Heterogeneous Networks

JYVÄSKYLÄN YLIOPISTO

JYVÄSKYLÄ STUDIES IN COMPUTING 73

Nataliya Kohvakko

Context Modeling and Utilization in Heterogeneous Networks

Esitetään Jyväskylän yliopiston informaatioteknologian tiedekunnan suostumuksella
julkisesti tarkastettavaksi yliopiston Agora-rakennuksessa (Ag Aud. 3)
joulukuun 29. päivänä 2006 kello 12.

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. 3, on December 29, 2006 at 12 o'clock noon.

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2006

Context Modeling and Utilization in Heterogeneous Networks

JYVÄSKYLÄ STUDIES IN COMPUTING 73

Nataliya Kohvakko

Context Modeling and Utilization in Heterogeneous Networks

UNIVERSITY OF JYVÄSKYLÄ

JYVÄSKYLÄ 2006

Editors
Tommi Kärkkäinen
Department of Mathematical Information Technology, University of Jyväskylä
Irene Ylönen, Marja-Leena Tynkkynen
Publishing Unit, University Library of Jyväskylä

URN:ISBN:9513927512
ISBN 951-39-2751-2 (PDF)

ISBN 951-39-2724-5 (nid.)
ISSN 1456-5390

Jyväskylä University Printing House, Jyväskylä 2006

ABSTRACT

Kohvakko, Nataliya
Context Modeling and Utilization in Heterogeneous Networks
Jyväskylä: University of Jyväskylä, 2006, 154 pages
(Jyväskylä Studies in Computing
ISSN 1456-5390; 73)
ISBN 951-39-2751-2 (PDF), 951-39-2724-5 (nid.)
Finnish summary
Diss.

This PhD thesis is devoted to context modeling and utilization in heterogeneous networks. A special view onto context as a concept is presented. As a possible implementation of research contribution presented within this work a conceptual architecture of context-aware mobile environment is considered. The context model presented in this dissertation is based on the semantic networks data model. Interactive reasoning mechanism is considered as a way of context exchange between a context-aware application and a context-provisioning system. One of the main contribution of this work is multidimensional structural representation of semantic network and relevance potential function, which allows intelligent extraction of potentially relevant context for its consumer.

Keywords: context-awareness, domain model, heterogeneous networks, context modeling, ubiquitous computing, ontology, subject-oriented context analysis, triple data model, context consideration depth, relevance potential function, semantic network dimensionality.

Author Nataliya Kohvakko
Department of Mathematical
Information Technology
P.O. Box 35 (Agora)
FIN-40014 University of Jyväskylä
Finland

E-mail: ngerman@cc.jyu.fi

Supervisors Professor Veikko Hara
Department of Mathematical
Information Technology
University of Jyväskylä
Finland

Professor Pekka Neittaanmäki
Department of Mathematical
Information Technology
University of Jyväskylä
Finland

Professor Vagan Terzian
Department of Mathematical
Information Technology
University of Jyväskylä
Finland

Reviewers Professor Kimmo Raatikainen
Department of Computer Science
University of Helsinki
Finland

Professor Tatiana Gavrilova
Department of Intelligent Computer Technologies
St.-Peterburg State Polytechnic University
Russia

Opponent Professor Kimmo Salmenjoki
Department of Computer Science
University of Vaasa
Finland

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my supervisors - Professor Pekka Neittaanmäki, Professor Vagan Terziyan and Professor Veikko Hara.

I am extremely grateful to Pekka Neittaanmäki for his invaluable guidance and support during all these years of studies and research work in Finland, for his belief in me and his friendliness.

I thank Professor Vagan Terziyan for our creative discussions, for his careful criticism, for his friendly attitude and for the inspiration he was giving to me all this time, especially, when I ran out of energy.

I would like to express my gratitude to Professor Veikko Hara, who provided me with an industrial point of view onto my research, for our productive discussions and his interest in my work.

This work was carried out in the course of the several projects InBTC (Innovations in Business and Technology) in Agora Center supported by TEKES 2002-03 and COMAS graduate school 2003-05, therefore I am grateful to the COMAS, AgoraCenter, TEKES and TeliaSonera Inc., where I have got some support as well.

Also, I would like to thank the Department of Mathematical Information Technology of the University of Jyväskylä, where I had the fortune to conduct my research work.

I thank Nokia, where I am currently working, and, personally, Kauko Keränen and Sami Rantanen for the interest they have always expressed towards my dissertation and for the extra working time I was allowed to use to finish my thesis, whenever I really needed it.

I would like to thank Kimmo Raatikainen and Tatiana Gavrilova for their proficient and very useful comments in their review of this thesis. My best gratitude goes also to Steve Legrand for his helpful remarks towards language improvement.

I wish to express my gratitude also to my colleague, Dmytro Zhovtobryukh for our interesting and productive discussions concerning our research and other related matters. Thanks to my parents Elena and Anatoliy for having reared and encouraged me to enable me to reach this point in my life, and for their love and belief in me. I am extremely grateful to my husband Dmitri and my daughter Ulrika for their love, unconditional support and patience during these challenging and often difficult times. My gratitude towards them is greater than I will ever be able to express.

Jyväskylä, 10 December, 2006

Nataliya Kohvakko

LIST OF FIGURES

LIST OF TABLES

LIST OF ACHRONYMS

2G	Second Generation Mobile Communications System
3G	Third Generation Mobile Communications System
3GPP	Third Generation Partnership Project
3GPP2	Third Generation Partnership Project 2
4G	Fourth Generation Mobile Communications System
B3G	Beyond 3G
BDI	Belief-Desire-Intention
BRPF	Base Relevance Potential Function
CA	Context Acquirer
CAA	Context-Aware Application
CAC	Context-Awareness Capability
CAME	Context-Aware Mobile Environment
CASA	Context-Awareness Supporting Agency
CCD	Context Consideration Depth
CEP	Context Exchange Protocol
CoOL	Context Ontology Language
CP	Context Provider
CS	Context Source
DCCD	Dimensional Context Consideration Depth
DP	Data Processor
DRPF	Dimensional Relevance Potential Function
GPRS	General Packet Radio Service
IEEE	The Institute of Electrical And Electronics Engineers
IPR	Information Provisioning System
ITU	The International Telecommunication Union
LAN	Local Area Network
LMSC	LAN/MAN Standard Committee
MAN	Metropolitan Area Network
MSDA	A Middleware System That Supports Context-Aware Service Discovery
OMA	Open Mobile Alliance
OWL	Web Ontology Language
OWL DL	OWL Description Logic
PCCD	Planar Context Consideration Depth
QoC	Quality of Context
RCRF	Reactive Context Relevance Function
RDF	Resource Description Framework

RDQL	A Query Language for RDF
RPF	Relevance Potential Function
RuleML	Rule Markup Language
RUW	Rule Utility Weight
SN	Semantic Network
SRPF	Simple Relevance Potential Function
Ubicomp	Ubiquitous Computing
URI	Universal Resource Identifier
URL	Universal Resource Locator
W3C	World Wide Web Consortium
WWRF	Wireless World Research Forum
XML	Extensible Markup Language
XPath	XML Path Language
XQuery	An XML Query Language

CONTENTS

1 INTRODUCTION

What is context? There are many definitions of context, some general and some more specific: generally speaking, “context is everything”, whereas for the area of context-aware computing the classical definition given by Dey [23] is “Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves”. Hyperdictionary [40] that is based on WordNet [96] gives a general definition of context as follows: (context is) “the set of facts or circumstances that surround a situation or event”. This definition is clear and formal, but not complete because there can be a context for an object as well, as can be deduced from the Dey's definition.

Nowadays, 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. For such cases, the Dey’s definition of context no longer suits, because it is oriented towards human-computer interaction. That definition is modified by [5], which eliminates the drawback - “context information is any information and operands that can be used to characterize the situation of an entity, where an entity can be a person, a place, a physical, a networking, a storage, a service or computational object”. The weak point of this definition is that the authors are talking about entity as an object - which is not always necessarily the case. It is clear that an event or a situation (a set of objects being in a particular state at a certain point of time) among other concepts can also have context, especially if we keep in mind the dictionary’s definition.

These current definitions of context suit only for some particular domains and for certain classes of tasks. However, when talking about global connectivity and interoperability, it becomes clear that some common, maximally general and universal definition of context is needed, a definition, which would nevertheless clearly separate what is context and what is not, and which would suit any environment, any context consumer in any kind of situation.

From the viewpoint of a user of telecommunications, context can be understood as data about changes occurring in the environment. This environment most likely includes a communication network, the devices used by the users, physical surroundings and the users themselves. The context may identify a variety of the changes, from those related to network conditions to those concerning the user's mood. Context-awareness as a capability is usually divided into context extraction, context interpretation and context utilization. Nevertheless, no one seem to know what context is. There is very little work available on context characterization, which is however essential for developing context-aware systems.

This doctoral dissertation is devoted to the problem of context utilization in emerging global network environment. Recently, many scientific and technological advances have taken place on the way to global connectivity and interoperability of different networking standards and software platforms. We believe context-awareness to be one of the core enabling technologies for future global computing environment, which will most likely include fixed and mobile networks made up of different devices, migrating applications, intelligent automated knowledge processing, global personalization, etc. We consider semantic networks as a good instrument for semantic knowledge representation of domains for automated processing.

1.1 Research area

Ubiquitous Computing [94], Ambient Networks [5], [6], Intelligent Networks [39], 4G Mobile Communications [27], Wireless Internet [60] are names, which are used for referencing future intelligent, adaptive, heterogeneous network allowing fixed and wireless access to global resources and services anywhere, at any time, with any device. Semantic Web [12], [74] is part of this development and forms a backdrop to it – it is intended to advance automated intelligent knowledge processing and better co-operation in the current Web and, thus, in the future Wireless Internet. Context-awareness is seen as one of the core fundamental properties of future networks, see, for example [5], [6], [27], [60]. The grand challenge in the context-awareness area is to create a flexible context modeling framework, where the objective is to have efficient means of presenting, maintaining, sharing, protecting, reasoning, and querying context information [60].

We present a multidisciplinary research work, which lies in the intersection of a variety of technological areas and operates within a few fundamental theories. Figure 1 presents the knowledge areas which are touched in this work.

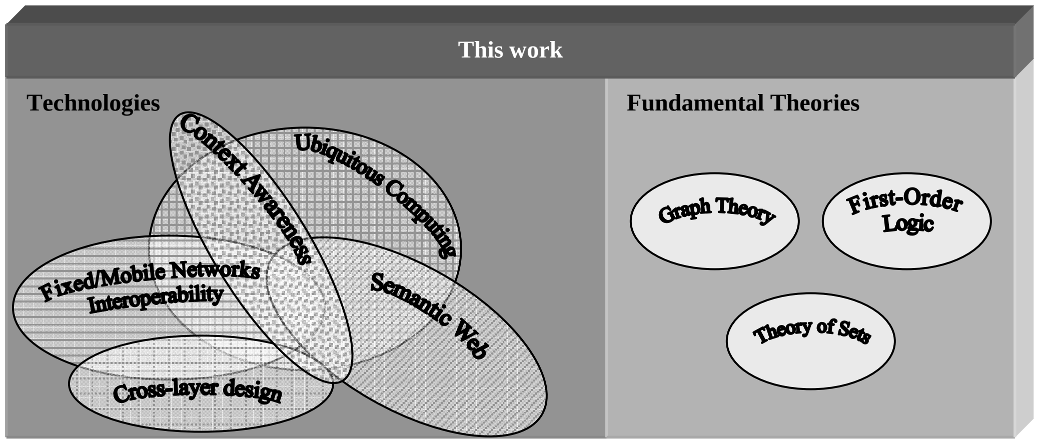


FIGURE 1 Dissertation placement in general area

The Technologies part of Figure 1 shows that the technologies considered in this work are not separated and have mutual intersections – many of the problems are considered simultaneously in different areas and different solutions are proposed. In this work we do not go deep to any direction, but absorb some basic ideas of each area, try to make use of their main achievements and try to find some common view onto a context, which would be valid for all of them. The Fundamental Theories part of the figure represents theories, which are used for modeling of context in this dissertation.

1.2 Overview of the approach

We use a combination of structural and mathematical approach for context handling. Our mathematical contribution is based on fundamental theories from discrete mathematics, basics of which can be found, for example in [33]. We introduce mathematical definitions for potentially relevant context extraction from a RDF (Resource Description Framework) [66] semantic network of domain knowledge. The more specific the request from a context consumer is, the more precise the result of a context extraction.

One of the main assumptions for the design is that the context consumer and the information system do not have precise knowledge about information structures, needs and intentions of each other. The interactive reasoning principle is based on mutual assumptions of both participants: the context consumer tries to specify its needs not knowing the structure of the available knowledge; the information provisioning system tries to select contextually relevant information for the context consumer not knowing its needs.

Design of a context-aware mobile environment is based on the idea that all of its participants can be implemented separately, can use diverse techniques and can perform different functionalities. However they must use the same context exchange protocol. A context-aware application may be considered as operable also without context information. That information, nevertheless, could potentially improve the quality of its functionality.

We understand *time* being a very important issue in context processing within dynamic environments. It is, however, left out of the scope of the major part of this work, requiring a separate comprehensive study. We assume that all information has some time stamp and that there is some filtering of outdated context performed in both sides – i.e., on the side of the context consumer as well as on the side of the information provisioning system.

We do not consider security issues, either. While understanding the importance of security, we nevertheless leave it for further research.

1.3 Contributions

In this work we consider several aspects of context modeling and use contributing into the general aim of the research area, i.e., ambient awareness capability development.

Context-aware mobile environment architecture is proposed, and represents the general view on possible locations of context acquisition, storing, provision and domain-interoperability support. The following issues are studied:

– development goals of context-aware heterogeneous networking environment;
– interaction principles between its participants;
– a set of possible functions performed by a context-awareness supporting agency as a context provisioning system;
– context provider as an initial source of information;
– general architecture of application with context-awareness capability.

Context as a concept is discussed in depth resulting in:

– a conclusion that context is a relational property, and that contextuality of the information depends on the particular situation in which it is used;
– a presentation of four properties of context;
– a brief overview of existing context measures;
– a discussion around the usual treatment of the term "context-aware application" ;
– an introduction to the context-aware interactive reasoning principle.

Context information modeling is based on the classical semantic networks data model described by ontological languages. Contribution of this work into that model presented in corresponding section consists in:

– structuring of RDF-network according to potential context relevance to a certain subject of attention;
– definition of RDF-network dimensionality;
– introduction of the context consideration depth concept.

Context provision consideration is followed by an introduction to relevance potential function, which can be used for potentially relevant context extraction from the existing semantic network of domain knowledge.

Context-aware decision making process is presented as a part of a context consumer functionality. In addition to the basic McCarthny's rule-based decision making approach the following concepts are introduced:

- rule utility weight;
- context fact significance (defining context relevance within a context consumer).

The division into context consumer and context provisioning system in this work is, in fact, very relative. The main assumption is that the client (context consumer) has a limited amount of memory resources and processing power, while the system is very powerful. However, intelligent context provision and processing could be integrated within one agent which would then be able to perform both functions: intelligently share context information with other agents and make own decisions based on available information. Moreover, the presented context analysis technique (fact significance calculation) could be used for intelligent target-based context provision, which is, however, out of the scope of this work.

1.4 Organization of the thesis

In this section we briefly outlined the research area, our approach and contribution of the dissertation. The rest of the work is organized as follows.

Section 2 provides necessary background in the field of research. In Sub-section 2.1 we present foundations of the Semantic Web, and the current standards and recommendations, relevant to this work. Sub-section 2.2 contains an overview of context-aware computing as a research area including context modeling as a primary research direction of this work. In Sub-section 2.3 we describe historical perspective, trends, and development aims of future world of interconnected networks. The sub-section also provides a brief overview of research communities in the field, and of research directions and challenges. Sub-section 2.4 contains a description of the research problem considered in this dissertation and its research questions. In Sub-section 2.5 we provide a motivation for this approach.

Section 3 presents context-aware mobile environment as a reference framework, which shows possible implementation of theoretical outlines given within this work. In Sub-section 3.1 we present a general architecture of the environment and outline its development goals with respect to functionality. Within this section we also consider several participants of the environment – in 3.2 we discuss context-awareness supporting agency, in 3.3 we present an architecture of context provider, in 3.4 we talk about context exchange protocol, and 3.5 contains considerations about context-aware application. We compare the environment presented here with other works in Sub-section 3.6 and conclude in 3.7.

In Section 4 we present a general understanding of context. In Sub-section 4.1, we consider contextuality as a relational property of information. Relevance, being a core property of context information, is discussed within Sub-section 4.2. We present a philosophical model of context in Sub-section 4.3 and its general properties in Sub-section 4.4. Such important aspects of context information as its precision, formality, probability of correctness, up-to-dateness and other similar issues are dealt with in Sub-section 4.5. We introduce interactive context-aware reasoning principle in Sub-section 4.6. Context-awareness as a capability of application is considered in Sub-section 4.7. The Sub-section 4.8 discusses the issues of context information storing. Sub-section 4.9 concludes Section 4.

Section 5 presents semantic network as an underlying information structure of the thesis. In 5.1 we discuss basic definitions and semantic meaning of ontologies, 5.2 presents RDF data model and its use, some additional definitions are presented in Sub-section 5.3. In 5.4 we discuss the structure of RDF network. We present our concepts of context consideration depth and dimensional context consideration depth in Sub-sections 5.5 and 5.6 correspondingly. Discussion about context modeling based on ontology languages is presented within Sub-section 5.7. We conclude the section in 5.8.

We present our main mathematical contribution in Section 6. In 6.1 we describe where mathematical techniques can be used within interactive reasoning process. We discuss context relevance function definition possibility within Sub-section 6.2. We introduce our RPF (Relevance Potential Function) within 6.3 in its different modes – simple RPF, predicative RPF, and decontext RPF - and provide a validation for it. The Sub-section 6.4 presents a dimensional variant of the relevance potential function and its validation. Cumulative relevance potential function is introduced within Sub-section 6.5. We present a calculation example in Sub-section 6.6. Some possible extensions to RPF are shown within Sub-section 6.7. We comprehensively discuss possible applications of the presented mathematics in Sub-section 6.8, and we conclude the section in 6.9.

Section 7 is devoted to context processing from a context consumer point of view. Context-aware application as a context consumer is characterized within 7.1 and 7.2. States and sources of context information are discussed in 7.3. We present an example approach for context facts derivation in 7.4 and propose decision enabling rules to for context-aware actions in Sub-section 7.5. Sub-section 7.6 introduces rule utility weights, which are used for conflict resolution and context analysis, as shown in 7.8. Primitive context reasoning model is described in 7.7 for the process of context-aware reasoning. Sub-section 7.9 depicts the constituents of context request from a consumer to an information provisioning system. We conclude in Sub-section 7.10. In Section 8 we summarize the main results of this work and outline some directions for future research.

2 BACKGROUND

In this section we provide necessary background in the field of research. In Sub-section 2.1 we present the foundations of the Semantic Web, and the current standards and recommendations that are relevant to this work. Sub-section 2.2 contains an overview of context-aware computing as a research area including context modeling as a primary research direction of this work. In Sub-section 2.3 we describe historical perspective, trends, and development aims of future world of interconnected networks. It also provides a brief overview of research communities acting in the field, research directions and challenges. Sub-section 2.4 contains a description of the research problem, considered in this dissertation and its research questions. In 2.5 we present out motivation for the approach.

2.1 Semantic Web

The idea and the aim

The amount of content in the Internet has grown to such an extent that it has become a rather complex task to find some particular information in that global network. Despite the fact that search engines such as Yahoo, Google, AltaVista, etc. are fast and powerful, they return to the user huge amounts of information relevant to the given keywords and it takes sometimes a really long time to find the information the user is looking for. However, if the user were provided only with a small part of the available relevant information, the looked-for information might not be found.

Another aspect of the normal Internet search practice is that the search is done based on the exact keywords and phrases given by the user. The same things might have very different labels within different communities, and as a consequence the user would not receive information that is relevant but described in other terms than what the user uses.

Since artificial intelligence in its classical philosophical sense does not yet equal human intelligence, people need to provide a kind of technique for machines and software which would enable some kind of “understanding” of the meaning of available information. In other words, provide some machine-understandable knowledge base for software in order to get more intelligent automated treatment, by that software, of the resources available in the global network.

The idea of semantic annotation of the web resources appeared in the early '90s and has been actively evolving up to now. A comprehensive list of dedicated works can be found in [50]. The original article [12], which 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 [12] were later revised in [74] with respect to current situation.

The idea of semantic annotation of the web resources appeared in the early '90s and is actively growing now under the new web architecture called the Semantic Web. 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 [50].

What is the idea of the Semantic Web then? Simply put, each resource in the web (page, service, printer, server, agent, etc.) is provided with formal annotation describing its purpose, functionality, and relation to other resources or any other information, which could be useful for automated dealing with the resource. Usually, this annotation is provided by humans, for example, annotation of a WWW page is provided at the time of its publication in the Internet, and annotation for a printer is provided together with its installation. However, semantic annotation could also be generated automatically were the required information available for the performing software.

The current situation on the web is such that some of the resources are supported with some kind of semantic annotation while some of the resources are not - this is usually based on the needs. There are many mechanisms already in the web, which can successfully deal with semantically annotated resources. Resources still lacking semantic annotations are processed as previously, using normal well-known practices.

Internet is not only a multimedia content provider, but also a service network with service applications and resources that can be effectively found and provided to the user in an appropriate form, place and time. Furthermore, the user who needs a service, may move through networks, change physical location, and move across different devices. Such user freedom requires additional facilities – the user has to be able to obtain services even in a completely new environment in which he or she should also be able to communicate.

As a conclusion to the above consideration we can say that the global network resources need to be improved in order to meet growing user demands with more intelligent techniques, supporting, e.g.:

– semantic searches (when software agents are performing search on behalf of the user knowing the user's needs and analyzing semantics of the available information);
– service discovery (when a new user comes to an environment unknown to her or him, the user needs to know which services are available and how to get them);
– intelligent automatic or automated processing of the information;
– appropriate transformation of the available information according to the user’s demands (for example, transformation of information resources or services according to the user’s terminal)
– automatic (performed by software agents) negotiations about the available services, their costs and content with service providers, etc.

Current standards and recommendations

W3C is a World Wide Web Consortium [92] 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 Semantic Web. Many scientific achievements have been already based on a variety of Semantic Web languages specifications, which are defined by W3C and other entities. Actually, there is already a group of specifications for different purposes, allowing different levels of formality and semantics. The general map, in Figure 2, includes some of the current Semantic Web languages in relation to other information and knowledge representation tools and approaches.

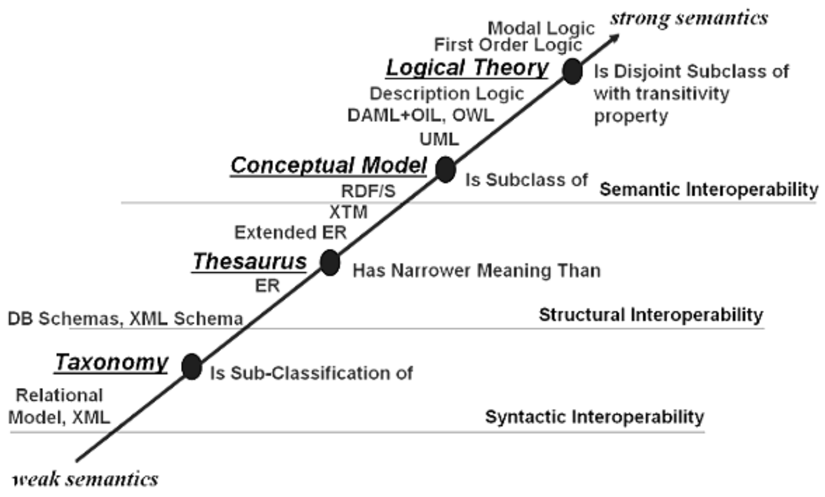


FIGURE 2 Ontology Spectrum: One View [55]

Currently, the W3C consortium offers a set of language specifications, which allow structural representation of available knowledge. Some of them are less formal, some include a number of restrictions and additional vocabularies allowing 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) [26] 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 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 [66]. The RDF consists of the RDF data model and vocabulary definition (RDF schema). It has a flexible xml-based syntax, which provides a formal semantics and provable inference. 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 issues:

– 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 [57].

OWL DL [57] 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 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.

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[4] Buchingae [16], RuleML (The Rule Markup Language) [91], RDQL (A Query Language for RDF) [65] 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. Results of SPARQL queries can be ordered, limited and offset in number, and presented in several different formats [77]. Recently, W3C has been actively developing SPARQL group of specifications, which include SPARQL Query language [77], SPARQL Protocol [76] and SPARQL Query XML Results Format [78].

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 use of XPath within URIs and XML attribute values. XPath operates on the abstract, logical structure of an XML document, rather than 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 [98].

XQuery is another language under W3C work. 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 [99].

2.2 Context-aware computing

Context awareness capability provision

In highly dynamic mobile communication environments context identifies a wide variety of context information related to user preferences, computing and communication environment and physical surroundings. In the presence of mobility, context changes occur frequently. This makes context-awareness challenging. Reactions to context changes are essential to the user and, therefore, applications should take them into account.

Theoretical foundation of context-aware computing is being advanced by such researchers as Ahind Dey et al. (Georgia Institute of Technology), Harry Chen et al. (University of Maryland), Albrecht Schmidt et al. (University of Lancaster) and others. The main developers of industrial solutions during the last fifteen years have been Xerox PARC [58], Oracle & Olivetti Research Lab [7], MIT Media Lab [52], Georgia Lab Tech [28] and many others.

Context-awareness capability is being studied under various research areas and directions. A huge amount of research and development work is conducted in the area of mobile computing [10], [17], [21], [49], [68], [67], [72], etc. Ubiquitous, Pervasive and Wearable Computing are the directions which most strongly focus on the context-awareness, since it is one of their key enablers [19], [34], [36], [61], [62], [61]. However, many of the research activities concerning the topic can be found within the Semantic Web [3], [18], [32], [46], [54], Artificial Intelligence [11], [14], [45], [59,] [73] and other research areas.

The most developed context-awareness techniques are location-aware computing, sensor networks and smart environments [10], [19], [36], [62]. In the first of these areas context-awareness is related to user location and presence awareness; the second area concentrates on acquisition, processing and suitable computer representation of environmental physical data; the third area usually considers context as a situation of the user in a concrete environment. We can roughly split context-awareness as a process into four constituents shown in Figure 3.

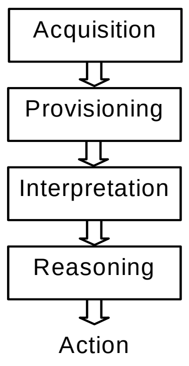


FIGURE 3 Context-awareness Support

Context acquisition is responsible for acquiring context information from various sources in the environment, including physical sensors, user profiles, communication sessions, and applications. This is a research direction that has been intensively developed in the recent past and focuses mainly on hardware development. A large amount of work is being devoted to design and enhancement of various sensors and on deployment of distributed sensor network architectures. Some early examples of sensor development projects are MediaCup [1] (Institute of Telematics, University of Karlsruhe) and SmartFloor [2] (Georgia Institute of Technology). The wearable computing paradigm also makes use of sensor hardware, which is primarily oriented on sensing human physiological reactions and includes such products as StartleCam[3] (MIT Media Lab).

Context provisioning consists of delivering, storing, registering, possibly transforming and replicating context information. Typically context provisioning tasks need to be supported by specific middleware services. Recently, much attention has been paid to the design of middleware systems for the universal provision of context-awareness to mobile applications [19], [21], [31], [36], [75]. In the near future, there will probably be some of them operating in network spaces. In this area the greatest attention is being paid to elaboration of functionality, which consists in facilitation of context delivery from context sources to applications. The current development trend is mainly focused on implementation of *reflective* middleware providing support for dynamic customization of service delivery. Past projects include Chisel[4] (Trinity College Dublin), Heywow[5] (German Aerospace Center, Institute of Communications and Navigation), PACE [6] (University of Queensland); some examples of ongoing projects are Sparkle[7] (The University of Hong Kong), Dynamos[8] (The University of Helsinki) The majority of middleware-oriented projects design software architectures enabling development of context-aware applications on top of their specific middleware systems. The problem of the area from our point of view is that the designed middleware are using somewhat differing approaches for representing context, which makes interoperability quite complicated. With the growth of context-awareness technology the amount of available context will grow enormously. In such a situation it makes no sense to have many task-oriented locally organized context-aware systems if they are unable to exchange information.

Context interpretation corresponds to the process of learning about the context, particularly about what the end-user really wants. Context characterization is an important aspect of context interpretation consisting of determining all possible context types and their characteristics. Unfortunately, this task is rather poorly addressed within the current research activities related to context-awareness. Only proprietary solutions that are effective for context characterization are available. Despite that researchers in the field indicate the exceptional importance of context

---

[1] http://mediacup.teco.edu/overview/engl/overview.html
[2] http://www-static.cc.gatech.edu/fce/smartfloor/
[3] http://vismod.media.mit.edu/tech-reports/TR-468/node3.html
[4] http://www.dsg.cs.tcd.ie/Chisel
[5] http://www.heywow.com/p0en01001001000.html
[6] http://www.itee.uq.edu.au/~pace/
[7] http://www.csis.hku.hk/~clwang/projects/sparkle.html
[8] http://virtual.vtt.fi/virtual/proj2/dynamos/

categorization for future context-aware systems: "A categorization of context types will help application designers uncover the most likely pieces of context that will be useful in their applications" (Ahind Dey).

Context reasoning represents a decision-making procedure that produces the necessary adjustments in the application behavior so that they adequately take into account the current context of the execution environment. Context reasoning is also a research direction progressing in a fairly restricted manner. Usually every single context-aware system proposes its own specialized reasoning mechanism for creation of adaptive applications. Furthermore, these mechanisms are most likely not reusable since they are strictly oriented towards certain context types and tightly built-in underlying middleware architectures. A significant and aspiring-to-universality example of context reasoning engines' development is MyCampus[9] environment (Carnegie Mellon University). One more important problem within the context reasoning is sophisticated decision making – decision feasibility area is non-orthogonal due to a large number of context variables that are frequently correlated and conflicting.

Contemporary achievements in context utilization usually represent proprietary solutions making use of concrete context types. Location and time are currently the two most utilized contexts. Major application categories are location-based services for mobile users, navigational systems - Cyberguide[10] (Georgia Tech), Guide[11] (University of Lancaster), and presence management, for example, The Context Toolkit[12] (University of Berkley).

In the '90s context-aware computing was considered mainly as a part of human-computer interaction research area and was mainly devoted to acquisition, formalization and the use of real world information surrounding the computer user [9], [17], [23], [71], [72]. Nowadays, with the growth of technologies, the area is becoming much larger – researchers want to use context information as much as possible. Many new research directions related to context have arisen. The main problem is that the use of context is quite specific in different areas, and the solutions usually do not have much use outside their own application areas.

There are many definitions for context in both specialized and general research areas, for example in [17], [23], [40], [72], [100]. Such situation has taken place because there is no uniform definition, which would suit all the intuitive understandings that people have about context. Therefore, people are defining context according to their needs and feelings. The most widely used definition is the one given by [23]: it defines context with respect to user-computer interaction, but one needs to remember that the user is not always present in context-aware computing. According to [71] there are three categories of context – user context,

[9] http://www.cs.cmu.edu/~sadeh/mycampus.htm
[10] http://www3.cc.gatech.edu/fce/cyberguide/
[11] http://www.guide.lancs.ac.uk/overview.html
[12] http://www.cs.cmu.edu/~anind/context.html

physical context (physical environmental data) and computing context. Computing context includes characteristics of connection, network and user device capabilities, service application execution circumstances, etc. Most of such information exists and can be tracked within certain exchange protocols, control procedures, profiles, etc. Using this information may add efficiency to applications and control operations on different network layers; however, it is usually not made available for wider use. This question is broadly addressed in the Ambient Networks project [6], where a new network architecture utilizing network (computing) context is being developed. Cross-layer design [20], [73], is an emerging research area addressing information exchange between different network layers.

The future computer networks seem to be context-aware by their very nature [5] – context-awareness is no longer seen as a challenge, but as a natural property of future applications. In the research areas mentioned significant progress have been made on the way of understanding and using context, however the results obtained can not be directly used for provision of general context-awareness, because the context-aware solutions developed are usually task-oriented and are not interoperable in a general sense. Our earlier work [48] on service characterization shows, how many different services exist in networking environment, therefore, it becomes just not possible to provide own context model for each of them without some common approach.

Context modeling

Many different context modeling approaches have recently gained in popularity. The most representative of these are [81]: key-value models, markup scheme models, graphical models, object oriented models, logic based models and ontology based models. Modeling of context is usually understood as a formal representation of concepts and their relationships within some domain from the real world or from the computing world. The use of various modeling approaches creates the problem of interaction between the different domains that are described by the suggested models. Such interaction is already difficult due to a difference in notation of the same concepts across domains. However, it becomes even more complicated when these domains are described using different modeling approaches.

In much of the research done context is treated as something separate from other information, a set of context parameters is defined and special techniques for context processing are provided, for example in [30]. We do not agree with such a position. Back in 1997 in [25] Edmonds wrote: “Perhaps a more natural way of formalizing context in order to compare different conceptions, is to not distinguish contexts from other objects of reasoning, learning etc. but to be chosen such that certain objects act like contexts in certain circumstances”.

One of the most popular context modeling approaches consists in building a formal model of the context (which, in fact, is the domain) related to the concrete task or group of tasks [32], [35], [84], [93]. Such models do not differ much from the huge variety of conceptual models of similar domains, which may not have been built under the name of context. However, in many of these models context is still separated from other available information and processed separately.

From our point of view there is no need to model context for each particular task separately from the model of the domain itself. Rather, there is a need to build models of larger domains, such as networks or personal devices, and to use relevant parts of them. Ontologies are particularly suitable for describing and storing semantic annotations of different domains. Furthermore, a lot of effort has been put into building ontologies of particular domains. We believe they can be efficiently utilized in context awareness provision.

Despite the fact that the basic philosophy behind many of the existing context models differs from ours, this does not contradict with our own context modeling technique. We offer a new approach for relational consideration of context, which can be easily applied to many of the existing models. Many of the modeling techniques describe domains with the help of objects, their properties and relationships. In the case of conceptual models we have concepts, possible relationships and possible properties; in the case of realizations we have instances, real properties and real relationships with their values. The core difference between the models is in their formal representation of these things and, thus, in their computing processes. The only requirement for a model to be used for working with context under the subject-oriented approach is that there should be a place for a subject in it.

Our CCD (Context Consideration Depth) modeling technique could be adapted to many other research frameworks in the area. For example, McCarthny's context model [51] with the basic relation *ist(c,p)* (*p* is true in a context *c*) could use our context sets as a context condition *c*. A context broker in [18] after building an ontology of the domain could use CCD for the selection of task-relevant context.

The research paper [11] presents the dimensions of context representation dependence. Three fundamental dimensions are defined:

– partiality – the portion of the world, which is taken into account;
– approximation – the level of detail at which a portion of the world is represented;
– perspective – the point of view from which the world is observed.

The subject-oriented approach in this work comes with a technique which solves the task of context partiality – out of the available context information using our CCD model it is easy to select context which is potentially relevant to the task. The perspective aspect is also touched – the subject, from which context selection starts and the concept of context-on-dimension represent a point of view from which the world is observed.

Ontological representation of conceptual information is seen nowadays as a good tool for building a global base of structured and formalized knowledge. It allows global exchange and referencing of conceptual knowledge and makes it possible to represent the real world in a way that can be understood by computers. We agree with [81] that the most promising context models are those which utilize ontologies. However, an ontology is merely a good instrument for representing and storing information about concepts and their relationships, but some special techniques on-top of ontologies for context-awareness support are to be developed. Therefore, there is a need for an approach for context modeling that could be used in all areas of context-aware computing and form the base for global context-awareness capability.

2.3 Moving towards wireless future

Historical perspective

In the past the world of communication was split into two almost disjoint major areas: fixed and mobile communications. These, however are now converging. While the former was mainly thought in terms of the traditional Internet, which is the dominant and nearly global communication system, the latter one embraced a number of competing wireless technologies. Fixed communication networks provided a solid base for high-speed, fixed access and reliable data services while mobile communication systems, originally designed for circuit-switched voice transmission, supported wireless network access on the move at the expense of reduced data rates.

Human demands have stimulated requirements for an evolution of technologies in which different standards start to converge towards each other. New ways of wireless Internet connection have been provided and data transmission through voice networks has become possible. This process has resulted in an emergence of many different standards and technologies (many of them hybrids) on different networking layers in order to solve certain important tasks, to have certain desirable features, or to be flexible enough to be applied in different contexts.

The first generation of mobile communications has resulted in invention of novel microprocessors and digitalization of a control link between the mobile phone and a base station. However, voice communication has remained analog. Second Generation mobile communications system (2G), that is GSM (Global System for Mobile Communications) discussed in [63], can be characterized by digitalization of actual voice signal in addition of the control layer signaling. Late 2G, referred to widely as 2.5G or GPRS (General Packet Radio Service), an overview of which can be found in [13], has brought data packet services to cellular mobile systems. But these devices were still limited. Third Generation mobile communications system (3G), various specifications of which can be found in [1], promised a wide variety of smart services (voice, fax, e-mail, multimedia, web browsing etc.) with a seamless global access roaming capability. Fourth Generation mobile communications system (4G), vision and roadmap of which is deeply discussed in [27], is seen as a ubiquitous global networking solution. The basic idea of 4G is the following: "4G will deliver low cost multi-megabit/s sessions any time, any place, using any terminal" [69]. Figure 4 shows historical evolution of mobile technologies.

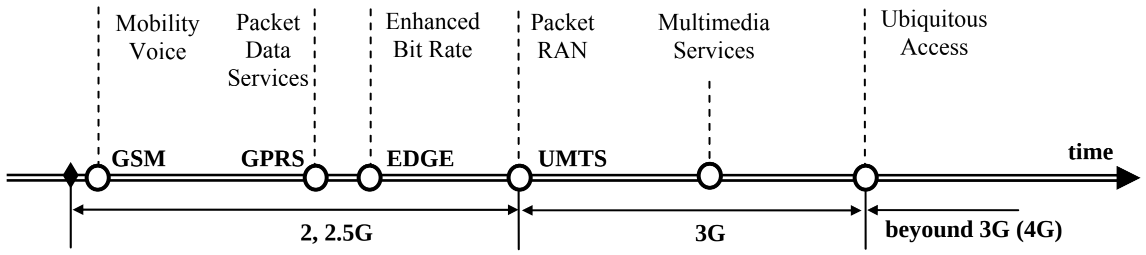


FIGURE 4 Mobile technologies evolution

Wireless Internet

Current trends in networking area clearly show that the era when almost any user could get connected to the global network with different devices and receive rich, fast and personalized multimedia services is not far away from now. It will be a new generation of global mobile communication systems, which will affect almost all aspects of human life.

Different kinds of issues, related to such vision of the near future, have been under extensive discussion during last years. The first such vision was introduced already in the early '90s by an American scientist Mark Weiser, who has been working in this area from the very beginning up to now. His paper [94] describes a vision of ubiquitous computing, which was really revolutionary at the time of publication. "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it" – this is the first phrase of that historical paper in the Scientific American in 1991 [94]. So, the main idea of Ubicomp (Ubiquitous Computing) was that computing is so deeply embedded in our everyday life that we do not think about it anymore. Electronic tabs, pads, notebooks, blackboards, personal digital assistants, vehicle computers, digital cameras, thermostats, conditioners, stereo systems, etc. – they all are interconnected, synchronized, and recognized; each of these and other types of devices have their right place and right time for use; switching of information takes place automatically from one device to another; the users is always authenticated and tracked in a proper way; as much as possible, technical operations are done automatically and are invisible to the user; and information that user receives is limited to that which is really of her or his interest.

During the last decade there have been many different views on the future, appearing in literature under different names. The authors of [8] explore the idea of *nomadic computing*: “nomadicity may be defined as the system support needed to provide a rich set of computing and communication capabilities and services to nomads as they move from place to place in a transparent, integrated and convenient form”. In [70] Mahadev Satyanarayanan discusses emerging *pervasive computing area* with regards to the situation in 2001 and Weiser’s idea, and points out open research questions in the area. Recent activities in the area use also the terms *ambient networks* [6], (systems) *beyond 3G*, for example, in [29], and *4G Generation Mobile Communications* [27]. We agree with Kimmo Raatikainen [60], who refers to “the ‘thing’ as a Wireless Internet”, that all the above treatments are very close to each other.

The current maturity of IP, meaning its new version IPv6[41] with mobility support [43], allows presuming that it will be used as a base for future integrated global network solution as presented in Figure 5.

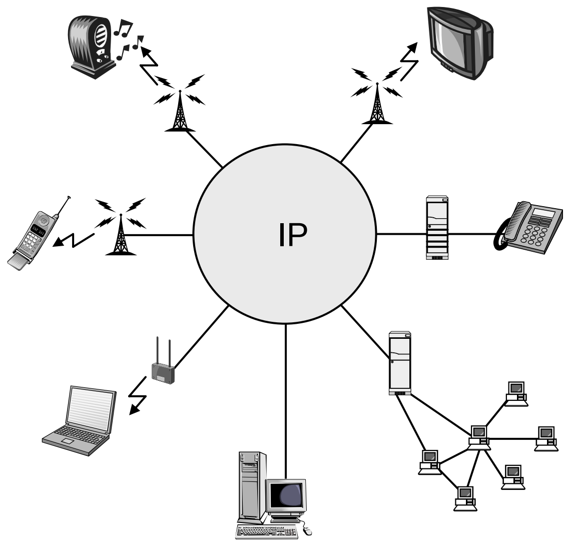


FIGURE 5 All-IP network

The works clarifying requirements, research challenges, architecture and needed technologies for 4G are already ongoing. For example, Mobile IT Forum [53] did comprehensive study for clarification of possible use cases, user and business needs, technology roadmapping, system configuration and applications of the fourth-generation mobile communications systems and published the results in “Flying Carpet” brochure [27].

Research communities

There are different research communities, who study various issues related to Wireless Internet. We present here some of them, and the most recent overview of the relevant organizations can be found in [83].

Wireless World Research Forum (WWRF) is a consortium of many leading mobile telecommunications companies, research organizations and universities. Its goal is to ensure momentum, strategic orientation and impact of the research on wireless communication beyond 3G [97].

The Institute of Electrical and Electronics Engineers (IEEE) is a leading authority in many technical areas, including development and promotion of technologies beyond 3G. IEEE 802 LAN/MAN Standard Committee (LMSC) develops standards for Local and Metropolitan Area Networks and short-range communications [37]. Some of the IEEE 802 standards, according to [83], will play an integral part in future wireless world.

B3G/4G at the International Telecommunication Union (ITU) have undertaken high-level vision work to help facilitate global consensus on basic system concepts [42].

Along regional initiatives, we would like to mention the *European Framework Programmes for Research* (ongoing FP6, starting FP7) which are platforms for collaborative research in Europe [42] and *Mobile IT Forum* [53], which was created to realize Future Mobile Communication Systems and Services such as the fourth-generation mobile communications systems and mobile commerce services [83].

Standardization issues are dealt with mainly by *Third Generation Partnership Project* (3GPPx) [1], [2], *The Internet Engineering Task Force* (IETF) [38] and *Open Mobile Alliance* (OMA) [56].

Research directions and challenges

Ubiquitous computing from the very beginning has been a well-focused, concrete research idea despite seeming slightly idealistic. During the last fifteen years the area has grown so much that when people talk about ubiquitous computing, they consider it more as a vision than as a research area. We can say that the majority of current research and development activities in the world of computing and networking are directed towards ubiquitous computing.

Quite a many research areas are contributing to future Wireless Internet mainly as its enablers, amongst them:

– Context-aware computing,
– Wearable computing,
– Sensor networks,
– Embedded systems,
– Intelligent environments,

- Smart antennas and other advanced radio equipment,
- Converged networks,
- Location-aware services,
- Human-Computer Interaction, and
- Artificial Intelligence.

One of the practical achievements in the area is Indus [90] programming language and its corresponding platform. Indus enables implementation of software agents and software components - both new abstractions in the language. The software platform of the same name is intended to support Ubiquitous, Autonomic and Adaptive computing. The Indus platform thus enables modeling of any application as a set of concurrently executing agents that cooperatively execute tasks by coordinating with each other and composing components [90].

In [60] the author expresses an opinion that for bringing MarkWeiser's dreams to reality the world's research needs to reconsider some foundations of mobile computing and communications. We agree with [60] and our work confirms this claim as we try to consider context in computing from a very conceptual, almost philosophical point of view. Already in 2001, see for example [82], it was clear that the main feature and the challenge of future interconnected world would be its diversity, which requires seamless integration of different technologies and high level of adaptability as far as the applications are concerned.

Diversity of communication networks, services and devices requires certain set of technologies to be developed. Kimmo Raatikainen in [60] divides the Wireless Internet research space on the following areas:

Reconfigurable systems include the following issues:

- self-awareness support,
- dynamic reconfigurability,
- ad-hoc communities maintenance,
- service and resource discovery,
- efficient filtering of incoming notifications,
- decision rules for reconfiguration,
- software upload and download together with on-line upgrades and rollbacks.

Context modeling is the most problematic part of context-awareness research, there is no common view on the issue since almost all information can be contextual, moreover each applications has its own view on context. The grand challenge is to create a flexible context modeling framework. In this area one of important research directions is distributed data management.

Security, trust and privacy in Wireless Internet contain the following research issues:

- protecting system against unauthorized modifications,
- program validation and verification,
- trust modeling
- efficient and controlled sharing of information fragments,

- certificate management,
- ad-hoc communities implications.

Software development and maintenance is one of very important aspects in the area, extensively studied within research community. There is a clear need of creating adequate service or software architecture with just enough level of details. One of the main goals of service architecture creation is its modularity – any module can be replace without disturbing others.

New *programming models* are needed for creation of context-aware, adaptive and personalized applications, which are able to adjust their behavior and configuration according to changes in the environment.

Efficient wireless connectivity is another important aspect of Wireless Internet. One of the core problems with wireless links is their instability, which should be efficiently handled. Recently, wireless networking has been under extensive development, however, there is still many open issues:

- adaptation to varying physical conditions,
- seamless co-operation between mobility mechanisms,
- Quality-of-Service and security,
- support of multiple simultaneous flows and different transport protocols,
- bit-efficient wire formats,
- implication of ad-hoc networking,
- group communication,
- multicast and multihoming,
- personal, session and network mobility management.

Ambient awareness as a goal

In [5] the concept of Ambient Networks is presented. The aim of the Ambient Networks project is to make network itself context-aware in such a way that not only end user applications have context-awareness capability, but also some inner network functions (like roaming, mobility support, connection establishment, session control, etc.) are context-aware. Service reference model [101] is used in Figure 6 on the next page to present the idea of cross-layer context circulation – any kind of context is available on any layer of networking functionality. This is a very challenging long-term perspective, realization of which would clearly advance current networking technologies.

Context information is, at least in principle, non-critical information about the network, its devices and users [5]. The [5] defines the following aims for ContextWare (software for context-awareness support):

- To make the network itself context-aware and at the same time make context information about the networks available to high-level applications and services.
- To clearly separate use of context information from its collection.

– To provide various levels of support, from "atomic" pieces of context to more complex interpretation algorithms.

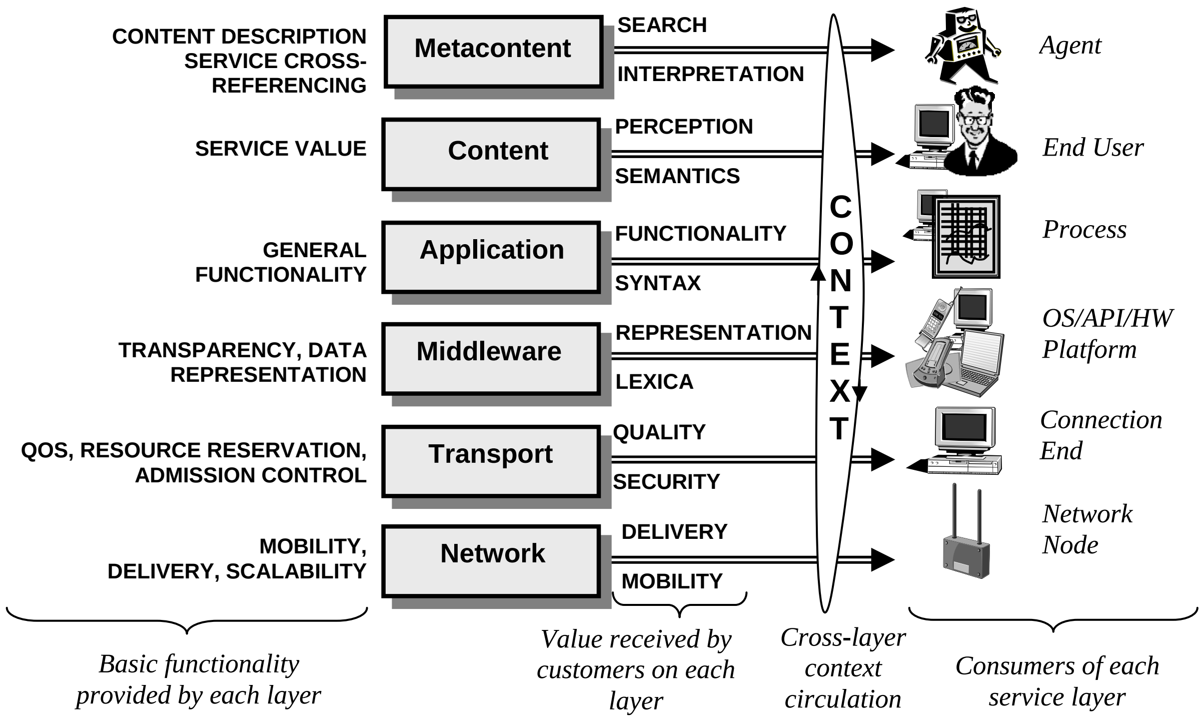

CONTENT DESCRIPTION SERVICE CROSS-REFERENCING
Metacontent
SEARCH
INTERPRETATION
Agent
SERVICE VALUE
Content
PERCEPTION
SEMANTICS
End User
GENERAL FUNCTIONALITY
Application
FUNCTIONALITY
SYNTAX
Process
TRANSPARENCY, DATA REPRESENTATION
Middleware
REPRESENTATION
LEXICA
OS/API/HW Platform
QOS, RESOURCE RESERVATION, ADMISSION CONTROL
Transport
QUALITY
SECURITY
Connection End
MOBILITY, DELIVERY, SCALABILITY
Network
DELIVERY
MOBILITY
Network Node
CONTEXT
Basic functionality provided by each layer
Value received by customers on each layer
Cross-layer context circulation
Consumers of each service layer

FIGURE 6 Ambient awareness

Our context–aware mobile environment architecture very much supports the idea of ambient awareness – context providers could be located anywhere in the network (environment) starting from very low-layer communication protocols up to intelligent services provided in the Internet. On the other hand, a context consumer can be any software or hardware object in the interconnected computing space. The core fundamental idea of this dissertation is that in upcoming ambient pervasive environments any information can be contextual and should be the way to utilize it.

2.4 The problem and research questions

There is a lot of work going on within the topic of context-awareness, especially in Internet research. However, not so much has been done with respect to domain-independent characterization of context as a concept, especially, considering that the networks can be heterogeneous, include different kinds of users capable of moving along with devices, various applications migrating throughout diverse devices and different kinds of devices moving across different environments.

Also, the problem of information processing is brought up once we assume that there are quite a lot of information available to applications and devices in the environment and that the information has a chaotic nature. Who will process the information? Some middleware system existing in the environment? This could well be the case, but who would guarantee in this case that the system would know exactly what this or that application needs? An application might be happy to provide the system with an exact request about the needed information, but how could the application be able to know its precise needs if it didn’t know what kind of information existed in the system? It seems that this problem could be easily solved by an empty answer to user request, but what if some relevant information nevertheless existed in the environment with some different identifiers or semantically near to the requested one? Having this information the application would possibly behave in a very different and much more appropriate way.

It is quite a challenge for modern telecommunications to bring the context-awareness capability to contemporary telecom applications. This is not just because the aim is to introduce more intelligent and adaptive applications, but also, and most importantly, because the nature of up-to-date telecommunication environments is highly dynamic, making instantaneous reaction to contextual changes quite desirable or even required by the user.

Considering migrating applications and heterogeneous environments, it is very easy to come to the conclusion that knowledge about surrounding situation could be very useful, in some cases – essential. On the other hand, especially, if environments are dynamic not only parametrically, but also by their structure (for example, in case of migrating applications support), it is clear that it is not possible to know precisely in advance (on a design or compilation stage), what kind of information might exist in an environment in which the device or application is operating.

For each situation there is a certain piece of relevant information, which would lead to the best decisions and, thus, actions; however, nobody can know, beforehand, what is that relevant piece of information. The system is not aware of the beliefs and intentions of the information consumer and, thus, cannot know what is relevant for that consumer. On the other hand, the consumer does not know the structure and content of the information that could be provided by the system. Therefore we can see that there is a need of dynamic collaboration between the information provisioning system of the environment and the information consumer based on mutual assumptions.

The current problem in information collection and representation area is that the majority of the existing context-aware solutions are domain-oriented and task specific. Due to this reason quite often a well-functioning context-aware engine cannot be easily applied to another domain or to a slightly different task. Context-aware applications use quite different information models, don’t have common information exchange protocols and, thus, are not able to exchange context information between each other or receive it from the environment if it is not designed specially for their needs. This work does not provide a new information model, instead it uses the RDF semantic network as a base for knowledge representation and context extraction.

We believe that the full potential of semantic networks has not yet been utilized in modern science and industry. Semantic knowledge is in text files stored somewhere in the web or computer. Adding semantic information to real data is very useful for automatic processing of that data; nevertheless, the world is not as static as to be properly described by text files updated once or twice a year. In our opinion RDF representation of data can be efficiently used for exchange of dynamic information between running applications and devices. Dynamic data can be supported by references to its semantics, which can be utilized when needed. This would considerably advance information exchange capabilities and make closer the aim of global interoperability. Therefore, the problem is how to make RDF networks dynamic and utilize them in real-time for a wide range of applications starting from low-level networking driver, going though end-user service and ending with a huge information server in the global network?

Concluding the above discussion, we can point out the following research questions of this work:

– How to model context not separating it from general knowledge?
– How to treat context in a universal way?
– How to take into account subjectivity of context information?
– How semantic networks can be used for efficient context retrieval without changing their structure?
– How to extract relevant context from information base taking into account needs of each particular consumer?
– How to split context processing between environment and consumer?
– How to take into account absence of predefined knowledge?

2.5 Brief outline and motivation of our approach

In this section we have presented a brief overview of several research and technological areas, within which the present work can be placed. We consider context-awareness to be one of key enablers of future Wireless Internet. One of the most problematic issues in context-aware computing is absence of common understanding of context as a concept, which can make collaboration and interoperability in context processing almost impossible.

The traditional research approach in context-aware computing is “from application to theory”. A research group may investigate the needs of a particular application or a group of applications and then design a theory or an architecture supporting the needed functionality. We believe that this approach doesn’t work with context because of the subjective nature of context – for each context-aware application there are many different perceptions of context.

In this work we try to look at context from a very basic, conceptual point of view. We consider context not as a set of properties of certain objects at a given time but as a relational property of any information and try to find a way of handling context's subjectivity.

Since there is no commonly approved reference model of future networks so far, we present our own heterogeneous mobile environment framework and use it for showing various participants and processes involved in context processing within such environment.

We selected semantic network formally specified by Resource Description Framework [66] as an underlying information model due to several reasons:

– It models, to a certain extent, the way humans process natural knowledge and, thus, context can be extracted from semantic network the way that is similar to the way humans acquire and process context.
– RDF and RDF-based languages are well specified and standardized for practical applications.
– Advanced research results in reasoning and knowledge processing have been obtained within the Semantic Web community, These can be used for dealing with context.
– Semantic Web languages have great potential to bring semantic interoperability into global networking environment.

We believe that the results presented in this dissertation advance current state in the field of context-aware computing. The rest of the work is organized as follows: Section 3 presents a conceptual view onto an environment for which some of the results of this work could be used; Section 4 considers context as a concept; Section 5 presents a model of context within a semantic network and introduces the principle of interactive reasoning; Section 6 provides a mathematical foundation for operating with the presented context model; Section 7 shows the main processes taking place within a context-aware application that would be operating within the presented framework; the work concludes in Section 8.

3 CONTEXT-AWARE MOBILE ENVIRONMENT

This section discusses the architectural view of the author on the environment in which some of the results of this dissertation could be applied. It is described here and not at the end of the thesis since we wanted to have a look at the problem from a practical point of view – starting from the needs. Having in mind the general picture significantly simplifies reading and understanding of the more theoretical and narrow considerations given in the rest of the thesis. In Sub-section 3.1 we present general architecture of the environment and outline its development goals with respect to its functionality. Within this section we consider several participants of the environment – in 3.2 we discuss context-awareness supporting agency, in 3.3 we present the architecture of context provider, in 3.4 we talk about context exchange protocol, 3.5 contains considerations about context-aware application. In Sub-section 3.6 we compare the environment presented here with other works and conclude in 3.7. Security issues related to CAME were left outside the scope of this inquiry, belonging to a different area of expertise. Interested parties can further develop the presented framework by taking up the issue of security support.

3.1 Overview

Context-aware mobile environment, presented in this section is a general framework idea for a development of interoperable context-aware dynamic systems. The wide development goals set for such kinds of environments consist of the following:

- Independent, simple and rapid development of context-aware applications.
- Flexibility of context provision mechanism – context providers may be developed by any developer and may be used by any application, not only by the application for which they were designed.

- Context-aware applications are functional without any context information. However context information would improve their functionality.
- Context-aware applications are not obligated to use context provisioning software. There would be a possibility of direct information exchange between context-aware applications and context providers.
- Independence of context-aware applications from the data format of the original context source.
- Context aggregation capability.
- Context domain independency – the technique designed can be applied to any context domain
- Possibility of unobtrusive provision by context provisioning software of some complicated functions such as complex context reasoning, inter-domain context exchange, context prediction, dynamic context source association, semantic interpretation of context situations, etc.

Ambient networks [5], [6] are based on the principle of general context-awareness – a network itself is context-aware and at the same time context information about the network is available to high-level applications and services. However, the authors of [5] claim that their context-awareness solutions are not applicable in current networks and need new standards and network structures. A brief sketch of the context-aware mobile environment is presented below. The main goal is to have a general architecture which could be applied on current networks and would utilize the theoretical outlines presented in this work.

Our *context-aware mobile environment* (CAME) relies on the principle of functional and implementation independence of each of its components. There are four components in CAME:

- context exchange protocol (CEP),
- context-aware applications (CAA),
- context providers (CP),
- context awareness supporting agency (CASA).

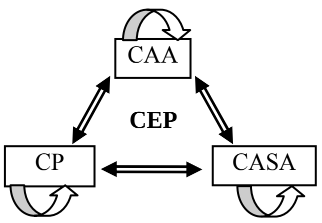


FIGURE 7 Information exchange scheme between the CAME members

The idea illustrated in Figure 7 consists in the following: each participant of CAME may be able to directly exchange information with any other entity of the environment including themselves using the context exchange protocol. A key issue is to have a common knowledge representation model and a common protocol to be used for context information exchange between the parties. CEP developed by Nokia [49] is used as an example, but in principle it could be any other suitable protocol.

The data model behind CAME is deemed to be RDF-based dynamically updated semantic network, meaning that the instance data is not permanently storied in text files, but is distributed along the environment, while links to particular data sources (context providers) are made available. Such approach for data storage and collection would enable global semantic interoperability. Figure 8 represents the idea of how such interoperability could be provided using a selected data model within a dynamic environment.

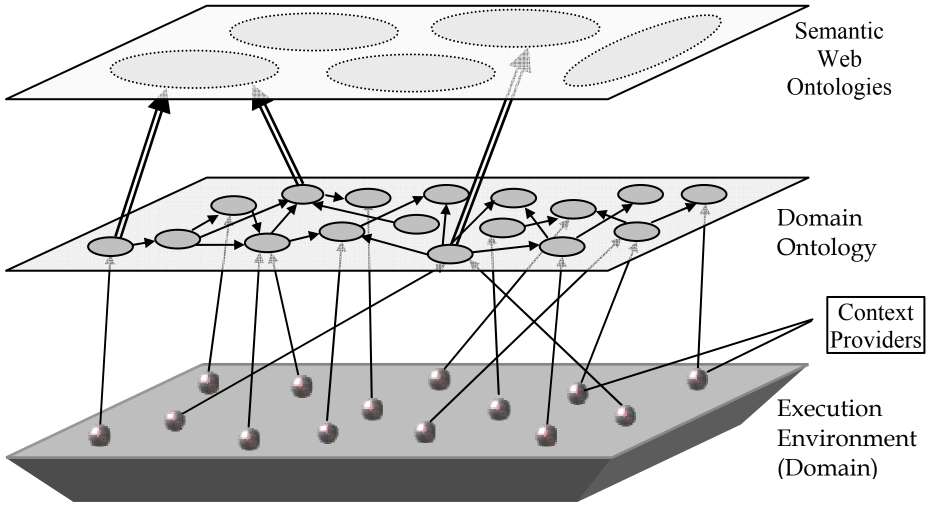


FIGURE 8 Context Providers within Global Knowledge

3.2 Context-awareness supporting agency

Context-awareness supporting agency is environment-oriented. CASA is not mobile; it has its home environment and permanent reference, which is public for CAA. When an active CAA starts context exchange session and does not have direct access to required CPs it must refer to the home CASA and negotiate over these issues.

CASA can handle context control and provisioning functions, which might be needed for context-aware applications. Each realization of CASA may provide a different set of services depending on the needs of the particular domain. The general list of functions which CASA could perform consists of the following:

1) Collect, store and provide information about available context providers to context-aware applications and external agencies.
2) Support applications with the mechanism of dynamic context provider selection (in the case of multiple available context providers supporting the same context variable).

3) Provide on-demand conflict resolution mechanisms in case of conflicting, redundant or poor context information.
4) Provide a communication bridge between context-aware applications and context providers when needed.
5) Support context information exchange between environments.
6) Support semantic annotation of existing context providers in the domain.
7) Provide context prediction mechanisms within the domain.
8) Take care of context provision and communication with correspondent CPs for passive context-aware applications.
9) Support the full version of context exchange protocol.
10) Provide context reasoning for complicated tasks, whenever applications are not capable of doing it.

The main and the only compulsory function is the first one; the rest is optional and depends on the implementation.

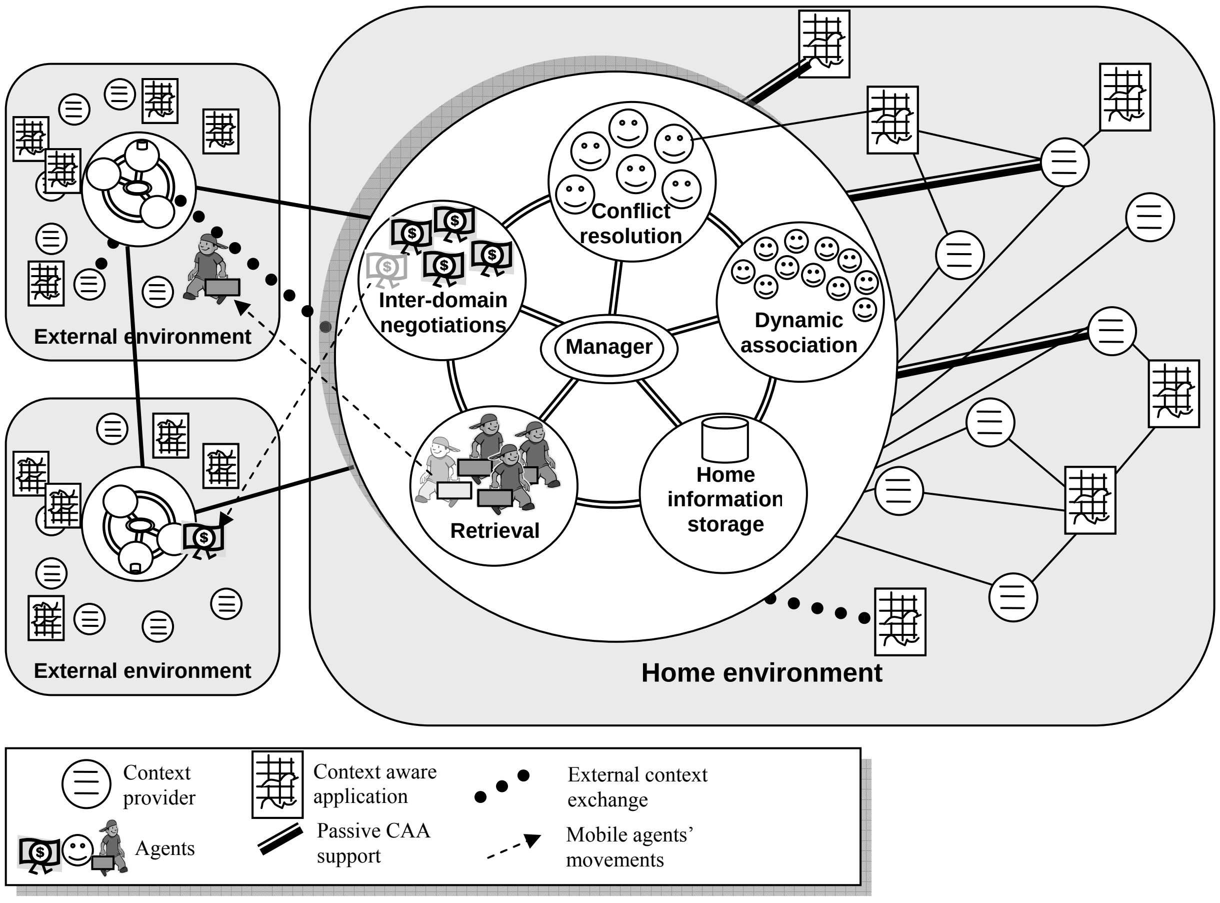


FIGURE 9 CAME interaction pattern example

Context-aware agency should not be seen as just one monolithic application. Figure 9 contains an example structure of CASA, showing the interactions within its home environment and with external environments. There is the main unit – the manager that handles short and simple tasks which are not time-consuming. Other functions are scheduled to task-oriented units, where either the task or CAA is associated with a special agent or is handled in some other way. Negotiations with external environments and global retrieval tasks are performed by specialized mobile agents. Different CASAs have connections to each other, however not necessarily all-to-all. The set of functions shown in Figure 9 were selected just to exemplify the concept and should not be treated as a complete set or as functions required for realization.

3.3 Context provider

Context provider is a general term used in context-aware computing. In different research areas it plays different roles and is given various functions. The basic function of a context provider in CASA is to transform context information from the context source (CS) data format to the form required by the context exchange protocol and to pass context information to the consumer.

Context provider is CS-oriented, meaning that the context provider working with some context source CS 1 may be incompatible with CS 2, even though they provide the same information (but in different output form). There is no special environment which would provide communication facilities between the context source and the context provider; the context provider should be able to acquire information directly from the context source. Furthermore, the context provider must support the data format of the context source. By transforming context information to the form required by CEP the context provider “makes it understandable” for anyone else using the same exchange protocol.

The CP is activated when there is a need for its work (some application may need context which the CP is able to provide). When the application does not need any more information from the context provider, it should notify it about the fact (release). When the last consumer using information from the CP releases it, the CP deactivates itself and switches to the “sleep” operation mode doing nothing or just monitoring context information in case it is still required to support context history. One context provider corresponds to one context triple described in Section 5.3. It provides contextual information (object) about an entity (subject) having a certain relationship (predicate).

Figure 10 on the next page schematically depicts some possible locations for context providers. We assume that CAME, at this point of time, would be widely distributed all over the computing world and many developers would have developed different context providers for various purposes. We suspect that the initial development of context providers would be aimed for particular context-aware applications which need concrete context information. However, since any application can use any context provider, with time a great many commonly used context variables would be provided by the already functioning CPs for the use of various applications.

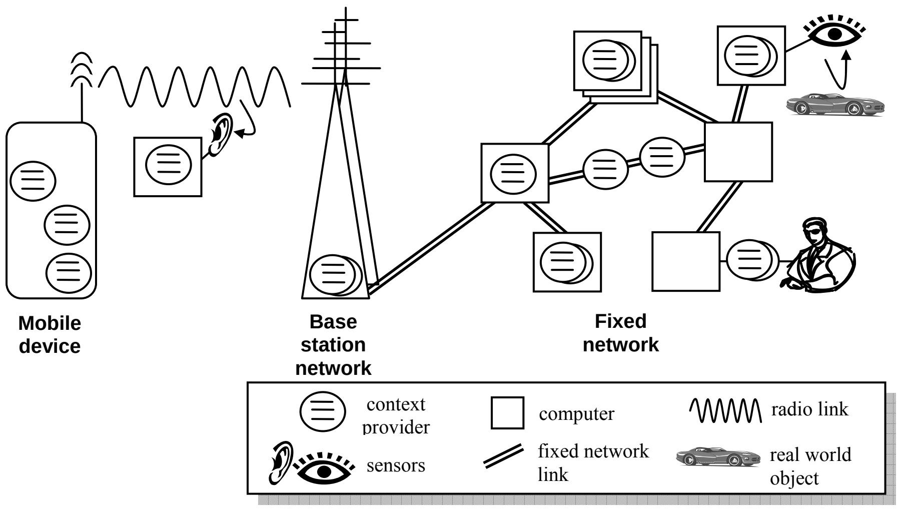


FIGURE 10 Schematic representation of possible context providers' location

The Figure 11 below represents the interoperability scheme between two different environments. Each environment has its own (environment-specific) CASA, which acts as a mediator (if needed) for information exchange between CAAs and CPs located in different environments.

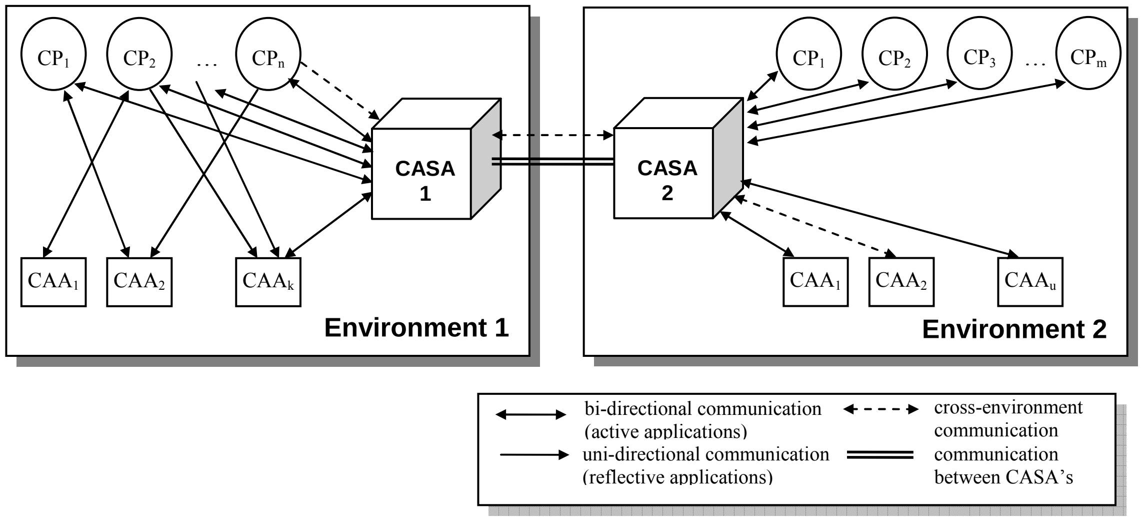

CP₁
CP₂
…
CPₙ
CASA 1
CAA₁
CAA₂
CAAₖ
Environment 1
CP₁
CP₂
CP₃
…
CPₘ
CASA 2
CAA₁
CAA₂
CAAᵤ
Environment 2
bi-directional communication (active applications)
uni-directional communication (reflective applications)
cross-environment communication
communication between CASA's

FIGURE 11 Interoperability between context-aware environments

Context history is a set of values taken by the context triples within a time period prior to the current moment of time (t, t-1, t-2, t-n). In many cases there is no need for context history support, but for some purposes it might be necessary. For context history support within CAME there are two realization possibilities:

a) CP itself does not support context history. In this case CASA plays the role of a context consumer and CP is permanently active. CASA is responsible for storing, accessing and providing context history information related to such CP.

b) CP supports context history. In this case all context history control is handled by the context provider. It provides context history information on-demand, upon receiving correspondent requests via CEP. As a matter of fact, such CP can not be in a completely inactive state since it should always be ready to receive and store new context data from the context source.

The complexity of a CP software implementation can vary a lot. This depends mainly on the nature of the context source. For example, obtaining context information from some protocol is simple (if it is not encrypted), but getting radio link condition information from the received signal sample is much more complicated.

The basic working principles of a context provider consist of the following:

- CP acquires information from the context source in the format of CS.
- CS is not meant to be aware of CP.
- One CP corresponds to one context variable.
- CP provides information in the form of CEP, the context exchange protocol.
- CP must support the basic set of CEP control statements.
- CP might support fully or partially the set of Quality of Context characteristics, but this is not mandatory.
- CP must provide a semantic description of the provided data in a form of context triples with a reference to the domain ontology (see Figure 8).

Figure 12 on the next page represents a conceptual structure of context provider with a complete set of possible operations and blocks. A brief overview of each operational block of the context provider (Figure 12) is given below.

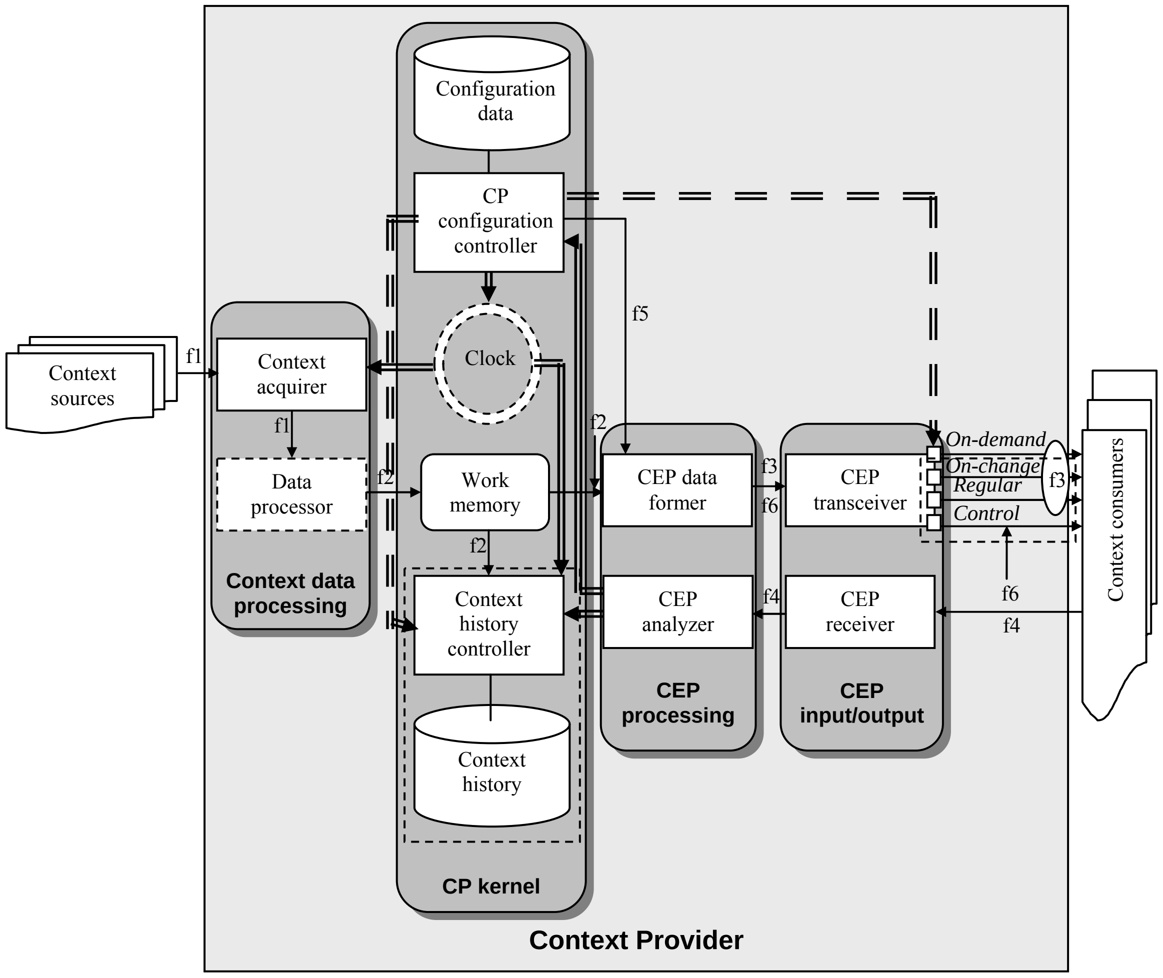
Context source(s) is an external unit for context provider. Context source can be a sensor, a device, an agent, an application, a software procedure (function), a user, a protocol, an object, a context producer or any other possible context source. CP is aware of CS's nature, but CS may be not aware about CP.

Consumer is another external object to the context provider. Consumer can be any entity supporting context exchange protocol, but usually a context-aware application, a context-awareness supporting agency or another context provider.

Context acquirer (CA) is responsible for getting information from the context source. This block is always dependent from the type of CS. There is no special environment for transport of context information from CS to context acquirer; therefore it is a task of CA.

Data processor (DP) is an optional block. Its task is to transform context information to the form of a context variable represented by a context provider. It is especially needed when it is necessary to extract some specific context from more general data, calculate some derived information or aggregate information from several context sources.

Context history controller (CHC) and *context history storage* (CHS) are responsible for tracking context history and provision of previous values of context variable on-demand. When CP does not care of context history these blocks are absent. Context history controller adds the new value of the context variable to storage always when the next one arrives. It deletes from the storage those values which are older than the defined history factor of the context variable.



f1 – context data in format of context source	CP – context provider
f2 – context data in inner CP format	CEP – context exchange protocol
f3 – context data in CEP format	⟹ control messaging
f4 – incoming requests in CEP format	⟶ data
f5 – non-context data in inner CP format	⬚ optional block
f6 – non-context data in CEP format	□ channel transmission controller

FIGURE 12 Conceptual structure of context provider

Clock is a synchronizing context acquirer, context history controller and CEP data former. The time slot of clock is equal to modification rate of the context variable served by CP. The clock is an optional block and not needed when CP is working always in on-demand mode.

Work memory contains current value of the context variable. It is changed by the data processor or context acquirer (when there is no data processing).

Configuration controller performs three basic functions – it reconfigures corresponding blocks when the reconfiguration request comes from a context consumer, gives information about the current configuration of CP when it is requested and controls configuration data storage. When CP is non-configurable (dashed control links are absent) – the configuration controller only gives configuration information on configuration requests.

CEP data former transforms context information from the inner CP format to the format of context exchange protocol, additionally it forms non-context (control) CEP-responds.

CEP analyzer is used for analysis of incoming CEP requests and sending them in the inner format to the corresponding CP blocks, which are responsible for their execution.

CEP transceiver and *receiver* provide connection between CP and context consumers. In case of a fully functioning CP, output channel is divided into four sub-channels: on-demand - context variable (CV) is sent if there is at least one consumer connected to this sub-channel; on-change - CV is always sent through this channel if its value is different from the previous value; regular - the value of CV is sent with a certain period of time; and control - for non-context information. Channel subdivision is optional, each channel can be enabled or disabled. There may be only one channel working for on-demand, regular or on-change modes. In such a case the same channel will be used for non-context information.

Depending on the implementation requirements CPs can differ with respect to the following properties:

1) Configurability

Configurable – configuration parameters may be changed dynamically.

Non-configurable – configuration is fixed and cannot be changed without re-programming and re-installation of the CP

2) Output pattern

On-demand – context data is received and provided on-demand

Regular – context data is received and provided with a certain time interval

On-change – context data is received regularly, and provided when differing from the previous

Combined – context data is received regularly, but provided on-demand

Complex – any combination of the previous setups within one CP

3) Activity

Always active

Active, when used

The conceptual structure of the simplest possible context provider is given in Figure 13.

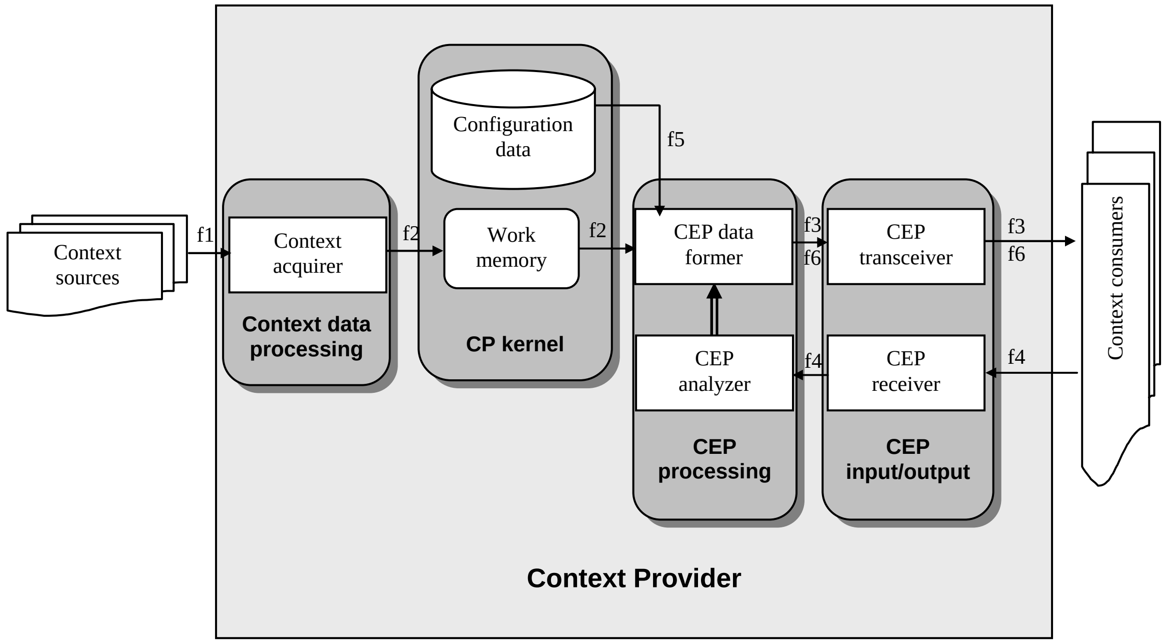


FIGURE 13 Conceptual structure of the simplest possible context provider

3.4 Context exchange protocol

The purpose of CEP is to enable information exchange between the components of CAME. It must include at least the following statements:

- request/response for the value of the context variable,
- request/response for meta-context information,
- activation/deactivation of context provider,
- request/response for available context variables (context providers).

The first version of the context exchange protocol has been designed by Nokia [49], and could be used as the basis. Each participant in the context-aware environment should support the basic or the extended version of the protocol. Applications support only the parts of the protocol that suffice for their needs; CASA supports a version of the protocol that covers its functionality.

3.5 Context-aware application

The basic idea of the context-aware application structure is in separation of context-awareness capability (CAC) from the whole application's functionality. CAC should have some influence on the basic functionality of CAA when it is needed, for example, for changing some variables or in function calls, but other parts of the application should not be strongly dependent on CAC. If there is no context information available, the application should work normally (“forgetting” that it is context-aware), but "normally" here would not necessarily mean “in the best possible way”. It is important to note that sometimes the absence of context information could be an important context factor for some particular context-aware applications; in such cases CAC should handle the situation. CAC should be designed in such a way that existing applications could be easily extended with this capability.

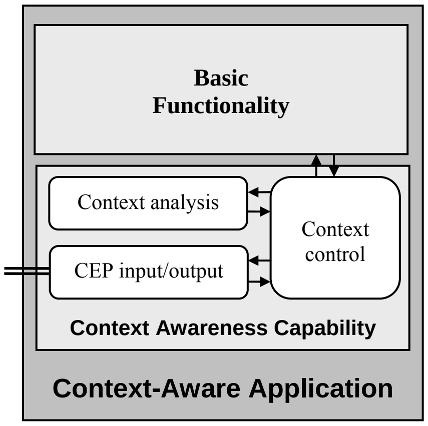


FIGURE 14 Logical structure of context-aware application

A context-aware application may be interactive or passive. *Interactive* applications are able to communicate with other members of CAME, at least with CP. Such application should support context exchange session with CP or CASA and release CP when there is no further need for its service. A *reflective* context-aware application has just a CEP input channel and processes any context information coming from that channel, disregarding its source. If the passive CAA is one-variable-oriented, it ignores all CEP messages other than context information about the variable.

The complexity of a context-aware application can vary a lot, as it happens in the case of CPs as well. CAA can perform complicated context-based reasoning, prediction, semantic analysis, etc. The simplest possible context-aware application would simply receive CEP messages from one CP and use information about one context variable provided by this CP.

3.6 CAME within context-aware environments

Currently, there is a huge variety of different techniques developed for context acquisition, processing, analysis, utilization and provision. CAME would be flexible enough to allow using most of them. The only requirement for the software to participate in CAME is to support the common communication protocol. Applications may directly receive information from context providers without any mediation engine, however, they can also use a mediation service when it is needed. This work does not specify the functionality of Context Awareness Supporting Agency in detail, therefore other scientific and practical results achieved in the area could be utilized as well.

The core idea of CAME is that context-aware applications should not be obliged to interact with the whole system, but, nevertheless, be able to utilize all the functionalities it provides. These applications must be served according to their needs and not any further than that. The future systems will introduce mobile software applications which are able to migrate between devices and environments. Would there be any reason for an application to analyze the domain ontology if it came to the environment for a couple of seconds only and needed to know just a very small amount of the context information? On the other hand, some applications would probably not be satisfied with the reasoning that the system may provide and would want to draw their own inferences – why not to allow that to them?

Another important aspect is that the use of a uniform context exchange protocol allows cross-layer information exchange. Context providers might be located at any layer of the networking functionality and provide information not only for applications and other CPs, but also for low-level networking software and hardware that have common communication channels with CP and support context exchange protocol. Furthermore, context information from applications could be passed to lower network layers, and in some cases this might even prove quite useful. Of course, CEP [49] in its current version is not completely satisfactory for such global information exchange, but it could be further developed or other protocol could be selected instead.

Since the basic idea of CAME is quite general and domain independent, it is difficult to compare it with monolithic, domain-oriented middleware such as [19], [21] and [75]. An important future direction of the research on CAME would be its implementation in different computing environments utilizing previously achieved environment-specific research results in context-aware computing.

For example, [36] provides an interesting logic for dynamic selection of context provider. In this work, in addition to getting information from context sensors, CP also performs some context interpretation and provides higher-level context. This idea is not completely contradictory to our idea about using a small amount of provided information, although the "amount" might be a little bit more in this case.

The Gaia Context Infrastructure [61] seems very similar to the framework presented in this work. Many different data-oriented context providers have been developed within the Gaia infrastructure, intelligent agents are used for performing different context provisioning services, and different reasoning mechanisms are used as well. At the core of Gaia resides a triple data model resembling the one in our work. CAME is, in principle, different from Gaia in one essential property – it has open module architecture. The parts of CAME interact through an open interface and allow any interesting party to use the different services of the system. Recent research achievements towards support and use of context develop, implement and verify different functionality aspects that are presented in this work within CAME as a conceptual framework.

In [67] and [68] the authors present an approach for context-aware service provisioning within mobile devices (smart phones) and the way for efficient exchange of context information between interested parties. The authors combine different approaches for context-awareness support and select the most appropriate for service provisioning depending on the situation. Their idea of integrating multiple strategies for context provisioning very much corresponds to the conceptual idea of open context-aware mobile environment presented in this work. The context information should be collected, managed, stored and provided depending on its nature, needs and capabilities of the requesting applications and characteristics of the particular environment in which such environment functions. The important issue is that different environments should be able to exchange their specific context information to interested parties in other environments.

In [64] the authors present MSDA – a middleware system that supports context-aware service discovery in heterogeneous networking environments. The presented middleware function through different communication networks and discover services available for the user in its current situation taking into account also networking context. The general architectural solution is very similar to the one presented within this dissertation – each particular communication network is granted with its own MSDA representatives, which take case of context acquisition from the environment and propagation of it to other interested parties (context managers). The architecture is nice and well-developed, and its strength is in that it supports cross-platform context-aware functionality. However, service discovery supported by MSDA is only one service which is needed in pervasive environment. What would happen in a pervasive environment, if each context-aware cross-platform (even very nice and useful) service acquired, propagated, stored and managed context information each in its own manner? MSDA is a middleware, and the difference between an ordinary service and such middleware is that the latter provides services not to end-users, but to applications and, most likely, does not have user interface.

We believe that if some common context provisioning infrastructure would be available for MSDA, it would benefit a lot from it – context acquisition and management would not be anymore the MSDA responsibility, and MSDA would be able to concentrate on its primary function – service discovery. The way the context is managed within MSDA might be very efficient and could be used, for example, in a future context provisioning system, like CAME. However, our strong belief is that it should be decoupled from individual applications into some common, globally available infrastructure.

The problem with the currently existing context-aware applications (usually, prototypes) is that each of them has its "own view" into context and "other's view" (for example, some context-provisioning system's) does not, usually, correspond to the needs of each concrete application, thus, cannot be utilized. The main aim of this PhD dissertation has been the introduction of the idea of transformation of "global context" (that is, in fact, just the usual information space) into small subjective pieces of information, which would be contextual for concrete cases. A set of techniques and concepts has been developed supporting such view onto context.

3.7 Conclusions

In this section we presented the concept of context-aware dynamic mobile environment, in which all components are working independently from each other. The concept of context provider was studied in depth, and the functionality requirements to context provisioning system (context-awareness supporting agency) have been clarified. The environment was presented mainly with the purpose of providing the reader with a vision on the issue – how the global context-awareness capability could be provided by the environment to all interested consumers (applications, services, agents, etc.).

The theoretical outlines given in the rest of the dissertation are aimed to be a base for different functional parts of such environment. Of course, it is not possible to study, within one dissertation, all the aspects and functioning modules of such system in detail, therefore the scope is sharpened to the study of questions which most interest the author and which are not clearly clarified within current science and technology.

4 CONTEXT AS A CONCEPT

In this section we present a general understanding of context. In a discussion presented in Sub-section 4.1, we consider contextuality as a relational property of information. Relevance, being a core property of context information, is discussed within Sub-section 4.2. We present a philosophical model of context in Sub-section 4.3 and its general properties in 4.4. Such important aspects of context information as its precision, formality, probability of correctness, up-to-dateness and others are dealt with in Sub-section 4.5. Context-awareness as an application capability is considered in Sub-section 4.6. Within Sub-section 4.7, we introduce interactive context-aware reasoning principle. The Sub-sections 4.8 discusses the issues of context information storing. Sub-section 4.9 concludes the section.

4.1 Contextuality as a relational property

Let us assume that we have the whole world described via a semantic network. What is context? Context can never be defined absolutely – "this is 'context', but this is not. Context is always defined with respect to something – "context of something". Furthermore, we can not know in advance the context for a particular situation. In this work we promote the view that any kind of information may be contextual for some particular cases.

In this thesis we speak about *subject of attention*, where context characterizes the situation of the subject of attention. By subject of attention we mean a subject of action or analysis. A subject can be information, object, event, person, situation, application, equipment, logical entity, etc. Here it is important to notice that the subject itself is not context information - however, it may be a context for another subject. The set of information which we consider to be a context depends on the point of view – on the subject of attention.

Many researchers, especially from the location-based services research area, would say that time, location and identity of the user are always part of this context information. There are some situations in which we have the user, but time and location do not form a context for the application which is interacting with the user. For example, the task of a watch, which is a mobile application, is to show the time and nothing else. However, if the watch needs to be adjusted when the user moves through a time zone, the user location is a context for the user of this application.

Let us consider the situational model shown in Figure 15: “Steve is lounging on a beach chair on the beach at 2 pm”.

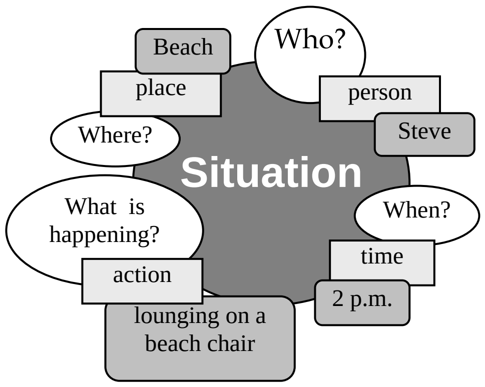


FIGURE 15 Example situation

Case 1: For a Steve’s friend, who calls Steve telling him that a meeting is to be held the following day, the subject of attention is Steve. Everything else is something additional about Steve, naturally; the context is unknown to the caller and might be not interesting for him (Figure 16a). Of course, he can ask Steve about Steve's context, including his whereabouts (on the beach), and even come and keep company, but this would have no connection with the purpose of his call.

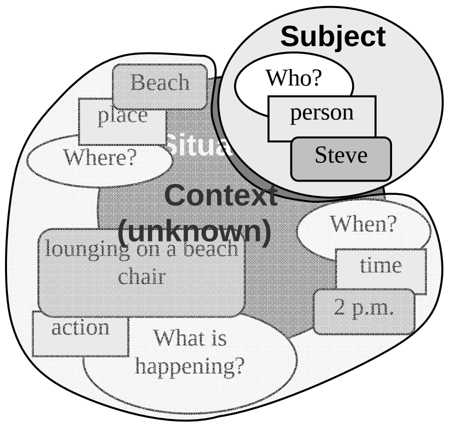


FIGURE 16a Context-subject relationship (case 1)

Case 2: The beach chair is broken and a repairman comes to fix it. He is not interested about Steve or about the time at that moment, he just observes that somebody is lounging on the beach chair (Figure 16b). He may, among other things, ask Steve's name and politely implore him to change to another beach chair (in this case he would show context-aware behavior), or, in a less likely case, throw him down and start repairing the beach chair disregarding his presence.

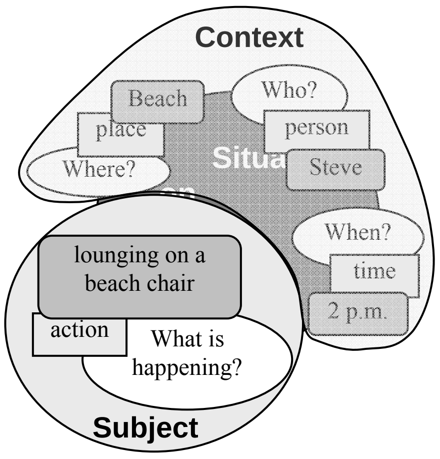


FIGURE 16b Context-subject relationship (case 2)

Case 3: When Steve's friend told to their boss that Steve was lounging on a beach chair on the beach, the boss became angry because at 2 p.m. Steve was supposed to be at his work place. He called Steve to inform him that he was fired, but when Steve explained that his leg was broken and that he could not stand up, his boss calmed down. In this case the subject is the situation itself, which could provide a justified reason for firing an employee (Figure 16c). However, context information helped Steve's boss to make the right decision.

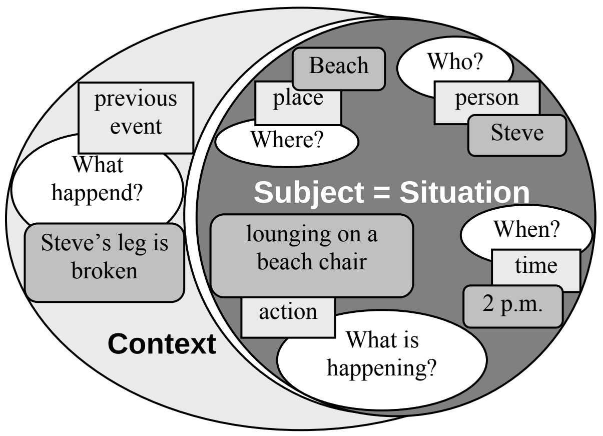


FIGURE 16c Context-subject relationship (case 3)

These examples show us that the domain information clearly contain two different parts:

a) subject of the main attention – information that is initially known by somebody, who is analyzing the situation
b) context – information that has some semantic relationship with the subject of attention; context might have some impact on the decisions and actions of the person, who is analyzing the situation, but might be of no interest; context available in the domain might completely satisfy the information needs of the analyzer, but might be not enough.

From the above examples we can also see that the division of the information into content and context completely depends on the piece of information that the context consumer has – all information having some semantic relationship (not only direct, but also distant) with content (except content itself) is a context.

Therefore, let us define the terms *context, context consumer* and *content* as we consider them in this work:

context of the given piece of information is any information having known direct or distant semantic relationship with it

context consumer – a person, software or hardware entity analyzing the information domain and able to receive and use context

content is a subject of the main attention of context consumer – the information, which is initially known by the context consumer at the beginning of context analysis process

Further discussion around this definition is deferred to Sub-section 4.3.

Despite the fact that the domain might have a lot of information semantically related with content, it doesn't mean that this information will be necessarily interesting and useful for the context consumer. Thus, we come to the question of context relevance, which is discussed in the next section.

4.2 Context relevance

When analyzing the examples above with respect to the relevance of context information, one can come to a conclusion that relevance depends not only on the content, but also, to even a greater extent, on the context consumer. If two different context consumers had the same content, they might look for different contextual information depending on their goals and intended actions.

The discussion above brings us to the main question: how to define the context relevance in each particular case – which information is substantial (predictive), which is contextual, and which is irrelevant? These questions suggest that in this case the definition of context relevance may be the most important for further context processing. Edmonds in [25] also considers this problem: "for at any particular time the participants will need to have chosen an appropriate internal context from the context-as-a-resource in order to correctly interpret the communication".

Since the perception of context as we have suggested in this thesis is not yet prevalent in context-aware computing, the implication is that so far the question of context relevance has not been strongly addressed in context-aware computing. Many context reasoning techniques are based on the assumption that there is a certain key piece of context, which can be used for reasoning purposes. Others consider all available context information. Those oriented towards some particular task(s), define a certain set of context parameters suitable for their needs, for example [30].

Strang *et al.*[80] pay particular attention to context relevance. They suggest the following set of definitions: "A context is the set of context information characterizing the entities relevant for a specific task in their relevant aspects. An entity is relevant for a specific task, if its state is characterized at least concerning one relevant aspect. An aspect is relevant if the state with respect to the aspect is accessed during a specific task or the state has any kind of influence on the task…The situation is a set of all known context information." This description of relevance seems a little bit weak. First of all, it is quite hard to define the kind of influence the state might have on the task. Potentially, any information may have influence. Second, "if the state…is accessed during a specific task" – it can be clearly seen that some information (states) might not be accessed in a normal execution process. However, it is precisely in those cases where context-awareness is especially needed (for example, in rare failures) that this information might be the most relevant out of all the available information. The definition given is thus not suitable for such cases. And third, "all known context information" might consist of a quite large amount of data, especially if there is a possibility of inter-domain context exchange.

In [34] the authors directly enumerate relevant context parameters with respect to different players in the world of mobile devices, e.g. relevant context for mobile devices includes basic hardware features such as CPU power, memory and user interface, network connection characterized by the bandwidth, delay and bit error rate, and user specific information primarily consisting of user preferences.

We suggest the use of the BDI (Belief-Desire-Intention)[95] model from the theory of intelligent agents. The context consumer has its BDI and is looking for context information which would describe more precisely the content (which usually is part of its "beliefs") and be relevant with respect to his BDI. Therefore, it is reasonable to conclude that context relevance depends on the content and BDI of context consumer. Naturally, if BDI of the context consumer changed, the relevance of context information might also change.

Therefore, we come to the definition of context relevance:

context relevance is a function defining significance of the context information as one of its main arguments to the context consumer with respect to the given content

From our point of view the most suitable range for context relevance function would be [0,1], where "0" means that the information is irrelevant, "1" means that the context information is definitely relevant to the consumer with respect to the given content. In principle, context relevance is, most likely, continuous – some piece of information might be "a little bit more relevant" than another piece of information. However, in this dissertation we suggest the use a simplified model for context relevance, a model that originates from the area of data mining. The large set of available attributes of some object is divided into three parts: predictive – affecting the result alone; contextual – having affection together with some other attributes; irrelevant – having no relevance to the goals of analysis as described in [45].

Since we are talking about context, it would not be aesthetic to use the term contextual context, that's why we substitute the term and use *secondary context* instead. Thus, we come to the definition of context relevance range values used in this work. The given set of context information can be divided into three parts:

a) predictive context – information, definitely relevant to the current BDI of a context consumer;
b) secondary context – information, which might be relevant to the consumer's BDI if some additional facts would give enough information for such conclusion;
c) irrelevant context – information, which is irrelevant according to the current situation of the consumer.

4.3 Philosophical model of context

Considerations above give enough background for building a philosophical model of context. Let us assume that we have an information domain with full semantic connectivity, meaning that any information has some kind of semantic relationship with other information directly or through yet other information. The physical world is like this – it is always theoretically possible to find some connection between two very different facts even though this connection would be semantically very distant. The reason, why in real life some facts could be considered being completely mutually irrelevant is that the information analyzer does not have access to the whole world's information and is not able to build a semantic connection.

Let us divide the available information domain (ideally – the world) to content and context. We can not do it in general, but we can do it, assuming that we have a concrete consumer of information (be it a human, agent, software, device, or whatever), which has in mind some particular goals to be realized, possesses pieces of information and intends to do something (BDI – beliefs, desires, intentions).

The context consumer knows or requests some very definite, clearcut information known to it beforehand and required by it for performing a task – this is content (we can call it also "focus information"). All other information, even if available, is not as essential as its focus information for it. However, if the context consumer assumes that it could perform its tasks better (more precisely, faster, simpler) had it some additional information about the content it already has in its possession, the consumer might be interested for that additional information. It might request for all available information, which, of course, cannot be satisfied in a real situation (the domain would prove too large). More realistically, the consumer might request some of the most relevant pieces of information, which might tell it more about already available content. Then, maybe a little bit more, and so on, depending on the situation, i.e., how much time the consumer has at its disposal, how well it could understand the information received, its interests, goals, etc.

So, in principle, context is infinite within the domain (all semantically connected information potentially can be relevant). However, we clearly see that context may be more or less relevant (interesting) to the context consumer with its BDI. Figure 17 below visualizes the relationships between the main players of context analysis: context consumer, information domain, content, context and context relevance.

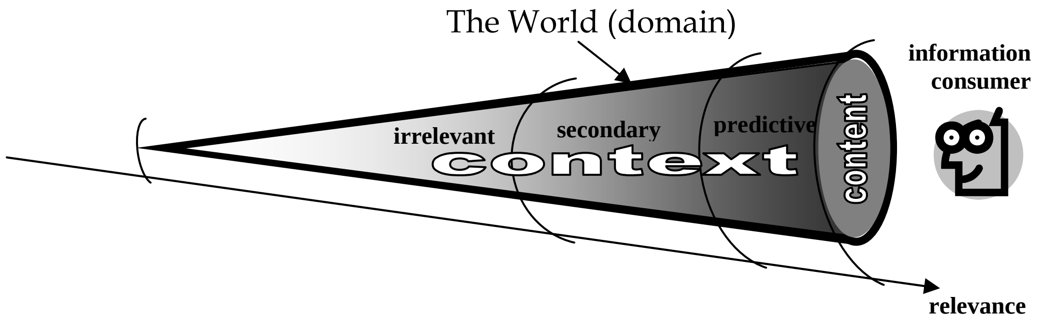


FIGURE 17 Context perspective

So, basically, all information exists with its semantic relationships independently of the information consumer. However, each consumer considers this information from a different perspective in each different situation.

4.4 General properties of context information

Having analyzed research devoted to context along various research directions, we can come to a conclusion that context by its nature is nothing but "information" as we usually understand the word. Furthermore, it is not even any special kind of information, since it is easy to find cases where some set of information can be regarded as “context” in one situation but not in another one. Leaving aside any particular context research areas it can be stated that context information is only logically different from the rest of information and, hence, inherits most of the general information properties, including diversity, quality, completeness, correlativity, actuality, structure, etc.

Having such understanding of context now, we may come to the definition that “context is everything”. However, it is not so, since we are able to distinguish between context and non-context in particular situations. The most important concept here is “particular situation”, which allows us to split the information to main, surrounding (additional, contextual) and non-relevant (which, however, exist in some form and can be referenced as information about the rest of the world) information.

Paul Dourish [24] makes the following assumptions about the nature of context:

1) Contextuality is a relational property that holds between objects or activities.
2) The scope of contextual features is defined dynamically.
3) Context is an occasioned property, relevant to particular settings, particular instances of action, and particular parties of that action.
4) Context arises from activity.

These assumptions reflect our own point of view and lead us to the conclusion that *context is not a special set of facts about concrete entities or environments, but an abstract set of information having a special relationship to all other information.* In describing this “special relationship” we define the following general properties of context information.

The first property of context is its *dynamicity*. The context is a set of information, and dynamicity does not refer to the dynamic nature of the information in this set, which can be static or dynamic, but to the dynamic nature of the set itself. This means that the cardinality and contents of the context set may change due to various factors, such as time, task, dynamicity of the domain, reasoning conclusions, etc. The dynamicity of context means that the relationship between contextual and non-contextual information cannot be defined in advance once and for all at the design stage, excluding those cases where context-aware system is single-task oriented.

The second property of context information is its *subsidiarity*, which means that, in principle, using of context information is not obligatory. Context information expands and concretizes information about the subject. Only as much context as is necessary for a better understanding of the main subject needs to be taken into account. Here we do not treat context as part of the semantics of the subject, as it does not respond to the question “what the subject is”, but tells “where, when, how, why, etc.”

The third property of context information is its *expandability*. In many cases there is the possibility to expand context information with some additional facts. Even when the whole information available at the domain is already involved, there may be a need for another domain to support it with additional information.

Relevance is the fourth property of context information. Relevance limits practically and potentially available context information with respect to concrete needs. It is defined by a particular task or situation, e.g. some application function (which information about the user is relevant for building a suitable user interface?). For example, let the subject be the situation “the user is sitting in front of the computer”. What is the context in this case? The context information gives answers to the questions: “Who is the user?”, “What kind of computer is he using?”, “Where is the computer located?”, “What is the user doing?”, and “Is he alone?” among many others. It is not possible to enumerate the whole context surrounding this single subject. However, it is possible to limit context information about the subject by its relevance to some particular case, e.g., for the user’s wife it may be relevant to know whether the user has cleaned the apartment.

One more property of context is its *centricity*. This property actually represents our idea of subject orientation: context information is always surrounding some central concept, subject, or entity. It is not just a simple set of facts, it is a set of facts characterizing the situation of a particular subject. The further the context information is semantically from its center the less relevance potential it has.

The presented properties are aimed to characterize the logical difference between contextual and non-contextual information. The correspondences between the assumptions of Dourish [24] given above and some of the properties introduced here can be stated as follows: dynamicity reflects the second and the third assumption, expandability is related to the second, relevance almost entirely corresponds to the third, and centricity can be considered as supporting the first, the second and the third assumptions, presented by Dourish in [24].

The aim of [24] is to reconsider context as a problem of interaction, but not that of representation. The main idea is not to find out “what is context and how it can be encoded?”, but to propose an interactional model of context which answers the questions “how and why, in the course of their interactions, do people achieve and maintain a mutual understanding of the context for their actions?”. In [24] the attention is directed to the analysis of action consequences and prerequisites with respect to its context and to the difference of context understanding between different people. Our research work reflects a similar understanding of the nature of context, but goes along another direction and mainly treats context as a relational problem (that is why the discussion about the fourth assumption from

[24] is left out from our scope). We try to justify the relationships between content, context and other information. However, like Dourish, we do not separate these, but try to find principles for selection of relevant context out of the whole information domain with respect to the task and particular circumstances.

4.5 Contextual information measures

Practical usage of context information sometimes requires some additional knowledge about the nature and properties of context information considered including knowledge about the types of values, source of information, etc. This kind of information may prove essential, for example, in the following tasks:

– resolving conflicts of data in a context reasoning process,
– interpreting of context source semantics online,
– interpreting unknown contexts.

There are two levels of context information: the context itself and its meta-context that is information about context data (the context of a context). Let us consider a simple example of a context statement:

"the network of the user1now is GPRS".

In principle, this statement alone gives us the needed information about the content – *user1*. However, we might need, in some situations, some additional information about the context information, e.g. "what are the networks available to the user" (a set of possible values), "when is 'now'", "how old is the information", "how sure is the source about the reliability of information", etc. This kind of information is particularly important for conflict resolution during a context reasoning process; however it also might be needed for context reasoning itself.

Recently, considerations about that kind of information have appeared in the literature under the name of Quality of Context (QoC) [5], [15]. In [15] the QoC is defined as follows: *Quality of Context is any information that describes the quality of information that is used as context information*. Thus, QoC refers to information and not to the process or hardware component that possibly provide the information.

The different aspects of Quality of Context are the subject of interesting discussion in [15]. The authors suggest the following set of the most important QoC parameters from their point of view:

– precision – how exactly the provided context information mirrors the reality,
– probability of correctness,
– trustworthiness – how likely it is that the provided information is correct,
– resolution – granularity of information, and
– up-to-dateness – the temporal validity of context information.

Despite the fact that there are such parameters as resolution and precision, there is no place for many other parameters, although some of these could be quite useful, for example a set of possible values (if context data is not continuous). On the other hand, precision, for example, would make no sense in the case of computing context with enumeration data types. It seems that the authors' perspective is more oriented towards physical context, which is usually acquired by physical sensors. In [36] the parameter *refresh rate* is considered in addition to others, which are the same as in [15].

In [5] the necessity of QoC definition is brought into limelight and parameters such as *precision*, *probability of correctness*, *trustworthiness*, *resolution* and *validity period* are considered. In Context Exchange Protocol [49] the basic unit of context information is context atom. Some of the parameters there, which could be classified as QoC, are *modification timestamp* and *confidence*.

An extensible ontology for quality of (context) information is built in [31]. The four most common parameters considered are: *accuracy*, *resolution*, *certainty* and *freshness*. The ontology itself seems to be clear and useful, but its utility needs further investigation not presented in [31].

Based on the literature analysis and our own experience we can conclude that the set of QoC parameters very much depends on the nature of the data they describe. Investigation of this problem is still open and is also out of the scope of this work. In [15] the necessity of QoC is introduced, the term defined, and the relationship between Quality of Context, Quality of Service and Quality of Device shown. However, the QoC parameters themselves are not explored. This question is out of the scope of the thesis as well. Nevertheless, considering problems that are essential to solve for context-aware computing, we would like to raise these questions to QoC research: What can be the fullest possible set of QoC parameters? What kinds of parameters should be applied to each particular type of context information? Which QoC parameters are common? What is the essence of each QoC parameter and how much it influences context analysis and utilization? How and at which cost different QoC values could be obtained? How tracking QoC parameters could complicate context-aware supporting systems?

4.6 Context-awareness of applications

Let us think once more about how people get and treat context information. A newborn baby boy is not able to use even his hands, as he doesn't know what they are, nor does he deliberatively react to sounds – in other words he is not able to analyze context. In principle, he perceives surrounding information, but is not yet able to understand it. Step-by-step, by gradually getting more and more experienced about the world and about his place in it the small child learns to think, analyze and use the perceived information – to be more and more context-aware. Humans are relatively universal systems and after years of experience they are intelligent enough to obtain and use huge amounts of context information. Applications are much more specialized and usually need to be able to use only a certain set of possible information about the context. Therefore, for the majority of applications it is not necessary to have learning capability or understand the whole context in order to use some part of it - however, they should be capable of receiving the context they need.

Based on the above considerations let us define context-awareness for applications as a *capability of an application to receive and use context information.* Receiving information does not necessarily mean acquisition – if an application does not get context information directly itself, it must have an input channel to be able to receive the context from some other source.

Nowadays, applications are so complex that, according to our definition of context-awareness, almost any application can be called context-aware. These applications usually have a plethora of subjects of attention and a lot of information describing their state and the situation around. Nevertheless, they are not called context-aware. In practice, in current understanding *context-aware application is an application which is able to receive and use context information that is initially located outside the application. A context-aware application is able to perform its functions correctly without context information.* However, "correctly", does not mean "in the best possible way".

Consider now two adults watching TV news. One of them is a professional economist; the other one a worker of the plant X. The same news about selling the plant abroad is treated very differently by each of them: the worker may be concerned about workers losing their jobs, while the economist looks at it from the viewpoint of finance. Not only their treatment varies, they also get different *context information* about the main subject, of the plant being sold: the economist pays more attention to the costs and revenues, the worker is concerned about the reasons of the sale and future plans of the new owner – which gives him an idea about the number of job positions to be reduced. When the worker's wife enters the room, she merely hears something about the sale. She becomes interested and starts asking the men for additional (context) information – "what is sold", "why", "when", "to whom", "for what price", etc.

Based on this example it is reasonable to consider two types of context acquisition:

– *requested* – the needed context information is requested from the context source,
– *selective* – the relevant context is selected from the available context information.

This categorization compares well with the traditional subscribe/query model in that it takes into account the specifics of context information much better. Although there may be a lot of information in different forms (real world information, different computer formats and information exchange mechanisms) around any subject, quite often there is no infrastructure to provide a subscription capability. The question of context acquisition in such cases concerns obtaining a set of relevant information for a particular task (selective context acquisition). Another frequent situation is when the relevant context information is stored inside some context source with communication capabilities but which cannot be directly accessed – in this kind of cases the party interested in the context information should make a request about it to the context source (requested context acquisition).

Let us consider two degrees of application context-awareness – *low* and *high*. Applications with high degree of context-awareness can be called context-oriented or *context-based applications*. Their major functionality is based on context information. This becomes apparent if we look at some of the examples. In location-based systems geographical information is associated with physical coordinates as it makes no sense having information about some place without any knowledge of its location. A user personal assistant may perform actions such as timetable scheduling, act as meetings and interactions organizer, issue reminders etc. - this type of utility can not work without information about what the user may be doing at any one moment, about what he is going to do, what his goals and preferences are, that is, user context information is needed.

A low degree of context-awareness (*context-aware applications*) means that application usually performs some task which is not directly concerned with context. However, in some circumstances the application takes into account changes of context and has to adapt its execution with respect to it. For example, there may be a videoconference service supporting video, audio and text communication. Let's assume that the service can track the user moving between different terminals (for example, let's assume he travels to work by bus where he uses his mobile phone, then while at work prolongs the conference with his laptop). Initially, the user has only a voice connection, later on a terminal with a full set of capabilities. In this example, the basic functionality of the service is to organize a conference and to provide connectivity between the conference participants. As soon as one of the participants moves from one terminal to another the conference service must take into account this context change and reconfigure its functionality with respect to the change.

4.7 Interactive context-aware reasoning principle

Let us think about how people are using the word "context" in the real world. For example, your friend may occasionally tell you: "Mary got married last summer". If you don't know anything about this fact, you ask for CONTEXT: "Which Mary, your sister?" If the answer is yes, you keep asking (you know Mary and you are interested): "Where was the wedding? Who is the husband? Is he rich?", If it's another Mary, you might just reply "I don't know her" and would not talk about the initial fact anymore – you would not be interested.

Why to ask? You asked the first question, because you didn't know how to interpret the information – you are building the missing associations to the received fact with knowledge that you already have. Your additional questions arose after you had received the initial answer and realized that you were interested in knowing more. In principle, disinterested outsiders would clearly see that you were after context information. You definitely had reasons for asking, although your friend might not have been at all aware of your BDI, and you might not have been sure that you would receive answers to all of your questions. However your friend is able to provide you with some context information as long as he has at least part of the answers to your questions. On the other hand, you could just say "Tell me more about it" and your friend could give you a part of the context information, guessing on what kind of information you might be expecting.

In practice, when people think about context, they don't think that, in principle, everything can be a context. They consider context to be the most relevant information according to the situation (which includes their beliefs, desires, intentions at a fairly precise moment of time and regarding the concrete subject of attention, in other words, content). On the other hand, even though they know their BDI, they don't have all the context information available for them at first. They don't know, what is relevant, they just assume and ask, and only when they receive answers to their questions, they can decide whether it is relevant or not, and whether they already have enough context information or not.

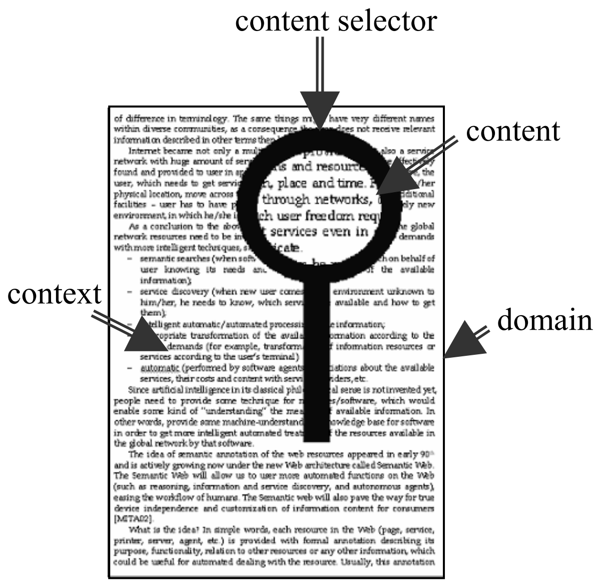


FIGURE 18 Content selection

Let us build the following association (see Figure 18). Content is a container with an adjustable size, a container which is filled by some content selector consumed by a content consumer. A content selector can be a person, software, agent – anyone, who can select, based on any kind of assumptions, some small part of information out of the whole domain for any kind of purpose. Content can be also selected by a content consumer. Therefore, content is information under focus of attention (subject of attention).

Consumer's beliefs are part of the world. Content is another part of the world. If the information that the consumer already has is enough for analysis and use of content, the consumer does not need any additional information. If it is not enough – by asking for additional context information about content, the consumer can build links between content and his beliefs based on intentions that he has. Therefore, the consumer receives information needed for his reasoning piece-by-piece – interactively.

The interactive reasoning principle presented in this section is based on the following assumptions:

- Content is a piece of information being known by and under primary attention of, at a certain time, a concrete information consumer.
- Relevance is a function, which structures all information except content with respect to the BDI of its (information) consumer and content. In other words, relevance function defines how relevant is this or that information to the content regarding the consumer of content.
- Content is one (small) part of the world (domain), context is its bigger part, and has semantic connections with content.
- Consumer (and its BDI) may be a part of content or context or both
- Information is asynchronously dynamic (each unit of information has its own dynamicity level).
- Information is represented as a set of RDF statements (semantic network).
- Content and context are sets of RDF statements.
- Information system does not know consumer's BDI.
- Consumer is not aware of context.

We will never know the relevance function exactly, because we assume that the available information domain is much larger than the consumer is able to process during a reasonable period of time.

A content consumer has his beliefs, desires and intentions (BDI). This is nothing more than his *interpretation function* of incoming information. This function might be heuristic, logical or of some other nature. The fact is that the information interpretation function (consumer's BDI) is known only by the information consumer and is not known by the source of information (information system). On the other hand the information consumer does not have knowledge about the nature and structure of the information in the domain. Part of this information might or might not have semantic relation to his inner information. Only analyzing part-by-part information from the domain the consumer is able to decide how much more information he would need, what this information might contain, and how he could use the available information.

Let us use the term *system* to refer to the information domain and information sources available for a consumer. Let's agree that this system contains a capability (represented by one or more sources) to get and process somehow a consumer's requests and provide him with requested information or information about its unavailability.

We thus have

BDI	– information interpretation function known by a consumer, but not known by the system
R	– information relevance function known only afterwards (when the consumer received some information from the system and decided about its relevance with respect to his BDI)
CNT	– content – a set of information, known by the consumer
C	– context – all information in the information domain except content having semantic relation with any distance to the content (this means that if content is not presented in the information domain, there is no context in it).
I	– a set of theoretically available information to the consumer, union of his content and domain

The consumer is able to generate relevance function for the information that he has. In accordance with the generated relevance function, he arranges the incoming context information to predictive, secondary and irrelevant. Relevant information is used for decision making, whereas potentially relevant information requires further info to be utilized or considered irrelevant.

The following issues still need clarifying as regards supporting interactive reasoning process, presented in this work (see Figure 20 on the next page):

context relevance function

since the context relevance function cannot be known, it should be appropriately modeled and substituted by the reactive relevance function, a function which is able to define relevance of the context information coming to the consumer from the system

context request

after analyzing incoming data the consumer makes request to the system for further information – content and parameters of this request should be defined

relevance potential function

after getting a consumer's request the system selects context information which is potentially relevant to the consumer and corresponds to his request – the way of potentially relevant context selection should be presented

The basic unit of information used in this work is *information set* – a set of triples. Content, context and information domain are represented by information sets. The most general graph, representing the interactive reasoning principle would resemble Figure 19 on the next page.

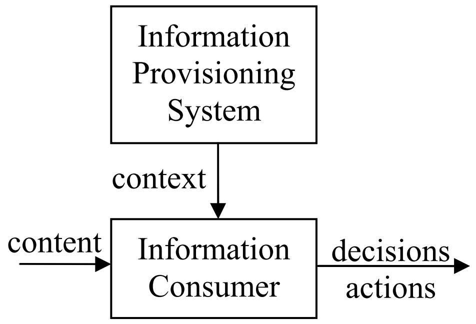


FIGURE 19 Functional system with an additional information source

The idea is the following. Content is basic information, coming to the consumer. He MUST be able to make decisions and perform actions based on it even though its quality might not be very high (it might be inconsistent, incomplete, etc.). However, there is, if needed, a potential possibility of having more information that might have the effect of making his decisions more accurate and actions more appropriate. This additional information is context of content.

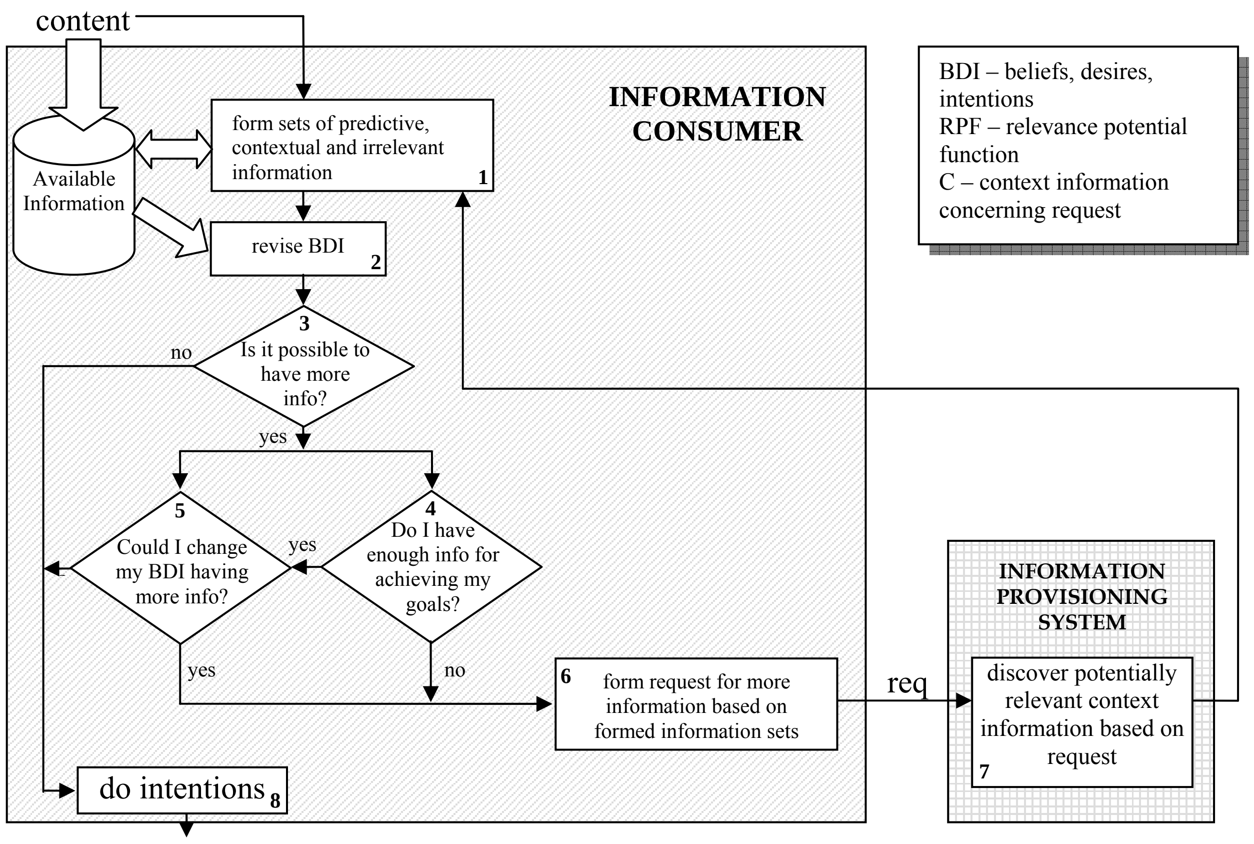

content
INFORMATION CONSUMER
Available Information
form sets of predictive, contextual and irrelevant information 1
revise BDI 2
3 Is it possible to have more info?
no
yes
5 Could I change my BDI having more info?
yes
4 Do I have enough info for achieving my goals?
yes
no
6 form request for more information based on formed information sets
do intentions 8
req
INFORMATION PROVISIONING SYSTEM
discover potentially relevant context information based on request 7
BDI – beliefs, desires, intentions
RPF – relevance potential function
C – context information concerning request

FIGURE 20 Interactive context-based reasoning algorithm

The algorithm presented in Figure 20 from previous page reflects the principle of interactive communication between information consumer (later, consumer) and information provisioning system (IPR). The figure shows on an abstract level the main actions taking place in interactive reasoning.

Input	The algorithm starts when a consumer receives a piece of information (content) from some information source (this can be IRP or some other source of information) through its usual information channel. Initially, all received information is considered predictive (primary)
Blocks 1-3	Based on the received information the consumer modifies its BDI (if needed) and observes whether there might be a possibility of getting more information (existence of an information provisioning system, and availability of a communication channel with it)
Blocks 4-5	Based on the information that the consumer already has, he decides, whether he really wants to have additional information
Block 6	The consumer forms an information set (content) for IPS, about which he wants to have contextual (additional) information; together with content he provides system with a set of parameters, which describe how much and what kind of information the consumer would like to receive
Block 7	Based on the consumer's request IPR selects potentially relevant information for him. The response comes to the block 1 for further analysis
Output	The interactive process ends with the consumer performing intentional actions in the following cases: when the consumer decides that he does not need new information anymore; when the consumer decides that new information would not affect his actions; when the system does not have any new information satisfying the consumer's requests; when the system is not available anymore

4.8 Storing of information

The concepts presented in this work are based on the assumption that semantic and instance information is efficiently stored – processing of text files with ontology description would be definitely too slow, especially in case of dynamically changing information. The research work in this direction is already ongoing.

In [79] authors present an ongoing project, in which they try to use relational databases for storing ontological information. The author of this dissertation completely agrees with their statement: "we need architectures where ontologies can be resolved locally, where relevant parts of huge ontologies can be downloaded incrementally, and where queries can be processed partly on mobile devices, and partly on backend servers, depending on available resources, capabilities and workload" [79].

We believe that the future of global semantic interoperability and ambient awareness is in efficient storage, management and distributed context-aware processing of knowledge, represented by semantic networks. The current Semantic Web languages allow suitable representations (modeling) of knowledge, however, its efficient acquisition, storage, provision and exchange is still an open question. Maybe, relational distributed databases could be adapted for this purpose, maybe the scientific world will find revolutionary new ways of storing and processing of distributed knowledge. The presented work is intended to be one step forward towards the described direction.

4.9 Conclusions

In this section we presented a new understanding of context, which is as a red line going through the whole presenting work. We can conclude the above consideration as follows:

- context is not a certain set of parameters, which can be defined in advance,
- context is not isolated from other information, but can be only dynamically selected from the information domain,
- context is a relational property defining set of information in the domain semantically connected with the information that context consumer already has,
- context is not necessarily relevant to context consumer, the relevance of context is defined by consumer's needs (BDI) in each particular situation.

The benefits of the presented understanding of "context" are:

- it is general enough to be applied for any application areas,
- it allows efficient separation of information collection, storing, context extraction and interpretation functionality between participating parties,
- it allows context-aware application analyzing information domains without having predefined knowledge of the structure of information.

The difficulties of the presented concepts for the application:

- context relevance can not be defined at one moment neither by context consumer nor by information system if the information domain is large,
- applications performing context-based reasoning require possibility of dynamic interaction with the information system,
- semantic interpretation of incoming information is to be performed by context consumer for context analysis.

5 CONTEXT INFORMATION MODELING

This section presents a semantic network as the underlying information structure of the thesis. In 5.1 we discuss basic definitions and semantic meaning of ontologies. RDF data model and its use is presented in 5.2. In Sub-section 5.3 we discuss the structure of RDF network. We present our concepts of context consideration depth and dimensional context consideration depth in Sub-sections 5.5 and 5.6 correspondingly. Discussion about context modeling based on ontology languages is presented within Sub-section 5.7. We conclude the section in 5.8.

5.1 Ontology – basic definitions, semantic meaning

It is not enough that entities are semantically annotated, also the software analyzing the annotated entities need to be able to understand that annotation. For example, a WWW page about the company *xyz* could provide the following formal annotation about itself:

resource:	*www-page*
owner:	*xyz*
content:	*about*
last update:	*20-mar-2006*

A software agent analyzing such formal definition would recognize that the page has 4 properties, the names of those properties and their values. However, if the agent had no knowledge about the meaning of the words *resource, owner, content, last update, WWW page, about* and *mar* it would not be able to analyze the annotation autonomously.

The agent needs explicit knowledge about the concepts used in the description, in other words, there should be some database which would include those concepts and their mutual relationships, and which would provide their semantic meanings to the agent. For this reason, ontologies were introduced to the Semantic Web community.

An ontology is a document or file that normally defines the relationships between terms in any particular domain of discourse. The best known definition of ontology is "a specification of conceptualization". In practical terms, ontologies can be defined as finite yet extensible controlled vocabularies, as observing unambiguous interpretation of classes and term relationships, and as having strict hierarchical subclass relationships between classes. In addition, ontologies typically allow property specification and value restriction on a per class basis, as well as inclusion of individuals (class instances) in the ontology [50].

The main purpose of ontologies, as it is currently seen by Semantic Web the research community, is to bring semantics and intelligent automated information processing to the so-called Converged Web – a global network, which would include a variety of access networks, devices, services, resources, processes, etc. integrated, interoperated and widely accessed.

5.2 RDF data model and its use

Let us use a classical triple data model, which forms the base for Resource Description Framework (RDF), a framework for representing information on the Semantic Web [66].

The motivation behind choosing this particular data model partially coincides with the motivation for RDF – it is simple and easy for applications to process and manipulate. The second argument is that there is a huge amount of work done, based on such information representation model, which can be reused in context processing. A great number of ontologies have been built using RDF and RDF-based languages, enabling the use of semantic analysis in context processing. However, one of the most important reasons for our interest in the RDF model is that the subject of attention in our context interpretation can be easily slotted into it.

Resource Description Network is nothing else than a modern standardized way of representing classical semantic networks. The RDF data model is based on triples, which consist of a subject, an object and a predicate connecting them.

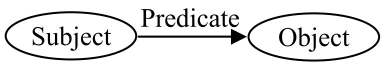


FIGURE 21 RDF graph data model [66]

Each triple has three parts: subject, predicate (also referred to as property) and object. A simple description of its semantics would be the following: an object is related to a subject by a relationship defined by a predicate. The literal enumeration *(Subject, Predicate, Object)* corresponds to the graphical representation given in Figure 21. A set of such triples is called an RDF graph.

According to the RDF specification, each node and link of the graph can be represented by a literal or via Universal Resource Identifier (URI) which is similar to URL (Universal Resource Locator) and acts as a link to the resource in the web, which further describes the entity. Blank nodes are also allowed. Ontologies described in the previous section use the same data model, but with more restrictions and an additional specification representing conceptual space of the real objects.

The main goal of the research community working on RDF and OWL is to enhance the current web by adding structured and organized semantic data, which would be available world wide. RDF (and RDF-based) descriptions located in the web refer to different ontologies available in the web and can give more semantic information about them. These ontologies, in turn, might refer to other ontologies containing more conceptual information or other domains and so on. For the web it means the possibility of automated processing and global interoperability – unknown terms in semantic descriptions could be clarified by more generic, known concepts.

However, the use of RDF and ontologies seems to us to involve much more than updating the current web with a knowledge base of concepts. Semantic networks represent conceptual spaces, somewhat mirroring the way humans perceive and analyze information.

The main difference between human perception of information and the majority of applications analyzing information is that a human is able to receive previously unknown information, analyze it and understand it by building the missing semantic links with the information he already has; applications usually need to know beforehand what is the information received in order to be able to use it. Of course, there are many techniques developed introducing learning capability to applications, but they are usually quite complicated to implement, need powerful applications, have large database inside (or available) and need some special, usually manual, teaching processes being executed before they are able to function. Thus, due to the ensuing complexity, their use is not widespread.

On the other hand, a human (possessing the necessary prior knowledge), on receiving previously unknown information, will try and get some additional information, which would support his understanding, from the environment (by asking, reading or watching, etc.). This is a way that is very similar to the principle of interactive reasoning presented later in this work. The author believes that semantic network as a data model and, particularly, RDF-based languages due to their popularity in applications of various kinds, are very suitable for representing information from any domain and between domains for context-aware processing. The initial goal of the Semantic Web efforts is to introduce semantics to simple data or objects in the web. The aim of this work is to use this semantic information for context-aware applications.

5.3 Some definitions

Returning to our considerations about context let us recall that any information which can be used to characterize the situation of a subject of attention can be treated as a context. Therefore, let us define the basic unit of context information as a *context fact*:

(s, Predicate, o)

which represents information o related to the subject s via the relationship described by Predicate.

The above proposition means that the subject and the object in a triple model have their concrete values. The semantic constituent here is the relationship between the subject and the object described by the predicate. The triple above can always be thought of as context information with respect to the subject *s* since *Predicate* and *o* describe the situation of the subject *s*.

For context utilization as a context fact, one quite often needs a variable object or a variable subject, which is undefined in advance. In such cases conceptual definitions of context instances play the role of variables. Thus, the triple may have a variable object and/or variable subject. A conceptual description of context does not actually give us context information as such, but it has context potential – when the variables take their concrete values, the description becomes a context fact. Therefore, for the convenience of reference, let us use the term *context triple* for any triple *(Subject, Predicate, Object)* having a static or variable subject and object. In principle, *Predicate* may also be a variable, but such cases we postpone for our future work and these are thus taken out of the scope of this thesis.

Let us introduce two more terms.

Context set as a set of context triples. A context set may contain triples with variables or defined values. *Context situation* is a context set represented by context facts (context triples with concrete values).

The idea of our further modeling is to find techniques of this kind for composing context sets which would be useful in context interpretation and reasoning.

5.4 RDF network structure

Let us consider a huge semantic network as a model of the world. That network may contain any kind of knowledge we care to invent: concrete objects, resources, or entities; relationships between objects (predicates, properties); classifications and conceptual definitions of objects, relationships, etc. – in principle, anything at all. The basic RDF definition allows such description of knowledge domain (in general case, world) via a set of triples *(subject, predicate/property, object)*.

So, any kind of knowledge can be represented by RDF semantic network, which is based on triple data model and four kinds of participants:

- resource
- property
- literal
- statement.

Therefore, by definition, an RDF-described semantic network is very heterogeneous, chaotic and complex. It may contain real objects, their classification, their class hierarchy, etc (see Figure 22).

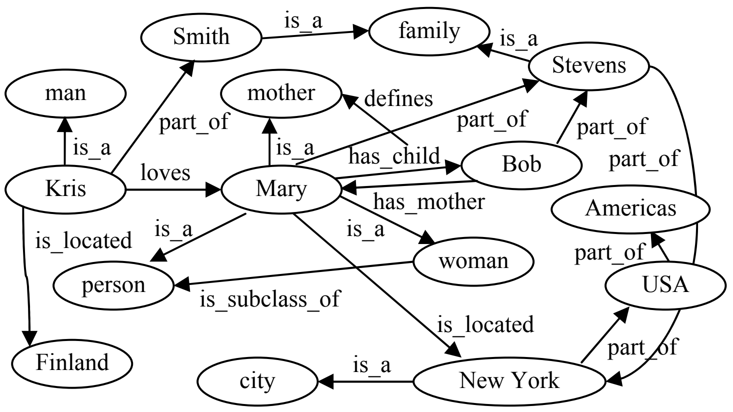


FIGURE 22 Semantic network

RDF itself holds no special role for ontology. Ontology as a concept was introduced with the purpose of clear (or almost clear) separation between classes (conceptual definitions) and their natural realizations – instances. Basically, when we talk about "Mary" and "Chris", we mean real people, who live somewhere, were born at a certain moment of time in a concrete place even though we do not specify where and when.

Introduction of ontological languages, such as OWL made the world of semantic (RDF) networks much easier via the introduction of a set of restrictions to that part of semantic network (SN), which has a lesser degree of dynamicity and higher degree of conceptuality and, therefore, is more universal in the sense of terminology (vocabulary).

So, now, the graph from the Figure 22 would look as shown in Figure 23 on the next page.

Now, let us consider the following example. When we say "Chris is a man", we define "Chris" as an object, which belongs to the class "man". It automatically means that since "Chris is_a man", it derives a set of properties, which define the class "man" and even some values for some properties, specific for this class, e.g. "chest is small".

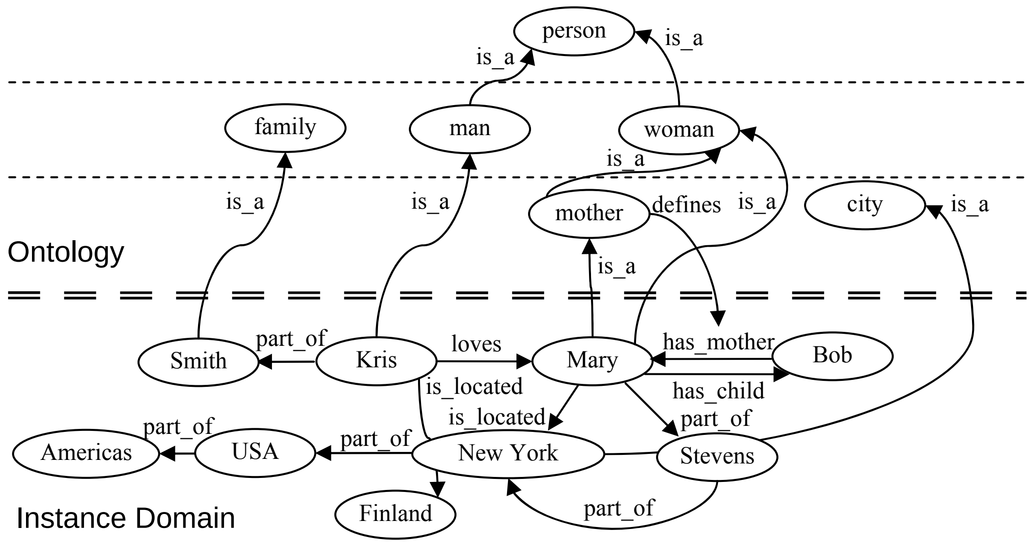


FIGURE 23 Semantic network with separated ontology

In principle, if we say "Man is_a person" it follows that "Chris is_a person", because "man" is a concrete class, which is an instance of the parent class "person". It is a conceptual entity for us, but with respect to the class "person", it is its concrete instance. Using terminology from the graph theory (trees): in both cases the child node derives some properties from the parent node, but also has his own ones that make it unique with respect to its siblings. The conclusion arising from the above consideration is that the relationship "is_a" defines "*degree of conceptuality*" in semantic network.

In the semantic network organized in a way represented in Example 2 it is much easier to operate: we organize information by conceptuality criteria – the more static and conceptual part is described by a more restricted and formal ontological language; the more dynamic instants part (real world objects) is described according to the real situation and real existing objects/entities with links to corresponding concepts.

In Figure 23 we divided SN with respect to the relationship "is_a", which indicates belonging to some class. In principle, "is_a" is the same kind of predicate as "loves" or "has_child". However, it is different in the sense that it clearly divides SN to two or more parts according to the level of conceptuality.

We can also find other kinds of "special predicates", for example, "part_of". Such relationship clearly separates semantic network on different logical domains. The fact that some object or a set of objects belongs to a certain domain may yield quite a lot of information – usually, instances from the same domain have many similar properties; if the domain is a part of another domain, then all entities inside it are also parts of the other domain enclosing it.

The third "special" predicate would be "is_located", which divides the information in semantic network in groups according to the physical location of corresponding entities.

Figure 24 shows the division of our example semantic network on different domains. Whereas the property "is_a" divides semantic network into information spaces with different levels of conceptuality, the property "part_of" divides it into groups of entities belonging to different logical domains, and the property "is_located" divides the network into groups of entities belonging to the same physical domain.

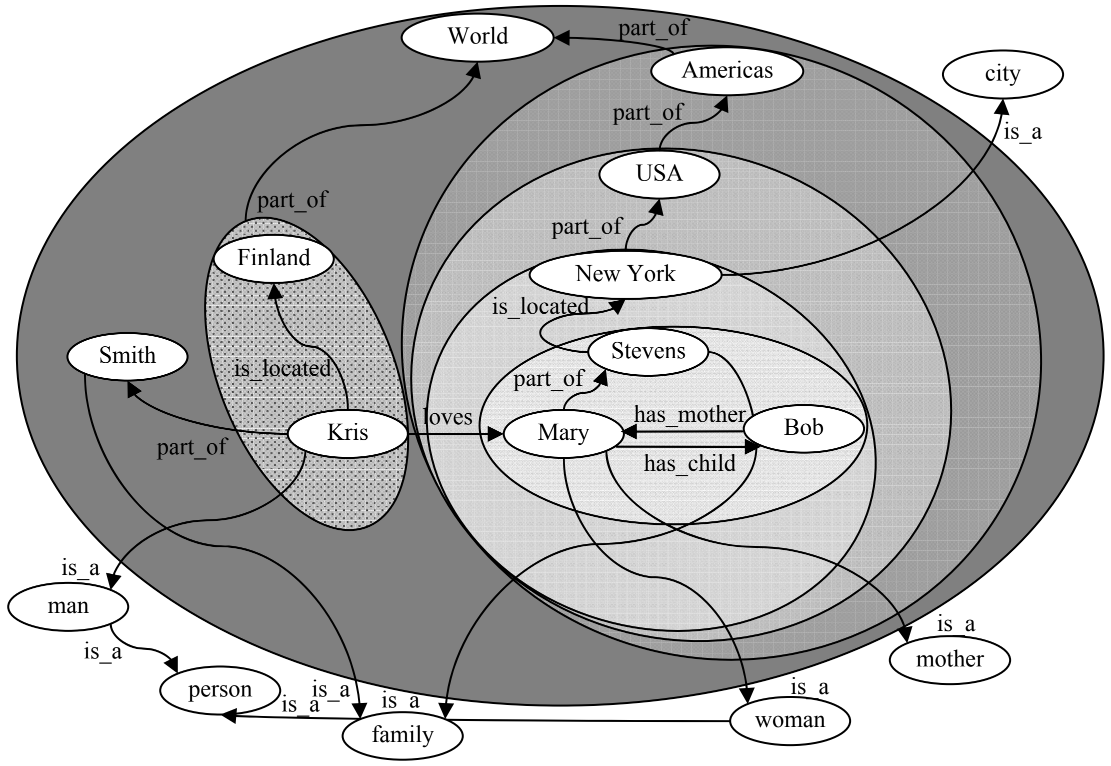


FIGURE 24 Domain separation within semantic network

What is common between these three properties? They all partition the information in the semantic network with respect to some common properties. Such partitioning can make analysis of semantic network much easier and faster. There are also other properties of such kind, however we leave them out of the scope of this dissertation for future research. It may be that such partitioning predicates possess the transitivity property: if (A,P,B) and (B,P,C) then (A,P,C), but this needs to be proved.

In this thesis we aim to understand such properties as dimensions of semantic network – one dimension corresponds to one special property, which we will call *dimensional property*. The use of such approach is described in Section 6.

Many predicates have synonyms (OWL: equivalentProperty) and antonyms (OWL:inverseOf). This is also true for dimensional predicates. For example, "is_a" for classes (not for real objects) has a synonym "subclass_of" and an antonym "superclass_of". "Part_of" has an antonym "contains" etc. Of course, these should not be considered as different properties, but some technique in the information provisioning system should be used for their correspondent identification.

5.5 Context consideration depth

According to our point of view any *entity in the world has potential to have a contextual relationship with a particular subject if there can be found an unbroken sequence of semantic relationships between the entity and the subject of attention through intermediate entities or directly*. The distance between the subject of attention and the entity under consideration defines the strength of this potential.

Consider a future situation in which there might be a huge amount of context information available at the computer network to any context-aware entity. Without an effective approach to limit the analysis of context in such a situation, it would become a bottleneck for the whole system. This situation is especially problematic for applications which do not know in advance the quantity and content of context information they might need. Our subject-oriented view onto context allows working with context information of different depths with respect to the subject.

Let us consider the following example domain (Figure 25).

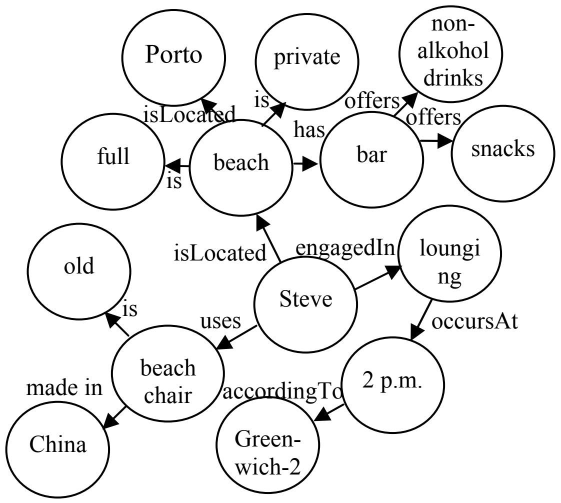


FIGURE 25 Example domain

The participants of our example domain are interconnected by directed semantic links. Note that the example network is, basically, a tree, with the root "Steve".

Context set of "Steve" is {

("Steve", engagedIn, "lounging")
("lounging", occursAt, "2 p.m.")
("2 p.m.", accordingTo, "Greenwich-2")
("Steve", isLocated, "beach")
("beach", is, "private")
("beach", is, "full")
("beach", isLocated, "Porto")
("Steve", uses, "beach chair")
("beach chair", is, "old")
("beach chair", madeIn, "China")
("beach", has, "bar")
("bar", offers, "non-alcoholic drinks")
("bar", offers, "snacks")}

Let us consider "Steve" to be a subject of attention and analyze the context with respect to it: first we consider the entities and their corresponding links directly connected to the subject, then we look at the entities with two hops distance and so on. Let us refer to the distance between the subject and the context as *context consideration depth (CCD)* and to the direct context of the subject as *first order context* (Figure 26a on the next page), to the 2-hop context as *second order context* (Figure 26b on the next page), and to the n-hop context as *n-order context.*

The procedure of potentially relevant context selection disregarding its target CCD will be referred to in this work by *contextualization*. This suits us very well since it means that having the subject of attention we are collecting its context represented by context triples.

In many cases one needs to select the subject knowing the object which has a contextual relationship with the subject. We'll call this procedure decontextualization. Usually this term is understood as taking out the basic fact from its context for further storing and processing without contextual information. However, the term also suits well for the purposes of this work, where *decontextualization* is considered as finding the subject, the context of which is described by a known object and their relationship. In other words, having contextual information we find the entity, which is described by that context.

Let's agree that by default the semantic network is considered planar – no dimensions, no domains, nor any kinds of properties (predicates). For such cases the depth of context consideration depth could be defined more formally as follows:

First order context of the subject S is a set with no replications of the existing triples in the domain in which S plays a role of the subject (semantic relationship starts from S).

N-order context of the subject S is a first order context of all nodes connected with S by (n-1)-hops path starting at S in a domain graph.

Remark 1:

If the n-order context set contains the same triples as the (n-1)-order context set they must be excluded from the n-order context set and not counted as an n-order context of the subject.

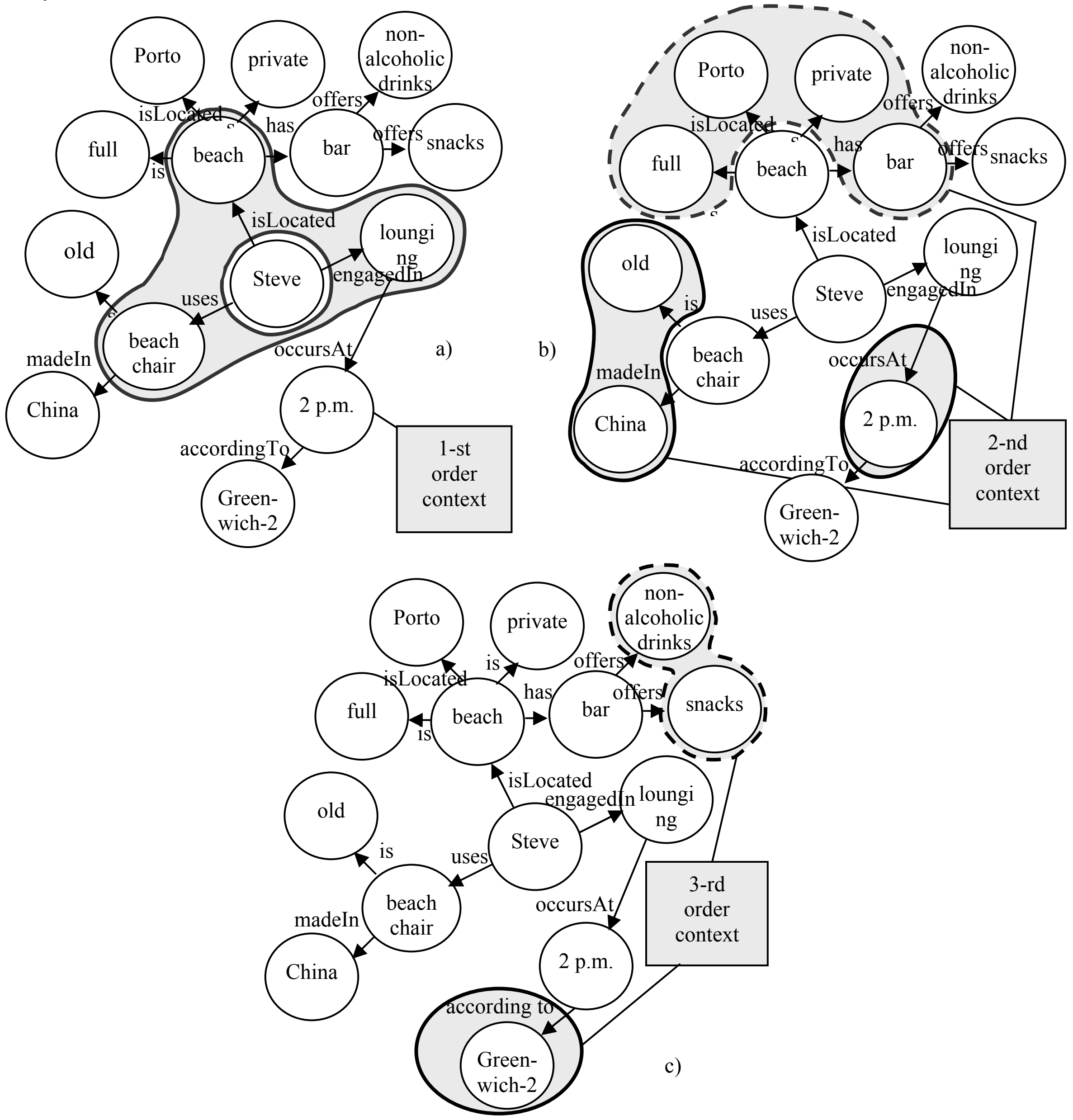


FIGURE 26 Context depth of the subject "Steve"

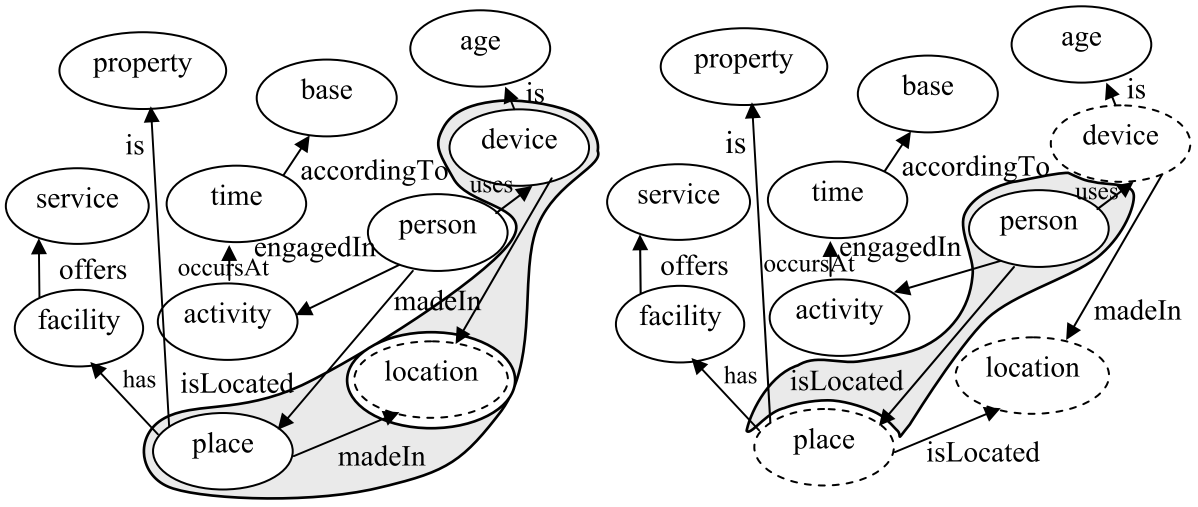


FIGURE 27 First and second order decontext of the object "location"

Remark 1 above is introduced for the case of loops in a domain model. If some node has semantic relationship with itself it shouldn't be counted many times. The graphic representation of Axiom 1 is shown in Figure 28 below.

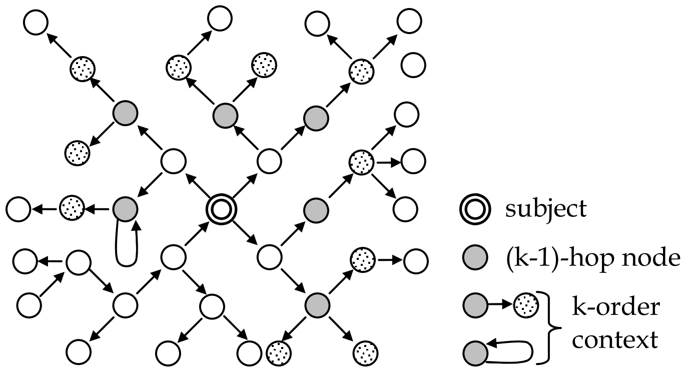


FIGURE 28 Graphical representation of the remark 1

5.6 Dimensional context consideration depth

It is important to note that in our first example *s* and *o* are instances, and not concepts, so, basically, our example semantic network is planar with respect to its conceptual dimensionality – the dimensional property "is_a" - and does not contain any conceptual definitions. Another dimension in our example is location – the dimensional property "is_located" (see Figure 29 on the next page).

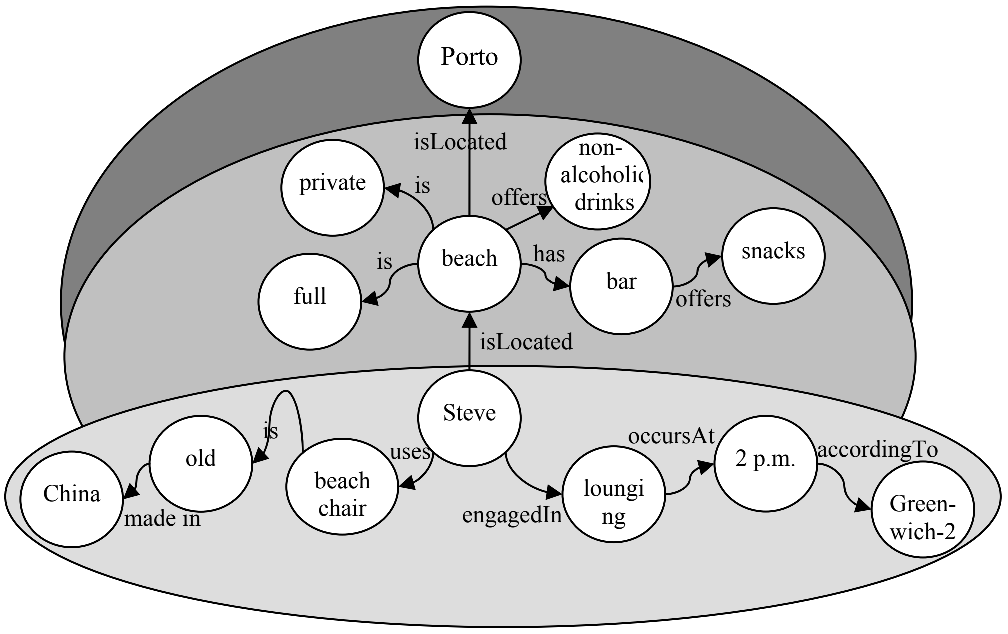


FIGURE 29 Semantic network structure along the dimension "is_located"

The presented dimensionality concept can be easily understood for the dimensional property "is_a". How to treat other dimensions? Very simple if you imagine that the beach-chair remains permanently on that beach and has some in-built electronic equipment for supporting context-awareness. It needs information about people located at the beach, about their actions, states, etc.; it might need information related to the beach in general, for example, about the working hours in that locality, about how many people are there, about the kind of weather there, etc,; it would be less likely that it would need information about Porto as a city, about its other objects and even less likely that it would need information about Portugal.

So, an object within a certain domain usually needs information about the same domain, and very rarely about the domains above it. Exactly the same applies to conceptual dimension – an intelligent agent operating within some particular vocabulary doesn't need any conceptual definitions – it knows how to interpret instant information.

However, once an object lands on an unknown domain, it behaves differently: omitting particularities of the host domain, it initially tries to get more and more general information in order to obtain sufficient information for understanding the information within the host domain.

Due to the importance of the issue in context analysis, our context modeling technique should provide an easy way for operating along and within dimensions. For this purpose the author suggests using two different values for context consideration depth – one, within one plane of the dimension, and another one "jumping" hops between different planes of that same dimension – *dimensional context consideration depth (DCCD).*

Let's select a dimension in some semantic network (Figure 30) and one subject of attention as well. The context consideration depth value will tell us how many hops will be analyzed within each dimensional plane along the chosen dimension starting from the subject of attention. Dimensional CCD will tell us how many planes along the dimension will be analyzed.

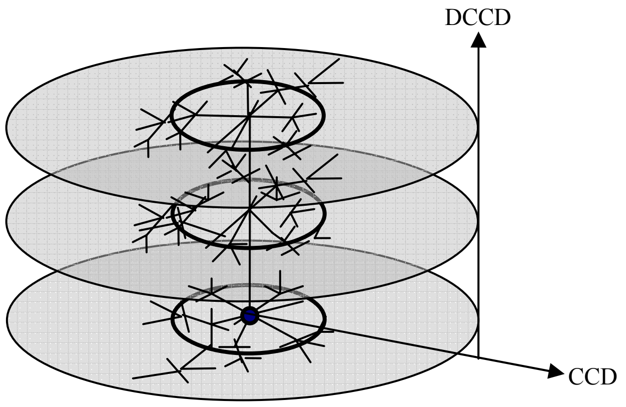


FIGURE 30 Schematic representation of semantic network structuring

The distance in n-hops between the subject of attention and some object is called n-order context. We can define the distance between the subject of attention and the corresponding upper-level (plane) object, connected with the subject of attention by k-hops of the dimensional property, as the k-level context. Therefore, within a particular dimension:

The dimensional plane of some subject S along a dimension defined by P (dimensional predicate) is formed by all predicates in the domain connected with S by some semantic paths, which do not contain P.

First order context of the subject S is a set with no replications of the existing triples in which S plays the role of subject (semantic relationship starts from S) within the same dimensional plane with the subject.

First-level context of the subject S along a dimension defined by predicate P is the dimensional plane of the subject O, which participates as an object in the predicate (S,P,O) of the information domain.

K-level context of the subject S along dimension P is the dimensional plane of the subject O, which has a k-distant semantic path with S starting from S, where all predicates are equal to P.

N-order context of the subject S is a first order context of all nodes connected with S by an (n-1)-hops path starting at S in a domain graph within the same dimensional plane with the subject.

K-level, N-order context of the subject S along a dimension defined by P is an N-order context of the object O, which is connected with S by a K-hops semantic path with predicates equal to P.

5.7 Context modeling based on ontology languages

The RDF-based information representation could be used in any application area which requires some information exchange between all kinds of independent computer environment objects – hardware, low-level software, middleware, applications, agents, etc. Using some common information model would allow interoperability between very different kinds of software, the only condition would be that they have an information exchange channel.

Of course, each particular domain would have its own domain vocabulary, maybe similar, maybe different from others. However, a common data representation model and availability of conceptual meta-information would enable applications to analyze information from different domains with different attribute names and under different circumstances.

There are many information sources in the computing/networking environment – sensors, protocols, triggers, etc. It would be quite easy and useful to automatically maintain a semantic network information space by gathering that information, following changes occurring in the environment and storing it in an ontology-based domain database, which can be distributed even within the domain if needed. A general idea of such environment can be found in this work in Section 6 (context-aware global mobile environment). Of course, problems of security and trust can not be avoided in building such information space. These, however, are also left out of the scope of this work.

Semantic Web languages are being actively developed. The usual problem with ontologies and the world of mobile devices is that ontologies are quite static and heavy to be of use in highly dynamic and resource-poor mobile environments. Small-sized ontologies are not very useful for information sharing due to their scarce distribution around the Semantic Web, and bigger ontologies require a lot of memory and computation power for analysis. Quite often ontologies built for specific domains are not interoperable, and universal ontologies do not work well in specific environments.

The biggest problem from the author's point of view is that there are no RDF-based database management (information provisioning) systems. The information currently is stored in text files, which are available through the Internet as web-resources. Such storing makes access and processing of such information quite slow despite the fact that modern processing resources are quite powerful. There are some research efforts in the direction of efficient store and representation of semantically enriched data [22], [44]. We believe that ontology-oriented distributed databases with an efficient system of intelligent data search are very much needed by current technologies in order to make converging networks semantically enriched, context-aware and globally interoperable.

One more problem existing with ontological representation of information spaces is that they are quite static – often instance data is stored together with an ontology, which makes dynamic use of it quite problematic. Context-awareness by its definition means that information about the environment is dynamically changing, and context-aware application is an application that is able to appropriately react on such changes. This means that even though ontologies as an information model are very useful for context-awareness capability provision the way they are currently stored should be revised since it cannot be adapted for dynamic data. Embedding parts of ontological representation into the environment and separating data from its conceptual description together with infrastructural solutions would probably solve the problem. The idea of context providers, providing a small part of the semantic description of provided data and linking that to global descriptions, has many advantages, among them:

- Domain information may be changed locally and dynamically without rebuilding the ontology.
- In many cases of reasoning on context there is no necessity to analyze the whole ontology as only a small amount of information is needed and can be provided directly by CP.
- Parts of the domain semantic description may be easily substituted or enriched by updated information.
- Introduction of new context providers is quick and easy – they just have to register themselves in the domain information base.
- Analysis of the domain semantics does not necessarily need a mediation service – any kind of software is able to interact with CPs directly and draw their own inferences.
- Processing of semantic information may be spread, with respect to its complexity, between devices of different capabilities.

In this work we neither specify which language can be used for semantic network representation nor do we provide examples using any specific RDF-based language. Classical RDF is very simple and expressive. Together with OWL it has different kinds of extensions, produced by research groups working, for example, with representation of context. As it has been mentioned already, [47] extends RDF with triple-to-container relationship, which is used for contextual descriptions. In [3] authors noted that OWL is not expressive enough for representation of role-like relationships and overcame this limitation by introduction of OWL-DL ontology for modeling activities of mobile users. Context ontology language (CoOL), which is introduced in [80], relies on aspect-scale-concept model and, basically, links the entity to its contextual information through special links, a set of restrictions and additional parameters.

Context consideration depth considered in this section does not define the relevance of context. According to the dynamic nature of context we cannot define a context set in advance without information about the task for which it might be needed. However, it defines the potential of information to be contextually relevant to the subject of attention in a particular situation.

It is very likely that for some applications it will be necessary to analyze a great amounts of information having some level of depth with respect to the subject. This might be needed, for example, during semantic comparison of two entities from different domains. While a comparison of direct context information might not give any result, a comparison of the second or third order context could give a much better understanding of the semantic distance between the two entities.

Common knowledge structure for context-awareness would allow global interoperability for context-aware applications. The main long-time purpose of context-awareness capability development is to enable applications to properly behave in unknown environments and under unknown circumstances. Local, domain or task-oriented context-aware applications fail in that. Using RDF-based information representation allows using not only context data, but also semantic information around it, which can be seen as progress towards the desired direction.

The aim of this work is not to introduce any new extension or ontology to existing semantic network languages, but to try to find a way that would allow efficient usage of all achievements in the area.

5.8 Conclusions

Within this section we introduced a new concept of context consideration depth, found the way of structuring the RDF semantic network, introduced the concept of "dimension" in semantic network and dimensional context consideration depth. A comprehensive discussion about context modeling based on ontology languages is given in sub-section 5.7.

The next section extends the above ideas and provides necessary mathematical foundations for intelligent context selection.

6 INFORMATION PROVISIONING SYSTEM: INTELLIGENT CONTEXT SELECTION

This section concentrates on intelligent context information provisioning to the context consumer. We present our main mathematical contribution within this section. In 6.1 we describe where some mathematical techniques can be used within interactive reasoning process. We discuss context relevance function definition possibility within Sub-section 6.2. We introduce our relevance potential function within 6.3 in its different modes – simple RPF, predicative RPF, and decontext RDF - and provide a validation of it. Sub-section 6.4 presents a dimensional variant of relevance potential function and its validation. Cumulative relevance potential function is introduced within Sub-section 6.5. We present a related calculation example in Sub-section 6.6. Possible extensions to RPF are dealt with in Sub-section 6.7. We comprehensively discuss possible applications of the presented mathematics in Sub-section 6.8 and conclude the section in 6.9.

6.1 Process description

Based on the request by a consumer the system should select contextual information, which could be potentially relevant to the given subject of attention with a certain degree of relevance potential. Relevance of the context can not be known by the system since it doesn't know consumer's beliefs, desires and intentions. Therefore, some prediction technique for context relevance needs to be developed.

The system is based on interactive reasoning principle, presented in Section 4: a consumer specifies a subject of his attention, and the system responds with some context information, which could be relevant to the consumer from the system's point of view. The consumer analyses the received information, revises its own BDI, investigates the decision making possibilities and requests the system for more information if needed. This cycle might be repeated many times depending on the resources and intentions of the consumer.

One main goal of relevance prediction is minimization of the amount of information given to the consumer (if all available information were to be given, the consumer wouldn't be able to process it due to lack of time and processing resources); another main goal is to minimize the number of information exchange iterations between the consumer and the system. Nevertheless, the consumer should be provided with a reasonable amount of context information which has potential to be relevant to his needs.

This section considers the process from the point of view of the system. Where to apply the developed techniques within an interactive reasoning mechanism is shown in Figure 31 below.

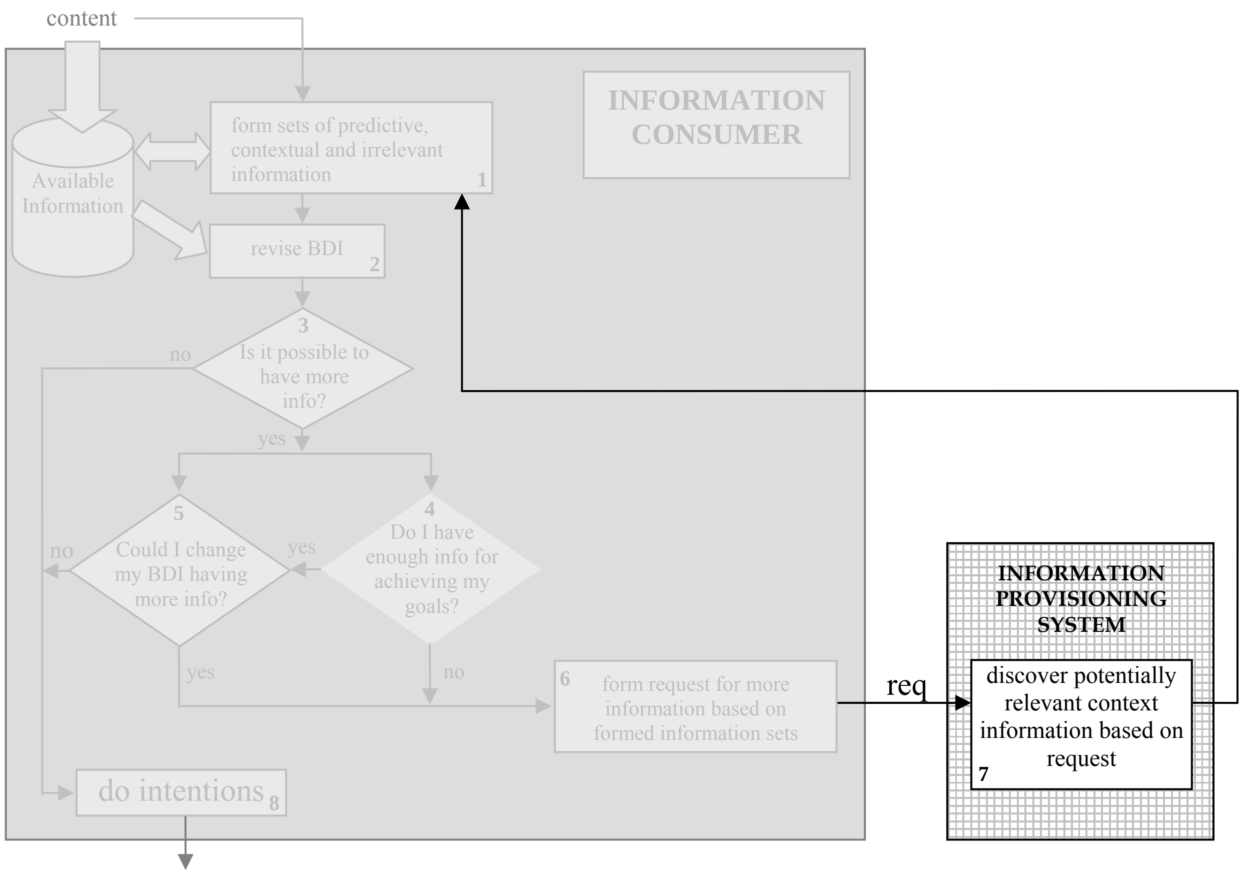

content
INFORMATION CONSUMER
Available Information
form sets of predictive, contextual and irrelevant information 1
revise BDI 2
3 Is it possible to have more info?
no
yes
5 Could I change my BDI having more info?
4 Do I have enough info for achieving my goals?
no
yes
yes
no
6 form request for more information based on formed information sets
do intentions 8
req
INFORMATION PROVISIONING SYSTEM
discover potentially relevant context information based on request 7

FIGURE 31 Information provisioning system within interactive reasoning

Relevance potential function (RPF) is used by the context provisioning system for discovery of context which could be potentially relevant to particular context consumer in its particular situation. The result of a potential context discovery is a context set (set of triples), sent to the consumer as a response for its request[13]. BRPF (Base RPF) is a base function used for context discovery of variable arguments, presented in Table 1.

TABLE 1 Relevance potential function arguments

	Argument	*Description*	*Is necessary?*	*Default value*
f	context search focus	subject of attention, focus of context discovery	yes	-
n	planar context consideration depth	number of hops, at which context is discovered starting from f within one dimensional plane	no	1
D	dimensional property	If defined, dimensional context consideration depth is taken into account	no	-
l	dimensional context consideration depth	Number of planes considered in context discovery within dimension defined by D	no	0

6.2 Context relevance function

If the context relevance function were known by the information provisioning system, it would be very easy to provide a relevant context for the consumer. Such systems have usually quite powerful processing resources and could analyze the whole available information domain, arrange information with respect to the context relevance function and provide the most relevant information to the consumer.

Unfortunately, it is not possible for a system to know the context relevance function for every application. This work assumes that each context consumer (e.g. application, agent) has its own reasoning engine, which might be small and simple, or perhaps very intelligent and powerful. Reasoning engines could be very different, they are embedded into the context consumer, and the reasoning principles most probably could not be passed to any other entity.

[13] In principle, some other entity could also make request for the context consumer. For example, there may be some host agent, into which the guest application enters, in the environment. There is some information about the environment which is supposed to be shared with each guest application and the host agent is making a request for that information to the system on behalf of the incoming context consumer.

Therefore, it is possible only to define a value range for an unknown context relevance function, which is the following:

$$R_{CNT}^{BDI}(I) \in [0,1]$$
$$R_{CNT}^{BDI}(CNT) = 1$$
$$R_{CNT}^{BDI}(C) \in (0,1)$$
$$R_{CNT}^{BDI}(I - CNT - C) = 0,$$

where *R* = context relevance function; *CNT* = content; *BDI* = beliefs, desires and intentions of context consumer; *C* = context; *I* = information domain.

Let's approximate the context relevance function by the *reactive context relevance function (RCRF)*, which is located within the context consumer and defines the relevance of potentially relevant context information, which is received by the consumer from the system. The value range of RCRF is the following:

$$RR_{CNT}^{BDI}(C) \in \{0, 0.5, 1\},$$

where *0* = irrelevant, *1* = relevant, *0.5* = might be relevant (needs more info).

Each triple in a context set is analyzed by the user with the reactive context relevance function and as a result three information sets are formed – relevant context (used in decision making); secondary context (might be relevant, needs more info); and irrelevant context (not used). This work does not provide a detailed description of the reactive relevance function; it is left for the context consumer implementation.

6.3 Base relevance potential function

Triple border operator

For further definitions we need to introduce the following *triple border operator*:

$$L^k(s,p,o) = \begin{cases} o, \text{if } k = 1 \\ s, \text{if } k = -1 \\ (s,p,o), \text{if } k = 0 \end{cases}, \tag{6.1}$$

where $k \in \{-1,0,1\}$, $L(\cdot) = L^1(\cdot)$, *(s, p, o)* = triple.

If the operator is applied on a single object, its result will be the argument:

$$L^k(s) = s, \forall k \in \{-1,0,1\}, \tag{6.2}$$

where *s* = any single object.

Applied to a set of triples *L* returns a union of results obtained from each triple in the set:

$$L^k(T) = \{L^k(t) \mid t \in T\}, \tag{6.3}$$

where t = triple, T = a set of triples.

Therefore, operating on a set of triples, L^k will return:

if k=1: a set of objects from triples forming T;

if k=-1: a set of subjects from triples forming T;

if k=0: T.

The semantic meaning of this operator is that, when positive, it returns the end node of the semantic relationship (object in the triple); when negative, it returns the start node of the semantic relationship (subject in the triple), assuming that an analysis of the semantic relationship is performed in a direction opposite to the direction of the semantic link.

Simple RPF

The focus of context search can be one entity in a semantic network, triple or set of triples. By default, no dimensional property is defined, therefore BRPF considers semantic network to be planar and disregards any possible dimensionality. Therefore, let's introduce the *simple relevance potential function* (SRPF) from the subject of attention:

$$\delta(f) = \{(f, p_i, o_i) \mid i = 1,\dots V_S^-\}, \tag{6.4}$$

where f = context discovery focus, V_S^- = number of semantic links starting from f, p_i = semantic link connecting the subject f to the corresponding object o_i. The presented first-order RPF returns set of triples, in which the given focus plays a role of subject.

Let's agree that 0-order RPF from any subject of attention will be:

$$\delta^0(f) = \{f\}. \tag{6.5}$$

High-order form of RPF would be the following: $\delta^n(f)$, where n is a CCD level, f – subject of attention. $\delta^1(f)$ (or $\delta(f)$) returns a first order context set of the subject S, $\delta^2(S)$ – second order context, $\delta^n(f)$ – n-order context.

Second-order context of the subject f is a first-order context of all objects in a set of triples forming its first-order context.

Let's agree that RPF from a non-ordered[14] set of simple objects is equal to a union of RPF's from each of these objects:

$$\delta(O) = \bigcup_{o \in O} \delta(o). \tag{6.6}$$

Now we are ready to define second-order RPF:

$$\delta^2(f) = \delta(L(\delta(f))). \tag{6.7}$$

Similarly, n-order context of the subject f:

[14] meaning, not triples, which are ordered sets

$$\delta^n(f) = \delta(L(\delta^{n-1}(f))), n > 0\,. \tag{6.8}$$

In RDF semantic networks a basic element is a triple, in which a subject and an object are connected by a predicate (property). Subject, object and predicate can be literals or links to resources. However, there is a number of RDF extensions, which allow to link not only atomic objects, but also triples and sets of triples. The above definition of relevance potential function allows operation on any type of such semantic relationships. Figure 32 represents examples of first-order context of a simple object (entity), a triple and a set of triples (semantic network) when they are linked to atomic objects.

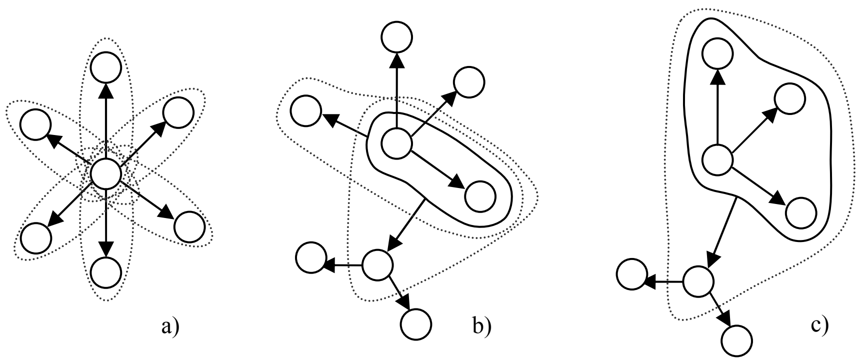


FIGURE 32 Different focus of context discovery - a) object; b) triple; c) semantic network

In a macroscopic view there are also many-to-many relationships, for example, semantic annotation of some domain is linked to a namespace, which defines different terms used in annotation. The defined RPF would still be valid for such kind of relationships. However, in practice, such complex objects (network) are usually grouped under a certain name and all their group relationships are defined through an intermediate node that represents the group as a whole.

Predicative RPF

For dimensional context discovery we also need to have possibility to select only those context triples, which represent certain contextual relationship R between subject and object. Therefore, let us define the *predicative context as a context set with the triples having the given predicate value*. Predicative context is a subset of potentially relevant context set with certain level of CCD. Default BRPF from subject of attention and defined predicate is:

$$\delta(f|p) = \{(f, p_i, o_i) | \, i = 1,\dots,V_S^-, p_i = p\}, \tag{6.9}$$

where f = context search focus, V_S^- = outdegree of f, p = defined predicate connecting the subject f to the corresponding object o_i. This function filters from normal BRPF only those triples, predicate of which is equal to p.

Correspondingly, second and n-order predicative context similarly to (6.7) and (6.8):

$$\delta^2(f|p) = \delta(L(\delta(f|p))|p), \tag{6.10}$$

$$\delta^n(f|p) = \delta(L(\delta^{n-1}(f|p))|p), n > 0. \tag{6.11}$$

In the same way,

$$\delta^0(f|p) = \{f\}. \tag{6.12}$$

Figure 33 represents the result of the function $\delta^2(Steve|\ isLocated)$. The focus of context discovery is "Steve", the limiting predicate is "isLocated", and the context consideration depth is equal to 2.

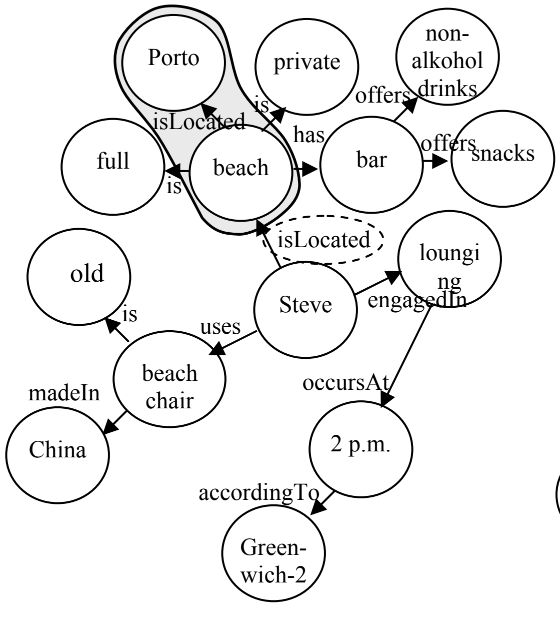


FIGURE 33 Example of predicative context

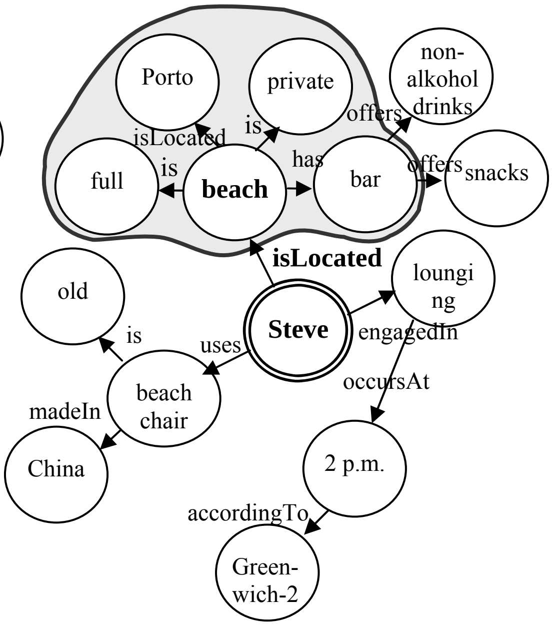


FIGURE 34 The combination of simple and predicate RPF

The Figure 34 graphically illustrates the combination of a simple and predicative RPF and shows the resulting set of the function $\delta(\delta(Steve|\ isLocated))$.

Decontext RPF

The procedure of decontextualization is similar to previously described contextualization, but in the opposite direction. The first order decontext of the focus subject f is a set of triples:

$$\delta^{-1}(f) = \left\{ (s_i, p_i, f) \middle| \, i = 1, \ldots, V_S^+ \right\}, \tag{6.13}$$

where f = decontext discovery focus, V_S^- = indegree of f, p_i – semantic link connecting corresponding subject s_i with the object f.

Let us call the result of (6.13) a *first order decontext*, and respectively, $\delta^{-n}(f)$ = *n-order decontext*.

The second-order decontext of f can be calculated similarly to the second-order context:

$$\delta^{-2}(f) = \delta^{-1}(L^{-1}(\delta^{-1}(f))). \tag{6.14}$$

Finally, n-order decontext of f:

$$\delta^{n}(f) = \delta^{-1}(L^{-1}(\delta^{n+1}(f))), \; n < 0. \tag{6.15}$$

Decontext predicative RPF calculation can be performed as follows:

$$\delta^{-1}(f|p) = \left\{ (s_i, p_i, f) \middle| \, i = 1, \ldots, V_S^+, p_i = p \right\}, \tag{6.16}$$

where f = context search focus, V_S^+ = indegree of f, p = defined predicate connecting the corresponding subject s_i to the object f, being a focus of decontext discovery.

Second and n-order decontext predicative RPF:

$$\delta^{-2}(f|p) = \delta^{-1}(L^{-1}(\delta^{-1}(f|p))|p), \tag{6.17}$$

$$\delta^{n}(f|p) = \delta^{-1}(L^{-1}(\delta^{n+1}(f|p))|p), n < 0. \tag{6.18}$$

For further definitions, let us introduce the following function:

$$sign(n) = \begin{cases} -1, n < 0 \\ 0, n = 0 \\ 1, n > 0 \end{cases}. \tag{6.19}$$

Combining the simple and decontext RPF we come up with the following general formula:

$$\delta^{n}(f) = \delta^{q}(L^{q}(\delta^{n-q}(f))), \text{where } q = sign(n), \tag{6.20}$$

For predicative RPF the combined formula will be similar:

$$\delta^{n}(f|p) = \delta^{q}(L^{q}(\delta^{n-q}(f|p))|p), \text{where } q = sign(n). \tag{6.21}$$

Validation of base relevance potential function

The validity of (6.20) for context and decontext extraction can be verified substituting different values for n.

Let n less than *-1*, then in (6.20) $q = sign(n) = -1$ and

$$\delta^{n}(f) = \delta^{-1}(L^{-1}(\delta^{n-(-1)}(f)) = \delta^{-1}(L^{-1}(\delta^{n+1}(f))),$$

which is equal to (6.18) as it should be.

When *n=-1* the result of (6.20) should be the first-order decontext. Let's substitute n by -1, then

$$\delta^{-1}(f) = \delta^{-1}(L^{-1}(\delta^{-1-(-1)}(f)) = \delta^{-1}(L^{-1}(\delta^{0}(f))),$$

according to the definition (6.5): $\delta^0(f) = f$, therefore

$$\delta^{-1}(L^{-1}(\delta^0(f))) = \delta^{-1}(L^{-1}(f)),$$

according to (6.2): $L^{-1}(f) = \{f\}$, thus, finally

$$\delta^{-1}(L^{-1}(f)) = \delta^{-1}(f),$$

that is the first-order decontext of the subject *f*.

When n=0, according to (6.5) the result should be f .
Let's substitute *n* by zero into (6.20):

$$\delta^0(f) = \delta^0(L^0(\delta^0(f))) = \delta^0(L^0(f)) = \delta^0(f) = f .$$

Substituting positive *n* into (6.20):

$$q = sign(n) = 1,$$

$$\delta^n(f) = \delta(L(\delta^{n-1}(f)))$$

that corresponds to *n*-order context of the subject *f* defined by (6.8).

For *n*=1 the result of (6.20) should be a first-order context.

$$q = sign(1) = 1,$$

$$\delta(f) = \delta(L(\delta^0(f)) = \delta(L(f)) = \delta(f).$$

Formula (6.21) is similar to Formula (6.20), therefore its validation can be done in the same way.

6.4 Dimensional RPF

We suggest to separate context analysis direction (contextualization and decontextualization) along the dimension and within the dimensional plane. Therefore there are four directions of dimensional BRPF context discovery (see Table 2 on the next page).

Let us denote the *dimensional relevance potential function* (DRPF) by *Y* and introduce its general form:

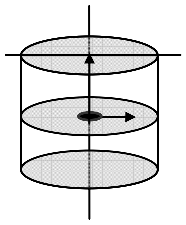
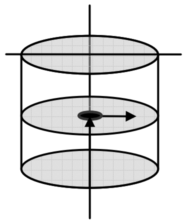
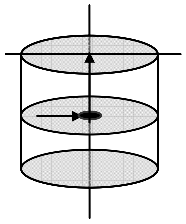
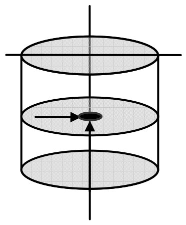
$$\Upsilon^n(f;D^l),$$

where n = horizontal context consideration depth (within each separate dimensional plane), D = dimensional property, l = dimensional context consideration depth.

This function is expected to operate within semantic dimensional space in the following way. Space is divided into spheres, the boundaries of which are defined by the given dimensional property. This is the most general case when the given focus of attention has several direct semantic relationships defined by the given dimensional property. For simplicity of understanding we'll consider the case, where there is only one chain of semantic relationships connecting the subject with other objects in the domain via a given dimensional property and we will refer to each dimensional level (sphere) as a dimensional plane. Operating within a certain dimensional plane is defined by DCCD, the function should return *n*-level context within the dimensional plane, *n* is defined by *planar context consideration depth*

(PCCD). Relationships with the dimensional predicate should be included into the result; however, they should be left out during calculation of each next-order context within the dimensional plane. The reason for such limitation is the following. If they were to be taken completely out of consideration, a part of context would be lost; if they were to use dimensional relationships for the calculation of n-order context within plane, they would shift us to the next level (dimensional plane) of context consideration, thus context discovery would no longer be performed within the given dimensional plane.

TABLE 2 Context discovery directions

PCCD	*DCCD*	*Planar semantic links*	*Dimensional semantic links*	*Schematic representation*
positive	positive	outgoing	outgoing	
positive	negative	outgoing	incoming	
negative	positive	incoming	outgoing	
negative	negative	incoming	incoming	

Let's agree on boundary cases. When PCCD and DCCD are equal to 0, the result of DRPF would be its argument (focus of context discovery). When PCCD is 0, and DCCD given, the result would be a focus of context discovery at the dimensional plane at DCCD distance from the focus (in a more general case, it would be a set of focuses). When DCCD is 0, the operation is performed within the dimensional plane of context discovery focus.

For operating within dimensional planes, we need to exclude dimensional relationships from the planar context discovery. Let's define *predicative exclusion operator*, which excludes from the set of triples, those whose predicate is equal to the given (semantically, its basic filtering).

$$\hat{P}(t|p) = \begin{cases} \{t\}, t = (s, p', o),\ p' \neq p \\ \varnothing, t = (s, p', o),\ p' = p \end{cases}, \tag{6.22}$$

where *t* is a triple or a single object, *p* = given predicate.

Applied to the set of any objects:

$$\hat{P}(T|p) = \bigcup_{t \in T} \hat{P}(t|p), \tag{6.23}$$

where *T* = set of objects (triples or single objects).

So, if $\hat{P}()$ is applied to a single entity, which is not a predicate, it is considered as a set of one element and will always return the argument.

The following formulas define dimensional RPF for different boundary cases:

$$Y^0(f;D^0) = f, \tag{6.24}$$

$$Y^0(f;D) = L(\delta(f|D)), \tag{6.25}$$

$$Y^0(f;D^l) = L(\delta^l(f|D)), l > 0, \tag{6.26}$$

$$Y^0(f;D^l) = L^{-1}(\delta^l(f|D)), l < 0. \tag{6.27}$$

Combining (6.26) and (6.27):

$$Y^0(f;D^l) = L^v(\delta^l(f|D)), v = sign(l). \tag{6.28}$$

Based on the above formulas, let's define DRPF:

$$Y(f;D^0) = \delta(f), \tag{6.29}$$

$$Y^{-1}(f;D^0) = \delta^{-1}(f), \tag{6.30}$$

$$Y^2(f;D^0) = \delta(\hat{P}(Y(f;D^0)|D)), \tag{6.31}$$

$$Y^{-2}(f;D^0) = \delta^{-1}(\hat{P}(Y^{-1}(f;D^0)|D)), \tag{6.32}$$

$$Y^n(f;D^0) = \delta(\hat{P}(Y^{n-1}(f;D^0)|D)), \text{where } n > 0, \tag{6.33}$$

$$Y^n(f;D^0) = \delta^{-1}(\hat{P}(Y^{n+1}(f;D^0)|D)), \text{where } n < 0, \tag{6.34}$$

Combining (6.33) and (6.34):

$$Y^n(f;D^0) = \delta^q(\hat{P}(Y^{n-q}(f;D^0)|D)), \tag{6.35}$$

where $q = sign(n)$.

Finally, using (6.24), (6.28), and (6.35) the dimensional relevance potential function can be defined in the following way:

$$Y^n(f;D^l) = Y^n(Y^0(f;D^l);D^0)\,. \tag{6.36}$$

$$Y^n(f;D^0) = \delta^q(\hat{P}(Y^{n-q}(f;D^0)|D)), \tag{6.37}$$

where $q = sign(n)$.

$$Y^0(f;D^l) = L^v(\delta^l(f|D))\,, \tag{6.38}$$

where $v = sign(l)$.

$$Y^0(f;D^0) = f\,. \tag{6.39}$$

where $l =$ dimensional context consideration depth, $n =$ planar context consideration depth, $l \in Z, n \in Z$, $f =$ focus of context discovery, $D =$ dimensional property.

Validation of dimensional relevance potential function

Let's consider different cases of the Y-function calculation.

For *l > 1, n=0:* zero-context along dimension *D*.
In case of zero CCD, planar RPF returns the argument. Logically, since DCCD is given, DRPF should return the focus of context discovery at the *l* dimensional plane with respect to dimensional property *D* at the positive direction (see Figure 35a on the next page). For this case potentially relevant context can be extracted using (6.38):

$$Y^0(f;D^l) = L(\delta^l(f|D))\,,$$

which will give the objects of *l*-order predicative context of the subject *f* that corresponds to a focus of context discovery on the dimensional plane *l*.
However, the value can be substituted also into (6.36):

$$Y^0(f;D^l) = Y^0(Y^0(f;D^l);D^0)\,,$$

applying (6.38)

$$Y^0(f;D^l) = L(\delta^l(f|D))$$

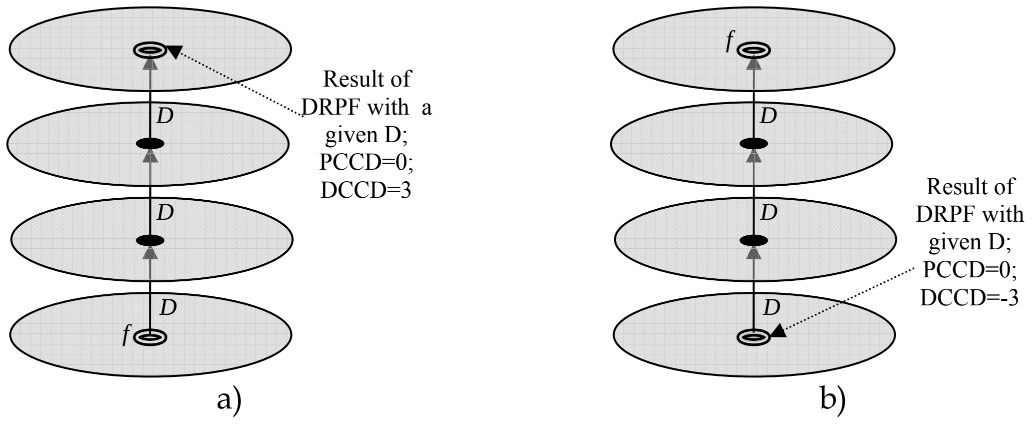
we'll receive a focus of discovery on the plane *l*. According to (6.39):

$$Y^0(Y^0(f;D^l);D^0) = Y^0(L(\delta^l(f|D));D^0) = L(\delta^l(f|D)).$$

For *l < 1, n=0:* zero-decontext along dimension *D*
Similarly to the previous case for *l<0,* using (6.36), (6.38) and (6.39):

$$Y^0(Y^0(f;D^l);D^0) = Y^0(L^{-1}(\delta^l(f|D));D^0) = L^{-1}(\delta^l(f|D))\,,$$

which corresponds to the expected result (see Figure 35b on the next page).



Result of DRPF with a given D; PCCD=0; DCCD=3
D
D
D
f
a)
f
D
D
D
Result of DRPF with given D; PCCD=0; DCCD=-3
b)

FIGURE 35 Expected results of DRPF with PCCD=0

For *l = 0:* in this case the context retrieval should be performed within the same dimensional plane where focus of discovery is located. Substituting values into (6.37), for n>1:

$$Y^{n}(f;D^{0})=\delta(\hat{P}(Y^{n-1}(f;D^{0})|D)),$$

that is a first-order context of *n-1* context within the dimensional plane 0.

For *n<1*, according to (6.37)*:*

$$Y^{n}(f;D^{0})=\delta^{-1}(\hat{P}(Y^{n+1}(f;D^{0})|D)),$$

that is a first-order decontext of *n+1* decontext within the dimensional plane 0.

For *n=1; l=1:* the result should be a first-order context within the dimensional plane 1. Using (6.36), (6.38) and (6.39):

$$Y(f;D)=Y(Y^{0}(f;D);D^{0})=$$
$$Y(L(\delta(f|D));D^{0})=\delta(\hat{P}(Y^{0}(L(\delta(f|D));D^{0}))|D)=$$
$$=\delta(\hat{P}(L(\delta(f|D))|D)=\delta(L(\delta(f|D))).$$

that corresponds to the expected result.

For *n=-1; l=-1:* the result should be a first-order decontext within the dimensional plane
-1. Using (6.36), (6.37), (6.38) and (6.39):

$$Y^{-1}(f;D^{-1})=Y^{-1}(Y^{0}(f;D^{-1});D^{0})=Y^{-1}(L^{-1}(\delta^{-1}(f|D));D$$
$$\delta^{-1}(\hat{P}(Y^{0}(L^{-1}(\delta^{-1}(f|D));D^{0}))|D)=$$

$$=\delta^{-1}(\hat{P}(L^{-1}(\delta^{-1}(f|D))|D)=\delta^{-1}(L^{-1}(\delta^{-1}(f|D))),$$

that corresponds to the expected result.

6.5 Cumulative relevance potential function

In most of cases, the context consumer would like to know all available context to some extent. The previously described RPF points out some context at a certain distance from the subject of attention in a semantic network, however it is also possible that the consumer would like to know all context with a distance from the subject that is not greater than a certain value. The concept is presented on the figure 36.

The result contains the whole context up to some CCD (DCCD), therefore let's call the corresponding selection function *cumulative relevance potential function*. Its calculation is based on RPF. For a planar case (not considering any dimensionality):

$$\Delta^n(f) = \bigcup_{i=1}^{|n|} \delta^{qi}(f), \tag{6.40}$$

where $q = sign(n)$.

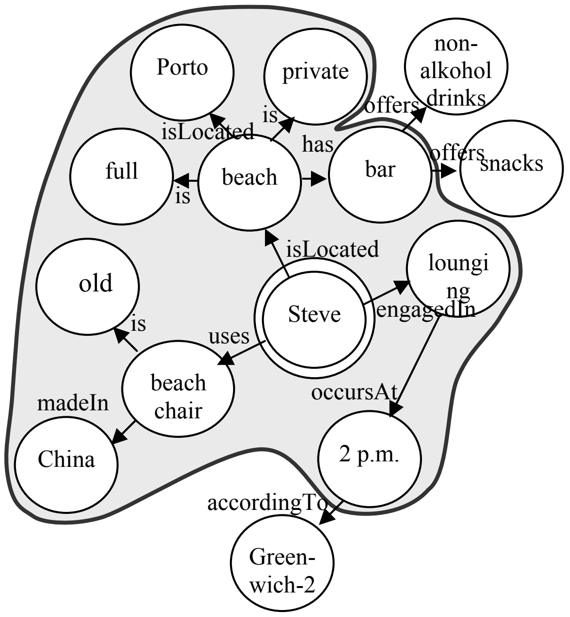


FIGURE 36 Cumulative context of the second order

Cumulative context is a set collected of all context triples beginning from the first order context of a subject up to some CCD level at each dimensional plane defined by dimensional property D up to a given DCCD level .

Figure 36 represents the 2-nd order cumulative context of the subject Steve in our sample domain with undefined dimensionality.

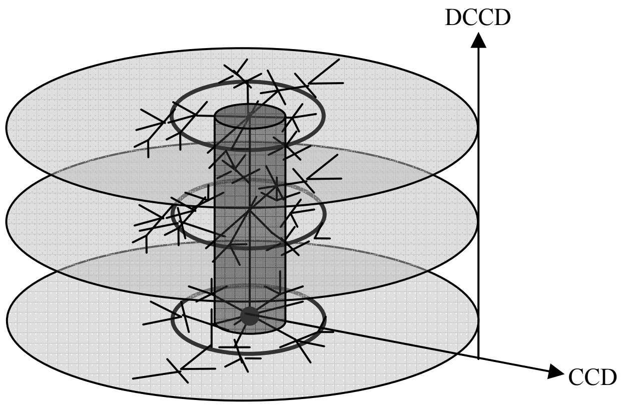


FIGURE 37 Schematic representation of cumulative context for semantic tree with root at focus point

Figure 37 above shows schematically the concept of cumulative context, when some dimension in semantic network is selected (the cumulative context is represented by the inner cylinder). Finally, dimensional cumulative context can be calculated using the following formula:

$$\Delta^n(f;D^l) = \bigcup_{j=0}^{|l|}\bigcup_{i=1}^{|n|} Y^{qi}(f;D^{vj}), \tag{6.41}$$

where $q = sign(n), v = sign(l)$.

6.6 RPF calculations example

Let us build a simple model of the domain from our example with Steve, as presented in Figure 38 on the next page. The semantic graph (a) on the left is based on real available information, the nodes representing entities (objects, activities, properties), and the links representing semantic relationships between entities. The graph (b) on the right is the conceptual model of the domain and contains semantic information about the entities themselves. In other words, the graph (a) in Figure 38 is the instance representation of the domain, whereas the graph (b) is its conceptual representation. The dashed links correspond to the property "is_a".

Why is it important to effect such a strict differentiation between these two? In a dynamic and resource-limited world of mobile applications a set of concepts in a particular domain remains much more stable than the real-world objects and processes that it represents. By separating instances from their conceptual descriptions it is possible to distribute their processing more efficiently.

Table 3 on the next page contains sets of triples *(subject, predicate, object)* describing the domain under consideration.

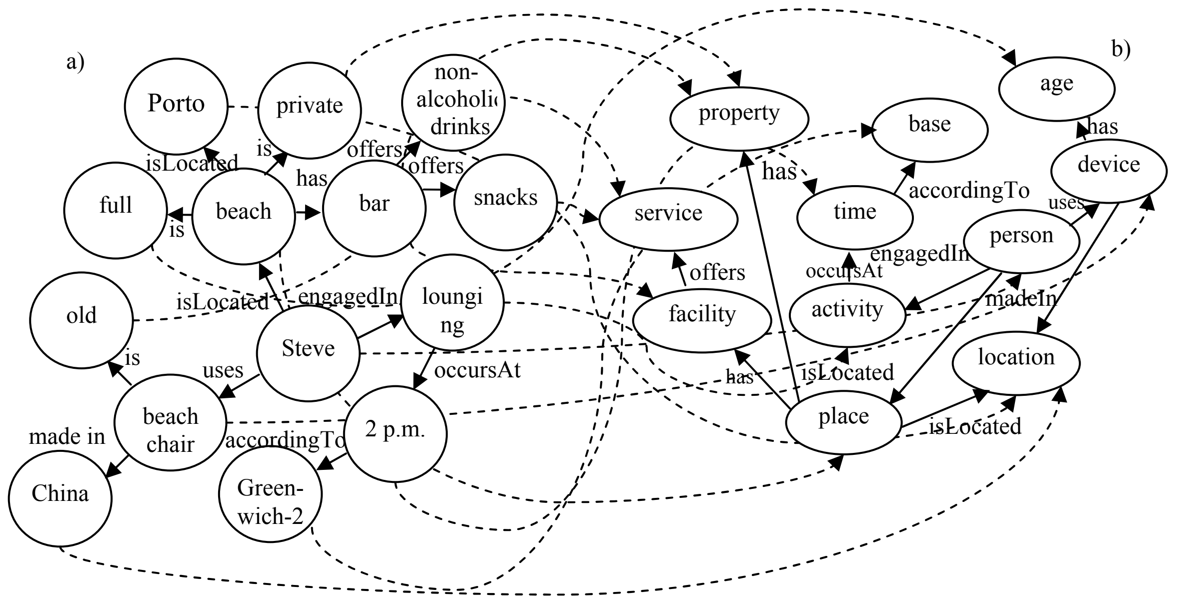


FIGURE 38 Sample domain model

TABLE 3 Formal description of the sample domain

Instances (Subject Value, Predicate, Object Value)	*Concepts (Subject Class, Predicate, Object Class)*
("Steve", is_a, "person")	
("beach", is_a, "place")	
("full", is_a, "property")	
("Porto", is_a, "location")	
("private", is_a, "property")	
("bar", is_a, "facility")	
("non-alcoholic drinks", is_a, "service")	
("snacks", is_a, "service")	
("lounging", is_a, "activity")	
("2 p.m.", is_a, "time")	
("Greenwich-2", is_a, "base")	
("beach-chair", is_a, "device")	
("China", is_a, "location")	
("old", is_a, "property")	
("Steve", engagedIn, "lounging")	(person, engagedIn, activity)
("lounging", occursAt, "2 p.m.")	(activity, occursAt, time)
("2 p.m.", accordingTo, "Greenwich-2")	(time, accordingTo, base)
("Steve", isLocated, "beach")	(person, isLocated, place)
("beach", is, "private")	(place, has, property)
("beach", is, "full")	

(continues)

TABLE 3 Formal description of the sample domain (continues)

(“beach”, isLocated, “Porto”)	(place, isLocated, location)
(“Steve”, uses, “beach chair”)	(person, uses, device)
(“beach chair”, is, “old”)	(device, has, age)
(“beach chair”, madeIn, “China”)	(device, madeIn, location)
(“beach”, has, “bar”)	(place, has, facility)
(“bar”, offers, “non-alcoholic drinks”)	(facility, offers, service)
(“bar”, offers, “snacks”)	

The model presented above is not a model of context, but a model of domain. Since the selected model of the domain has full connectivity, all information can be contextual, the only question is what will be relevant and what not for each particular case? This example represents a semantic network which clearly has two dimensional planes with respect to the predicate *“is_a”* (dimension of conceptuality).

If the aim is to operate with instances around the subject “Steve”, then the DRPF with dimensional property “is_a” and default DCCD (equal to 0) could be used with some value for CCD. For example,

Y^2(“Steve”; (”is_a”)^0)=
{(“lounging”, occursAt, “2 p.m.”),
(“beach”, is, “private”),
(“beach”, is, “full”),
(“beach”, has, “bar”),
(“beach”, isLocated, “Porto”),
(“beach chair”, is, “old”),
(“beach chair”, madeIn, “China”)}.

If the only conceptual plane is to be analyzed, then for this particular case the DCCD should be set to 1. For example,

Y^2(“Steve”; (”is_a”)^2)=
{(activity, occursAt, time),
(place, has, facility),
(place, has, property),
(place, isLocated, location),
(person, uses, device),
(device, is, age),
(device, madeIn, location)}.

The transitional triples from the first plane to the second one can be obtained as follows:

δ(“Steve”|”is_a”) =
{(“Steve”, is_a, “person”),
(“beach”, is_a, “place”),

(“full”, is_a, “property”),
(“Porto”, is_a, “location”),
(“private”, is_a, ”property”),
(“bar”, is_a, “facility”),
(“non-alcholic drinks”, is_a, “service”),
(“snacks”, is_a, “service”),
(“lounging”, is_a, “activity”),
(“2 p.m.”, is_a, “time”),
(“Greenwich-2”, is_a, “base”),
(“beach-chair”, is_a, “device”),
(“China”, is_a, “location”),
(“old”, is_a, “property”)}.

6.7 RPF extensions

Based on the previously described base and dimensional relevance potential functions it is possible to develop a quite powerful engine for context provision via extensions of base functions. The basic idea of extensions is seen as reducing the necessity of context information processing by the mobile application by requesting it from context provisioning service. In such case application should specify for the system what kind of processing is needed to be done. From its own side the system should support a corresponding functionality, in order to be able to provide necessary services for the application. The extensions to RPF can be various, some useful examples are given below.

Restricting queries

The main idea is to limit incoming context information by special conditions. For example, “only certain predicates are allowed”, or, “not including information about certain entities”, etc. Such restriction is nothing more than filtering outcome results with some condition. This seems to be a suitable place to include the time aspect and other qualitative parameters of context information (only context “not older than 1 day”; “having dynamicity rate not higher than once in a month”, etc.).

Multidimensional queries

Combining potentially relevant context along different directions can give interesting and useful context patterns. For example, the combination of “located_at” and “plays_role” could give a context of a certain work place regarding people employed there and present at a certain time. Moreover, there is no necessity to consider people by their identities, as only their roles matter. For example, “at Nokia Jyväskylä there are currently 3 managers, 10 engineers and 2 cleaners” – this information gives a clear picture, and answers the question whether it would be possible to perform certain tasks with the present personnel?

Allowing relaxation

There are quite a many techniques and approaches developed for ontology analysis with respect to synonymous properties/classes identification and ontology mappings. Those achievements could be efficiently used for context query relaxation – when an application defines some parameters for context search, which do not correspond to the ones in the domain semantic network, substituting them by synonyms might give great results for the application.

Inter-Domain Search

When there is a need and a possibility to use information from different domains (the general idea is described within CAME), context provisioning system might have such capability. The same RPF would be in the core, only the scope of the search would widen. Such capability would be especially useful for off-line search of context with high level of stability.

The set of RPF extensions presented is not complete, however it shows, how results of RPF-based context extraction could be used for further processing. Currently, there are many techniques developed allowing efficient analysis of RDF-based semantic networks, however, the general problem with their usage is that processing becomes quite slow on big amounts of data. The main advantage of the subject-centric approach for context extraction is that it allows quick and easy extraction of potentially relevant context from a large and complex semantic network that can be further processed by smart reasoning techniques.

6.8 Discussion

Ontological representation of data and knowledge is proven to be a powerful tool for achieving semantic interoperability between different applications among various interconnected environments and domains. However, reasoning of one particular application on some more or less extensive ontology (semantic network in a more general case) becomes impossible in mobile environment due to high level of dynamicity and lack of resources. Ontological reasoning, as it is, can not be directly applied by small mobile applications, which usually have efficiency as a primarily requirement for their design as shown, for example, in [3]. However, mobile environment is constantly changing and heterogeneous. It is clear that context-awareness capability would significantly enhance adaptability and quality of services provided for mobile users. On the other hand, context by its nature is semantic, multi-domain, multi-sided and subjective knowledge needing semantic processing and interoperability.

There are applications, which consider context within a particular small domain, build their own knowledge model, acquisition architecture and do reasoning within it. It may be a working and efficient approach for that particular application with its particular tasks, however, generally (if many such applications are acting within intersecting domains) it means that huge amounts of the same data is multiplied many times, separately acquired, and utilized, which leads to a total performance decrease in the environment. Therefore, we have the following conflicting circumstance:

– A mobile context-aware application wants to consider context from its own subjective viewpoint.
– The environment "doesn't want" to be overloaded by a separate context-related functionality of each particular application, preferring to provide a centralized service for acquisition, tracking and store of context data.
– Processing of all information available at the centralized environment is impossible for the mobile application due to lack of resources.

Centralized[15] (within the domain or environment) acquisition and storing of available information and knowledge, which could be contextually relevant to different applications, definitely requires a semantic interoperability technique since each application has its own point of view on the issues such as "what is context" and "how to interpret it". Interconnection of different domains, environments and cross-domain (for example, portable) applications implicitly require such interoperability. Semantic networks represented by ontological languages are able to provide semantic interoperability, but their processing is too slow to be performed within mobile applications.

Therefore, it becomes clear, that in order to follow ontology-based interoperability provision approach there is a need for efficient architecture for context-related functionality provision in order to achieve the desired result – ambient awareness. We see three key enablers for such architecture:

– splitting of reasoning process between application and environment;
– relevant context extraction for processing in each particular case;
– efficient storage mechanism for ontology-like data and knowledge.

In [3] the authors present an approach for splitting of reasoning process and suggest RDF-based data storage in relational databases. A concept of "off-line ontological reasoning", to be used together with rule-based reasoning for efficiency, is introduced. In their approach, reasoning is often performed based on simple data attributes and does not require ontological processing. Ontology-based data is precomputed by the system and reconsidered at the time of service request only in

[15] I.e., provided by the environment, for example, some context-provisioning middleware. It can be distributed by the architecture while controlled by the environment, not by the application.

some particular cases, when it is needed. These questions are not considered within this dissertation; however, the suggested solutions fit well into the idea of context-awareness supporting agency presented in this work. In addition to context data acquisition from [3] and request-based context provision described in our work, the agency could provide different kinds of additional supporting services for thin mobile clients, one of which could be off-line ontological reasoning.

Despite the fact that initial definitions in [80] are quite general, the authors point out a relatively promising way for relevant context selection. Relevance condition is a filter, which is used to identify one or more relevant entities out of the whole available set in the context provider domain. This filter corresponds to first-order context presented in this thesis (in [80]: "get all entities where you know something about the aspect place"). The second level filter corresponds to the partial second-order context (in [80]: "get all entities where you know that the current state with respect to the aspect *place* is *near*"). The third-level filter corresponds to QoC based selection of relevant context. The technique that the authors are using for relevant context selection is based on F-Logic queries, but it is not defined formally. Our context consideration depth model expands the idea of relevant context filtering into generally undefined level of consideration depth, adds a dimensional view onto semantic network based context data and gives more formal definitions for potentially relevant context selection.

The main contribution of this thesis regarding context provision is representing the semantic network as a multi-dimensional information space, which can be efficiently used for extraction of potentially relevant context for different purposes, subjectively along the point of view of the concrete task (application, context consumer) and in particular circumstances. The proposed model for structuring SN-based information space allows easy extraction of sub-networks according to the needs of the requesting entity. Some examples of relevant context extraction strategies presented in [89] are given below as an application areas for the context modeling approach presented in this work. Each of the described strategies reflects a certain class of context-aware applications, the reasoning of which is based on special relationships between classes and objects composing a local semantic network.

Part_of based context extraction

One of the most widespread classes of context-aware applications, to which location-based services belong, is based on nesting of information domains – a table is a part of a room, the room is a part of a building, the building is a part of a street, the street is a part of a city and so on.

When main reasoning or, in the more general case, the main application's functionality is based on object's direct properties (in our case, the table), the characteristics of entities, to which the object belongs as a *part_of* are considered to be a contextual properties of it with stronger or weaker degree of relevance. It is almost obvious that the properties of the room, where the table is located influence the behavior (properties) of the table; it might be also important, in what kind of building it is, what is the weather usually like at the place, where it is located and so on. In the terminology of [89], the properties of the table would be predictive, of the room, street, etc. – i.e., contextual ones. The Figure 39 represents the concept of nested domains.

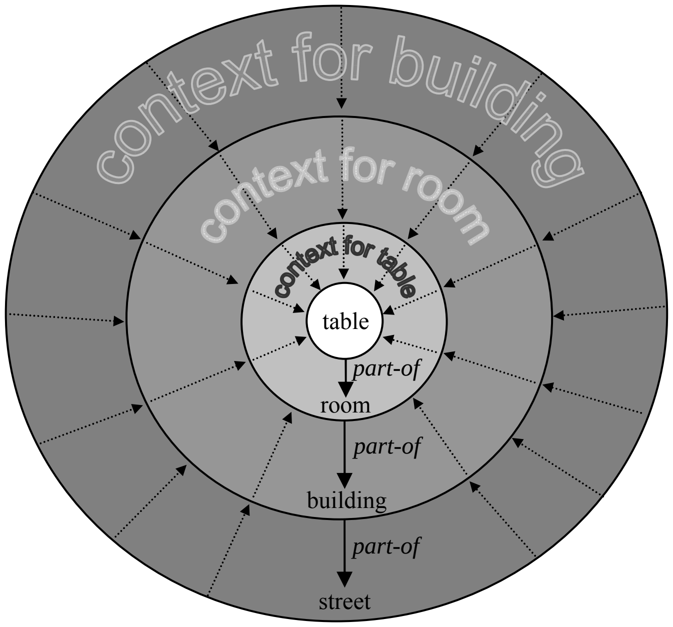


FIGURE 39 Part_of based context

The model that is presented in this thesis for potentially relevant context selection allows easy extraction, on-request, of a sub-network based on the property "part_of" for further processing directly by the context-aware application.

***Is_a* based context extraction**

Another direction of context analysis is based on classification of available instances along the space of conceptual definitions. Each object/entity usually belongs to some class, which is a sub-class of a more general class and so on. This is a classical application area for ontology-based reasoning. When there is direct information about some object to deal with, in many cases it is useful to know what kind of class hierarchy is behind it. One of the example applications could be an analysis of information represented with a vocabulary that is not completely known, to find out the meaning of the unknown entities and their properties. Information about super-classes in this case is contextual, it helps understanding of entity's semantic meaning and selection of appropriate techniques to operate with it. Returning to the example with table, let's imagine that a guest from another planet will come to the Earth and see the table. He wouldn't know what it is and what to do with it. While analyzing table's upper classes (example shown in Figure 40), he would become aware of its general properties and might decide that he wouldn't need to use it at all. The information about the class hierarchy is contextual along the level of conceptualization.

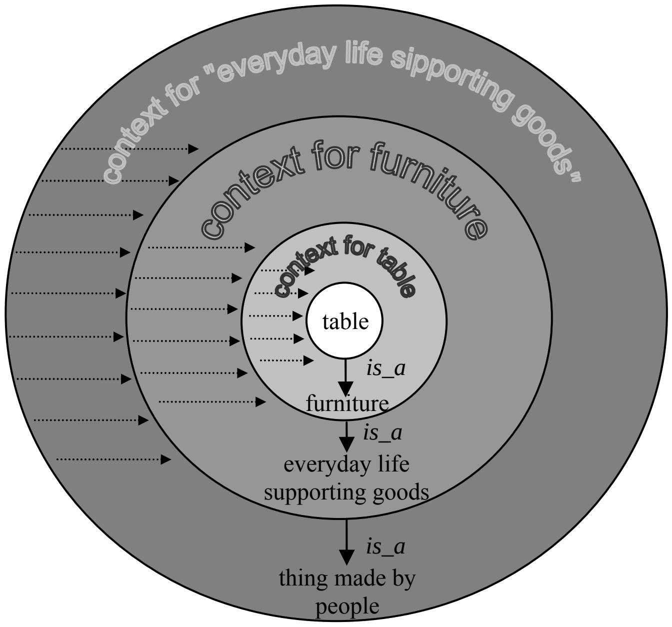


FIGURE 40 Is_a based context

The potentially relevant context selection approach presented in this work can be directly used for extraction of class hierarchy of certain objects or classes. The advantage of using the presented technique particularly for this class of applications consists in the uniformity of the context extraction technique – a completely different set of applications could be served with context information in the same way. From a human point of view, treatment of class hierarchy is very different from treatment of "part_of" hierarchy; however, sub-network selection looks very similar – "select connected entities up to certain level of depth", "how entities (in this case classes) are interconnected with each other on each level of abstraction?", "what kinds of properties a certain super-class of the object of attention has?", etc.

Role-based **context extraction**

The context extraction strategy related to objects with a proactive behavior is referred to in [89] as a “role-based context extraction”. It is another stratum of context, which can be used for solution of certain tasks.

This might be perceived as a combination of “part_of” and “is_a” properties, where “part_of” defines the organization and “is_a” defines the role-corresponding class within this organization. However, this is not so. The role defines an entity's expectations within a certain community, but not its actual behavior, while belonging to some class means full correspondence to its defined properties. Moreover, a role-based context would not return all the attributes of the organization (community), in which the entity is involved. The proactive unit (usually, a person or agent) exhibits certain behavior, which might often correspond to its role. An example of role-based nested context is shown in Figure 41.

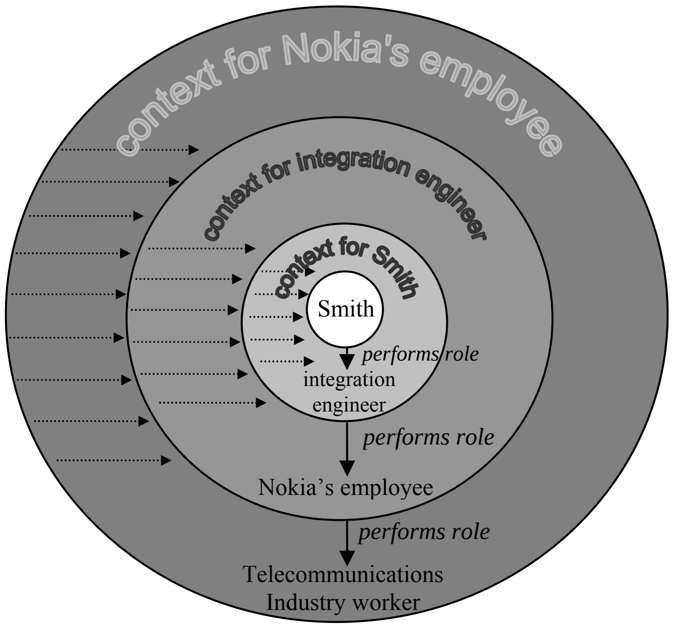


FIGURE 41 Role-based context

Usually, proactive objects perform, at the same time, several roles within different communities. For example, John Smith can be a director of research center X, the father of two children, a moderator of an Internet forum and a fan of football club Y. As mentioned in [89] combination of these roles could give quite useful context information for the corresponding class of tasks; a personal digital assistant planning the person’s schedule would definitely benefit from this information.

The context model presented in this work allows direct extraction of role-based context for further processing, for example, in case of conflicting role descriptions, when additional decision making is needed for resolving such conflicts.

***Interface-based* context extraction**

Interface-related type of context is the most subjective one. It divides corresponding applications into very small groups, serving of which with relevant context is, however, of great importance. "Interface" is a certain view onto a set of properties describing the object/entity of interest. It might correspond to a graphical interface of an application or a specific service, provided to an intelligent agent. The idea of interface-based context highly reflects the concept of "point of view" onto context. Figure 42 represents the four different interface-related contexts: software context would be relevant to a software engineer analyzing a functionality error in a mobile device; mechanical context would offer information to a production factory planning to supplement material order; usability context would be interesting to a costumer buying a device; location context can be used by a location-based service, for example, for sending a local map to a device.

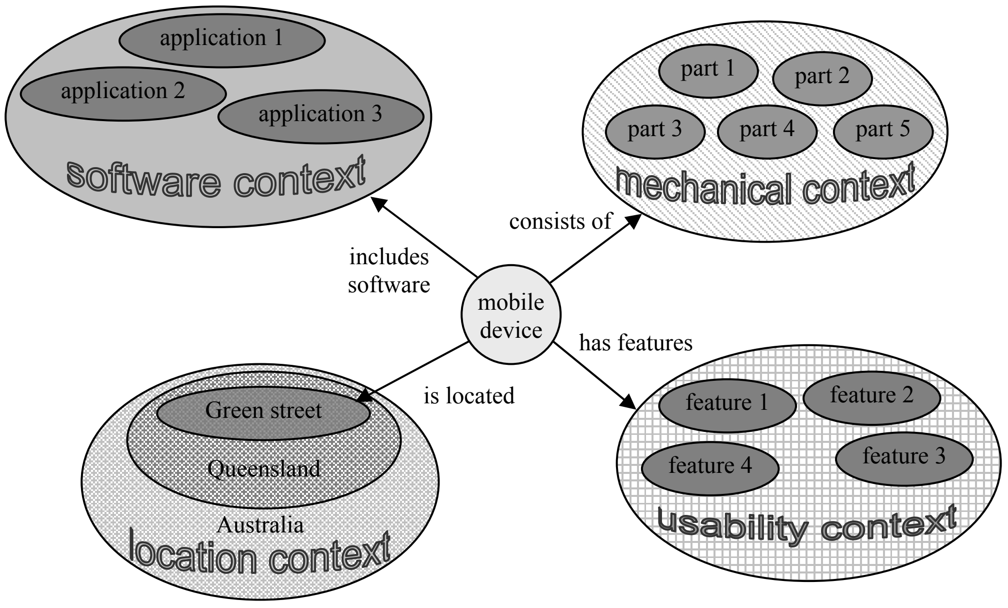


FIGURE 42 Interface-based context

Usually, these kinds of applications have needs so specific that they themselves have to perform all the constituents of context-awareness capability support – acquisition, storing, semantic annotation, reasoning, conflict resolution, etc. On the other hand, such applications are, usually, quite small and the main functionality they perform does not require complicated reasoning; therefore, it becomes quite inefficient and useless to make them context-aware. In the author's point of view, this is the reason why not many context-aware applications are currently functioning in real mobile environments. An exception to this is location-aware services, which use basically only one context parameter – location (we don't consider time here since it is usually available for any application within any environment).

Such applications can be considered, basically, as the main consumers of the suggested technique for potentially relevant context selection. An application like this would give some small piece of information to the context provisioning system, which in turn would provide it with a sub-set of context information, which very likely would correspond to its needs.

Vocabulary(ontology)-based context

A classical task in the Semantic Web is how to build correspondence between different ontologies describing the same concepts in different terms. This thesis in no way pretends to provide solution for this complex problem. However, intelligent applications, dealing with ontology mapping could also be efficiently served by a context-provisioning system based on the presented subject-oriented approach.

Figure 43 gives an example of four equivalent ontologies. Ontologies have a nested nature; thus, using the presented context modeling approach, an easy movement along the horizontal (within ontologies) and vertical (along conceptuality levels) directions of ontological information space is possible.

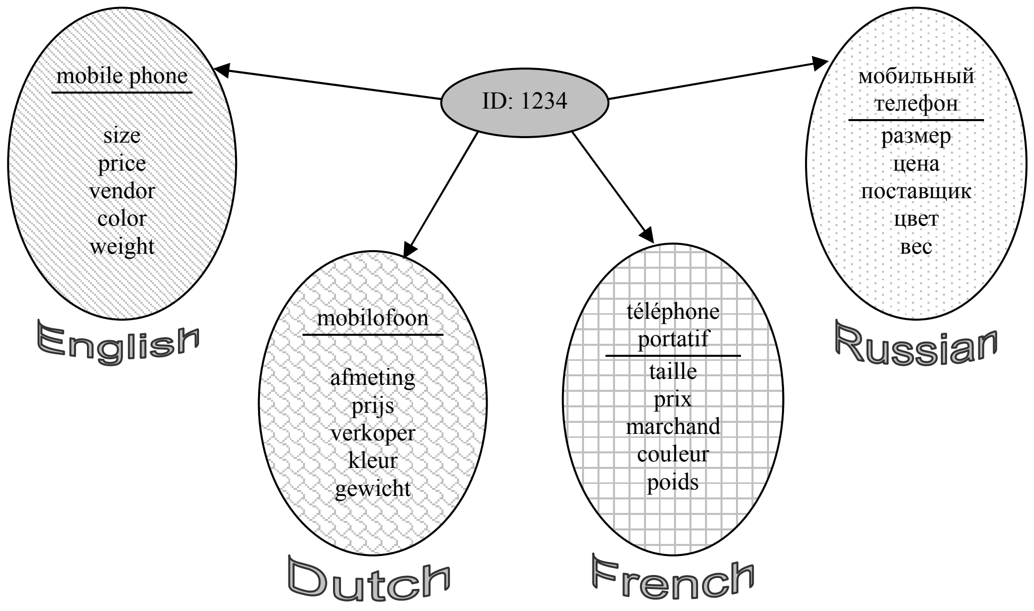


FIGURE 43 Ontology-based context

Implicitly defined context

The question of knowledge space partitioning has been raised already in [59], where the authors define semantic metanetwork as a set of semantic networks located on top of each other, where each next (meta-)level of the network defines the configuration of the previous one. This concept has been further developed, for example, reasoning with semantic metanetwork was comprehensively studied in [85].

This idea was further developed within the context of probabilistic reasoning; Bayesian metanetworks were used in [86] for modeling web-user preferences and further studied in [87]. Bayesian metanetwork, according to [87] consists of Bayesian networks, which are superimposed on each other in such a way that conditional or unconditional probability distributions associated with the nodes of each previous probabilistic network depend on probability distributions associated with the nodes of the next network. The operation of Bayesian metanetwork and semantic metanetwork is different; however, the general idea (see figure 44) is the same – each higher-level network defines the structure of the lower-level network.

Later research further develops and applies, in practice, the theoretical foundations presented. In [47] the authors suggest, as an extension to RDF network, a contextual condition for RDF statement that defines its truth or falsity under certain conditions. That extension would apply to an RDF statement and link it by a relation "trueInContext" ("falseInContext") to the container of other RDF statements, which form the contextual constraint to the statement in question. An application of the framework for industrial resources is presented in [46].

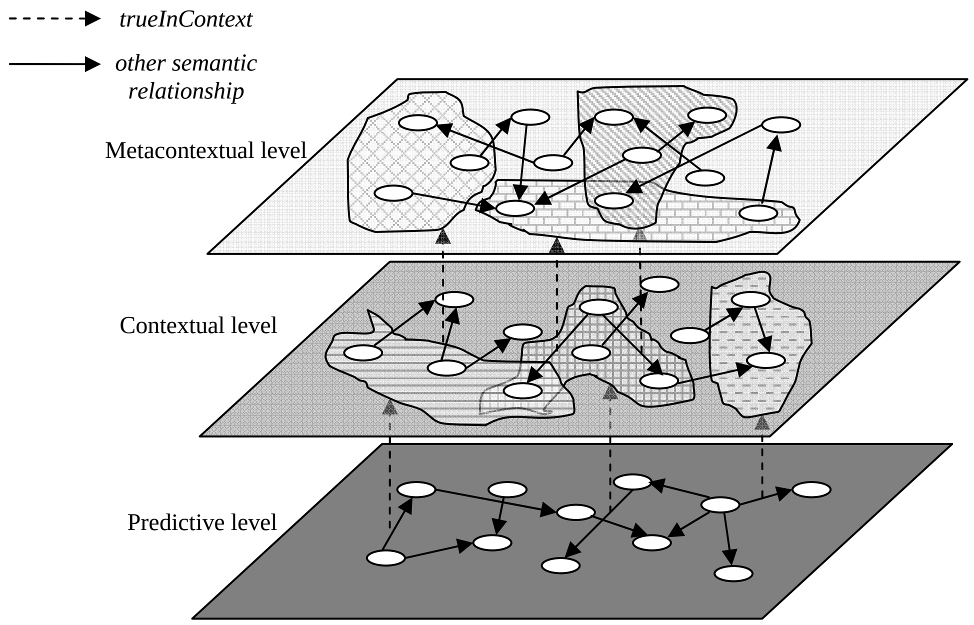


FIGURE 44 Context-sensitive semantic metanetwork

The presented framework for context-sensitive metadata description introduces an implicit definition of context within semantic (RDF) network as a contextual condition for RDF statement. Context extraction from such semantic networks is also possible using the context retrieval mechanism presented in this dissertation.

The described case of semantic network is different in that a semantic relationship (in this case, *trueInContext*) is associated not with an object but with the whole statement. It can be also considered as a triple, in which, however, “subject” is a statement, “object” is a set of triples and “predicate” is *trueInContext*. This is a one-to-many type of semantic relationship where certain property of a triple is defined by a group of other triples. In the more general case, it can be a set of statements connected to a set of other statements by a certain many-to-many semantic relationship.

When presenting the philosophical model of context, it was mentioned that “subject of attention” can be a simple object or groups of objects, such as statement, situations, groups of people, etc. This can be treated as a special, contextual, dimension of semantic network and operated in the same way as other, more simple dimensional relationships.

The described relationship between semantic sub-networks is not such a freakish case of semantic relationship as one might assume. For example, when some certain information space is defined using a certain vocabulary, it is exactly this kind of relationship – semantic network of a given domain is “describedInTermsOf” a concrete vocabulary that is another semantic network. Basically, both of these semantic networks are sets of triples, which as a whole are connected by a certain semantic relationship.

The context model presented here is still valid for context of complex subjects (for example, triple). The only difference is in the interpretation – in ordinary semantic networks subject is usually understood to be a simple entity (object) having semantic links to other objects. In networks with group relationships, as in context-sensitive semantic metanetworks, a group (in particular, a triple) can also play a role of a subject in a semantic relationship. Thus, when such group is considered as a subject for a potentially relevant context consideration, those semantic links which connect the group as a whole, and not its members separately, should be considered as having a contextual relationship to a group as a subject of attention.

6.9 Conclusions

Most likely the majority of context-aware applications would not need such powerful engine as that provided by RPF. In many cases only some very limited set of context information is needed by an application due to its small size and limited functionality. The more precisely context information is wanted to be analyzed and interpreted the more powerful should the BDI engine inside the context consumer be.

One clear advantage of the presented context discovery engine is that it suits for a wide range of context consumers – from plain thin clients to powerful reasoners. The second advantage is that the relevance potential function in the core of the presented engine is simple and universal – it can be extended by a wide range of intelligent services provided by the context information system for context consumers.

Use of such function would enable the context provisioning system to collect and utilize context patterns, provided for different applications in different situations. This issue is not considered in this thesis; however, its great potential utility in context-awareness capability provision should be noted.

7 CONTEXT CONSUMER: DECISION MAKING PROCESS

This section is devoted to context processing from a context consumer point of view. Context-aware application is characterized as context consumer in 7.1 and 7.2. States and sources of context information are discussed in 7.3. We present an example approach for context facts derivation in 7.4 and propose decision enabling rules to be used for context-aware actions in 7.5. The Sub-section 7.6 introduces rule utility weights, which are used for conflict resolution and context analysis in 7.8. A primitive context reasoning model, showing the process of context-aware reasoning, is described in 7.7. Sub-section 7.9 depicts constituents of context request from consumer to information provisioning system. We conclude in Sub-section 7.10.

7.1 Context-aware application as a context consumer

One of the main assumptions in this work enabling presented interactive reasoning is that context consumer and information provisioning system can be implemented, in practice, using very different techniques and approaches. The main thing to remember is that the data model behind their functionality should be the same, i.e., semantic networks (described using RDF, or RDF-based languages). In addition, they need an information exchange channel with some common protocol. Therefore, a context consumer's reasoning engine can be based on any reasoning technique. This section, however, represents the author's own vision on the issue.

The main assumption of a consumer involved in decision making is that it is able to make decision(s) based on content information only without any context. However, if there is a possibility to obtain more information from some source (for example, information system, or another software entity), which would be contextually relevant, the decisions subsequently taken might be more effective and appropriate for that consumer's needs.

The actions needed to be performed by the consumer in order to make optimal context-aware decisions (see figure 45) are the following:

- based on the received content information and BDI, form a request to the system about context information that could be relevant and support better decisions
- analyze the received context information, revise BDI, investigate the decision making potential and ask for more information if needed (repeat a reasonable number of times)

The main goal of context-aware interactive decision making performed by a consumer is to make the best possible decisions within the least possible number of information exchange iterations in the information system. Therefore, a precise definition of subject(s) of attention should be sent to the system within a context request at each iteration. This would need to be done in order to obtain information which would lead to the fastest possible clarification of the context situation in one side and to the most preferable solutions in the other side.

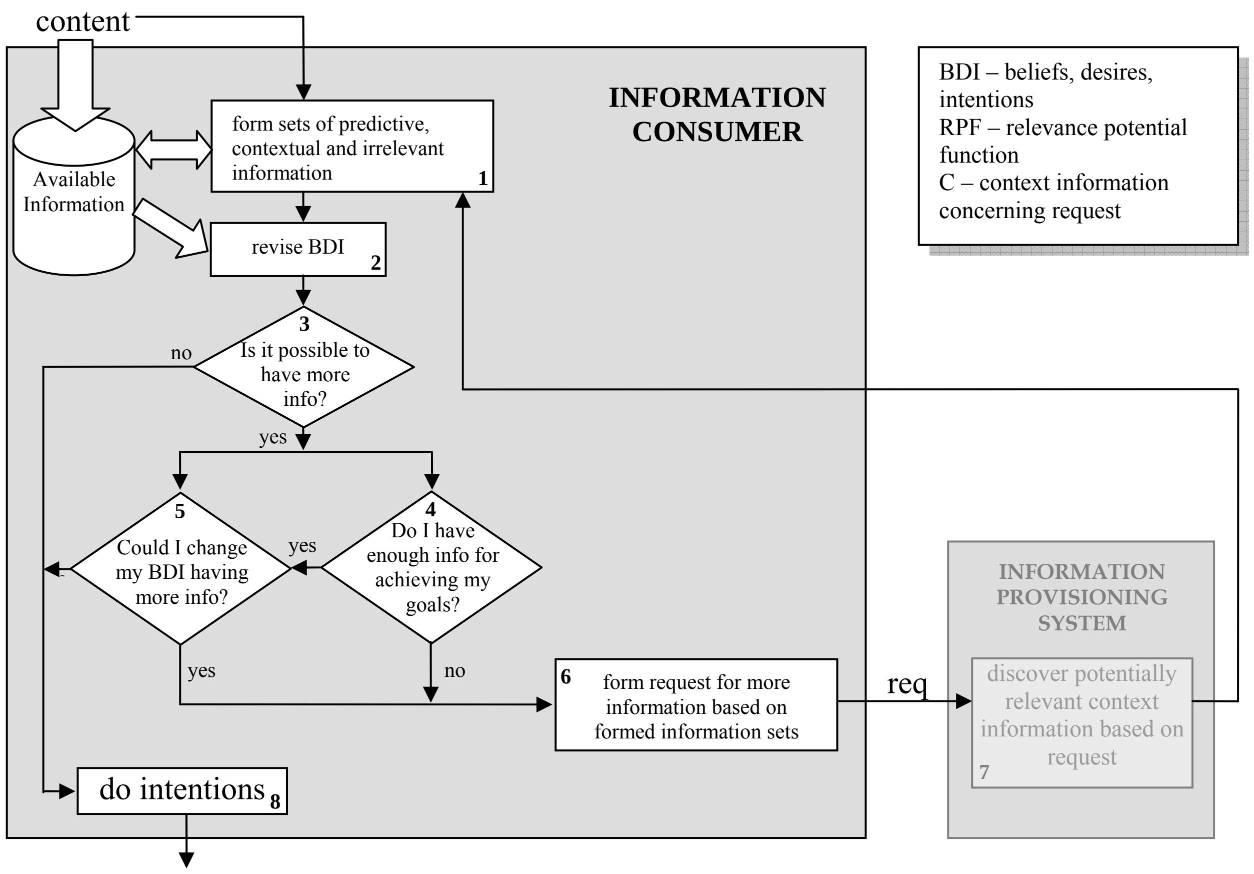

content
INFORMATION CONSUMER
Available Information
form sets of predictive, contextual and irrelevant information 1
revise BDI 2
3 Is it possible to have more info?
no
yes
5 Could I change my BDI having more info?
4 Do I have enough info for achieving my goals?
yes
yes
no
6 form request for more information based on formed information sets
req
do intentions 8
BDI – beliefs, desires, intentions
RPF – relevance potential function
C – context information concerning request
INFORMATION PROVISIONING SYSTEM
discover potentially relevant context information based on request 7

FIGURE 45 Context consumer within interactive reasoning

7.2 Characteristics of context-aware application

Let's define the following implicit characteristics of a context-aware application:

- parameters,
- structure,
- utility.

Parameters define a set of input and output parameters of the application. *Structure* is a set of simple functions (modules) and their combination conditions. Combination of simple functions, which is able to produce an output of the application is its *configuration*. Utility is a function which defines the effectiveness of any application's configuration with respect to its goals (utility function can be based on resource usage, cost of performing actions, precision of calculations, probability of target result achievement, etc). *Contextual conditions* define the applicability of context-sensitive operational rules (modules, features) for different contexts.

In [88] the author distinguishes the following attributes within context-aware application: target attribute, predictive attribute and contextual attribute. While the actions performed by a context-aware application use predictive attributes as an input parameters, the structure of an application used for producing output is defined by contextual attributes. Let's explain the above definitions with the help of the following artificial example.

Let context-aware application be a function, calculating the sign of the input variable x:

$$y = \mathrm{sign}(x)$$

The calculation can be done using the following formula:

$$y = \begin{cases} 0, x = 0 \\ \dfrac{x}{|x|}, x \neq 0 \end{cases}$$

Therefore, depending on certain characteristic of the input value (is it equal to 0 or not) the result is calculated. Thus, if we consider this trivial example as a context-aware application, we have:

Target parameter:	y
Predictive parameter:	x
Two calculating functions:	f1: y = 0
	f2: y = $\dfrac{x}{\lvert x\rvert}$
Contextual parameter:	z = "x, equalTo, 0".

For this case, the structure of the application (context-sensitive way of its functionality) can be defined using the following conditions:

(f1, trueInContext, z)

(f2, falseInContext, z)
(f2, trueInContext,$\neg$z)
(f1, falseInContext,$\neg$z).

In this example, the contextual parameter can be obtained from the predictive one, however it can be (as in most cases) a completely different parameter(s), correlated or uncorrelated with a predictive one(s).

7.3 States and sources of context information

In the majority of logic-based decision making systems it is assumed that information facts (statements) can have only two states: true or false. Usually, it is assumed that if a statement is not true, it automatically means that it is false. However, in the real life this is not the case. How many times has it happened that, when someone asks you about something, you answer "I don't know". This means that it is not known whether the fact in question is true or false. In order to get a clearer idea additional information is needed. For the purpose of this exercise, let us agree that context facts have the following three states:

"true", "false" and "n/a"

"True" and "false" indicate that it is known whether the fact is true or not. "N/a" means that either there is no information about the fact or the information provider carries conflicting information, which doesn't allow any conclusions to be made about whether the fact is true or false.

In the theory of intelligent agents there are three classical ways for obtaining information, which is used as a base for decision making:

– observation,
– communication,
– reasoning.

Observed information refers to information that an intelligent agent is able to observe (or is somehow informed) directly from the environment, e.g. "temperature is 15°C" (meaning that the circumstance occurs in the current place at the current time, where/when observation is done). Communicated information refers to information which is requested and received from other agents or information sources. The main difference between these two is that in the second case the information is not necessarily true, and can be conflicting or uncertain. Reasoned information refers to information that is produced by the agent itself based on the received information and his implicit reasoning rules. For example, having information that "it is not sunny" and "people are using umbrellas", "it is over 0°C" the agent can conclude that "it is raining". Such separation of incoming information channels applies well to the interactive reasoning principle presented here.

Content refers to something coming to the context consumer through its own input channels. The consumer, basically, doesn't need any decision making or reasoning in order to get it. For the context consumer, content is the main information. Content arrives through its main input channels and supports its main functionality; thus, the context consumer is not able to operate without content. Content can be whatever: a state of a radio link received via listening to a radio channel receiver, air temperature obtained from electronic thermometer, etc. Therefore, content is information that is independently observed by the consumer.

Context information received by the context consumer from an information system and based on his request is, definitely, communicated information: the consumer asks about some issue, and the system responds with some information about it (or with "no info"). Moreover, the context consumer might also be able to request some information not only from the information system, but also from other context consumers, who share information exchange channels with it and are able to answer to given requests.

However, there is a third way to obtain information – by producing it. If the context consumer is able to draw conclusions that are based on the received information, then it needs much less information obtained by observation or communication and can use its own reasoning rules for producing information facts necessary for decision making.

7.4 Context facts derivation

The most suitable formula to apply here would be the McCarthny's formula [51]: true_in_context (*a, b*), where *b* is a logical function of information facts (these might be observed, communicated or even derived) and *a* is true if *b* is true. Negation also applies here, therefore, derivation rules could be ¬true_in_context (*a; b*), which means, basically, false_in_context (*a; b*), where *a* is the argument, and *b* the condition. Here it is important to note that different derivation rules could "switch" certain facts on and off. For example, the following fact: "it is autumn" (considering a certain time point) is expected to be true in the context "it is October", however if in addition to the fact that it is October we have the fact that "it happens in Australia" we can say that definitely "it is not autumn", but "it is spring". So, we have the following derivation rules available:

R1: true_in_context ("it is autumn"; "it is October")

R2: false_in_context ("it is autumn"; "it is October"∪"it happens in Australia")

R3: true_in_context ("it is spring"; "it is October" ∪"it happens in Australia")

Therefore, we need to define the priority of those rules and the way they work with each other.

Context has one interesting property – cumulativity. Information can be inconsistent, conflicting or insufficient, but as long as the context consumer is receiving context, he has more and more information - newer context is accumulated and added to the previously received (except in cases where it becomes outdated[16]).

This property could be used for conflict resolution between such rules as R1 and R2 in our example.

Therefore, let us define *consistency rules* for conflicting context information.

– Each derivation rule can be used only once.
– Derivation rule is valid (true) as long as it contains valid information and there is no satisfied contradicting rules with the same argument.
– If two or more contradicting rules with the same argument are satisfied at the same time and the conditions of the first is a subset of the conditions of the second, the rule with the least number of conditional facts is processed first. Other rules cannot be applied until the conflict is resolved.
– Rules, which have undefined facts in the condition (n/a) cannot be applied.

What this means in practice? Let us illustrate it by the following example.

Example 1.

A context consumer has the following derivation rules (collected in Table 4):

R1:	true_in_context ("it is autumn"; "it is September")
R2:	false_in_context ("it is autumn"; "it is September"∪"it happens in Australia")
R3:	true_in_context ("it is spring"; "it is September" ∪"it happens in Australia")
R4:	false_in_context ("it is autumn"; "it is July")
R5:	true_in_context ("it is summer"; "it is July")
R6:	true_in_context ("it is autumn"; "previous month was August")
R7:	false_in_context ("it is autumn"; "previous month was August" ∪"it happens in Australia")
R8:	true_in_context ("it is spring"; "previous month was August" ∪"it happens in Australia")
R9:	true_in_context ("it is September" ; "previous month was August")

TABLE 4 Derivation rules for example 1

	It is September	It is September ∪ It happens in Australia	It is July	Previous month was August	Previous month was August ∪ It happens in Australia
It is spring		R3: true	R4: false		R8: true
It is summer			R5: true		
It is autumn	R1: true	R2: false		R6: true	R7: false
It is September				R9: true	

[16] We do not consider time issue here assuming that the information is valid for a certain time period, during which derivations and decisions are made

So, we have four context facts to be confirmed or rejected: “it is autumn”, “it is spring”, “it is summer”, “it is September”.

1. Let us assume that initially a context consumer gets the information that “Previous month was August”. Based on own derivation rules the context consumer makes the following conclusions:

Available context facts:
“Previous month was August”

	It is September	It is September ∪ It happens in Australia	It is July	Previous month was August	Previous month was August ∪ It happens in Australia
It is spring					
It is summer					
It is autumn				R6: **TRUE**	
It is September				R9: **TRUE**	

Derived context facts:
“It is September”

Disabled derivation rules:
n/a

2. Derived context fact satisfies the other rule:

Available context facts:
“Previous month was August”
“It is September”

	It is September	It is September ∪ It happens in Australia	It is July	Previous month was August	Previous month was August ∪ It happens in Australia
It is spring					
It is summer					
It is autumn	R1: **TRUE**			**TRUE**	
It is September				R9: **TRUE**	

Derived context facts:
“It is Autumn”

Disabled derivation rules:
n/a

3. Then from some other information source the consumer gets the context fact that “It happens in Australia”. Conclusions:

Available context facts:
“Previous month was August”
“It is September”
“It happens in Australia”
“It is Autumn”

	It is September	It is September ∪ It happens in Australia	It is July	Previous month was August	Previous month was August ∪ It happens in Australia

It is Spring		R3: **TRUE**			R8: **TRUE**
It is Summer					
It is Autumn	Disabled by R7, R2	R2: **FALSE**		Disabled by R7	R7: **FALSE**
It is September				**TRUE**	

Derived context facts:
"It is Spring"
¬"It is Autumn"

Disabled derivation rules:
R7: true_in_context ("It is Autumn"; "It is September")
R6: true_in_context ("it is Autumn"; "previous month was August")

Let us consider a conflicting case where a consumer receives two context facts at the same time: "Previous month was August" and "It happens in Australia". In this case rules R7 and R6 are conflicting. According to the third consistency rule the derivation process is spread to two phases:

Phase 1.

Available context facts:
"Previous month was August"
"It happens in Australia"

	It is September	It is September ∪ It happens in Australia	It is July	Previous month was August	Previous month was August ∪ It happens in Australia
It is Spring					
It is Summer					
It is Autumn				R6: **TRUE**	
It is September					

Derived context facts:
"It is Autumn"

Disabled derivation rules:
n/a

Phase 2.

Available context facts:
"Previous month was August"
"It happens in Australia"
"It is Autumn"

	It is September	It is September ∪ It happens in Australia	It is July	Previous month was August	Previous month was August ∪ It happens in Australia
It is Spring					R8: **TRUE**
It is Summer					
It is Autumn				Disabled by R7	R7: **FALSE**
It is September				R9: **TRUE**	

Derived context facts:
"It is Spring"
¬"It is Autumn"
"It is September"

Disabled derivation rules:

R6: true_in_context (“It is Autumn”; “Previous month was August”)

Then the conditions of the R1, R2 and R3 can be satisfied. Similarly, resolving the conflict between R1 and R2, R1 works first:

Available context facts:

“Previous month was August”
“It happens in Australia”
“It is Spring”
¬”It is Autumn”
“It is September”

	It is September	It is September ∪ It happens in Australia	It is July	Previous month was August	Previous month was August ∪ It happens in Australia
It is Spring					**TRUE**
It is Summer					
It is Autumn	R1: **TRUE**			Disabled by R7	Disabled by R1
It is September				**TRUE**	

Derived context facts:

”It is Autumn”

Disabled derivation rules:

n/a

then applying R2 and other satisfied conditions:

Available context facts:

“Previous month was August”
“It happens in Australia”
“It is Spring”
”It is Autumn”
“It is September”

	It is September	It is September ∪ It happens in Australia	It is July	Previous month was August	Previous month was August ∪ It happens in Australia
It is Spring		R3: **TRUE**			**TRUE**
It is Summer					
It is Autumn	Disabled by R2	R2: **FALSE**		Disabled by R7	Disabled by R1
It is September				**TRUE**	

Derived context facts:

¬”It is Autumn”

Disabled derivation rules:

R1: true_in_context (“it is autumn”; “it is September”)

The above example shows context derivation dynamics on two different sets of initial data. In this example, the conflicts are resolved successfully. The reason is not in the universality of the presented consistency rules, but in more or less consistent set of derivation rules available to the reasoner. However, in more or less real reasoning situations, especially when the derivation rules might be obtained dynamically from some sources with a known degree of trust, situations, where conflicts cannot be resolved as easily might appear. This question is raised in Sub-section 7.6.

7.5 Decision enabling rules

In this work we suggest the use of "*decision enabling rule*. That rule produces a context fact, which clarifies whether a certain decision can be taken or not. Such approach provides the context consumer with the following benefits:

- Context derivation and decision enabling processes are unified and can be performed using the same algorithms.
- Context consumer is able to base its decision on any additional factors, and decide which decision to take if more than one are available.
- Context consumer is able to search for a more desirable decision even though some basic decision has already been allowed.
- Action can be based on several allowed decisions (service can be composed from several allowed parts).

This subsection describes decision enabling rules used by context consumer for making decisions.

Each decision making rule would be represented by a set of facts (are true) in the current (decision making) time moment. So, the basic input for decision making is a set of context facts, disregarding the way how they have been obtained (observed, communicated or derived).

Decision making rules are similar to context derivation rules – each decision making rule is a logical multiplication of context facts with a special argument "decision D allowed". The general form of a decision making rule would be the following:

$$RD_I = true\_in\_context(\text{"decision } I \text{ allowed"}; \bigcup_{i=0}^{kI} fact_i),$$

where I is the decision number, kI = the number of facts considered to be preconditions for making the decision I, $fact_i$ = a positive or negative precondition of the decision I (context fact being true or false).

Let us consider the following possible decisions for context-aware information provider:

Decision 1: transmit through channel A
Decision 2: transmit through channel B
Decision 3: wait for 5 seconds
Decision 4: cancel transmission

The following set of decision enabling rules could be applicable here:

RD_1: true_in_context (“decision 1 allowed”; “channel *A* is available”, “receiver is available”*);*

RD_2: true_in_context (“decision 2 allowed”; ¬“channel A is available”, “channel B is available”, “receiver is available”*);*

RD_3: *true_in_context* (“decision 3 allowed”; ¬“channel A is available”, ¬“channel B is available”, “waiting time less then 15 seconds”*);*

RD_{41}: *false_in_context* (“decision 4 allowed”; “waiting time less then 15 seconds”*);*

RD_{42}: *true_in_context* (“decision 4 allowed”; “¬receiver is available”).

RD_{41} and *RD_{42}* are conflicting; the conflict resolution approach is described in the following section.

7.6 Rule utility weight

Context derivation rules and decision making rules are similar by their nature, therefore they are considered together in this section. Let us assume a situation where we have two conflicting rules with the same argument and different conditions:

R1: *true_in_context* (“information can be received”; “information provider is available”, “information provider is ready to send the information”);

R2: *false_in_context* (“information can be received”; “network is busy”).

Moreover, information satisfying both of them is coming at the same time, so we will have received the following set of facts:

“information provider is available”,
“information provider is ready to send the information”,
“network is busy”.

One would say that at the stage of forming rules it is necessary to define additional rule or conditions which would resolve this conflict, for example, R1 could be the following:

R1: *true_in_context* (“information can be received”; “information provider is available”, ¬“network is busy””).

However, there are many situations in which it is possible for a reasoner to have a nice, completely consistent set of derivation/decision making rules. For example, in the following situations:

– when rules are produced automatically, without human intervention, for example, when based on some observations;
– when some derivation (or decision making) rules are received in run-time from some source, for example, from some reasoning agent or some control engine;
– when it is desired that context derivation and decision making is done even on an incomplete set of data.

In these cases, it would be very difficult for the reasoner to modify the available set of rules in order to get a completely consistent set of rules. However, conflicts need to be resolved in any case. For information conflicts resolution, and with the aim of increasing optimality of conclusions/decisions made by the context reasoner, let us introduce rule utility weight.

> *Rule utility weight (RUW) is a real number within the range* [*0,1*]*, which defines a utility relationship between rules inside a context reasoner.*

Utility relationship can be understood as follows:

1) If there are two conflicting derivation rules with the same argument and different sets of conditional data, the one with the highest utility weight will be considered. In practice, this means that if one rule says "false" and the other rule says "true" about the same fact, and the rule satisfaction conditions differ, the one with higher RUW has a higher probability to be right

2) If more than one decision can be made, the one with the higher RUW will be selected. This means that a decision making rule preferable for the context reasoner should have a higher RUW value

3) If context derivation and decision making can be done at the same time (on the same set of data), context derivation should be done first. This means that decision making rules should usually have lower rule utility weight than context derivation rules to allow context processing to be done before decision making (except in those cases where making decision is much more preferable than further information processing for the reasoner).

Let us suggest the following policy for work with RUW:

– RUW values are dynamic and can be adjusted by the context reasoner.
– By default (if RUW are not defined), all rules have the same weight – 1.
– Disabled rules/decisions have an utility weight equal to 0.
– Rules are processed according to their utility weights – rules with higher weights are processed earlier.
– Conflicting rules with the same RUW have an equal impact on the result:
    a) if those rules define a fact, truth/falsity of it becomes unknown;
    b) if those rules define a decision, its allowability is unknown.
– Non-conflicting decision making rules with the same RUW mean that any decision available can be made with the same utility by the reasoner.

Rule utility weights will not be defined any further in this thesis. We presume that they could be defined, for example, as follows:

– implicitly within the reasoner if created in design time;
– if received during run-time, there might be some smart formula, which defines *RUWs,* for example, according to the degree of trust to the source of information, importance of information source for reasoner, etc.;

- dynamic re-calculation of decision making rules utility function could be done based on decisions' consequences (cost, required resources, received benefits, etc.).

7.7 Context reasoning model

In the beginning of this section we agreed that a context consumer behaves as follows:

- The consumer receives content information from some information source.
- The context consumer is able to make decisions based on content information only.
- The context consumer is able to make more suitable (efficient, appropriate) decisions if more (context) info is available.

The decision flow of a reasoner within the context consumer is shown in Figure 46 below. Based on the received context information the consumer makes all possible derivations and can see which decisions are already available and which will require more info.

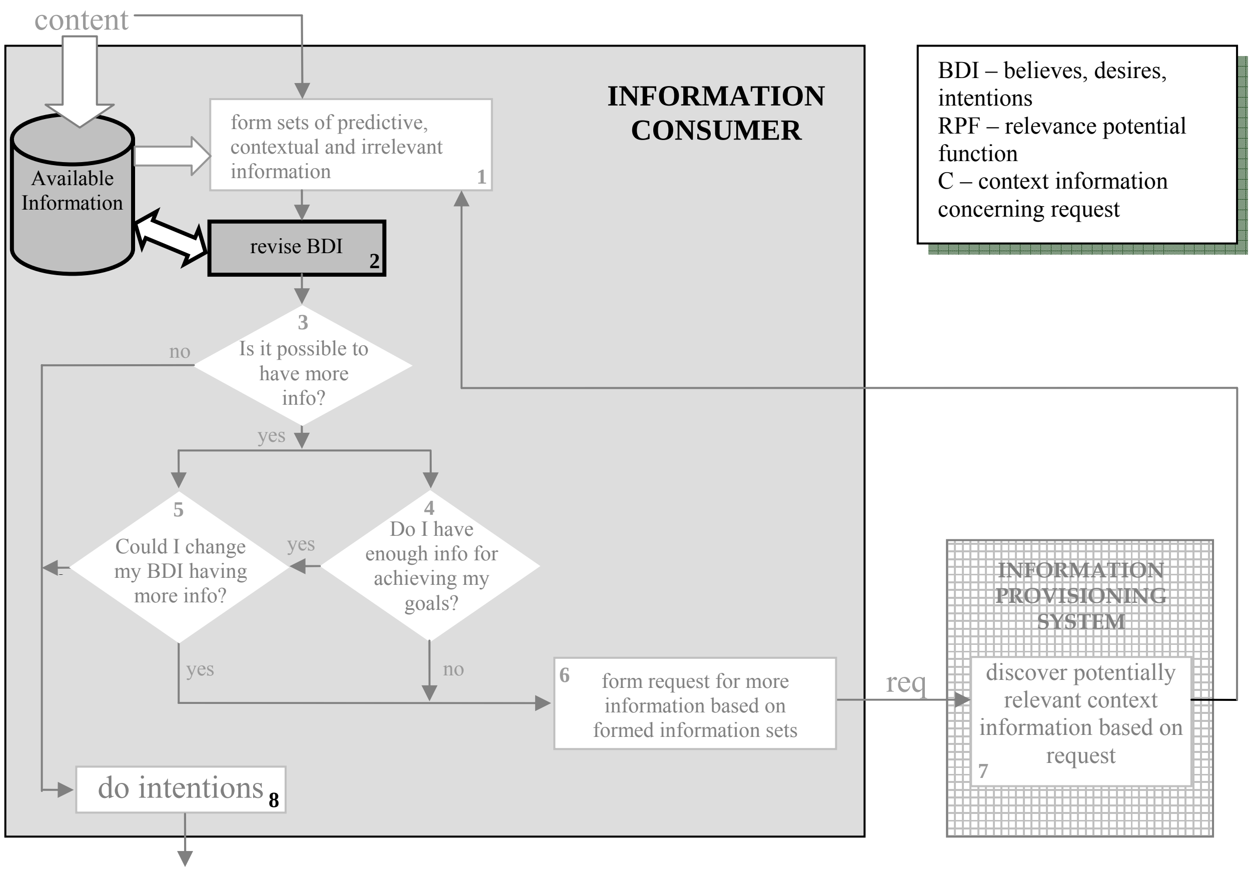

content
INFORMATION CONSUMER
Available Information
form sets of predictive, contextual and irrelevant information 1
revise BDI 2
3 Is it possible to have more info?
no
yes
5 Could I change my BDI having more info?
yes
4 Do I have enough info for achieving my goals?
yes
no
6 form request for more information based on formed information sets
req
INFORMATION PROVISIONING SYSTEM
discover potentially relevant context information based on request 7
do intentions 8
BDI – believes, desires, intentions
RPF – relevance potential function
C – context information concerning request

FIGURE 46 Decision flow of the context reasoner within a context consumer

Context reasoning itself described in this work is, basically, nothing new – based on a set of logical rules and incoming information set, context derivation and decision making possibility evaluation is performed. The decision making model used for reasoning is depicted in Figure 47 below.

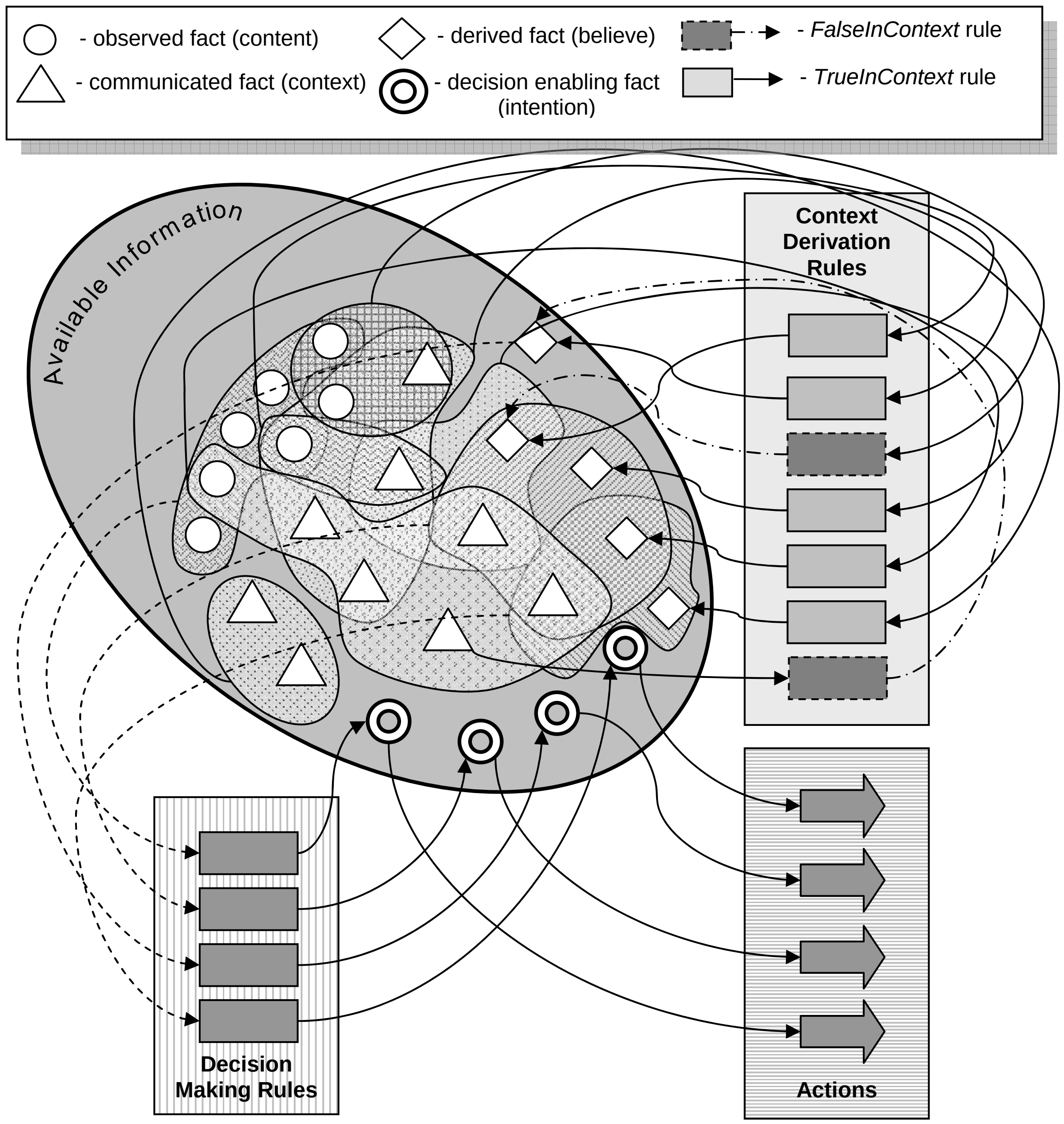


FIGURE 47 Decision making model

There are two sets of rules, which are similar by their structure, but different by their purpose – derivation rules and decision enabling rules. Both types of rules can be positive (true_in_context) - setting the corresponding fact to “true” - and negative (false_in_context) – the corresponding fact is set to “false”. All facts are stored within one information storage and can have mutual influence to each other. Each rule might be assigned with a rule utility weight, which is used for conflict resolutions and prioritization of decisions made. Let us use the following example to show how information (rules and facts) is organized within a context reasoner.

Example 2

There is a smart service application, which performs some multimedia contents delivery to mobile user. Initially, it is able to transmit information through a given channel (channel 1), for which it is able to observe the following set of necessary information (content): availability of the channel, allocation success, availability of data receiver, response of the receiver, own readiness to send the data.

TABLE 5 Set of facts for example 2

	Description	*Type*
f1	channel 1 available	observed
f2	transmission data ready	observed
f3	channel 1 allocated	observed
f4	response from receiver received via channel 1	observed
f5	channel 1 sender ready	observed
f6	receiver available	observed
f7	waiting time more then 20 sec	observed
f8	transmission through channel 2 extremely slow	communicated
f9	transmission through channel 1 extremely slow	communicated
f10	transmission permission through channel 2 received	communicated
f11	response from receiver received via channel 2	communicated
f12	channel 2 allocated	communicated
f13	channel 2 available	communicated
f14	network is too slow	derived
f15	channel 2 ok	derived
f16	transmission through channel 2 is fast	communicated
f17	channel 2 sender ready	communicated
f18	want to transmit through channel 2	derived
f19	ready to transmit through channel 2	derived
f20	channel 2 is not going to be busy within 25 sec	communicated
f21	receiver and channel 2 are ready	derived
f22	transmission through channel 1	decision enabling
f23	transmission through channel 2	decision enabling
f24	transmission not allowed	decision enabling
f25	wait 5 sec	decision enabling

After some time of functioning it receives, from a trusted supervising agent, information that there is one more transmission channel in the system, a channel that is faster and more preferable to use even if channel 1 is available and especially when it is not. In addition the application receives a small adapter (channel 2 sender), which actually performs the transmission through channel 2. The supervising agent has provided the smart service with some set of rules, describing the circumstances in which data can be transmitted through channel 2. and some of the rules service is communicated from other agents working in the same environment.

Therefore, the smart service has a set of slightly redundant rules, which is consistent (rules consistency is checked when a new rule is received from an external source). The Table 5 above contains a set of facts is utilized by the service's derivation rules. The set of derivation rules available for the smart service is presented in the Table 6.

TABLE 6 Set of rules for example 2

	Rule	*Resulting Fact*	*Context Set*	*Type*	*U*
R1	true_in_context	f22	f1, f2, f3, f4, f5, f6	decision enabling	0.7
R2	true_in_context	f23	f16,f15,f18,f19,f20,f21	decision enabling	0.9
R3	true_in_context	f24	f14	decision enabling	0.6
R4	true_in_context	f25	f19, f20, f21	decision enabling	0.65
R5	true_in_context	f15	f6, f9, f10, f13	context derivation	0.5
R6	true_in_context	f14	f6, f7, f8, f9, f10, f13	context derivation	0.5
R7	false_in_context	f15	f5, f7, f8	context derivation	0.4
R8	true_in_context	f18	f9, f15, f16	context derivation	0.5
R9	true_in_context	f19	f2, f10, f13, f16, f17, f20	context derivation	0.5
R10	true_in_context	f21	f11, f12	context derivation	0.5
R11	false_in_context	f14	f17	context derivation	0.5

7.8 Context analysis

The information in the context consumer is stored and processed uniformly – all information is represented as a set of context facts, which, in fact, is a semantic network with incomplete connectivity. Derivation and decision making processes are represented by logical rules, which use context facts as an input.

Context analyzer is intended for analysis of available facts and prediction of decision making possibility. Decision making possibility needs to be predicted in order to clarify which facts should be requested from the information system (or some other source) at each iteration of interactive reasoning in order to perform smart and effective decision making. The decision flow of context analyzer inside a context consumer is shown in Figure 48 below.

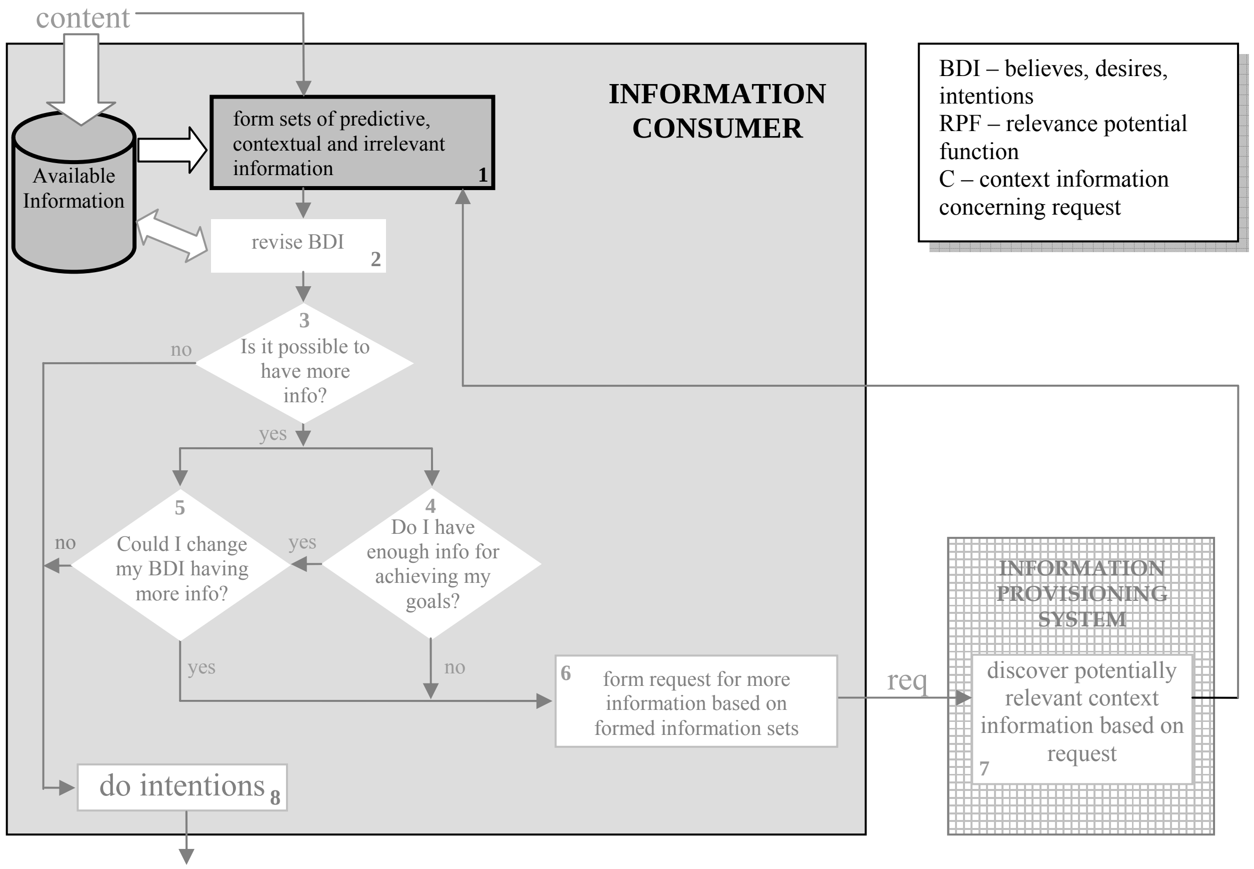

content
INFORMATION CONSUMER
Available Information
form sets of predictive, contextual and irrelevant information 1
revise BDI 2
3 Is it possible to have more info?
no
yes
5 Could I change my BDI having more info?
4 Do I have enough info for achieving my goals?
no
yes
yes
no
6 form request for more information based on formed information sets
req
INFORMATION PROVISIONING SYSTEM
discover potentially relevant context information based on request 7
do intentions 8
BDI – believes, desires, intentions
RPF – relevance potential function
C – context information concerning request

FIGURE 48 Decision flow of the context analyzer within a context consumer

Context analyses are used for optimization of interactive reasoning, which has the following aims:

- request the smallest possible amount of information,
- take the best possible decisions (according to the decision enabling rules' utility weight),

– clarify situation with the smallest possible number of information exchange iterations (between information source(s) and context consumer).

To reach these aims, it is important to know which facts are preferable to know earlier then others in order to make better decisions within shortest possible time. For this, the simplified knowledge model described below is suggested to be used.

The main initial assumption is that context derivation rules and decision making rules do not contain the operator "or" – all facts and their negations are connected by the logical operator "and". However, there could be more than one rule producing the same fact (context or decision enabling). In that case it is quite easy to transform rule with "or" to several "and"-based rules:

"channel A is available" *and* "receiver is available" *or*
"channel B is available" *and* "receiver is available"=>
"transmission allowed"

This logical rule can be broken into two decision enabling rules:

R1: true_in_context("transmission allowed"; "channel A is available", "receiver is available")
R2: true_in_context(""transmission allowed"; "channel B is available", "receiver is available")

Negations can be applied to one fact or any set of facts, for example (both are possible):

true_in_context ("transmission allowed"; "channel B is available", ¬"receiver unavailable")

or

true_in_context ("transmission not allowed"; ¬("channel B is available", "receiver is available"))

Let us simplify the previously described context derivation model as follows. Each decision enabling rule containing derived facts should be substituted by a set of extended rules containing only observed and communicated facts. This is done via substituting derived facts by a set of context facts used for their derivation. For example:

R1: true_in_context (q; a, b, c)
R2: true_in_context (t; a, b)
R3: true_in_context (t; g, h)
R4: true_in_context (w; q, t, s)

R4 can be substituted by two extended rules:

RE41: true_in_context (w; a, b, c, s)
RE42: true_in_context (w; a, b, c, g, h)

So, basically, the set of extended rules represents all possible combinations of direct (observed or communicated) facts, where satisfaction of any extended rule leads to making decision.

Let us consider now negative rules "false_in_context" and how they could be transformed into rules with non-derived facts, using the following example.

R1: true_in_context (q; a, b, c)
R2: false_in_context (q; t, r)
R3: true_in_context (w; q, f, d)
R4: true_in_context (w; o, p, z)

Satisfaction of the facts *a, b, c, f*, and *d* is a necessary and sufficient condition for *w* being true. Satisfaction of *t* and *r* is sufficient for q to be false and thus for R3 not to be true. However, according to the rule R4 satisfaction of *o, p* and *z* is enough (sufficient condition) for *w* to be true. This means that we can not substitute rule R3 by the rule: false_in_context (*w; ¬(t, r)*) because it would not be neither a necessary nor sufficient condition for *w* to be false and would become conflicting with rule R4. We also can not substitute R3 by the rule: true_in_context (*w; a, b, c, f, d, ¬(t, r)*), because in this case *¬(t, r)* would become a necessary condition for w to be true although the initial set of rules does not require implicit knowledge about *t* and *r*. So, if t and r are defined, the rule true_in_context (*w; a, b, c, f, d, ¬(t, r)*) works well, however, if they are not defined, such rule would require a clarification of their values.

To avoid unnecessary complication of context analysis process, let us use only positive rules for context analysis and let them affect the result during decision making process. Based on the example 2 presented in Sub-section 7.7 let us build the set of extended rules (see Table 7).

TABLE 7 Set of extended rules for example 2

	Rule	*Decision Enabling Fact*	*Context Set*	*Obtained as follows*
RE1	true_in_context	f22	f1, f2, f3, f4, f5, f6	
RE2	true_in_context	f23	f2, f6, f9, f10, f11, f12, f13, f16, f17, f20	f15 substituted using R5, f18 substituted using R8, f19 substituted using R9, f21 substituted using R10
RE3	true_in_context	f24	f6, f7, f8, f9, f10, f13	f14 substituted using R6
RE4	true_in_context	f25	f2, f10, f11, f12, f13, f16, f17, f20	f19 substituted using R9, f21 substituted using R10

Let us observe, what are the decision enabling situations described in Table 6.

Decision enabling fact: Transmission through channel 1
Context set:

f1 – “channel 1 available”
f2 – “transmission data ready”
f3 – “channel 1 allocated”
f4 – “response from receiver received via channel 1”
f5 – “channel 1 sender ready”
f6 – “receiver available”

Decision enabling fact: Transmission through channel 2
Context set:

f2 – “transmission data ready”
f6 – “receiver available”
f9 – “transmission through channel 1 extremely slow”
f10 – “transmission permission through channel 2 received”
f11 – “response from receiver received via channel 2”
f12 – “channel 2 allocated”
f13 – “channel 2 available”
f16 – “transmission through channel 2 is fast”
f17 – “channel 2 sender ready”
f20 – “channel 2 is not going to be busy within 25 sec”

Decision enabling fact: Transmission not allowed
Context set:

f6 – “receiver available”
f7 – “waiting time more then 20 sec”
f8 – “transmission through channel 2 extremely slow”
f9 – “transmission through channel 1 extremely slow”
f10 – “transmission permission through channel 2 received”
f13 – “channel 2 available”

Decision enabling fact: Wait 5 sec
Context set:

f2 – “transmission data ready”
f10 – “transmission permission through channel 2 received”
f11 – “response from receiver received via channel 2”
f12 – “channel 2 allocated”
f13 – “channel 2 available”
f16 – “transmission through channel 2 is fast”
f17 – “channel 2 sender ready”
f20 – “channel 2 is not going to be busy within 25 sec”

The only observed facts (content) are initially available and the goal of the context analyzer is to evaluate the significance of each unknown fact in order to decide which facts to request from the information source (context provisioning system).

For this purpose we introduce the following formula. The significance *S* of the fact *i*:

$$S(f_i) = \sum_{j=1}^{D} \frac{ad_{ij} \cdot U'_j \cdot e_i}{\mathrm{Re}_j}, \tag{7.1}$$

where D = number of possible decisions; f_i = fact *i; ad_{ij}* = affection power of the fact *i* to the decision *j*; U'_j = normalized rule utility weight; e_i = the number of entries of the fact *i* in the decision *j* enabling extended set; Re_j = the number of extended rules enabling the decision j.

Affection power of fact *i* on decision *j* is, basically, the probability of making decision *j* (or rejecting the correspondent rule if the fact is known to be "false") knowing fact *i*. It is calculated in the following way:

$$ad_{ij} = \max_{k=1}^{\mathrm{Re}_j}\left(\frac{1}{nf_k}\right), \tag{7.2}$$

where nf_k = number of unknown facts constituting the extended rule *k* enabling decision *j*.

Only valid rules (with U>0 are taken into account). It is very important to note here that only unknown facts are considered. Such approach significantly increases the probability of making a decision with a concrete fact, when all (or many) other facts in the rule are known.

Normalized utility weight is calculated based on the initial (not the extended) set of rules as follows:

$$U'_z = \frac{U_z}{\sum_{k=1}^{D} U_k},$$

where *D* is the number of possible decisions; z = the index of a certain decision.

All decision enabling rules in the extended set have the same U′ values as the initial rules have (even though there would be several extended rules corresponding to the same normal/initial decision enabling rule, each of them would be assigned the same normalized utility value that the initial rule has). In Sub-section 6.2, which describes context derivation approach, it is mentioned that if the rule is proved not to be satisfied (one of the constituting facts is known to be false), the utility of this rule becomes 0. This is the case also for the extended rules set and means that useless rules are not taken into account while calculating significance values for the facts.

Number of entries in the decision *j* enabling set indicates the number of decision *j* enabling rules in which fact *I* is participating.

Let us calculate the significance values for each fact used in derivation rules and decision enabling rules in the example with smart service (results are given in Table 8).

Normalized utility weights for extended decision enabling rules in this case are:

Rule	U'
RE1	0,2456
RE2	0,3158
RE3	0,2105
RE4	0,2281

TABLE 8 Fact significance calculation

	RE1			RE2			RE3			RE4			S_i
	ad_{i1}	e_{i1}	S_{i1}	ad_{i2}	e_{i2}	S_{i2}	ad_{i3}	e_{i3}	S_{i3}	ad_{i4}	e_{i4}	S_{i4}	
f1	0.1667	1	0.0409	0	0	0	0	0	0	0	0	0	0.0409
f2	0.1667	1	0.0409	0.1	1	0.0316	0	0	0	0.125	1	0.0285	0.1010
f3	0.1667	1	0.0409	0	0	0	0	0	0	0	0	0	0.0409
f4	0.1667	1	0.0409	0	0	0	0	0	0	0	0	0	0.0409
f5	0.1667	1	0.0409	0	0	0	0	0	0	0	0	0	0.0409
f6	0.1667	1	0.0409	0.1	1	0.0316	0.1667	1	0.0351	0	0	0	0.1076
f7	0	0	0	0	0	0	0.1667	1	0.0351	0	0	0	0.0351
f8	0	0	0	0	0	0	0.1667	1	0.0351	0	0	0	0.0351
f9	0	0	0	0.1	1	0.0316	0.1667	1	0.0351	0	0	0	0.0667
f10	0	0	0	0.1	1	0.0316	0.1667	1	0.0351	0.125	1	0.0285	0.0952
f11	0	0	0	0.1	1	0.0316	0	0	0	0.125	1	0.0285	0.0601
f12	0	0	0	0.1	1	0.0316	0	0	0	0.125	1	0.0285	0.0601
f13	0	0	0	0.1	1	0.0316	0.1667	1	0.0351	0.125	1	0.0285	0.0952
f14	0	0	0	0	0	0	0	0	0	0	0	0	0
f15	0	0	0	0	0	0	0	0	0	0	0	0	0
f16	0	0	0	0.1	1	0.0316	0	0	0	0.125	1	0.0285	0.0601
f17	0	0	0	0.1	1	0.0316	0	0	0	0.125	1	0.0285	0.0601
f18	0	0	0	0	0	0	0	0	0	0	0	0	0
f19	0	0	0	0	0	0	0	0	0	0	0	0	0
f20	0	0	0	0.1	1	0.0316	0	0	0	0.125	1	0.0285	0.0601
f21	0	0	0	0	0	0	0	0	0	0	0	0	0
f22	0	0	0	0	0	0	0	0	0	0	0	0	0
f23	0	0	0	0	0	0	0	0	0	0	0	0	0
f24	0	0	0	0	0	0	0	0	0	0	0	0	0
f25	0	0	0	0	0	0	0	0	0	0	0	0	0

Since we have only 4 extended decision making rules (the model is quite simple), according to the calculated significance the facts can be divided into 6 groups, presented in Table 9.

TABLE 9 Summary table of facts significance calculation results

S	*Facts*
0.1076	f6
0.1010	f2
0.0952	f10, f13
0.0667	f9
0.0601	f11, f12, f16, f17, f20
0.0409	f1, f3, f4, f5
0.0351	f7, f8

Analyzing the obtained results one can see that the values are distributed mainly according to the number of decisions in which the concrete fact is participating. However, if the facts have the same participation rate, those facts that participate in decision enabling rules with a lesser number of other participating facts “win”. The rule utility weight has also impact on the significance value. For example, facts f6 and f2 participate in extended rules 3 times each, however, f6 is better to know first, since f3 is more probable to be taken than f4, which requires many more conditions to be satisfied and, consequently – more information to be known. The utility rates of RE3 and RE4 are not much different, that is why they do not have a significant impact in this case. Facts f10 and f13 take the next place, they have a participation rate of 3 and their significance value could be the same as for f2; f2 participates in RE1, RE2, RE4; f10, and f13 in RE2, RE3 and RE4. RE1 is preferable to RE3 according to its utility weight, that is why f10 and f13 have lower significance value then f2.

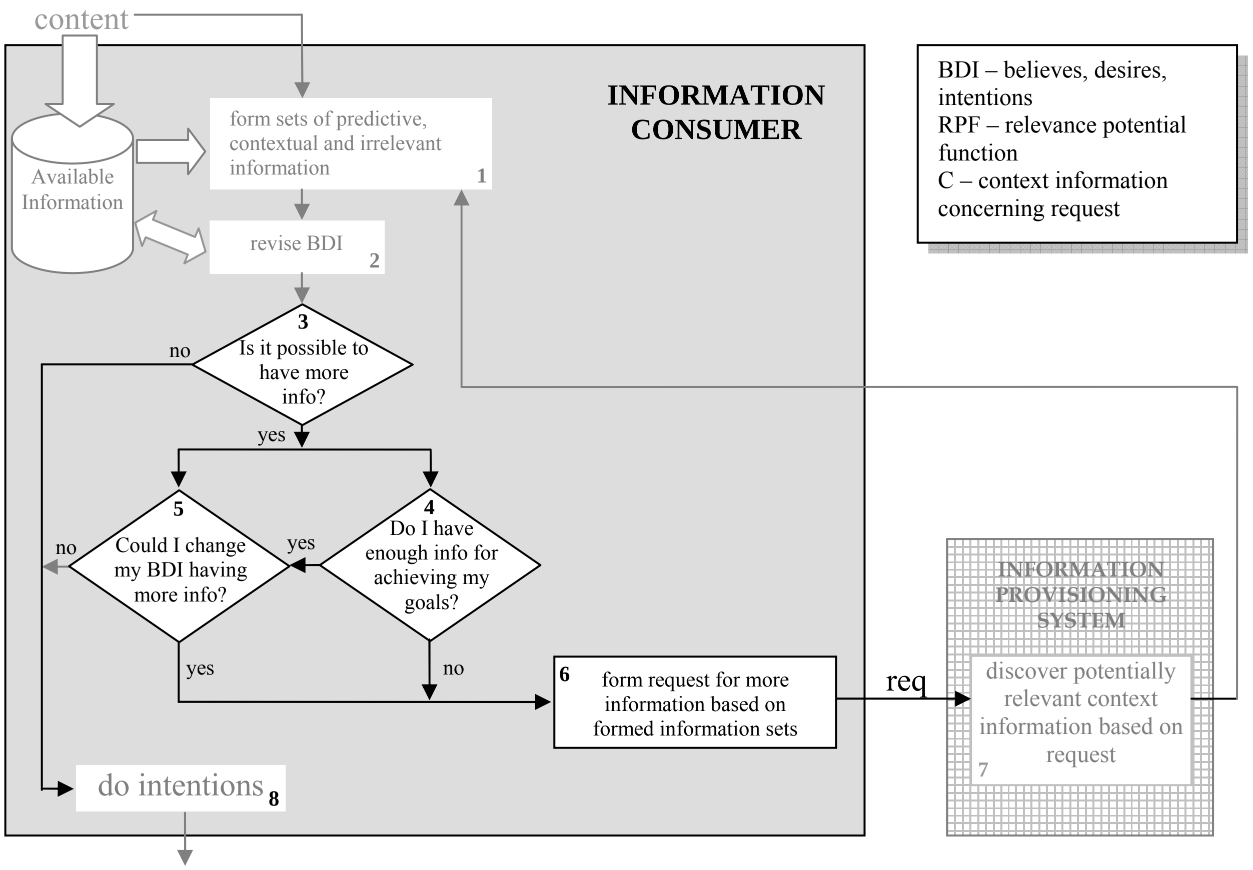
7.9 Forming context request

In the previous section a very basic example of context reasoner is presented. It is described mainly with the purpose of showing how context information is supposed to affect actual actions of the context consumer, introducing the concept of rule utility weight and demonstrating how the context fact significance could be calculated. Having calculated the significance values for different facts, the context consumer should ask the system for those context facts which are most significant for it at the moment of forming the request. Interaction of the reasoner with the context provisioning system would consist in asking for different facts from the system and receiving short responses with the requested facts if they exist in the system and their first-order context. Development of more intelligent reasoner is left out from the scope of this study since this thesis concentrates mainly on context modeling and its utilization aspect, and not on context-aware reasoning.

The purpose of this work is to find an approach for context processing that would efficiently serve both small "stupid" clients and powerful "smart" agents, while making their life easier by providing them with only a minimum amount of as relevant as possible context information.

Since the described reasoner, due to its primitiveness, is not able to form context requests that would utilize the whole power of the presented context provisioning approach, let's consider a more general case where we do not know the way the request is formed by the reasoner, and let's define the kinds of values the context consumer is able to give to the system in the context request.

Figure 49 below shows the decision flow of a context request for a context consumer within an interactive reasoning architecture.



content
INFORMATION CONSUMER
Available Information
form sets of predictive, contextual and irrelevant information 1
revise BDI 2
3 Is it possible to have more info?
no
yes
5 Could I change my BDI having more info?
4 Do I have enough info for achieving my goals?
no
yes
yes
no
6 form request for more information based on formed information sets
req
INFORMATION PROVISIONING SYSTEM
discover potentially relevant context information based on request 7
do intentions 8
BDI – believes, desires, intentions
RPF – relevance potential function
C – context information concerning request

FIGURE 49 Forming a context request

This work is conceptual rather than practical;, therefore, it does not provide any real application of the presented concepts. This sub-section is intended to provide a description of a context request given by the context consumer to the context provisioning system. Due to the conceptuality level of this work, we are not going to see here any ready-for-use context exchange protocol specification. CEP [49] could possibly be adapted for the purposes of context request and response exchange with respect to the concepts presented. Maybe some new protocol needs to be developed. The main idea is that the protocol should be "understandable" by different entities acting in a heterogeneous environment with mobile components. The general format of context request would be the following (spelling is given in Table 10):

([FUNCTION]; FOCUS; [CCD]; [CCD DIR]; [DP]; [DCCD]; [DCCD DIR])

Optional parameters are marked by [ ], and if not defined, default values are used. The Table 10 describes the meaning of each parameter.

TABLE 10 Context request parameters

Parameter	*Set of possible values*	*Default value*	*Description*
FUNCTION	BRPF, CRPF	BRPF	Function to be used for context extraction – basic relevance potential function or cumulative RPF
FOCUS	entity, triple	-	Literal description or reference to an entity in a semantic network; triple
CCD	natural number	1	Context consideration depth for horizontal search
CCD DIR	- or +	+	Direction of context search, + context, - decontext
DP	property	-	Literal or a reference to an existing property
DCCD	natural number	0	Dimensional depth for context search along the dimension defined by DP
DCCD DIR	- or +	+	Direction of dimensional context search, + context, - decontext

7.10 Conclusions

This section considers processes taking place in context consumer – context reasoning and context facts analysis. The reasoning model is enhanced with rule utility weights, used for conflicts resolution and decisions prioritization.

Fact significance value, calculation of which is introduced in this section allows efficient context information requesting from the information source (context provisioning system). Those facts which have more effect on effective decision making receive greater significance value and should be requested from the information source first. In fact, context fact significance value reflects the relevance of the context information to decision making of the context consumer – the more relevant the fact is to the consumer's BDI the greater the significance value it will receive after calculations.

The reasoning mechanism and the context fact analysis that are presented together in this section are not considered to be significant research contributions; however, they are intended to show the reader the processes taking place within a context-aware application and to serve as an example of their possible implementation.

As it is shown in the description of CAME, a context-aware application can be implemented in any way and use any kind of reasoning technique or operate on just a planar data. The only requirement is that it should be able to specify its needs in a way that would enable the context provisioning system to provide it with the needed context. The main attribute, giving the information system some "hints" of what kind of context the application would need, is a focus of context discovery – the target attribute, or object of interest of the application (can also be a triple, set of objects).

The general form of context request to the provisioning system is presented in this section. It shows the kinds of possibilities the context-aware application has for presenting to the system the scope of one's context needs. The context request presented utilizes the theoretical foundations developed within Section 6. A general conclusion to this game between the system and the application would be the following: the more knowledge the application has about the information available in the system, the more precise the context request it can give to it; on the other hand, the more specific the request delivered to the context-provisioning system, the higher the probability of providing the application with really relevant context.

8 CONCLUSIONS AND FUTURE WORK

Context-awareness capability is one of the core enablers of future networking. Understanding of context as a concept, its proper modeling and processing is the most important of the unsolved problems in moving towards ambient awareness. The summary of this work's achievements contributing into solving of this problem is presented in 8.1. Sub-section 8.2 depicts directions for future work.

8.1 Summary

The survey of literature devoted to context in Artificial Intelligence [14] raises the questions: "Does context belong to the knowledge base or to a particular context base? What are the relationships between context and meta-knowledge, context and knowledge representation, context and time, context and decision? What are the relationships between contextualization process and control knowledge?" This work has tried to answer some of these questions.

A general idea of context-aware mobile environment is introduced in the beginning of the work. We consider CAME be potentially capable of providing ambient awareness for modern communication networks. Theoretical contributions presented later in the dissertation are supposed to be used in different functional parts of such environment.

After the discussion about the nature of context, some properties of context information such as dynamicity, subsidiarity, expandability and relevance are defined; it was found that context information inherits most of the general information properties, including diversity, quality, completeness, correlativity, actuality, and structure. The core issues discussed in this thesis are context consideration depth, relevance potential function and fact significance value – all them reflect contextual relevance of the information in a particular situation of a particular context consumer.

The main theoretical contribution of the dissertation is in understanding of context as a relational property around particular subject of attention. A lot of research effort is put to the development of context-aware applications in different areas. However, there are no universal and flexible solutions which could be applied in different application areas without a global redesign of the whole system. This situation prevails because existing solutions are too task-oriented and there is no universal approach to provide context-awareness capability in modern computer networks. The notion of context as a concept is deeply discussed and a philosophical model of context is presented.

We introduced context relevance potential function, which is used by system for information search. Also the structure of user's request to the system and the principles of it's forming has been presented. Context search is based on the subject-oriented approach for context analysis and principle of floating focus introduced in the work. In addition to main technical contribution context as an abstract notion is discussed in depth.

Another important achievement of the work is introduction of interactive reasoning principle. Two entities are participating in the process – context information consumer (can be any kind of software) and contextual information provisioning system. Consumer is able to operate without interacting with system, however additional contextual information would positively affect its decisions and actions. System contains a lot of information about the domain, which is organized as RDF semantic network and is able to get more information from similar systems operating in another domains. Precondition is that context information consumer (it can be any kind of software) does not have predefined knowledge about the structure and content of the available information. On the other hand the system does not know beliefs, desires and intentions of the consumer in order to provide him exactly the information he needs. Due to huge volume of information in system the consumer is not able to process all available information in a reasonable time. Aim is that the system and the consumer are interactively searching for needed information. System is providing possibly relevant information to the consumer piece by piece; after each iteration based on received context, consumer specifies more precisely what kind of information would be interesting for him.

We consider interesting and promising contribution of this work definition of RDF-network dimensionality, which allows structured processing of extensive and somewhat chaotic RDF knowledge space. While we believe this work introduces views and techniques which would potentially bring together all separated areas of context-aware computing, much remains to be done. The following section presents directions of future work.

8.2 Future work

Context-aware computing is a developing research area, which still contains many research challenges. We believe that the main sticking point of it is the absence of common understanding of context's nature, which we have addressed in this work. However, there is still quite a lot to do on the way towards ambient awareness.

Particularly, it would be interesting for us to study time component of context, which we have almost omitted in this work. Quite a lot of new knowledge can be obtained analyzing historical changes of particular contexts and level of their dynamics. In addition to time issue, it would be interesting to study Quality of Context, which is just briefly touched in this work within Sub-section 4.5.

Another important question is security. In this work we assume that there is quite a lot of data and knowledge available for different context consumers. How to handle security in such extensive information space? It is obvious that all information should not be available to all interested parties. On the other hand, current security techniques are, most likely, not oriented towards such globally aware environments.

We stated already that from our point of view semantic networks have great potential to be used as a basic information model in global knowledge space. Not in their current form, though. We strongly believe that there is a need of efficient, distributed, fast databases, which would store, retrieve and provide semantically enriched dynamic information. It seems most unlikely that modern relational and object-oriented databases would suit for this purpose.

As a future research direction of this dissertation we can also point out the need of practical implementation and validation of the achieved results.

BIBLIOGRAPHY

[1] 3GPP - 3rd Generation Partnership Project, http://www.3gpp.org (visited 30.11.2006).

[2] 3GPP2 - 3rd Generation Partnership Project 2, http://www.3gpp2.org (visited 01.12.2006).

[3] Agostini, A., Bettini, C. & Riboni, D. 2006. Experience Report: Ontological reasoning for Context-aware Internet Services, In Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops: IEEE Computer Society, 8-12.

[4] Algae RDF Query Language, http://www.w3.org/2004/05/06-Algae (visited 31.10.2006).

[5] Ambient Networks ContextWare 2005. Deliverable 6.1 Mobile and Wireless Systems beyond 3G. Project 507134 WWI Ambient Networks, submission date 17.01.2005, available at http://www.ambient-networks.org/phase1web/publications/D6-1_PU.pdf (visited 01.12.2006).

[6] Ambient Networks Project, www.ambient-networks.org (visited 12.05.2006).

[7] AT&T Laboratories Cambridge, http://www.cl.cam.ac.uk/research/dtg/attarchive (visited 02.12.2006).

[8] Bargrodia, R., Chu, W. W. & Kleinrock, L. 1995. Vision, Issues, and Architecture for Nomadic Computing. IEEE Personal Communications 2(6), 14-27.

[9] Barrett, K. & Power, R. 2003. State of the Art: Context Management, M-Zones Project Deliverable 1.1, available at http://www.m-zones.org/deliverables/d1_1/d1_1.php4 (visited 01.12.2006).

[10] Beigl, M., Zimmer, T. & Decker, C. 2002. A location model for communicating and processing context. Personal and Ubiquitous Computing 6(5-6), 341-357.

[11] Benerecetti, M., Bouquet, P. & Chidini, C. 2000. Contextual reasoning distilled. Journal of Experimental & Theoretical Artificial Intelligence 12(3), 279-305.

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

[13] Brasche, G. & Walke, B. 1997. Concepts, services, and protocols of the new GSM Phase 2+ General Packet Radio Service. IEEE Communications Magazine 35(8), 94-104.

[14] Brezillon, P. 1999. Context in Artificial Intelligence: I.A. Survey of the literature. Computer & Artificial Intelligence 18(4), 321-340.

[15] Buchholz, T., Kupper, A. & Schiffers, M. 2003. Quality of context: What it is and why we need it. In Proceedings of the Workshop of the HP OpenView University Association (HPOVUA 2003), Geneva.

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

[17] Chen, G. & Kotz, D. 2000. A survey of context-aware mobile computing research. Technical Report TR2000-381. Hanover: Dartmouth College.

[18] Chen, H. & Finin, T. 2003. An ontology for context aware pervasive computing environments. The Knowledge Engineering Review 18(3), 197 - 207.

[19] Choi, J., Shin Dongkyoo & Shin Dongil 2005. Research and Implementation of the Context-Aware Middleware for Controlling Home Appliances. IEEE Transactions on Consumer Electronics 51(1), 301 - 306.

[20] Conti, M., Maselli, G., Turi, G. & Giordano, S. 2004. Cross-layering in mobile ad hoc network design. IEEE Computer 37(2), 48-51.

[21] Davidyuk, O., Riekki, J., Rautio, V.-M. & Sun, J. 2004. Context-aware middleware for mobile multimedia applications. In Proceedings of the 3rd international Conference on Mobile and Ubiquitous Multimedia. ACM International Conference Proceeding Series, Vol. 83. New York: ACM Press, 213-220.

[22] Dehainsala, H., Pierra, G. & Bellatreche, L. 2006. Managing instance data in ontology-based databases. Research Report 3. Laboratory of Applied Computer Science, France, available at ttp://www.lisi.ensma.fr/ftp/pub/documents/reports/2006/2006-LISI-003 DEHAINSALA.pdf (visited 02.12.2006).

[23] Dey, A.K. 2001. Understanding and using context. Personal and Ubiquitous Computing 5 (1), 4-7.

[24] Dourish, P. 2004. What we talk about when we talk about context. Personal and Ubiquitous Computing 8(1), 19-30.

[25] Edmonds B. 1997. A simple-minded network model with context-like objects. Proceedings of the 2nd European Conference on Cognitive Science, Manchester, UK, 181-184.

[26] 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).

[27] Flying Carpet. Towards the 4th Generation Mobile Communications System. Version 2.0. Mobile IT Forum (MITF), 4th Generation Mobile Communications Committee, available at http://www.mitf.org/public_e/archives/index.html (visited 02.12.2006).

[28] Georgia Institute of Technology, http://www.gatech.edu/ (visited 02.12.2006).

[29] Going beyond 3G. 2005. Nortel technical Journal, Issue 2 (Sept 2005), 1-47.
[30] Gross, T. & Prinz, W. 2004. Modelling shared contexts in cooperative environments: concept, implementation, and evaluation. Computer Supported Cooperative Work 13 (3-4), 283-303.
[31] Gu, T., Pung, H. K. & Zhang, D. Q. 2005. A service-oriented middleware for building context-aware services. Journal of Network and Computer Applications 28 (1), 1-18.
[32] Gu, T., Wang, X. H., Pung, H.K. & Zhang, D.Q. 2004. An ontology-based context model in intelligent environments. In Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference, San Diego, California, USA, 270-275.
[33] Hein, J.L. 2002. Discrete mathematics. (2nd Edition) Portland State University: Jones and Bartlett Publishers.
[34] Held, A., Buchholz, S. and Schill, A. 2002. Modeling of context information for pervasive computing applications. In Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics, Orlando, FL, USA.
[35] Henricksen, K., Indulska, J. & Rakotonirainy A. 2002, Modeling context information in pervasive computing systems. In Proceedings of the First international Conference on Pervasive Computing. F. Mattern and M. Naghshineh (Eds.) Lecture Notes In Computer Science, Vol. 2414. London: Springer-Verlag, 167-180.
[36] Huebscher, M. C. & McCann, J.A. 2004. Adaptive middleware for context-aware applications in smart-homes. In Proceedings of the 2nd Workshop on Middleware For Pervasive and Ad-Hoc Computing. ACM International Conference Proceeding Series, Vol. 77. New York: ACM Press, 111-116.
[37] IEEE - The Institute of Electrical and Electronics Engineers, http://www.ieee802.org (visited 01.12.2006).
[38] IETF - The Internet Engineering Task Force, http://www.ietf.org (visited 01.12.2006).
[39] Intelligent Networks, http://www.mobilein.com/intelligent_networks.htm (visited 01.12.2006).
[40] Internet hyperdictionary, http://www.hyperdictionary.com/dictionary/context (visited 18.10.2004).
[41] Internet Protocol. Version 6 (IPv6) Specification. December 1998, available at http://tools.ietf.org/html/rfc2460 (visited 01.12.2006).
[42] ITU - International Telecommunication Union http://www.itu.int/home/imt.html (visited 01.12.2006).
[43] Johnson, D., Perkins, C. & Arkko J. 2004. Mobility support in IPv6. RFC 3775, available at http://www.ietf.org/rfc/rfc3775.txt (visited 02.12.2006).
[44] Kang, Z. & Wang, H. 2005. Implementation and application of ontology databases with user-defined rules (UDR) supported. In Proceedings of The

First International Conference on Semantics, Knowledge and Grid. IEEE Computer Society, p.82

[45] Kaykova, H., Terzian, V. & Omelayenko, B. 2000. Recognizing bounds of context change in on-line learning. In M. Khosrowpour (Ed.) Challenges of Information Technology Management in the 21st Century. Information Resource Management Association International Conference, Anchorage: Idea Group Publishing, 236 – 239.

[46] Kaykova, O., Khriyenko, O., Naumenko, A., Terziyan, V. & Zharko A. 2005. RSCDF: A dynamic and context-sensitive metadata description framework for industrial resources. Eastern-European Journal of Enterprise Technologies 3(2), 55-78.

[47] Khriyenko, O. & Terziyan V. 2006. A framework for context-sensitive metadata description. International Journal of Metadata, Semantics and Ontologies 1(2), 154-164.

[48] Kohvakko, N. 2004. Characterization of services in heterogeneous communication environments. In Proceedings of the 1st International Workshop on Ubiquitous Computing. Porto: INSTICC Press, 3-10.

[49] Lakkala, H. 2003. Context Exchange Protocol specification. Version 1.0. Helsinki: Nokia, available at http://www.mupe.net (visited 19.03.2004).

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

[51] McCarthy, J. & Buvač, S. 1994. Formalizing context (Expanded Notes). Technical Note STAN-CS-TN-94-13. Stanford: Stanford University.

[52] MIT Media Lab, Cambridge, http://www.media.mit.edu/ (visited 02.12.2006).

[53] Mobile IT Forum, http://www.mitf.org/index_e.html (visited 01.12.2006).

[54] Nikitin, S., Terziyan, V., Tsaruk, Y. & Zharko, A. 2005. Querying dynamic and context-sensitive metadata in Semantic Web. In V. Gorodetsky, J. Liu, & V.A. Skormin (Eds.) Autonomous Intelligent Systems: Agents and Data Mining. Proceedings of the AIS-ADM-05. St. Petersburg: Springer, 200-214.

[55] Obrst, L. 2004. Ontologies and the Semantic Web: An overview. Presentation. MITRE, Center for Innovative Computing & Informatics. Presentation, July 13, 2004, available at
http://colab.cim3.net/file/work/Expedition_Workshop/2004-04-28_Multiple_Taxonomies/Obst_20040428.ppt (visited 04.12.2006).

[56] Open Mobile Alliance, http://www.openmobilealliance.org (visited 04.12.2006).

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

[58] Palo Alto Research Center, http://www.parc.xerox.com/ (visited 02.12.2006).

[59] Puuronen, S. & Terziyan, V. 1992. A Metasemantic Network. In E. Hyvonen, J. Seppanen & M. Syrjanen (eds.) SteP-92 - New Directions in Artificial Intelligence, Publication of the Finnish AI Society, Vol.1., Otaniemi, Finland, 136-143.

[60] Raatikainen, K. 2005. A new look at Mobile Computing. In Proceedings of International Workshop on Convergent Technologies (IWCT 2005), Oulu, Finland, available at http://www.cs.helsinki.fi/u/kraatika/Papers/RaatikainenIWCT2005.pdf (visited 04.12.2006).

[61] Ranganathan, A. & Campbell R.H. 2003. A middleware for context-aware agents in ubiquitous computing environments. In Proceedings of ACM/IFIP/USENIX International Middleware Conference, Rio de Janeiro, Brazil, 143-161.

[62] Ranganathan, A., Al-Muhtadi, J. & Campbell R.H. 2004. Reasoning about uncertain contexts in pervasive computing environments. IEEE Pervasive Computing 3 (2), 62-70.

[63] Ranhema, M. 1993. Overview of the GSM System and Protocol Architecture. IEEE Communications Magazine 31(4), 92-100.

[64] Raverdy, P.-G., Riva, O., Chaphelle, A., Chibout, R. & Issarny V. 2006. Efficient context-aware service discovery in multi-protocol pervasive environments. In Proceedings of 7th International Conference on Mobile Data Management (MDM'06), Washington: IEEE Computer Society.

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

[66] Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, 10 February 2004, available at http://www.w3.org/TR/rdf-concepts (visited 12.10.2005).

[67] Riva, O. & di Flora C. 2006. Contory: A smart phone middleware supporting multiple context provisioning strategies. In Proceedings of the 26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW'06), Washington: IEEE Computer Society, 68.

[68] Riva, O. & Toivonen, S. 2006. A hybrid model of context-aware service provisioning implemented on smart phones. In Proceedings of the 3rd IEEE International Conference on Pervasive Services 2006 (ICPS'06), Lyon, France, 47-56.

[69] Rouffet, D., Kerboeuf, S., Cai, L. & Capdevielle V. 2005. 4G MOBILE, Alcatel Telecommunications Review - 2nd Quarter 2005, available at http://www1.alcatel-lucent.com/publications/abstract.jhtml?repositoryItem=tcm%3A172-262211635 (visited 02.12.2006).

[70] Satyanarayanan, M. 2001. Pervasive Computing: Visions and challenges, IEEE Personal Communications 8(4), 10-17.

[71] Schilit, B., Adams, N. & Want R. 1994. Context-aware computing applications. In Proceedings of IEEE Workshop on Mobile Computing Systems and Applications, Santa Cruz, USA, 85-90.

[72] Schilit, W. A. 1995. System architecture for context-aware mobile computing. PhD thesis. New York: Columbia University.

[73] Setton, E., Yoo, T., Zhu, X., Goldsmith, A. & Girod B. 2005. Cross-layer design of ad hoc networks for real-time video streaming. IEEE Wireless Communications Magazine 12(4), Special Issue on Cross-Layer Protocol Engineering For Wireless Mobile Networks, 59-65.

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

[75] Sorensen C.-F. at al. 2004. A context-aware middleware for applications in mobile ad hoc environments. In Proceedings of the 2nd Workshop on Middleware for Pervasive and Ad-Hoc computing. ACM International Conference Proceeding Series, Vol. 77. New York: ACM Press, 107 – 110.

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

[77] 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).

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

[79] Specht, G. & Weithöner, T. 2006. Context-aware processing of ontologies in mobile environments. In Proceedings of the 7th International Conference on Mobile Data Management. Washington: IEEE Computer Society, 86.

[80] Strang T., Linnhoff-Popien C. & Frank K. 2003. CoOL: A context ontology language to enable contextual interoperability. In Proceedings of 4th IFIP WG 6.1 International Conference on Distributed Applications and Interoperable Systems, Paris, France, 236.

[81] Strang, T. & Linnhoff-Popien, C. 2004. A context modeling survey. In Proceedings of The Sixth International Conference on Ubiquitous Computing - Workshop on Advanced Context Modelling, Reasoning and Management, Nottingham, GB.

[82] Sun, J.-Z., Sauvola, J. & Howie, D. 2001. Features in future: 4G visions from a technical perspective. In Proceedings of Global Telecommunications Conference (GLOBECOM '01), Vol.6. San Antonio: IEEE Computer Society, 3533-3537.

[83] Tafazolli, R. (Ed.) 2006. Technologies for the Wireless Future. Wireless World Research Forum (WWRF), Vol. 2. Chichester: John Wiley & Sons.

[84] Tazari, M-R., Grimm, M. & Finke, M. 2003. Modelling user context. In J.A. Jacko and C. Stephanidis (Eds.) Proceedings of the 10th International Conference on Human-Computer Interaction, Vol.2. Theory and Practice (Part II) Mahwah: Erlbaum, 293-297.

[85] Terziyan, V. & Puuronen, S. 2000. Reasoning with multilevel contexts in semantic metanetworks. In P. Bonzon, M. Cavalcanti & R. Nossun (Eds.) Formal Aspects in Context - Applied Logic Series, Vol. 20. Kluwer Academic Publishers, 107-126.
[86] Terziyan, V. & Vitko O. 2003. Bayesian metanetwork for modelling user preferences in mobile environment. In A. Gunter, R. Kruse & B. Neumann (Eds.) Advances in Artificial Intelligence - Lecture Notes in Computer Science, Vol. 2821. Berlin: Springer-Verlag, 370-384.
[87] Terziyan, V. 2005. A bayesian metanetwork. International Journal on Artificial Intelligence Tools 14(3), 371-384.
[88] Terziyan, V. 2006. Bayesian metanetwork for context-sensitive feature relevance. In G. Antoniou et al. (Eds.) Proceedings of the 4-th Hellenic Conference on Artificial Intelligence, Lecture Notes in Computer Science, Vol. 3955. Heidelberg/Berlin: Springer, 356-366.
[89] Terziyan, V. 2006. Bayesian reasoning based on predictive and contextual feature selection. International Journal on Artificial Intelligence Tools, submitted 28 September 2006, available at http://www.cs.jyu.fi/ai/IJAIT-2006.doc (visited 02.12.2006).
[90] The Indus platform Developer site, http://live.aumeganetworks.com (visited 02.12.2006).
[91] The Rule Markup Initiative, http://www.ruleml.org (visited 31.10.2006).
[92] W3C – World Wide Web Consortium, http://www.w3.org/2001/sw, (visited 02.12.2006).
[93] Wang, X.H., Zhang, D.Q., Gu, T. & Pung, H.K. 2004. Ontology based context modeling and reasoning using OWL. In Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, Washington: IEEE Computer Society, 18.
[94] Weiser, M. 1991. The computer for the twenty-first century. Scientific American 265(3), 94-101.
[95] Woodridge, M. 2000. Reasoning about Rational Agents. Cambridge/London: The MIT Press.
[96] WordNet - a lexical database for the English language, http://wordnet.princeton.edu (visited 21.10.2005).
[97] WWRF – Wireless World Research Forum, http://www.wireless-world-research.org (visited 01.12.2006).
[98] XML Path Language (XPath). Version 1.0. W3C Recommendation, 16 November 1999, available at http://www.w3.org/TR/xpath (visited 02.12.2006).
[99] XQuery 1.0: An XML Query Language. W3C Proposed Recommendation, 21 November 2006, available at http://www.w3.org/TR/xquery (visited 02.12.2006).

[100] Xynogalas, S.A., Chantzara, M.K., Sygkouna, I.C., Vrontis, S.P., Roussaki, I.G. & Militades A.E. 2004. Context management for the provision of adaptive services to roaming users. IEEE Wireless Communications 11(2), 40-47.

[101] Zhovtobryukh D. & Kohvakko N. 2004. Service reference model for modern communications. In S.M.Furnell & P.S. Dowland (Eds.) Proceedings of the 4th International Network Conference, Plymouth: University of Plymouth, 221-228.

YHTEENVETO (FINNISH SUMMARY)

Tässä väitöskirjassa käsitellään kontekstien eli asiayhteyksien mallintamista ja käyttöä tulevaisuuden monimuotoisissa verkkoympäristöissä. Kontekstitietoinen matkapuhelin voisi esimerkiksi tietää, että se on tällä hetkellä kokoushuoneessa, ja puhelimen käyttäjä on istunut alas. Tällöin tietämällä ja yhdistämällä asiayhteyksiä puhelin voi estää ei-tärkeät puhelut pääteltyään käyttäjän olevan kokouksessa.

Kontekstitietoisuudesta ennustetaankin yhtä pääteknologiaa tulevaisuuden globaaleissa verkkoympäristöissä. Tämä tutkimustyö esittelee uusia kontekstin käsittelymalleja perustuen semanttisten verkkojen tietomalleihin.

Väitöskirjassa esitellään uusi interaktiivinen päättelymenetelmä, jonka avulla käyttäjäsovellus saa jonkin kontekstin tietojärjestelmältä ja tutkittuaan sen sisällön sovellus pyytää lisää informaatiota tarvittaessa. Lähtökohtana on se, että sovellus ei tiedä minkälaista tietoa on tarjolla ja että tietojärjestelmä ei tiedä minkälaista tietoa sovellus haluaa. Tietojärjestelmä etsii kontekstin uudella matemaattisella menetelmällä, joka kerää käyttäjäsovellukselle mahdollisesti relevanttia tietoa ympäröivästä tilanteesta.

Tutkielman käytännön tuloksena esitellään kontekstitietoinen arkkitehtuuri mobiiliympäristöihin, joka soveltuu esitellyn teorian toteuttamiseen.