INTERNATIONAL STANDARD

ISO 19119

Second edition
2016-01-15

Geographic information — Services

Information géographique — Services

Reference number
ISO 19119:2016(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information.

The committee responsible for this document is ISO/TC 211, *Geographic information/Geomatics*.

This second edition cancels and replaces the first edition (ISO 19119:2005), which has been technically revised. It also incorporates the Amendment ISO 19119:2005/Amd 1:2008.

Introduction

The widespread application of computers and use of geographic information systems (GIS) have led to the increased analysis of geographic data within multiple disciplines. Based on advances in information technology, society's reliance on such data are growing. Geographic datasets are increasingly being shared, exchanged, and used for purposes other than their producers' intended ones. GIS, remote sensing, automated mapping and facilities management (AM/FM), Spatial Data Infrastructure (SDI), traffic analysis, geopositioning systems, and other technologies for Geographic Information (GI) are entering a period of radical integration.

This International Standard provides a framework for platform neutral and platform specific specification of services that can enable users to access, process and manage geographic data from a variety of sources, potentially for various distributed computing platforms (DCPs).

— "a framework for platform neutral and platform specific specification of services" means that this International Standard provides requirements for how services shall be specified in such a way that one service can be specified independently of one or more underlying distributed computing platforms. The framework provides requirements for a further mapping to specific platforms in order to enable conformant platform specific specifications to ensure conforming and interoperable service implementations.

— "access, process and manage" means that geodata users can query remote databases and control remote processing resources and also take advantage of other distributed computing technologies, such as software delivered to the user's local environment from a remote environment for temporary use;

— "from a variety of sources" means that users will have access to data acquired in a variety of ways and stored in a wide variety of relational and non-relational databases;

— "across a generic computing interface" means that ISO 19119 interfaces provide reliable communication between otherwise disparate software resources that are equipped to use these interfaces;

— "within an open information technology environment" means that this International Standard enables geoprocessing to take place outside of the closed environment of monolithic GIS, remote sensing, and AM/FM systems that control and restrict database, user interface, network and data manipulation functions;

— services shall be categorised according to a service taxonomy based on architectural areas and may also be categorised according to a usage life cycle perspective, as well as according to domain specific and user defined service taxonomies, providing support for publication and discovery of services.

The difference between this version of this International Standard and the previous ISO 19119:2005 version is the following:

This International Standard has defined a set of requirements and related abstract tests for the specification of services according to enterprise, computational, information, engineering and technology viewpoints. This International Standard has defined a set of requirements for categorizing services according to service taxonomies. The service metadata has been moved to ISO 19115-1.

Service policies, service contracts including service level agreements (SLAs) are currently not specified as part of this International Standard, as these are considered most relevant for service deployment and service ownership, which is not currently a focus for this International Standard.

Geographic information — Services

1 Scope

This International Standard defines requirements for how platform neutral and platform specific specification of services shall be created, in order to allow for one service to be specified independently of one or more underlying distributed computing platforms.

This International Standard defines requirements for a further mapping from platform neutral to platform specific service specifications, in order to enable conformant and interoperable service implementations.

This International Standard addresses the Meta:Service foundation of the ISO geographic information reference model described in ISO 19101-1:2014, Clause 6 and Clause 8, respectively.

This International Standard defines how geographic services shall be categorised according to a service taxonomy based on architectural areas and allows also for services to be categorised according to a usage life cycle perspective, as well as according to domain specific and user defined service taxonomies, providing support for easier publication and discovery of services.

2 Conformance

2.1 Claiming conformance

Any product claiming conformance with the conformance classes in this International Standard shall pass all the associated requirements described in the abstract test suite given in Annex A.

2.2 General

This International Standard defines six conformance classes shown in Table 1 to Table 6, matching the six requirements classes described in Clause 7 to Clause 12. Any service claiming conformance to any requirements class in this International Standard shall pass all of the tests listed in the corresponding conformance class, which are described in detail in the abstract test suites in Annex A. Each test relates to one or more specific requirements, which are explicitly indicated in the description of the test.

2.3 Enterprise viewpoint

The enterprise viewpoint conformance class is shown in Table 1.

Table 1 — Enterprise viewpoint conformance class

Conformance class	**/conf/enterpriseviewpoint**
Requirements	/req/enterpriseviewpoint (Table 11)
Tests	All tests in A.2

2.4 Computational viewpoint

The computational viewpoint conformance class is shown in Table 2.

Table 2 — Computational viewpoint conformance class

Conformance class	**/conf/computationalviewpoint**
Dependency	/conf/enterpriseviewpoint
Requirements	/req/computationalviewpoint (Table 12)
Tests	All tests in A.3

2.5 Information viewpoint

The information viewpoint conformance class is shown in Table 3.

Table 3 — Information viewpoint conformance class

Conformance class	**/conf/informationviewpoint**
Dependency	/conf/uml (2.4)
Requirements	/req/informationviewpoint (Table 18)
Tests	All tests in A.4

2.6 Service taxonomies

The service taxonomy conformance class is shown in Table 4.

Table 4 — Service taxonomies conformance class

Conformance class	**/conf/servicetaxonomies**
Dependency	/conf/uml (2.4)
Requirements	/req/servicetaxonomies (Table 19)
Tests	All tests in A.5

2.7 Engineering viewpoint

The engineering viewpoint conformance class is shown in Table 5.

Table 5 — Engineering viewpoint conformance class

Conformance class	**/conf/engineeringviewpoint**
Dependency	/conf/uml (2.4)
Requirements	/req/engineeringviewpoint (Table 26)
Tests	All tests in A.6

2.8 Technology viewpoint

The technology viewpoint conformance class is shown in Table 6.

Table 6 — Technology viewpoint conformance class

Conformance class	**/conf/technologyviewpoint**
Dependency	/conf/uml (2.4)
Requirements	/req/technologyviewpoint (Table 27)
Tests	All tests in A.7

NOTE The definition of an abstract test suite appears in ISO 19105.

3 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 10746-1, *Information technology — Open Distributed Processing — Reference model: Overview — Part 1*

ISO 19101-1:2014, *Geographic information — Reference model — Part 1: Fundamentals*

ISO 19103, *Geographic information — Conceptual schema language*

ISO 19115-1:2014, *Geographic information — Metadata — Part 1: Fundamentals*

[SoaML] *Service oriented architecture Modeling Language v 1.0.1*, May 2012, OMG standard[1)]

4 Terms and definitions and abbreviations

4.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

4.1.1
capability
real-world effect that a *service* (4.1.12) provider is able to provide to a service consumer

[SOURCE: SOA-RAF]

4.1.2
computational viewpoint
viewpoint (4.1.15) on an ODP system and its environment that enables distribution through functional decomposition of the system into objects which interact at *interfaces* (4.1.8)

[SOURCE: ISO/IEC 10746-3:2015, 4.1.1.3]

4.1.3
distribution transparency
property of hiding from a particular user the potential behaviour of some parts of a distributed system

Note 1 to entry: Distribution transparencies enable complexities associated with system distribution to be hidden from applications where they are irrelevant to their purpose.

[SOURCE: ISO/IEC 10746-2:2009, 11.1.1]

4.1.4
engineering viewpoint
viewpoint (4.1.15) on an ODP system and its environment that focuses on the mechanisms and functions required to support distributed interaction between objects in the system

[SOURCE: ISO/IEC 10746-3:2009, 4.1.1.4]

4.1.5
enterprise viewpoint
viewpoint (4.1.15) on an ODP system and its environment that focuses on the purpose, scope and policies for that system

[SOURCE: ISO/IEC 10746-3:2009, 4.1.1.1]

1) http://www.omg.org/spec/SoaML/1.0.1/

4.1.6
entity
something that has separate and distinct existence and objective or conceptual reality

4.1.7
information viewpoint
viewpoint (4.1.15) on an ODP system and its environment that focuses on the semantics of information and information processing

[SOURCE: ISO/IEC 10746-3:2009, 4.1.1.2]

4.1.8
interface
named set of *operations* (4.1.10) that characterize the behaviour of an *entity* (4.1.6)

Note 1 to entry: See 8.2 for a discussion of interface.

4.1.9
interoperability
capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units

[SOURCE: ISO/IEC 2382:2009, 2121317]

4.1.10
operation
specification of a transformation or query that an object may be called to execute

Note 1 to entry: An operation has a name and a list of parameters.

Note 2 to entry: See 8.2 for a discussion of operation.

4.1.11
real world effect
actual result of using a *service* (4.1.12), rather than merely the *capability* (4.1.1) offered by a service provider

Note 1 to entry: See 8.3 for a discussion of service.

[SOURCE: OASIS RAF, 3.2.3]

4.1.12
service
distinct part of the functionality that is provided by an *entity* (4.1.6) through *interfaces* (4.1.8)

4.1.13
service chain
sequence of *services* (4.1.12) where, for each adjacent pair of services, occurrence of the first action is necessary for the occurrence of the second action

4.1.14
technology viewpoint
viewpoint (4.1.15) on an ODP system and its environment that focuses on the choice of technology in that system

[SOURCE: ISO/IEC 10746-3:2009, 4.1.1.5]

4.1.15
viewpoint (on a system)
form of abstraction achieved using a selected set of architectural concepts and structuring rules, in order to focus on particular concerns within a system

[SOURCE: ISO/IEC 10746-2, 3.2.7]

4.1.16
workflow
automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules

4.2 Abbreviations

API	Application Programming Interface
BPEL	Business Process Execution Language
BPMN	Business Process Modelling Notation
CORBA	Common Object Request Broker Architecture
CSL	Conceptual schema language
DAG	Directed Acyclic Graph
DCP	Distributed Computing Platform
DEM	Digital Elevation Model
DTD	Document type definitions
EJB	Enterprise Java Beans
ERP	Enterprise Resource Planning
GIOP	General Inter-ORB Protocol
GFM	General feature model
HTI	Human Technology Interface
HTML	Hypertext Markup language
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as a Service
IDL	Interface Definition Language
IIOP	Internet Inter-ORB Protocol
INSPIRE	Infrastructure for Spatial Information in Europe
IT	Information Technology
J2EE	Java 2 Enterprise Edition with EJB
JDBC	Java Data Base Connectivity
OASIS	Organization for the Advancement of Structured Information Standards

OCL	Object Constraint Language
ODBC	Open Database Connectivity
ODMG	Object Database Management Group
ODP	Open Distributed Processing (see RM-ODP)
OGC	Open Geospatial Consortium
OMG	Object Management Group
ORB	Object Request Broker
OWL	Web Ontology Language
PaaS	Platform as a Service
QoS	Quality of Service
QVT	Query/View/Transformation
REST	Representational state transfer
RDF	Resource Description Framework
RMI	Remote Method Invocation
RM-ODP	Reference Model of Open Distributed Processing (ISO/IEC 10746)
RPC	Remote Procedure Call
SaaS	Software as a Service
SDI	Spatial Data Infrastructure
SDAI	Standard Data Access Interface (ISO 10303-22)
SOA	Service Oriented Architecture
SoaML	Service oriented architecture Modelling Language (OMG)
SOAP	Simple Object Access Protocol
SOF	Service Organizer Folder
SPS	Spatial Planning Service
SQL	Structured Query Language
UML	Unified Modeling Language
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WFS	Web Feature Service
WMS	Web Map Service
XML	Extensible Markup Language

XML RDF XML Resource Description Framework

XSLT XML Stylesheet Language Transformations

Concepts from schemas defined in some other International Standards are designated with names that start with bi-alpha codes as follows:

TM ISO 19108:2002 Temporal Schema, Temporal Objects

5 Notation

5.1 General

This International Standard describes how to describe a service. In addition to stating the rules for creating service descriptions, this International Standard provides guidance through examples.

5.2 Conformance class

Conformance to this International Standard is possible at a number of levels, specified by conformance classes (Clause 2). Each conformance class is summarized using the template shown as Table 7.

Table 7 — Conformance class template

Conformance class	**/conf/{classM}**
Dependency	[identifier for another conformance class]
Requirements	/req/{classA}
Tests	[reference to clause(s) containing tests]

All tests in a class shall be passed, so dependencies are on other conformance classes (see Resolution 570 of ISO/TC 211, N3262). Each conformance class tests conformance to a set of requirements packaged in a requirements class (Clause 7 and Clause 8).

5.3 Requirements class

Each normative statement (requirement or recommendation) in this International Standard forms part of a specific requirements class. In this International Standard, each requirements class is described in a discrete clause or subclause and summarized using the template shown as Table 8.

Table 8 — Requirements class template

Requirements class	**/req/{classM}**
Target type	[artefact or technology type]
Dependency	[identifier for another requirements class]
Requirement	/req/{classM}/{reqN}
Recommendation	/req/{classM}/{recO}
Requirement	/req/{classM}/{reqP}
Requirement /Recommendation	[repeat as necessary]

All requirements in a class shall be satisfied, so the requirements class is the unit of re-use and dependency. Hence, the value of a Dependency requirement is another requirements class.

5.4 Rules

All rules are normative and each rule is presented using the following template where /req/[classM]/[reqN] identifies the requirement or recommendation. The use of this layout convention allows the normative provisions of this International Standard to be easily located by implementers.

/req/[classM]/[reqN]	[Normative statement]

5.5 Identifiers

Each requirements class, requirement and recommendation has an identifier in the form of a path or partial URI. The identifier supports cross-referencing of class membership, dependencies, and links from each conformance test to requirements tested. The identifier can be appended to a URI that identifies the standard as a whole in order to construct a complete URI which allows the requirements class, requirement or recommendation to be identified in an external context. For example, following the URI scheme for ISO standards [IETF RFC 5141], a URI denoting the ISO 19109 standard is as follows:

http://standards.iso.org/iso/19109/2

The URI for each requirements class has the following form:

http://standards.iso.org/iso/19109/2/req/[classM]

The URI for each requirement or recommendation has the following form:

http://standards.iso.org/iso/19109/2/req/[classM]/[reqN]

5.6 Conceptual schemas

Conceptual schemas in the normative part of this International Standard are presented in the Unified Modeling Language (UML) in conformance with ISO 19103. UML diagrams are presented in compliance with ISO/IEC 19505-2.

5.7 Descriptions of concepts

Concepts from UML are presented in all capitals, e.g. CLASS, PACKAGE, ROLE, ATTRIBUTE, ASSOCIATION.

5.8 Architecture patterns

An architecture pattern expresses a fundamental structural organization or schema for software services. It identifies a set of services, specifies their responsibilities, and includes rules and guidelines for organizing the relationships between them. Services, implemented by classes and objects, can use design patterns but this level of detail is outside the scope of this International Standard.

Table 9 provides a listing of the elements of a pattern. When specific architecture patterns are defined in this International Standard, these elements are suggested to be used.

Table 9 — Elements of a pattern

Element of a pattern	Description of element
Name	The name is a word or short meaningful phrase that describes the pattern. The name is extremely important since it is used to reduce communication overhead. Nick-names or synonyms may be provided.
Problem	This is a statement of the problem which describes its intent, goals and objectives it wants to reach within the given context and forces. Often, the forces oppose these objectives, as well as each other.

Table 9 *(continued)*

Element of a pattern	Description of element
Context	Context defines the preconditions under which the problem and its solution seem to recur, and for which the solution is desirable. This defines the pattern's applicability. It can be thought of as the initial configuration of the system before the pattern is applied.
Forces	The forces are considerations that should be weighed to reach the best solution. Forces define the kinds of trade-offs that should be considered in the presence of the tension or dissonance they create. The forces answer the question: "Why is this a hard problem?"
Structure	Structure defines the static relationships and dynamic rules describing how to realize the desired outcome. The structure description is accomplished through a collaboration diagram.

6 Overview of geographic services architecture

6.1 Purpose and justification

The definition of service includes a variety of applications with different levels of functionality to access and use geographic information. While specialized services will appropriately remain an area for service producers, standardization of the interfaces to those services allows interoperability between proprietary products. Geographic information system and software developers will use this International Standard to provide general and specialized services that can be used for all geographic information. The approach of this International Standard is integrated with the approaches being developed within the more general world of information technology, in particular related to service oriented architectures.

The geographic services architecture specified in this International Standard has been developed to meet the following purposes:

— provide an abstract framework to allow coordinated development of specific services;

— enable interoperable data services through interface standardization;

— support development of a service catalogue through the definition of service metadata;

— allow separation of data instances and service instances;

— enable use of one provider's service on another provider's data;

— define an abstract framework which can be implemented in multiple ways.

6.2 Relationship to ISO 19101-1

Table 10 — Reference model conceptual framework

Reference model conceptual framework				
Level\Foundation	**Interoperability**			**Procedural standards**
	Semantic foundation	**Syntactic foundation**	**Service foundation**	
Meta-meta	Meta-meta:Semantic	Meta-meta:Syntactic	Meta-meta:Service	Meta-meta:Procedural
Meta	Meta:Semantic	Meta:Syntactic	Meta:Service	Meta:Procedural
Application	Application:Semantic	Application:Syntactic	Application:Service	Application:Procedural
Instance	*Instance:Semantic*	*Instance:Syntactic*	*Instance:Service*	*Instance:Procedural*

Table 10 shows the service ISO 19101-1 Reference model conceptual framework with the Service foundation for Interoperability, which is being addressed by this International Standard as follows:

— **Meta-meta:Service**

  Meta-meta:Service consists of standards that serve as foundation for the definition of rules and methodologies for the development of geographic information processing and services for, but not limited to, the discovery, the access and the processing of geographic information.

  This International Standard is relating to other standards for this level (Meta-meta:Service), in particular, ISO RM/ODP, OASIS SOA RM, ISO 19101-1:2014, ISO 19103, ISO 19109 and OMG SoaML.

— **Meta:Service**

  Meta:Service consists of standards defining rules and methodologies for the modelling and development of geographic information processing and services. This International Standard is addressing this level in particular.

— **Application:Service**

  Application:Service consists of definitions of standardized geographic information services. Capabilities of the service agree with the Application:Semantic level. Application:Service includes the following:

  — standards for geographic human interaction services;

  — standards for geographic model/information management services;

  — standards for geographic workflow/task management services;

  — standards for geographic processing services:

    — spatial (i.e. vector, coverage, and imagery and gridded data);

    — thematic (e.g. web services for mapping, delivering data about features, and filtering data);

    — temporal;

    — metadata;

  — standards for geographic communication services.

— **Instance:Service**

  Instance:Service is included in the reference model conceptual framework for completeness but is not part of the scope of this International Standard. It consists of service instances (including Web services) complying with services defined as part of Application:Service.

6.3 Interoperability reference model based on ISO RM-ODP

This International Standard is developed based on a system architecture approach to system design known as the Reference Model of Open Distributed Processing (RM-ODP); see ISO/IEC 10746-1. Architecture is defined as a set of components, connections and topologies defined through a series of views. The geographic infrastructure enabled by this International Standard will have multiple users, developers, operators and reviewers. Each group will view the system from their own perspective. The purpose of architecture is to provide a description of the system from multiple viewpoints. Furthermore, architecture helps to ensure that each view will be consistent with the requirements and with the other views.

Table 11 shows how the RM-ODP viewpoints are utilized in this International Standard.

Table 11 — Use of RM-ODP viewpoints in this International Standard

Viewpoint name	Definition of RM-ODP Viewpoint (ISO/IEC 10746-1)	How viewpoint is addressed in this International Standard
enterprise viewpoint	See 4.1.5.	See Clause 7, enterprise viewpoint.
computational viewpoint	See 4.1.2.	See Clause 8, computational viewpoint.
information viewpoint	See 4.1.7.	See Clause 9, information viewpoint.
engineering viewpoint	See 4.1.4.	See Clause 11, engineering viewpoint.
technology viewpoint	See 4.1.14.	See Clause 12 technology viewpoint; also to be addressed by platform-specific service specifications.

The enterprise viewpoint is concerned with the purpose, scope and policies of an enterprise or business and how they relate to the specified system or service. An enterprise specification of a service is a model of that service and the environment within which the service operates. It covers the role of the service in the business and the human-user roles and business policies related to the service.

The computational viewpoint is concerned with the interaction patterns between the components (services) of the system, described through their interfaces. A computational specification of a service is a model of the service interface seen from a client and the potential set of other services that this service requires to have available, with the interacting services described as sources and sinks of information.

The information viewpoint is concerned with the semantics of information and information processing. An information specification of a system is a model of the information that it holds and of the information processing that it carries out.

The engineering viewpoint is concerned with the design of distribution-oriented aspects, i.e. the infrastructure required to support distribution. An engineering specification of a system defines a networked computing infrastructure that supports the system structure defined in the computational specification and provides the distribution transparencies that it defines. This viewpoint defines the following distribution transparencies: access, failure, location, migration, relocation, replication, persistence and transaction. Security may also be a mechanism.

The technology viewpoint describes the implementation of a system in terms of a configuration of technology objects representing the hardware and software components of the implementation. It is constrained by cost and availability of technology objects (hardware and software products) that would satisfy this specification. These may conform to platform-specific standards that are effectively templates for technology objects.

In the computational and information viewpoint clauses of this International Standard, specific approaches that shall be followed for defining geographic information services are provided. For the engineering and technology viewpoints, this International Standard defines how a particular service shall be mapped on to an implementation technology, such as Web services, REST services, SQL, CORBA, Internet or similar technologies.

6.4 Service abstraction

Various definitions for services exist; however, in this International Standard, the following definition for services is used:

A service is a distinct part of the functionality that is provided by an entity through interfaces.

Other standard developments in this area have introduced more elaborate definitions. The following definition for a service can be found in the OASIS SOA Reference Architecture Framework [SOA-RAF].

*A **service** is a mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description. A service is provided by an entity (the service provider) for use by others, but the eventual*

consumers of the service may not be known to the service provider and may demonstrate uses of the service beyond the scope originally conceived by the provider.

OMG SoaML defines service as follows [SoaML].

*A **service** is value delivered to another through a well-defined interface and available to a community (which may be the general public). A service results in work provided to one by another.*

A Service represents a feature of a Participant that is the offer of a service by one participant to others using well defined terms, conditions and interfaces. A Service designates a Port that defines the connection point through which a Participant offers its capabilities and provides a service to clients.

In this context, participants, ports, service description and capabilities are defined as follows.

— Participants: participants are either specific entities or kinds of entities that provide or use services. Participants can represent people, organizations, or information system components. Participants may provide any number of services and may consume any number of services.

— Ports: participants provide or consume services via ports. A port is the part or feature of a participant that is the interaction point for a service, where it is provided or consumed. A port where a service is offered may be designated as a “Service” port and the port where a service is consumed may be designated as a “Request” port.

— Service description: the description of how the participant interacts to provide or use a service is encapsulated in a specification for the service (there are two ways to specify a service interaction) as a simple UML Interface or as a two-way ServiceInterface. These different ways to specify a service related to the service oriented architecture (SOA) approach and the complexity of the service, but in each case, they result in interfaces and behaviours that define how the participant will provide or use a service through ports. The service descriptions are independent of, but consistent with, how the provider provides the service or how (or why) the consumer consumes it. This separation of concerns between the service description and how it is implemented is fundamental to SOA. A service specification specifies how consumers and providers are expected to interact through their ports to enact a service, but not how they do it.

— Capabilities: participants that provide a service should have a capability to provide it, but different providers may have different capabilities to provide the same service; some may even “outsource” or delegate the service implementation through a request for services from others. The capability behind the service will provide the service according to a service description and may also have dependencies on other services to provide that capability. The service capability is frequently integral to the provider’s business process. Capabilities can be seen from two perspectives: capabilities that a participant has that can be exploited to provide services and capabilities that an enterprise needs that can be used to identify candidate services.

Regardless of how services are identified, they are formalized by service descriptions. A service description defines the purpose of the service and any interaction or communication protocol for how to properly use and provide a service. A service description may define the complete interface for a service from its own perspective, irrespective of any consumer request it might be connected to. Alternatively, the agreement between a consumer request and provider service may be captured in a common service contract defined in one place and constraining both the consumer’s request service interface and the provider’s service interface.

This approach to service modelling relies on model-driven techniques to separate the logical implementation of a service from its possible physical realizations on various platforms. This separation of concerns both keeps the services models simpler and more resilient to changes in underlying platform and execution environments. Using this approach, service models can support a variety of technology implementations and tool support can help automate these technology mappings.

Figure 1 defines the relationship between the various types of service specifications. SV_ServiceSpecification defines services without reference to the type of specification or to its implementation. An SV_PlatformNeutralServiceSpecification provides the abstract definition of a specific type of service but does not specify the implementation of the service. Service types are

given in the geographic service taxonomy in 10.8. SV_PlatformSpecificServiceSpecification defines the implementation of a specific type of service. There may be multiple platform-specific specifications for a single platform-neutral specification. CodeList ServiceMetadata:DCP contains different alternatives for target technology platforms. SV_Service is an implementation of a service. The requirements for these specifications are addressed in this International Standard, in particular in Clause 10.

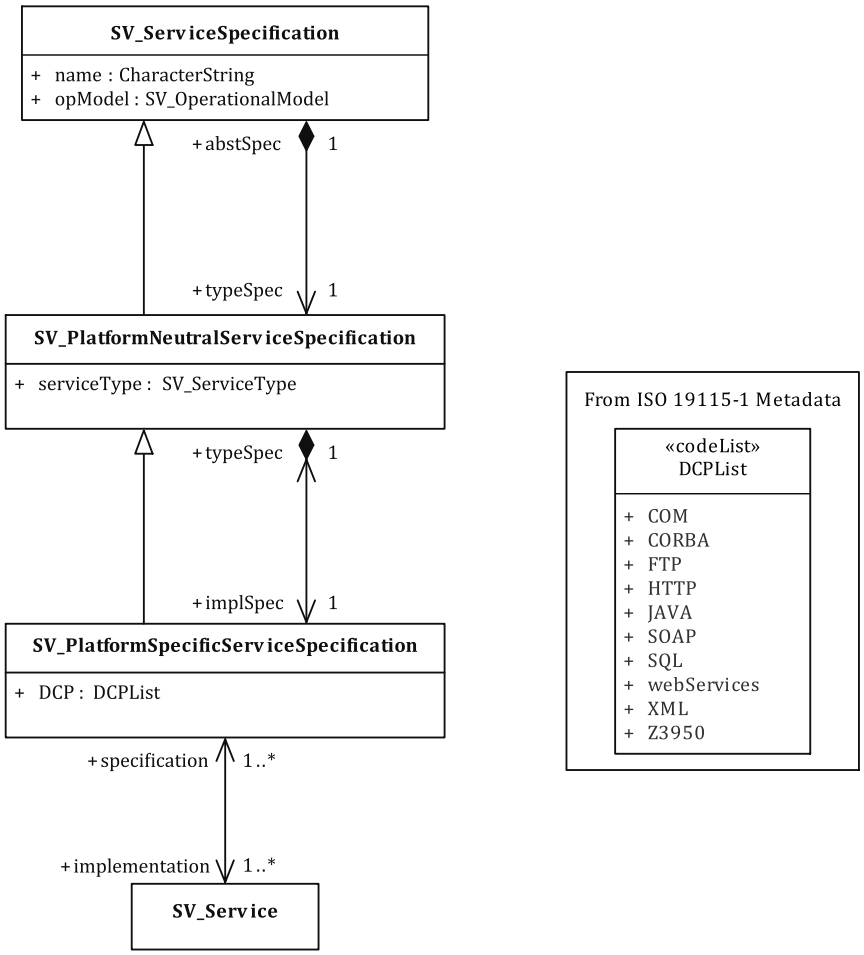


Figure 1 — Abstract and implementation service specifications

6.5 Interoperability

Interoperability is the capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units.

Two components 1 and 2 (see Figure 2) can interoperate (are interoperable) if 1 can send requests 3 for services to 2, based on a mutual understanding of 3 by 1 and 2, and if 2 can similarly return mutually understandable responses 4 to 1.

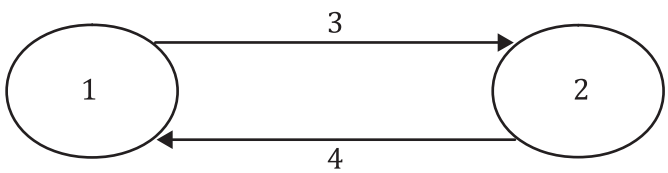


Figure 2 — Interoperability

This means that two interoperable systems can interact jointly to execute tasks. For the geographic domain, the following description of the term "geographic interoperability" is applicable:

"Geographic interoperability" is the ability of information systems to a) freely exchange all kinds of spatial information about the Earth and about the objects and phenomena on, above, and below the Earth's surface, and b) cooperatively, over networks, run software capable of manipulating such information.

Interoperability in the geographic information context is further described in ISO 19101-1.

The ODP viewpoint abstraction provides a framework for describing a system at several abstraction levels. In this International Standard, interoperability is viewed in terms of the different abstraction levels provided by RM-ODP. This International Standard focuses, from different viewpoints, on how semantic and syntactic interoperability of geographic metadata and geographic data can be supported.

When two different organizations have independently developed distributed systems, each can be described according to the RM-ODP viewpoints, and interoperability between the systems can be discussed with respect to each of the five RM-ODP viewpoints.

For each interoperability aspect, a distinction is made between syntactical interoperability and semantic interoperability. Syntactical interoperability ensures that there is a technical connection, i.e. that the data can be transferred between systems. Semantic interoperability ensures that the content is understood in the same way in both systems, including by those humans interacting with the systems in a given context.

6.6 Use of other geographic information standards in service specifications

A service specification shall include relevant information models from the appropriate geographic information in the ISO geographic information suite of standards. The corresponding UML models may be used in the definition of the service interfaces as appropriate.

7 Enterprise viewpoint: A context for services

7.1 Enterprise viewpoint

The Enterprise viewpoint describes the context for a system and a set of services. It concentrates on the objectives, business rules and policies that need to be supported by systems and services. An enterprise specification of a service is a model of that service and the environment with which the service interacts. It covers the role of the service in the business and the human-user roles and business policies related to the service. In the context of service specifications, there is a particular focus on the use cases and external functionally related to the particular services.

Experience with the development of services has shown that it is very useful to have models for the enterprise viewpoint, focusing on generic descriptions of the usage, and typical process for usage. This helps to shape the understanding of the functionality and constraints that are placed on systems and services. It also helps concrete project and development activities to place the needs of project and development activities in the context of both existing and available standards and services, as well as supporting the identification of new services.

The enterprise viewpoint contains the scope and structure of the service being described. This viewpoint will explain why the service is needed and what it is intended to do, and relates the service to its objectives. It should be clear and concise and targeted for a nontechnical audience.

The Enterprise viewpoint is a human-oriented description of the service. The purpose of this viewpoint is to provide a good understanding of the service and its actions for people who may have an interest in the service. The other viewpoints will also have a computer-oriented specification of the service for the purpose of supporting realization and conformance testing of the service.

Geographic information services aim at supporting a multitude of users and organizations to support the variety of work processes in which they are involved. In order to reach their goals of enhanced spatial

data sharing and processing, they should cover the business requirements of as many organizations and processes as possible.

A work process is defined as the way in which organizations create products, services or policies. It is a succession of structured and interconnected activities across time and space which, starting from an identifiable input, result in a defined output in the form of a product or service. In order to obtain the desired output, the input should be transformed. Ideally, the transformation that occurs in the process should add value to the input and create an output that is more useful to the recipient either upstream or downstream. Traditionally, work processes occurred within single organizations, but increasingly cross organizational and even country boundaries. Often, a process is divided into several sub-processes due to complexity, which can in turn be sub-divided in a series of activities and tasks. Therefore, the simple input-throughput-output model will rather consist of several interconnected input-throughput-output chains whereby the output of one sub-process serves as the input for another sub-process. Services with service chaining can be used to support such work processes.

Many processes create products based on other products, which is, for example, the case for manufacturers of cars. For work processes dealing with policy preparation, monitoring and evaluation, decision making, or service provision, the notion of data and information flows is crucial. Data and information are needed as input, to services and service chains, in order to process them and to create new data and information that can be used to make decisions, to serve other organisations, policy makers or even individual citizens.

7.2 Enterprise viewpoint service specifications

The requirements for creating a service specification in the enterprise viewpoint are formalized as a requirements class summarized in Table 12.

Table 12 — Requirements class for Enterprise viewpoint service specifications

Requirements class	**/req/enterpriseviewpoint**
Target type	Service description
Requirement	/req/enterpriseviewpoint/servicename
Requirement	/req/enterpriseviewpoint/servicetypes
Requirement	/req/enterpriseviewpoint/purpose
Requirement	/req/enterpriseviewpoint/scope
Requirement	/req/enterpriseviewpoint/capabilities
Requirement	/req/enterpriseviewpoint/community
Recommendation	/rec/enterpriseviewpoint/scenarios

/req/enterpriseviewpoint/ servicename	Service name shall be described as a textual string stating the name of this service.

NOTE 1 This is a human-readable identification for this service (e.g. Web Map Service, Web Feature Service, Sensor Planning Service, Feature Extraction Service). The name might also have a corresponding short name (e.g. WMS, WFS, SPD, FES).

/req/enterpriseviewpoint/ servicetypes	The service shall be categorised according to its service type based on the architectural service taxonomy in this International Standard, and may also be categorised according to other service taxonomies.

The service purpose should clearly describe the intentional goals of the service. Defines the purpose the service intends or resolve, to perform or accomplish.

/req/enterpriseviewpoint/ purpose	Service purpose shall be described as a textual paragraph stating the scope and objective of the service

The service purpose should clearly describe the intentional goals of the service. Defines the purpose the service intends or resolve, to perform or accomplish.

/req/enterpriseviewpoint/ scope	Service scope shall be described as a textual paragraph stating the capabilities that will be provided through the service, and may also state relevant capabilities that are out of scope.

NOTE 2 The capabilities can be described as a list of different capabilities and the real world effects provided by the service. Capabilities generate real-world effects that can be as simple as sharing information or can involve performing a function as part of a complex process or changing the state of other related processes

/rec/enterpriseviewpoint/ capabilities	Service capabilities shall be specified as a list of the functionalities offered by the service and their corresponding real world effect.

NOTE 3 The capabilities can be described as a list of different capabilities and the real world effects provided by the service. Capabilities generate real-world effects that can be as simple as sharing information or can involve performing a function as part of a complex process or changing the state of other related processes.

/req/enterpriseviewpoint/ community	Service community shall be described as a textual paragraph stating roles of potential actors involved in using the service

The roles should be described in terms of the high level provisioning and use of capabilities with data exchanges between the user/consumer roles and the role of the service.

/rec/enterpriseviewpoint/ scenarios	Service usage may be specified through a process diagram (i.e. BPMN and/or UML use cases) to show service usage in the context of a possible usage process.

This is a description of relevant usage scenarios. These scenarios represent the conceptual model for the use of the interfaces/operations provided by the service.

The scenarios are used to describe a typical usage of the service from an enterprise viewpoint. The scenarios should describe the primary flow. If the scenarios have alternative flows, these should also be documented.

Simple alternative flows can be documented in text within the primary flow. Complex alternative flows should be described separately.

The scenarios could be described as a sequence of steps, but it is recommended that diagrams be utilized to augment the narrative description of each business scenario. The use of BPMN and/or UML use case diagrams/templates is recommended.

Scenarios will better describe the service than explanatory text, since they are illustrations of the role the service is envisioned to play. This should be done for representative scenarios and it does not need to be exhaustive.

The scenarios can also be related to test cases that describe the specific functions and objectives for validating usage of a service. Specific actions are identified and might be measured against expected testing results and outcomes.

Specifications of Quality of Service (QoS) can be useful to associate with service descriptions. For this purpose, there is a recommendation to use the “UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms (QFTP) [QoS].

7.3 Examples of relevant standards

— Business Process Model and Notation (BPMN). This a standard from OMG that is used for the purpose of both enterprise and business-oriented modelling, as well as for technology mapping to process execution with technologies, such as BPEL.

— Use cases (UML). Use cases with use case templates[5] have been the most used form for documentation of typical user needs for system and services functionality, within the geographic information community. The UML standard provides only a graphical form for diagramming of use cases, while the geographic information community typically has adopted various forms of use case templates for the technical documentation of use cases.

— Agile requirements engineering with user stories. The software engineering community has recently evolved a set of approaches around agile methods that focuses on close interactions between potential system users and developers, and focuses on more light-weight user stories as input to a system development backlog. A user story can be expanded further into a use case.

— Business Motivation Metamodel (BMM) (http://www.omg.org/spec/BMM/). This is a standard from OMG that provides a foundation for modelling of vision, goals and objectives for an enterprise, with mappings to tactics for solutions including use of processes and services.

— UML4ODP Enterprise specification profile (ISO/IEC 19793). This standard provides a UML profile for all of the main concepts defined in the RM-ODP enterprise viewpoints. It is a good reference and foundation for doing full RM-ODP enterprise viewpoint modelling but in the geographic information community, it has so far been a preference to use a more light weight approach with BPMN and/or Use cases. See example in Reference [41].

7.4 Example and tools

Annex D provides an example of a use case-based methodology for the identification of needed resources in a geographic information service context, with an example use case template in Annex E including both data and service resources.

To illustrate service specifications for the various viewpoints, some elements of examples will be related to parts of ISO 19123 and a Sensor Planning Service and in particular, the GetCapabilities operation used in both WMS and WFS.

8 Computational viewpoint: A basis for service interfaces and chaining

8.1 Component and service interoperability and the computational viewpoint

The computational viewpoint is concerned with describing the entities of a distributed system independent of implementation and semantic content. It describes the interaction patterns between the entities and their interfaces. To be able to interoperate from the computational viewpoints, two systems should be *interface-and-services-interoperable*. Two systems are interface-and-services-interoperable if they agree on the set of services offered by the entities of the two systems and the interfaces to these entities. If standardized interfaces are defined, the entities of one system will be able to request services from entities in another system.

Clause 8 provides the following:

— defines the concepts of services, interfaces and operations and the relations between these concepts;

— provides an approach to physical distribution of services using an n-tier architecture;

— defines a model for combining services in a dependent series to achieve larger tasks, e.g. service chaining;

— defines a service metadata model to support service discovery through a service catalogue.

8.2 Services, interfaces and operations

Definitions and relationships of several terms are provided in 8.2. The following terms are used extensively in this International Standard:

— service: distinct part of the functionality that is provided by an entity through interfaces;

— interface: named set of operations that characterize the behaviour of an entity;

— operation: specification of a transformation or query that an object may be called to execute. It has a name and a list of parameters.

These terms are related to each other as depicted in Figure 3, which shows that services are specified by a set of interfaces that are a set of operations. Interfaces are implemented as ports that make services available to users.

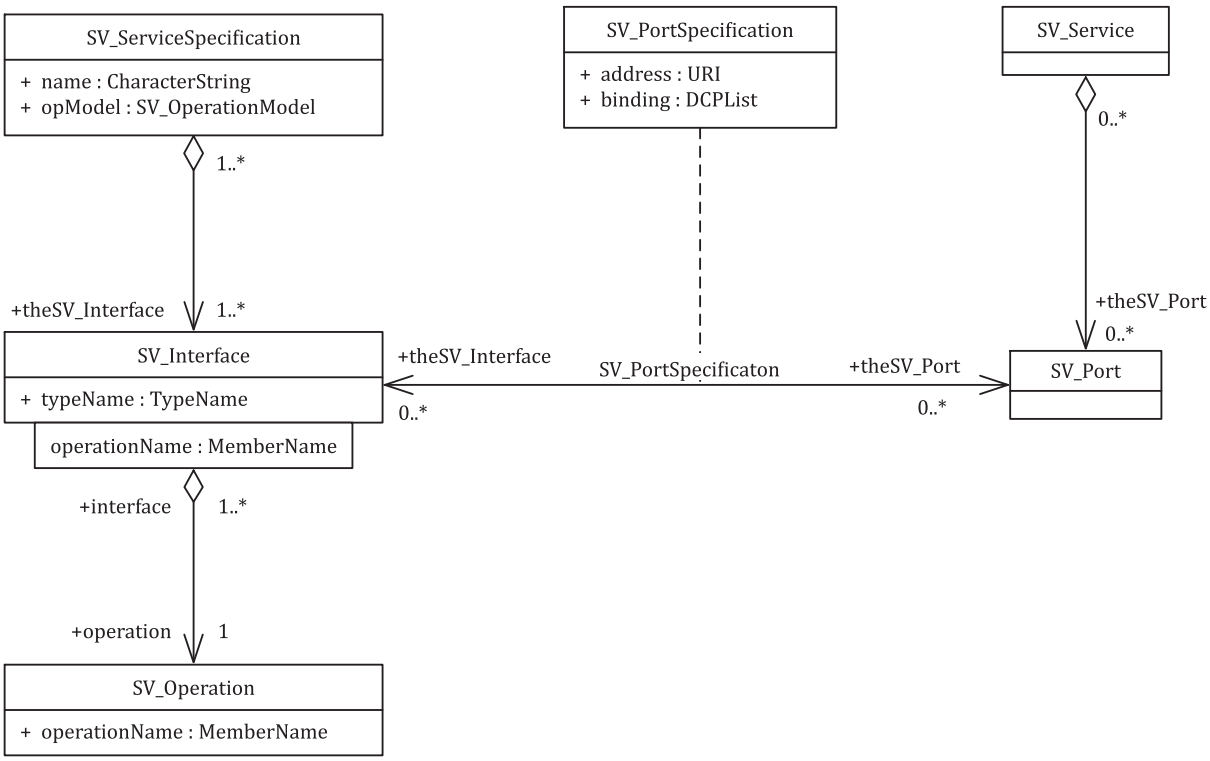


Figure 3 — Service definition relationships

The aggregation of interfaces in a service is for the purpose of defining functionality of value to the users. Users in this context are either software agents or human users. A service provides functionality that adds value. The value is apparent to the user who invoked the service.

The aggregation of operations in an interface and the definition of interface are defined for the purpose of software reusability. Interfaces are defined in order to be reusable for multiple service types. The syntax of an interface may be reused with multiple services with different semantics.

Services of multiple types may be aggregated. The service types are defined consistent with the service taxonomy described in Clause 10. When a service provides functionality beyond that of a single category in the service taxonomy, it will be an aggregate service. Service chaining results in aggregate services as defined in 8.4.

Interfaces are abstract specifications separate from the concrete deployment or data format bindings. The specification of an interface includes a static portion that includes definition of the operations. The

specification of an interface includes a dynamic portion that includes any restrictions on the order of invoking the operations.

An implementation of an interface is a port. The implementation includes implementation of the platform-specific specification and a method to identify the service, e.g. an address.

An implementation of a service may be associated with a specific dataset or it may be a service that can be used to operate on multiple, unspecified datasets. The first case is referred to as a tightly coupled service while the second case is referred to as a loosely coupled service (see 8.4.1).

Interfaces are defined through operations. An operation specifies a transformation on the state of the target object or a query that returns a value to the caller of the operation. An operation shall be an abstract description of an action supported by the interface. Operations contain parameters.

The computational viewpoint is concerned with the interaction patterns between the components (services) of the system, described through their interfaces. A computational specification of a service is a model of the service interface seen from a client and the potential set of other services that this service requires to have available, with the interacting services described as sources and sinks of information. In the context of multi style SOA, the service specification might also include signals (events) that are generated or received and support both synchronous and asynchronous interactions and both RPC oriented and document-oriented/RESTful styles of interaction. The computational viewpoint is the core viewpoint for the identification of interfaces and services.

8.3 Computational viewpoint service specifications

8.3.1 Requirements class for computational viewpoint service specifications

The requirements for creating the service specification part for the computational viewpoint are formalized as a requirements class summarized in Table 13.

Table 13 — Requirements class for Computational viewpoint service specifications

Requirements class	**/req/computationalviewpoint**
Target type	UML service model
Dependency	ISO 19103 (Conceptual schema language) ISO 19115-1 (Metadata)
Requirement	/req/computationalviewpoint/interfaces
Requirement	/req/computationalviewpoint/operations
Recommendation	/rec/computationalviewpoint/behaviour
Recommendation	/rec/computationalviewpoint/pre_and_post_conditions
Requirement	/req/computationalviewpoint/servicechaining
Requirement	/req/computationalviewpoint/servicemetadata
Recommendation	/rec/computationalviewpoint/servicechaining

8.3.2 Service interfaces with operations

/req/computationalviewpoint/interfaces	Services shall be described in a platform neutral way using abstract interfaces for simple one way interfaces and <<Service Interface>> for more complex multi-way interfaces. Interfaces for a service can be categorised as mandatory or optional.

These two approaches to specifying a service can be described as follows.

— Simple Interfaces: The simple interface focuses attention on a one-way interaction provided by a participant on a port represented as a UML interface. The participant receives operations on this port and may provide results to the caller. This kind of one-way interface can be used with "anonymous" callers and the participant makes no assumptions about the caller or the choreography of the service. The one-way service corresponds most directly to simpler "RPC style web services", as well as many object-oriented programming language objects. Simple interfaces are often used to expose the "raw" capability of existing systems or to define simpler services that have no protocols. Simple interfaces are the degenerate case of the ServiceInterface where the service is unidirectional (the consumer calls operations on the provider) the provider does not call back the consumer and may not even know who the consumer is. For the kind of interfaces that reflects two messages only to realize a request and reply interaction, there is no need to split this into a bi-directional service on the specification level, as the two-way interaction then is given by the patterns of the architectural style.

— ServiceInterface based: A ServiceInterface-based approach allows for bi-directional services, those where there are "callbacks" from the provider to the consumer as part of a conversation between the parties. A service interface is defined in terms of the provider of the service and specifies the interface that the provider offers, as well as the interface, if any, it expects from the consumer. The service interface may also specify the choreography of the service, what information is sent between the provider and consumer and in what order. A consumer of a service specifies the service interface they require using a request port. The provider and consumer interfaces should either be the same or compatible. If they are compatible, the provider can provide the service to that consumer. The consumer should adhere to the provider's service interface, but there may not be any prior agreement between the provider and consumer of a service. Compatibility of service interfaces determines whether these agreements are consistent and can therefore be connected to accomplish the real world effect of the service and any exchange in value. The ServiceInterface is the type of a "Service" port on a provider and the type of a "Request" port on the consumer. In summary, the consumer agrees to use the service as defined by its service interface and the provider agrees to provide the service according to its service interface. Compatibility of service interfaces determines whether these agreements are consistent and can therefore be connected. The ServiceInterface approach is most applicable where existing capabilities are directly exposed as services and then used in various ways or in situations that involve one or two parties in the service protocol.

/req/computationalviewpoint/operations	Operations in interfaces shall be described in a platform neutral way showing their input and output parameters (and exceptions).

Operations shall be specified as follows: for each operation (main action), describe the operation (action) name, operation (action) purpose and operation (action) input/outputs in terms of parameters, and further refined and described through the Information viewpoint. For the document/message centred document style, the input and output parameters shall be typed <<MessageTypes>>.

The SoaML standard also provides support for the specification of service contracts and service architectures, represented through UML collaborations, but the use of these is not required in this International Standard.

In an abstract interface, all operations are shown with their input and output parameter in a logical form, related to the definition in corresponding UML models described in the information viewpoint.

Experience has shown that most services are being specified with simple interfaces, where there is no complex two-way protocol between the user and the provider of a service. Sometimes, however, it is the case that more complex two-way interaction is needed. This International Standard is following the recommendation of SoaML to use specifications with standard UML interfaces for simple protocols and to make use of the service interface concept of SoaML only when more complex two-way protocols are needed.

Figure 4 gives an example of description of simple interfaces (source: OGC SOS 2.0).

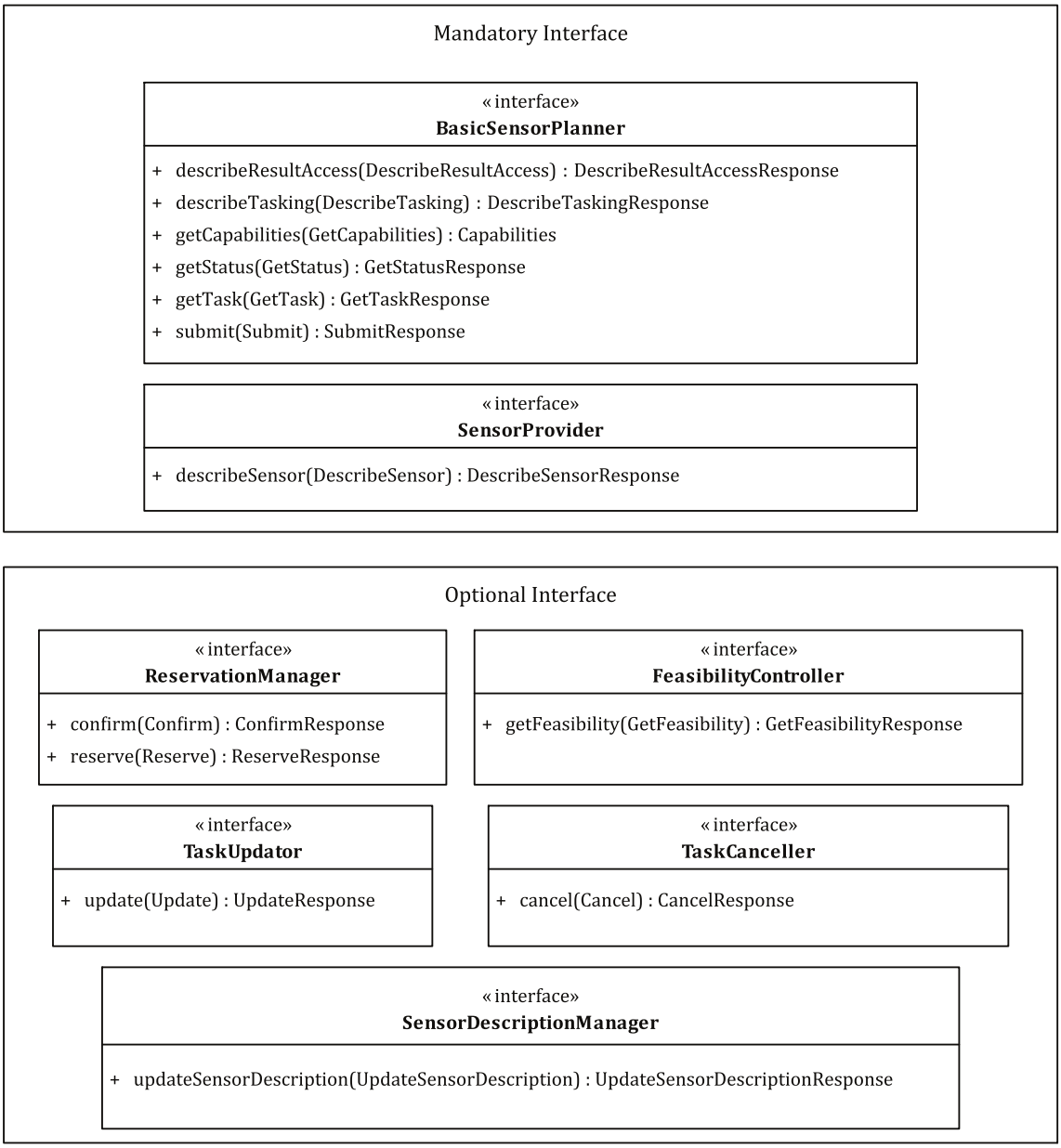


Figure 4 — Example of description of simple interfaces

EXAMPLE BasicSensor planner with operation *GetCapabilities*. This operation allows a client to request and receive service metadata documents that describe the capabilities of the specific server implementation. This operation also supports negotiation of the specification version being used for client-server interactions.

8.3.3 Service behaviour and constraints

/rec/computationalviewpoint/behaviour	The behaviour of services should be illustrated with UML sequence diagrams, showing possible sequence operations. The potential states and state transitions for a service should be illustrated with UML state diagrams.

Figure 5 gives an example on use of sequence diagram showing possible sequenced use of operations (source: OGC SPS).

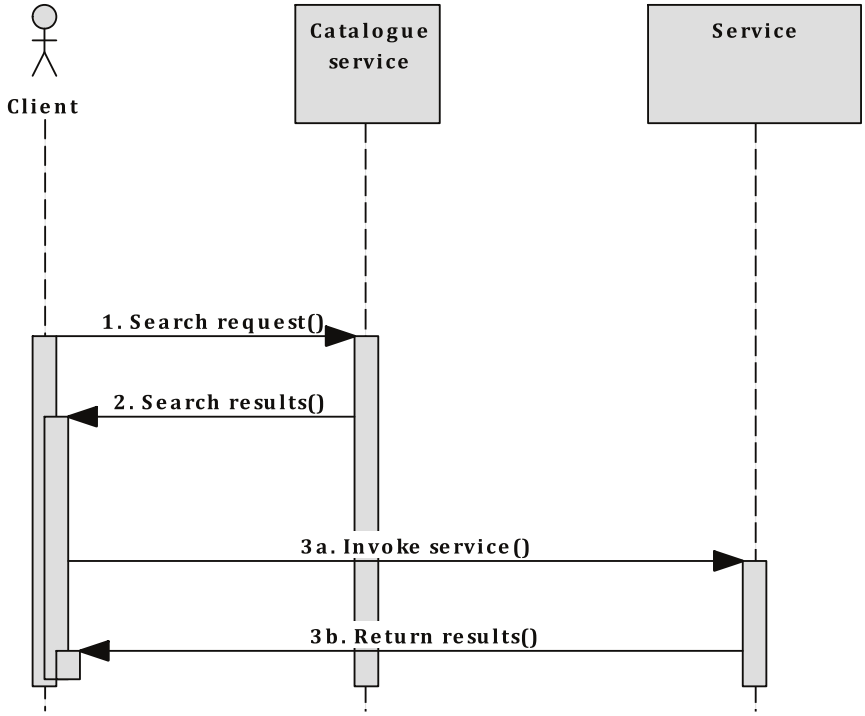


Figure 5 — Example of use of sequence diagram showing possible sequenced use of operations

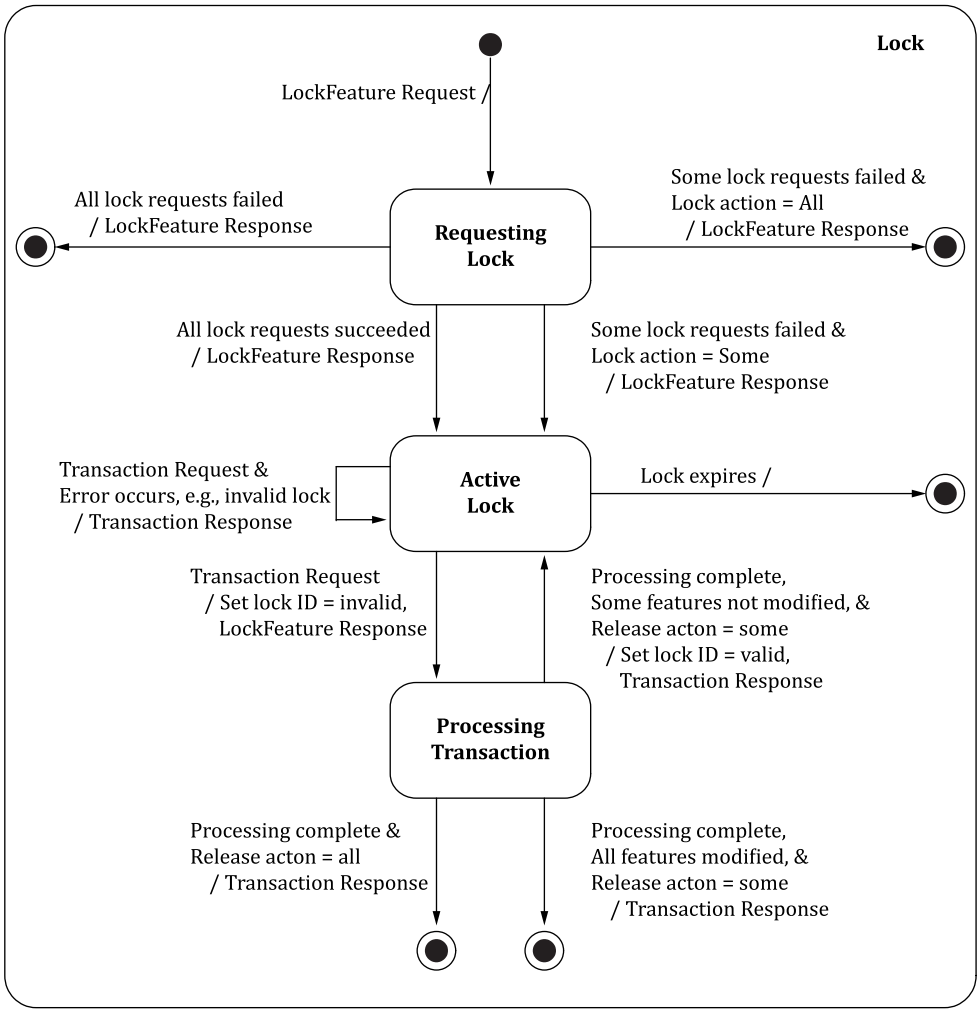


Figure 6 — Locking process during transaction

Figure 6 gives an example on a locking process during transaction (source: ISO 19142).

/rec/computationalviewpoint/ pre_and_post_conditions	The pre-conditions and post-conditions and invariants of an operation should be shown through expressions in OCL.

8.4 Service chaining

8.4.1 General

A model for combining services in a dependent series to perform complex tasks is defined in 8.4.

The syntactic issues of service chaining, e.g. the data structure of a chain, is addressed in 8.4. Examples of service chaining are provided in Annex B.

/rec/computationalviewpoint/servicechaining	Possible composition/construction of services may be shown through service chaining. Review the data structure for a service chain and determine whether it is a directed graph. Determine whether the nodes of the service chains are services. An architecture should provide a means to make a chain extant and transferable between users and for users to evaluate the validity of a chain. An extant service chain is required for the translucent chaining pattern. If an architecture claims to implement the transparent chaining pattern, confirm that a human user is controlling the execution of the chain. If an architecture claims to implement the translucent chaining pattern, confirm that a service chain can be made extant separate from human users and that control of the chain execution is accomplished separate from human users. For the transparent chaining pattern, the architecture should provide the human user with mechanisms to determine services of value, e.g. a service catalogue or service organizer folder.

This International Standard enables users to combine data and services in ways that are not predefined by the data or service providers. This level of data/service interoperability will be achieved in stages. At first, service catalogues will hold entries with tight data/service binding. Eventually, the infrastructure will be available for a user to determine which data can be acted on by a loosely coupled service. This capability will be enabled by the infrastructure of the larger domain of IT.

8.4.2 Anatomy of a service chain

8.4.2.1 UML modelling of a chain

Figure 7 provides a UML model of a chain.

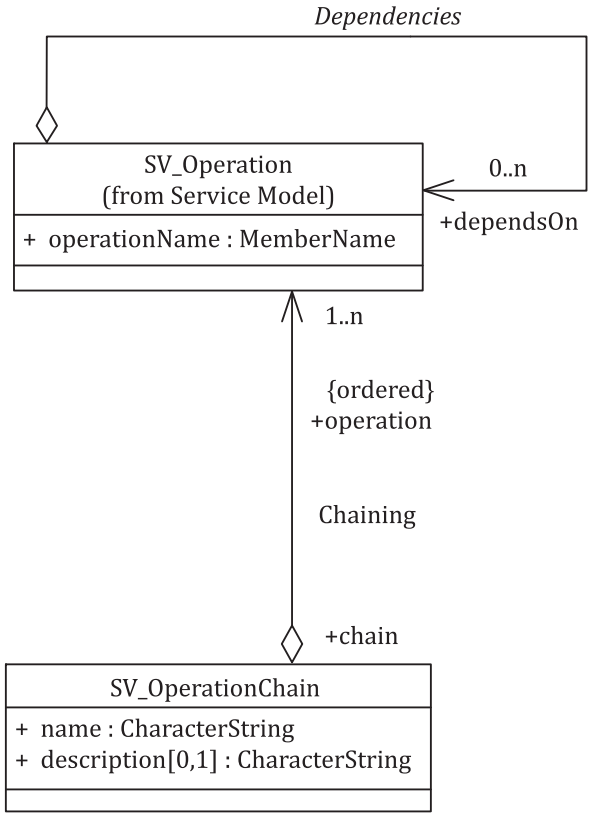


Figure 7 — Basic service chain

In accordance with ISO 19103, the modelling of directed graphs for service chains can be done using the Unified Modeling Language (UML) activity graphs. An activity graph represents the states of executing a service chain. Table 14 identifies the elements of an activity graph. A related approach is to use BPMN as a modelling language for service composition and orchestration. BPMN (see ISO/IEC 19510) has been developed with an explicit aim of being able to support web service composition in BPEL,[45] in addition to the aim of supporting business process modelling from a user point of view.

Table 14 — UML Activity Graph entities

Element of an UML activity graph	Description for service chaining
Activity State	State that represents the execution of a service, typically triggered by the invocation of an operation.
Transition	Relationship between two states. A transition indicates that specified Actions are performed in the first state and the second state is entered when a specified Event occurs and specified Guard Conditions are satisfied.
Branch/Merge	Beginning/end of alternative threads in an activity graph.
Fork/Join	Beginning/end of concurrent threads in an activity graph.
Signal	Asynchronous communication between services intended to trigger transitions in an activity graph.

8.4.3 Service chain modelling

8.4.3.1 Chains as directed graphs

A directed graph is a set of nodes connected by edges, where a direction is associated with each edge.

The action of making the input of one service dependent upon another service leads to treating service chains as directed graphs, where each service is a node in the graph and references to service interactions form the edges. In some cases, the directed graph structure is implicit. In other cases, it is necessary to make the notion of a processing graph explicit and allow such graphs to be considered as entities in their own right.

Explicit representation of a service chain allows the chain to be visually represented and passed to a chain-execution service, e.g. workflow service. When explicitly formed into a data structure, a service node contains two types of information: parameters and sources. Parameters in a service node provide the configurations of the service for the particular chain in which the service class is being used. Sources in a service node indicate the sources of input data to the node.

The arcs of a directed graph can be of several types and these are detailed below as service interactions. The following are some characteristics of directed graphs when used for service chains.

— Cyclic or acyclic directed graph. Directed graphs without loops, i.e. acyclic, are simpler. In some applications, an iterative approach is needed; therefore, the chain will be cyclic with conditions in the control function to address convergence.

— Chains can be considered templates or as immutable graphs. A template is a directed graph that defines the chain based on abstract classes, including identification of each service type. A template can be instantiated, as an immutable graph, at which time the service instances are fixed.

/rec/computationalviewpoint/servicegraphs	Directed graphs should be used for the model of service chains and the characteristics of the chain should be described.

/req/computationalviewpoint/servicenodesandarcs	If directed graphs are used, then — the nodes in the directed graph shall be a representation of service, and — the arcs of the directed graph shall represent the service chain.

/rec/computationalviewpoint/servicenodesandarcs	If directed graphs are used, the following additional elements should be used to characterize the service chain when modelling (the list is not exhausted): — parallel or serial chains; — iteration; — data transport types; — parameters in nodes; — variations in control design pattern.

The following presents the additional elements that should be used to characterize the service chain when modelling:

— parallel against serial chains: Does the directed graph have parallel paths based on branches or are only serial chains permitted? Potential branch types include if/else, merge, switch and trigger.

— iteration: Does a node in the directed graph operate as an iteration, e.g. while and count loops?

— data transport types: Does the directed graph allow variations in the links between nodes reflecting different methods for transporting data or invoking the service?

— parameters in nodes: Do the description of nodes in the directed graph contain parameters that can be changed?

— variations in control design pattern: Pull processing against push processing.

Directed graphs can be modelled using a number of different methods, one of these are UML sequence diagrams. An example for this can be found in 8.4.6, where sequence diagrams are used to model the different architectural patterns.

8.4.4 Services organizer folder

Services are of many types as indicated in 10.1. Only a subset of available services is applicable to a specific situation, e.g. image analysis. A service organizer folder (SOF) is an aid for users in finding services applicable to their situation. A user may construct an SOF and then make that SOF available to other users performing tasks in a similar situation.

A services organizer folder is a data structure that contains references to a set of services that are applicable to a given situation. The SOF need not contain service chains but may contain just individual services.

8.4.5 Services to enable service chaining

Table 15 provides a list of services that are needed to enable service chaining. Details on the services can be found in Clause 10. Some of the services are generic to all IT domains. Other services are specific to geographic data and the large size of geographic datasets.

Table 15 — Services that enable service chaining

Architectural types	Generic IT services	Geographic services
Boundary/human interaction services	— Service-centric service for defining, controlling and providing status information of the service chains — Catalogue-centric service that views and browses metadata about services	— Catalogue-centric service that locates, browses, and manages metadata about spatial data — Spatial-centric service for editing, displaying, querying, and analysing map data — Calculation-centric service allowing viewing and manipulation of geographic data using a spreadsheet format
workflow/task services	— Workflow enactment service to define, invoke, provide status information and control service chaining (interaction with other workflow services, optional) — Service chain validation service — Resource reservation and co-allocation mechanism for both storage system and other resources, such as networks, to support the end-to-end performance guarantee required for predictable transfer	—
processing services	—	— Geographic processing services (see 10.8.5)
model/Information management services	— Service instance metadata catalogue, with discovery and management sub-services — Service type registry, with discovery and management sub-services — Brokering — Mediation	— Geographic dataset instance — Geographic metadata catalogue with discovery, access and management sub-services

Table 15 *(continued)*

Architectural types	Generic IT services	Geographic services
system management service	— Authorization and authentication — Payment methods — Privacy of client — Performance measurement and estimation techniques for key resources involved in data grid operation, including storage systems, networks, and computer — Instrumentation services that enable the end-to-end instrumentation of storage transfer and other operation	—
communication services	— Messaging mechanisms — Large data object transfer — Remote file and executable management: provides access to secondary storage as if it were local — Format conversions	— Geographic format conversions

8.4.6 Architecture patterns for service chaining

8.4.6.1 General

The architecture patterns for service chaining use the structure defined in 5.8.

There are many options for the allocation of service chaining services to components. Different allocation approaches reflect different priorities for different applications: user in the loop against. user supervision. To demonstrate the breadth of the trade space defined by this variation, the following three design patterns are offered that vary the allocation of the control function:

— user-defined (transparent) chaining: the human user manages the workflow;

— workflow-managed (translucent) chaining: in which the human user invokes a workflow management service that controls the chain and the user is aware of the individual services;

— aggregate service (opaque): in which the user invokes a service that carries out the chain, with the user having no awareness of the individual services.

In addition to the difference in visibility of the services to the user, a key distinction between these patterns is the difference in control. In transparent chaining, the control is exclusively with the user. In translucent, a workflow service is present which controls the chain execution, perhaps with oversight by the human. In the aggregate pattern, the aggregate service exclusively performs the control function with no visibility by the user.

8.4.6.2 User-defined (transparent) chaining

8.4.6.2.1 Name

As the name implies, the user defines and controls the order of execution of the individual services. Details of the services are not hidden from the user; hence, the alias for this pattern is Transparent Chaining.

8.4.6.2.2 Problem

In this pattern, the user is knowledgeable about how services can be combined. The user discovers and evaluates the available services, determines their fitness to the need, determines a valid sequence

of services and controls the chaining. This pattern presupposes a knowledgeable user. The user is provided information sufficient to make the control decisions.

8.4.6.2.3 Context

The user refers to a service catalogue in order to discover services of interest. A specific chain does not exist before the user begins. The user has the ability to define a valid chain and/or be able to modify the implied chain if there are failures in execution.

8.4.6.2.4 Forces

The user should be able to design an efficient chain that will execute. The inputs and outputs of the individual services should be compatible, or an intervening service shall be added, e.g. format translation. These patterns assume that each service has sufficient resources to run efficiently, but the user may need to choose services based on network considerations, e.g. network bandwidth, security, authorization. The semantic correctness of the chain is judged by the user; issues such as when data are re-gridded in a chain will affect the validity of the results. A user may iterate a chain until an acceptable result is achieved, resulting in a chain that can be saved and used by others, perhaps using the workflow-chaining pattern.

8.4.6.2.5 Structure

The user-defined chaining architecture pattern is shown in Figure 8. The steps are described in Table 16.

NOTE The unique feature of the transparent pattern is that the chain is defined and controlled by the user. In the figure, the user discovers an available service through a catalogue service. Alternatives for the user to select services are part of this pattern. For example, a service organizer folder could be substituted for the catalogue.

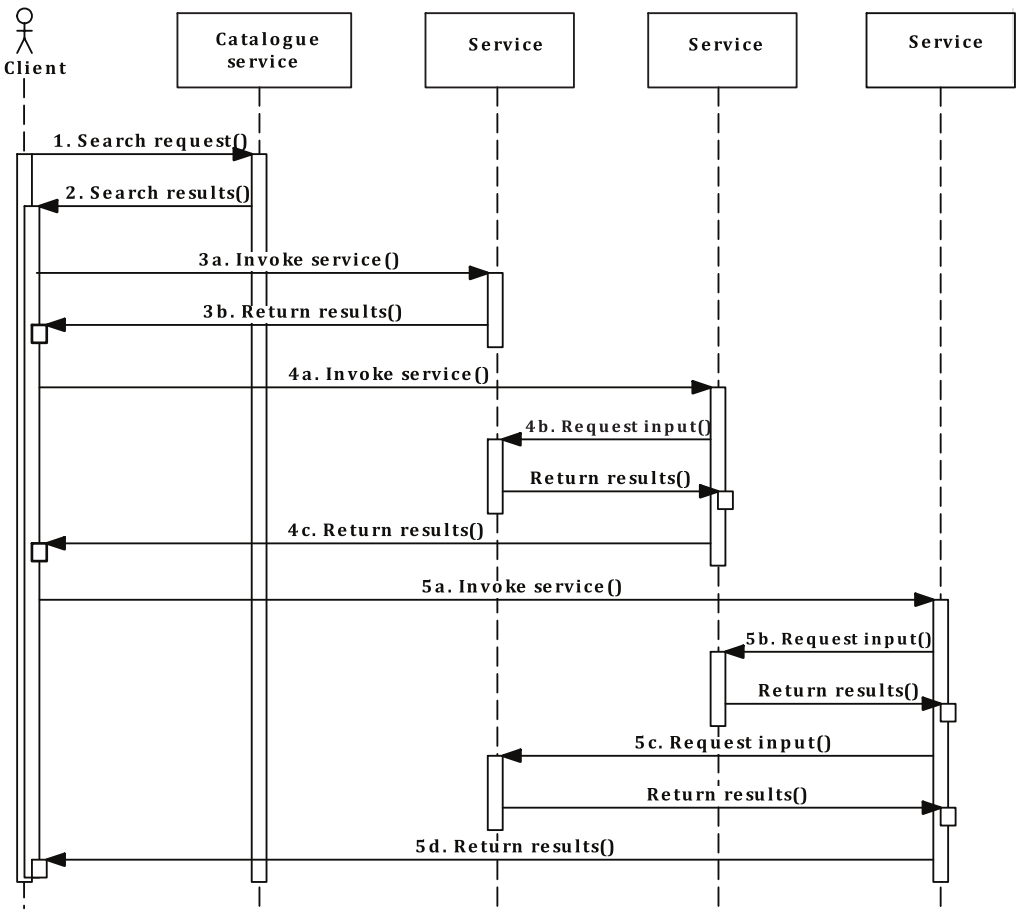


Figure 8 — Transparent chaining — User-defined chaining architecture pattern

Table 16 — Description of steps in Figure 8

Step 1. Search request	A human uses a client to send a search request (or series of searches) to a catalogue service. The catalogue service provides queries on service metadata.
Step 2. Search results	Catalogue service returns metadata about services of interest to the user. For this example, the user has found three services that will be chained.
Step 3a. Invoke service Step 3b. Return results	User invokes a service using the client, causing a result to be available for a subsequent service. Results are returned to the client.
Step 4a. Invoke service Step 4b. Request input Step 4c. Return results	User invokes a second service using the client. The request includes a reference to the results from the previous step. The service creates a result that is available for the next service. Results are returned to the client.
Step 5a. Invoke service Step 5b. Request input Step 5c. Request input Step 5d. Return results	User invokes a third service using the client. The request includes references to the two previous services. This third service returns a result to the client. Results are returned to the client.

8.4.6.3 Workflow-managed (translucent) chaining (Orchestration)

8.4.6.3.1 Name

As the name implies, in this pattern, the execution of the chain is managed by a workflow service (or multiple workflow services). The user's involvement in the steps of the chain is mostly one of watching the chain execute the individual services that are apparent to the user, hence the alias of translucent chaining. A key distinction for this pattern is the existence of a defined chain prior to the user executing the pattern.

This pattern is similar to that of orchestration that is well known in the Information Technology community. An orchestration defines the sequence and conditions in which one service invokes other services in order to realize some useful function, that is, an orchestration is the pattern of interactions that a Web service agent shall follow in order to achieve its goal

8.4.6.3.2 Problem

In this pattern, the user relies on a workflow service to execute a predefined chain of services. The user has determined an existing chain assumed to produce results of interest to the user. The user may need to provide parameters particular to specific instance, but relies on the workflow service to carry out the chain.

8.4.6.3.3 Context

The user knows of a workflow service and has selected a chain of interest. The user interacts with the workflow service to execute the chain including providing parameters specific to the data instances of interest to the user.

8.4.6.3.4 Forces

To reduce the user's workload, the workflow service handles details of the distributed computing aspects of executing the chain. Although the predefined chain is assumed to have a degree of semantic validity, by evaluating interim results, the user can evaluate the semantic validity of the specific instance of this processing. For example, for a service that includes an iterative algorithm, the user may need to judge if convergence to a sufficient degree of accuracy has been achieved.

8.4.6.3.5 Structure

The workflow-managed (translucent) chaining architecture pattern is shown in Figure 9. The steps are described in Table 17.

There may be multiple workflow services. If there is more than one, the workflow services should coordinate to carry out the predefined chain. In the extreme case, each service in the chain contains a workflow service and the chain is passed along with the service results. The unique features of the translucent pattern are the existence of a predefined chain and the user's awareness of the chain.

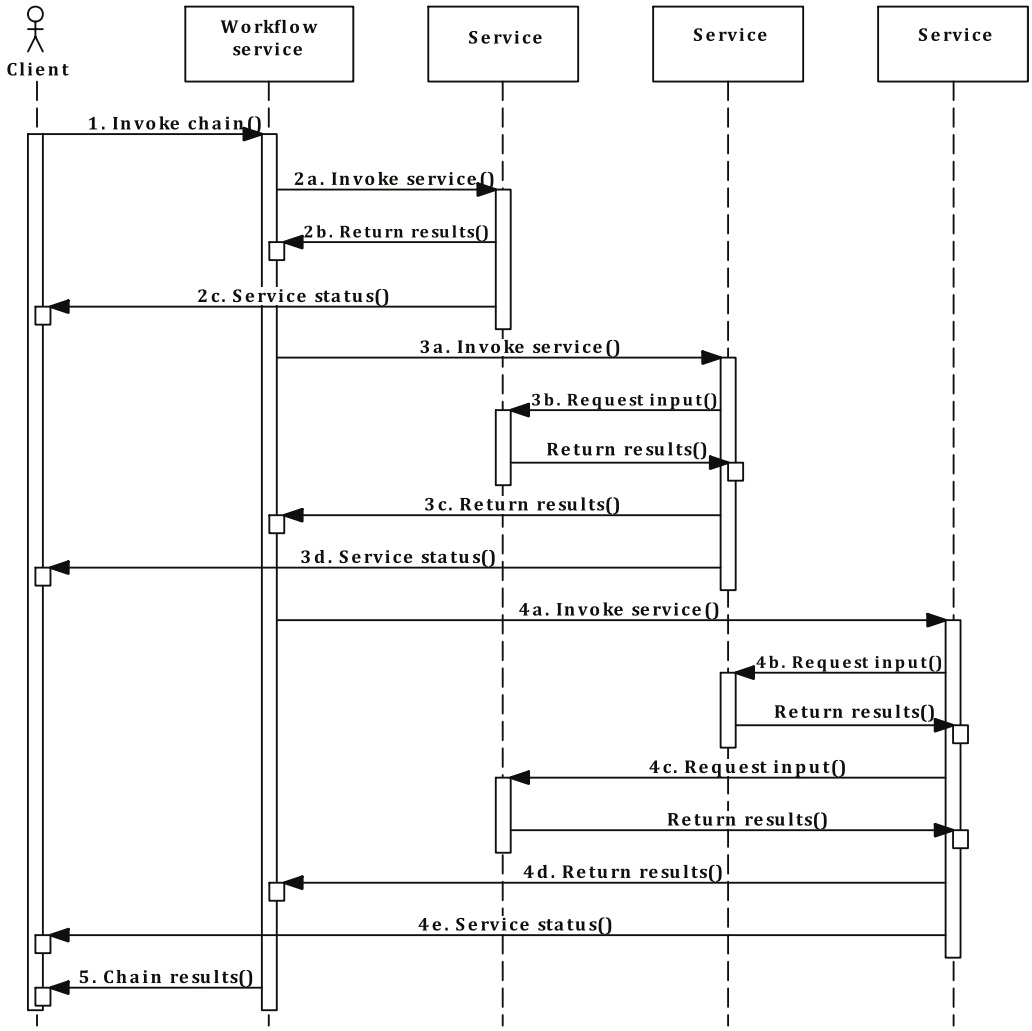


Figure 9 — Translucent chaining — Workflow-managed chaining architecture pattern

Table 17 — Description of steps in Figure 9

Step 1. Invoke a chain	A human uses a client to request that a workflow service execute a chain. The user may be allowed to modify some aspects of the chain prior to execution.
Step 2a. Invoke service Step 2b. Return results Step 2c. Service status	The workflow service determines the services in the chain and invokes the first service. The service informs the workflow service of the completion of the task. Return the results to the workflow service. Status of the service may be provided directly to the client. The client may stop the workflow.

Table 17 *(continued)*

Step 3a. Invoke service Step 3b. Request input Step 3c. Return results Step 3d. Service status	Upon notification of completion of the first service, the workflow service determines the next service in the chain and invokes it. The second service requests results from the first service. The service informs the workflow service of the completion of the task. Return the results to the workflow service. Status of the service may be provided directly to the client. The client may stop the workflow.
Step 4a. Invoke service Step 4b. Request input Step 4c. Request input Step 4d. Return results Step 4e. Service status	Upon notification of completion of the second service, the workflow service determines the next service in the chain and invokes it. The third service requests results from the first and second services. The service informs the workflow service of the completion of the task. Return the results to the workflow service. Status of the service may be provided directly to the client. The client may stop the workflow.
Step 5. Chain results	Upon notification of completion of the last service, the workflow service informs the client of the completion of the chain.

8.4.6.4 Aggregate service (opaque-chaining - Choreography)

8.4.6.4.1 Name

As the name implies, in this pattern, the services appear as a single service that handles all coordination of the individual services behind the aggregate service. The user has no awareness that there is a set of services behind the aggregate, hence the alias of opaque chaining.

This pattern is similar to that of choreography that is well known in the Information Technology community. Choreography is the definition of the sequences and conditions under which multiple cooperating independent agents exchange messages in order to perform a task to achieve a goal state.

8.4.6.4.2 Problem

In this pattern, the user relies on an aggregate service to execute a predefined chain of services. The user has discovered the aggregate service and may have no knowledge of how the aggregate accomplishes the service. The user may need to provide parameters particular to the specific instance, but relies on the aggregate service to carry out the chain.

8.4.6.4.3 Context

The user knows of an aggregate service, perhaps not knowing that a chain of services implements the aggregate. The user interacts with the aggregate service to execute the chain including providing parameters specific to the data instances of interest to the user.

8.4.6.4.4 Forces

To reduce the user's workload, the aggregate service handles all details of the multi-service aspects of executing the chain. Although the aggregate service chain is assumed to have a degree of semantic validity, by evaluating interim results, the user can evaluate the semantic validity of the specific instance of this processing. For example, for a service that includes an iterative algorithm, the user may need to judge if a sufficient degree of convergence has been achieved. These intermediate results need not reveal the underlying services.

8.4.6.4.5 Structure

The aggregate service (opaque-chaining) architecture pattern is shown in Figure 10. The steps are described in Table 18.

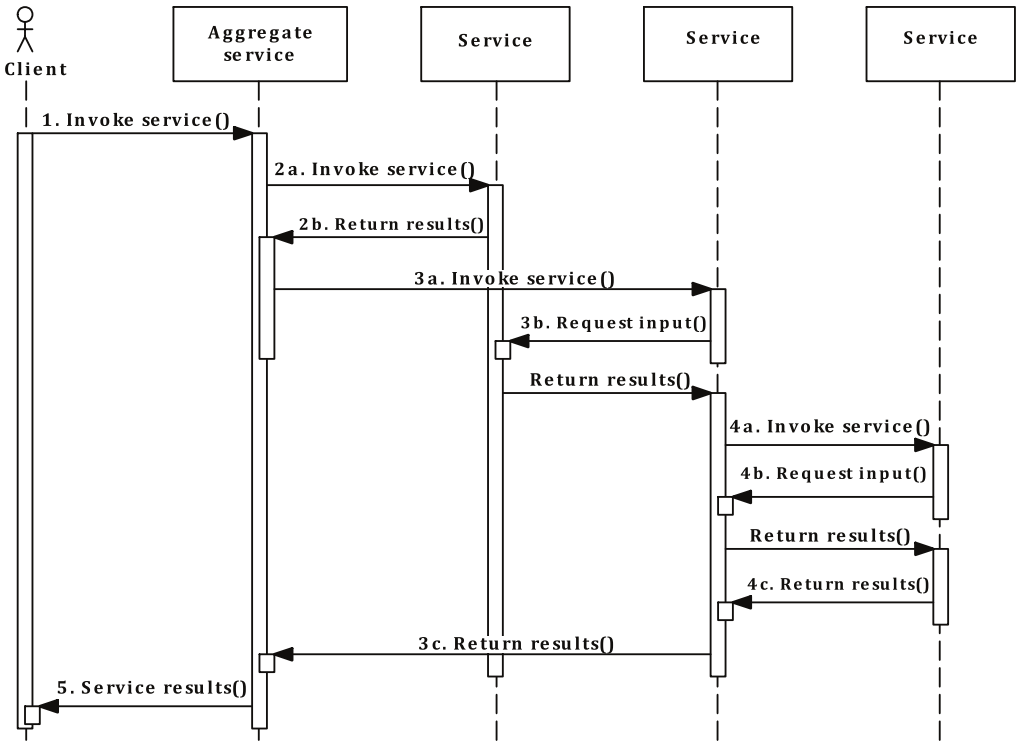


Figure 10 — Opaque chaining

Table 18 — Description of steps in Figure 10

Step 1. Invoke service	A human uses a client to request that an aggregate service executes a chain. The service will appear to the user as a single service (the user might be unaware of the service chain).
Step 2a. Invoke service Step 2b. Return results	The aggregate service determines the services in the chain and invokes the first service. The service informs the aggregate service of the completion of the task and returns the results to the aggregate service.
Step 3a. Invoke service Step 3b. Request input Step 3c. Return results	Upon notification of completion of the first service, the aggregate service determines the next service in the chain and invokes it. The second service request results from the first service. The service receives the input from the first services. The (second) service invokes a third service, without the knowledge of the aggregated services (actions specified in Step 4 are performed). After the third service has completed its tasks, the second service is informed. The (second) service then informs the aggregate service of the completion of the tasks and returns the results to the aggregate service.
Step 4a. Invoke service Step 4b. Request input Step 4c. Request input Step 4d. Return results	Upon notification of completion of the second service, the aggregate service determines the next service in the chain and invokes it. The third service request results from the first and second services. The service informs the second service of the completion of the task and returns the results to the second service.
Step 5. Chain results	Upon notification of completion of the last service, the aggregate service informs the client of the completion of the chain and returns a result to the client.

8.4.7 Variations on chaining patterns

The five chaining patterns in Figure 10, and further explained in Table 18, could be combined in a variety of ways.

Each of the lowest level services shown in the pattern diagrams could in turn implement a chain. This is recursive composition of services supported by the opaque pattern. A service chain can become a new service. The ability to define recursive composition of services provides scalability and support for top-down progressive refinement, as well as for bottom-up aggregation.

The patterns could be used to define how a library of chains is constructed. A knowledgeable user could build chains using the transparent pattern. Through iterative use of the transparent pattern, a chain is constructed that produces valid results. Chains are then made available for wider use following the translucent pattern. Certain chains may become routinely used and an aggregate service is built as an interface.

An example need for a translucent or opaque chaining pattern occurs in decision support. The decision-maker is an individual using decision-support aids to help make a decision. An example of a decision-support aid is a service chain. The decision-support aid developer is an individual who "integrates" chains of services into decision-support aids.

Another type of service interaction can be considered as chaining, where a user makes a request of a lead service and the lead service then invokes a secondary service which invokes a tertiary service. Each of the services responds to the request when it has sufficient information from the underlying services. In this way, there is no explicit chain but rather a chain is implied.

8.5 Service metadata

/req/computationalviewpoint/servicemetadata	Service metadata shall be described according to ISO 19115-1:2014, 6.5.14.

Service metadata are related to Quality of Service (QoS).

8.6 Simple service architecture

The following simplifying assumptions should be considered when implementing a message-based architecture to support service chaining. Systems claiming to be instances of simple service architecture should comply with the following.

— Message-operations. For simplicity, it is desirable to model the operations as messages. A message operation consists of a request and response. Requests and responses contain parameters as the payload, which is transferred in a uniform manner independent of content. Simple applications are characterized by message exchange patterns such as one-way (or event) and two-way (or synchronous) request response interactions. A service specification should make such simple exchange applications as easy as possible to create and to use.

— Separation of control and data. A client controlling a service may not want the full results of the service. For example, the user may have no need for the potentially voluminous intermediate products in a service chain. Only the final result of a service chain may be needed by the client. Therefore, operations of an interface should separate the control of the service from the access to the data resulting from a service. A client should have the option of receiving just the status of an operation and, separately, the data should be accessible through a separate operation.

— Stateful service against stateless service. For simplicity, it is desired that a service be stateless, i.e. that a service invocation be composed of a single request-response pair with no dependence on past or future interactions. This will not always be possible. For some services, preconditions should be set and iteration may be required; then it will be necessary to model the service with a state diagram having multiple states. Transitions between the states are triggered by operations.

— Known service type. All service instances are of specific service types and the client knows the type prior to runtime. Clients shall contain software for accessing the service type prior to encountering service instances of the type in an implemented architecture. The assumption is that the client knows the service types.

— Adequate hardware. The services described in this International Standard are software implementations running on hardware hosts. This International Standard assumes that the issues of hardware hosting of the software are transparent to the user. It is assumed that the service has adequate hardware, i.e. hardware assignment is transparent to user.

8.7 Examples of relevant standards

— Service-oriented architecture Modeling Language (SoaML) (see http://www.omg.org/spec/SoaML/1.0/). This is a standard from OMG that provides a UML profile and a metamodel for the modelling of services. All major UML tool vendors support it, but the support for the collaboration modelling part of SoaML has had less tool features and is thus not required for use within this International Standard.

— UML4ODP Computational specification profile (ISO/IEC 19793). ISO/IEC 19793 provides a UML profile for all of the main concepts defined in the RM-ODP computational viewpoint. It is a valuable reference and foundation for doing full RM-ODP computational viewpoint modelling but for the geospatial community, it has been suggested to do have a more lightweight approach.

8.8 Examples and tools: Service modelling with SoaML

The Service-oriented architecture Modeling Language (SoaML) specification defines a UML profile and a metamodel for the design of services within a service-oriented architecture. The goals of SoaML are to support the activities of service modelling and design and to fit into an overall model-driven development approach, supporting SOA from both a business and an IT perspective. Within this International Standard, it is, in particular, the part of SoaML for the specification of simple interfaces and complex service interfaces with use of message types that is being used.

A further description of the metamodel of SoaML can be found in Annex F.

9 Information viewpoint: A basis for semantic interoperability

9.1 Information model interoperability and the information viewpoint

Achieving *information model interoperability* is one of the main goals of the ISO geographic information suite of standards. Many of the other International Standards in this suite, i.e. ISO 19107, ISO 19115-1, etc., are primarily focusing on defining the content of the information that is being processed by the services and exchanged between services. The information viewpoint is defined in the ODP to include a static information model and a dynamic information model. The semantics of service interactions, e.g. what services make sense to chain, are developed in 8.4.

The Information Model describes the data which comprises the inputs and outputs of the service and its operations. Information content representations (logical description) of the inputs and outputs should be represented in a platform neutral way, conformant with ISO 19103 and ISO 19109 and other relevant standards in the ISO geographic information suite of standards.

To be able to interoperate in the information viewpoint, two systems shall be *information model interoperable*. To achieve information model interoperability, the two systems shall be both *syntactically interoperable and semantically interoperable*.

— Syntactically interoperable: two systems are syntactically interoperable if they use the same structure for the information that flows between the systems and is processed by the systems.

— Semantically interoperable: two systems are semantically interoperable if they have a common understanding of the semantics of the information that flows between the systems and is processed by the systems.

The common structural models with use of the general feature model (GFM) address syntactic interoperability. In the ISO geographic information suite of standards, the models based on GFM allows

for representation of various types of features, all having the same structure. To achieve semantic interoperability for feature types, it is, in addition, necessary to match or make mappings between feature-type definitions from feature-type catalogues. This issue does not fall in the scope of this International Standard but is introduced in ISO/TS 19150-1, to be further developed in ISO 19150-3[2)].

The information viewpoint in ISO RM-ODP describes the information that flows in a system and is processed by a system. It focuses on feature type definition, i.e. the definition of geographic concepts, their properties, and the relation between geographic concepts. As such, the GFM is the metamodel for feature type definition which lead to development of application schemas. An application schema may be exposed in various languages; ISO 19109 provides UML rules while ISO 19150-2 provides OWL rules. An application schema can be further in a physical model which describes how feature type are stored in databases and communicated between system components

The information viewpoint is also concerned with the semantics of the information processing. Each particular service will need to define its syntactical interfaces through operations and its semantics through description of the meaning of the operations and their legal sequencing. The latter can be done through pre-conditions and post-conditions and invariants in OCL, and by UML state diagrams, as described in 8.3.

9.2 Information viewpoint Service specifications

The requirements for creating the service specification part for the information viewpoint are formalized as a requirements class summarized in Table 19.

Table 19 — Requirements class for Information viewpoint service specifications

Requirements class	**/req/informationviewpoint**
Target type	UML service model
Dependency	ISO 19103 (Conceptual schema language)
Requirement	/req/informationviewpoint/servicemodel dependencies
Requirement	/req/informationviewpoint/operation input/output/exception parameters

/req/informationviewpoint/ service model dependencies	The service model shall be shown as a UML package with dependencies to other UML models that are used.

2) This International Standard is under development.

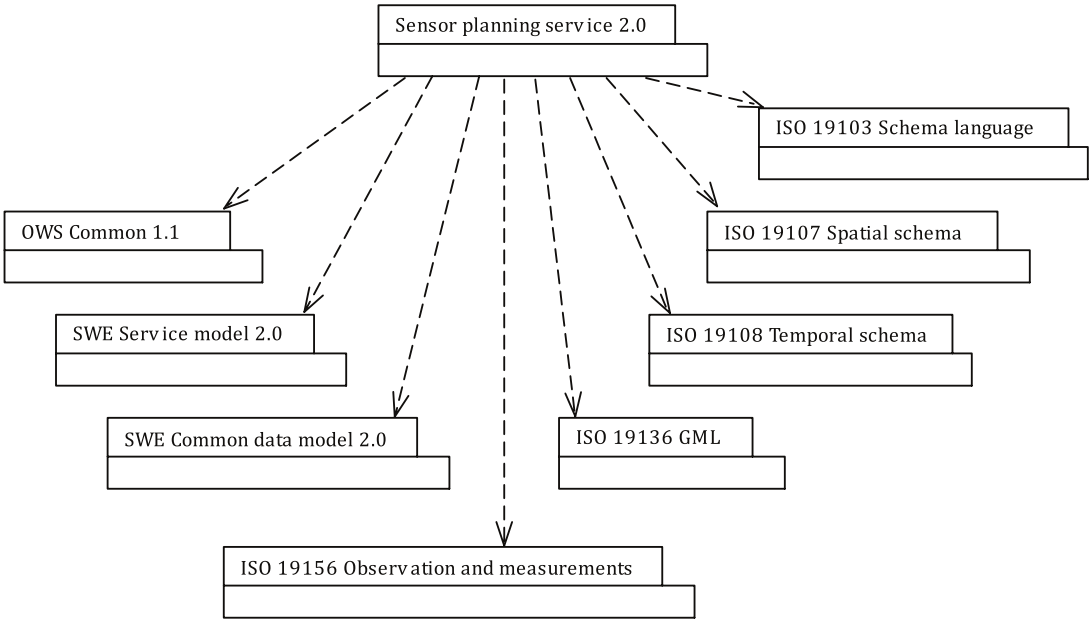


Figure 11 — Service model dependencies

Figure 11 shows service model dependencies for a Sensor Planning Service.

A service Operation is any Operation of an Interface provided or required by a Service or Request. Service Operations may use two different parameter styles: document-centred (or message-centred) or RPC (Remote Procedure Call)-centred. Document-centred parameter style uses MessageType for ownedParameter types and the Operation can have, at most, one in, one out and one exception parameter (an out parameter with isException set to true). All parameters of such an operation shall be typed by a MessageType.

/req/informationviewpoint/ operation input/output parameters	The input/output parameters and exceptions of operations in the service interfaces (service payloads) shall be described as regular UML types for classic RPC centred parameter style or with use of <<MessageTypes>> for document/message-centred parameter style, using information models according to ISO 19103 and ISO 19109 and further in a UML model data dictionary table for each operation with name, definition, data type/value and multiplicity/use for each element.

For service models not using document/message-centred style, other classes with other stereotypes (like featureType) is allowed as input/output parameters.

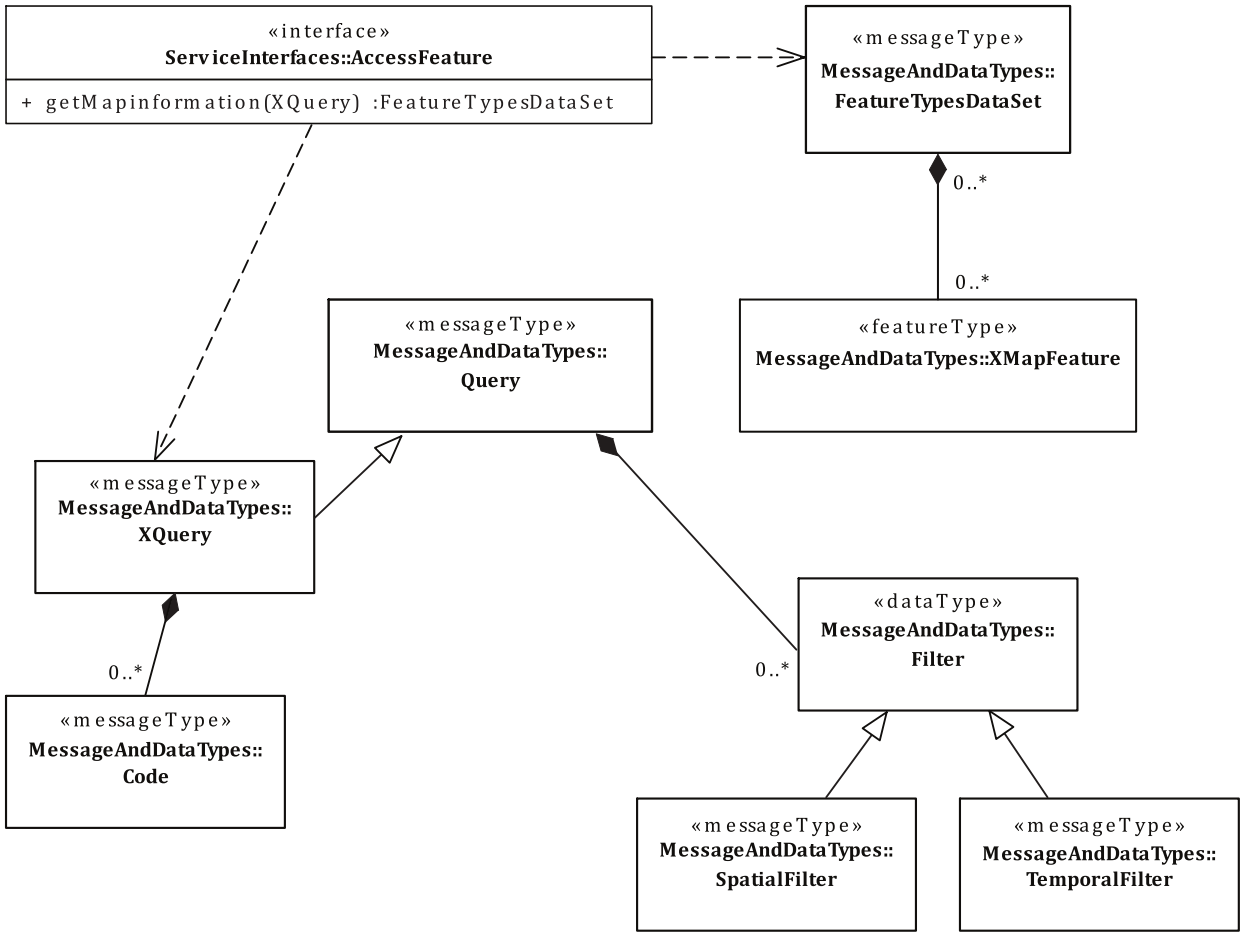


Figure 12 — Operation Input/Output parameters defined as MessageTypes for document-centred parameter style

Figure 12 shows input/output parameters defined for the operation getMapInformation, with XQuery input parameter and FeatureTypeDataSet as output parameter (result) defined through MessageTypes.

Input/output parameters for operations with a document/message centred parameter style are modelled using the << MessageType>> concept from SoaML [ref SoaML], as described in 9.2.

A MessageType is a kind of value object that represents information exchanged between participant requests and services. This information consists of data passed into, and/or returned from, the invocation of an operation or event signal defined in a service interface. A MessageType is in the domain or service-specific content and does not include header or other platform specific details, implementation or protocol-specific information.

There will be a transformation from the specification of a MessageType in the information viewpoint to the representation and encoding of the corresponding data in a platform-specific specification and implementation in the technology viewpoint.

MessageTypes represent service data exchanged between service consumers and providers. Service data are often a view (projections and selections) on information or domain class models representing the (often persistent) entity data used to implement service participants.

MessageTypes are used to aggregate inputs, outputs, and exceptions to service operations as in WSDL. MessageTypes represent “pure data” that may be communicated between parties. It is then up to the parties, based on the SOA specification, to interpret this data and act accordingly. As “pure data” message types may not have dependencies on the environment, location, or information system of either party. This restriction rules out many common implementation techniques such as “memory pointers,” which

may be found inside of an application. Good design practices suggest that the content and structure of messages provide for rich interaction of the parties without unnecessarily coupling or restricting their behaviour or internal concerns. The terms Data Transfer Object (DTO), Service Data Object (SDO), or value objects used in some technologies are similar in concept, though they tend to imply certain implementation techniques. A DTO represents data that can be freely exchanged between address spaces, and does not rely on specific location information to relate parts of the data. An SDO is a standard implementation of a DTO. A Value Object is a Class without identity and where equality is defined by value not reference. Also, in the business world (or areas of business where EDI is commonplace), the term Document is frequently used. All these concepts can be represented by a MessageType.

MessageType should generally only be applied to DataType since it is intended to have no identity. However, it is recognized that many existing models do not clearly distinguish identity, either mixing Class and DataType, or only using Class. Recognizing this, SoaML allows MessageType to be applied to Class, as well as DataType. In this case, the identity implied by the Class is not considered in the MessageType. The Class is treated as if it were a DataType.

The MessageType will be able to aggregate content specified based the models from the various ISO geographic information standards, with a foundation in the modelling approaches from ISO 19103 and ISO 19109.

It is possible to do a mapping to a RPC style of operations, as well as to other communication styles. Where a service Operation may have any number of in and out parameters and may have a return parameter as in UML2. This can be done based on rules specified in the operation mappings for the engineering and/or technology viewpoints. In this context, the MessageTypes should be viewed as an abstract specification with potentially different implementation technologies. Transformations to different encodings can be handled according to the encoding framework of ISO 19118, with ISO 19136 as an example for how to realize the ISO 19118 framework for the GML specification based on XML.

It is the intent of message type that it represents data values that can be sent between participants. Where message types contain classes as attributes or aggregated associations, the message type will contain a “copy by value” of the public state of those objects. Where those objects contain references to other objects, those references will likewise be converted to value data types.

Attachments are used to model elements that have their own identity when they are taken out of the system. An Attachment denotes some component of a message that is an attachment to it (as opposed to a direct part of the message itself). In general, this is not likely to be used greatly in higher level design activities, but for many processes attached, data are important to differentiate from embedded message data. For example, a catalogue service may return general feature data as a part of the structured message but coverage/image as attachments to the message; this also allows us to denote that the encoding of the coverage/image can be binary (as opposed to the textual encoding of the main message). Attachments may be used to indicate part of service data that can be separately accessed, reducing the data sent between consumers and providers unless it is needed.

NOTE Work is ongoing in ISO/TC 211 to provide instructions and tooling support for automatic creation of relevant UML model documentation in terms of a producing table from the model dictionary with descriptions for model elements, including Name, definition, data type and value, multiplicity and use. Once this becomes available, it is assumed that such tables will be produced automatically from the actual UML models as part of the detailed documentation of the models.

10 Service taxonomies

10.1 Need for multiple service taxonomies

This subclause contains a description of taxonomy of various services. There exist multiple possible taxonomies for services based on various classification dimensions. The purpose of defining taxonomies in this International Standard is to have ways of identifying geographic services

There are many ways to classify services, depending on the selected perspective. The taxonomy introduced in the first version of this International Standard created a taxonomy based on an architectural reference

model. Later, a need has been seen for classifying services based on a usage life cycle perspective, for instance, as the classification into one more of the following service types: “discovery”, “view”, “download”, “transformation”, or “invoke”. Sometimes, a service is a composition from other services and represents an aggregate that might include more than one basic service. In such situations, it might be relevant to categorize the aggregated services to be of more than one type within the taxonomy. The need for, and evolution of, service taxonomies might evolve and change in the same way as codelists, and it needs thus a more flexible representation than being represented as only one type.

This International Standard presents two different service taxonomies, based on an architectural perspective and a life cycle perspective. A service shall be classified according to at least one of these taxonomies. The representation of this will be done in the service metadata, as described in ISO 19115-1.

10.2 Service taxonomies and requirements

The requirements for classifying a service related to one or more service taxonomies are formalized as a requirements class summarized in Table 20.

Table 20 — Requirements class for Service taxonomies

Requirements class	**/req/servicetaxonomies**
Target type	Service description
Dependency	ISO 19103 (Conceptual schema language)
Requirement	/req/servicetaxonomies/service type – architecture
Requirement	/req/servicetaxonomies/service type - lifecycle
Recommendation	/rec/ servicetaxonomies /service type - user-defined

10.3 Architectural reference model

The *Architectural reference model* defines a structure for *geographic information services* in the context of generic IT services. The basis for the *Architectural reference model* has been an analysis of various kinds of reference models for system interfaces in the IT domain focusing on main IT architectural elements and how these potentially might need to be extended to support the special needs of geospatial systems and services. The Architectural reference model was described previously in ISO 19101-1, but has now been moved to this International Standard, because of its close relationship to service taxonomies.

10.4 Definition of the Architectural reference model

The *Architectural reference model* defines a structure for *geographic information services* and a method for identifying standardization requirements for those services. This model provides an understanding of what types of services are defined in the ISO geographic information suite of standards and distinguishes these services from other information technology services. The *Architectural reference model* shows how to determine which aspects of geographic information will need to be standardized to support the operation of those services. Thus, the model provides guidance to the program of standardization undertaken in the ISO geographic information suite of standards, as well as for geographic services in general. Other standards bodies that are standardizing geographic information may also consult the *Architectural reference model* for guidance.

10.5 Uses of the Architectural reference model

The *Architectural reference model* is intended for developers of geographic information services, for GIS developers and for GIS users. This model

— defines classes of information technology services, providing a framework for identifying individual *geographic information services*, and

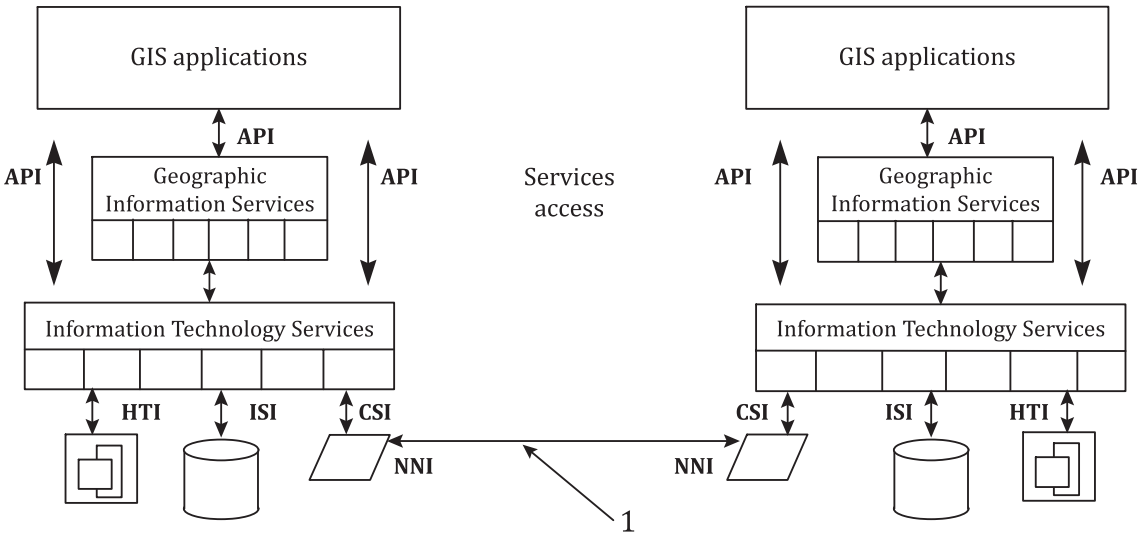
— provides a method for determining requirements for specification of services related to *geographic information*.

The *Architectural reference* model ensures an integrated view on geographic services. Service developers who need to ensure consistency with the ISO geographic information suite of standards should refer to Clause 10 to identify which services are supported by their specification. These services should fall within the six classes of geographic information services described in Clause 10, Figure 14.

10.6 Overview of the Architectural reference model

10.6.1 Services and service interfaces

The *Architectural reference model* shown in Figure 13 also shows how GIS applications utilize capabilities provided by services.



Key

1 Data sharing and transfer based on common conceptual models
API Application Programming Interface
HTI Human Technology Interface
ISI Information Services Interface
CSI Communications Services Interface
NNI Network to Network Interface

Figure 13 — Architectural reference model

The diagram shows application systems and services residing at different computing sites linked by a network. *Services* are capabilities provided for manipulating, transforming, managing, or presenting information. *Service interfaces* are boundaries across which services are invoked and across which data are passed between a service and an application, external storage device, communications network, or a human being. The diagram shows the following four interfaces.

— **Application Programming Interface** (API) is the interface between services and application systems. This is the interface used by application systems to invoke *geographic information services*.

— **Communications Services Interface** (CSI) is the interface across which applications and services access data transport services to communicate across a network. Different computing networks may be connected through a special interface known as the network-to-network interface (NNI).

— **Human Technology Interface** (HTI) allows the human end user to access the computing system. This interface includes graphic user interfaces and keyboards.

— **Information Services Interface** (ISI) is a boundary across which database services are provided, allowing persistent storage of data.

10.6.2 Identifying services and service interfaces for geographic information

The architectural Reference Model for geographic information has two key aspects.

— The separation of *geographic information services* from more generic information technology services defines capabilities that are specific to the manipulation, transformation, storage and exchange of geographic information. Figure 13 shows the separation of *geographic information services*. In 10.7.2, the *Architectural reference model* describes six classes of geographic information services. The standards in the ISO geographic information suite of standards define specific services within these classes.

— Service interfaces provide access to geographic information services and enable exchange of data between services and service users, information storage devices and networks. The *Architectural reference model* identifies general types of interfaces that are used by geographic information services. The purpose of this method is to guide the standardization of geographic information in order to enable the interoperability of GIS in distributed computing environments.

The definition of service interfaces enables a variety of applications with different levels of functionality to access and use geographic information. While specialized services will remain an area for proprietary products, the interfaces to those services will be standardized. Geographic information system and software developers will use these standardized interfaces to define and implement geographic information services.

10.7 Types of geographic information services

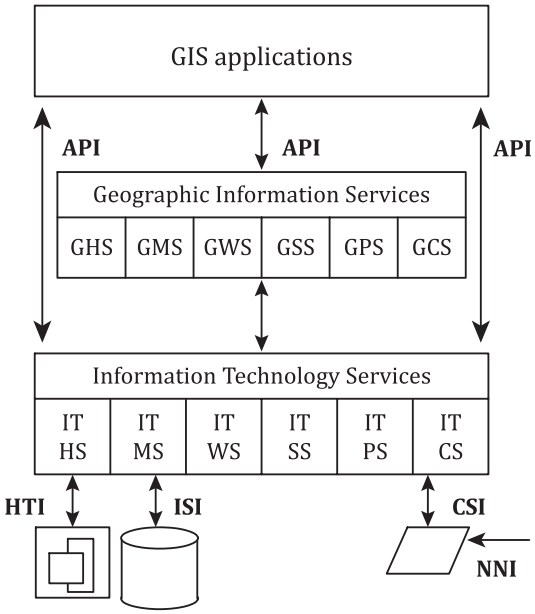
10.7.1 Requirement for service taxonomy

/req/servicetaxonomy/ service type – architecture	A service shall be categorised to belong to one or more (for an aggregated service) of the following service architecture types for geographic information services: human/boundary interaction, model/information management, workflow/task management, processing-(spatial – thematic or -temporal), communication and/or management/security.

There are six classes of information technology services that are important for geographic information. Further detail is provided on the extension of each of these classes for geographic information, see 10.7.2.

10.7.2 Types of information technology services relevant to geographic information

This International Standard identifies six classes of generic information technology services of particular importance for geographic information. Each of these classes provides a basis for definition of services that are specific to geographic information. These classes and their extensions for geographic information are depicted in Figure 14 and defined below.



Key

G	Geographic	WS	Workflow/Task Services
IT	Information Technology	SS	System Management Services
HS	Human Interaction Services	PS	Processing Services
MS	Model Management Services	CS	Communication Services

NOTE The approach is to define Geographic Information Services in each of the six groups, where general Information Technology services do not meet the requirements.

Figure 14 — Six classes of services

— **Model/Information Management Services** are services for management of the development, manipulation and storage of metadata, conceptual schemas and datasets.

— **Human/Boundary Interaction Services** are services for management of user interfaces, graphics, multimedia and for presentation of compound document, user interface dialogues and interaction with other system boundary elements like physical sensors.

— **Workflow/Task Services** are services for support of specific tasks or work-related activities conducted by humans. These services support use of resources and development of products involving a sequence of activities or steps that may be conducted by different persons.

— **Processing Services** are services that perform large-scale computations involving substantial amounts of data. Examples include services for providing the time of day, spelling checkers and services that perform coordinate transformations (e.g. that accept a set of coordinates expressed using one reference system and converting them to a set of coordinates in a different reference system). A processing service does not include capabilities for providing persistent storage of data or transfer of data over networks.

— **Communication Services** are services for encoding and transfer of data across communications networks.

— **System Management Services** are services for the management of system components, applications and networks. These services also include management of user accounts and user access privileges.

Not every information technology service needs to be changed or specialized to be useful for processing geographic information. The different standards in the ISO geographic information suite of standards

should indicate whether a service is a generic information technology service or whether it is specialized for geographic information.

10.7.3 Extension of service types for geographic information

The six service classes identified in 10.7.2 may be extended to define classes of geographic information services. Standards in the ISO geographic information suite of standards that address these specializations are identified below.

— **Geographic Information Model/Information Management Services**. The specialization of this class of services focuses on management and administration of geographic information, including conceptual schemas and data. Specific services within this class are identified in this International Standard. These services are based on the content of those standards in the ISO geographic information suite of standards that standardize the structure of geographic information and the procedures for its administration, including ISO 19107, ISO 19108, ISO 19109, ISO 19110, ISO 19111, ISO 19112, ISO 19115-1 and ISO 19157. Examples of such services are a query and update service for access and manipulation of geographic information and a catalogue service for management of feature catalogues.

— **Geographic Information Human/Boundary Interaction Services**. This class of services focuses on providing capabilities for managing the interface between humans and Geographic Information Systems, and other system boundaries like physical sensors. This class includes graphic representation of features, described in ISO 19117.

— **Geographic Information Workflow/Task Management Services**. The specialization of this class of services focuses on workflow for tasks associated with geographic information, involving processing of orders for buying and selling of geographic information and services.

— **Geographic Information Communication Services**. The specialization of this class of services focuses on the transfer of geographic information across a computer network. Requirements for Transfer and Encoding services are found in ISO 19118.

— **Geographic Information Processing Services**. The specialization of this class of services focuses on processing of geographic information. ISO 19116 is an example of a processing service. Other examples include services for coordinate transformation, metric translation and format conversion.

— **Geographic Information System Management**. The specialization of this class of services focuses on user management, security and performance management.

10.8 Geographic architecture services taxonomy

10.8.1 Geographic architecture services taxonomy requirements

The geographic services taxonomy consists of the titles of the categories (see Table 21) and the definitions for the categories. Systems compliant to this International Standard shall use the geographic services taxonomy to organize their services. A specific service shall be categorized in one and only one category, unless it is an aggregate service that may perform services from more than one category which then allows the aggregated service to belong to multiple categories.

It is not required that a system provides any service listed. It is required that if a system provides a service named in 10.8 that the service shall be categorized as defined. A service catalogue compliant with this International Standard shall categorize service metadata instances in the categories of the geographic service taxonomy.

If a service uses the name of an example service, the service provides the functionality that is defined. For example, if a service entitled “catalogue viewer” is provided, it performs the services defined for the catalogue viewer in the geographic human interaction services category. Systems providing services should name services as found in the service examples.

Table 21 — Geographic architecture services taxonomy

— Geographic boundary/human interaction services
— Geographic model/information management services
— Geographic workflow/task management services
— Geographic processing services
— Geographic processing services — spatial
— Geographic processing services — thematic
— Geographic processing services — temporal
— Geographic processing services — metadata
— Geographic communication services
— Geographic system management and security services

10.8.2 Geographic boundary/human interaction services

Geographic boundary/human interaction services is a category in the geographic service taxonomy. The following are examples of human interaction services for working with geographic data and services.

— Catalogue viewer: client service that allows a user to interact with a catalogue to locate, browse, and manage metadata about geographic data or geographic services.

— Geographic viewer: client service that allows a user to view one or more feature collections or coverages. This viewer allows a user to interact with map data, e.g. displaying, overlaying and querying. An example is the viewer client generator defined in ISO 19128.

— Geographic viewer — animation: geographic viewer that allows a human to sequence views of the same geographic location at different times.

— Geographic viewer — mosaicing: geographic viewer that allows combination of views of geographic data for adjacent areas into a single view.

— Geographic viewer — perspective: geographic viewer that allows the viewpoint to be changed; for example, to specify how high off the ground, what direction, and from what angle a viewpoint is seeing a scene.

— Geographic viewer — imagery: geographic viewer that visualizes coverage data including the mapping of sample dimensions in the coverage to colours in the display.

— Geographic spreadsheet viewer: client service that allows a user to interact with multiple data objects and to request calculations similar to an arithmetic spreadsheet, but extended to geographic data.

— Service editor: client service that allows a user to control geographic processing services. Views include understanding a service, composing/scripting service chains, invoking a service, status of a service, scheduling services for peak performance times, and invoking a service chain.

— Chain definition editor: provides user interaction with a chain definition service.

— Workflow enactment manager: provides user interaction with a workflow enactment service.

— Geographic feature editor: geographic viewer that allows a user to interact with feature data, e.g. displaying, querying; supports feature annotation. The user controls view orientation, perspective, depth cueing, hidden-line/surface, light-sources, transparency, and texture mapping onto the objects. Objects in view can be picked or drawn on to generate new objects in the model.

— Geographic symbol editor: client service that allows a human to select and manage symbol libraries. ISO 19117 is relevant to symbol libraries.

— Feature generalization editor: client service that allows a user to modify the cartographic characteristics of a feature or feature collection by simplifying its visualization, while maintaining its salient elements — the spatial equivalent of simplification.

— Geographic data-structure viewer: client service that allows a user to access part of a dataset to see its internal structure, to request creation of new objects from parts of an object being browsed and to request a check of an object, e.g. type checking.

10.8.3 Geographic model/information management services

The following are examples of model/information management services for working with geographic data and services.

— Feature access service: service that provides a client access to and management of a feature store. An access service may include a query that filters the data returned to the client. ISO 19125, ISO 19142 and ISO 19143 are relevant to feature access.

— Map access service: service that provides a client access to a geographic graphics, i.e. pictures of geographic data. ISO 19128 is relevant to map access.

— Coverage access service: service that provides a client access to and management of a coverage store. Coverage is considered as a special case of Feature. An access service may include a query that filters the data returned to the client. ISO 19123 and ISO 19111 are relevant to coverage access.

— Coverage access service — sensor: service that provides access to coverage where the source of the coverage data are a real-time sensor, i.e. not a persistent store.

— Sensor description service: service that provides the description of a coverage sensor, including sensor location and orientation, as well as the sensor's geometric, dynamic and radiometric characteristics for geoprocessing purposes.

— Product access service: service that provides access to and management of a geographic product store. A product can be a predefined feature collection and metadata with known boundaries and content, corresponding to a paper map or report. A product can alternately be a previously defined set of coverages with associated metadata.

— Feature type service: service that provides a client access to and management of a store of feature type definitions. The static and dynamic information models for a feature type catalogue are provided in ISO 19110.

— Catalogue service: service that provides discovery and management services on a store of metadata about instances. The metadata may be for dataset instances, e.g. dataset catalogue, or may contain service metadata, e.g. service catalogue. ISO 19115-1 is relevant to catalogue service for dataset metadata. ISO 19115-2 is relevant for service metadata.

— Registry service: service that provides access to a store of metadata about types. Types are vocabularies that can be organized and related to each other. Example registries are information community registries, type dictionaries, service registries and schema registries.

— Gazetteer service: service that provides access to a directory of instances of a class or classes of real-world phenomena containing some information regarding position. An information model for a gazetteer is provided by ISO 19112.

— Order-handling service: service that provides a client with the ability to order products from a provider, including the formulation of quotes on orders, selection of geographic processing options, submission of an order, statusing of orders and billing and accounting of users' orders.

— Standing order service: order-handling service that allows a user to request that a product over a geographic area be disseminated when it becomes available. Such dissemination includes receiving, preparing (i.e. reformatting, compressing, decompressing, etc.), prioritizing, and transmitting the geographic information requested through standing queries or profiles.

10.8.4 Geographic workflow/task management services

The following are examples of workflow/task management services for working with geographic data and services. A geographic workflow enactment service should support tracking of lineage and provenance information during the workflow.

— Chain definition service: service to define a chain and to enable it to be executed by the workflow enactment service. This includes information about its starting and completion conditions, constituent activities and rules for navigating between them, user tasks to be undertaken, references to applications which may be invoked, definition of any workflow relevant data which may need to be referenced, etc. Chain definition service may also provide a chain validation service.

— Workflow enactment service: the workflow enactment service interprets a chain and controls the instantiation of services and sequencing of activities. This is done through one or more co-operating workflow management engines, which manage the execution of individual instances of the various services. A workflow enactment service maintains control data either centralized or distributed across a set of workflow engines. Workflow control data include the internal state information associated with the various services under execution and may also include check-pointing and recovery/restart information used by the workflow engines to coordinate and recover from failure conditions.

— Subscription service: service to allow clients to register for notification about events. Events are defined by a service that performs an activity resulting in the event. Events are catalogued by the subscription service. Clients identify events of interest, e.g. receipt of data with a specific geographic extent. When an event occurs, the subscription service sends notification to all clients who have registered an interest in the event. Once an event occurs, a subscription service may cause an activity to occur, e.g. delivery of a product.

10.8.5 Geographic processing services

10.8.5.1 Relation of geographic processing services to general feature model

The taxonomy within the processing services category is based on the General Feature Model as presented in ISO 19109. Processing services modify the properties of Features; therefore, processing services categories are based on the property types for features given by the General Feature Model. The processing services category is subdivided into the categories for geographic processing services shown in Table 22.

Table 22 — Geographic processing services taxonomy

— Geographic processing services — spatial
— Geographic processing services — thematic
— Geographic processing services — temporal
— Geographic processing services — metadata

10.8.5.2 Geographic processing services — spatial

The following is a non-exhaustive listing of geographic processing services — spatial.

— Coordinate conversion service: service to change coordinates from one coordinate system to another coordinate system that is related to the same datum. In a coordinate conversion, the parameters' values are exact. Coordinate conversion services include map projection services. ISO 19111 is relevant to coordinate conversion.

— Coordinate transformation service: service to change coordinates from a coordinate reference system based on one datum to a coordinate reference system based on a second datum. A coordinate transformation differs from a coordinate conversion in that the coordinate transformation parameter values are derived empirically; therefore, there may be several different estimations (or realizations). ISO 19111 is relevant to coordinate transformation.

— Coverage/vector conversion service: service to change the spatial representation from a coverage schema to a vector schema or vice versa. A standard relevant to vector schema definition is ISO 19107. A standard relevant to coverage schema definition is ISO 19123.

— Image coordinate conversion service: coordinate transformation or coordinate conversion service to change the coordinate reference system for an image. A standard relevant to image coordinates is ISO 19123; standardization relevant to image coordinates is also discussed in ISO/TR 19121.

— Rectification service: service that projects a tilted or oblique image onto a selected plane or other surface. The plane is often horizontal, but can be tilted to achieve some desired condition, such as to better fit the local surface of the earth.

— Orthorectification service: rectification service that removes image displacement due to variation in terrain elevation. Orthorectification requires use of digital elevation data, usually in grid form.

— Sensor geometry model adjustment service: service that adjusts sensor geometry models to improve the match of the image with other images and/or known ground positions.

— Image geometry model conversion service: service that converts sensor geometry models into a different but equivalent sensor geometry model.

— Subsetting service: service that extracts data from an input in a continuous spatial region either by geographic location or by grid coordinates.

— Sampling service: service that extracts data from an input using a consistent sampling scheme either by geographic location or by grid coordinates.

— Tiling change service: service that changes the tiling of geographic data.

— Dimension measurement service: service to compute dimensions of objects visible in an image or other geodata. An alternative name for this service is “image mensuration services”.

— Feature manipulation services: register one feature to another, an image, or another dataset or coordinate set; correcting for relative translation shifts, rotational differences, scale differences, and perspective differences; verifying that all features in the Feature Collection are topologically consistent according to the topology rules of the Feature Collection, and identifying and/or correcting any inconsistencies that are discovered.

— Feature matching service: service that determines which features and portions of features represent the same real-world entity from multiple data sources, e.g. edge matching and limited conflation.

— Feature generalization service — spatial: service that reduces spatial variation in a feature collection to increase the effectiveness of communication by counteracting the undesirable effects of scale reduction.

— Route determination service: service to determine the optimal path between two specified points based on the input parameters and properties contained in the Feature Collection; may also determine the measured distance between two points along a specified path based on the properties supported in the Feature Collection further, may determine the length of time it takes to follow a route through the geographic data in the Feature Collection.

— Positioning service: service provided by a position-providing device to use, obtain and unambiguously interpret position information and determines whether the results meet the requirements of the use. A standard relevant to position services is ISO 19116.

— Proximity analysis service: given a position or geographic feature, finds all objects with a given set of attributes that are located within a user-specified distance of the position or feature.

10.8.5.3 Geographic processing services — thematic

The following is a non-exhaustive listing of geographic processing services — thematic.

— Geoparameter calculation service: service to derive application-oriented quantitative results that are not available from the raw data themselves.

— Thematic classification service: service to classify regions of geographic data based on thematic attributes. Classification of coverages (including images) subdivides a coverage into regions based on attribute values. Classification of features sorts features into groups based on attribute values or feature associations.

— Feature generalization service — thematic: service that generalizes feature types in a feature collection to increase the effectiveness of communication by counteracting the undesirable effects of data reduction.

— Subsetting service: service that extracts features or coverage elements from a larger set based on thematic characteristics.

— Spatial counting service: service that counts geographic features of a given type within a specified area.

— Geographic information extraction services: services supporting the extraction of feature and terrain information from remotely sensed and scanned images.

— Image processing service: service to change the values of thematic attributes of an image using a mathematical function. Example functions include convolution, data compression, feature extraction, frequency filters, geometric operations, nonlinear filters and spatial filters.

— Reduced resolution generation service: service that reduces the resolution of an image.

— Image manipulation services: services for manipulating data values in images; changing colour and contrast values, applying various filters, manipulating image resolution, noise removal, “striping”, systematic-radiometric corrections, atmospheric attenuation, changes in scene illumination, etc.

— Image understanding services: services that provide automated image change detection, registered image differencing, significance-of-difference analysis and display and area-based and model-based differencing.

— Image synthesis services: services for creating or transforming images using computer-based spatial models, perspective transformations, and manipulations of image characteristics to improve visibility, sharpen resolution, and/or reduce the effects of cloud cover or haze.

— Multi-band image manipulation: services that modify an image using the multiple bands of the image. Examples include ratioing, principal components transformation, intensity-hue-saturation colour space transformation, de-correlation-stretching.

— Object detection service: service to detect real-world objects in an image.

— Geoparsing service: service to scan text documents for location-based references, such as a place names, addresses, postal codes, etc., in preparation for passage to a geocoding service.

— Geocoding service: service to augment location-based text references with geographic coordinates (or some other spatial reference).

10.8.5.4 Geographic processing services — temporal

The following is a non-exhaustive listing of geographic processing services — temporal.

— Change detection services: service to find differences between two datasets that represent the same geographical area at different times.

— Temporal reference system transformation service: service to change the values of temporal instances from one temporal reference system to another temporal reference system. ISO 19108 is relevant to temporal reference systems. Using the terminology of ISO 19108, a temporal reference system transformation service replaces the TM_Position value of a given TM_Instant with an equivalent TM_Position value associated with a different temporal reference system.

— Subsetting service: service that extracts data from an input in a continuous interval based on temporal position values.

— Sampling service: service that extracts data from an input using a consistent sampling scheme based on temporal position values.

— Temporal proximity analysis service: service that, given a temporal interval or event, find all objects with a given set of attributes that are located within a user-specified interval from the interval or event.

10.8.5.5 Geographic processing services — metadata

The following is a non-exhaustive listing of geographic processing services — metadata.

— Statistical calculation service: service to calculate the statistics of a dataset, e.g. mean, median, mode and standard deviation; histogram statistics and histogram calculation; minimum and maximum of an image; multi-band cross-correlation matrix; spectral statistics; spatial statistics; other statistical calculations.

— Geographic annotation services: services to add ancillary information to an image or a feature in a Feature Collection (e.g. by way of a label, a hot link, or an entry of a property for a feature into a database) that augments or provides a more complete description.

10.8.6 Geographic communication services

The following are examples of communications services for working with geographic data and services.

— Encoding service: service that provides implementation of an encoding rule and provides an interface to encoding and decoding functionality. A standard relevant to encoding is ISO 19118.

— Transfer service: service that provides implementation of one or more transfer protocols, which allows data transfer between distributed information systems over off-line or online communication media. To successfully transfer data between two systems, the sender and receiver need to agree on the transfer protocol to be used. A standard relevant to transfer is ISO 19118. For some geographic datasets, large data-object transfer is required.

— Geographic compression service: service that converts spatial portions of a feature collection to and from compressed form.

— Geographic format conversion service: service that converts from one geographic data format to another.

— Messaging service: service that allows multiple users to simultaneously view, comment about, and request edits of feature collections. This service allows collaboration involving geographic data.

— Remote file and executable management: service that provides access to secondary storage of geographic features as if it were local to the client.

10.8.7 Geographic system management and security services

The following is an example of system management and security services for working with geographic data and services.

— GeoRM – Geospatial Right Management (ISO 19153).

10.9 ISO suite of International Standards in geographic architecture services taxonomy

Table 23 provides a mapping for some of the standards in the ISO geographic information suite of standards to the architectural reference model categories, as classified for the geographic architecture service taxonomy in 10.8.

Table 23 — Mapping examples from the ISO geographic information suite of standards to the geographic architecture services taxonomy

Architectural reference model category	Relevant ISO geographic information standards
Geographic boundary/human interaction services	ISO 19117 *Geographic information — Portrayal*
	ISO 19128 *Geographic information — Web Map server interface*
Geographic model/information management services	ISO 19107 *Geographic information — Spatial schema*
	ISO 19110 *Geographic information — Methodology for feature cataloguing*
	ISO 19111 *Geographic information — Spatial referencing by coordinates*
	ISO 19112 *Geographic information — Spatial referencing by geographic identifiers*
	ISO 19115-1 *Geographic information — Metadata*
	ISO 19123 *Geographic information — Schema for coverage geometry and functions*
	ISO 19125-1 *Geographic information — Simple feature access — Part 1: Common architecture*
	ISO 19128 *Geographic information — Web Map server interface*
Geographic workflow/task management services	(not currently specialized in the ISO geographic information suite of standards)
Geographic processing service	ISO 19107 *Geographic information — Spatial schema*
	ISO 19108 *Geographic Information — Temporal schema*
	ISO 19109 *Geographic information — Rules for application schema*
	ISO 19111 *Geographic information — Spatial referencing by coordinates*
	ISO 19116 *Geographic information — Positioning services*
	ISO 19123 *Geographic information — Schema for coverage geometry and functions*
	ISO 19118 *Geographic information — Encoding*
Geographic communication services	(not currently specialized in the ISO geographic information suite of standards)
Geographic system management services	ISO 19153 *Geospatial Digital Rights Management Reference Model (GeoDRM RM)*

10.10 Geographic service chaining validity

A service chain combines services to produce results that the individual services could not produce alone. The early parts of this clause have defined type of services that could be chained. The syntactic issues of service chaining, e.g. data structure of a chain, architecture patterns for chaining, is defined in 8.4, which also defines how to construct chains; it does not address whether the results of a chain are semantically valid. The human user that constructed a new chain or invoked an existing chain of services should determine semantic validity of the results of a service chain. Here, it is assumed that

the chain is syntactically correct, i.e. the input and output types internal to a chain match and the chain produces a result. Some factors to consider in the semantic evaluation of a chain result are listed below.

— Appropriateness of starting data: Are the based datasets suited to the subsequent processing? For example, are the accuracy and resolution of the data and thematic values relevant?

— Effect of services on data: How do the individual services affect the data, e.g. error sources and propagation?

— Sequence of the services: How does the order of the chain affect the results? For example, should a spatial operation, e.g. orthorectification, be performed before or after a thematic operation, e.g. resampling the attribute values?

The evaluations depend upon understanding the services, e.g. through review of the service metadata, but also rely upon the users' understanding of the combinations of the services.

10.11 User-perspective Lifecycle model for Services

/req/servicetaxonomy/ service type - lifecycle	If a service is classified from a lifecycle perspective, the service shall be categorised to belong to one or more (aggregated service) of the following service lifecycle types for geographic information services: discovery', 'view', 'download', 'transformation', or 'invoke'.

The lifecycle-based perspective for the description of services has originated from the development in INSPIRE related to networked services for spatial data infrastructures (SDIs). This suggested the need to describe services from a more usage oriented life cycle perspective, from publication of services in a Register service, then discovery of services with a Discovery service, to View services and Download/access services, and potential for various invoked services. Security and right management is in this International Standard suggested to be orthogonal to the lifecycle. This service taxonomy is adopted as a service type lifecycle taxonomy.

The core components of the SDI reference model are shown in Figure 15.

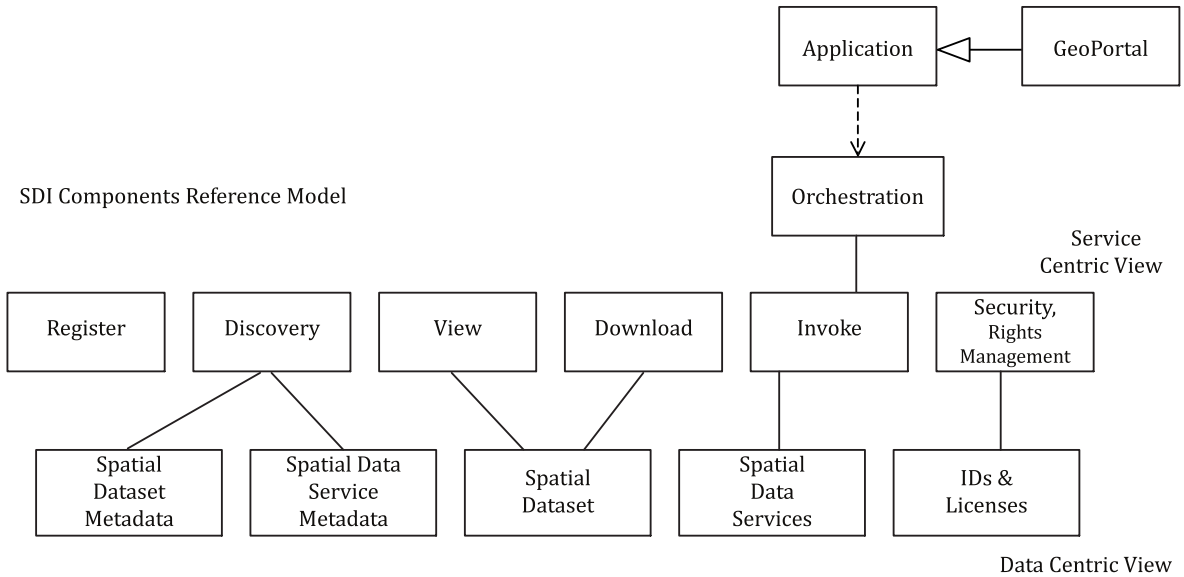


Figure 15 — Core components of the SDI reference model

The primary organizing structure is determined by the following generic core lifecycle components:

— **register (publish)**: for describing and publishing resources;

— **discovery**: for searching and discovery of resources;

— **view**: for visualizing of resources;

— **download**: for downloading and exchanging resources with the following two possible subtypes:

  — **download**-bulk-transfer (i.e. FTP);

  — **download-API-access** (i.e. WFS);

— **invoke**: for interacting with resources;

— **orchestration and composition**: for providing aggregated resources including, in particular, workflows for service composition;

— **security and rights management**: for managing access rights to resources.

10.12 User-defined service taxonomies

/rec/informationviewpoint/ service type - user-defined	Services may be categorised according to new user-defined service taxonomies. Typically agreed as a service taxonomy within a domain community.

The approach to support classification of services according to more than one service taxonomy and also to allow for one service to potentially belong to more than one service type within a service taxonomy, allows also for the creation of new user-defined service taxonomies.

10.13 Services organizer folder (SOF)

10.13.1 Grouping of services

Services organizer folders (SOFs) were introduced in 8.4.4 as groupings of services that are used for a specific task. The remainder of 10.13 provides examples of SOFs based on the geographic architecture services taxonomy defined in 10.8. Data models/application schemas/ontologies plays an important role in thematic processing and model/information management for processing services.

10.13.2 Image exploitation SOF

Image exploitation services are required to support most aspects of image exploitation, including precision measurement of ground positions and of object dimensions. For example, a variety of services are needed for extracting features from images or digital elevations from stereoscopic images. Image exploitation services are widely implemented and used in photogrammetric systems, currently using custom interfaces.

Table 24 provides an example Image exploitation SOF.

Table 24 — Image Exploitation SOF

Geographic architecture services taxonomy	Image exploitation services
Geographic boundary/human interaction services	Geographic viewer — mosaicing
Geographic model/information management services	Coverage access service Feature access service Catalogue service
Geographic workflow/task services	Chain definition service Workflow enactment service
Geographic processing services	

Table 24 *(continued)*

Geographic architecture services taxonomy	Image exploitation services
Geographic processing services — spatial	Coordinate conversion service Coordinate transformation service Image coordinate conversion service Orthorectification service
Geographic processing services — thematic	Geoparameter calculation service Thematic classification service Geospatial matching service Image processing service
Geographic processing services — temporal	Subsetting service Subsampling service
Geographic processing services — metadata	Statistical calculation service
Geographic communication services	Geographic format conversion service
Geographic system management services	Registry service

10.13.3 Geographic data fusion SOF

Geographic data fusion is a framework of services for the synthesis of data originating from different sources. It aims at obtaining information of greater quality; the exact definition of greater quality will depend upon the application. Geographic fusion is accomplished through organizing, relating and linking disparate sources of location-based information. Geographic fusion relates to the fusion of two or more geographic elements and the fusion of geographic elements with other sources of structured and unstructured data from distributed sources.

Table 25 provides an example of Geographic data fusion SOF.

Table 25 — Geographic data fusion SOF

Geographic architecture services taxonomy	Image exploitation services
Geographic boundary/human interaction services	Geographic viewer
Geographic model/information management services	Map access service Feature access service Catalogue service Gazetteer service Feature type service
Geographic workflow/task services	—
Geographic processing services	
Geographic processing services — spatial	Feature matching service
Geographic processing services — thematic	Object detection service Change detection service Geoparsing service Geocoding service

Table 25 *(continued)*

Geographic architecture services taxonomy	Image exploitation services
Geographic processing services — temporal	—
Geographic processing services — metadata	—
Geographic communication services	Encoding service Messaging service
Geographic system management services	—

10.14 Semantic information models

From the services perspective, the information viewpoint is focusing on the information used by services. The information viewpoint is concerned with the semantics of information and information processing. An information specification for a system is a model of the information that it holds and of the information processing that it carries out. In the context of the computational view, there is a particular focus on the information being used and provided by the particular services.

This also provides the foundation for the information viewpoint for services. From the computational view, the focus within the information viewpoint is on the identification of the information that is used and produced by services. A key aspect is the model-driven approach. Specific spatial domains define thematic information areas that are modelled using information models. The results are application schemas that provide conceptual models. The conceptual models are published in data specifications including feature catalogues and these are implemented in data warehouses. Combined with standardized and SOA conforming encoding, the result is clearly structured and standardized accessible spatial data.

Recent developments in the field of disclosing spatial information on the web has evolved around linked data. Linked data are a concept of a Web of Data where data can be found on the web and contain links to other data distributed over the Web. As data are published conforming to the Web architecture, integration of data and multiple usage become more prominent and less design dependent. This makes it slightly different to the traditional SOA approach. Traditionally, the information structuring in SOA is based on identification of domains in which information and semantics are defined. Linked data bridges the gaps between the thematic domains by providing a mechanism to work without predefined information boundaries. In this regard, linked data fits into the web 2.0 philosophy of participatory information sharing by interlinking of data. The input/output parameters of services can be references for linked data. Linked data has also been applied for the description of services in efforts such as linked USDL (see http://www.linked-usdl.org/).

Linked data when related to the SOA should be seen as a complementary way to enlarge the possibilities for spatial data integration and spatial to non-spatial data integration and broaden its use outside the geospatial community.

Linked data are published in the Resource Description Format (RDF). This is an alternative way of representing information in class diagrams based on expressions known as triples in the form of subject-predicate-object expressions. GML is closely related to RDF and can be transformed if considerations on HTTP URI's and stable links are taken care of. Figure 16 (from Reference [7]) shows the integration of linked data in the SDI concept. The structured geospatial data serve as data source for linked data interfaces.

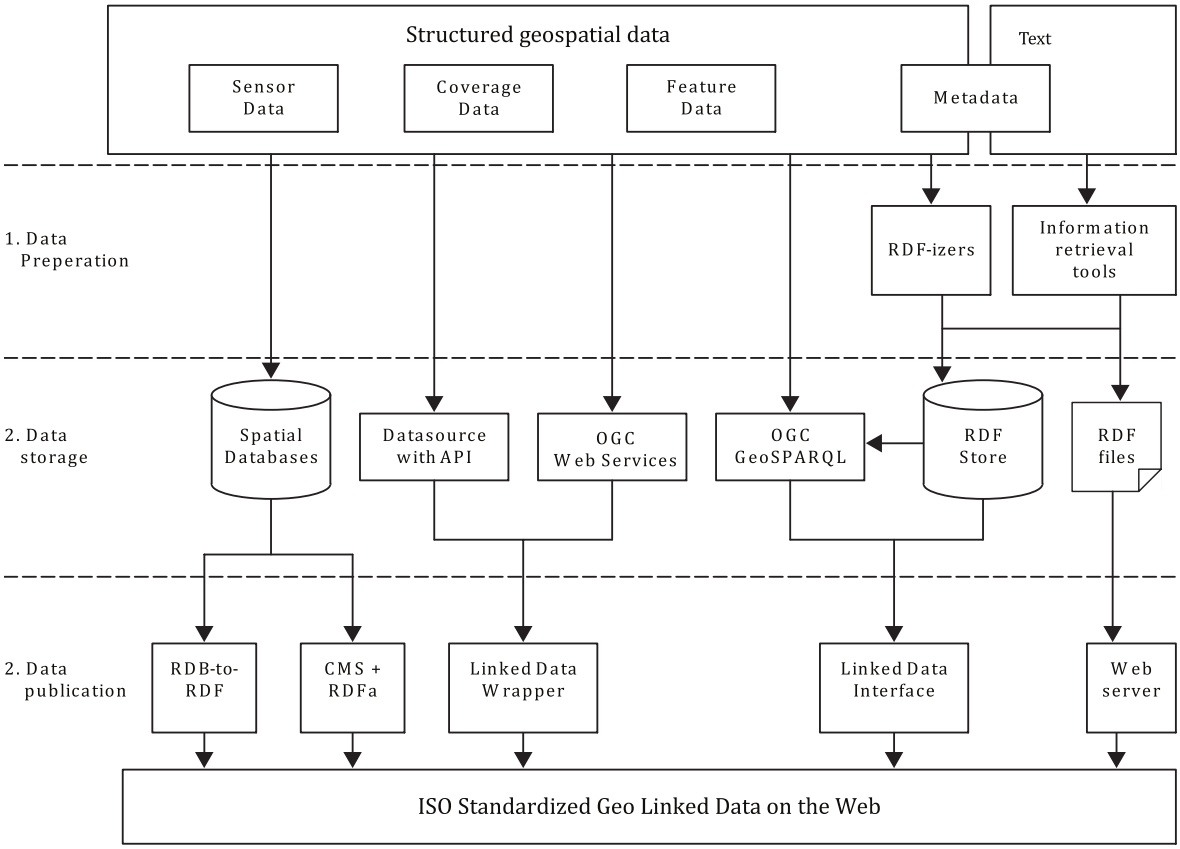


Figure 16 — Technologies and standards involved in Linked Spatial Data

The essential pillars of Linked Data[3] are traditional web technologies and use of lightweight techniques for data model representation. The former depends on the use of Uniform Resource Identifiers (URIs) as reference points. A URI may be used to uniquely identify both data and non-data resources.[1] Resolvers map a URI to the physical location of the resource or, in the case of non-data resources, to a description.

Linked Data are usually implemented as common HTML for human interfaces, plus RDF for links with machine-processable semantics. RDF provides a structure for any form of description and is the basis of the Semantic Web.[2] RDF describes resources in the form of triples (subject-predicate-object).[40] A basic typing mechanism for subjects, predicates and objects is available as RDF-Schema (RDF-S). RDF-S allows for extensions in order to specify domain-dependent subtypes and thus allow for a domain vocabulary in its own namespace. RDF comes with different encodings, one of which is RDF/XML. The key elements of RDF/XML for linking are “rdf:about” (identifiers or anchors) and “rdf:resource” (pointers or links). Resources become a set in which elements are connected with links. By these means, users can navigate between data like browsing through web pages. Generally, each piece of data contains link(s) to other data. However, leaf nodes or end points of the graph may make use of any other format, which may not support linking. Content-negotiation in HTTP allows client applications (like browsers) to negotiate various data representations.[9]

Although RDF is recommended for implementing the Linked Data as a single global model for all data sources, other structured formats such as GML can support semantic linking and it is expected that also other technologies will evolve around this.

10.15 Examples of relevant standards

— UML, XML, GML as general languages (graphical and text-based) for information modelling.

— ISO 19103 for modelling of information and application schemas using UML.

— ISO/IEC 19793 provides a UML profile for all of the main concepts defined in the RM-ODP information viewpoint. It provides a foundation for full RM-ODP information viewpoint modelling but within the geographic information community, it has so far been a preference to use an approach with UML class diagrams, in combination with the use of XML, GML and potentially semantic technologies like OWL and Linked Open Data with RDF.

10.16 Examples and tools

The INSPIRE program provides an example of establishing a spatial information base in this case to support environmental policy on national and pan-national level. Several technical reports, implementation rules and guidelines for data specifications resulted from this program. Regarding the information viewpoint and the actual production of data specifications, a set of documents is available. The presentation of the documents in the order below in itself is already an example of how the information viewpoint is addressed.

— *Definition of Annex Themes and Scope*

  Identification and description of 34 spatial data domains regarding their definition and scope. Examples of domains are hydrography, transport networks, administrative units, and geology.

— *Methodology for the development of data specifications*

  The process and proposed methodology of developing application schema and feature catalogues is explained. Use cases lead to identification of information requirements. These are transferred to initial spatial object types and subsequent application schemas. An iterative process to test, validate and restructure is described to arrive at specifications that fulfil described requirements.

— *INSPIRE Generic Conceptual Model*

  Basic rules and principles are laid down to which all application schema should comply. This document bridges the gap between the conceptual standards on geo-information modelling and its application in specific domain models. For instance, reusable patterns are introduced on modelling unique identifiers, temporal models and a meta model is presented by defining dedicated stereotypes.

— *Guidelines for the encoding of spatial data*

  Guidelines are presented to guide this implementation in GML application schema in a harmonized way and provide additional specifications on top of general GML standards.

— *INSPIRE data specifications*

  There are 34 documents each dealing with a separate theme. Each includes the domain definition, use case description, application schema, feature catalogue, and portrayal.

11 Engineering viewpoint: A basis for distribution and communication patterns

11.1 Distribution transparencies and the engineering viewpoint

The engineering viewpoint focuses on mechanisms for distribution, distribution transparencies, and support services such as security and persistence. Distribution transparencies enable complexities associated with system distribution to be hidden from applications where they are irrelevant to their purpose. The engineering viewpoint also introduces various possible architectural styles and communication patterns.

Location transparency masks the need for an application to have information about location in order to invoke a service. Location transparency is handled by name-servers that transparently map logical names to physical server addresses. The underlying mechanisms used may differ between physical communication solutions.

Replication transparency masks the fact that multiple copies of a service may be provided in order to provide reliability and availability. Replication transparency is handled to the extent that different calling semantics can be transparently implemented in the framework, dependent on the policies desired. The supported semantics are a shallow copy (copy of one object without its associated objects), a partial copy (copy of one object with its direct associated objects) and a deep copy (copy of one object and transitively, a copy of all its associated objects until no more objects are reachable).

There are other distribution transparencies that are also important to relate to, such as the following:

— failure transparency: masks failures and recoveries of objects;

— federation transparency: masks interworking across multiple administrations;

— group transparency: masks the use of a group of objects to provide an interface;

— migration transparency: masks the relocation of an object;

— resource transparency: masks passivity and reactivation;

— persistence transparency: hides the actual activation and deactivation of objects from a persistent store, and the actual storage mechanisms and representation format used;

— transaction transparency: hides coordination for achieving the transactional properties;

— security transparency: hides the mechanisms that are being used for authentication and authorization.

In order to achieve interoperability between different platforms, it is necessary to have mappings between the platforms' support for these transparencies. This can be done through a higher abstraction layer which maps to the implementation and representation of these services in various platforms.

11.2 Distributing components using a multi-tier architecture model

To support flexible deployment, IT architectures are structured as multi-tiered distributed architectures. As a reference model, logical four-tier architecture is presented with discussion on variations in different physical architectures. The logical architecture is the arrangement of services and associated interfaces that are present in the system (see Figure 17). The physical architecture is the arrangement of components and associated interfaces that implement the services. The components are hosted on hardware computing resources or nodes.

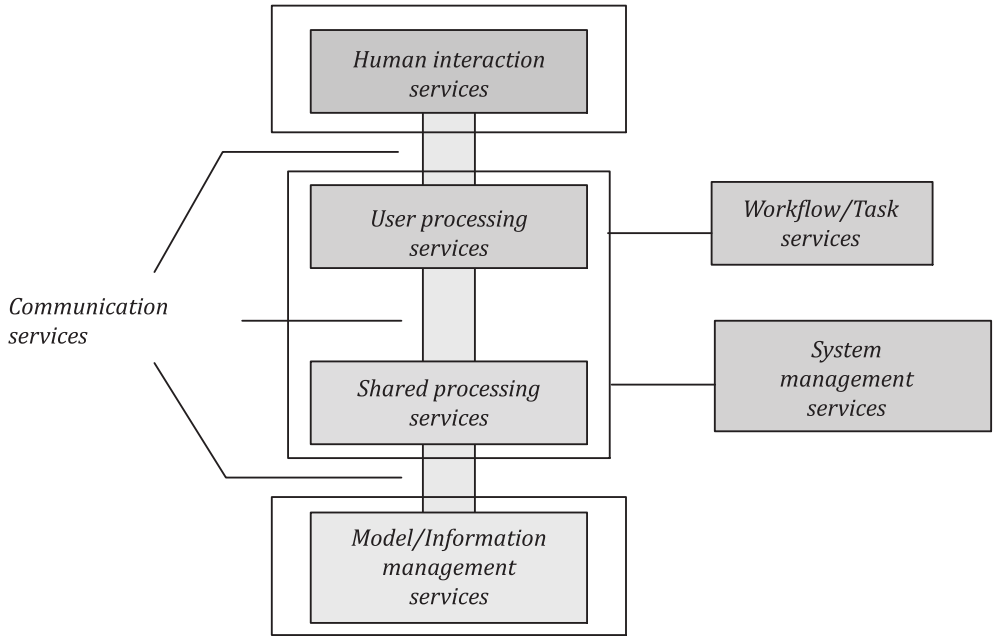


Figure 17 — Logical multi-tiered architecture

The architectural reference model, as defined in 8.4, structures the types of services of an IT system. Each tier can contain both IT-general services and GIS-extended services for that tier.

— The human interaction services tier is responsible for physical interaction with the user, through display and input media and an appropriate dialogue. This might be separated into a presentation tier and a dialogue tier.

— The user processing services tier is a part of the processing services responsible for the functionality required by the user.

— The shared processing services tier is part of the processing services responsible for common services (both domain specific and general) that can be used by multiple users.

— The model/information management services tier is responsible for physical data storage and data management.

— The workflow/task services are a set of services that can be viewed as a specialized processing service.

— The communication services are responsible for connecting the various tiers together. The communication services are present as the connections between the other service tiers.

— The system management services are orthogonal to the multi-tiered architecture and might be introduced in multiple tiers.

The logical architecture can be mapped to multiple physical architectures. All tiers could be mapped into one monolithic application or could be mapped using different physical client-server architectures. Figure 18 to Figure 20 show mappings to various physical architectures.

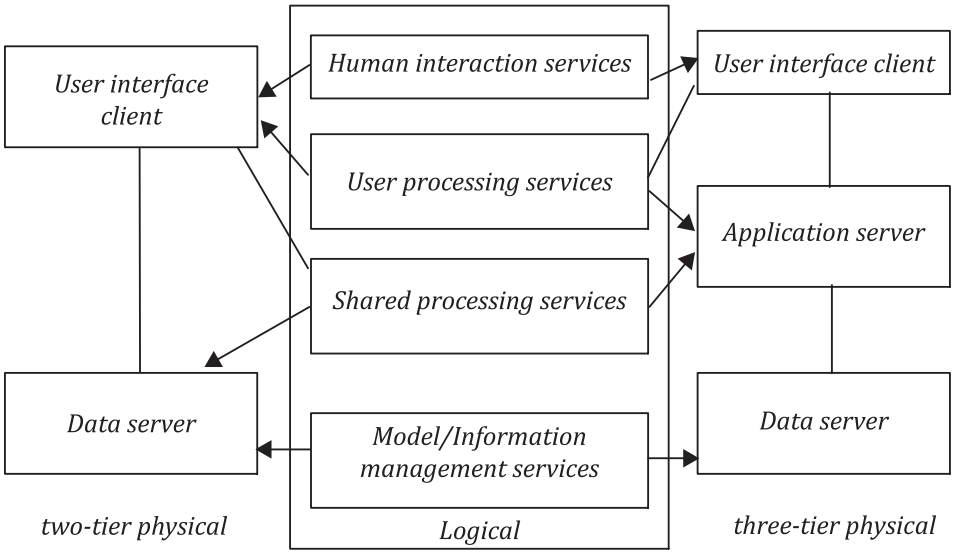


Figure 18 — From logical four-tier to physical two-tier or three-tier architecture

In Figure 18, a data server contains the logic that interfaces either with a data storage system or with some other type of external data source, such as a data feed or an external application system. The data server provides model/information management services.

An application server contains components that are responsible for processing services. An application server may provide both user processing services and shared processing services.

A user interface client provides interaction services, contains the logic that presents information to an external source and obtains input from that source. In most cases, the external source is a human end-user working at their own computer, although the external source might also be process-oriented. The client logic generally provides menus of options to allow the user to navigate through the different parts of the application and it manipulates the input and output fields on the display device. Frequently, the presentation component also performs a limited amount of input data validation.

As shown in Figure 18, a two-tier physical architecture typically consists of a user interface client interacting directly with a data server. User services are normally executed in the user interface client, while the data server provides shared processing services.

As shown in Figure 19, a three-tier physical architecture introduces an intermediate application server that is responsible for the execution of shared processing services, sometimes also for user services. A major advantage of using a three-tier distributed information system is that the user can choose how to combine components to perform tasks. In an interoperable component environment, the user can select components that perform similar tasks and combine the chosen components to best produce the information that is needed for application.

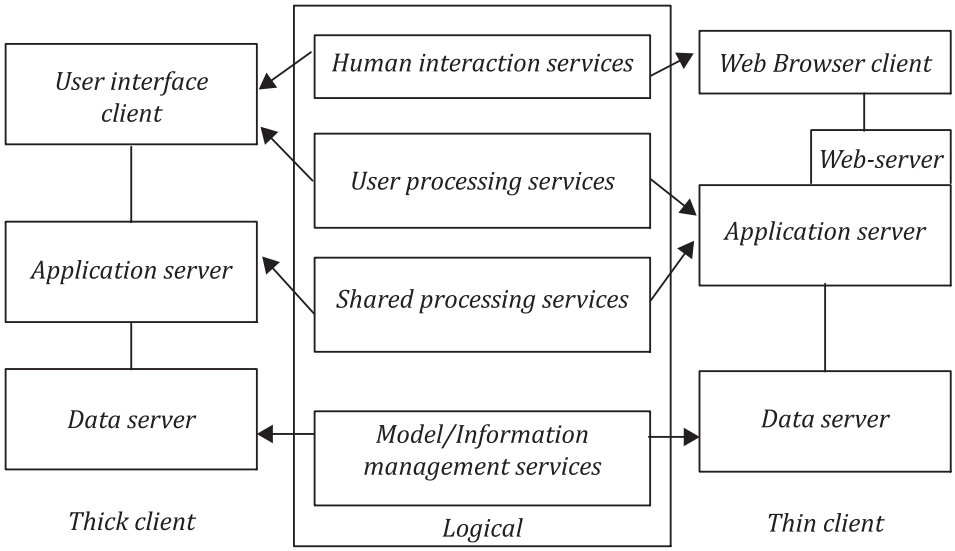


Figure 19 — Mapping logical four-tier to thick and thin clients

As shown in Figure 19, a thick user interface client architecture will typically contain a larger part of the functionality in the user service. A thin user interface client (typically, a web browser) will mostly contain user dialogue and presentation code. A web browser client is a user interface client that interacts with a web server, using the Internet HTTP protocol with content represented in HTML and/or XML.

A platform-independent abstract specification can include specification of both user interface (UI) plus service and data/information aspects. This means that a large specification can be broken into different parts, each addressing a different part of a total system specification (see Figure 20). The various parts can typically be mapped into different kind of specific technologies.

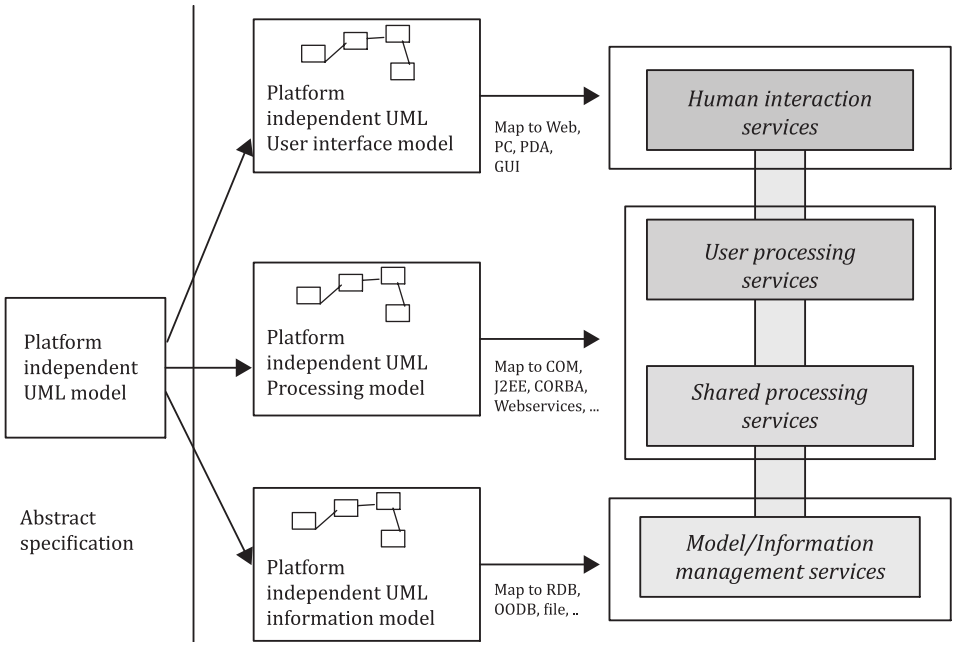


Figure 20 — Mapping from platform independent UML models

11.3 Distribution transparencies

The engineering viewpoint is concerned with the design of distribution-oriented aspects, i.e. the infrastructure required to support distribution. An engineering specification of an ODP system defines

a networked computing infrastructure that supports the system structure defined in the computational specification and provides the distribution transparencies that it defines. ODP defines the following distribution transparencies:

— access;

— failure;

— location;

— migration;

— relocation;

— replication;

— persistence;

— transaction.

Security may also be a mechanism. There is a particular focus on the different mechanisms to support a multi style service-oriented architecture (SOA) in a unified logical model, including both RESTful and Synchronous services, as well as events.

11.4 Engineering viewpoint Service specifications

The requirements for creating the service specification part for the engineering viewpoint are formalized as a requirements class summarized in Table 26.

Table 26 — Requirements class for Engineering viewpoint service specifications

Requirements class	**/req/engineeringviewpoint**
Target type	Mapping models
Dependency	/req/computationalviewpoint/
Dependency	/req/informationviewpoint/
Requirement	**/req/engineeringviewpoint/architectural style mapping**

/req/engineeringviewpoint/ architectural style mapping	The architectural style(s) supported for the service shall be described and reflected in refined interface and operation specifications that show how the specified operations from the computational viewpoint will be realized in the selected architectural style(s) (i.e. RPC, OWS, REST, SOAP, etc.). This includes a mapping for the requirements of the computational viewpoint, with interfaces, operations, behaviour, pre-conditions and post-conditions, and service chaining.

The form of the platform specific interfaces and operations might vary depending on the chosen architectural style and platform, i.e. RPC-oriented, message-oriented, RESTful, and document-oriented. It is possible to describe different alternative architectural styles and technology representations for the same service.

Services with interfaces and operations should be described in an architectural style dependent way showing how concrete interfaces are derived from the interfaces specified in the computational viewpoint. Describe the mappings from interfaces in the computational viewpoint to interfaces in the engineering viewpoint, according to chosen architectural style(s). This can either be done through the description of transformation rules (i.e. using QVT or similar transformation language) or through tables or other forms of mapping descriptions.

Operations in the concrete interfaces should be described according to the selected architectural style(s) showing their input and output parameters (and exceptions). Each input and output parameter shall be further described through the technology viewpoint. Describe the mappings from operations in the computational viewpoint to interfaces in the engineering viewpoint, according to chosen architectural style(s). This can either be done through the description of transformation rules (i.e. using QVT or similar transformation language) or through tables or other forms of mapping descriptions.

The behaviour of services may be shown through the use of UML sequence diagrams, showing possible sequenced use of operations, if the concrete sequences need to be further refined from the computational viewpoint.

The pre-conditions and post-conditions and invariants of an operation may be shown through expressions in OCL, if the concrete pre-conditions and post-conditions need to be further refined from the computational viewpoint.

In 11.5 and 11.6, examples of some various possible architectural styles are presented.

11.5 Multi-style SOA

A multi-style service-oriented architecture is one in which the service-oriented architectural style coexists with other architectural styles, whereas service-oriented architectural style is an architectural style that restricts the roles, characteristics and allowed relationships of services and service consumers.[43]

Most initiatives for the specification and standardization of geo-information resources (e.g. services, models, and formats) have adopted various Web technologies as their protocols and encodings.

In recent years, the World-Wide Web has undergone important changes. The advent of new technologies [e.g. AJAX and JSON (see http://www.json.org/)], new services (e.g. Web 2.0 services) and new architectural approaches (e.g. REST) has implied a more common usage of multiple architectural styles. Some of the following constraints are still adhered to within the different styles of service oriented architectures.

— Common payload (the actual data that is being exchanged) and protocol: each service provides an interface that is invoked through a payload format and protocol that are understood by all the potential clients of a service.

— Published and discoverable interfaces: each service has a published and discoverable interface that allows systems to search for services that are best suited for their purposes.

— Loose coupling: services are connected to other services and clients using standard, dependency-reducing, decoupled message-based methods such as XML document exchanges.

— Multiple communication interfaces: services can implement separately defined communication interfaces.

— Composability: Because services are coarse-grained reusable components that expose their functionality through a well-defined interface, systems can be built as a composition of services and evolve through the addition of new services.

In 11.6, a number of relevant architectural styles are briefly described.

11.6 Relevant architectural styles

11.6.1 Service-oriented architectures

Service-Oriented Architecture (SOA) can be considered both from an organizational and business perspective, as well as from a technical perspective. SOA is a means of organizing solutions that promotes reuse, growth and interoperability. It is not itself a solution to domain problems but rather an organizing and delivery paradigm that enables more value to be gained by using capabilities which

are locally "owned" and those under the control of others. SOA reflects the reality that ownership boundaries are a motivating consideration in the architecture and systems design.

The central focus of SOA is the task or business function. The central concept of SOA is the service: a mechanism to enable access to a set of one or more capabilities. A service enables users to perform arbitrarily complex tasks involving the resources which are handled by the service provider and not directly exposed to the user. SOA defines a class of architectures which enable loosely-coupled access to generic capabilities provided by service providers. The generality of services in terms of information, structure, semantics, behaviour, action and process models require the provision of functionalities supporting visibility and awareness through service description and policy definition. This makes SOA powerful but complex, especially if only simple tasks are required.

In the World-Wide Web, the emergence of Simple Object Access Protocol (SOAP) provided a common specification for service invocation between Web components. SOAP, originally designed for conveying remote methods invocation in XML, being fully suitable for generic messaging between objects, evolved in a more general standard for sending services calls targeted to end points exposed in the Web and addressed through a specific URL. Further specifications such as WSDL (see http://www.w3.org/TR/wsdl), UDDI (see http://www.oasis-open.org/committees/uddi-spec/), WS-I (see http://www.ws-i.org/), WS-*, etc. from various standardization bodies, mainly W3C (see http://www.w3.org/) and OASIS (see http://www.oasis-open.org/) make the SOAP suite, a complete set of standards for building SOA over the Web providing service description, cataloguing, security and so on. Indeed, this is currently the most widespread solution for e-Business and e-Government systems.

SOAP Version 1.2 is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. A SOAP Message is made up of a header and a body. The optional SOAP Header element contains application specific information (like authentication, payment, etc.) about the SOAP message. The SOAP body provides a mechanism for transmitting information to an ultimate SOAP receiver, which is the service provider. An important characteristic is that a SOAP message could be transmitted using any protocol as long as it allows the transfer of the serialized Infoset to the destination. Several transport mechanisms (bindings) are defined for SOAP using application-level protocols such as HTTP and SMTP. This generality is obtained at the expense of the loss of protocol-specific characteristics. For example, the HTTP binding utilizes HTTP as a transport-level protocol: the semantics of the request line and of most of the HTTP headers are actually lost.

11.6.2 Representational State Transfer (REST)

The W3C document "Web Services Architecture" (see http://www.w3.org/TR/ws-arch/) makes SOAP the fundamental basis for a "new" Web, a Web of exposed services instead of shared documental resources. But the great success of SOAP in many application fields like e-Business and e-Government did not guarantee the same success in other fields and applications characterized by different requirements. SOAP fits well to SOA, where different organizations expose complex services (e.g. banking transactions, travel reservations, commercial orders) implemented in background facilities which can be composed in workflows for carrying out high-level business processes. These great capabilities have the drawback of a complex infrastructure for services discovery, description, etc. However, common Web applications, in particular the so-called Web 2.0 services, are light services dedicated to publish and access structured and semi-structured information (e.g. Websites, Web interfaces to databases and repositories, Content and Document Management Systems, blogs, etc.), a context where SOA seems to be overloading. W3C have proposed a new vision of the Web architecture to make it conform to its original concept. Such architecture is based on an architectural style named Representational State Transfer (REST).

REST is a resource-oriented style for distributed systems defined to describe the original Web architecture and to guide its future evolution preserving its fundamental characteristics, scalability. Although REST is defined bearing in mind the Web, all the architectures satisfying the REST constraints are REST-based architectures, not only the Web itself. The term "RESTful" was introduced to describe

system architectures based on the REST style. RESTful architectures present the following two essential characteristics deriving from the Uniform Interface constraint:

a) all the significant resources are addressed and accessible through the same set of methods (common interface);

b) logical connections between resources are made explicit as hyperlinks.

NOTE REST is not a technology. In particular, REST is neither simply XML+HTTP nor any HTTP API which seems to be a common misunderstanding.

11.6.3 Web 2.0

It is still difficult to agree on what the Web 2.0 really is. One definition is *"a set of economic, social, and technology trends that collectively form the basis for the next generation of the Internet — a more mature, distinctive medium characterized by user participation, openness, and network effect"*. Comparing some of the most known applications, some commonalities that can be considered the principles behind the Web 2.0 are the following:

a) Web As Platform;

b) Harnessing Collective Intelligence;

c) Data are the Next Intel Inside;

d) End of the Software Release Cycle;

e) Lightweight Programming Models;

f) Software Above the Level of a Single Device;

g) Rich User Experiences.

In order to facilitate the development of the so-called Web 2.0 applications, specific strategies, patterns, and technologies have been developed or improved. They are often indicated with well-known (and often misunderstood) terms.

— *Mash-up (or mashup)*: A mashup is a technique for building applications that combine data from multiple sources to create an integrated experience. In the Web 2.0, this is often done directly in the browser (consumer mashup) using open Web APIs (e.g. Google Maps API, Wikipedia API, OpenLayers, etc.). The mashup approach helps to achieve the "Web As Platform" and "Lightweight Programming Models" principles.

— *Lightweight technologies*: technologies that do not require a heavy upfront investment or operational requirements. These are simpler and less cumbersome to work with. The downside is that lightweight technologies can be less feature rich than their more "heavyweight" alternatives. Examples of technologies considered lightweight are Javascript Object Notation (JSON) for semi-structures data representation, and Javascript/ECMAScript as programming language. In the Web 2.0, they help to reach the rapid development required by the "End of the Software Release Cycle" and the "Lightweight Programming Models" principles.

— *AJAX*: The term AJAX (or Ajax) was coined to a new approach to web applications development characterized by the following:

  — standards-based presentation using XHTML and CSS;

  — dynamic display and interaction using the Document Object Model;

  — data interchange and manipulation using XML and XSLT;

  — asynchronous data retrieval using XMLHttpRequest;

  — JavaScript binding everything together.

The term was actually a shorthand for Asynchronous Javascript and XML. However, Ajax now encompasses the use of specific technologies like Javascript and XML. Other technologies can be adopted for scripting (e.g. Adobe Flash instead of Javascript) or data representation (e.g. JSON instead of XML). Even the XMLHttpRequest object can be replaced by other techniques for asynchronous retrieval of Web resources (e.g. by use of IFrames). What actually characterizes Ajax is its application model: an Ajax application eliminates the start-stop-start-stop nature of interaction on the Web by introducing an intermediary, an Ajax engine, between the user and the server. The Ajax engine allows the user's interaction with the application to happen asynchronously, independent of communication with the server.

12 Technology viewpoint: A basis for cross platform interoperability

12.1 Infrastructure interoperability and the technology viewpoint

The technology viewpoint of ISO RM-ODP is concerned with the underlying infrastructure in a distributed system. It describes the hardware and software components used in a distributed system. To achieve interoperability in the technology viewpoint, an infrastructure that allows the components of a distributed system to interoperate is needed. This infrastructure, which may be provided by a distributed computing platform (DCP), allows objects to interoperate across computer networks, hardware platforms, operating systems and programming languages. This viewpoint also represents the platform specific specification with mappings to any technology specific aspects of the relevant distributed computing platform that has not been covered by the architectural style description from the engineering viewpoint.

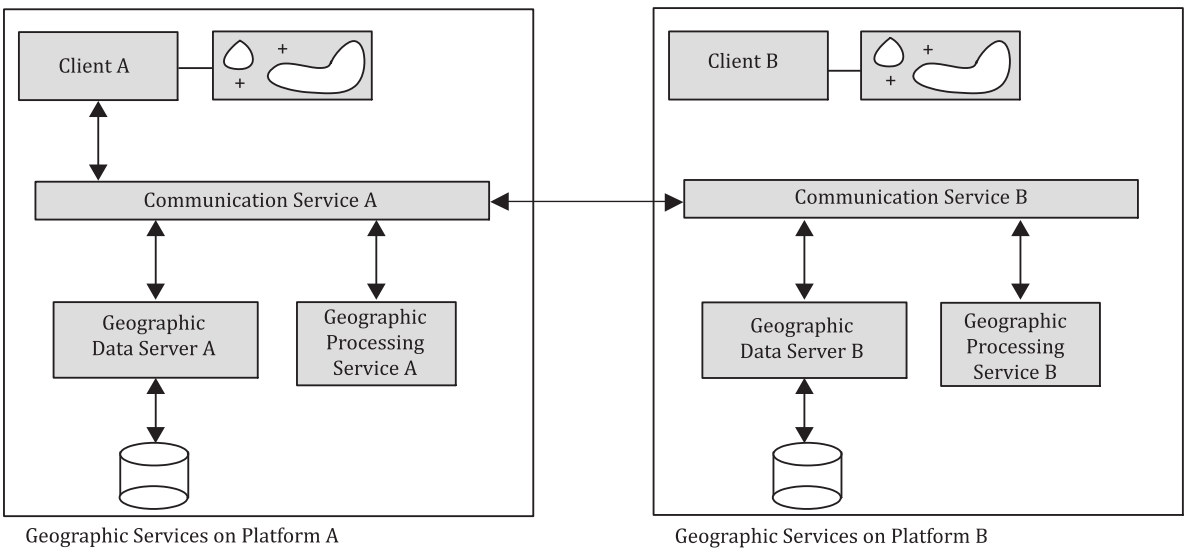


Figure 21 — Technology viewpoint model of the interoperability reference model

The communication service allows the components in a distributed system to interoperate. In the two systems depicted in Figure 21, communication service A allows the components of system A (client A, geodata server A and GIS service component A) to interoperate, while communication service B allows the components of system B to interoperate. To allow system A to interoperate with system B, the components of system A shall be able to request services from the components of system B and vice versa. The following two cases are to be considered.

— **The two systems use the same DCP**

  As an example, OMG has defined the general inter-ORB protocol (GIOP) and internet inter-ORB protocol (IIOP) standards that allow two systems that use CORBA to interoperate. These standards define how objects in system A can invoke services from objects in system B. Java RMI also have support for allowing objects in one system to invoke services from objects in another system.

— **The two systems use two different DCPs**

If the two systems use different DCPs, they can interoperate through the use of special "bridging" tools. These bridging tools allow objects that use one DCP to interoperate with objects that use another DCP.

12.2 Need for multiple platform-specific specifications

It is assumed that one platform-neutral service specification from the computational and information viewpoint will be the basis for multiple platform-specific service specifications through different architectural styles mappings in the computational viewpoint and for different distributed computing platform mappings through the technology viewpoint. ISO 19125 is an example of a set of platform-specific service specifications for SQL, Object Linking and Embedding\Component Object Model (COM/OLE), and CORBA, and has demonstrated the need for at least three different implementation specifications of the same platform-neutral specification. Later evolutions has shown the need for other platform specific specifications, like WFS:OWS, WFS:SOAP and WFS:REST.

Multiple platform-specific specifications are necessary because of the variety of DCP's and the differences in the way in which they support the functional requirements. One platform-neutral service specification is needed to support interoperability of multiple platform-specific specifications. It is foreseen that a future set of standards could be made to describe the rules for mappings from specifications in this International Standard to specific platforms, similar to how the ISO 19118 now describes a generic framework for encoding and ISO 19136 describes how this can be realized with an encoding in XML and GML. These rules can then be applied when new services are being specified according to this International Standard as part of the process of defining platform specific specifications for services.

12.3 Conformance between platform-neutral and platform-specific service specifications

Platform-neutral models shall be described in UML in accordance with the requirements for the computational and information viewpoints and also following the requirements in ISO 19103.

Platform-specific models in the computational and technology viewpoint may be described in UML, together with a description of their mapping to the corresponding platform-neutral models. It is also allowed to describe the platform-specific models directly in a platform-specific language such as SQL, CORBA/IDL, web services description language, etc., as long as the mapping to the corresponding platform independent model is well defined.

Development of service specifications may proceed from platform-specific to platform-neutral or from platform-neutral to platform-specific. In either case, a service specification shall not be considered complete according to the full service profile until it has a platform-neutral model and at least one platform-specific model (see Figure 22).

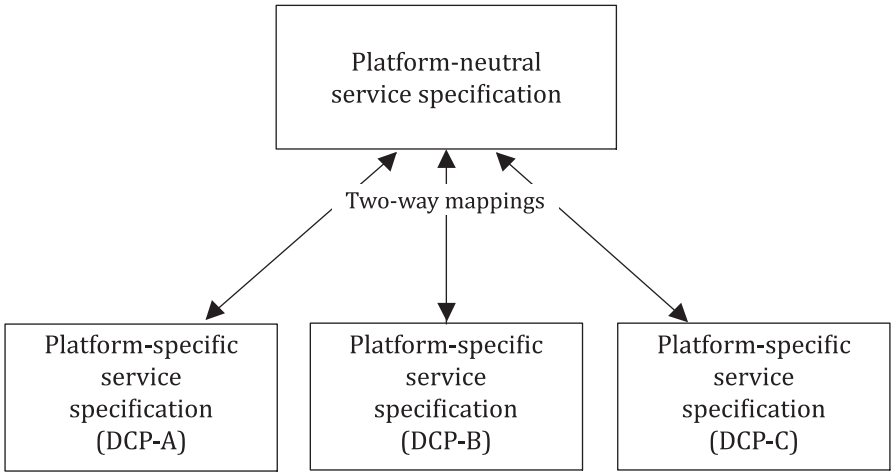


Figure 22 — From platform-neutral abstract specifications to multiple platform-specific specifications

A platform-neutral service specification should be defined meeting the requirements for the enterprise, computation and information viewpoints. A platform-specific service specification shall show how it meets the intentions of the platform-neutral specification, by showing how the various parts of the platform-neutral service specification are mapped onto the platform-specific service implementation, based on the architectural style mapping in the engineering viewpoint and the technology mappings in the technology viewpoint. In order to support interoperability between different implementation specifications, the reverse mapping back to the concepts in the platform-neutral model may be defined. As relevant, a platform-specific specification should include the encoding of information according to the framework of ISO 19118, i.e. by using technology specific encodings, i.e. such as ISO 19136 with GML/XML.

12.4 From platform-neutral to platform-specific specifications

Any service specification that addresses a particular platform, i.e. Web services (SOAP, REST, OWS) or other, shall include a detailed mapping specification from the basic data types used in the platform-neutral UML models to corresponding types in the platform.

Implementation specifications for a particular platform may be described through the use of the UML, if platform-specific aspects have been introduced through the use of platform-specific stereotypes, tagged values and constraints. UML profiles with platform-specific types exist for various platforms.

Annex D describes some example principles for mappings of the platform-neutral services of the computational and information viewpoint to a variety of DCPs.

12.5 Technology objects

The technology viewpoint describes the implementation of the ODP system in terms of a configuration of technology objects representing the hardware and software components of the implementation. It is constrained by cost and availability of technology objects (hardware and software products) that would satisfy this specification. These may conform to platform-specific standards that are effectively templates for technology objects.

12.6 Technology viewpoint service specifications

12.6.1 Requirements class for technology viewpoint

The requirements for creating the service specification part for the technology viewpoint are formalized as a requirements class summarized in Table 27.

Table 27 — Requirements class for Technology viewpoint service specifications

Requirements class	/req/technologyviewpoint
Target type	UML service model
Dependency	ISO 19103 (Conceptual schema language)
Dependency	/req/computationalviewpoint/
Dependency	/req/informationviewpoint/
Requirement	/req/technologyviewpoint/technology mappings

12.6.2 Technology mappings

/req/technologyviewpoint/ technology mappings	The technology mappings shall be described based on mappings from the information, computational and engineering viewpoints. This includes a mapping for the elements of the information viewpoint with the operation input/output/exception parameters to their related technology representation and encoding. Any needed further mapping from the computational viewpoint and the architectural style from the engineering viewpoint should be further mapped for interfaces, operations, behaviour, pre-conditions and post-conditions and service chaining.

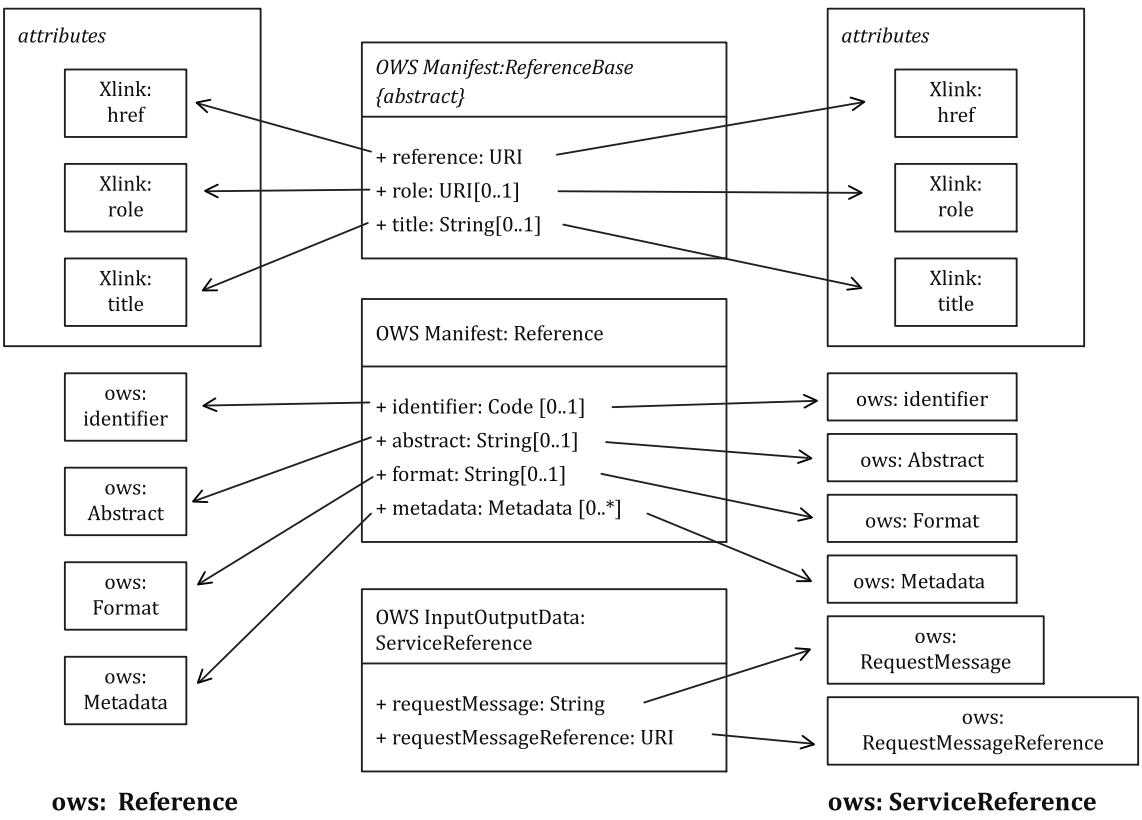


Figure 23 — Example of mappings from UML information models to XML representation

A Key Value Pair (KVP) encoding of the GetCapabilities operation request may be as shown in Table 28, with example values appropriate for WCS 1.0.0.

Table 28 — GetCapabilities operation request HTTP/URL parameters for KVP

Name and example[a]	Optionality and use	Definition and format
Service = WCS	Mandatory	Abbreviated service type identifier text.[b]
Request = GetCapabilities	Mandatory	Operation name text.
AcceptVersions = 1.0.0,0.8.3	Optional When omitted, return latest supported version.	Prioritized sequence of one or more specification versions accepted by client, with preferred versions listed first.
Sections = Contents	Optional When omitted or not supported by server, return complete service metadata document.	Comma-separated unordered list of zero or more names of sections of service metadata document to be returned in service metadata document.
UpdateSequence = XXX (where XXX is character string previously provided by server)	Optional When omitted or not supported by server, return latest service metadata document version.	Service metadata document version, value is "increased" whenever any change is made in complete service metadata document.
AcceptFormats = text/xml	Optional When omitted or not supported by server, return service metadata document using MIME type "text/xml".	Prioritized sequence of zero or more response formats desired by client, with preferred formats listed first.
AcceptLanguages = en-CA,fr_CA	Optional When not supported by server, return human readable text in a language of the server's choice.	List of languages desired by the client for all human readable text in the response, in order of preference. For every element, the first matching language available from the server shall be present in the response.

[a] All parameter names are listed here using mostly lower case letters. However, any parameter name capitalization shall be allowed in KVP encoding.

[b] A specific OWS specification shall define the abbreviated service type identifier to be used by all implementing services.

An example of a GetCapabilities request message encoded in XML is as follows.

```
<?xml version="1.0" encoding="UTF-8"?>
<GetCapabilities xmlns="http://www.opengis.net/ows/2.0"
xmlns:ows="http://www.opengis.net/ows/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/ows/2.0
fragmentGetCapabilitiesRequest.xsd" service="WCS"
updateSequence="XYZ123" acceptLanguages="en-CA">
<!- Maximum example for WCS. ->
<AcceptVersions>
<Version>1.0.0</Version>
<Version>0.8.3</Version>
</AcceptVersions>
<Sections>
<Section>Contents</Section>
</Sections>
<AcceptFormats>
<OutputFormat>text/xml</OutputFormat>
</AcceptFormats>
<AcceptLanguages>
<Language>en-CA</Language>
<Language>fr-CA</Language>
</AcceptLanguages>
</GetCapabilities>
```

This example includes all of the possible XML attributes and elements, but only the service attribute is required, within the required GetCapabilities root element.

GetCapabilities request SOAP encoding

Specific OWS servers may implement a SOAP transfer of the GetCapabilities operation request using the XML encoding specified above.

12.7 Architectural classification according to cloud computing service categories

Cloud computing has a significant potential in the context of the management, preservation and sharing of data. It is a computing model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. Thus, cloud computing focuses on the delivery of computing as a service, where shared resources, software and information are provided to computers and other devices as a utility, usually over the Internet.

The different service types can also be categorized according to their relevance for emerging cloud services, starting with a classification for the application level and software as a service (SaaS), but also further down to platform as a service (PaaS) and infrastructure as a service (IaaS). The SaaS, PaaS and IaaS providers all have a range of both generic and specific services that can be individually classified with the service taxonomies already proposed.

Annex A
(normative)

Conformance

A.1 Levels of specification

The test suite is structured into six conformance classes (see Clause 2). Each test relates to one or more specific requirements, which are explicitly indicated in the description of the test.

The framework, concepts and methodology for conformance and testing for ISO geographic information suite of standards are defined in ISO 19105.

Any specification, including a profile or functional standard that claims conformance with this International Standard, shall pass the corresponding requirements described in the abstract test module in A.2 to A.7.

This distinguishes between the following three different levels of specifications:

— platform-neutral service specifications: service that specifies a specific geographic service in a platform-neutral way, meeting the requirements of the enterprise, computational and information viewpoints;

— platform-specific service specifications: environment or language-specific service specification that has been derived from a platform-neutral service specification and is claimed to be a profile standard of this, meeting the requirements of the engineering and technology viewpoints;

— platform-specific service implementations: actual implementation of a service according to the specifications from the technology viewpoint that can be conformance tested against the abstract test suite of a platform-specific service specification.

A.2 Enterprise viewpoint

A.2.1 Servicename

The test for use of a Servicename is as follows:

a) test identifier: /conf/enterpriseviewpoint/servicename;

b) test purpose: verify that the servicename has been specified according to the rules in this International Standard;

c) test method: inspect the service description to verify that the requirement /req/enterpriseviewpoint/servicename has been obeyed for the servicename;

d) reference: /req/ enterpriseviewpoint/servicename;

e) test type: basic.

A.2.2 Servicetypes

The test for use of service types is as follows:

a) test identifier: /conf/enterpriseviewpoint/servicetypes;

b) test purpose: verify that this service has been categorised with servicetypes according to the rules in this International Standard;

c) test method: inspect the service description to verify that the requirement /req/enterpriseviewpoint/servicetypes has been obeyed for the classification according to service types;

d) reference: /req/enterpriseviewpoint/servicetypes;

e) test type: basic.

A.2.3 Service purpose

The test for use of a Service purpose is as follows:

a) test identifier: /conf/enterpriseviewpoint/servicepurpose;

b) test purpose: verify that the service purpose has been specified according to the rules in this International Standard;

c) test method: inspect the service purpose to verify that the requirement /req/enterpriseviewpoint/servicepurpose has been obeyed for the service purpose;

d) reference: /req/ enterpriseviewpoint/servicepurpose;

e) test type: basic.

A.2.4 Service scope

The test for use of a Service scope is as follows:

a) test identifier: /conf/enterpriseviewpoint/servicescope;

b) test purpose: verify that the service scope has been specified according to the rules in this International Standard;

c) test method: inspect the service scope to verify that the requirement /req/enterpriseviewpoint/servicescope has been obeyed for the service scope;

d) reference: /req/enterpriseviewpoint/servicescope;

e) test type: basic.

A.2.5 Service capabilities

The test for use of Service capabilities is as follows:

a) test identifier: /conf/enterpriseviewpoint/servicecapabilities;

b) test purpose: verify that the service capabilities have been specified according to the rules in this International Standard;

c) test method: inspect the service capabilities to verify that the requirement /req/enterpriseviewpoint/servicecapability has been obeyed for the service capabilities;

d) reference: /req/enterpriseviewpoint/servicecapabilities;

e) test type: basic.

A.2.6 Service community

The test for use of Service community is as follows:

a) test identifier: /conf/enterpriseviewpoint/servicecommunity;

b) test purpose: verify that the service community has been specified according to the rules in this International Standard;

c) test method: inspect the service community to verify that the requirement /req/enterpriseviewpoint/servicecommunity has been obeyed for the service community;

d) reference: /req/enterpriseviewpoint/servicecommunity;

e) test type: basic.

A.2.7 Service scenarios

The test for use of Service scenarios is as follows:

a) test identifier: /conf/enterpriseviewpoint/servicescenarios;

b) test purpose: verify that the service scenarios have been specified according to the rules in this International Standard;

c) test method: inspect the service scenarios to verify that the requirement /req/enterpriseviewpoint/servicescenarios has been obeyed for the service scenarios;

d) reference: /req/enterpriseviewpoint/servicescenarios;

e) test type: basic.

A.3 Computational viewpoint

A.3.1 Service interfaces

The test for description of Service interfaces is as follows:

a) test identifier: /conf/computationalviewpoint/interfaces;

b) test purpose: verify that all interfaces in the service model have been specified according to the rules in this International Standard;

c) test method: inspect the service model to verify that requirement /req/computationalviewpoint/interfaces has been obeyed for all specified interfaces;

d) reference: /req/computationalviewpoint/interfaces;

e) test type: basic.

A.3.2 Service operations

The test for description of service operations in interfaces is as follows:

a) test identifier: /conf/ computationalviewpoint/operations;

b) test purpose: verify that all operations in all interfaces for the service model have been specified according to the rules in this International Standard;

c) test method: inspect the service model to verify that requirement /req/computationalviewpoint/operations has been obeyed for all specified interfaces;

d) reference: /req/computationalviewpoint/operations;

e) test type: basic.

A.3.3 Service behaviour

The test for description of service behaviour is as follows:

a) test identifier: /conf/ computationalviewpoint/behaviour;

b) test purpose: verify that all behaviour for the service model has been specified according to the rules in this International Standard;

c) test method: inspect the service model to verify that requirement /req/computationalviewpoint/behaviour has been obeyed for behaviour specifications;

d) reference: /req/computationalviewpoint/behaviour;

e) test type: basic

A.3.4 Operation pre_and_post_conditions

The test for description of operation pre-conditions and post-conditions is as follows:

a) test identifier: /conf/ computationalviewpoint/pre-conditions and post-conditions;

b) test purpose: verify that specified pre-conditions and post-conditions have been described according to the rules in this International Standard;

c) test method: inspect pre-conditions and post-conditions to verify that requirement /req/computationalviewpoint/pre-conditions and post-conditions has been obeyed for stated specifications of pre-conditions and post-conditions;

d) reference: /req/computationalviewpoint/pre-conditions and post-conditions;

e) test type: basic

A.3.5 Service chaining

The test for description of a Service chaining is as follows:

a) test identifier: /conf/ computationalviewpoint/service chaining;

b) test purpose: verify that service chains have been described according to the rules in this International Standard;

c) test method: inspect service chains to verify that requirement /req/computationalviewpoint/service chains has been obeyed for stated specifications of service chains;

d) reference: /req/computationalviewpoint/servicechaining;

e) test type: basic

A.4 Information viewpoint

A.4.1 Service dependencies

The test for description of service dependencies is as follows:

a) test identifier: /conf/ informationviewpoint/servicemodel dependencies;

b) test purpose: verify that specified service dependencies have been described according to the rules in this International Standard;

c) test method: inspect the service dependencies to verify that requirement /req/informationviewpoint/service dependencies has been obeyed for stated specifications of pre-conditions and post-conditions;

d) reference: /req/computationalviewpoint/servicemodel dependencies;

e) test type: basic.

A.4.2 Operation input/output/exception parameters

The test for description for operation input/output/exception parameters is as follows:

a) test identifier: /conf/informationviewpoint/input/output/exception parameters;

b) test purpose: verify that specified input/output/exception parameters have been described according to the rules in this International Standard;

c) test method: inspect input/output/exception parameters to verify that requirement /req/informationviewpoint/input/output/exception parameters has been obeyed for stated specifications of input/output/exception parameters;

d) reference: /req/informationviewpoint/input/output/exception parameters;

e) test type: basic.

A.5 Service taxonomies

A.5.1 Geographic Service type - architecture

The test for classification of a service according to the geographic service type based on architecture area classification is as follows:

a) test identifier: /conf/servicetaxonomy/ service type – architecture;

b) test purpose: verify that the selected service type for the service is according to the rules in this International Standard for geographic service type based on architecture;

c) test method: check that the requirement /req/servicetaxonomy/ service type – architecture has been obeyed for the selected service type; if the service has a title identical with an example service in 10.8, determine whether the service provides the functionality that is defined in the example;

d) reference: /req/servicetaxonomy/ service type - architecture;

e) test type: basic.

A.5.2 Geographic Service type – life cycle

The test for classification of a service according to service type based on the geographic service life cycle area classification is as follows:

a) test identifier: /conf/servicetaxonomy/ service type – life cycle;

b) test purpose: verify that the selected service type for the service is according to the rules in this International Standard for geographic service type life cycle;

c) test method: check that the requirement /req/servicetaxonomy/ service type – life cycle has been obeyed for the selected service type;

d) reference: /req/servicetaxonomy/ service type – life cycle;

e) test type: basic.

A.6 Engineering viewpoint

A.6.1 Architectural style mapping

The test for description for the architectural style mapping(s) is as follows:

a) test identifier: /conf/engineeringviewpoint/architectural style mapping;

b) test purpose: verify that specified architectural style mapping(s) have been described according to the rules in this International Standard;

c) test method: inspect the architectural style mapping(s) to verify that requirement //req/engineeringviewpoint/architectural style mapping has been obeyed for the specified architectural style mapping(s);

d) reference: /req/engineeringviewpoint/architectural style mapping;

e) test type: capability.

A.7 Technology viewpoint

A.7.1 Technology mappings

The test for description for technology mappings is as follows:

a) test identifier: /conf/technologyviewpoint/technology mappings;

b) test purpose: verify that specified technology mappings have been described according to the rules in this International Standard;

c) test method: inspect the technology mappings for conformance between the platform neutral and platform specific service specifications to verify that requirement /req/technologyviewpoint/ technology mappings has been obeyed for stated specifications of the technology mappings;

d) reference: /req/technologyviewpoint/ technology mappings;

e) test type: capability.

Annex B
(informative)

Example user scenarios

B.1 Example 1: Service chaining for remote sensed data

B.1.1 Summary

Working in a distributed environment, a user of geographic data first locates several datasets and services, and then chains the services to produce the resulting geographic products of interest.

A user wants to determine the land cover classification of a geographic area, e.g. wetlands, urban, etc., using remote sensed coverage data. The user wants a graphical image of the analysis that can be used in a presentation or document. The user wants to include an existing base map in the image.

This scenario is an example of the transparent type of service chaining. The functionality could also be achieved using translucent or opaque service chaining. In the case of opaque chaining, the user would interact with an aggregate service.

B.1.2 Preconditions

The user accesses various geographic information services using "thin" client software located on the user's hardware. The user has access to a network over which the services are accessed. Financial transactions and user authorization issues are ignored in this scenario. The user (or the user's client software) knows the network location of a catalogue service for geographic data and services. The user is familiar with coverage data and classification services.

B.1.3 Detailed steps

The detailed steps are as follows.

a) The user accesses a catalogue service to search for coverage data with a geographic extent of interest to the user. The user identifies several coverages with data over a specific geographic area of interest. Also using the catalogue service, the user discovers a digital elevation model covering the same geographic extent.

b) The user accesses a catalogue service to search for geographic services of interest to the user. The user identifies several orthorectification services and several classification services. The user reviews the service metadata for the services and determines the best orthorectification and classification services for use on the coverages that were selected in the prior step. The user also identifies a portrayal service using the catalogue service.

c) The user examines the coverages by providing a reference to the coverages to the portrayal service. Based on a visual examination, the user estimates whether the coverages would be appropriate for this use.

d) The user accesses the orthorectification service. The access operation includes references to the coverage and DEM. The orthorectification service provides the user with a reference to an orthorectified coverage.

e) The user accesses the classification service. The access operation includes a reference to the orthorectified coverage. The user interacts with the semi-automated classification service, identifying training areas of known land cover type. Statistical measures and visual techniques are provided to the user to determine if the land cover classification has been precise. The classification

service result is a feature collection consisting of polygons identifying the classified regions over the geographic extent of interest.

NOTE The order of services could be switched, first performing the classification and then the orthorectification. The user needs to make this type of decision based on knowledge of the services.

f) The user combines the orthorectified coverage and the feature collection into a viewable image using the portrayal service.

g) The user accesses the catalogue service to locate a feature collection of transportation data over the same geographic extent.

h) The user combines the orthorectified coverage, the classification feature collection, and the transportation feature collection into a viewable image using the portrayal service. The user chooses an alternative symbology set for the transportation features and requests the portrayal service to recreate the composite image.

B.1.4 Post-conditions

Image file that can be viewed, printed or integrated into other desktop applications is on the user's computer.

B.1.5 Service chain directed acyclic graph

The scenario presented in B.1 is using the "transparent" design pattern described in 8.4.6.2. With the transparent design pattern, the human user controls the flow of the service chaining and the service chain is not explicitly materialized. When moving to one of the other design patterns, translucent or opaque, the service chain is an explicit DAG as described in 8.4.3.1. Figure B.1 shows the DAG for this scenario.

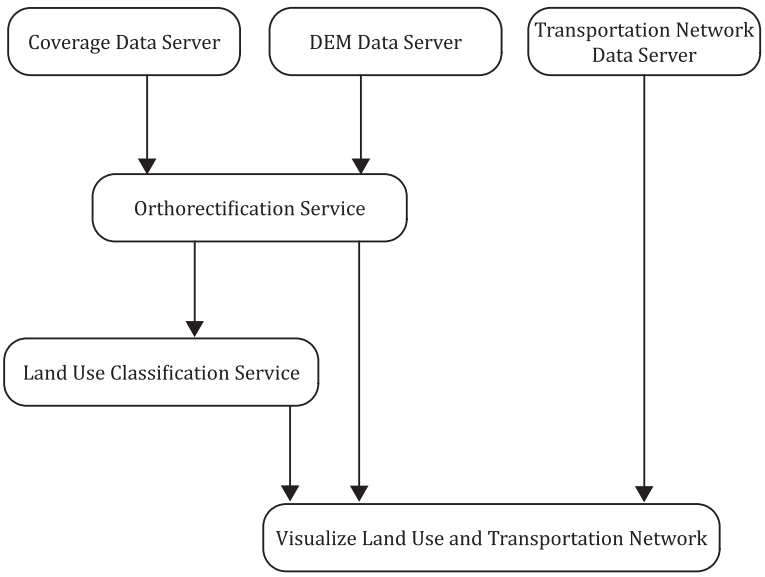


Figure B.1 — DAG for service chaining scenario example

B.2 Example 2: Roadside services

B.2.1 Summary

The roadside services scenario (see Figure B.2) illustrates the various future network-resident services that might come into play when a motorist tells his/her "personal information appliance" that he/she wants to find a service, e.g. a pizza restaurant, near the highway on which he/she is travelling.

B.2.2 Precondition

The motorist has a personal information appliance that has access to network services while travelling on the highway.

B.2.3 Detailed steps

The detailed steps are as follows.

a) Voice recognition software interprets a spoken request as a query for the nearest restaurant.

b) The broker submits the request along with coordinates and direction from the on-board positioning service to a roadside services database that reports the three nearest restaurant locations to a mapping service.

c) The mapping service queries a road map database and a set of preferences describing how the user prefers to have the data presented: map, voice directions, etc.

d) The mapping service presents route information and preferences to the presentation service, which packages information for delivery to the user.

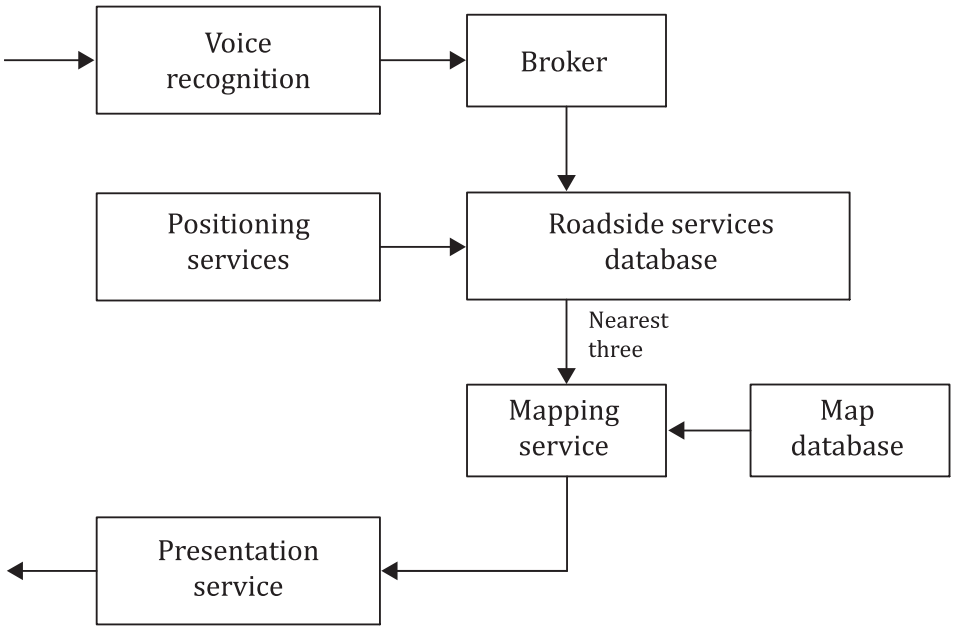


Figure B.2 — Roadside service scenario

Annex C
(informative)

Principles for mapping to distributed computing platforms

C.1 From platform-neutral to platform-specific specifications

Annex C illustrates the principles of mapping from a platform-neutral specification to a platform-specific specification through partial examples for several distributed computing platforms (DCPs). The intention is to demonstrate the principle rather than to fully specify DCP mappings.

An implementation for a particular platform can be described by using a UML model in the platform-specific specification. Platform-specific aspects are introduced through the use of platform-specific stereotypes, tagged values, data types and constraints. UML profiles with platform-specific types exist for various platforms, such as the OMG UML profile for CORBA and the Java community UML profile for Java/EJB.

The following environments and DCPs are considered as potential targets for platform-specific profiles of platform-neutral specifications:

— SQL;

— CORBA using ISO IDL;

— Java 2 Enterprise Edition with EJB (J2EE);

— COM+;

— EXPRESS/SDAI;

— ODMG;

— C++ and other more traditional commercial object oriented programming languages;

— Internet/http/Web Services, with XML and other encodings;

— Web services – with WSDL specifications, WS:WSDL;

— OGC style web services, WS:OGC;

— RESTful web services, WS:REST.

The following various encodings can be used:

— XML encoding;

— JSON encoding;

— ATOM encoding.

The goal of this Annex is to show how a platform-specific service specification is conformant with a platform-neutral service specification and the associated geographic information standards that are used by the service.

The existing UML profiles for various DCPs represent the concepts available in a target environment. It is possible to do the mapping from the platform-neutral UML model to a platform-specific UML model (see Figure C.1). There is then a straightforward mapping from the platform-specific UML model to a

platform-specific specification directly in the relevant language of the target environment, such as COM or CORBA IDL, SQL, SOAP, REST or similar.

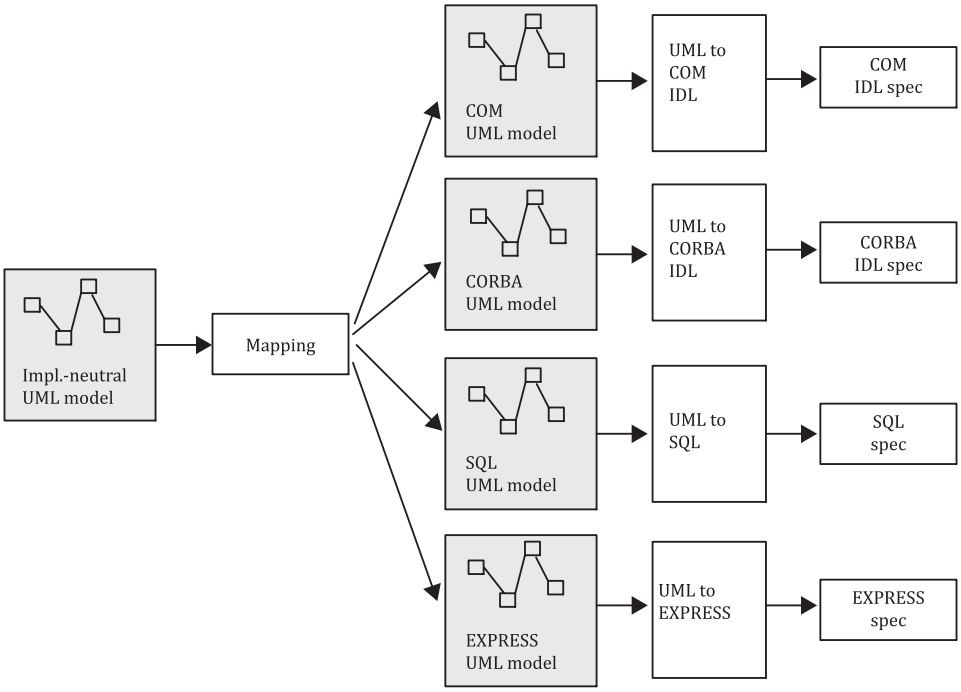


Figure C.1 — From platform-neutral to specific models through UML mappings

The final platform-specific specification is expressed in the appropriate language for the target environment and going through a platform-specific UML model is the only logical path that can be used.

The mappings may be done by hand or (semi-) automatically, based on a set of defined rules. The goal is to create a platform-specific specification that is conformant with the platform-neutral model, thus meeting its intended functionality and information requirements. It is assumed that the first platform-specific specifications will be done manually, and that providing a description of how the mapping has been done and how the elements of the platform-neutral model are reflected in the platform-specific specification will show conformance.

A platform-neutral specification is conceptual and might not have described the exact basis for representation of basic types. For example, a platform-neutral specification might not specify if a Real is to be represented as a float or a double, and with what kind of precision. This needs to be decided for a platform-specific specification. When a decision is made, it is necessary to link this decision back to the platform-neutral specification, so that future platform-specific specifications can make representation decisions that do not violate the potential for interoperability.

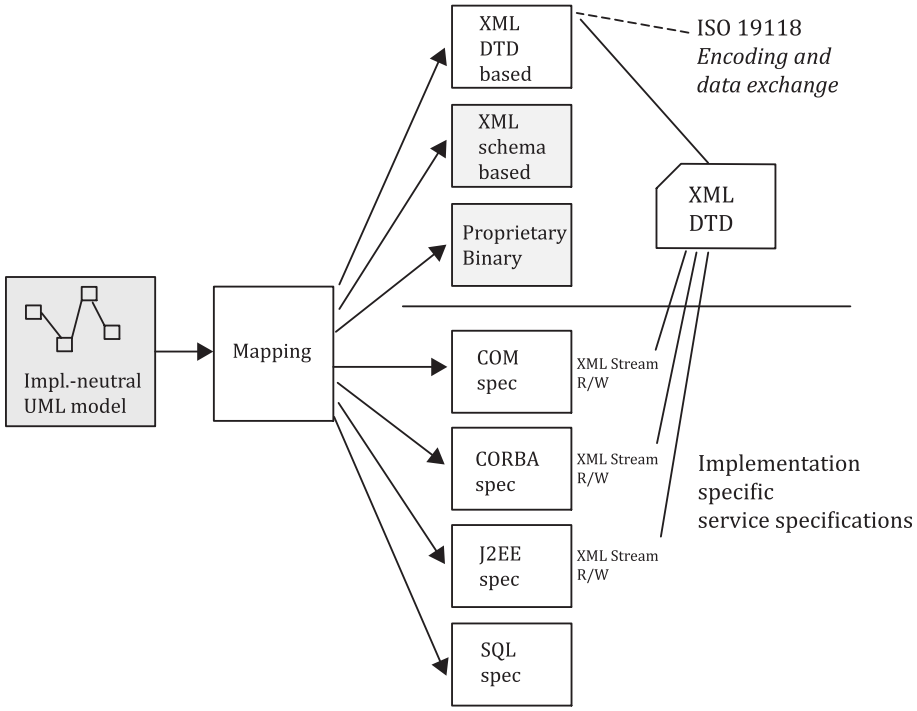


Figure C.2 — From platform-neutral to platform-specific specifications

Figure C.2 illustrates a direct mapping, without going through a platform-specific UML model. The top part also illustrates that the same approach can be used for exchange formats, such as the XML DTD defined in ISO 19118, and optimized binary formats from the same base model.

Platform-neutral models are described in UML in accordance with the rules and guidelines in ISO 19103. These models are mapped to an XML-based encoding format in accordance with ISO 19118. Platform-specific models are created for target implementation environments and DCPs according to a mapping. The mapping describes how each construct of a platform-neutral model is handled on the platform-specific level. These principles are briefly described in C.2 to C.8.

C.2 UML constructs used in the ISO geographic information suite of International Standards

The following UML constructs are used in the ISO suite of geographic information standards and need to be mapped to be platform-specific constructs:

— packages;

— classes;

— attributes;

— basic data types;

— associations;

— operations;

— constraints.

The ISO 19103 basic data types (Table C.1) need to be mapped into concrete types in the various implementation environments.

Table C.1 — Summary of some of the ISO 19103 basic data types

Data type	Description
Number - Integer	an integer number
Number - Real	a float floating point real number, or a double that describes a signed, approximate, numeric value with a typical binary precision 53 (zero or absolute value 10[-308] to 10[308]).
Number - Decimal	number with decimals
CharacterString	string of characters
Date	string that follows the ISO 8601 formats for time
Time	string that follows the ISO 8601 formats for date
DateTime	string which follows the combined date/time format of ISO 8601
Boolean	quantity that takes the values TRUE or FALSE. – or UNKNOWN ?

C.3 Platform-neutral technology model

Figure C.3 shows a platform-neutral model of various services available in a typical target environment, such as COM, CORBA or Java 2 Enterprise Edition. It shows that there is a basic IT technology support for the various services described in the extended OSE model.

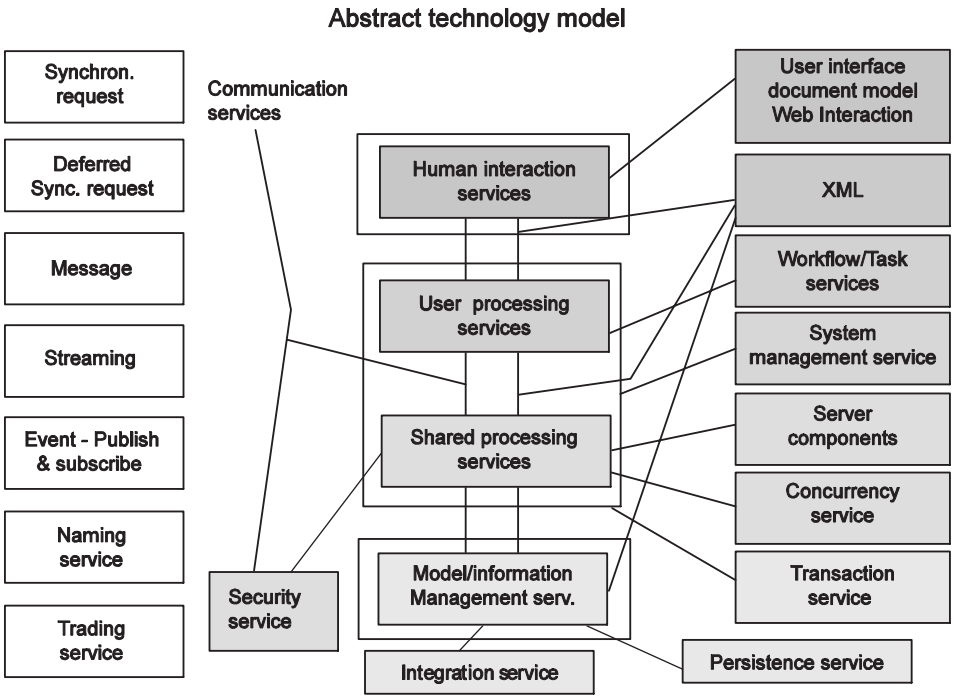


Figure C.3 — Platform-neutral abstract technology model

Human interaction services are supported by various user interface mechanisms, including web-browsers.

Communication services are supported by a principal set of 3 + 2 interaction modes: operations (synchronous or deferred synchronous requests), signals (event publish/subscribe or asynchronous messaging) and flows (with streaming). Associated with this is general naming and trading services.

Workflow/Task services are supported by a set of workflow/task services.

System management services are supported by various kinds of user, application and security management services.

Processing services are separated into user and shared services, where user services typically are supported in a single-user mode, while shared services add functionality for server-side and multi-user support with concurrency and transactions.

Model/Information management services are supported through data storage, persistence and manipulation services, potentially including various legacy system/format integration services.

It is possible to describe platform-neutral services using these generic platform-neutral concepts, and to define a mapping for how to support these in various environments.

C.4 to C.8 describe how various implementation environments and DCPs provide these services.

C.4 Mapping to CORBA-specific service specifications

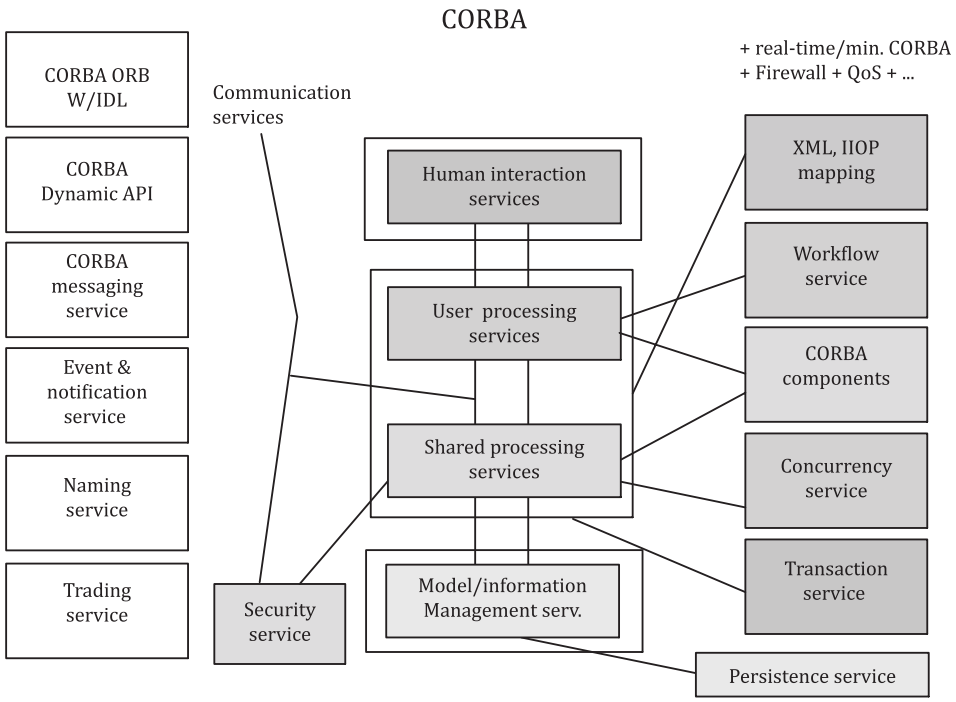


Figure C.4 — CORBA Technologies related to abstract architecture model

Human interaction services are not directly supported.

Communication services are supported by the CORBA ORB and dynamic API, as well as with the CORBA messaging service and event and notification service. Further support for communication of XML-structures will be provided by the CORBA XML-value mapping.

Workflow/Task services are supported by the CORBA workflow service.

System management services are supported by CORBA security and associated user services.

Processing services are supported by server-side CORBA-objects and the concurrency and transaction service. In CORBA 2, the CORBA components model will provide further services for server-side objects.

Model/Information management services are supported by the CORBA persistence service.

For mappings from platform-neutral UML models to CORBA, the following apply:

— packages: typically mapped to CORBA modules;

— classes: typically mapped to interfaces;

— attributes: typically mapped to attributes of interfaces;

— basic data types: mapped according to Table C.1;

— associations: typically mapped to access operations in the involved interfaces; in the case of attributes of the association, it might be mapped to a separate association interface;

— operations: typically mapped to operations in interfaces; exceptions are mapped to CORBA exceptions;

— constraints: typically not mapped directly into an implementation.

Table C.2 — From ISO 19103 to CORBA

ISO 19103 **data type**	**Suggested CORBA mapping**
Number - Integer	long, short, unsigned short, unsigned long
Number - Real	float, double
Number - Decimal	decimal
CharacterString	Wstring, string
Date	date
Time	time
DateTime	DateTime
Boolean	boolean

C.5 Mapping to MS COM-specific service specifications

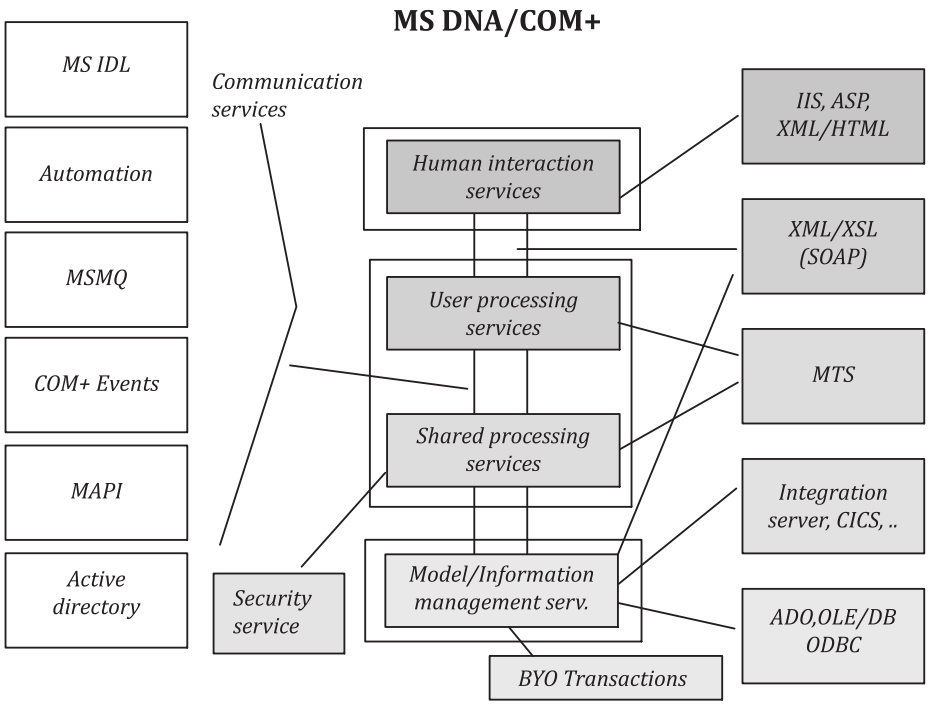


Figure C.5 — MS DNA/COM Model

Human interaction services are supported by the Windows windowing system and through support for web-browsers, typically with web-server support such as Active Server Pages and increased support for XML and XSLT.

Communication services are supported by the DCOM and Automation for synchronous requests, as well as with MSMQ messaging service and the COM+ event and notification service. Further support for communication of XML-structures is being provided through Simple Object Access Protocol (SOAP), providing a synchronous RPC-mechanism using HTTP and XML.

Workflow/Task services are not directly supported.

System management services are supported by Windows NT/2000 Security and associated user services.

Processing services are supported by server-side COM-objects and the associated MTS concurrency and transaction service.

Model/Information management services are supported by OLE/DB, ADO and ODBC.

For mappings from platform-neutral UML models to COM, the following apply:

— packages: typically mapped to COM modules;

— classes: typically mapped to COM interfaces;

— attributes: typically mapped to attributes of interfaces;

— basic data types: mapped according to Table C.3;

— associations: typically mapped to access operations in the involved interfaces; in the case of attributes of the association, it might be mapped to a separate association interface;

— operations: typically mapped to operations in interfaces; exceptions are mapped to CORBA exceptions;

— constraints: typically not mapped directly into an implementation.

Table C.3 — From ISO 19103 to MS COM

ISO 19103 **data type**	**Suggested COM mapping**
Number - Integer	long, short, unsigned short, unsigned long
Number - Real	float, double
Number - Decimal	decimal
CharacterString	char
Date	date
Time	time
DateTime	DateTime
Boolean	boolean

C.6 Mapping to J2EE/EJB-specific service specifications

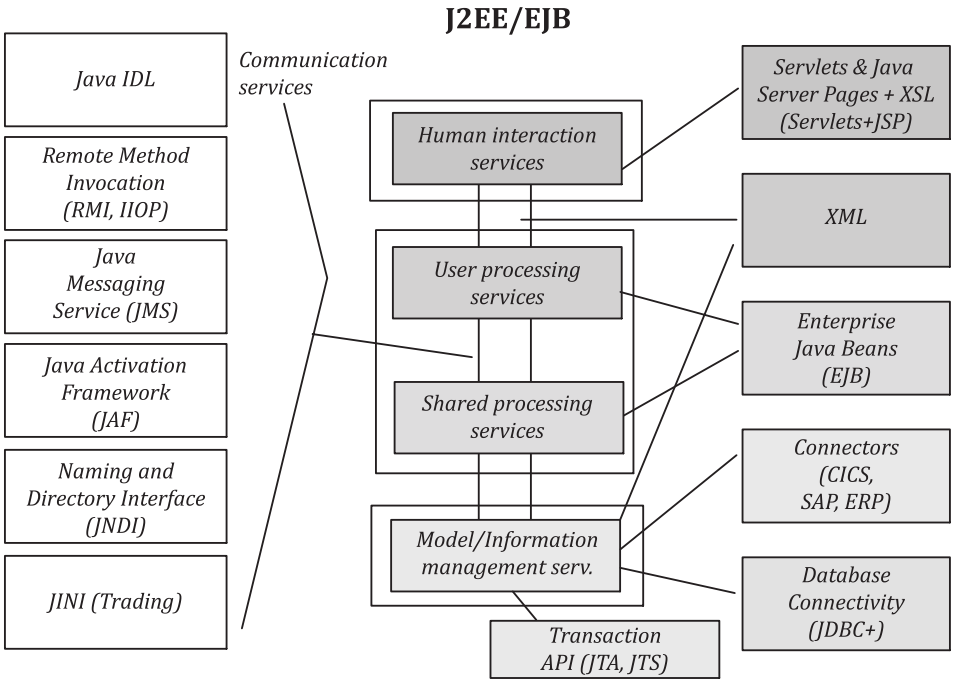


Figure C.6 — J2EE/EJB model

Human interaction services are supported by the Java windowing system and through support for web-browsers, typically with web-server support such as Java Server Pages (JSP) and increased support for XML and XSLT.

Communication services are supported by Java RMI, as well as with the Java messaging service and event and notification through the messaging service. Further support for communication of XML-structures will be provided by the Java XML-API.

Workflow/Task services are not supported directly.

System management services are supported by Java Security and associated user services.

Processing services are supported by server-side Enterprise Java Beans (EJB) and the associated concurrency and transaction service.

Model/Information management services are supported by the JDBC and Java persistence services, as well as the current Java serialisation.

For mappings from platform-neutral UML models to Java, the following apply.

— packages: typically mapped to Java modules;

— classes: typically mapped to interfaces;

— attributes: typically mapped to attributes of interfaces;

— basic data types: mapped according to Table C.4;

— associations: typically mapped to access operations in the involved interfaces; in the case of attributes of the association, it might be mapped to a separate association interface;

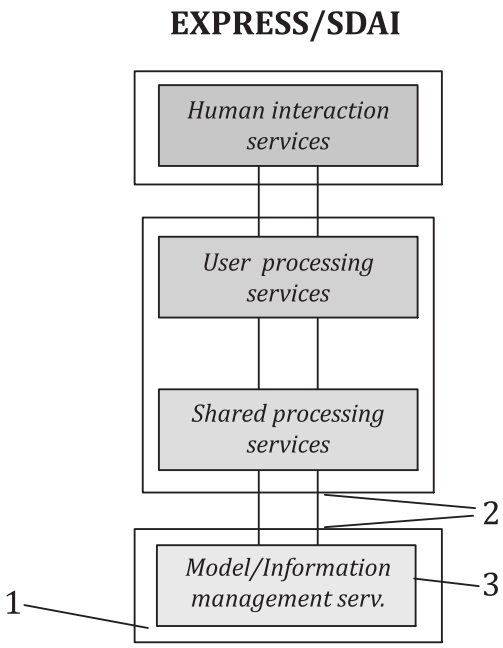
— operations: typically mapped to operations in interfaces; exceptions are mapped to Java exceptions;

— constraints: typically not mapped directly into an implementation.

Table C.4 — From ISO 19103 to Java 2

ISO 19103 **data type**	**Suggested Java mapping**
Number - Integer	int, short, long
Number - Real	Float, double
Number - Decimal	Decimal
CharacterString	String
Date	Date
Time	Time
DateTime	DateTime
Boolean	Boolean

C.7 Mapping to EXPRESS/SDAI-specific service specifications



Key

1 underlying storage and file systems
2 SDAI interface
3 data stored according to EXPRESS schemas

Figure C.7 — EXPRESS/SDAI model

The EXPRESS environment is focusing on providing model/information management services.

For mappings from platform-neutral UML models to EXPRESS, the following apply:

— packages: typically mapped to schemas;

— classes: typically mapped to entities;

— attributes: typically mapped to attributes of entities;

— basic data types: mapped according to Table C.5;

— associations: typically mapped to attributes of the involved entities;

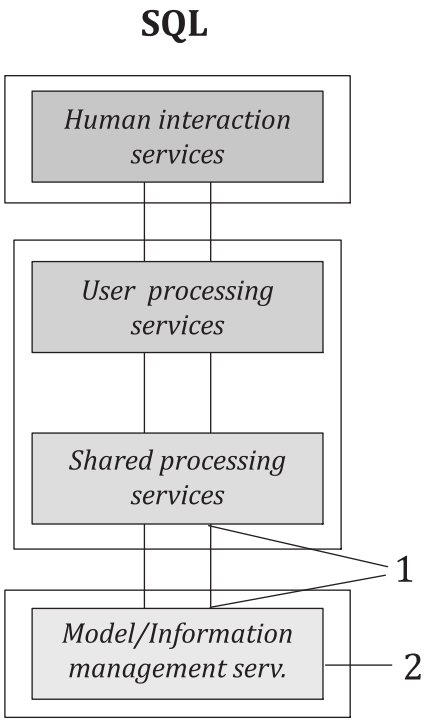
— operations: typically mapped to events in EXPRESS-2;

— constraints: typically mapped into EXPRESS rules.

Table C.5 — From ISO 19103 to EXPRESS

ISO 19103 **data type**	**Suggested EXPRESS mapping**
Number - Integer	Integer
Number - Real	Real
Number - Decimal	Decimal
CharacterString	String
Date	Date
Time	Time
DateTime	DateTime
Boolean	Boolean, Logical

C.8 Mapping to SQL-specific service specifications



Key

1 SQL API

2 data stored according to SQL table representations

Figure C.8 — SQL model

The SQL environment is focused on providing model/information management services.

For mappings from platform-neutral UML models to SQL, the following apply:

— packages: typically mapped to groups of tables;

— classes: typically mapped to tables;

— attributes: typically mapped to columns in tables;

— basic data types mapped according to Table C.6;

— associations: typically mapped to tables;

— operations: typically mapped to stored procedures/functions or are not supported;

— constraints: typically not mapped directly into an implementation.

Table C.6 — From ISO 19103 to SQL

ISO 19103 **data type**	**Suggested SQL mapping**
Number - Integer	INTEGER, SMALLINT
Number - Real	NUMERIC, REAL, FLOAT, DOUBLE PRECISION
Number - Decimal	Decimal
Binary	BIT, BIT VARYING, BINARY LARGE OBJECT
CharacterString	(NATIONAL) CHARACTER VARYING
Date	DATE
Time	TIME
DateTime	TIMESTAMP (WITH TIMEZONE), INTERVAL
Boolean	BOOLEAN

Annex D
(informative)

Use case-based methodology

Annex D provides an overview about a geographic information-oriented Use Case Analysis Methodology. This methodology is derived from the SERVUS methodology[43] that aims at a Design Methodology for Information Systems based upon Geospatial Service-oriented Architectures and the Modelling of Use Cases and Capabilities as Resources.

The SERVUS methodology relies upon a resource model as a common modelling language which is derived from the Representational State Transfer (REST) architectural style for distributed hypermedia systems as conceived in Reference.[6] Hereby, a resource is considered to be an information object that is uniquely identified, may be represented in one or more representational forms (e.g. as a diagram, XML document or a map layer) and support resource methods that are taken from a limited set of operations whose semantics are well-known (uniform interface). A resource has own characteristics (attributes) and is linked to other resources forming a resource network. Furthermore, resource descriptions may refer to concepts of the domain model (design ontology) using the principle of semantic annotation, yielding so-called semantic resources.

Figure D.1 shows the focus of the RM-ODP viewpoints typically during the steps of service analysis, design and implementation. The main viewpoint during initial analysis is the enterprise viewpoint. This also serves as the foundation for describing the resources (in terms of data, services and/or sensor information) which is required. An initial step will then be to compare the requested resources with potentially offered resources through a discovery and search process, in order to identify if the request for resources can be met by resources that already are available.

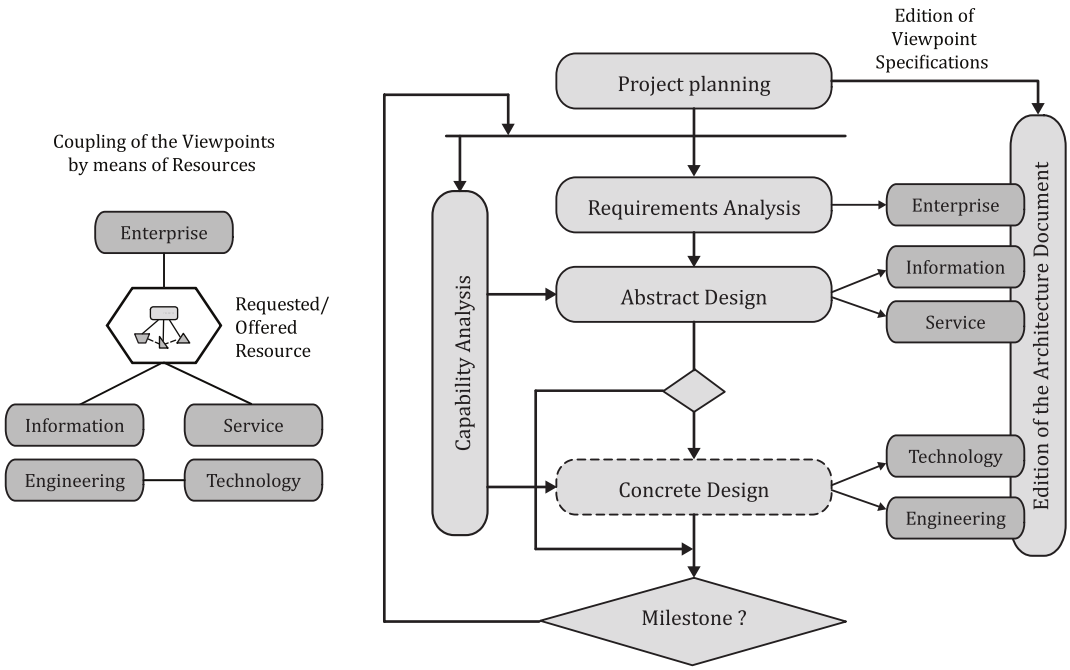


Figure D.1 — Relationship of RM-ODP viewpoints and analysis and design

Use case modelling has been shown to be an efficient and powerful approach to reach a common understanding of the system itself and its behaviour. In interdisciplinary projects, involving thematic experts from different domains (e.g. geospatial, environmental) as well as IT experts, it is as challenging

to reach consensus on a common terminology. Otherwise, the consequences would include different interpretations and assumptions about the systems to be developed. Thus to avoid misunderstandings, use case descriptions should be based on a common vocabulary, stemming from a glossary and a thesaurus whenever possible.

The description of use cases is necessary to capture all functional and non-functional requirements of the system. The use cases also describe the interaction between the users and the system. Use cases are the most common practices for capturing and deriving requirements. The requirements of the system are described in a narrative way with minimal technical jargon.

In the geospatial context, use cases are typically described in a semi-formal way, based on a structured textual description in tabular form derived from a template. Various European research projects, such as SANY [EU FP7 project no. 033564 Sensors Anywhere (SANY) - http:///www.sany-ip.eu], ENVIROFI (EU FP7 FI PPP project, ENVIROFI, http:///www.envirofi.eu) ENVISION (EU FP7 project no. 1234, ENVironmental Services Infrastructure with ONtologies, www.envision-project.eu), EO2HEAVEN [EU FP7 project no. 244100 Earth Observation and ENVironmental modeling for the mitigation of HEAlth risks (EO2HEAVEN) - http://www.eo2heaven.org/] and TRIDEC [EU FP7 project no. 258723 Collaborative, Complex and Critical Decision-Support in Evolving Crisis (TRIDEC) - http://www.tridec-online.eu], based the description of their use cases on a similar template.

Based upon this approach, additional information about the requested information resources (e.g. type and format of needed data) is necessary to completely describe a use case from both a user's and system's point of view. The requirements should be derivable from the use cases. The following three types of requirements can be identified:

— functional requirements;

— informational requirements;

— non-functional requirements.

Functional requirements can be derived from the sequence of actions (main success scenario, extensions and alternative paths). The informational requirements address data that is exchanged between two communication partners, i.e. between users and the system or between system components. The non-functional requirements cover all requirements that do not alter the foreseen functionality of the system, e.g. the quality of data and results.

This approach provides a basis for use case development. However, the SERVUS methodology proposes that beside the functional and non-functional requirements, the informational requirements are very important to complete the use case description. For a more detailed analysis and as a first step towards information modelling, it is necessary to consider input data, data format, data type, data encoding, and the desired format of the output data, too. Thus, the template contains additional issues like "Requested Information Resources".

The common form of a use case description is to describe it from the user's point of view where only the external perceivable behaviour is reflected. The described system is a black box for the user. This template should be used by both sides. The users and the system developers and operators. Both sides and all involved experts have to understand the use cases in the same way. Especially, the IT experts should understand the user's requirements because they have to develop the IT components on the basis of the descriptions.

It is expected that each use case will be described in a semi-formal way. A form was created to structure the textual description. The table represents the use case template and is shown in Annex E. The methodology describes the use case template items, explains what each item mean, instructs how to fill them out and includes additional examples and tips. Use Case Analysis Process

Figure D.2 illustrates the analysis phase as a prelude of the SERVUS Design Methodology.[44] As a first step of an analysis, iteration loop is a set of preliminary use cases (UC) is identified, mostly by those thematic experts who drive the study.

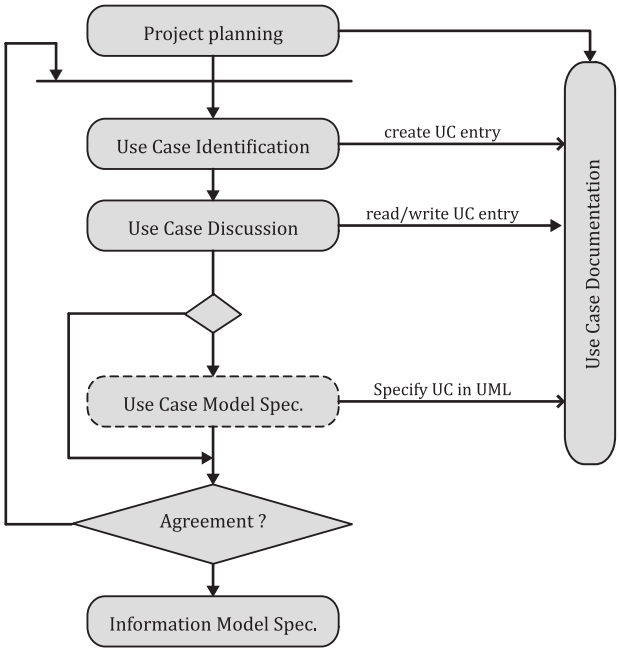


Figure D.2 — Procedure of use case analysis

The methodology proposes that use cases are initially described in structured natural language but already contain the list of requested resources. This description is the language, which is used in the use case discussion that takes place in workshops that are facilitated by the system analyst. Depending on the level of agreement that can be reached, the iteration loop is entered again in order to refine or add new use cases.

In order to identify inconsistencies and check the completeness of the use case model, the system analyst may transform the semi-structural use case description into formal UML specifications. However, these UML diagrams should still be on a high abstraction level such that a discussion with the end-user is possible. It is the advantage of this formal transition step already in an early analysis phase to detect inconsistencies and missing information as quickly as possible. The UML specification helps to discuss and check the use cases together with the thematic experts.

However, in addition to the usual UML use cases, they already comprise the links to the set of requested (information) resources, their representation forms and the requirements to create, read, write or delete them. Once an agreement is reached about the set of use case descriptions and related UML specifications, it is then up to the system analyst to specify the resulting information model taking the resource model as a first guidance.

Annex E
(informative)

Example — Use case template

This template is an extended version of the original template defined by Reference [5] in particular extended with a possibility to describe Requested Information Resources found suitable in an SDI setting.

Table E.1 — Description of the use case template

Use case template	Description	Examples
Use Case Name	Name of the use case.	Visualize proposed water height after the tsunami event.
Use Case ID	Unique identifier of a use case.	
Revision and Reference	Revision = version number of use case ID. Reference = URL of the use case (you get the URL by right-clicking on the entry in the index column).	V02, http://SDI.server.de/servlet/is/4900/
Use Case Diagram	Description of the UML use case diagram for the actual use case. The diagram should include and extend relationships if there are any use cases that are related. The actual UML diagram figure may be added at the bottom of the template by uploading a bitmap generated from a UML editor.	UML diagram with the use case notation.
Status	Status of the use case development.	One of the following: — planned; — in progress.
Priority of accomplishment (optional)	The priority of the use case to be considered when assessing its importance for a development cycle.	One of the following. — Shall have: The system shall implement this goal/assumption to be accepted. — Should have: The system should implement this goal/assumption: some deviation from the goal/assumption as stated may be acceptable. — Could have: The system should implement this goal/assumption, but may be accepted without it.
Goal	Short description (max. 100 characters) of the goal to be achieved by a realization of the use case.	System generates alerts based on user observations.
Summary	Comprehensive textual description of the use case.	The user opens the browser which shows map-window with the water height after the tsunami event in the affected area.
Category	Categorization of use cases according to overall reference architecture.	*Context dependent*

Table E.1 *(continued)*

Use case template	Description	Examples
Actor	List of users of the use case (actors).	Examples may be citizen, administrator or employee of a SDI agency.
Primary Actor (initiates)	Actor that initiates the use case execution.	
Stakeholder (optional)	Company, institution or interest group concerned by the execution of the use case.	
Requested Information Resources (optional)	Information category or object that is required to execute the use case or is being generated during the course of the use case execution. The requested information resource shall be listed together with its requested access mode (create, read, update or delete) or "manage" which encompasses all access modes.	— user observation (read) — user-specific effect (read, update) — alert (manage)
Preconditions	Description of the system/user status statement) that is required to start the execution of the use case. Note that use cases can be linked to each other via "preconditions". This means, a precondition for a use case can be either an external event or another use case. In this case, the use case ID should be provided in the field "preconditions".	The user has opened the portal successfully.
Triggers (optional)	(External) event that leads to the execution of the use case. Note that use cases can be linked to each other via "triggers". This means, a trigger for a use case can be either an external event or another use case. In this case, the use case ID should be provided in the field "triggers".	The user chooses water height forecast.
Main success scenario	Numbered sequence of actions (use case workflow) to be carried out during the execution of the use case.	1. User chooses assessment report. 2. User specifies one or more components (default should be all). 3. User sets a time-frame (last 24 h, last week, last month). 4. The system shows a report as graphical visualization.
Extensions	Extension of an action of the main success scenario. The action to be extended shall be referred to by its number (e.g. 1) appended by a letter (e.g. 1a).	1a. The user defines the temporal extent b. The user defines an unavailable temporal extent. A new dialogue window opens and requires a new temporal extent.
Alternative paths (optional)	Alternate path through the main success scenario w.r.t. an identified action.	4a. User can select to view report in different formats, e.g. tabular or graphical map.
Post conditions	Description of the system/user status (statement) that holds true after the successful execution of the use case.	Report is displayed on the screen.

Table E.1 *(continued)*

Use case template	Description	Examples
Non-functional requirements	Description of non-functional requirements for this use case with respect to performance, security, quality of service or reliability.	Display of report expected after 20 s at the latest.
Validation statement	List of statements that indicate how to validate the successful realization of the use case.	
Notes	Additional notes or comments (also by other users).	
Author and date	Author of use case, date of last edition.	

Annex F
(informative)

Service modelling – SoaML

Service oriented architecture Modeling Language (SoaML) is a UML profile and metamodel for service modelling standardised by OMG. The SoaML specification defines three different approaches to specifying services; simple interfaces, service interfaces and service contracts.

The SoaML specification defines a UML profile and a metamodel that extends UML to support the range of modelling requirements for SOA, including the specification of systems of services, the specification of individual service interfaces, and the specification of service implementations. The SoaML metamodel extends the UML metamodel to support an explicit service modelling in distributed environments. This extension aims to support different service modelling scenarios such as single service description, service-oriented architecture modelling, or service contract definition. This is done in such a way as to support the automatic generation of derived artefacts following the approach of Model Driven Architecture (MDA).

UML is a general-purpose modelling language for visualizing, specifying, constructing and documenting artefacts of software-intensive systems. A UML profile customizes UML for a specific domain or purpose by using extension mechanisms such as stereotypes and metaclasses. Figure F.1 shows the main stereotypes defined in the UML profile for SoaML, e.g. the stereotype «ServiceInterface» extends the UML metaclass *Class*.

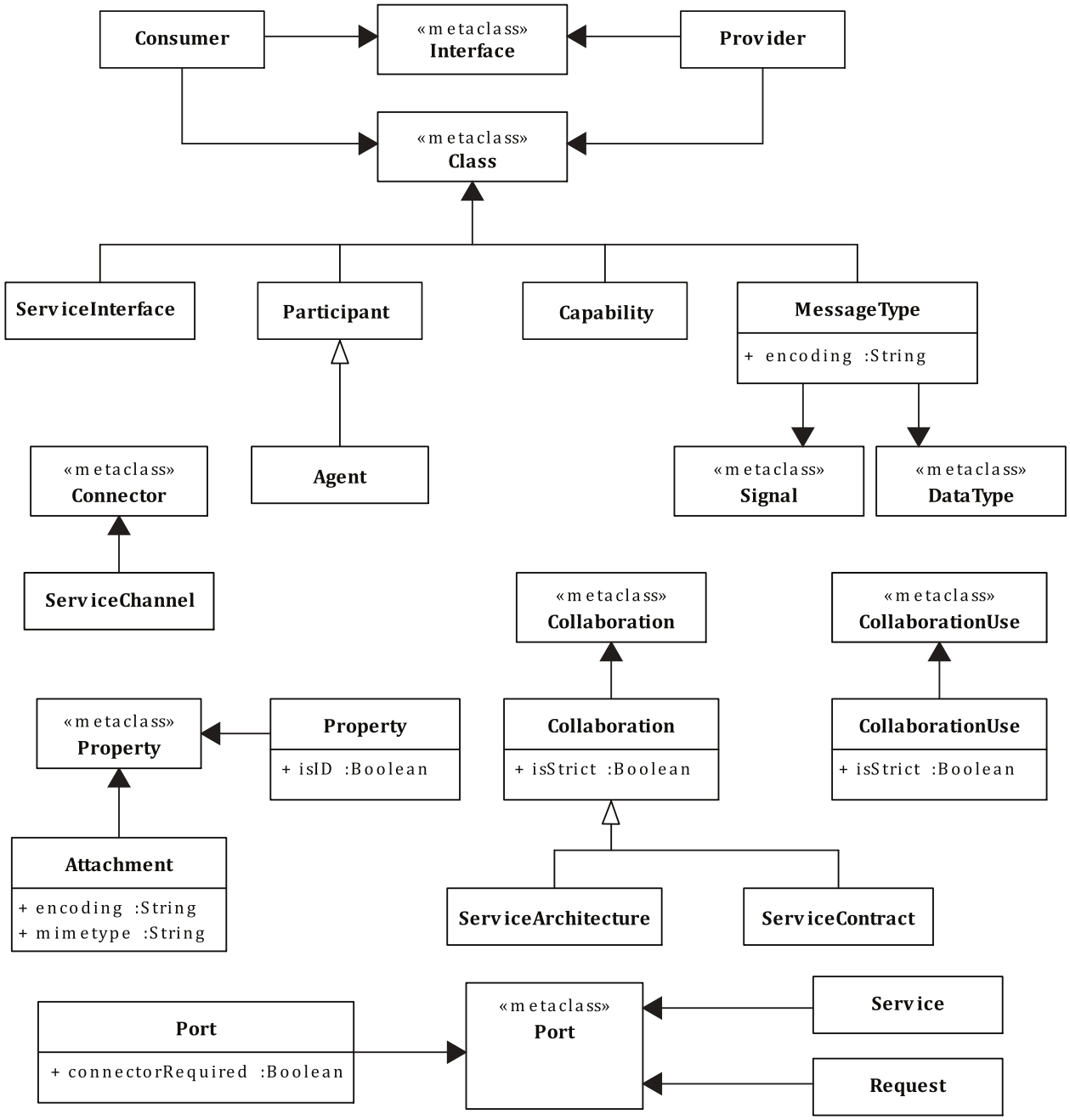


Figure F.1 — Main UML extensions defined as stereotypes in the UML Profile for SoaML

SoaML extends UML in six main areas.

— *Participants* are used to define the service providers and consumers in a system. A participant may play the role of service provider, consumer or both. When a participant acts as a provider, it contains *service ports* and when a participant acts as a consumer, it contains *request ports*.

— *Service interfaces* are used to describe the operations provided and required to complete the functionality of a service. A service interface can be used as the protocol for a service port or a request port.

— *Service contracts* are used to describe interaction patterns between service entities. A service contract is used to model an agreement between two or more parties. Each service role in a service contract has an interface that usually represents a *provider* or a *consumer* (service contracts are not required in this International Standard).

— *Services architectures* are used to define how a set of participants works together for some purpose by providing and using services. The services are expressed as service contracts in a service architecture (service architectures are not required in this International Standard).

— *Service data* (with MessageType) are used to describe service messages and message attachments. The *message type* is used to specify the information exchanged between service consumers and providers. An *attachment* is a part of a message that is attached to rather than contained in the message.

— *Capabilities* represent an abstraction of the ability to affect change. Capabilities identify or specify a cohesive set of functions or resources that a service provided by one or more participants might offer.

SoaML supports different approaches to SOA. This has resulted in the definition of different but overlapping language constructs in the UML profile. The specification distinguishes between three different approaches to specifying a service.

— The **simple interface**-based approach uses a UML interface to specify a one-way service interaction.

— The **service contract**-based approach extends a UML collaboration to specify a binary or n-ary service interaction.

— The **service interface**-based approach extends a UML class to specify a binary or n-ary service interaction. Both the service contract and service interface-based approaches entail the specification of simple interfaces, typically one for each of the roles participating in the service interaction. Thus, a service contract or a service interface can be seen as an extension of the simple interface-based approach.

SoaML supports the specification of multiple SOA architectural styles, such as both synchronous (RPC) style and asynchronous document/RESTful services style, and also events/signals and sensors as services.

Typically, the creation of a service model comprises the following aspects:

— identify services, the requirements they are intended to fulfil, and the anticipated dependencies between them;

— specify services including the functional capabilities they provide, what capabilities consumers are expected to provide, the protocols or rules for using them, and the service information exchanged between consumers and providers;

— defining service consumers and providers, what services they consume and provide, how they are connected and how the service functional capabilities are used by consumers and exposed by providers in a manner consistent with both the service specification protocols and fulfilled requirements;

— defining the policies for using and providing services and the quality of service provided.

Bibliography

[1] Berners-Lee T, Fielding R, Masinter L (1998). Uniform Resource Identifiers (URI): Generic Syntax. Internet Engineering Task Force (IETF) Memo – RFC 2396

[2] Berners-Lee T., Hendler J., Lassila O. The Semantic Web. Scientific American Magazine, 2001

[3] Bizer C. The Emerging Web of Linked Data. IEEE Intell. Syst. 2009, **24** (5) pp. 87–92

[4] Catalogue of OMG Specifications, Object Management Group. available at <http://www.omg.org/spec/>, accessed 15 June 2014

[5] Cockburn A. Writing Effective Use Cases. 2001, **ISBN-13** p. 9780201702255 [Addison-Wesley]

[6] Fielding R.T. Architectural Styles and the Design of Network-Based Software Architectures, PhD thesis. University of California, Irvine, 2000., available at http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm, accessed on 15 June 2014

[7] Francisco J., Lopez-Pellicer L.M., Vilches-Blazquez F., Zarazaga-Soria J., Muro-Medrano P., Corcho O. 2012). The Delft Report: Linked Data and the Challenges for Geographic Information Standardization, available at http://www.rcg.cat/articles.php?id=220, accessed on 15 June 2014

[8] Percivall G. 2003) OGC Reference Model, OGC 03-040, available at <http://www.opengeospatial.org/standards/orm> accessed 15 June 2014

[9] Holtman K, & Mutz A (1998). Transparent Content Negotiation in HTTP. Internet Engineering Task Force (IETF) Memo – RFC 2295

[10] ISO 10303-11, *Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual*

[11] ISO 10303-22, *Industrial automation systems and integration — Product data representation and exchange — Part 22: Implementation methods: Standard data access interface*

[12] ISO 8601, *Data elements and interchange formats — Information interchange — Representation of dates and times*

[13] ISO 19101-1, *Geographic information — Reference model — Part 1: Fundamentals*

[14] ISO 19105, *Geographic information — Conformance and testing*

[15] ISO 19107, *Geographic information — Spatial schema*

[16] ISO 19108, *Geographic information — Temporal schema*

[17] ISO 19109, *Geographic information — Rules for application schema*

[18] ISO 19110, *Geographic information — Methodology for feature cataloguing*

[19] ISO 19111, *Geographic information — Spatial referencing by coordinates*

[20] ISO 19112, *Geographic information — Spatial referencing by geographic identifiers*

[21] ISO 19116, *Geographic information — Positioning services*

[22] ISO 19117, *Geographic information — Portrayal*

[23] ISO 19118, *Geographic information — Encoding*

[24] ISO/TR 19121, *Geographic information — Imagery and gridded data*

[25] ISO 19123, *Geographic information — Schema for coverage geometry and functions*

[26] ISO 19125-1, *Geographic information — Simple feature access — Part 1: Common architecture*

[27] ISO 19128, *Geographic information — Web map server interface*

[28] ISO 19142, *Geographic information — Web Feature Service*

[29] ISO 19143, *Geographic information — Filter encoding*

[30] ISO 19150 (all parts), *Geographic information — Ontology*

[31] ISO 19153, *Geospatial Digital Rights Management Reference Model (GeoDRM RM)*

[32] ISO 19157, *Geographic information — Data quality*

[33] ISO/IEC 2382:2015, *Information technology — Vocabulary*

[34] ISO/IEC 9075 (all parts), *Information technology — Database languages — SQL*

[35] ISO/IEC 11179-3, *Information technology — Metadata registries (MDR) — Part 3: Registry metamodel and basic attributes*

[36] ISO/IEC 14882, *Information technology — Programming languages — C++*

[37] ISO/IEC 19505-2, *Information technology — Object Management Group Unified Modeling Language (OMG UML) — Part 2: Superstructure*

[38] ISO/IEC 19510, *Information technology — Object Management Group Business Process Model and Notation*

[39] ISO/IEC 19793, *Information technology — Open Distributed Processing — Use of UML for ODP system specifications*

[40] Klyne G, & Carroll JJ (2004). Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, 10 February 2004

[41] Béjar R., Latre M.A., Nogueras-Iso J., Muro-Medrano P.R., Zarazaga F.J. An RM-ODP Enterprise View for Spatial Data Infrastructures. Comput. Stand. Interfaces. 2012, **34** pp. 263–272. Available at: http://dx.doi.org/10.1016/j.csi.2011.10.001

[42] Schade S., Mazzetti P., Sabeur Z., Havlik D., Usländer T., Berre A. 2011). Towards a Multi-Style Service-Oriented Architecture for Earth Observations. EGU General Assembly 2011, Vienna, Austria

[43] Usländer T. Service-oriented Design of Environmental Information Systems. PhD thesis of the Karlsruhe Institute of Technology (KIT). Faculty of Computer Science, KIT Scientific Publishing, 2010., available at http://digbib.ubka.uni-karlsruhe.de/volltexte/1000016721

[44] Usländer T., & Batz T. How to Analyse User Requirements for Service-Oriented Environmental Information Systems. In: ISESS 2011. IFIP AICT, (Hřebíček J., Schimak G., Denzer R. eds.). Springer, Heidelberg, **Vol. 359**, 2011, pp. 165–72.

[45] Web Services Business Process Execution Language (WSBPEL) 2.0, Organization for the Advancement of Structured Information Standards (OASIS), available at https://www.oasis-open.org/standards#wsbpelv2.0

[45] ISO/IEC 10746-2:2009, *Information technology — Open distributed processing — Reference model: Foundations — Part 2*

[46] SOA-RAF Reference Architecture Foundation for Service Oriented Architecture Version 1.0, OASIS Standard, 4 December 2012 [3)]

3) http://docs.oasis-open.org/soa-rm/soa-ra/v1.0/cs01/soa-ra-v1.0-cs01.pdf

[47] SOA-RM Reference Model for Service Oriented Architecture 1.0, OASIS Standard, 12 October 2006.[4)]

[48] QoS UML tM Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms (QFTP) v1.1, April 2008, OMG standard[5)]

4) http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf

5) http://www.omg.org/spec/QFTP/1.1

ICS 35.240.70

Price based on 103 pages