

NAVI programme

DELIVERABLE PAM-7

SERVICE ARCHITECTURE AND METADATA
(PAM)

CORE SERVICE ARCHITECTURE FOR PERSONAL NAVIGATION

Version 0.999-3

31. 12. 2002

Contributors:
NAVI-PAM Project Group

Editor: Atte Kortekangas, VTT

Version history

Version	Date	Author(s)	Reviewer	Description
0.0-1	31.5.2002	(Project group)		First draft
0.0-2	28.9.2002	(Project group)		2nd draft
0.99	29.11.2002	(Project group)		Preliminary version
0.999	23.12.2002	(Project group)		Preliminary release version

Authors

Atte Kortekangas
VTT Information Technology
Tekniikantie 4 B
02044 Espoo
Finland
E-mail: atte.kortekangas@vtt.fi

Lassi Lehto
Finnish Geodetic Institute
P.O. Box 15
FIN-02431 Masala
Finland
E-mail: lassi.lehto@fgi.fi

Feedback, suggestions and expert consultation by

Mikko Lehtonen
VTT Building and Transport
P.O.Box 1800
FIN-02044 VTT
Finland
E-mail: mikko.j.lehtonen@vtt.fi

Reino Ruotsalainen
National Land Survey of Finland
Development Centre
P.O.Box 84
FIN-00521 Helsinki
Finland
E-mail: reino.ruotsalainen@nls.fi

Juha Törönen
VTT Information Technology
P.O.Box 1203
FIN-02044 VTT
Finland
E-mail: juha.toronen@vtt.fi

Matti Penttilä
VTT Information Technology
P.O.Box 1203
FIN-02044 VTT
Finland
E-mail: matti.penttila@vtt.fi
(Project NaviSearch)

Kalle Kärkkäinen
Locus Portal Inc.
Annankatu 31 – 33 C
FIN-00100 Helsinki
Finland
E-mail: kalle.karkkainen@locusportal.com, info@locusportal.com

Special thanks to Steering Committee on Architectural Issues for feedback and consultation

Especially to Antti Rainio, Markku Luoto and Jorma Marttinen

Abstract

This document summarises specific meta-level recommendations for the service architecture of a general, extensible and evolutionary realisation of personal navigation services in a network-oriented organisational context of individual users and a variety of organisational uses. The recommendations are reflected on a group of service pilot projects (NAVIttestbed, NAVImap and NAVIsearch) with the Finnish research programme NAVI for personal navigation services and some application-oriented projects within the programme and certain other related projects. A review of certain technologies, related with modelling of XML-based interfaces in UML, UML-based modelling techniques in general, and certain modelling cases based on UML, are included.

Contents

Abstract.....	iv
Contents.....	v
List of symbols and abbreviations.....	vii
I PART I – Executive Summary.....	1
I-1 Introduction.....	1
I-2 Main architectural recommendations.....	1
I-3 Rationale.....	2
I-4 Discussion.....	3
II PART II - Standards Framework.....	4
II-1 Introduction.....	4
II-1.1 Background → Architectural aspect of interface specification.....	4
II-1.2 Overview of proposed service architecture (Collection of services based on XML & HTTP).....	4
II-1.3 Associated Standards and Forums.....	5
II-1.3.1 .. HTTP.....	5
II-1.3.2 .. W3C: XML, XML Schema.....	5
II-1.4 General modelling tools: UML.....	6
II-1.5 Discussion.....	7
II-2 Background on XML-related specifications.....	7
II-2.1 Introduction.....	7
II-2.2 Generic XML and basic extension mechanisms.....	7
II-2.3 Example on Definition of XML Schema.....	9
II-2.4 References.....	10
II-2.5 SVG.....	11
II-2.6 Specification of XML-based interfaces by UML.....	11
II-2.6.1 .. Introduction.....	11
II-2.6.2 .. Case I: a systematic approach proposed by a tool designer group.....	11
II-2.7 Example on UML-based XML interfaces: The Finnish National Transport Telematics Development – Project KALKATI.....	14
II-2.8 Realisation of XML-based service Interfaces: HTTP and SOAP.....	18
II-2.8.1 .. HTTP.....	18
II-2.8.2 .. SOAP.....	19
II-3 Digital mapping standards – Review of related OGC activities.....	20
II-3.1 Introduction (Role, Consortium, relation to international standardisation).....	20
II-4 Interface standards for Location-based services created by LIF.....	20

	II-4.1	Introduction.....	20
	II-4.2	Overview of LIF Service Architecture.....	21
	II-4.3	Main architectural profiles for locationing services.....	23
	II-4.3.1	.. References	24
	II-4.4	LIF-MLP: an XML-based interface protocol for LBS.....	25
III	PART III	– Example Realisations by NAVI Pilot Projects.....	26
	III-1	Introduction to NAVI-Pilots (incl. "vertical ones": Image, MMM, WH@M)	26
	III-1.1	Project MMM — MultiMeetMobile	26
	III-1.2	WH@M — “the World in Your Hands on the Move”.....	27
	III-1.3	WAMPPI – WAP Multimedia Service Pilot.....	28
	III-1.4	IMAGE – Intelligent Mobility Agent for Complex Geographic Environments	29
	III-2	Introduction to NAVI service pilots and project goals	30
	III-3	NAVITestbed: A partial realisation of the LIF LBS architecture	31
	III-3.1	Introduction to the pilot service	31
	III-3.2	Basic Service Architecture (apply the modelling principles above).....	31
	III-4	NAVImap (and GIMODIG), Web Features and Web Mapping.....	32
	III-4.1	Introduction to goals of the pilot demonstration	32
	III-4.2	Basic Service Architecture.....	34
	III-4.3	XML Schema-based Data Model as a GML Application Schema	34
	III-4.4	An Online Geospatial Database	35
	III-4.5	An Implementation of the WFS Specification	35
	III-4.6	A WMS Implementation.....	35
	III-4.7	SVG Based Map Display	36
	III-4.8	Raster Image Support.....	36
	III-4.9	Discussion	36
	III-5	NAVI Search: Integration of Positioning, Map Services and OpenLS service query	37
	III-5.1	Introduction to goals of the pilot service demonstration.....	37
	III-5.2	Basic Service Architecture (apply the modelling principles above).....	37
IV	PART IV	– Tutorial Material on UML and Case studies.....	40
	IV-1	Architectural Modelling of Personal Navigation Services.....	40
	IV-1.1	Purpose and goals of architectural modelling	40
	IV-1.2	Overview of Methodology	41
	IV-1.3	A Short Guide to UML Use Case Notation.....	41
	IV-1.4	Components sources for bottom-up architecture.....	43
	IV-1.5	Methodology for top-down architectural analysis	43
	IV-1.6	Current Status of Architectural Modelling.....	44
	IV-2	Example: UML-based modelling of “ <i>Paikannussanasto</i> ” – the vocabulary of navigation & locationing technology in Finnish.....	45

List of symbols and abbreviations

2G — Second Generation

A — Interface between GERAN BSS and MSC

A-GPS — Assisted GPS

ATD — Absolute Time Difference

BSC — Base Station Controller

BSSAP-LE — Base Station System Application Part LCS Extension

BSSLAP — Base Station System Application Part

BTS — Base Transceiver Station

CBC — Cell Broadcast Center

CBC-BSC — Interface between CBC and BSC

CBC-SMLC — Interface between CBC and SMLC

D-GPS — Differential GPS

E-OTD — Enhanced Observed Time Difference

gsmSCF — GSM Service control function

HLR — (Home Location Register):

Lb — Interface between SMLC and BSC

LCS — Location Service(s)

LCS — Location Service

LMU — LCS Measurement Unit

MS — Mobile Station

MLC — Mobile Location Center

MSC — Mobile Switching Center

MSC/VLR — Mobile Switching Center in Visiting Location Register

PLMN — Public Land Mobile Networks

RIT — Radio Interface Timing

RRLP — Radio Resource Link Protocol

RTD — Real Time Difference

SCF — Service control function

SGSN — Serving GPRS Support Node (SGSN).

SMLC — (Serving Mobile Location Center

SMSCB — Short Message Service Cell Broadcast

SMLCPP — Serving Mobile Location Center Peer Protocol

TA — Timing Advance

TSG(s) — Technical Specification Group(s)

TSG-GERAN — TSG GSM/EDGE Radio Access Network

UDT — SCCP Unitdata message

UTC — Universal Coordinated Time

UMTS — Terrestrial Radio Access Network

I PART I – Executive Summary

I-1 Introduction

This report will summarise the results of project “*Service Architecture and Metadata — PAM*”, a horizontal supporting action of the Finnish national *NAVI Research Programme* on services, applications and methodology of *Personal Navigation*. The main topics of project work consisted of consultation on associated standards, patents, methodology and architectural recommendation.

The report consists of four parts. Part I will list the main architectural recommendations, including its rationale, a brief summary of the other parts of this report as well as other major reports by the project, to be presented as independent volumes.

Part II of this report will review specific topics related with design and specification of XML-based interfaces. The general concept is to employ XML-based methodologies as the technical realisation for extensible, self-documenting and publishable service interfaces. Related design methodologies based on *UML (Universal Modelling Language)* are reviewed, including a project case applying related methodology.

Part III will give a brief summary on the architectural framework of a number of service and application pilot projects of the NAVI Programme. The emphasis is on topics related with the proposed architectural model.

Part IV will contain material related with hands-on architectural modelling based on UML. There are tutorial material, general consideration and review of modelling activities carried during the project and a case study of applying UML modelling on “*Paikannussanasto*” (a vocabulary of personal navigation and positioning technology in Finnish, including a dictionary of corresponding terms in English and Swedish).

Other major reports produced during the project are the background report on standardisation activities related with Personal navigation Technology http://proxnet.vtt.fi/navi/pam/deliverables/PAM_D6_1_0.pdf¹ and a series of reviews on *patent abstracts*, distributed by a special arrangement to participating organisations.

I-2 Main architectural recommendations

It is recommended to use XML-based service interfaces. The following targets can and should be pursued in combination with this choice:

Interfaces should be based on public interfaces (preferably published, but at least publishable).

¹ Needs to be updated at about 2002-12-31, when D6 becomes available!

The interface should support extensibility, innovation and evolutionary development of information technology equipment and services.

The interface development should build on adoption of international standards and wide experience from related development well-established standardisation arena, including both industrial consortia and conventional (“official”) standardisation organisations.

Prominent organisations related with the present technology include generic XML protocols specifications by W3C. Locationing service protocols, initially specified by *LIF* (Location Interoperability Forum) and *Wap Forum*, are being targeted by *OMA* (Open Mobile Alliance), but possible success is to be verified in the future, only. GIS²- and mapping-related specifications have been worked on with success by OGC (Open GIS Consortium) and especially the interface definitions for location-based services by OpenLS (Open Location Services, a subgroup of OGC). Finally, UML-based modelling of XML-interfaces is the most prominent candidate for modelling methodology (UML has been specified by OMG, i.e. Object Management Group, but prominent software tool vendors have contributed to XML-modelling methodology).

There are long-term international standardisation activities related with telecommunication (e.g. in mobile communication), geographic information processing (e.g. ISO-TC-211) and transport telematics. One should keep eye on such developments, since they can provide well-established solutions for dedicated applications domains and specific contexts. They might even provide innovations and harmonised technical procedures of a more generalised nature applicable outside their original scope. Unfortunately, it is often not possible to review the contents of most of the official standards in accurate detail, since the publishing organisations are sometimes unusually picky about their copyrights to the material and it is often difficult to obtain the finished material in electronic form.

I-3 Rationale

Interfaces should be based on public interfaces, be published and be based on technology supporting publishing in order to support the neutral and reliable usage control of such interfaces. When the interfaces are public, the legal context of using such interfaces as part of commercial or public services is maximally clear (even though possible patent-related issues should still be verified). Anyway, it is most likely that none one will any longer be able to patent the idea of XML-based interfaces, defined by published XML-schema. Even if some is able to patent some specific interface scheme³, the claim could be circumvented by an alternative interface and XML-schema.

Also, a further study of the wide variety of security-related issues might lead to a conclusion that controlled security is easier to achieve by a standardised and published interface scheme. This is partly supported by the fact that it does not otherwise seem possible to adapt generally and internationally accepted legal principles to any artefacts or violations of the basic safety and security principles. It is not possible to find support for a

² *G(eographic) I(nformation) S(ystems)*

³ This is **not** to suggest of any landmark of professionalism by the authority granting such hypothetical patents.

claimed breach in court, unless the ground is clear both in legal and in technological terms.

The interface should support extensibility, innovation and evolutionary development of information technology equipment and services, since that will happen anyway, should the interface prove to be a success. The interfaces should adapt to different organisational environment (private customers, business-to-business etc.), economical compensation principle (transaction-based charges, subscription, public or promotional free material etc.), access control (public, chargeable, contractual or restricted material etc.)

I-4 Discussion

The list of architectural recommendations presented is a high-level one, essentially a recommendation for certain *architectural Meta levels*. This level can be viewed as *meta-architecture*, i.e. there are further levels of design needing coherence and associated guidelines for practical and successful designs. The latter levels are more customarily regarded as the information systems architecture. Since the industry of location-based services is young, the most important applications are evolving. Development projects are often funded by venture-like investments and rather little empirical experience has accumulated on the economical success of complete service and value chains, yet, more refined recommendations would be risky and (even more) speculative.

The current meta-architectural recommendations also suggest preparation for constant change – for the foreseeable future – which can be a safest bet even for cases where anomalies for the suggested meta-architecture are needed. In real life it is quite likely that e.g. some solutions targeting at complete service coverage, based on proprietary protocols and systems are expected or even likely. Also, it is difficult to forecast what such developments might be, since real “killer solutions” must at partially be based on innovation and novel discoveries on service gaps for some specific customer needs.

However, it rather clear that such revolutionary inventions definitely need a blend of various ingredients and unusual luck besides talent and hard work to realise and their inventors are unlikely to rigorously follow specific recommendations anyway. Therefore, the best one can prepare for is design for is effective and efficient interworking.

II PART II - Standards Framework

II-1 Introduction

II-1.1 Background → Architectural aspect of interface specification

An architecture for an information system refers to the available components, their properties and the general rules and methodology for realising applications. Thus, the description of architecture for an information system needs to list the possible components (with adequate detail), ways of combining the parts and planned-in mechanisms for extensions. While the topic and the result is typically rather abstract (since information systems mostly comprise of software, and especially, on the finalising design layer) illustrative presentation containing examples is needed. One will also typically benefit from special software tools targeting at illustrative and human-oriented description of the solution while also supporting highly iterative and conversational dialogue between all parties having direct interest in the specification procedure.

II-1.2 Overview of proposed service architecture (Collection of services based on XML & HTTP)

The proposed core architecture for personal navigation is targeted at a context where several (or a variable number) of organisations of various sizes and character, groups of people and individual citizens co-operate for realising location based services, more fundamental enabling technologies and services, higher level services and applications taking advantage of the underlying service layers. While there are several aspects of location-based services needing consideration from the privacy point of view and there are also other aspects that need to be realised by the aid of computer security technology, there is a need to categorise the principal approach, support variation, when needed, and adopt the most prevailing practices and standards being applied in the industry.

The general service architecture proposed is based on exchange of XML-encoded messages between parties. XML-encoded messages are special in the respect that textually encoded information is used for information transfer. Also, XML specifications refer to publicly available core specifications and extensions, whose specifications are supposed to be available for both parties of the target transaction. This is a sufficiently open and neutral or fair arrangement that is necessary with the evolving co-operation environment sketched above.

The relation of co-operating parties can be that of a customer and a service provider, typically resulting as the *client-server* relationship between the parties. The well-established *WWW-community* and their *HTTP-protocol* give a *de facto* solution such client-server models. However, there is more and more need for more *symmetric* and/or more *rigid* communication schemes, where there is need for communication between servers. Then, the servers don't only make a separate rendezvous to agree on their information exchange in isolation, but wish to agree based on more categorised model of

mutual trust, where public specification (i.e. the specification of the XML) is referred to. The most fundamental motivation behind such developments is economy and conceptual simplicity of the resulting rules of trust. It would be technically possible to create such relationships of trust in terms of pure client-server computing, but rules of ownership and especially the procedures and rules for settling possible conflicts of interest in this context would most likely make such an approach impractical.

II-1.3 Associated Standards and Forums

The current representation of the key architectural choices made with the NAVI program is not intended as a complete technical description of all possible standards having been used. Also, it is out of scope for this discussion to repeat the details of some widely used standards for which there are well-authored descriptions ranging from tutorials down to in-depth specification of every single detail. The associated forums being referred to below also contain such instructive material. Also, numerous educational texts and journal articles can be found via these links (although numerous other excellent representations on these topics are inevitably ignored).

II-1.3.1 HTTP

The *Hypertext Transport Protocol*, (HTTP) is essentially a generic mechanism for exchanging structured information as blocks of text, typically encoding the structure in terms of a *Markup Language* (e.g. HTML, or XML) encapsulated as a suitable carrier for (like MIME). There are variations of the scheme and the protocol is often unsymmetrical (like with conventional WWW-services), but the basic scheme is very simple, where the communicating parties just send a complete package of information in turn.

The lower transport layers of HTTP are typically realised as a TCP/IP-based service in the Internet context using standard technology. The standardisation of these layers belongs to the realm of *IETF* (*Internet Engineering Task Force*, an industrial consortium, see <http://www.ietf.org/>). The standards specifying HTTP are discussed in further depth in Section HTTP.

The higher layers specific to services and applications and the associated generic mark-up languages belong to the realm of W3C (World-Wide-Web Consortium, see <http://www.w3.org/>). The WWW (World-Wide-Web) is currently based on HTTP contents encoded as HTML (see <http://www.w3.org/MarkUp/>), while the return channel (user response) is either a simple substitution and transmission of server-provided form fields or some application-specific non-standard scheme based e.g. on Java and JavaScript. Currently, there is a transition from HTML to XML (and the intermediate form of XHTML emulating HTML-based structures with XML syntax).

II-1.3.2 W3C: XML, XML Schema

World Wide Web Consortium (<http://www.w3.org/>) is the originator and maintainer of the major standards specifications for generic data formats on abstract level⁴. The prevailing set of standards is based on expressing structured data by structured text, resulting as specific *mark-up languages*.

⁴ Abstract in the sense that information encoding is independent of implementation requirements for data storage, telecommunications, etc.

While the HTML mark-up language (cf. <http://www.w3.org/MarkUp/>) is still prevailing as the major basis for present WWW-technology, there has been a trend towards XML-based mark-up languages during the last few years. There is also a transitory (yet evolving) format called XHTML (cf. <http://www.w3.org/MarkUp/>), whose syntax is based on XML but whose semantics and identifier terminology closely follow the semantics for traditional HTML.

XML (cf. <http://www.w3.org/XML/>) is essentially a definition for the core of an extensible mark-up language only targeting at the structure of representation for information⁵. There are some important applications (or extensions) of XML. E.g. SVG (cf. <http://www.w3.org/Graphics/SVG/>) intended as general-purpose mark-up language for representing *vector graphics* like maps in the WWW context.⁶ The XML reference (<http://www.w3.org/XML/>) has further extensions on queries, references, structure of special documents, specification of appearance, but many of these are general-purpose mechanisms with little special interest for the current context.

However, the mechanisms for specification of XML extension are essential for any work involved with adaptation of XML-related technology for new applications or new contexts. A special notation called DTD (Document Type Definition) was originally designed for the purpose and much of existing special standards (like the LIF-MLP, see Section II-4.4) are defined in terms of DTD's. However, there is transition going on to do future specification by *XML Schema* (cf. <http://www.w3.org/XML/Schema>), an XML extension (application) with a specific semantics designed for the accurate description of any XML-related construction by an XML-based language⁷

XML-based protocols can also be used as a basis of symmetric, extensible and other specific general purpose protocols for object communication purposes. A specific example known as *SOAP* and specified at W3C will be considered in Section II-2.8.2.

II-1.4 General modelling tools: UML

When the task is to design complex information systems like *location-based service systems*, there is a need for effective tools to support the various phases of design. This is especially so, since the structure is geographically distributed, run by multiple operators (including LBS users) with shared responsibility on the correct structure and configuration of the integrated system (and applications) and the development cycle is more or less continuous.

The *Universal Modelling Language* (UML) specified by *Object Management Group* (OMG), cf. <http://www.omg.org/> is a widely adopted tool for illustrative, yet semantically accurate, description of complex (information) systems by a graphical notation. The UML specification is as such rather complex and not very illustrative (cf. <http://www.omg.org/technology/documents/formal/uml.htm> for the current version, i.e.

⁵ Unlike HTML, which is not extensible and inherently specifies certain rules or constraints for rendering the contents.

⁶ Non-scalable raster-based maps can also be represented as simple digital images, e.g. in PNG, GIF or TIFF (cf. <http://www.w3.org/Graphics/PNG/>), while SVG can also serve as an efficient format for representing raster maps or other images, but needs a heavier rendering machinery for output.

⁷ This is similar in concept for the well-proven technique like authoring the grammar for a natural language, say English, using the very same language.

version 1.4 as of now, version 2 under preparation). In practice, one needs a design tool to support drawing the UML diagram, syntax checking, cross-checking attribute values and carrying out modifications of attributes or more complex compounds to other parts dependent to them. OMG's web pages contain links to the variety of tools available for the purpose.

II-1.5 Discussion

There was originally the idea to include an extensive review of the practical modelling exercises done in UML with the current project as part of this report. However, we came to the conclusion that it is very difficult to support all the necessary requirements at the same time. The model should be representative enough for the most important topics in personal navigation. It should be illustrative and accurate as well. At the same time, the model should be generic enough to catch concepts with lasting value while hiding unnecessary detail and even contradictory approaches to personal navigation.

It is possible to achieve an acceptable compromise between these different goals with a typical modelling task whose only purpose is to serve the target of higher quality design for an information system. There, the modelling task does not need to look for balance for readability but can concentrate on parts deemed to critical for the design by various pragmatic points of view. On the other hand, all persons involved with the process must inherently be familiar with topic and have the common goal of willing to have a sufficiently thorough understanding of the result. There, the model is neither designed for the purpose of written document⁸ nor tutorial for the topic. Instead, it is intended as a tool for joint communication between people in charge of the variety of design targets.

II-2 Background on XML-related specifications

II-2.1 Introduction

The location protocol specifications by Location Interoperability Forum, LIF⁹, the Open GIS/Open LS service protocols and the WAP-related protocols by WapForum¹⁰ are all specified by XML-based extensions. In the following section we make a short review of the specification techniques for XML, i.e. the original specification model by DTD (inherited from the earlier SGML mark-up language for structured documents standardised by ISO) and the recent convention of specifying XML-extension by *XML-Schema*.

II-2.2 Generic XML and basic extension mechanisms

XML version 1 (see [W3C XML 1.0]) was specified syntactically in terms of DTD's (Document Type Definition's). This is special (and relatively simple) definition language targeted at syntax specification. DTD's have an origin with SGML specification, an

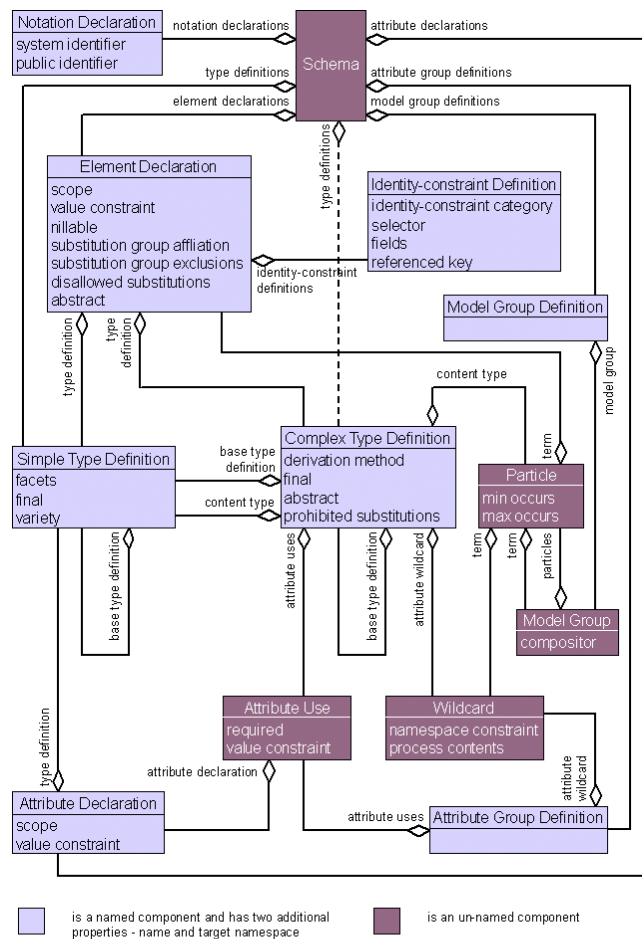
⁸ The resulting software and the technical documentation on it will be the main tool to serve the documentation purposes.

⁹ LIF has consolidated with Open Mobile Alliance (OMA), see <http://www.openmobilealliance.org>.

¹⁰ WAP Forum has been merged with Open Mobile Alliance (OMA), see <http://www.openmobilealliance.org>.

earlier and more complex documentation language concept standardised by ISO for general purpose document processing and archival. However, it is somewhat cumbersome to have a dedicated language with a unique syntax just for specification of XML-based formats. W3C has recently decided to adopt *XML Schema* as the new engine. This is a purely XML-based methodology for specification of XML-related formats.

Figure 1 illustrates the basic concepts of specification XML-based (syntactic) structures by XML Schemas. The illustration is an excerpt of the core specification for XML Schemas by W3C [W3C Schema Spec]. The basic idea of XML Schema is to specify a (meta-) language, by which one can specify the possible sentences in the target language (which is another version of XML). This is not unlike authoring a grammar to describe a natural language, or making a specification on how to make machine drawings by the aid of a directory of basic drawing components and rules of application.



XML Schema Component Data Model

Figure 1. XML Schema Concept Model (from [W3C Schema Spec]).

The only difference is that XML Schema must be absolutely specific so that it is possible to automate the generation of processing tools based on the Schema specification. On the other hand, XML Schema does not need to be unique in the sense that there could be

different realisations resulting in the same target language. However, as with the grammars for natural language, it is highly advantageous, if the Schemas would be standardised, so that there are no competing explanations for the most fundamental application protocols. This is even more important for the purposes of automated document exchange (as is typical with XML), because otherwise it becomes cumbersome or even impossible to relate the different specifications.

II-2.3 Example on Definition of XML Schema

The following example on defining an XML application protocol based by XML Schema is from the *W3C XML Primer* [W3C Schema Spec-0]. The protocol describes a simple purchase order.

Table 1. Example of an XML application protocol, to be defined by XML schema (the protocol is used for representing a purchase order).

The Purchase Order, po.xml

```
<?xml version="1.0"?>
<purchaseOrder orderDate="1999-10-20">
  <shipTo country="US">
    <name>Alice Smith</name>
    <street>123 Maple Street</street>
    <city>Mill Valley</city>
    <state>CA</state>
    <zip>90952</zip>
  </shipTo>
  <billTo country="US">
    <name>Robert Smith</name>
    <street>8 Oak Avenue</street>
    <city>Old Town</city>
    <state>PA</state>
    <zip>95819</zip>
  </billTo>
  <comment>Hurry, my lawn is going wild!</comment>
  <items>
    <item partNum="872-AA">
      <productName>Lawnmower</productName>
      <quantity>1</quantity>
      <USPrice>148.95</USPrice>
      <comment>Confirm this is electric</comment>
    </item>
    <item partNum="926-AA">
      <productName>Baby Monitor</productName>
      <quantity>1</quantity>
      <USPrice>39.98</USPrice>
      <shipDate>1999-05-21</shipDate>
    </item>
  </items>
</purchaseOrder>
```

Table 2. The corresponding XML schema for the application of **Table 1** is given below.

The Purchase Order Schema, po.xsd

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation xml:lang="en">
      Purchase order schema for Example.com.
      Copyright 2000 Example.com. All rights reserved.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:element name="purchaseOrder" type="PurchaseOrderType"/>
```

```

<xsd:element name="comment" type="xsd:string"/>

<xsd:complexType name="PurchaseOrderType">
  <xsd:sequence>
    <xsd:element name="shipTo" type="USAddress"/>
    <xsd:element name="billTo" type="USAddress"/>
    <xsd:element ref="comment" minOccurs="0"/>
    <xsd:element name="items" type="Items"/>
  </xsd:sequence>
  <xsd:attribute name="orderDate" type="xsd:date"/>
</xsd:complexType>

<xsd:complexType name="USAddress">
  <xsd:sequence>
    <xsd:element name="name" type="xsd:string"/>
    <xsd:element name="street" type="xsd:string"/>
    <xsd:element name="city" type="xsd:string"/>
    <xsd:element name="state" type="xsd:string"/>
    <xsd:element name="zip" type="xsd:decimal"/>
  </xsd:sequence>
  <xsd:attribute name="country" type="xsd:NMTOKEN"
    fixed="US"/>
</xsd:complexType>

<xsd:complexType name="Items">
  <xsd:sequence>
    <xsd:element name="item" minOccurs="0" maxOccurs="unbounded">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="productName" type="xsd:string"/>
          <xsd:element name="quantity">
            <xsd:simpleType>
              <xsd:restriction base="xsd:positiveInteger">
                <xsd:maxExclusive value="100"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
          <xsd:element name="USPrice" type="xsd:decimal"/>
          <xsd:element ref="comment" minOccurs="0"/>
          <xsd:element name="shipDate" type="xsd:date" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="partNum" type="SKU" use="required"/>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

<!-- Stock Keeping Unit, a code for identifying products -->
<xsd:simpleType name="SKU">
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="\d{3}-[A-Z]{2}"/>
  </xsd:restriction>
</xsd:simpleType>

</xsd:schema>

```

Further details on this example can be found in W3C XML primer [W3C Schema Spec-0].

II-2.4 References

[W3C XML 1.0] “Extensible Markup Language (XML) 1.0 (Second Edition); W3C Recommendation 6 October 2000” (Specification available at <http://www.w3.org/TR/REC-xml>).

[W3C Schema Spec-0] “XML Schema Part 0: Primer; W3C Recommendation 2 May 2001” (Current version of specification available at <http://www.w3.org/TR/xmlschema-0/>).

[W3C Schema Spec] “XML Schema Part 1: Structures; W3C Recommendation 2 May 2001” (Current version of specification available at <http://www.w3.org/TR/xmlschema-1/>).

[W3C Schema Spec-2] “XML Schema Part 2: Datatypes; W3C Recommendation 02 May 2001” (Current version of specification available at <http://www.w3.org/TR/xmlschema-2/>).

II-2.5 SVG

The acronym *SVG* stands for Scalable Vector Graphics. SVG is generic specification for representation of graphic information for on www-pages as XML. SVG was developed at W3C (<http://www.w3.org/Graphics/SVG/Overview.htm8>). An official W3C overview on SVG can be found via the W3C main page (or directly at <http://www.w3.org/Graphics/SVG/Overview.htm8> as of present).

II-2.6 Specification of XML-based interfaces by UML

II-2.6.1 Introduction

Architecture for an information system in general and for mobile location-based technology, as in the current context is a highly abstract concept and generally can benefit from figures, examples and other illustrative presentation methodology. However, higher accuracy and certain degree of formalism can make such presentations a more accurate means of communication on the meaning or *semantics* of the architectural description. As with any formalism which can also be regarded as an artificial language, there is the further step of learning to use it, i.e. to “learn to speak this language” until the concept can be understood or discussed at all. Nevertheless, once some fluency has been gained with the language, understanding the topics and communication on it will not only become more exact, but also becomes faster and more convenient.

The introduction of *UML (Unified Modelling Language)* for such a modelling tool is an example of desire to match the need for increased accuracy with highly illustrative presentation style. While, unfortunately, the combination of these goals is either mutually contradictory or needs prerequisite homework on learning the appropriate methodology, it is out of the scope of the current report to attempt to formulate everything in terms of an UML-based methodology. On the other hand, more specific and advanced discussion between experts at the stage of making decisions on system developments needs a systematic approach. While experts can discuss on high-level standards-based interfaces based on HTTP and XML, the illustrative properties of such fully text-based methodology is not sufficient for discussing work to be based on adaptation of new services and applications on top existing information systems and application contexts.

II-2.6.2 Case I: a systematic approach proposed by a tool designer group

A specification proposed for application of UML for specification of XML-based interfaces is given in “UML for XML Schema Mapping Specification” by Grady Booch, Magnus Christerson, Matthew Fuchs and Jari Koistinen found at web site of Rational Inc. (<http://www.rational.com/>) and dated as 12/08/99 [G. Booch, M. Christerson, M. Fuchs, J. Koistinen; 1999-8]). The mapping of elements between the XML-based

interface, specified by and XML-schema employing namespaces, and UML is summarised in Table 3.

Table 3. Mapping between XML specified by XML Schema and Namespaces [G. Booch, M. Christerson, M. Fuchs, J. Koistinen; 1999-8].

Stereotype	UML Construct	SOX Meaning
<<sox>>	Package	Indicates a full Schema
<<elementtype>>	Class	Element type definition
<<sequence>>	Nested Class	Sequence group from a content model
<<choice>>	Nested Class	Choice group from a content model
<<enumeration>>	Class – may be nested	Enumeration datatype – <i>can be UML enumeration</i>
<<scalar>>	Class – may be nested	Scalar datatype
<<varchar>>	Class – may be nested	Varchar datatype
<<implied>>	Attribute or Unidirectional Association	Indicates an implied attribute
<<required>>	Attribute or Unidirectional Association	Indicates a required attribute
<<default>>	Attribute or Unidirectional Association	Indicates a default attribute
<<fixed>>	Attribute or Unidirectional Association	Indicates a fixed attribute
<<content>>	Attribute or Aggregation	Indicates an atom in a content model

Table 4. Example on the conversion of a specific XML Schema definition (shown here) into a corresponding UML Schema specification (shown in Figure 2 below) [G. Booch, M. Christerson, M. Fuchs, J. Koistinen; 1999-8].

```

1. <schema uri = "urn:document:po.sox">
2. <datatype name = "DocStates">
3.   <enumeration datatype = "NMTOKEN">
4.     <option>Submit</option>
5.     <option>Accept</option>
6.     <option>Reject</option>
7.   </enumeration>
8. </datatype>
9.
10.<datatype name = "CountryCode">
11.  <enumeration datatype = "NMTOKEN">
12.    <option>USA</option>
13.    <option>ENG</option>
14.    <option>GER</option>
15.    ...
16.  </enumeration>
17.</datatype>
18.
19.<datatype name = "Price">
20.  <scalar digits = "5" decimals = "4"/>
21.</datatype>
22.
23.<elementtype name = "DocProcess">
24.  <model>
25.    <string datatype = "DocStates"/>
26.  </model>
27.</elementtype>

```

```

28.
29. <elementtype name = "Address">
30.   <model>
31.     <sequence>
32.       <element name = "name" type = "string"/>
33.       <element name = "quantity" type = "int"/>
34.       <element name = "cost" type = "Price"/>
35.     </sequence>
36.   </model>
37. </elementtype>
38.
39. <elementtype name = "InternatAddress">
40.   <extends type = "address">
41.     <append>
42.       <element name = "country" type = "CountryCode"/>
43.     </append>
44.     <attrdef name = "language" type = "LanguageCode">
45.       <default>ENG</default>
46.     </attrdef>
47.   </extends>
48. </elementtype>
49.
50. <elementtype name = "PurchaseOrder">
51.   <model>
52.     <sequence>
53.       <element name = "shipTo" type = "Address"/>
54.       <element name = "billTo" type = "Address"/>
55.       <sequence name = "lineItem" occurs = "+">
56.         <element name = "name" type = "string"/>
57.         <element name = "quantity" type = "int"/>
58.         <element name = "cost" type = "Price"/>
59.       </sequence>
60.       <element type = "DocProcess"/>
61.     </sequence>
62.   </model>
63. </elementtype>
64. </schema>
65. </schema>

```

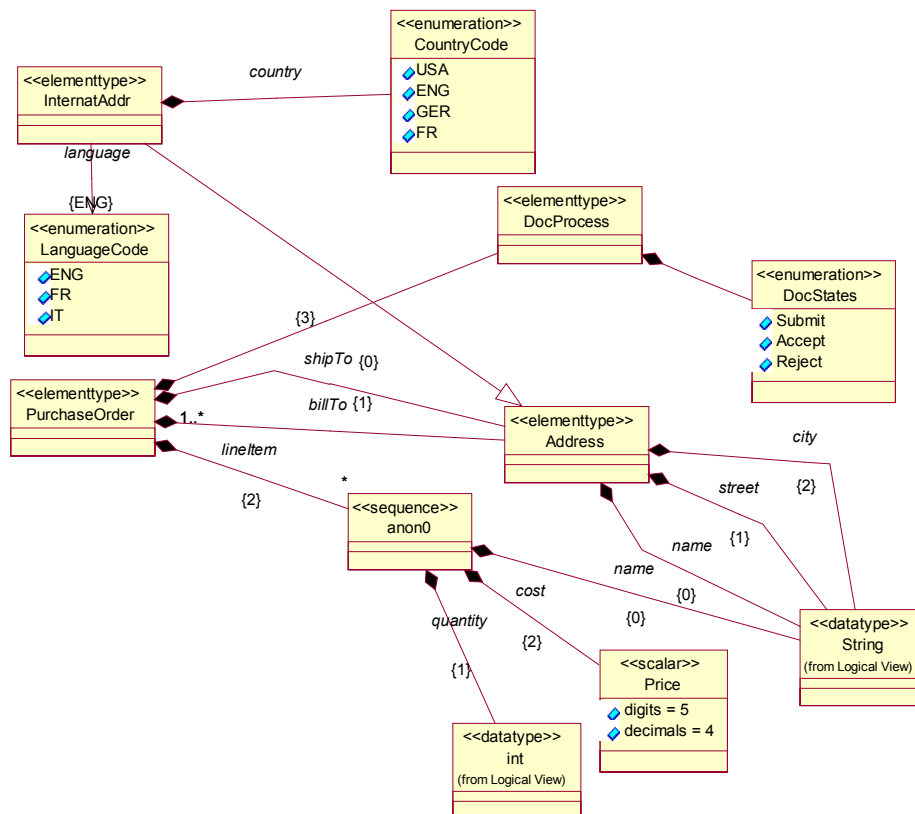


Figure 2. UML Schema corresponding to previous XML Schema example in Table 3 [G. Booch, M. Christerson, M. Fuchs, J. Koistinen; 1999-8].

In practice, one needs a special tool designed for modelling of XML interfaces in terms of UML (unless a sketchy approach targeting at a basic understanding and discussion of an interface is sufficient). For instance, there are versions of *Rational Rose* (cf. <http://www.rational.com/>) designed for the purpose.

II-2.6.3 Reference

[G. Booch, M. Christerson, M. Fuchs, J. Koistinen; 1999-8]: “*UML for XML Schema Mapping Specification*” by Grady Booch, Magnus Christerson, Matthew Fuchs and Jari Koistinen at http://www.rational.com/media/uml/resources/media/uml_xmlschema33.doc.

II-2.7 Example on UML-based XML interfaces: The Finnish National Transport Telematics Development – Project KALKATI

Project KALKATI (An acronym for “Kaikki liikennemuodot kattava liikenteen tietojärjestelmä”, in Finnish – with the meaning “an information for traffic covering all means of transport”) has been modelling a set of XML-based interfaces for transport telematics by UML. The material is available at <http://www.kalkati.net/>, but not yet released for public at the time of writing.

Figure 3 Represents the site logo. Figure 4 represents the interfaces being defined and Figure 5 and expanded view for components and versions for interfaces on schedules. Figure 6 and Figure 7 represent the actual schema and Figure 8 depicts the corresponding UML model.



Figure 3. Site logo for the KALKATI project web server.

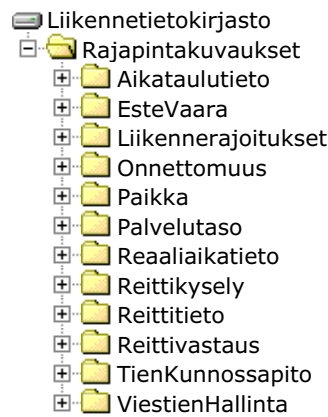


Figure 4. XML Interfaces to be specified with KALKATI.

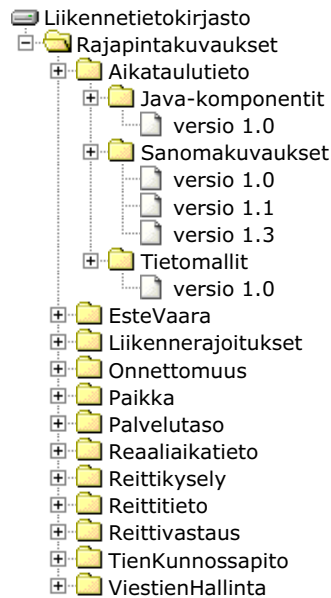


Figure 5. Expanded view of interfaces under construction: structure and versions for *schedules* have been expanded.

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- Aikataulutieto-sanoman rakennekuvaus V1.0. -->
<xsd:schema xmlns:xsd="http://www.w3.org/2000/10/XMLSchema">
  <xsd:element name="Aikataulutieto">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Vuorotieto" type="VuorotietoType" maxOccurs="unbounded"/>
        <xsd:element name="ViestienHallinta" type="ViestienHallintaType"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:complexType name="VuorotietoType">
    <xsd:sequence>
      <xsd:element name="Tunnistetieto" type="TunnistetietoType"/>
      <xsd:element name="Aikatieto" type="AikatietoType"/>
      <xsd:element name="MuuTieto" type="MuuTietoType" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="TunnistetietoType">
    <xsd:attribute name="Linjatunnus" type="xsd:string" use="required"/>
    <xsd:attribute name="Nimi" type="xsd:string" use="optional"/>
    <xsd:attribute name="Kulkumuoto" type="xsd:integer" use="optional"/>
    <xsd:attribute name="JLSuunta" type="xsd:string" use="optional"/>
  </xsd:complexType>
  <xsd:complexType name="AikatietoType">
    <xsd:sequence>
      <xsd:element name="Reittipistetieto" type="ReittipistetietoType"/>
    </xsd:sequence>
    <xsd:attribute name="Lahtoaika" type="xsd:integer" use="required"/>
    <xsd:attribute name="Saapumisaika" type="xsd:integer" use="optional"/>
    <xsd:attribute name="JLVoimassaoloaika" type="xsd:string" use="required"/>
  </xsd:complexType>
  <xsd:complexType name="MuuTietoType">
    <xsd:attribute name="Lahtopaikka" type="xsd:string" use="optional"/>
    <xsd:attribute name="Maarapaikka" type="xsd:string" use="optional"/>
    <xsd:attribute name="Lahtolaituri" type="xsd:string" use="optional"/>
    <xsd:attribute name="Saapumislaituri" type="xsd:string" use="optional"/>
    <xsd:attribute name="Hinta" type="xsd:string" use="optional"/>
    <xsd:attribute name="Kalenteripaiva" type="xsd:string" use="optional"/>
    <xsd:attribute name="Paivatyyppi" type="xsd:string" use="optional"/>
    <xsd:attribute name="Lisatieto" type="xsd:string" use="optional"/>
  </xsd:complexType>
  <xsd:complexType name="ReittipistetietoType">
    <xsd:sequence>
      <xsd:choice>
        <xsd:element name="Pysakki" type="PysakkiType"/>
        <xsd:element name="MuuPiste" type="MuuPisteType"/>
      </xsd:choice>
    </xsd:sequence>
    <xsd:attribute name="AikatMukOhitusaika" type="xsd:integer" use="required"/>
  </xsd:complexType>
  <xsd:complexType name="PysakkiType">
    <xsd:attribute name="Pysakkitunnus" type="xsd:integer" use="required"/>
    <xsd:attribute name="Nimi" type="xsd:string" use="optional"/>
  </xsd:complexType>
  <xsd:complexType name="MuuPisteType">
    <xsd:attribute name="Tunnus" type="xsd:string" use="required"/>
  </xsd:complexType>
  <xsd:complexType name="ViestienHallintaType">
    <xsd:sequence>
      <xsd:element name="ViestinLahetysaika" type="xsd:string"/>
      <xsd:element name="ViestinLahetNimi" type="xsd:string"/>
      <xsd:element name="ViestinLahetTunnus" type="xsd:nmtoken"/>
      <xsd:element name="ViestinTunnus" type="xsd:string"/>
      <xsd:element name="Versioaika" type="xsd:string" minOccurs="0"/>
      <xsd:element name="Tilannenumero" type="xsd:string" minOccurs="0"/>
      <xsd:element name="Tapahtumanumero" type="xsd:string" minOccurs="0"/>
      <xsd:element name="Lahdejarjestelma" type="xsd:string"/>
      <xsd:element name="TiedonValittajanTunnus" type="xsd:nmtoken" minOccurs="0" maxOccurs="2"/>
      <xsd:element name="TiedonValittajanNimi" type="xsd:string" minOccurs="0" maxOccurs="2"/>
      <xsd:element name="TietolahteenTyyppi" type="TietolahteenTyyppiType" minOccurs="0"/>
      <xsd:element name="TietolahteenTunnus" type="xsd:string" minOccurs="0"/>
      <xsd:element name="TietolahteenNimi" type="xsd:string" minOccurs="0"/>
      <xsd:element name="Luottamuksellisuus" type="LuottamuksellisuusType" minOccurs="0"/>
      <xsd:element name="TiedonLuotettavuus" type="TiedonLuotettavuusType" minOccurs="0"/>
      <xsd:element name="Kielikoodi" type="xsd:nmtoken" minOccurs="0"/>
      <xsd:element name="Kiireellisyys" type="xsd:string" minOccurs="0"/>
      <xsd:element name="TiedonJakelumuoto" type="xsd:string" minOccurs="0"/>
      <xsd:element name="Ennustekoodi" type="xsd:nmtoken" minOccurs="0"/>
      <xsd:element name="TiedonVarmistusaika" type="xsd:string" minOccurs="0"/>
      <xsd:element name="Voimassaoloaika" type="xsd:string" minOccurs="0"/>
      <xsd:element name="TiedonJakeluvai" type="xsd:string" minOccurs="0"/>
      <xsd:element name="PaivittainenTiedonPaivitysvali" type="xsd:string" minOccurs="0"/>
      <xsd:element name="TiedonJulkaisuaika" type="xsd:string" minOccurs="0"/>
      <xsd:element name="Kohdealue" type="KohdealueType" minOccurs="0"/>
    </xsd:sequence>
  </xsd:complexType>

```

Figure 6. XML schema definition for schedules; part I.

```

        <xsd:element name="TiedonJakeluarv" type="xsd:string" minOccurs="0"/>
        <xsd:element name="TiedonJakeluarvNimi" type="xsd:string" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="TietolahteenTyyppiType">
    <xsd:simpleContent>
        <xsd:restriction base="xsd:string">
            <xsd:attribute name="Koodi" type="xsd:char" use="optional"/>
            <xsd:attribute name="Teksti" type="xsd:string" use="optional"/>
        </xsd:restriction>
    </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="LuottamuksellisuusType">
    <xsd:simpleContent>
        <xsd:restriction base="xsd:string">
            <xsd:attribute name="Koodi" type="xsd:char" use="optional"/>
            <xsd:attribute name="Teksti" type="xsd:string" use="required"/>
        </xsd:restriction>
    </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="TiedonLuotettavuusType">
    <xsd:simpleContent>
        <xsd:restriction base="xsd:string">
            <xsd:attribute name="Koodi" type="xsd:char" use="optional"/>
            <xsd:attribute name="Teksti" type="xsd:string" use="optional"/>
        </xsd:restriction>
    </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="KohdearvType">
    <xsd:simpleContent>
        <xsd:restriction base="xsd:string">
            <xsd:attribute name="Koodi" type="xsd:char" use="optional"/>
            <xsd:attribute name="Teksti" type="xsd:string" use="optional"/>
        </xsd:restriction>
    </xsd:simpleContent>
</xsd:complexType>
</xsd:schema>

```

Figure 7. XML schema definition for schedules; part II

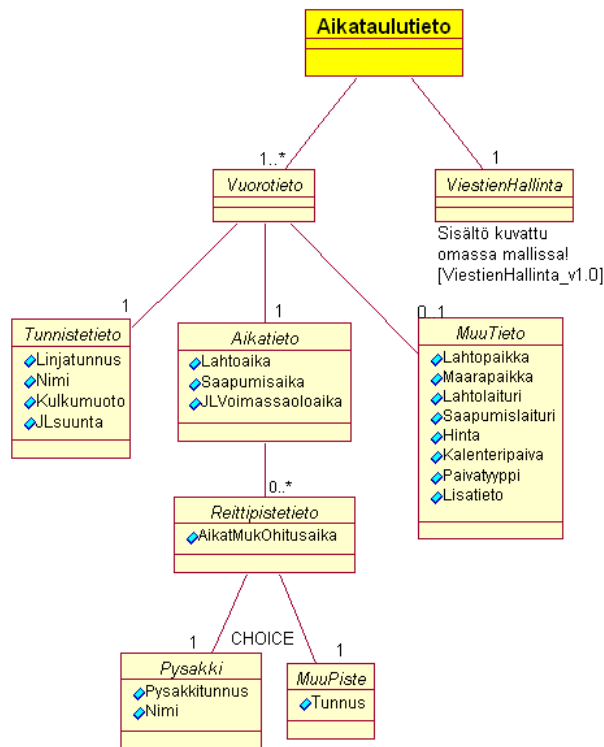


Figure 8. UML representation for XML schema.

II-2.8 Realisation of XML-based service Interfaces: HTTP and SOAP

II-2.8.1 HTTP

HTTP is an industrial de facto standard. Further specifications can be found at

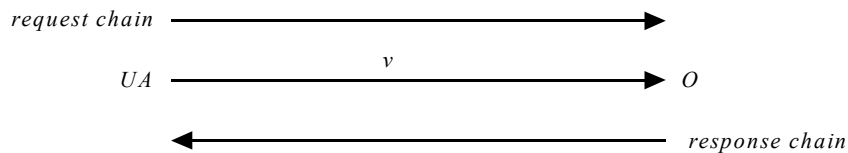
- <http://www.w3.org/> ==> HTTP
 - <http://www.w3.org/Protocols/> ... text ==> Rfc2616.txt (Available from W3C at <http://www.w3.org/Protocols/> or directly at <http://www.w3.org/Protocols/rfc2616/rfc2616.txt>) or
 - IETF at <http://www.ietf.org/rfc.html> or directly at <http://www.ietf.org/rfc/rfc2616.txt?number=2616>).

The general principle of the HTTP-based transmission is explained in the following quotation from rfc2616.txt (from June 1999, describing http protocol version 1.1):

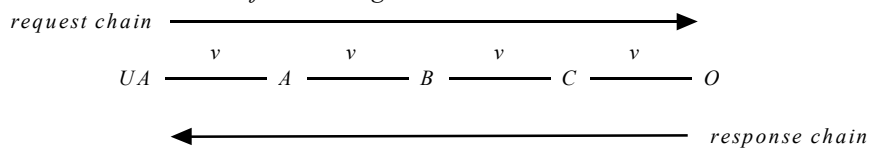
Overall Operation

The HTTP protocol is a request/response protocol. A client sends a request to the server in the form of a request method, URI, and protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content over a connection with a server. The server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity metainformation, and possible entity-body content. The relationship between HTTP and MIME is described in appendix¹¹....

Most HTTP communication is initiated by a user agent and consists of a request to be applied to a resource on some origin server. In the simplest case, this may be accomplished via a single connection (v) between the user agent (UA) and the origin server (O).



A more complicated situation occurs when one or more intermediaries are present in the request/response chain. There are three common forms of intermediary: proxy, gateway, and tunnel. A proxy is a forwarding agent, receiving requests for a URI in its absolute form, rewriting all or part of the message, and forwarding the reformatted request toward the server identified by the URI. A gateway is a receiving agent, acting as a layer above some other server(s) and, if necessary, translating the requests to the underlying server's protocol. A tunnel acts as a relay point between two connections without changing the messages; tunnels are used when the communication needs to pass through an intermediary (such as a firewall) even when the intermediary cannot understand the contents of the messages.

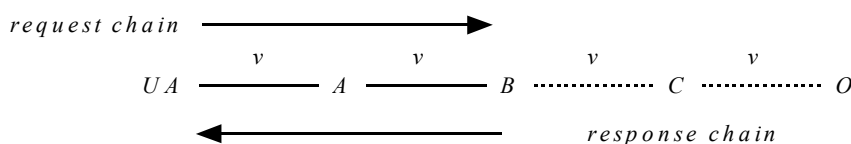


The figure above shows three intermediaries (A, B, and C) between the user agent and origin server. A request or response message that travels the whole chain will pass

¹¹ Please Consult the original RFC for details.

through four separate connections. This distinction is important because some HTTP communication options may apply only to the connection with the nearest, non-tunnel neighbor, only to the end-points of the chain, or to all connections along the chain. Although the diagram is linear, each participant may be engaged in multiple, simultaneous communications. For example, B may be receiving requests from many clients other than A, and/or forwarding requests to servers other than C, at the same time that it is handling A's request.

Any party to the communication which is not acting as a tunnel may employ an internal cache for handling requests. The effect of a cache is that the request/response chain is shortened if one of the participants along the chain has a cached response applicable to that request. The following illustrates the resulting chain if B has a cached copy of an earlier response from O (via C) for a request which has not been cached by UA or A.



Not all responses are usefully cacheable, and some requests may contain modifiers which place special requirements on cache behavior. HTTP requirements for cache behavior and cacheable responses are defined in section 13.

In fact, there are a wide variety of architectures and configurations of caches and proxies currently being experimented with or deployed across the World Wide Web. These systems include national hierarchies of proxy caches to save transoceanic bandwidth, systems that broadcast or multicast cache entries, organizations that distribute subsets of cached data via CD-ROM, and so on. HTTP systems are used in corporate intranets over high-bandwidth links, and for access via PDAs with low-power radio links and intermittent connectivity. The goal of HTTP/1.1 is to support the wide diversity of configurations already deployed while introducing protocol constructs that meet the needs of those who build web applications that require high reliability and, failing that, at least reliable indications of failure.

HTTP communication usually takes place over TCP/IP connections. The default port is TCP 80 [19], but other ports can be used. This does not preclude HTTP from being implemented on top of any other protocol on the Internet, or on other networks. HTTP only presumes a reliable transport; any protocol that provides such guarantees can be used; the mapping of the HTTP/1.1 request and response structures onto the transport data units of the protocol in question is outside the scope of this specification.

In HTTP/1.0, most implementations used a new connection for each request/response exchange. In HTTP/1.1, a connection may be used for one or more request/response exchanges, although connections may be closed for a variety of reasons (see section 8.1).

II-2.8.2 SOAP

For instance, the LIF Mobile Location Protocol (cf. Section II-4.4 below) has a structure, where information is transferred both ways between the client and the server as XML-encoded messages. Such operations can take advantage of generic middleware platform definition like SOAP to take care of communication layer. Although not currently used with the NAVI architectures considered here, SOAP might serve a conceptual framework and practical solution for some of these tasks. It is especially important to adopt some systematic approach to communication between service points, when the service

environment has become integrated enough to systematically support automated consultation by other automated services.

References for SOAP specifications can be found via the following links:

- <http://www.w3.org/> ==> SOAP/XMLP
 - <http://www.w3.org/2000/xp/Group/>
 - <http://www.w3.org/TR/2001/WD-xmlp-scenarios-20011217/>
 - <http://www.w3.org/TR/2001/WD-soap12-part1-20011217/>
 - <http://www.w3.org/TR/2001/WD-soap12-part2-20011217/>

II-3 Digital mapping standards – Review of related OGC activities

II-3.1 Introduction (Role, Consortium, relation to international standardisation)

Open GIS Consortium, OGC, is an industrial consortium of more than 200 companies, governmental agencies, universities and research organisations propagating common standards and practices for exchange, archival, processing and specification for information to be processed with *Geographic Information Systems* (GIS). Further information on OGC can be found at <http://www.opengis.org/>. Recently, OGC has sought for more intimate co-operation with the international official standardisation arena, i.e. ISO TC-211 working on a universal set of standards for the wide area of GIS-related standards.

The *OpenGIS Abstract Specification* series (see <http://www.opengis.org/techno/abstract.htm>) provide the framework or reference model for OpenGIS Implementation Specifications. This set can serve high-level guide to the variety of technical development of OpenGIS Specifications. The OpenGIS Implementation Specifications are engineering specifications that implement part of the Abstract Specification for particular distributed computing platforms (see <http://www.opengis.org/techno/implementation.htm>).

Topics of special architectural interest within the latter set include *GML* (*Geography Markup Language*, an XML-based extension), Web Map Server Interfaces and Web Feature Service as components of an interface to map collections. They are all used with NAVImap (cf. Section III-4) and considered in further depth in that context.

II-4 Interface standards for Location-based services created by LIF

II-4.1 Introduction

Location Interoperability Forum, LIF, was established as an *ad hoc* standardisation group on techniques related with network locationing technology in autumn 2000 by Ericsson, Motorola and Nokia. LIF became incorporated in autumn 2001 and has been merged with *Open Mobile Alliance*, OMA, in autumn 2002. Archives of LIF specifications were

available at <http://www.locationforum.org/> (and directly at <http://www.locationforum.org/publicdownload/>) until mid November 2002 (the LIF site was brought down in late November 2002).

Future documents will be directly available from OMA (see <http://www.openmobilealliance.org/>). Also, a collection of the most critical LIF documents (needed e.g. as reference for future specification work, other standardisation bodies etc.) are likely to be available at the OMA site in the near future.

II-4.2 Overview of LIF Service Architecture

The service architecture of LIF can be considered from several different aspects depending on whether one is interested in general concepts, networking systems or value added services, i.e. *Location-Based Services (LBS)* to be designed as applications of LIF-specific services. Also, LIF is not focused on providing a universal location-enabled solution for a general location-dependent context of usage, but specification of critical interfaces and associated methodology and procedures when location technology needs to be realised jointly in co-operation with independent enterprises and organisations or parties. Since LIF is organised around mobile communication business, the most relevant aspects are the underlying networking systems architecture and the interface protocols taking advantage of the enabling services of the former layer for provisioning of value-added location-based services.

The systems-oriented aspect closely corresponds with the general architecture of mobile communication networks, but the picture will become more and more technology-dependent, when a closer look will be taken. This is so, since the networking architecture varies e.g. between GSM and 3G technologies as well as between European, U.S. and Japanese technologies and their varieties. The interface or protocol aspect is more specific as characterised by the LIF-MLP protocol.

LIF interoperability testing group has documented the systems architectural view in connection with their effort to understand the various major networking systems components taking part in realisation of location services. The diagram in Figure 9 (quoted from [LIF TD 201 V.3.0.0 2002-2]) describes the LIF systems architecture.

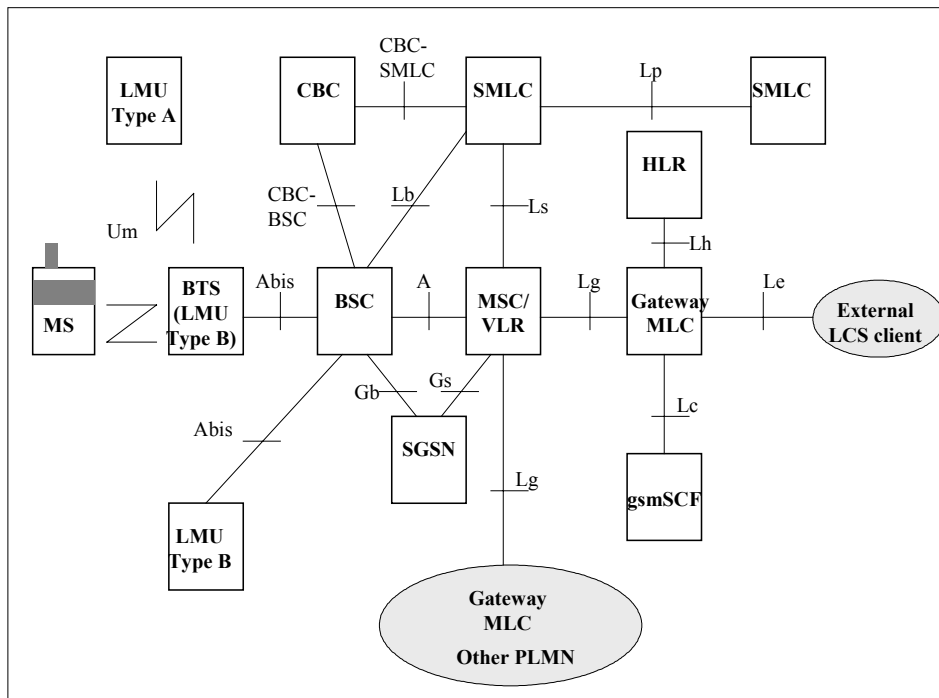


Figure 9. Logical LCS architecture in release 98/99 standard (Excerpt from [LIF TD 201 V.3.0.0 2002-2]; see text for an explanation).

A short explanation of the terminology might help understanding the concepts (list based on [LIF TS 202 V.3.0.0 2002-2]); there is a collection of these and other terms in *List of symbols and abbreviations* above.

- **MS (Mobile Station):** mobile phone, communicator or computer access to mobile network
- **LCS (LoCation Services):** LCS is a service concept in system standardisation. LCS specifies all the necessary network elements and entities, their functionality, interfaces, as well as communication messages, due to implement the positioning functionality in a cellular network. Note that LCS does not specify any location based (value added) services except locating of emergency calls
- **LMU:** LCS Measurement Unit
- **Type A LMU:** accessed exclusively over the air interface (Um interface): there is no wired connection to any other network element.(LMU)
- **Type B LMU:** accessed over the Abis¹² interface from a BSC. The LMU may be either a standalone network element addressed using some pseudo-cell ID or connected to or integrated in a BTS.
- **CBC (Cell Broadcast Center):** For Location Services, when a Cell Broadcast Centre (CBC) is associated with a BSC, the SMLC may interface to a CBC in order to broadcast assistance data using existing cell broadcast capabilities. The SMLC shall behave as a user, Cell Broadcast Entity, to the CBC (refer to 3GPP TS 03.41 [4]).
- **MLC:** Mobile Location Centre

¹² The reader may refer to Specifications of the GSM System.

- SMLC (Serving Mobile Location Centre):** The SMLC is either a separate network element of Cellular Network or Remote Access Network or integrated functionality in BSC that contains functionality required supporting LCS (LoCation Services). The SMLC manages the overall co-ordination and scheduling of resources required for the location of a mobile. It also calculates the final location estimate and estimates the achieved accuracy. The SMLC may control a number of LMUs (Location Measurement Units) for the purpose of obtaining radio interface measurements to locate or help locate MS (mobile service) subscribers in the area that it serves. The SMLC is administered with the capabilities and types of measurement produced by each of its LMUs. The radio interface timing measurement returned by an LMU to a SMLC has a generic status in usable for more than one location method.

II-4.3 Main architectural profiles for locationing services

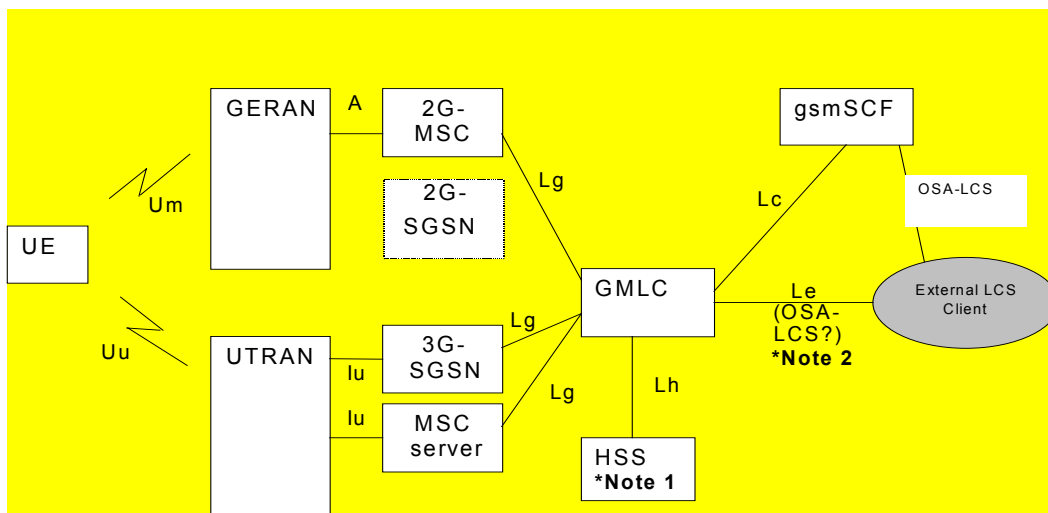


Figure 10. Architectural evolution towards UMTS.

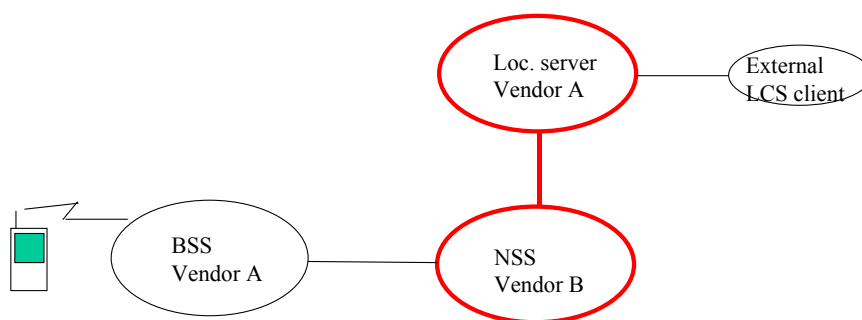


Figure 11. Case 1) NSS based LCS architecture; LCS feature and the core network from different vendors; LCS network topology, case 1 (Excerpt from [LIF TD 201 V.3.0.0 2002-2]).

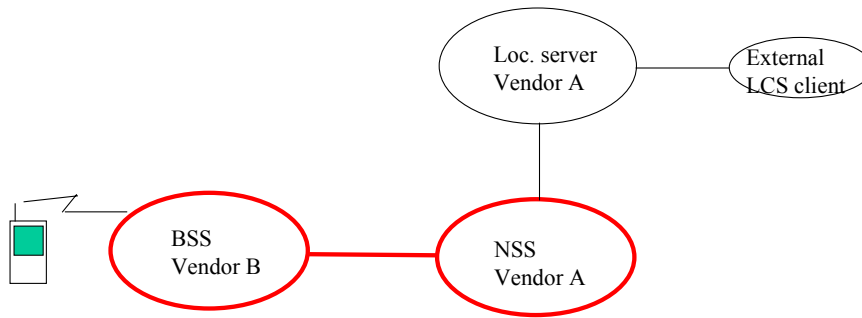


Figure 12. Case 2) NSS based LCS architecture; LCS feature and the core network from the same vendor, radio network from a different vendor; LCS network topology, case 2 (Excerpt from [LIF TD 201 V.3.0.0 2002-2]).

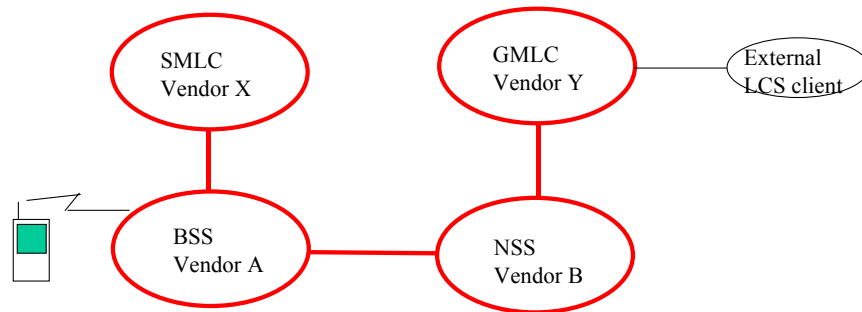


Figure 13. Case 3) BSS based LCS architecture, SMLC, GMLC, radio network and core network from different vendors; LCS network topology, case 3 (Excerpt from [LIF TD 201 V.3.0.0 2002-2]).

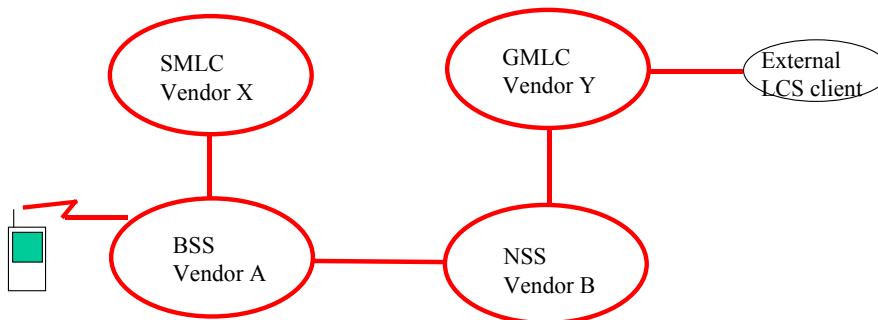


Figure 14. Case 4) Interoperability in case of end-to-end service; LCS network topology, case 4 (Excerpt from [LIF TD 201 V.3.0.0 2002-2]).

II-4.3.1 References

[G. Booch, M. Christerson, M. Fuchs, J. Koistinen; 1999-8] “UML for XML Schema Mapping Specification”, 1999 (Memo available at http://www.rational.com/media/uml/resources/media/uml_xmlschema33.pdf at the time of writing)

[LIF TD 201 V.3.0.0 2002-2] “Location Interoperability Forum; Interoperability Testing Group, The challenge with inter-operability in LCS”, Version 3.0.0, February

2002. (Document was available at <http://www.locationforum.org/publicdownload/LIF-TD-201-v3.0.0.zip> at the time of writing, until November 2002)

[LIF TS 202 V.3.0.0 2002-2] “Location Interoperability Forum; Interoperability Testing Group Location Services (LCS) Inter-operability Test (IOT) specification in GSM (Compliant to GSM Release 98/99 LCS Standards); (Document available at <http://www.locationforum.org/publicdownload/LIF-TS-202-v3.0.0.zip> at the time of writing, until November 2002)

II-4.4 LIF-MLP: an XML-based interface protocol for LBS

The LIF MLP Protocol is an XML-based protocol intended for a generic interface between a Locating Service (typically a network-positioning service to position mobile stations connected to mobile network) and various *Location-Based Services* (LBS)¹³. The complete specification has been made public by *Location Interoperability Forum* (LIF) and is being maintained by its successor organisation¹⁴. The specification was available at <http://www.locationforum.org/publicdownload/> (until November 2002). The specification includes the DTD for the MLP protocol extension and a detailed explanation of the context, contents and semantics of the various elements.

The specification is based on the idea of structured query and answer. The query may choose a basic format among a few alternatives and include further optional parts (or parameters). The whole query is then encoded as an XML record (according to the MLP DTD) and transmitted to the service point. It is the responsibility of the service side to parse and analyse the message and serve the result as requested. The result is similarly encoded as another XML-based MLP message according to the respective part of the specification and sent back to the client.

There are variations of this scheme (for instance, the client might ask for deferred services, i.e. the server will respond at some later instant so that the question is no more about a typical client-server transaction, but of a more symmetric type of relation of communication). However, the reader might notice that even the essence of protocol relation makes it not possible to map the protocol directly a standard HTTP/HTML-base WWW-transaction, but some symmetric scheme like *Soap* (cf. Section II-2.8.2) would be needed for the transport mechanism. Another possibility is to make a private (non-standard) extension of specification and attach a special *servlet* routine to process the request and uplink information from client at the server end.

¹³ The general class of LBS may include component services specially tailored for unique user applications.

¹⁴ LIF has consolidated with *Open Mobile Alliance* (OMA) in autumn 2002, see <http://www.openmobilealliance.org/>.

III PART III – Example Realisations by NAVI Pilot Projects

III-1 Introduction to NAVI-Pilots (incl. "vertical ones": Image, MMM, WH@M)

In this section we will review three application projects associated with the NAVI Programme in short. These projects are *MMM (MultiMetMobile)*, a national Finnish project finished in 2001 and funded by Tekes, *WH@M (World in your Hands on the Move)*, a project finished in 2002, and *IMAGE*, an ongoing international project, both with the *EU IST-Programme*. Each one of these projects have suggested and applied a certain architectural context, which is close correspondence with the general architectural context considered in this report.

III-1.1 Project MMM — MultiMeetMobile

The goal of project was to identify scientific results in transaction mechanisms for mobile billing, location-dependent applications and queries as well as compression techniques for mobile multimedia data. A pilot system was constructed for XML-based data interface for mobile location based services. The corresponding systems architecture is depicted in Figure 15.

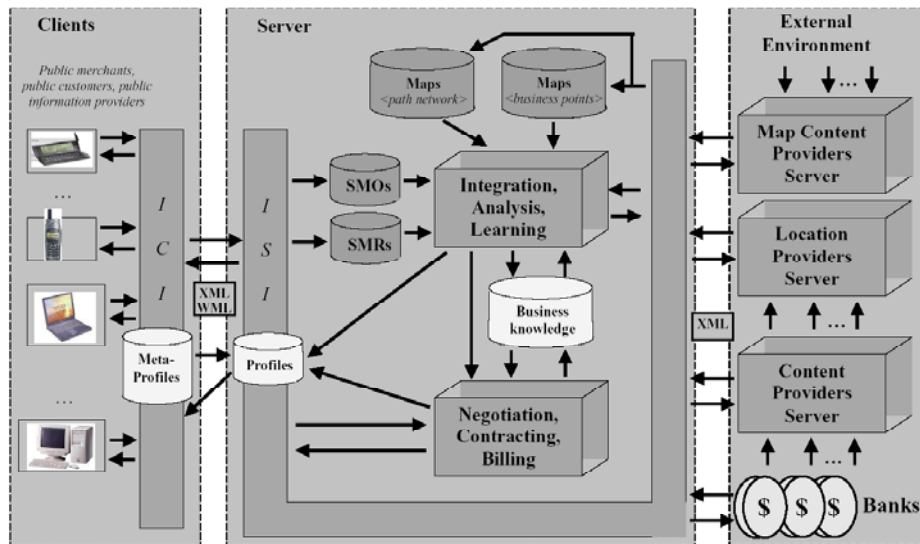


Figure 15. General MMM service architecture (from Vagan Terziyan: “Architecture for Mobile P-Commerce: Multilevel Profiling Framework”, in *Workshop Notes for the IJCAI.01 Workshop on E-business & the Intelligent Web*, August 2001. Paper is available at <http://www.csd.abdn.ac.uk/ebiweb/papers/terziyan.pdf>).

Further details on the MMM project were available at <http://www.cs.jyu.fi/~mmm/> at the time of writing (November 2002). The page also contains links to papers and further documentation on the project results.

III-1.2 WH@M — “the World in Your Hands on the Move”

Project *WH@M* has been a project with the EU-IST Programme (no. IST-1999-20676) carried out during 2000-2002. Further information on the project is available¹⁵ at <http://www.whamproject.com> and <http://www.viatek.fi/tampere/wham/>.

The objective of the project has been to develop a suite of software modules and services capable of collecting information from several sources (either public or private, for tourism, weather, maps, etc.) and capable of disseminating this information via mobile and stationary electronic channels. Customising the information by advanced user profiling techniques is of special interest. There have been three pilot sites, i.e. Athens and the transportation and facilities in the archipelago nearby, the cultural context and facilities in Madrid and the facilities at the Levi Mountain resort in Lapland.

Figure 16 depicts the general architecture of the WH@M solution. As indicated, the solution is based on enhanced web technology, where the material is presented as HTML, XML and WML. An application server works as an interactive mediator between adjacent information sources, booking services and external information sources via a general web-based service interface.

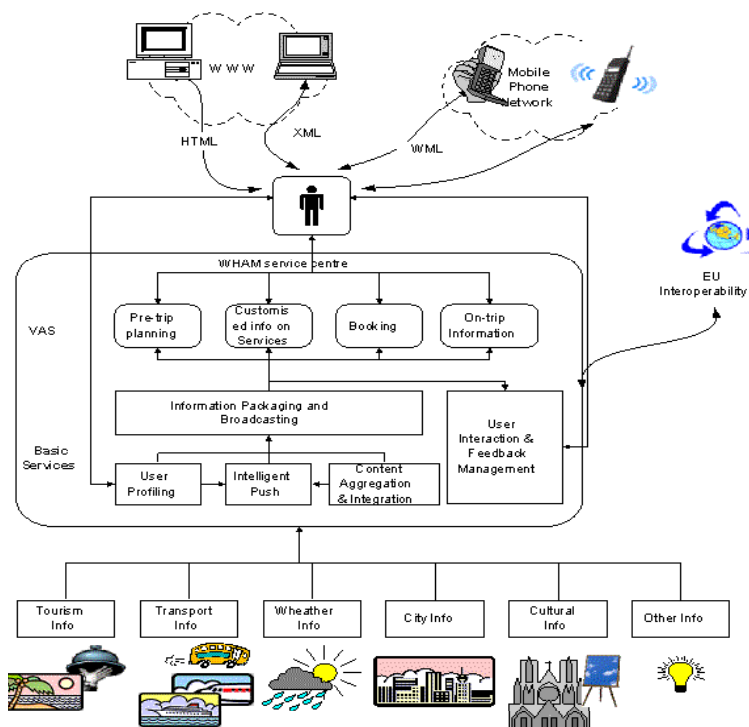


Figure 16. The WH@M systems architecture (from project web page at <http://www.whamproject.com/Approach.htm>).

¹⁵ At the time of writing as of November 2002.

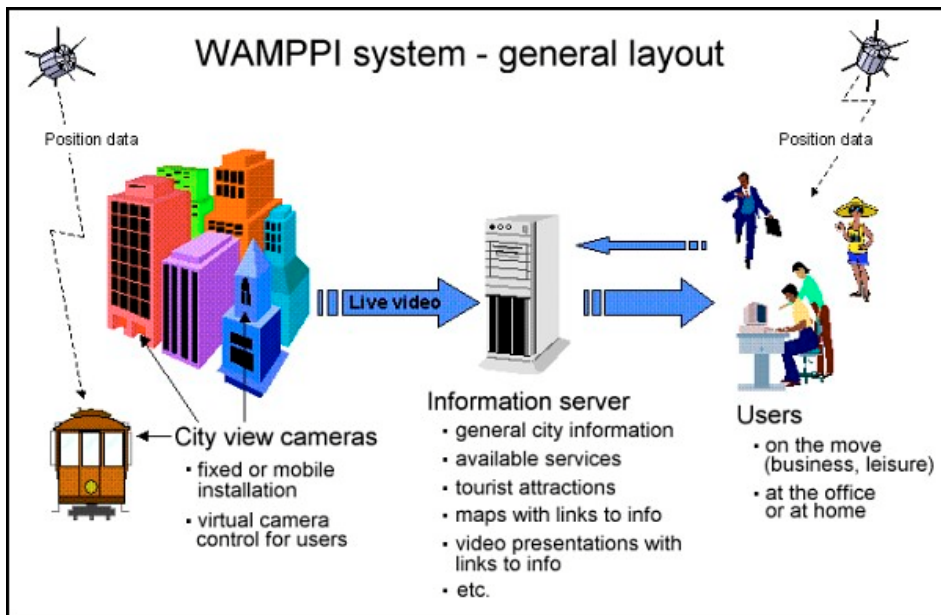


Figure 17. General system layout for the WAMPPI system.

III-1.3 WAMPPI – WAP Multimedia Service Pilot

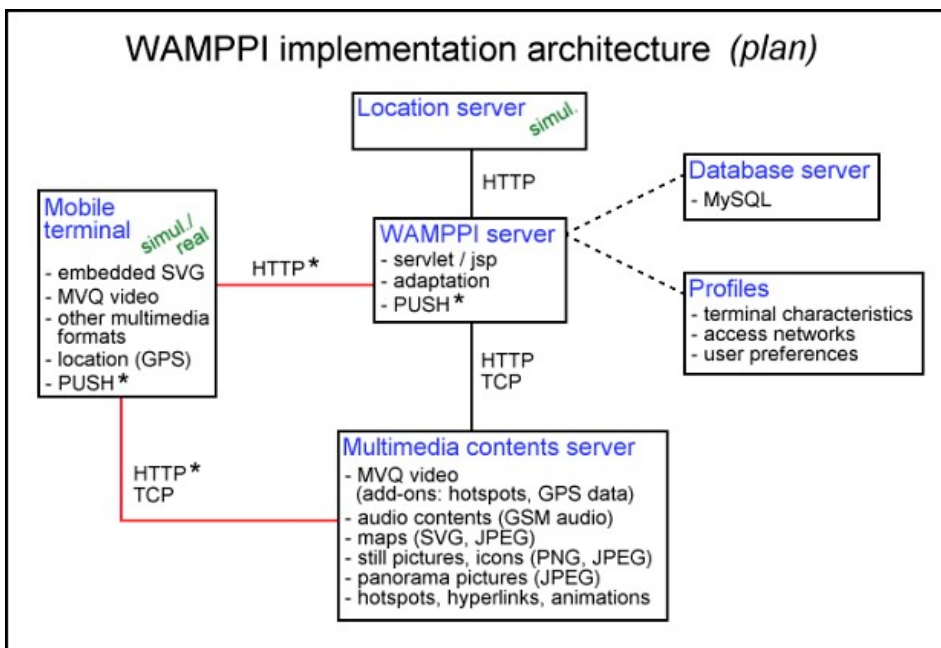


Figure 18. Implementation architecture for the WAMPPI project system.

The target of the WAMPPI project is to demonstrate a platform to support advanced mobile multimedia services. The networking technology is based on 2.5G/3G mobile access networks. The target platform includes tools to facilitate the development of scalable applications for the context. Figure 17 illustrates the general architectural concept, while Figure 18 summarises the service and protocol architecture. Since WAP

protocols are applications of XML, the overall architecture is closely related with the suggested general architectural outlines.

III-1.4 IMAGE – Intelligent Mobility Agent for Complex Geographic Environments

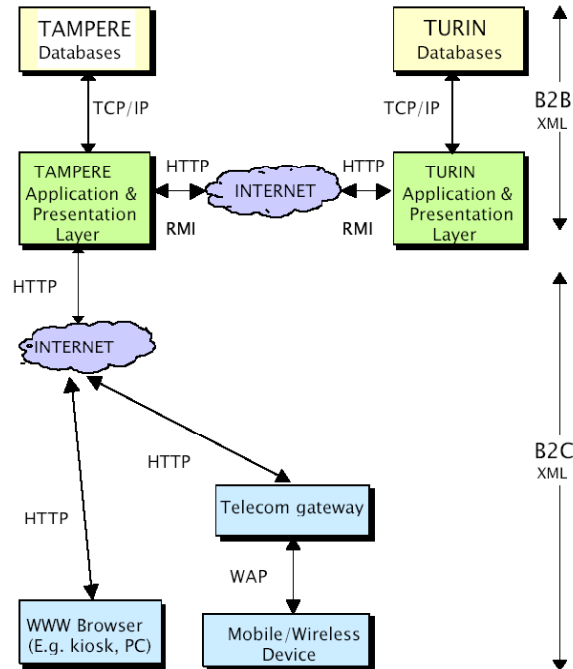


Figure 19. Overview of the *Image* system (from [<http://www.image-project.com/image/freedownload/deliverables/Imaged11v5cbPublicbprint.pdf>]).

The *Image* project is part of the EU-IST Programme (no. IST-2000-30047) started in late 2001 and scheduled to be finished in late 2003. The general architecture is based on web-based transactions, but as the project is in progress, some details are still under refinement. The architectural diagrams in section summarise certain parts, where XML-based protocols are used, while java-based interfaces (like RMI) are suggested for certain special interfaces. The reader should review the links for further details.

The *Image* architecture is an extension of the suggested architecture in the sense that there are also binary interfaces (i.e. Java RMI). According to our general recommendation, such interfaces are suitable for internal use by an organisation. However, in the longer run, pure XML-based interfaces are to be preferred for inter-organisational use, or public services.

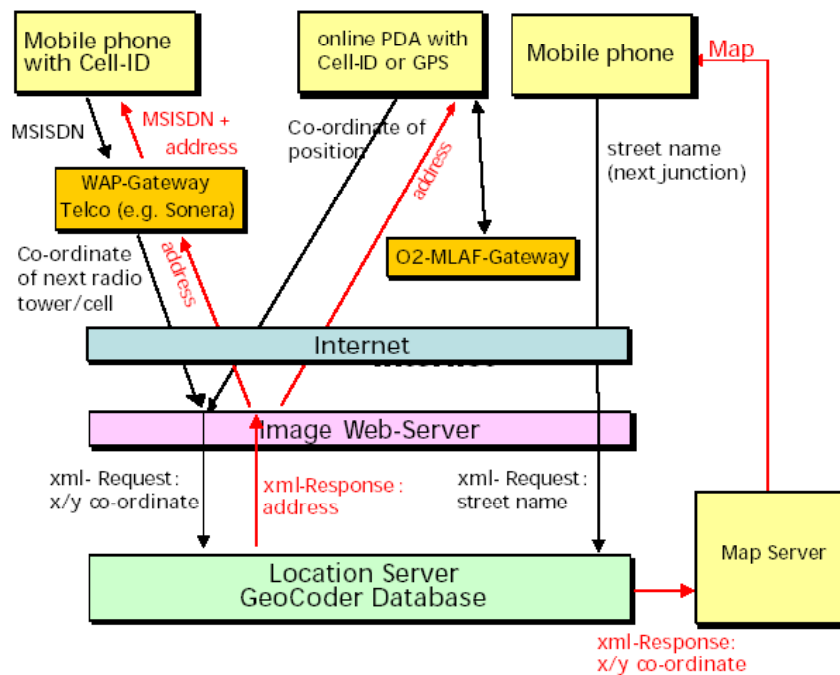


Figure 20. Positioning application with *Image* (from [<http://www.image-project.com/image/freownload/deliverables/Imaged11v5cbPublicbprint.pdf>]).

The general arrangement of the intended *Image demonstration system* is shown in Figure 19. Figure 20 illustrates the positioning architecture for the Image platform.

III-2 Introduction to NAVI service pilots and project goals

The NAVI programme includes four service pilot projects to demonstrate major component services needed for realising generic location-based services. The projects are:

NAVIttestbed – a generic service for locating a mobile terminal (GSM phone) based on LIF Mobile Location Protocol.

NAVImap – a generic service for representing (vector or raster format) digital maps according to OGC's GML specification

NAVIssearch – a generic realisation of a service directory, designed based on the type of services and interfaces provided by NAVIttestbed and NAVImap and preliminary information on the *OpenLS* specification by OGC for location-based services.

III-3 NAVItestbed: A partial realisation of the LIF LBS architecture

III-3.1 Introduction to the pilot service

NAVItestbed is a realisation of a locationing service for mobile handsets. The location information is to be accessed by Internet-related technology¹⁶. The specification of the services is intended to closely follow an earlier version of the LIF location protocol and its associated service architecture.¹⁷ In practice, the NAVItestbed version is a simplified version adapted for the existing GSM-specific commercially available location services in Finland. The locationing services were originally dependent on text messaging and mobile operators in Finland have only gradually introduced their respective services, which has reflected on the architecture.

III-3.2 Basic Service Architecture (apply the modelling principles above)

Figure 21 gives an overview on the NAVIsearch architecture. The essential NAVItestbed services are realised by the host (a computer system, depicted by the box drawn as a 3-d perspective mapping). Location-based services (denoted as applications in the figure) will contact the locationing services via *http*. Location information originates from sources external to the target system, i.e. specific extensions of the core mobile network system, provided by the mobile operator (providing the services for the mobile terminal in question). The system can employ the services of more than one mobile network.

Further information on NAVItestbed can be obtained at the NAVItest www-site at <http://www.vtt.fi/aut/kau/projects/navitest/>.

¹⁶ Typically for public Internet; the technology as such could also be applied for private networks using Internet-based technology.

¹⁷ Since LIF (*Location Interoperability Forum*) has been merged with OMA (Open Mobile Alliance) and the LIF Web site has been brought down, the OMA web site (<http://www.openmobilealliance.org/>) should be checked for possible links to past LIF-related information.

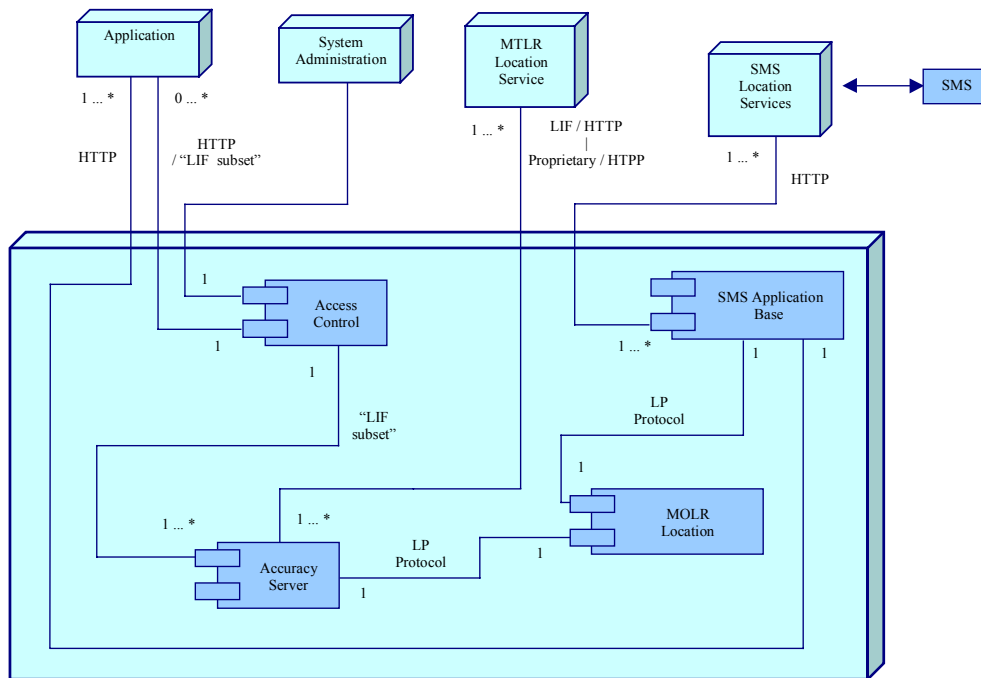


Figure 21. General systems architecture of NAVItestbed is shown above.

III-4 NAVImap (and GIMODIG), Web Features and Web Mapping

III-4.1 Introduction to goals of the pilot demonstration

The display of map data on mobile devices is currently under focus, especially because of the recent introduction of the concept of Location Based Services (LBS). Many useful location-aware services can be built without map display on the mobile terminal and some service categories can be realised even completely without access to spatial data resources. However, in some service types a significant improvement can be achieved in the usability if map data can be utilised. Map displays are most useful for instance in various guidance and navigation services. A fundamental design principle for this kind of service is that map display must be brought to the terminal via a wireless network connection on the moment the service is accessed. Service architecture based on pre-loaded map data does not seem appropriate in the context of dynamic mobile services.

Mobile use of network based map services poses some new challenges for the service providers. Firstly, the map display can, and should, be made context-sensitive and adaptable to the actual usage situation. Once location determination methods get accurate enough, even the position of the user might be taken into account. For this kind of service scenario to be feasible, map services cannot rely on pre-designed, generic, fit-for-all maps. Instead, mobile maps have to be tailored in real-time to adapt them into the needs and preferences of the user. Secondly, mobile maps will be specifically used while moving in an unfamiliar environment. This fact puts high demand on the portability of the service – it's nice to check one's location using a mobile map while on way from home to

office, but much more so when travelling in a foreign country. Use of standardised service interfaces is a key requirement to make flexible service roaming feasible.

The limited computing facilities available in current mobile devices sets significant limitations for the map display. The processing power of the terminals can be assumed to be constantly increasing, but on the other hand in the future even smaller devices will probably be introduced. So limited capacity terminals will be used also in the future. This kind of devices can typically display map data only as raster images or in the form of textual or audible information. An example of the latter would be a navigation service based on audible turning-point instructions. However, the use of vector-formatted map displays on mobile devices is an interesting topic to be investigated, because of the advantages it brings to the application. These advantages include better local interactivity, better support for personalised, ad hoc maps, good linking and animation support etc.

In mobile services up-to-date information is vital. Access to current data, independently of the time of the day or location of the user, is a fundamental requirement in these services. The first operative mobile services are specifically designed for checking the current value of the subject in question, examples ranging from stock prices to news feeds to bank accounts. When developing location based services, the same demand for up-to-date information concerns naturally also map data. This demand is especially important in the case of road information used in operative car navigation services. If the introduced roadblock does not instantly show up in the road database, the usefulness of the navigation service is questionable.

As a part of the activities of the NAVI programme, a testing environment is being set up to foster a common approach to certain essential protocols and interfaces in the service architecture for personal navigation in the national level. The Finnish Geodetic Institute has successfully proposed to develop a spatial data and map service, NAVImap, to be used inside the NAVI programme in tests related to various location-aware services that support personal navigation. The main idea presented in the proposal is to apply available international standards as widely as possible in the service to be built.

The main goals of the NAVImap project might be summarised as responses to above raised points. The need to provide an up-to-date map to the mobile user necessitates the provision of network based map services that can be accessed using over-the-air interfaces. By supporting fundamental Web technologies, like http-protocol, XML-encoding (GML, SVG, XHTML) and common Web image formats (JPEG, PNG), the NAVImap service is easily reachable from the Mobile Internet platform. The idea in NAVImap service is to retrieve data from a maintained geospatial database on the moment the request is made. Use of vector-formatted map information in the service provides an opportunity for various developers to test the emerging XML-based vector graphics encoding in their client applications. The adherence to the standardised access interfaces improves the usability and portability of the NAVImap service.

The framework considered above is also being extended to an international and mobile usage context with project GIMODIG. The partners with this project are from Finland, Germany, Denmark and Sweden. The goal is to support both WWW-like usage with conventional large-screen feature-laden browsers (as with NAVImap) and more recent shrink-wrapped terminals designed for mobile use. Further information on GIMODIG can be found at the project web site at <http://gimodig.fgi.fi/>, cf. Figure 22.



Figure 22. Project web site for GIMODIG at <http://gimodig.fgi.fi/>.

III-4.2 Basic Service Architecture

The main functional components in the NAVImap service can be listed as follows.

1. Provision of the XML Schema-based data model, as a GML 2.0 Application Schema, for the spatial datasets being introduced to the service,
2. Live connection to a database containing the selected datasets and provision of a GML 2.0 -compliant data service from that database,
3. A minimal server implementation of the OGC's Web Feature Service specification (Basic WFS), providing access to the datasets in the database,
4. A server implementation consistent with the OGC's Web Map Service interface specification, serving the same datasets in visual form,
5. The resulting map displays provided in the form of an SVG vector image,
6. For limited capacity devices the same displays available also as PNG and JPEG raster images.

In the following each of these components is discussed in some detail.

III-4.3 XML Schema-based Data Model as a GML Application Schema

The GML specification states a set of rules about how local Application Schemas are to be derived from the GML base constructs using the inheritance mechanism present in the XML Schema. This kind of data model has been designed for the data sets selected into the NAVImap service. The data model contains of a two-level container hierarchy

(FeatureCollection), in which the outmost element is called NAVImap. On the second level of the hierarchy there is a container element for each of the organisations providing data for the service (currently NAVIdigiroad, NAVInls, NAVIhelsinki). Under these FeatureCollections are the real GML Features (currently 9 different Feature types present).

The XML Schema document is read into a DOM structure when the service is started up. When a request is received through the DescribeFeatureType –interface of the WFS, an appropriate part of the schema is retrieved and returned by carrying out a proper XSLT transformation on the document. The XML tools employed in the NAVImap include free software products from Apache.org, like Xerces XML Parser, Xalan XSLT Processor and Batik SVG library. The Servlet Engine by which the services are run is a product called Tomcat, and as a Web server the ubiquitous Apache server is used. The operating system platform is Linux.

III-4.4 An Online Geospatial Database

The NAVImap service is based on free software components. As a database system, PostgreSQL relational database with its PostGIS spatial extension is used. Communication from the WFS implementation to the database is established via a JDBC driver. PostGIS stores spatial data in the database in a way compliant with the OGC's Simple Features specification. Spatial indexing is supported by an R-Tree-derived mechanism called Generalised Search Tree (GiST).

III-4.5 An Implementation of the WFS Specification

In the case of the NAVImap service, the WFS implementation is based on in-house software development at the FGI, and is coded as a Java Servlet. The NAVImap WFS implementation is a minimal realisation of the Basic WFS, i.e. the read-only version of the service. The WFS interfaces supported include GetCapabilities, DescribeFeatureType and GetFeature. Only the simplest form of the GetFeature spatial query constraint, a rectangle, is supported. Property constraints can be set only against a literal value. Complicated logical operations are not supported. The data retrieved from the database is written out as GML constructs. The transformation is simple, because both incoming and outgoing data streams comply with the same data model, namely Simple Features.

III-4.6 A WMS Implementation

Also the WMS module is a Java Servlet, programmed at the FGI. The server implements the WMS interfaces GetCapabilities and GetMap. Currently there is no support for other co-ordinate systems than the national Finnish reference systems, YKJ and KKJ. In addition to vector-based data sets stored in the NAVImap database, a raster background map can be requested and is retrieved from the MapSite of the National Land Survey (NLS) during the query processing. Conceptually WMS could be seen as a front-end service that would make use of some WFS resources in the background, but in the case of NAVImap these services are parallel, not sequential. The reason for this set-up is the performance gain that can be achieved by accessing the database directly from the WMS, without the help of a WFS server.

III-4.7 SVG Based Map Display

The NAVImap WMS server allows the resulting map to be encoded as an SVG image. Because of the differences in the data model the transformation from the internal Simple Features schema into the SVG constructs requires more computing effort than in the case of the GML encoding. The SVG image is created as a flat file in the local hard disc of the server and only a URL address pointing to this file is sent to the requesting client. Streaming SVG code directly from server to the client was also tested, but the approach did not seem to work over low capacity modem lines.

III-4.8 Raster Image Support

The WMS server also supports map delivery as raster images. In this case the server builds an SVG image from the data received from the database. This image is not written to a disc file, but constructed as a DOM tree. The SVG DOM structure is then processed by the Batik SVG library to produce the raster images either in JPEG or PNG format.

In the following illustration the internal schema of the NAVImap service is depicted. The functionality related to GetCapabilities query is identical in both services, WMS and WFS, but is shown only for the WMS server.

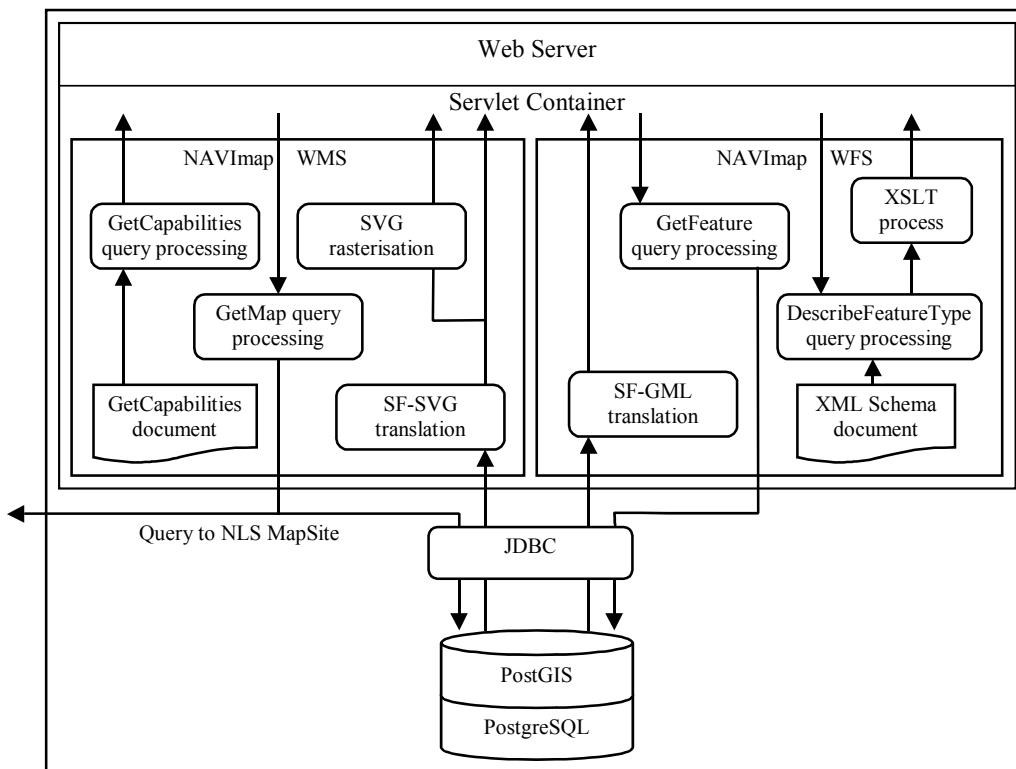


Figure 2.1. Internal structure of the NAVImap service.

III-4.9 Discussion

The NAVImap project was started at the FGI in the beginning of the October 2001. The first version of service to be developed became online at the beginning

of the year 2002. For now the service is restricted to the members of the NAVI programme only. The initial response from the NAVI membership has been encouraging. Various NAVI member organisations have shown a genuine interest in making use of the service to learn about and experiment with the applied standards. So far 25 organisations have registered as NAVImap users.

As the general network infrastructure is gradually evolving to support the delivery of graphic information over the air interface to a mobile device, and the capacity of the high-end mobile terminals is to soon enable vector based processing of map data, the introduction of the NAVImap –like test service is seen as being timely indeed. Together with the two other NAVI test platform services - NAVItestbed providing location information through the access interface of the Location Interoperability Forum (LIF) and NAVIsearch serving business locations via the Directory Service interface defined by the Open Location Services (OpenLS) initiative of the OGC - the NAVImap service forms an interesting testing environment for various location-aware applications.

III-5 NAVI Search: Integration of Positioning, Map Services and OpenLS service query

III-5.1 Introduction to goals of the pilot service demonstration

The service pilot demonstration projects with the NAVI programme have been set up in order to provide a platform to support a few generic key services for experimentation with and evaluation of more specialised location-based service developments within the programme and the associated network of interest group. These services, i.e. network locationing services, map distribution services and service directory services based on emerging international standards have not been available at the time of the critical project work.

Under these circumstances, the pilot services are necessarily simplified versions of the target standards being developed by international consortia, somewhat experimental by nature and necessarily frozen to certain preliminary stage of the standard. While more complete realisations of the services are most likely being developed by a number of companies, such products are not expected to be available before or at the time of completion of the respective standards.

III-5.2 Basic Service Architecture (apply the modelling principles above)

The diagram in Figure 23 illustrates the first demonstration architecture for NAVIsearch presented at LIF meeting in Helsinki in May 2002. This was early demonstration architecture with high emphasis on network-based positioning as part of the system (i.e. using the *NAVItestbed Platform*, cf. Section III-2).

The target demonstration architecture has been under development until the last few months of the NAVI Programme. Figure 25 illustrates the current view of the target architecture with extended functionality.

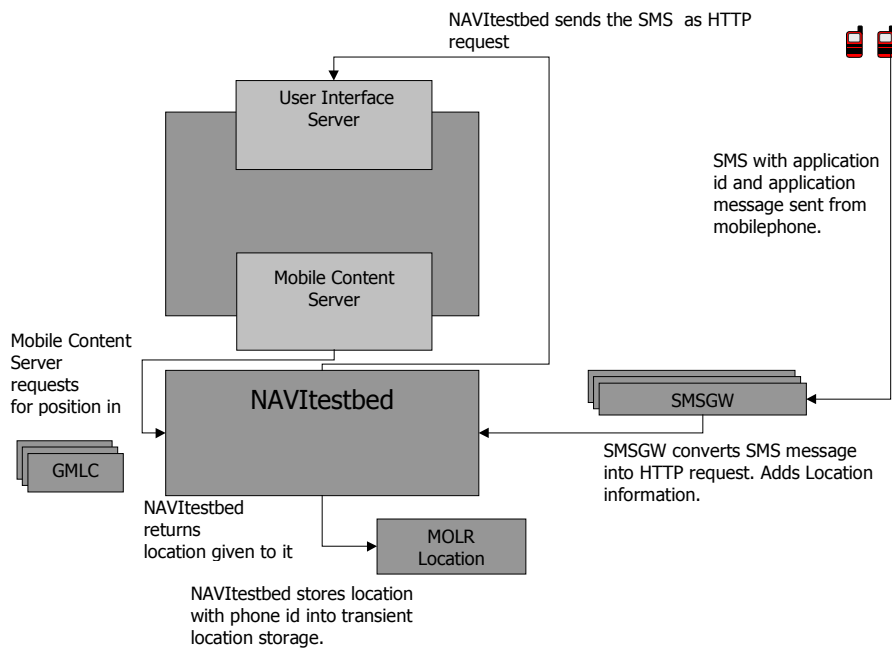


Figure 23. The first demonstration architecture for NAVIsearch (Demonstration presented at LIF Meeting, in Helsinki on 15. May 2002; Courtesy of Locus Portal Inc.).

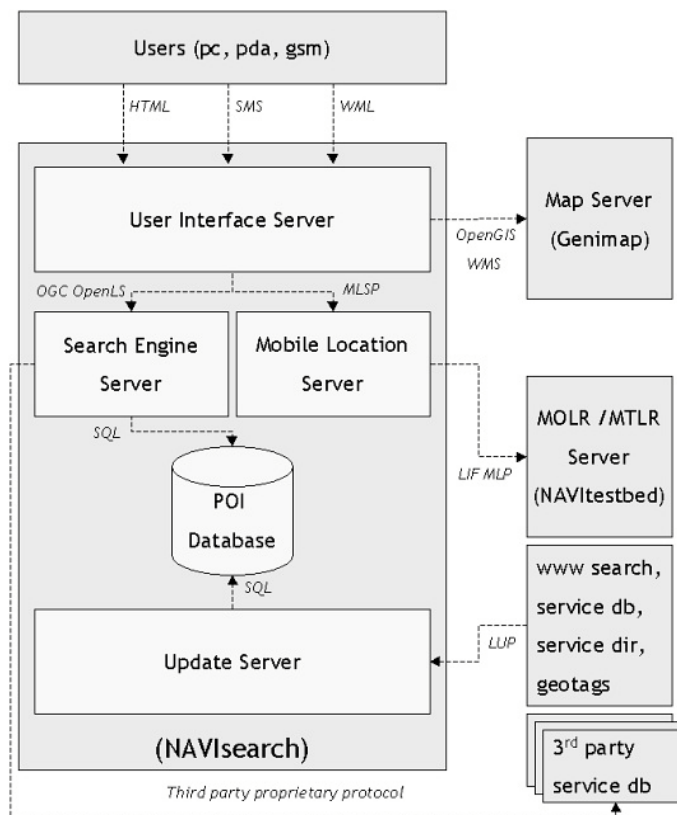


Figure 24. An Overview of NAVIsearch services (From <http://proxnet.vtt.fi/navi/search/rakennekaavio.html>).

IV PART IV – Tutorial Material on UML and Case studies

IV-1 Architectural Modelling of Personal Navigation Services

IV-1.1 Purpose and goals of architectural modelling

The term *architectural modelling* is associated with designing a set of specifications based on an adopted formalism to represent an abstract, but sufficiently accurate functional, logical, structural and even physical description of the target system. The resulting model could e.g. serve the following purposes, i.e.

- serving as part of general documentation of target topics
- serving as a platform and tool for more detailed analysis (and description of) critical parts of the overall system in further depth
- serving as starting point for the design of an information system leading to implementation and realisation of the target system

The latter application is the most typical context for applying an integrated modelling and design tool (typically applied, when complex systems are to be designed and realised as part of an integrated construction effort). This is not the case in our current project context, since the primary goal is not to design and realise a specific personal navigation service or application, but to design a set of principal architectural component specifications to be used separately in other development projects with immediate realisation-oriented goals. Thus, we must concentrate on topics with more fundamental significance, whereas “glue-like” standard parts created by *ad hoc*-principles will serve as an example, at best, and just add confusion, at worst. On the other hand, any modelling effort targeting at realisation has to fill in such portions of detail, as well!

There are two main alternatives to gain information for this model inference task, i.e.

1) The “bottom-up” approach, where the architecture can be viewed as a *compressed catalogue* of components and their external (or interface) characteristics, including the rules for combination of components, out of which the target realisations can be built of, and

2) The “top-down” approach, which concentrates on acquisition of information on the needs, usage specifications, constraints for the design and construction work for this specific application of architectural design.

Both of these approaches are necessary and supplementary aspects of the architectural specification process. The “bottom-up” approach deals with existing and available components, but is unable to cope with overlapping or superfluous components or components families as alternative choices are typically available in real-life situations. On the other hand, the “top-down” approach needs to be anchored to realisable building

blocks and essentially needs a proof of its concept by being decomposed into catalogue parts with mutually compatible interfaces.

Finally, similar semantic aspects can also be associated with the original meaning of the term *architecture*, i.e. the rich context of design of physical buildings. For instance, the *Architecture of Paris*, or the *Architecture of Alvar Aalto* are related with the bottom-up concept, i.e. collection of information on a large number of building component detail and broader-line concepts for the target concepts. On the other hand, you need an *architect* (a professional in *architecture*) if you are in need of a design for a building intended to be constructed, related with the “top-down” approach. Also observe that the “bottom-up” work can often be done without drawings or abstract design modelling machinery whereas pictures of the target architecture are the most essential components documents dealing with topic. On the other hand, and architect is unable to finish his or her work without a formal modelling gadgetry.

IV-1.2 Overview of Methodology

Architectural modelling is based on UML (Universal Modelling Language, see <http://www.omg.org/> or directly at <http://www.omg.org/technology/uml/index.htm>). As UML is very general (and complex) tool with many alternative (and often overlapping) ways to formulate the problem, a modelling tool has to be used as a practical device. We have been using Rational Rose (<http://www.rational.com/index.jsp>) as the modelling tool.

A general approach to architectural modelling is to start by *use case modelling*, which essentially attempts to specify and describe the functionality of target problem. Use case modelling will partition the bigger target problem into a set of simpler sub-problems as well as more generic super solutions that can be designed to effectively deal with a class of standard problems. The atomary use cases are called *activities*, whose causal relations can be specified in terms of *activity diagrams*.

The next step is to design *usage scenarios*, i.e. designing a logical, programmatic solution to basic personal navigation services. The basic functionality identified at the former step needs further support for machine-based realisation. The most essential new machinery at this stage will comprise of service objects that can realise memorisation functionality, i.e. reserve and release data storage as governed by functionality.

IV-1.3 A Short Guide to UML Use Case Notation

The following *Use Case Diagram* (Figure 26 and Figure 27) is an example of a simple use case model, i.e. a set of use cases and their directed (communications initiative) relations. One of the use cases, i.e. “Switching on Coffee Maker and Wait” is further described an Activity Diagram. The Example Activity diagram has a collection of most typical device symbols used in this context.

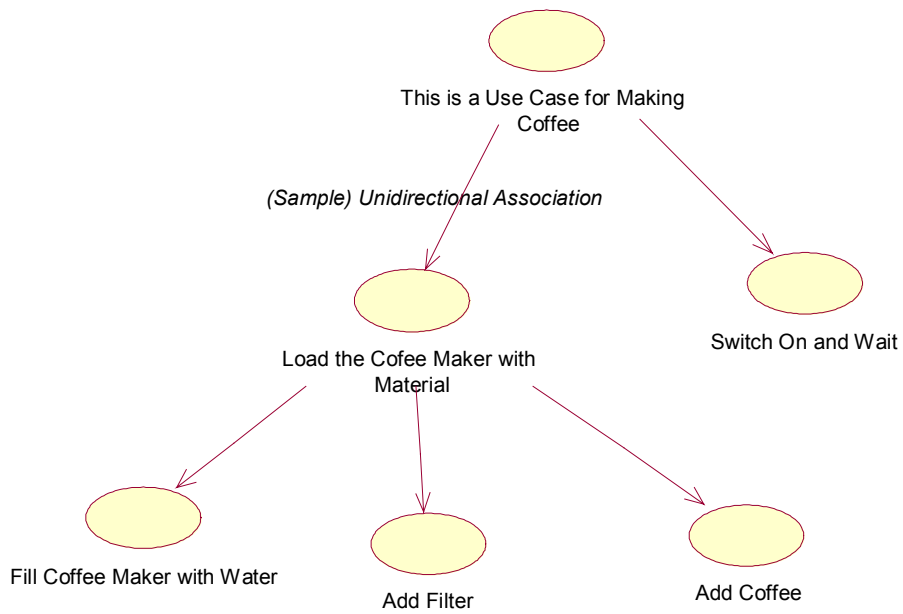


Figure 26 A simple example on Use Case Modelling: “making coffee”.

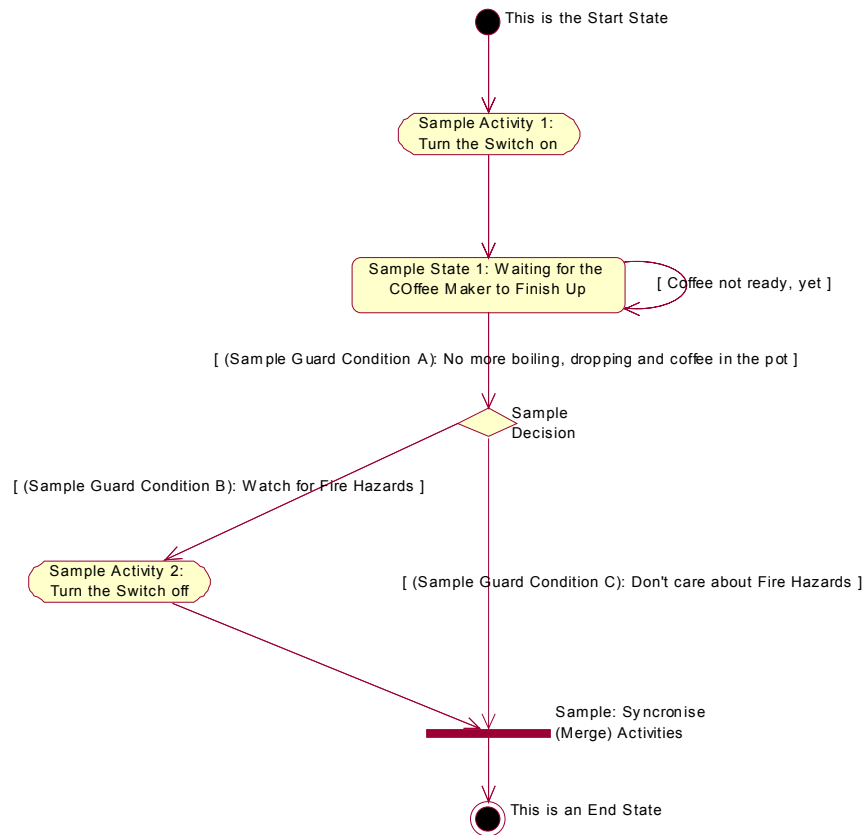


Figure 27. Activity diagram for “switch on wait” (i.e. actually making coffee).

The names have been chosen to illustrate the semantic meaning of each device.

The following links lead to further information on UML related with OMG (Object Management Group) Activities, i.e. <http://www.omg.org/uml/> or <http://www.uml.org/> (both have general information). Course-like material for personal education (free for personal, non-commercial use) can be found at <http://www.celigent.com/omg/umlrtf/tutorials.htm> (leading back to OMG as a set of transparencies, e.g. <http://cgi.omg.org/cgi-bin/doc?omg/2001-03-02/>, <http://cgi.omg.org/cgi-bin/doc?omg/2001-03-03> and <http://cgi.omg.org/cgi-bin/doc?omg/2001-03-04>).

IV-1.4 Components sources for bottom-up architecture

The possible sources for architectural components of the bottom-up approach are the following, i.e.

- 1) Existing or emerging standards,
- 2) Standard-like specifications from industrial consortiums,
- 3) Existing commercial products with wide acceptance, i.e. de facto standards, and
- 4) Other existing or emerging realisations with engineering-like technological or economical merit for “compact” sub-problems

The fourth context may seem more vague than the rest, but this type of components have often proved to survive in real-life situations, i.e. sound engineering principles prefer components designed

- i) For a controlled application in a known environment,
- ii) With wide and generic applicability in similar application contexts and purposes,
- iii) With technological simplicity, support of maximally modular design and economy in mind,
- iv) Simple and reliable interfaces with other modules, and
- v) Overall design, which will support quality control, service and maintenance as well as general reliability of the compound system by various, well understood and proven means.

It is out of the scope to extend the analysis to speculative component technologies. On the other hand, alternative or overlapping component architectures can sometimes be organised into alternative solution component families by purely bottom-up analysis.

IV-1.5 Methodology for top-down architectural analysis

This is essentially a mathematics-like, problem, i.e. the task is to characterise the properties of a family of components that can be combined with certain rules, i.e. certain properties of their mutual interfaces as well as some other constraints, i.e. measures for system complexity (including cost and performance).

Unfortunately, it is seldom possible to find mathematics-like rigorous rules for architectural measures. Instead, it is typically only possible to do calculus-like case-by-case analysis of the context. Architectural modelling by UML and associated tools, e.g. an applications-oriented Use-Case analysis targeting at a specification of model realisation (either on a logical or on a further specialised domain) is essentially a practical realisation of such a calculation-like empirical process.

IV-1.6 Current Status of Architectural Modelling

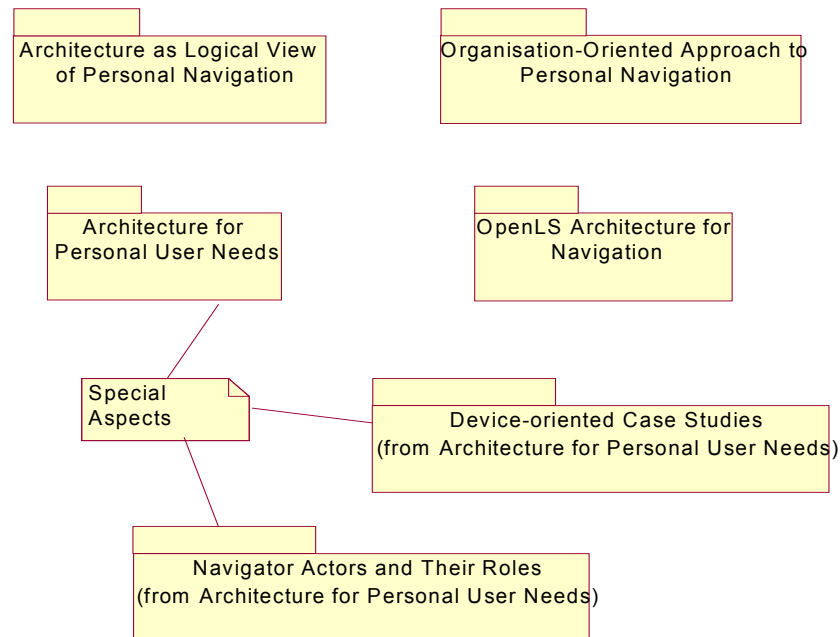


Figure 28 Top-level view of main aspects of architectural modelling.

Considerable rethinking of the role of architectural (top-down) modelling activity with the project has been underway. This state of the affairs has also suggested considerable reorganisation of the model especially at the topmost level. Since the resulting work has not been finished, yet (expected to be finished by the end of year 2001), a detailed review of the model does not serve a useful purpose and might be confusing. Therefore, we will only consider the current architectural model at the topmost level. Figure 28 describes the topmost level of the current model (extracted from http://proxnet.vtt.fi/navi/pam/arkkitehtuurimalli/kehitysversiot/arch_2001-10-31B/arch_2001-10-31B.html).

The current view is to consider architecture from four main points of view:

The first aspect is to consider the logic and functionality of personal navigation. This approach is essentially the one targeting at the definition of the personal navigation core services on a logical level, which was chosen as the starting point for modelling in spring 2001. It has become apparent that it might be difficult to find concrete applications for such an approach. Consequently, it is felt that there is not currently considerable interest for pursuing this approach within the network of interest, even though the approach will serve important documentation purposes, it was concluded that further work in this area should be carried out with lowered priority.

The second aspect is to consider personal navigation from the end user perspective. This is, in fact, the most traditional situation for applying top-down system modelling technology as an integral part of the systems or service and application software design process. However, since our current project is not directly involved in systems design, true application of this area of modelling is not fruitful unless some other project group (within the NAVI network) deeply interested in the process and the tentative results can be identified. Such a project group must also be willing to invest human resources on the discussion related with the modelling work. A couple of exploratory designs have been considered based on reverse engineering-like approach, but such exercises are not seen to serve a truly useful purpose as of now.

The third aspect is to consider personal navigation from an organisational point of view. In essence, we will take the organisational architecture more or less for granted and concentrate on the technological solution within this basic architectural framework, only. The motivation for sticking to the organisational architecture more or less rigidly can be based on various aspects. Of course, there could be established organisations, legislation or such to govern the business context, but it might also be useful to do some modelling for systematic analysis of various consequences of specific organisational scenarios.

The fourth aspect is to use architectural modelling for illustrating existing or emerging navigation architectures. In this case, the modelling approach is purely illustrative or documentary. The principal architecture is already given and the only degree of freedom is to develop specific extensions or design guidelines for special cases where appropriate. The *OpenLS* architectural case serves the latter purpose. Since the OpenLS Consortium is still working on its specifications and has only published preliminary documentation, the work needs to wait until further documentation becomes available. This is expected to happen by the end of Q1/2002.

The current state of the architectural model can be viewed at <http://navi-ohjelma.vtt.fi/navi/pam/arkkitehtuurimalli/kehitysversiot/>. The folder contains subdirectories of the form ".../arch_200X-XX-XX", where "200X-XX-XX" denotes the release date. Double clicking "root.html" in the corresponding folder will start viewing the model on a Java and JavaScript-enabled www browser (there is also a unique name like "arch_2001-06-12.html" in each folder).

IV-2 Example: UML-based modelling of “*Paikannussanasto*” – the vocabulary of navigation & locationing technology in Finnish

The section contains an example on UML modelling for use cases based on “*Paikannussanasto*” – the vocabulary of navigation & locationing technology in Finnish. This publication was produced within a dedicated project of the *NAVI Programme* by *Tekniikan sanastokeskus*. A copy may be obtained for reference as a paper-based publication from the publisher (cf. <http://www.tsk.fi/>) or electronically as a PDF-document at <http://www.tsk.fi/info/paikannussanasto.pdf>. The UML-based diagrams in Figure 29... Figure 38 illustrate the model. An electronic version of the model (including the terms in the vocabulary) is available for the members of “*NAVI-verkosto*” at http://proxnet.vtt.fi/navi/pam/paikannussanasto_UML/paikannussanasto.html (Within the NAVI-project www-pages project internal page set).

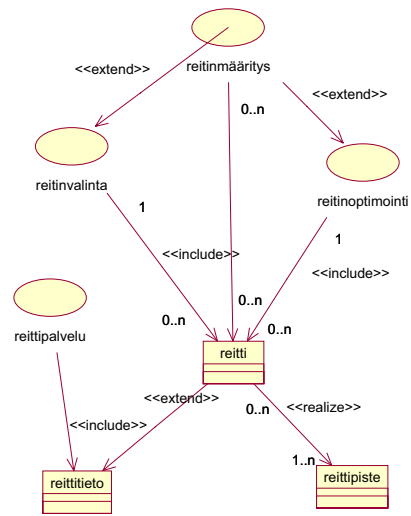


Figure 30. Route ("Reitti").

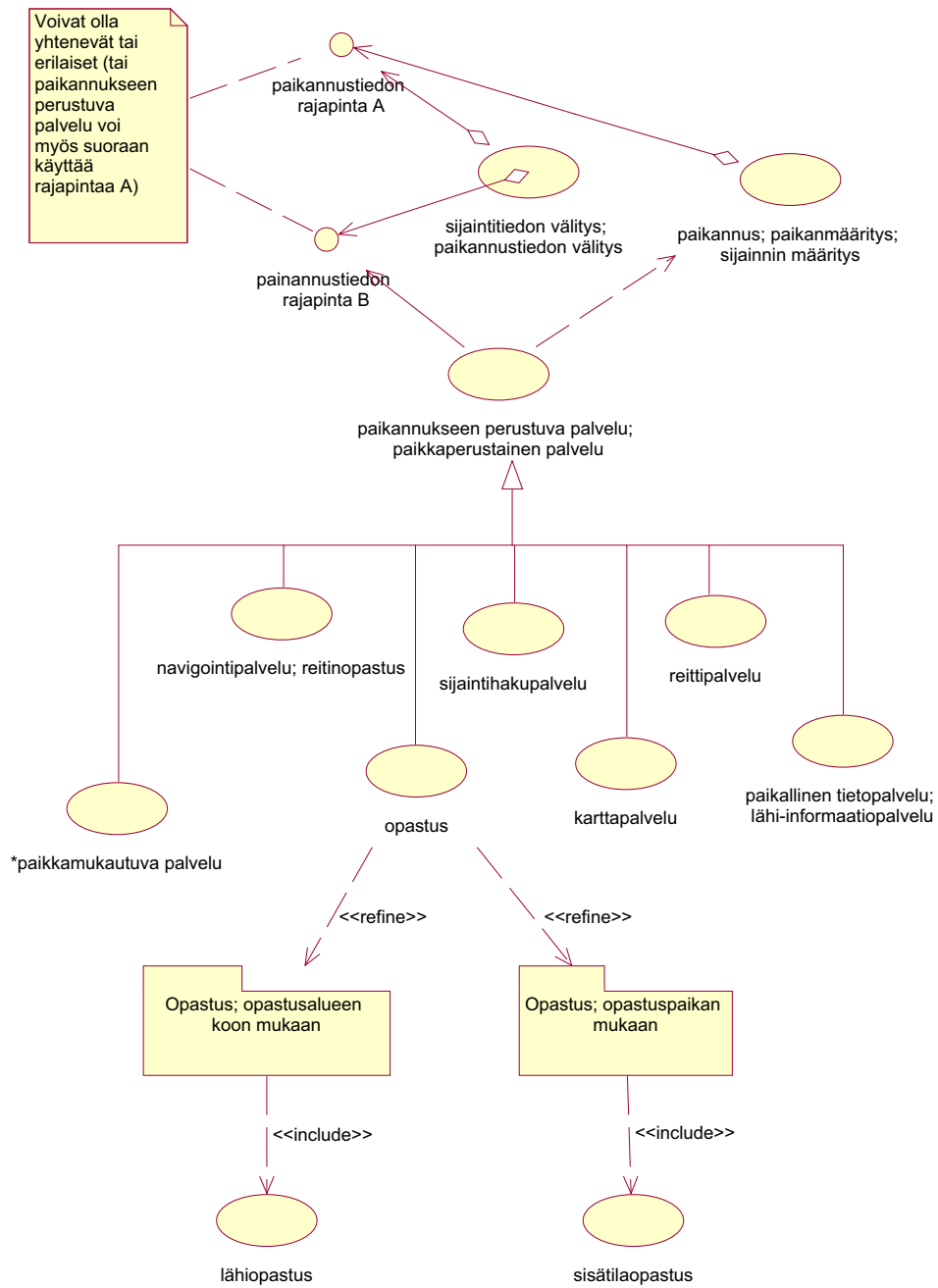


Figure 31. Location-based services ("Paikannukseen perustuvat palvelut").

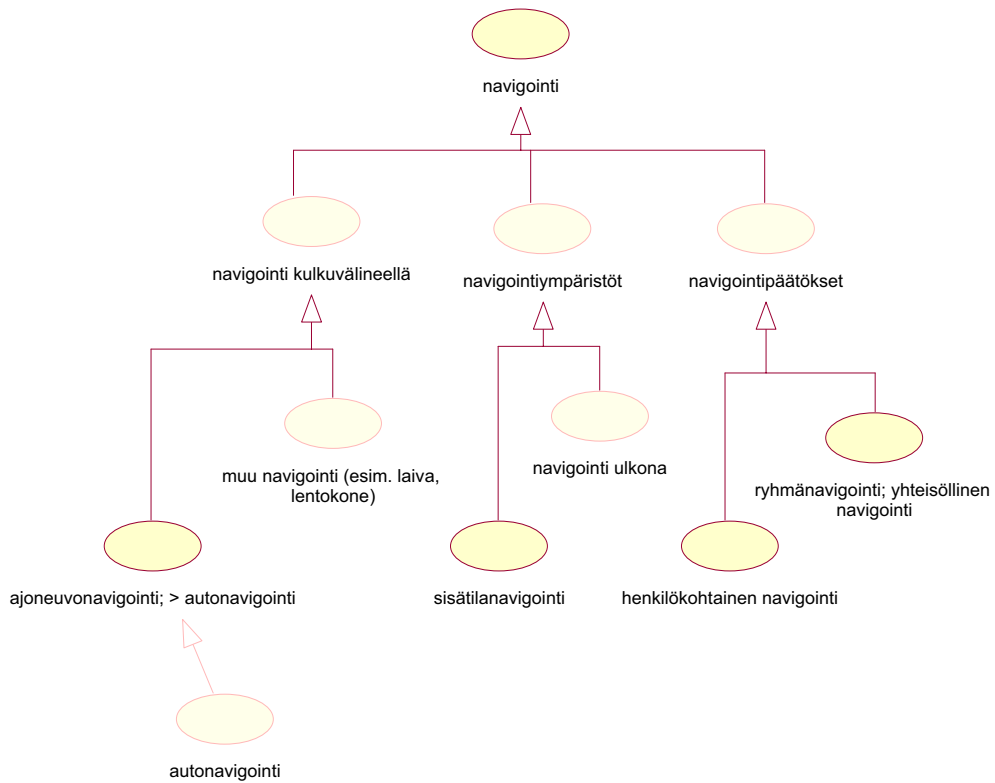


Figure 32. Navigation ("Navigointi").

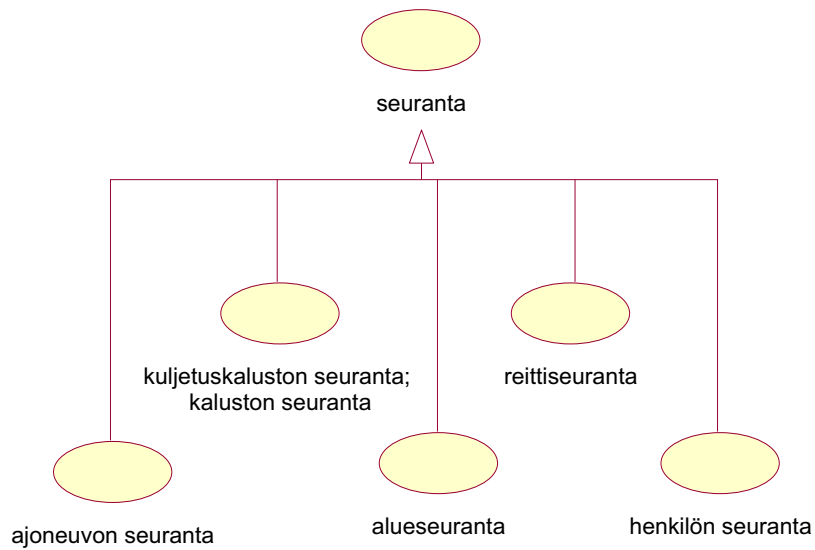


Figure 33. Tracking ("Seuranta").

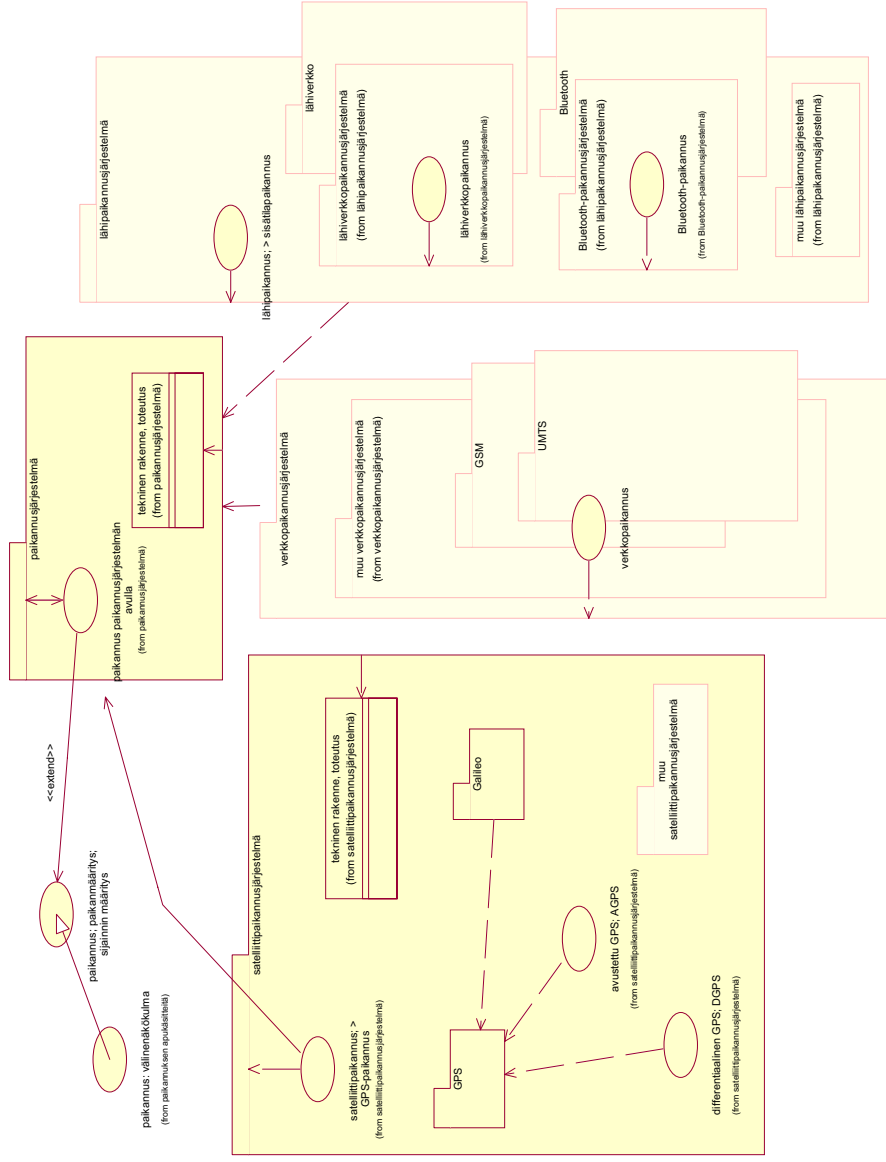


Figure 34. Positioning systems ("Paikannusjärjestelmät").

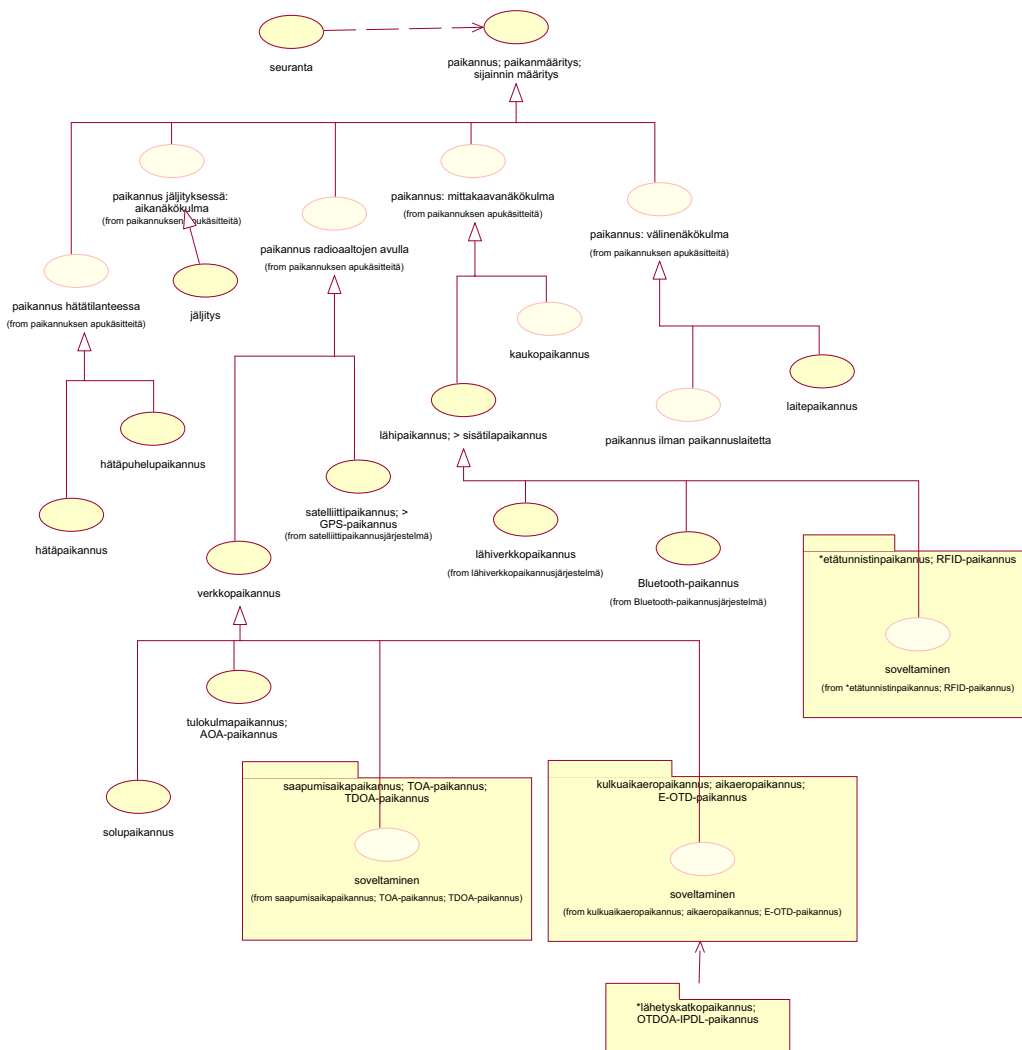


Figure 35. Positioning technologies ("Paikannusmenetelmät").

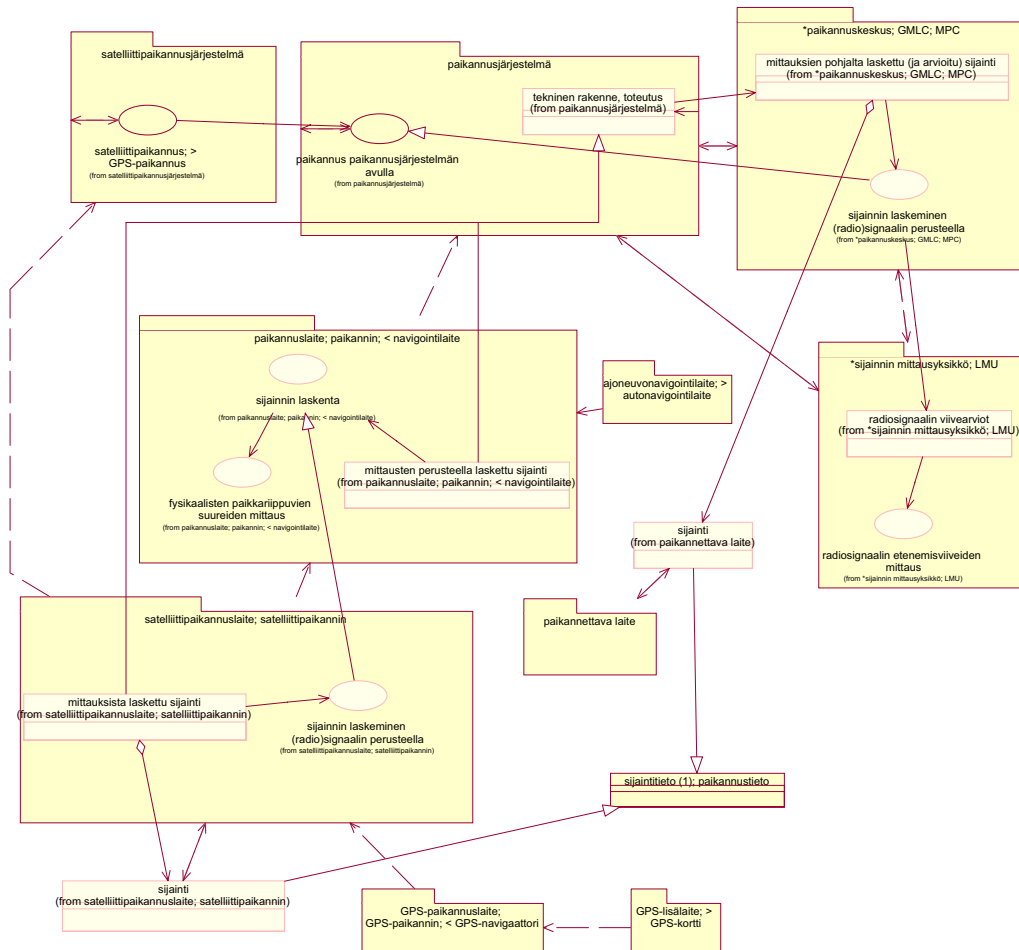


Figure 36. Positioning devices and equipment ("Paikannukseen liittyvät laitteet ja laitteistot").

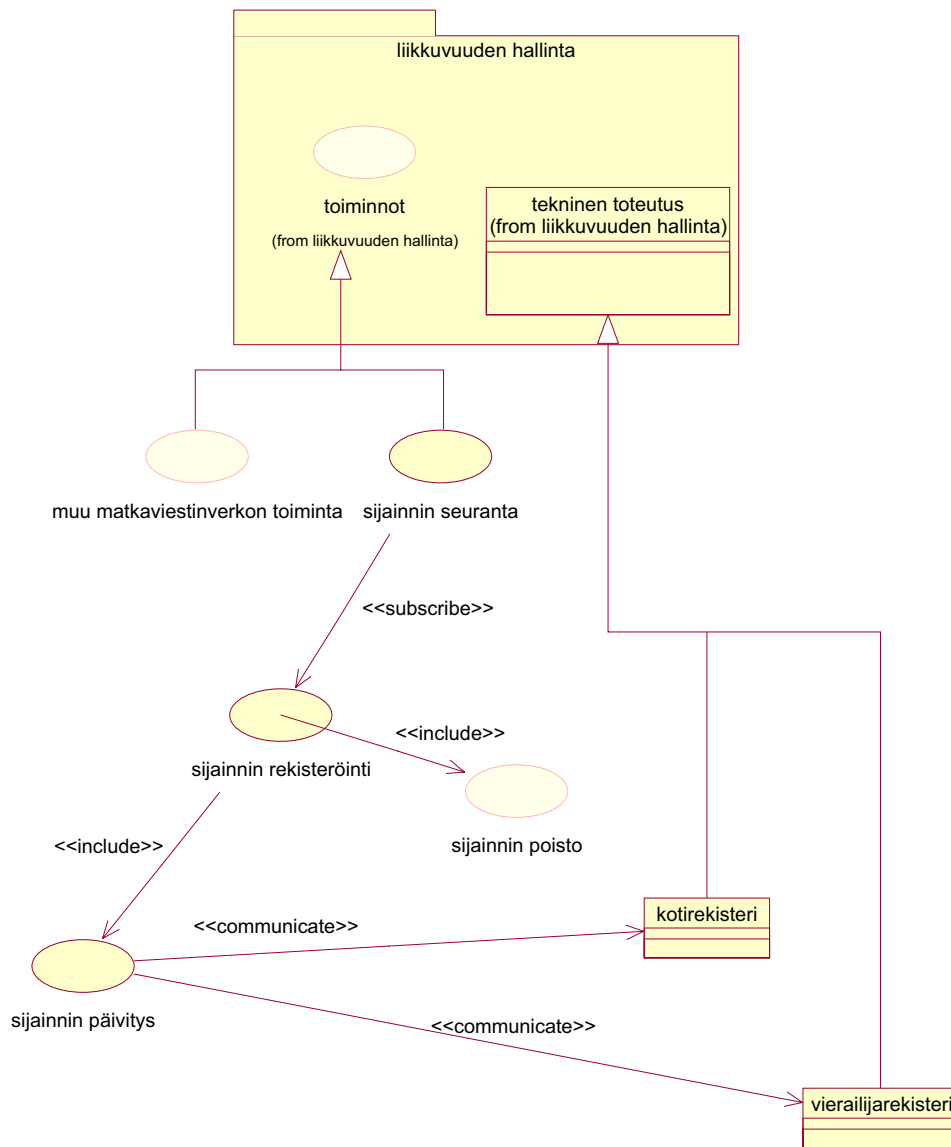


Figure 37. Roaming control ("Liikkuvuuden hallinta").

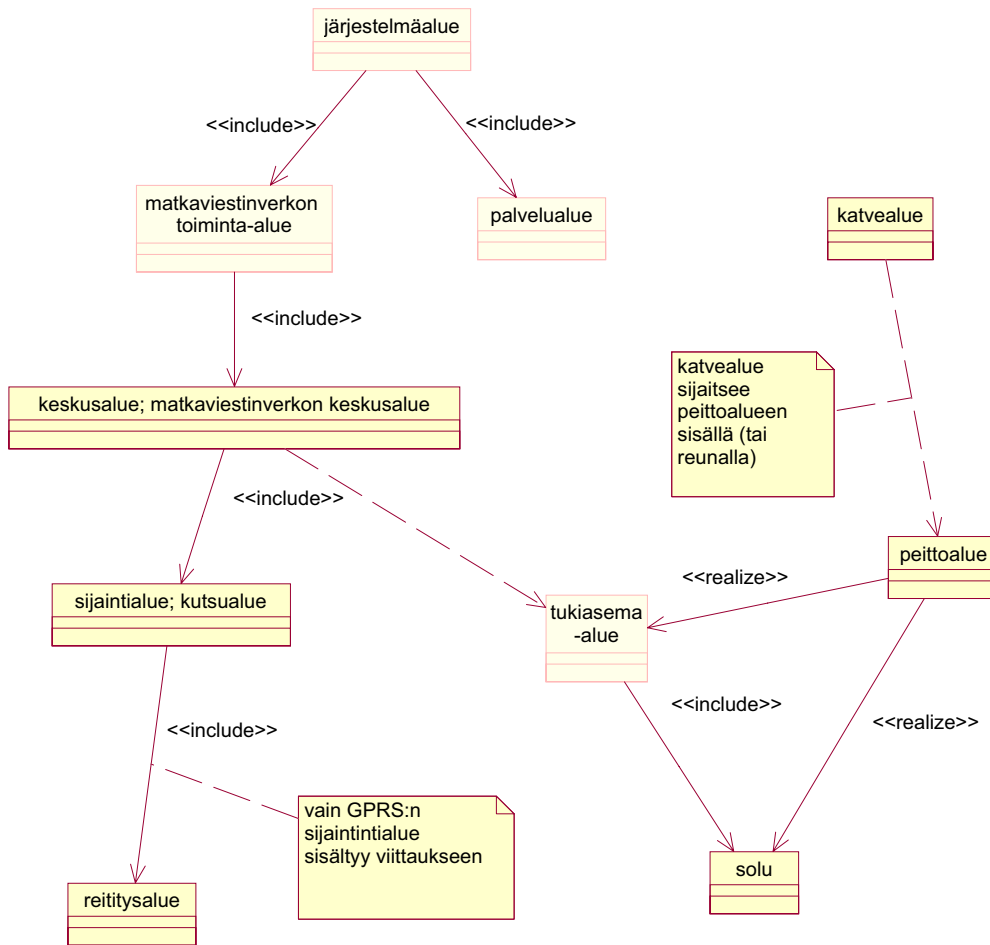


Figure 38. Regions ("Alueet").