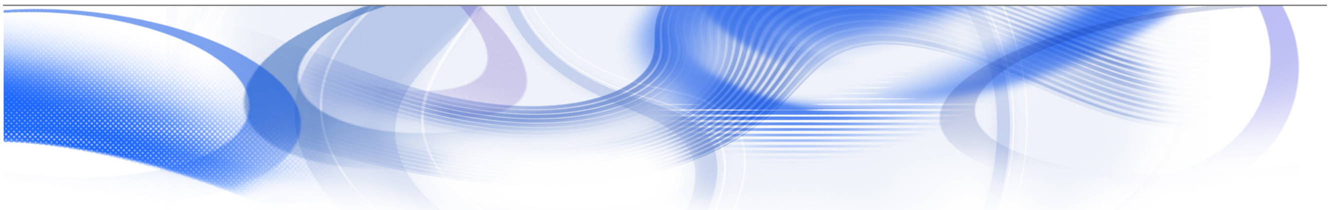
JRC Scientific and Technical Reports



Data State of Play

Compliance Testing and Interoperability Checking

S. Viganó
TXT e-solutions

M. Millot
European Commission DG JRC,
Institute for Environment and Sustainability

EUR 23045 EN - 2007

The mission of the Institute for Environment and Sustainability is to provide scientific-technical support to the European Union's Policies for the protection and sustainable development of the European and global environment.

European Commission
Joint Research Centre
Institute for Environment and Sustainability

Contact information

Michel Millot

Address:
European Commission, Joint Research Centre,
Institute for Environment and Sustainability
Spatial Data Infrastructure Unit
T.P. 262
Via E. Fermi, 2749
I-21027 Ispra (VA)
ITALY

E-mail: michel.millot@jrc.it
Tel.: +39-0332786146
Fax: +39-0332786325

http://ies.jrc.ec.europa.eu
http://www.jrc.ec.europa.eu

Stefano Viganó

Address:
TXT e-solutions
Via Frigia, 27
I-20126 Milano (MI)
ITALY

E-Mail: stefano.vigano@txt.it
Tel. +39-02257711
Fax.+39-022578994

http://www.txtgroup.com/it/index.shtml

Legal Notice

Neither the European Commission nor any person acting on behalf of the Commission is responsible for the use which might be made of this publication.

JRC 41822

EUR 23045 EN
ISBN 978-92-79-07817-0
ISSN 1018-5593
DOI 10.2788/55339

Luxembourg: Office for Official Publications of the European Communities

Printed in Italy

Table of Contents

1. Introduction

The situation of GIS data in Europe in the last years has pointed out different problems and critical situations mainly due to:

- Data policy restrictions
- Lack of co-ordination
- Lack of standards
- Difficulty (when not impossibility) in the reuse of existing data

This situation therefore aroused different needs like:

- Better information needed to support policies
- Improvement of existing information flows
- Differentiation across regions to be considered
- Revision of approach to reporting and monitoring, moving to concept of sharing of information

Above all, the ongoing general European integration development has created increasing demand for cross-border data services. This need is exemplified by the wide set of European projects and initiatives aimed at improving the data compatibility and developing integrated services. These include individual projects with generic approach like GINIE (GINIE, 2003), domain-specific developments like EuroRoadS (EuroRoadS, 2003) or EULIS (EULIS, 2003), and initiatives with wider scope like the INSPIRE framework.

The DIRECTIVE 2007/2/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 14 March 2007 establishes an Infrastructure for Spatial Information in the European Community (INSPIRE)

The Directive requires Member States to take a set of measures, ranging from the type of data shared between users and application up to the network of services that allow searching for, viewing and accessing these data.

The overall aim of the INSPIRE proposal is to improve the way in which spatial data held by public authorities supports environmental policy, by improving the harmonization of spatial data and the interoperability of spatial services and ensuring greater sharing of the data between public authorities and on-line access by the public.

INSPIRE lays down general rules for the establishment of an infrastructure for spatial information in Europe, for the purposes of environmental policies and policies or activities which may have a direct or indirect impact on the environment.

The INSPIRE directive therefore includes various proposals, each addressing a given environment or hazard like for example flooding, pesticides, climate changes etc.

The programme has been articulated on the following different phases:

- Preparatory phase (2004-2006)
  - Co-decision procedure

  - Preparation of Implementing Rules 2005 – 2008 …
- Transposition phase (2007-2008)
  - Directive enters into force
  - Transposition into national legislation
  - INSPIRE Committee starts its activities
  - Adoption of Implementation Rules by Committology
- Implementation phase (2009-2013)
  - Implementation and monitoring of measures

More details on this programme with respect to the argument of this study (see paragraph 1.3) will be given in the dedicated chapter 4.

1.1 Besides INSPIRE

INSPIRE is not the only one data infrastructure programme that provides an answer to the problems previously outlined. Outside the European Community boundary there are presently many more data infrastructures related to delimited geo-political regions in the world.

For example:

China	→	NFGIS (Based on ISO TC 211 and 19100 standards[1])
India	→	NGDI (National Geospatial Data Infrastructure)
Canada	→	CGDI (Canadian Geospatial Data Infrastructure)
USA	→	NSDI (National Spatial Data Infrastructure)
Australia	→	ASDI (Australian Spatial Data Infrastructure)

Each of this SDI has faced, or it is still facing, specific interoperability and harmonisation issues. A description of these data infrastructures is beyond the scope of this study, see [7] for a detailed overview.

[7] is a limited comparative review of the NSDIs in Australia, Canada and the United States on the one hand and the European INSPIRE-initiative for a ESDI on the other hand. The comparison reflects upon the similarities and the differences of these NSDI juxtaposed to the NSDI situation in the European Union, taking account as possible of the INSPIRE position papers.

1.2 Goal of this study

This study aims to analyse and evaluate the state of play for compliance and interoperability testing related to the INSPIRE technical chapters, i.e. data, metadata and services.

The INSPIRE directive will require the Member States to adopt and conform to a set of requirements called Implementing Rules. Any future INSPIRE implementation shall therefore comply with these rules and there will be the need of software products, solutions or architectures able to check/certify the respect of these rules.

[1] China has tried out national standards before converging to international ones.

This study investigates the market looking for the above mentioned solutions over each of the three INSPIRE Technical chapters and try to evaluate their pertinence with data, metadata and service IR.

In detail this specific document focuses only on the Data issue.

1.3 INSPIRE Present Situation

In the INSPIRE web site[2] it is possible to know the up to date situation and at the time of writing, the latest document available on the web is related to "The Methodology for the Development of Data Specifications (D 2.6)" [5].

[5] together with the Drafting Team's previous *Generic Conceptual Model* [4], and with the to be issued "Guidelines for the Encoding of the Spatial Data" constitute the baseline of the data specifications required for interoperability and harmonisation.

The next sub-paragraphs detail the interoperability/harmonisation issues with reference to documents [4] and [5].

1.4 Data Interoperability and Harmonisation

In [5] you can find the following description of the methodology adopted by DTs for the development of harmonised data specifications:

"It is based on a cyclic approach for incrementally growing INSPIRE's degree of definition and implementation based on user requirements, in combination with a set of anchor point milestones to engage stakeholder commitment and bring about feasible and mutually satisfactory system solutions".

This spiral-like approach (see Figure 1.1) is mainly due to the fact that it is not possible to identify a unique development process model for how to create and analyse use cases and turn them into an application schema. More than one only process can be used successfully as well.

We must anyway consider that as long as we are dealing with INSPIRE Data Specifications, the themes defined in the annexes of the Directive will be the only reference required to structure them. However, the data specification development process will be driven by environmental use cases which typically will involve data from several themes, i.e. specified in several INSPIRE Data Specifications.

[2] http://www.ec-gis.org/inspire/

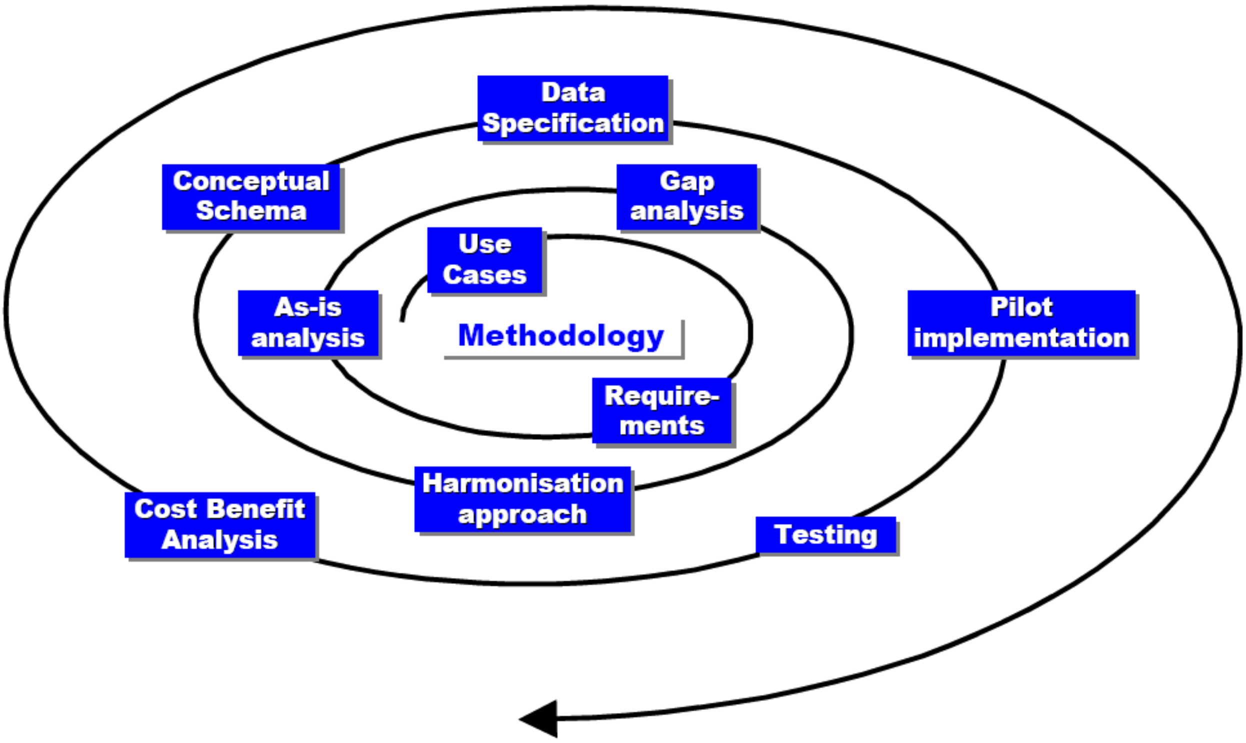


Figure 1.1: Spiral Approach to Definition of the Model Process ([5])

The spiral approach well detailed in the Harmonisation chapter of [4] “exemplifies a modelling approach where the user requirements are first modelled on the conceptual level and then converted to specifications on the implementation level. The application schema in UML will be converted to a GML/XML schema, where possible, or other community specific delivery formats that can be used, for example, in conjunction with the INSPIRE download Network Service.

An in-depth description of these languages will be given in the following paragraphs.

2. Purpose and Scope of the Document

In the previous chapter is introduced the argument of this study giving a picture of the existing scenario and the general situation related to the issues hereafter examined.

The third chapter describes the normative requirements expressed by the INSPIRE programme for the interoperability and harmonisation data, also including other possible standards not adopted by the programme.

Chapter 4 provides a synthesis of the INSPIRE requirements.

In chapter 5 the author gives a description of some products or software solutions that can be a valid help for the compliancy and interoperability testing activities, i.e. they check if a given solution can suit the previously expressed requirements. The document focuses only on few cases.

The next chapter outlines the pros and cons of this products according to the INSPIRE specifications. It also tries to present a critical view of the market scenario and the future evolution.

Chapter 7 is a collection of test cases related to some of the software solutions, described in the previous chapter.

Finally the last chapter draws the conclusion of this work with the final considerations to the data interoperability and harmonisation issue.

3. Normative References

When generic information must be shared and made available to different consumers, a set of needs arise to make the consumers able to acquire, understand and therefore use it.

These needs, generally speaking, involve what the literature defines as interoperability (of services) and harmonisation (of data). There are an important number of definitions on the web for these two words. The author has chosen the description given in the Wikipedia global Web encyclopaedia and exposed in the sections below.

The next paragraphs of this chapter first describe the INSPIRE approach to interoperability and harmonisation and finally try to review other methodologies like MDA or the emerging OWL.

3.1.1 Interoperability

The IEEE defines interoperability as the ability of two or more systems or components to exchange information and to use the information that has been exchanged.

With respect to software, the term interoperability is used to describe the capability of different programs to exchange data via a common set of business procedures, and to read and write the same file formats and use the same protocols.

According to ISO/IEC 2382-01, Information Technology Vocabulary, Fundamental Terms, interoperability is defined as follows: “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”.

These definitions have been quoted from the Wikipedia. For a definition more contextual to INSPIRE we propose to use the definition given in [5], where “interoperability is the ability of two or more autonomous entities to communicate and co-operate among themselves in a meaningful way despite differences in language, context or content”.

It must be underlined that the interoperability quality is not related to data sets. Interoperability is a feature related to (and can occur only between) services and systems. When different systems must share heterogeneous data sources, interoperability requires “wrapping data sources into services that conform to standards. The output of these services is what is to be harmonised, not their inputs (database schemas)”.

3.1.2 Harmonisation

Harmonizing data help to get a better consistency and comparability of data across information systems, usually within a particular program area. Consistency can be achieved through the cleansing of redundant or conflicting data and removal of storage replicas. The harmonization process makes information available for the integration of data systems and improves meaning and format across different systems. This result can be obtained by means of a data mapping process that compares the meanings and formats of involved data elements belonging to a specific area (for example GIS information).

The harmonisation process therefore applies transformation rules and definitions to the existing heterogeneous data elements in order to have a common representation of the same elements with an improved quality and consistency.

More details and information on the harmonisation issue can be found at the US EPA web site[3] where there is also a useful web page maintained by Environmental Data Registry. EDR allows you to search for and compare data elements from various sources (e.g., data standards, data systems). The Compare Tool is a fully integrated feature within the EDR.

An application schema will be described in a formal language, hereafter called the schema language, which captures the facts agreed upon by the different partners in the harmonisation process. The Generic Conceptual Model presented in [4] prescribes a profile of UML, as specified in ISO/TS 19103 and ISO 19109, to describe the application schema on the conceptual level – in line with the recommendations of the ISO 19100 series and the current practice in the GI community.

Ontology languages are currently a research topic and that in the future it may prove to be beneficial to express the ontology also in one of these emerging languages (e.g. OWL), but for the time being INSPIRE recommends to use a schema language that is well rooted and accepted in the GI and standards community, i.e. the UML profile specified by ISO/TS 19103 and ISO 19109.

Relevant ISO Standards

The following list includes the ISO standards involved with specifications from [4].

• UML	ISO 19103
• Conceptual Schema	ISO 19107/8/9
• Application Schema	ISO 19109
• Feature Type	ISO 19109
• Data Dictionary	ISO 19126
• Data Product Specification	ISO 19131
• Quality	ISO 19113/5
• Metadata	ISO 19115
• Multi Lingual	ISO 19139
• Portrayal	ISO 19117
• GML	ISO 19136
• Translating WFS	ISO 19142+43

The next sub-paragraphs give an up to-date picture of some of these standards

3.1.3 Unified Modelling Language

UML, as described in [15], is a language for specifying, constructing, visualizing, and documenting the artifacts of a software-intensive system. It fuses the concepts of Booch, OMT, and OOSE. The result is a single, common, and widely usable modelling language for users of these and other methods.

This language pushes the envelope of what can be done with existing methods. As an example, the UML authors targeted the modelling of concurrent, distributed systems to assure the UML adequately addresses these domains.

UML focuses on a standard modelling language, not a standard process. Although UML must be applied in the context of a process, it is our experience that different organizations and problem domains require different processes. (For example, the development process for shrink-wrapped

[3] Web page at: http://oaspub.epa.gov/edr/harmonize$.startup

software is an interesting one, but building shrink-wrapped software is vastly different from building hard-real-time avionics systems upon which lives depend.) Therefore, the efforts concentrated first on a common metamodel (which unifies semantics) and second on a common notation (which provides a human rendering of these semantics). The UML authors promote a development process that is use-case driven, architecture centric, and iterative and incremental.

The UML specifies a modelling language that incorporates the object-oriented community's consensus on core modelling concepts. It allows deviations to be expressed in terms of its extension mechanisms.

GML

The OpenGIS® Geography Markup Language Encoding Specification[4] is an XML encoding for the modelling, transport and storage of geographic information including the spatial and non-spatial properties of geographic features. The specification defines the XML Schema syntax, mechanisms, and conventions that:

- Provide an open, vendor-neutral framework for the definition of geospatial application schemas and objects.
- Allow profiles that support proper subsets of GML framework descriptive capabilities.
- Support the description of geospatial application schemas for specialized domains and information communities.
- Enable the creation and maintenance of linked geographic application schemas and datasets.
- Support the storage and transport of application schemas and data sets.
- Increase the ability of organizations to share geographic application schemas and the information they describe.

The introduction of the Geography Markup Language (GML) has opened new opportunities for creating integrated cross-border spatial data services. As an XML application GML facilitates data integration by providing open, vendor-independent syntax for encoding spatial data. However, individual GML Application Schemas are inherently dependent on the local data content and thus vary from country to country. Those differences are, for example, reflected in the corresponding OpenGIS® Web Feature Service (WFS) queries.

Consequently, a cross-border data service must support two-way transformations: those applied to incoming queries and those carried out on the delivered dataset. As the traffic in both directions is encoded in XML syntax, the same techniques can be employed in both tasks.

3.2 The INSPIRE Program

INSPIRE defines which information should be made available against which restrictions. In addition it defines the method by which information shall be made available, or which standards shall be used.

3.2.1 INSPIRE Transposition Phase

On 21 November 2006, the European Parliament and the Council reached agreement in conciliation on the final text of the Directive. The Directive has been formally approved by the Council and by the European Parliament on 29 January and 12 February 2007, respectively. INSPIRE was published as

[4] Refer to web page: http://www.opengeospatial.org/standards/gml

Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 in the Official Journal of the European Union on 25 April 2007, and entered into force on 15 May 2007 (reference [3]).

The development and implementation of INSPIRE follows a programme of work consisting of three phases: the Preparatory (2005-2006), the Transposition (2007-2009) and the Implementation (2009-2019) phases.

We have entered in the transposition phase on 15 May 2007. At the beginning of this phase, the Commission have put in place the organisation necessary to assure coordination at Community level and has been assisted for that purpose by relevant organisations and appropriate structures and mechanisms will be installed by the Member States.

Concerning spatial datasets, specific Implementing Rules shall be adopted for interoperability and harmonisation (where practical) within two years after the entry into force for data sets related to data themes specified in Annex I.

Requirements for harmonised data specifications:

- for Annex I, II, III:
  - definition and classification of spatial objects,
  - geo-referencing;
- for Annex I, II:
  - common system of unique identifiers for spatial objects;
  - relationship between spatial objects;
  - key attributes and corresponding multilingual thesauri;
  - how to exchange the temporal dimension of the data;
  - how to exchange updates of the data.

The interoperability issue faced during the transposition phase

The process that will generate the harmonised data specifications has been defined in order to maximize the re-use of existing requirements and specifications and minimize the burden for Member States' organisations at the time of implementation.

For this reason the process of developing Implementing Rules for interoperability of spatial datasets will be a complex one, involving a large number of stakeholders, with whom many interactions and consultations will take place.

The development of INSPIRE Implementing Rules for harmonised spatial data follow a two-step approach:

1. Development of a conceptual framework and specification methodology.

   This work has started in the Preparatory Phase and has been concluded lately, with deliverables [4] and [5].
2. Development of data specifications for each data theme listed in the annexes of the Directive.

   This work will be carried out according to the results of the previous step.

The second step comprises the data specification development of the data themes.

From the INSPIRE portal

The European Commission published in the INSPIRE portal[5] the Data Specifications and calls for comments on the Definition of Annex Themes and Scope D2.3.

The next paragraphs introduce the content of the INSPIRE documents D2.5, D2.6 and D2.7.

3.2.2 D2.5 Generic Conceptual Model

Document [4] addresses the definition of a Generic Conceptual Model for spatial objects across the data themes. It is based on the ISO 19100 series and is structured according to the identified components of data harmonisation. The conceptual model specifies the generic aspects of geometry, topology, time, thematic information, identifiers and relationships between spatial objects. The GCM described in [4] will be updated based on comments by SDICs and LMOs.

Document [4] does not provide a methodology and process for developing harmonised data specifications for INSPIRE and does not specify how data is to be encoded. It only provides a framework for the teams that will eventually develop the INSPIRE data specifications that will become part of the INSPIRE Implementing Rules by providing a common rule base.

To ensure that the spatial data infrastructures of the Member States are compatible and usable in a Community and transboundary context, the Directive requires that common Implementing Rules (IR) are adopted in a number of specific areas. These IRs are adopted as Commission Decisions, and are binding in their entirety.

In the ongoing work towards a generic conceptual model and a methodology for developing data specifications (deliverables D2.5 and D2.6 of the DT published in 2006) a general agreement on a road map for data harmonisation and the relevant components has been achieved. The 21 components/aspects are organised in three groups:

- INSPIRE Information Model (9 components, e.g. ISO 19100 profile)
- Operational components/registers (5 components, e.g. identifier management)
- Guidelines / best practice (7 components, e.g. consistency between data).

[5] More information available at: http://www.ec-gis.org/inspire/home.html

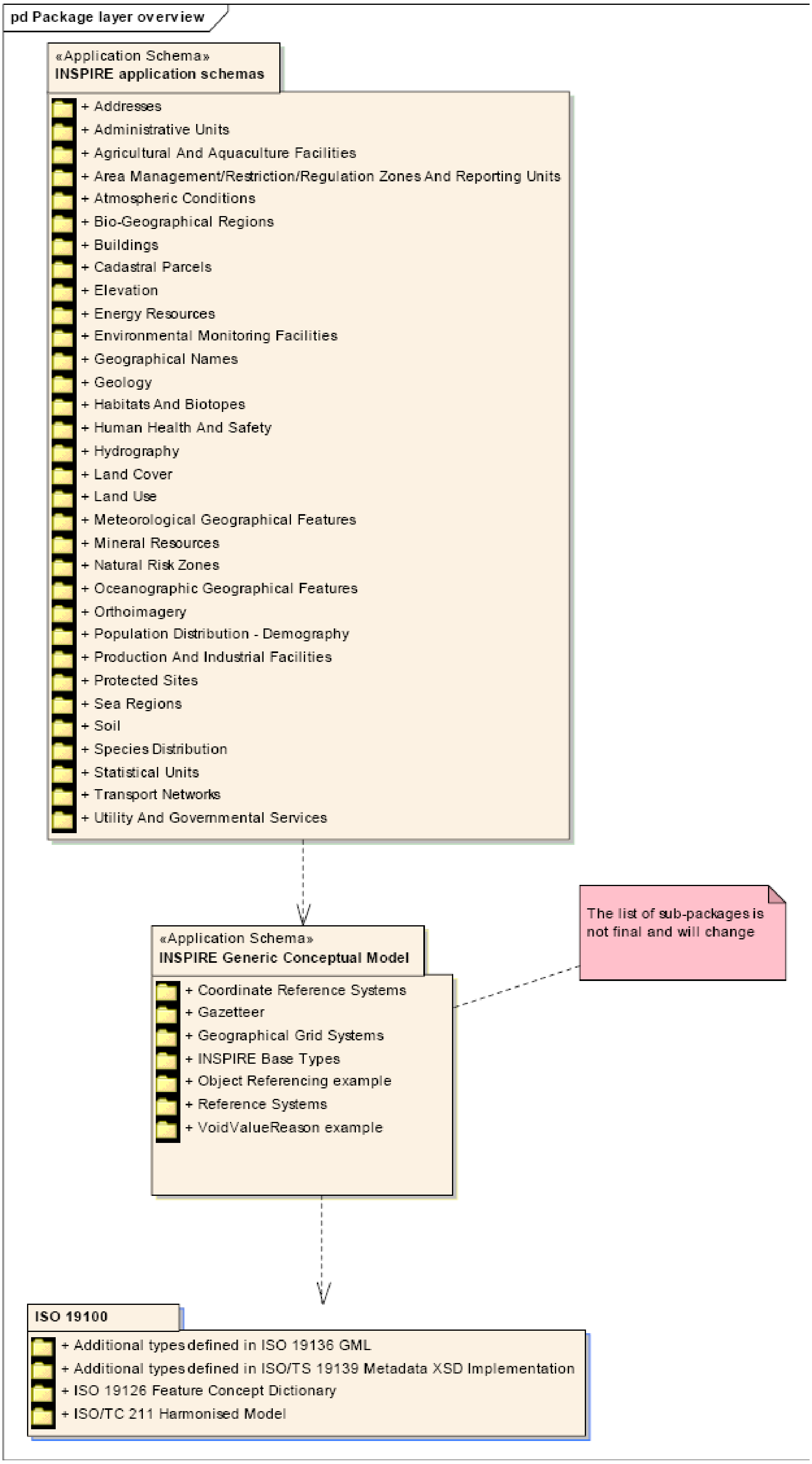

pd Package layer overview
«Application Schema»
INSPIRE application schemas
+ Addresses
+ Administrative Units
+ Agricultural And Aquaculture Facilities
+ Area Management/Restriction/Regulation Zones And Reporting Units
+ Atmospheric Conditions
+ Bio-Geographical Regions
+ Buildings
+ Cadastral Parcels
+ Elevation
+ Energy Resources
+ Environmental Monitoring Facilities
+ Geographical Names
+ Geology
+ Habitats And Biotopes
+ Human Health And Safety
+ Hydrography
+ Land Cover
+ Land Use
+ Meteorological Geographical Features
+ Mineral Resources
+ Natural Risk Zones
+ Oceanographic Geographical Features
+ Orthoimagery
+ Population Distribution - Demography
+ Production And Industrial Facilities
+ Protected Sites
+ Sea Regions
+ Soil
+ Species Distribution
+ Statistical Units
+ Transport Networks
+ Utility And Governmental Services
«Application Schema»
INSPIRE Generic Conceptual Model
+ Coordinate Reference Systems
+ Gazetteer
+ Geographical Grid Systems
+ INSPIRE Base Types
+ Object Referencing example
+ Reference Systems
+ VoidValueReason example
The list of sub-packages is not final and will change
ISO 19100
+ Additional types defined in ISO 19136 GML
+ Additional types defined in ISO/TS 19139 Metadata XSD Implementation
+ ISO 19126 Feature Concept Dictionary
+ ISO/TC 211 Harmonised Model

Figure 3.1: Schemas relationships

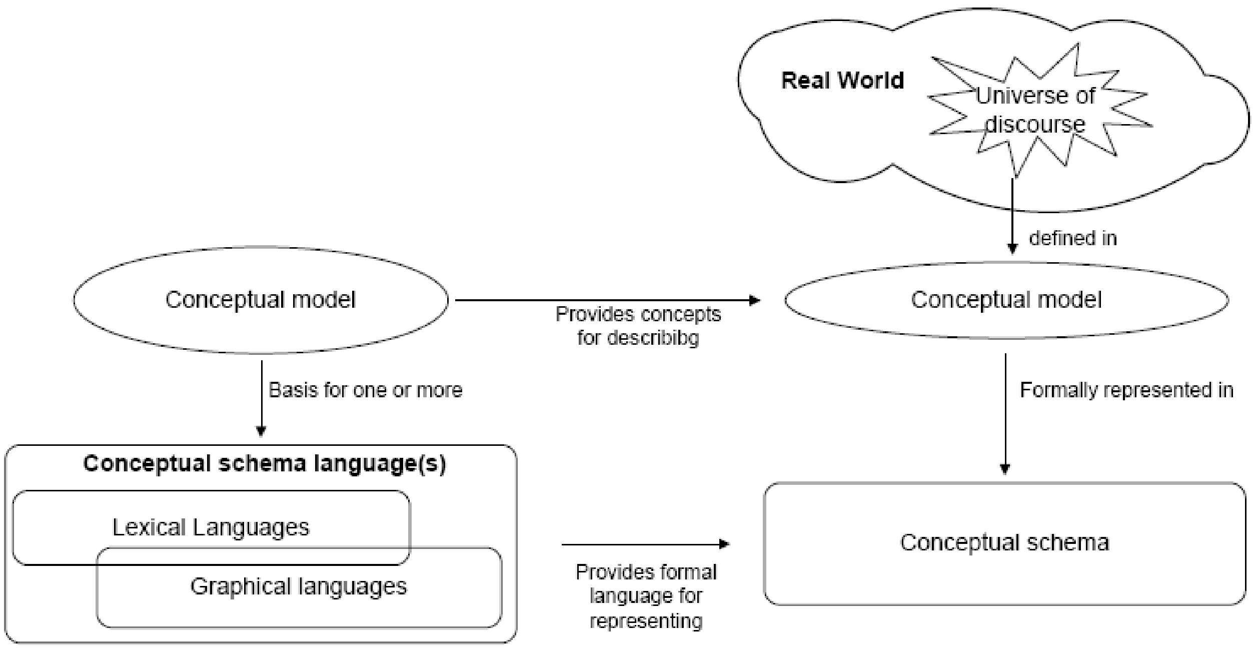


Figure 3.2: ISO 19101 Mapping of Conceptual Schema Into Real World

3.2.3 D2.6 Methodology for the development of data specifications

Document [5] specifies how individual INSPIRE spatial data themes have to be modelled based on the user requirements, the INSPIRE Generic Conceptual Model (reference [4]) and the relevant international and industrial standards.

The document describes the data state of play and the approach methodology required to define a common background for interoperability.

It is therefore not a draft IR, but a document that, together with user requirements and the other documents mentioned above, is targeted to help in the process of developing the IR for data specifications.

The result of applying DS-D2.6 will be the data product specifications, prepared according to ISO 19131:2007 *Geographic information - Data product specifications*, for the individual themes, i.e. conceptual information models that describe the relevant classes, their attributes, relationships and constraints, as well as other information as appropriate like data capturing information or data quality requirements.

The proposed methodology is based on results produced by the projects RISE[6] and MOTIIVE.

The INSPIRE document also underline the meaning of the “harmonisation” issue, adopted by the development process, i.e., a way to provide “*access to data through network services in a representation that allows for combining it with other INSPIRE data in a coherent way. This includes agreements about the different harmonisation components, described in details in generic Conceptual Model. In other words, by enabling interoperability, data can be used as if it had been harmonised.*”

In the present days the member states process data almost in a common way that [5] calls a “stovepipe” well outlined in the Figure 3.4 and so described: each member state uses input data according to different, often undocumented or ill-documented data specifications and uses different

[6] For RISE methodology, navigate to ftp://ftp.cordis.europa.eu/pub/ist/docs/environment/rise_en.pdf

methods to process the input data to produce more or less similar information relevant for policies within the Community.

The aim of a methodology that wants to define data specifications is therefore the understanding of user requirements in order to develop harmonised data specifications for the input data: all input data from the different member states must follows the same data specifications. Figure 3.5 describes this future scenario.

Input data itself and the related maintenance procedures will not change much. The great difference will be given by the network services of the member states that will provide data following the harmonised data specifications.

The proposed methodology adopt a “predictable and repeatable spiral development process model” in that new user requirements from a new application will be used to amend the existing data specification and incrementally growing INSPIRE’s degree of definition and implementation.

Since INSPIRE will be built based on existing spatial data, the existing data sets in the member states will be an important factor in the scoping of the INSPIRE Data Specifications in addition to the requirements from environmental policies.

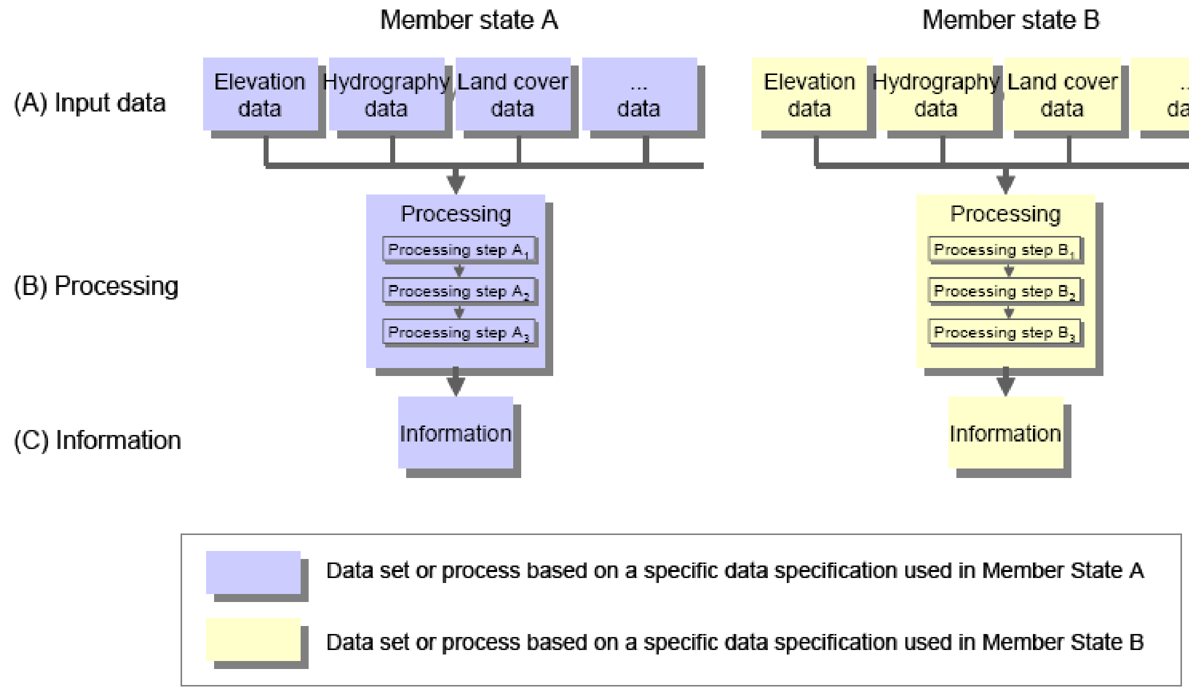


Figure 3.3: Data stovepipes (From [5])

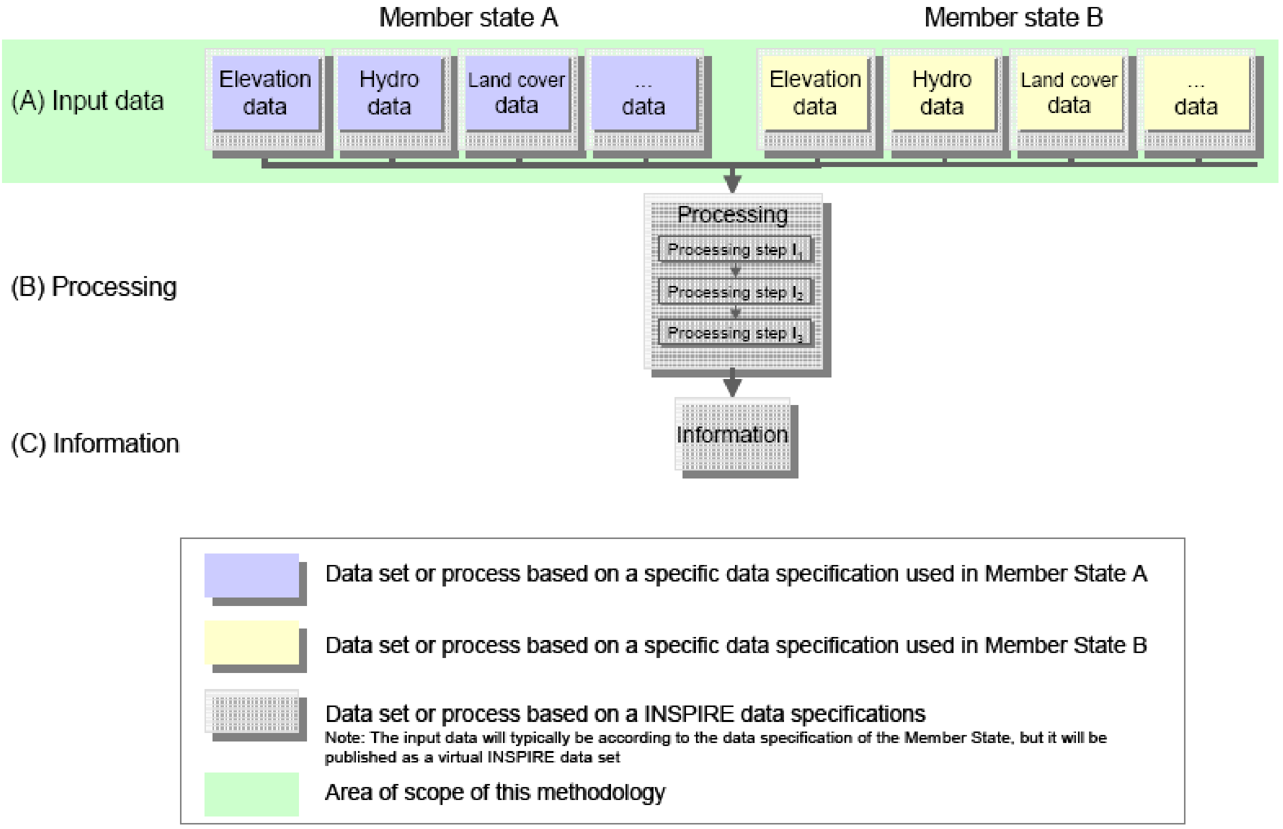

Member state A
Member state B
(A) Input data
Elevation data
Hydro data
Land cover data
... data
Elevation data
Hydro data
Land cover data
... data
(B) Processing
Processing
Processing step I1
Processing step I2
Processing step I3
(C) Information
Information
Data set or process based on a specific data specification used in Member State A
Data set or process based on a specific data specification used in Member State B
Data set or process based on a INSPIRE data specifications
Note: The input data will typically be according to the data specification of the Member State, but it will be published as a virtual INSPIRE data set
Area of scope of this methodology

Figure 3.4 – Harmonised data

Processing

In order to harmonise data and ensure interoperability, input data from the Member States must be processed and transformed according to the schema based on the conceptual model.

This UML model must be therefore mapped to a GML application schema before the user can consume the available information. The mapping will be achieved in (at least) three steps, since the platform independent UML model must be first converted into a UML platform specific model. Afterward it will be possible to adapt the intermediate UML model to the final GML application schema.

Publishing harmonised data over the web will be then possible with two different approaches:

- on-the-fly transformation;
- pre-conversion to a derived data set and subsequent publication.

Even if data transformation so far described does not directly imply consistency, the user requirements will often require ETL processes that will ensure changing data in order to create consistency with other data. It should be stressed that INSPIRE does not specify any service interfaces that support updating spatial data. I.e., access to INSPIRE data will be read-only.

But the need for supporting services helping in such transformation processes (think for example to Coordinate Transformation Services), should in principle be identified as part of the harmonisation process.

Testing

The test of the harmonised spatial data requires the definition and adoption of relevant use cases that will be thereafter executed with real data and also variations as well. Results will be then used to improve the data specification, following the spiral modelling previously described.

According to [5] a test will be hold as successful when:

- the spatial data from the member states will all conform to the draft INSPIRE data specifications;
- the application will access the data from the various download services;
- the application will be able to use the data from the various member states as if it would be from a single data set;
- the application will be able to perform all necessary actions to execute the use case.

3.2.4 D2.7 Guidelines for the encoding of the spatial data

This document will be released in the last months of 2007 and the content can be summarised in the following key issues:

- It specifies guidelines for the encoding of spatial data for the purpose of exchange between two systems
- Data encoding of spatial objects will be specified as part of the data specifications
- Data exchange is understood primarily as "access to data via services" which includes but is not limited to a download of a whole dataset → closely related to the Download service (Review by Consolidation Team and Drafting Teams should have started in August 2007)
- General approach based on the recommendations of CEN TR 15449 on encoding:
    - ISO 19136 (GML) is promoted as the ESDI encoding method when transferring spatial objects
    - ISO/TS 19139 is promoted as the ESDI encoding method when transferring information related to spatial data such as metadata, feature catalogues and dictionaries
    - The encoding of spatial objects will in general be model-driven, i.e. determined by the application schema
    - Coverage data will in general use existing encodings for the range part of the coverage function

From [4]:

Models: UML with reference to Enterprise Architect from Sparx Systems (http://www.sparxsystems.com/products/ea.html).

From [3]

A vital part of a data specification is the ISO 19109 application schema specified in UML. A UML application schema is a conceptual schema for data required by one or more applications, modelled in UML. This application schema describes the conceptual model for the data that is supposed to fulfil the identified user requirements.

3.3 Model Driven Architecture[7]

The Model Driven Architecture is a software design approach launched by OMG[8] in 2001. MDA provides a set of guidelines for structuring specifications expressed as models.

Quoting from [8][9], the main goal of MDA is to separate design from architecture and to generate required code starting from an abstract, human-described specification. A significant example of the application of these models is the Eclipse Foundation and specifically the Eclipse Modelling Project, where users can find different implementations of the OMG modelling standards.

The MDA approach defines a system functionality using a platform-independent model (PIM) with an appropriate domain-specific language. Then, given a platform definition model (PDM) corresponding to CORBA, .NET, the Web, etc., the PIM is translated to one or more platform-specific models (PSMs) that computers can run. The PSM may use different Domain Specific Languages, or a General Purpose Language like Java, C#, Python, etc.

The MDA model is related to multiple standards, including UML, MOF, XMI, EDOC, SPEM and CWM. The term “architecture” in MDAs does not refer to the architecture of the system being modelled, but rather to the architecture of the various standards and model forms that serve as the technology basis for MDA.

Of particular importance to model-driven architecture is the notion of model transformation. QVT is a specific standard language for model transformation defined by OMG. A model transformation is a set of operations required to convert a model M1 that conform to a given metamodel MM1 into a new model M2 conforming to different metamodel MM2. OMG issued a request for proposal by on MOF QVT to set up a standard compatible with the MDA recommendation suite (UML, MOF, OCL, etc.). The proposal requires that the abstract syntax of QVT should conform to a MOF 2.0 metamodel.

The MOF architecture has been a follow-up of UML at the time that OMG looked for a Metamodel architecture to define the UML. Quoting from Wikipedia, “MOF is designed as a four-layered architecture (see Figure 3.7). It provides a meta-meta model at the top layer, called the M3 layer. This M3-model is the language used by MOF to build metamodels, called M2-models.

The most prominent example of a Layer 2 MOF model is the UML metamodel, the model that describes the UML itself. These M2-models describe elements of the M1-layer, and thus M1-models;

[7] These information has been obtained mainly from Wikipedia (http://en.wikipedia.org)
[8] Main web page at http://www.omg.org/mda/.
[9] All the figures in this paragraph have been taken from [8]

these are the models written in UML. The last layer is the M0-layer or data layer and it's used to describe the real-world". In a few words, the profile feature of UML 2.0 can extend this language to cover virtually any domain by allowing the customisation of any MOF defined metamodel.

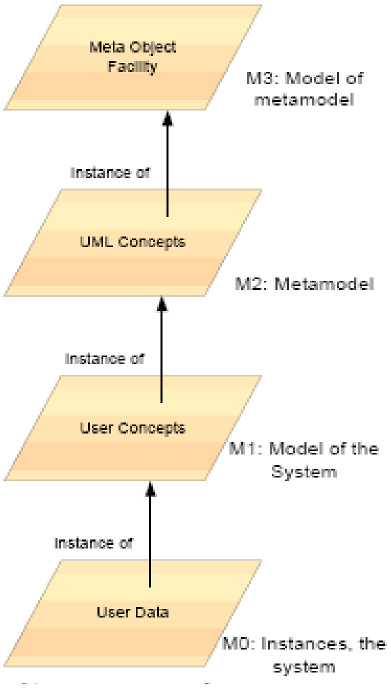


Figure 3.7 – MOF Architecture

The MDA approach makes use of different tools in order to get the final result (i.e. the code). These tools can be grouped in categories according to their ability to develop, interpret, compare, align, measure, verify, transform, etc. models or metamodels. While the initial models are created manually by human agents, derived models are created automatically by the formerly named programs. Models are both the required input and the generated output for all these programs.

A range of MDA tools are being developed by vendors and open source projects. Power RAD is such an approach developed by Outline Systems Inc. Microsoft is proposing the DSL tools approach which is a similar approach, not based on OMG standards. The Eclipse top level modelling project is currently developing a set of open source tools of various profiles (EMF, GMF, GMT, etc.). Another open source project called AndroMDA provides an extensible framework for generating code using virtually any technology/platform (e.g., .NET, Java, etc.) and is meant to be used repeatedly as part of the build process (i.e., instead of just generating starter code once at the beginning of a project).

MDA Tools are a superset of the UML tools, since their scope is much broader. This distinction can be made more general by distinguishing "variable metamodel tools" and "fixed metamodel tools". A UML CASE tool is typically a "fixed metamodel tool" since it has been hard-wired to work only with a given version of the UML metamodel (e.g. UML 2.1). On the contrary, other tools have internal generic capabilities allowing them to adapt to arbitrary metamodels or to a particular kind of metamodels.

In Figure 3.8 there is a typical MDA lifecycle model

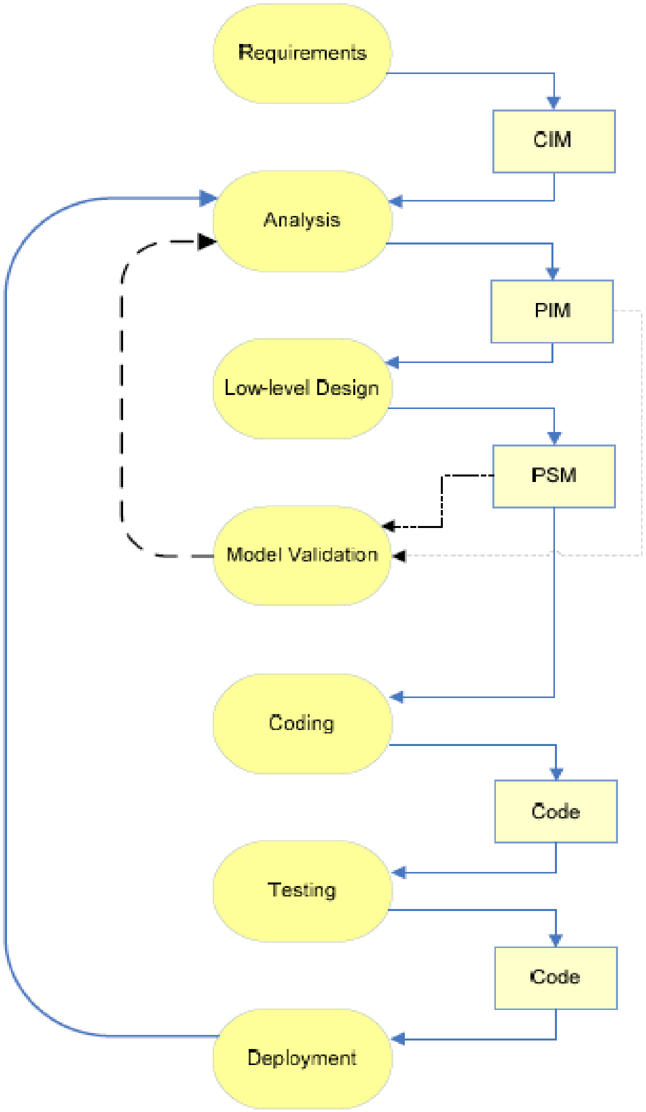


Figure 3.8 – MDA Lifecycle Model

The requirements that are in text form are modelled as CIM models. The CIM models give input to the analysis phase, where the PIM models are then developed. The low-level design is done by transforming the PIM models into PSM models with a transformation tool (PIM models give input to the low-level design phase, with that input PSM can then be created by the use of transformations).

The PIM and PSM models can be validated before they are transformed but that is not a mandatory requirement. If the PSM model is validated, iterations are performed back to the analysis level to update the PIM models (the dotted line illustrates those events). When the PSM models are completed they can be transformed into code. After the code is modified, testing and deployment is performed. If necessary, an additional iteration is made back to the analysis level.

In the next figure there is the schematisation of models and metamodels with the related languages. Models are described by the UML language, the UML language is defined by UML metamodels, which are expressed in the metalanguage MOF.

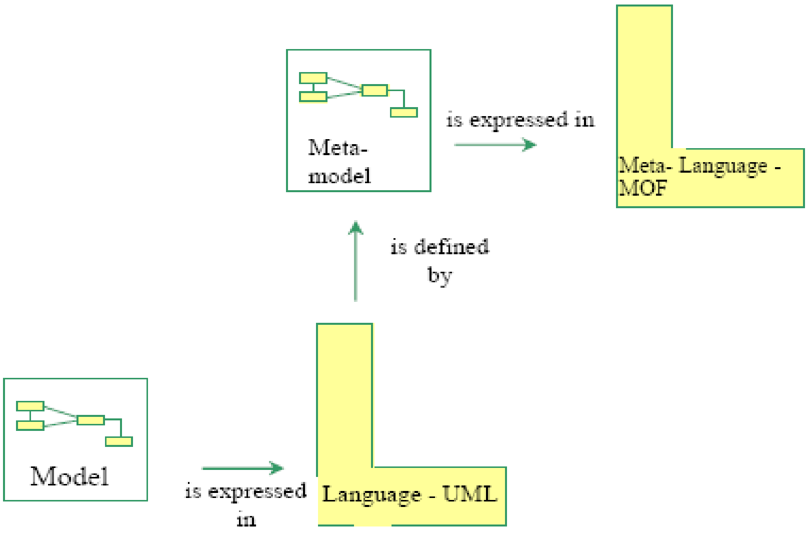


Figure 3.9 – Models, Metamodels, Languages and Metalanguages

The metamodel is of a higher level of abstraction than the modelling language. It describes the abstract syntax of a modelling language.

The metalanguage used in MDA is Meta Object Facility (MOF). The UML metamodel defines the language to be used when creating models. A metamodel is also a model and must therefore be defined by a language; in the case of MDA, MOF is used to define metamodels [20] as presented in Figure 3.9.

The aim of MDA is to decrease the cost of development and increase the quality of the product. Cost is affected as time to market is reduced with the MDA process.

3.4 Web Ontology Language

OWL is a language for defining and instantiating Web ontologies and it may include descriptions of classes, along with their related properties and instances. The definition given by W3C in [9] states that OWL is designed for use by applications that need to process the content of information instead of just presenting information to humans. It has more facilities for expressing meaning and semantics, allowing for greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S)[10] You can refer to the next sub-paragraph for a description of RDF. OWL accomplishes all this by providing additional vocabulary along with a formal semantics.

OWL is based on earlier languages OIL[11] and DAML+OIL[12]. In 2001 the World Wide Web Consortium created the "Web Ontology Working Group" in order to draft the abstract syntax and in 2004 OWL became a formal W3C recommendation[13].

OWL is seen as a major technology for the future implementation of a Semantic Web. It is playing an important role in an increasing number and range of applications, and is the focus of research into tools, reasoning techniques, formal foundations and language extensions.

The Semantic Web is a vision for the future of the Web, in which information is given explicit meaning, making it easier for machines to automatically process and integrate information available on the Web. Within this perspective the semantic web can be the right answer to the needs of interoperability and data harmonisation strongly required by metadata, services and, above all, data technical chapters of the INSPIRE programme.

The Semantic Web can benefit of XML's ability to define customized tagging schemes and RDF's flexible approach to representing data. The first level above RDF required for the Semantic Web is an ontology language what can formally describe the meaning of terminology used in Web documents. If machines are expected to perform useful reasoning tasks on these documents, the language must go beyond the basic semantics of RDF Schema. The OWL Use Cases and Requirements Document provides more details on ontologies, motivates the need for a Web Ontology Language in terms of six use cases, and formulates design goals, requirements and objectives for OWL.

OWL has been designed to meet this need for a Web Ontology Language, overcoming the limitations of XML and RDF outlined in the following list:

- XML provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents.
- XML Schema is a language for restricting the structure of XML documents and also extends XML with datatypes.
- RDF is a datamodel for objects ("resources") and relations between them, provides a simple semantics for this datamodel, and these datamodels can be represented in XML syntax.
- RDF Schema is a vocabulary for describing properties and classes of RDF resources, with a semantics for generalization-hierarchies of such properties and classes.

OWL adds more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjointness), cardinality (e.g. "exactly one"), equality, richer typing of properties, characteristics of properties (e.g. symmetry) and enumerated classes and uses the RDF mechanisms for data values.

[10] References at http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

[11] More information can be found in http://www.cs.man.ac.uk/~horrocks/Publications/download/2001/IEEE-IS01.pdf.

[12] See http://www.daml.org/2001/03/daml+oil-index for any reference on this language.

[13] Refer to http://www.w3.org/TR/owl-features/.

OWL has three increasingly-expressive sublanguages: OWL Lite, OWL DL, and OWL Full. OWL DL and OWL Full contain the same features of OWL-Lite, but OWL Full is more flexible about how these features can be combined.

OWL Lite uses only some of the OWL language features and has more limitations on the use of the features than the other two sublanguages. For example, classes can only be defined in terms of named superclasses (superclasses cannot be arbitrary expressions), and only certain kinds of class restrictions can be used. Equivalence between classes and subclass relationships between classes are also only allowed between named classes, and not between arbitrary class expressions. OWL Lite also has a limited notion of cardinality - the only cardinalities allowed to be explicitly stated are 0 or 1.
OWL Lite supports notions of ontology inclusion and relationships and attaching information to ontologies. Both OWL DL and OWL Full use the same vocabulary although OWL DL is subject to some restrictions.

3.4.1 The Resource Description Framework

This sub-paragraph gives a short summary of the RDF datamodel and schema.

RDF is a framework for representing information in the Web. The framework is designed so that vocabularies can be layered. The RDF and RDF vocabulary definition (RDF schema) languages are the first such vocabularies.

RDF has an abstract syntax that reflects a simple graph-based data model, and formal semantics with a rigorously defined notion of entailment providing a basis for well founded deductions in RDF data.

RDF is designed to represent information in a minimally constraining, flexible way. It can be used in isolated applications, where individually designed formats might be more direct and easily understood, but RDF's generality offers greater value from sharing. The value of information thus increases as it becomes accessible to more applications across the entire Internet.

RDF has been designed in order to offer the following features:

- having a simple data model
- having formal semantics and provable inference
- using an extensible URI-based vocabulary
- using an XML-based syntax
- supporting use of XML schema datatypes
- allowing anyone to make statements about any resource

3.5 Geospatial Interoperability Model (GIRM)

Geospatial interoperability is based on shared agreements governing essential geospatial concepts and their embodiment in communication protocols, software interfaces, and data formats.

It is already clear at this level of definition that geospatial interoperability is a more specific context with respect to the general approaches described in the previous paragraphs.

The Geospatial Interoperability Reference Model references standards and specifications needed for interoperability: it describes and explains them within a structured model of geospatial processing, as they apply to the design of geospatial software and services […]. The intent is to guide the scope and growth of geospatial applications and interoperability; and to detail how any geospatial software can plug into a larger infrastructure to draw on many different sources of data and services -- or to support a wide, diverse user audience[14].

This Reference Model is focused on mechanisms for effective cooperation between geoprocessing software components. Effective use of geospatial information in a given context may also require policies such as human interface guidelines, data content or portrayal requirements, or conventions for data storage or georeferencing. Such policies, which include several important content standards, are outside the scope of this Reference Model.

GIRM is intended not as a formal definition of standards to be implemented, but rather as a consultative tool to help decision makers define what standards apply to a given set of activities, technologies, or organizations, to facilitate interoperable geoprocessing.

3.5.1 Levels of abstraction

The Reference Model brings together standards at two different levels of abstraction, and under two different architectural viewpoints:

- Implementation specifications tell software developers how to express information or requests within a particular distributed computing environment (e.g., World Wide Web, CORBA, .NET). Implementation specifications generally include access protocols, object models, and naming conventions. Such specifications are specific to, and directly usable within, their target computing environment.
- Abstract models specify what information or requests are valid in principle, irrespective of individual computing environments. They define essential concepts, vocabulary, and structure (type hierarchy) of geospatial services and information transfer. These models set the stage for creating implementable specifications, and for extending existing ones to new environments.

Which of these to apply depends on the design lifecycle, and on the intended computing environment. Earlier design stages often draw on Abstract Models to sketch a system concept; whereas later implementation stages follow Implementation Specifications in detail. When it comes to writing software, if a suitable Implementation Specification already exists for the applicable computing environment, it should be the standard of choice. Otherwise, the relevant Abstract Model(s) should guide the design of a new Implementation Specification for that environment.

At either the abstract or the implementation level, standards of two different kinds may apply:

- Service invocation: these standards define the interfaces that allow different systems to work together, or the expected behaviour of software systems. The ISO/IEC Reference Model for

[14] A detailed description of GIRM can be found at http://gai.fgdc.gov/girm/v1.2/

Open Distributed Processing (RM-ODP) calls this the computation viewpoint; its focus is on invoking services effectively and unambiguously.

- Information transfer: these standards define the content of geospatial information or its encoding for transfer between different processing systems. In RM-ODP parlance, this is the information viewpoint, emphasizing efficient, lossless communication.

The next picture displays the stack required to publish GIS data for user applications, considered the consumers of the spatial information.

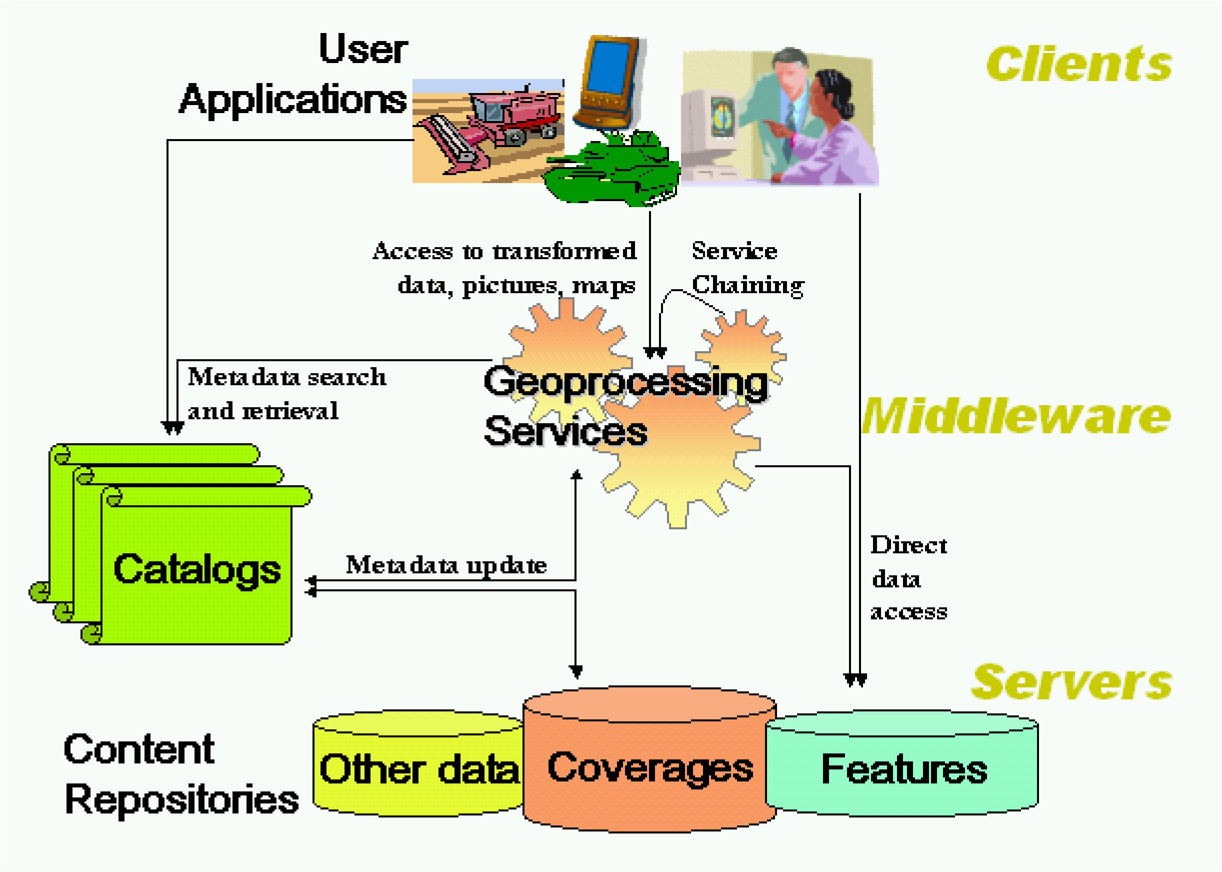


Figure 3.10 – GIRM Interoperability Stack

Components in this model are of four essential kinds:

1. User applications are the software usually seen by users; they may be highly customized analytical or field applications, or general-purpose viewers. They draw their input either directly from data repositories or from intermediate services that pre-process data for their use.
2. Catalogues allow clients and services to find out what repositories or services are available and appropriate for their use. Gazetteers are another such "meta-service"; they provide the geographic locations of place names.
3. Content repositories provide geospatial data in the form of features, coverages, and data objects or tables.
4. Geoprocessing services are the "workhorses" of the interoperability stack. They may simply draw maps from raw data; or they may perform advanced analytical functions such as feature

extraction or coordinate transformation. They provide data, maps, or other inputs to user applications – or to other services, in what's known as "service chaining."

3.5.2 Data and Data Access

Concerning geospatial data and access to this data, the model make a distinction between features and coverages, the first assumed as discrete data while the latter considered as measured values.

The abstract models applied to the two distinct data categories can be summarised in the following list:

- Features
  - Rules for application schema (ISO 19109:2005) contain the general feature model for ISO TC211. It guides the use of classes, relationships, interfaces, and properties in designing feature schemas for data transfers or transactions.
  - Feature cataloguing methodology (ISO 19110:2005) provides a basis for describing feature types to be pooled across a community of users.
  - Spatial Schema (ISO 19107:2003) provides a model of 2-dimensional and 3-dimensional geometry and topology, and related operators such as "buffer" or "intersects." OGC has adopted this model as Topic 1 (Feature Geometry) of its Abstract Specification.
  - Simple Features Common Architecture (ISO 19125-1:2004) provides further detail on the subset of features described in OGC's Simple Feature Access Implementation Specifications, including well-known encodings and a starter set of Spatial Reference Systems.
- Coverages
  - OGC and ISO share a general definition of coverages: ISO’s Schema for Coverage Geometry and Functions (ISO 19123:2005) defines the various types of coverages and their access functions. OGC's Abstract Specification, Topic 6 (The Coverage Type and its Subtypes) incorporates and extends ISO 19123.

In the last section of the web reference page we can find a list of standards already available on the market. These standards are publicly available thus allowing anyone to write software to encode or decode data in these formats. It must be underlined anyway that these standards are not open, in that they are not defined or maintained by a voluntary consensus process.

The most common, to mention some of them, are ESRI's Shapefile, MapInfo's MIF/MID format or AutoDesk's Drawing eXchange Format (DXF).

3.6 NERC DataGrid Project

The Natural Environment Research Council has a wide range of data holdings, held in technologies from flat files to relational databases. These holdings are relevant to a wide range of scientific disciplines, despite often having been collected on behalf of quite narrow specialised disciplines. The data holdings are stored across a wide range of archives, ranging from specialist professional data curators and archivists, such as the British Atmospheric Data Centre (BADC) and the British Oceanographic Data Centre (BODC) to files held on the hard disc of an individual scientist's PC.

The NERC DataGrid project intended to make these data resources available to user as one entity, so improving their ability to find and use data, with as seamless a transition as possible between these phases. In the pilot project, Atmospheric and Oceanographic data held in the BODC and BADC has been made available, with data from other disciplines funded by NERC being added in due course.

This was done by providing a decoupled metadata infrastructure using, where possible, tools, standards, and mechanisms that already exist or are under development within e-Science and the worldwide earth science communities.

NDG provides the infrastructure which addresses the following problems usually common to data interoperability:

- Discovery, i.e.: who has the right information. This is the realm of metadata.
- Access, i.e.: how to get it. This is the realm of services.
- Use, i.e.: how to make a use of it. This is the realm of data.

Like the web, the NDG has no “owner” or “central control”; data remain with data providers - be they managed data centres in the UK or abroad, or semi-managed data archives in large research groups. The location of the data can be transparent to the user, while still allowing data providers to maintain their intellectual investment by controlling access.

NDG does not generate metadata but provides a framework for it. Moreover, it does not provide information services rather data services. The architecture of the NDG is based on independent interacting web services. In particular it provides two community services which all NDG participants may avail themselves of:

- The “NDG Discovery Service”: a database of discovery information, harvested from data providers around the world, together with a portal web-site and a set of web-services, for exploiting that database.
- The “NDG Vocabulary Service”: a database of environmental thesauri and ontology tools to map between terms[15].

The data model Data types encompass both observational and model data. An initial abstract (weakly typed) application schema has been obtained from the data model first applied (September 2002) across the range of data maintained by both the BADC and BODC in the context of the pilot project.

BODC developed a vocabulary management system for the NERC DataGrid project based on an Oracle back-end fronted by a Web Service API incorporating automated version management and preservation of historic list versions. This has been adopted for SeaDataNet to cover the technical governance requirement.

[15] Available at the address: http://www.bodc.ac.uk/products/web_services/vocab/

The initial data model incorporates two central elements that apply across almost all environmental data:

1. logical data structure (or 'shape')
2. data location in time and space.

At the root level, a named dataset may contain both a number of parameters, and other datasets. A parameter is characterised by its name, physical units, and a flag indicating missing (or bad) data. A standard name from a controlled vocabulary may provide additional parameter type semantics (this includes the namespace authority, for example 'BODC data dictionary' or 'CF convention').

NDG is based around mature OGC descriptions of environmental data. The model is based on the ISO TC211 standards and refers to the Domain reference Model ISO 19101 and to the 19100 series. Data modelling use the conceptual modelling (ISO 19103) to define the application schema (ISO 19109). This schema is thereafter used to map an encoded dataset (ISO 19118) to the feature type catalog (ISO 19110).

NDG makes use of the MOLE schema, an XML schema that provides a common and standardized way of describing key aspects of datasets:

- Activities which generate data;
- Observation Stations at which the data are collected (or produced in the case of simulations);
- Data Production Tools;
- Data Entities themselves.

A second important component used by NDG is the Climate Science Modelling Language information model (both a UML description and an XML schema which is an "Application Schema" of the "Geographic Markup Language"). CSML provides format independent descriptions of the parameters and organization of datasets according to the "Sampling Features" and "Observations and Measurements" frameworks of the Open Geospatial Consortium (OGC).

Other components of this project include are the Discovery Infrastructure, Discovery Gateway[16] and Security Infrastructure, all documented at the NDG web site[17].

Note that a number of working links have been established between NDG and significant related projects. These include the US Earth System Grid (named collaborators on the NDG proposal), the EU MarineXML and forthcoming MOTIIVE (marine data harmonisation for GMES/INSPIRE) projects, and geospatial standards adoption projects of the World Meteorological Organisation (through the project lead, UK Metoffice) and the International Oceanographic Commission (through MarineXML and BODC).

[16] Available at the address: http://ndg.nerc.ac.uk/discovery/

[17] References at the web page http://ndg.badc.rl.ac.uk/

3.7 OGC standards

From *Standards-based data interoperability in the climate sciences*, available at:

http://journals.cambridge.org/download.php?file=%2FMAP%2FMAP12_01%2FS1350482705001556a.pdf&code=28fbc5f832132051d298f403f4d38dc7

The Open Geospatial Consortium17 (OGC) is an international consortium of over 300 industry, government, and academic institutions having the aim of developing open specifications for web-enabled interoperability of spatial data. OGC-conformant services are becoming widely deployed as standard solutions for exchange of geographic information. Specifications are developed through the oversight of various Working Groups and fast-track test bed exercises coordinated by the Interoperability Program. A considerable number of vendors, including, for example, ESRI and Cadcorp, are beginning to offer OGC compliant software.

There is considerable overlap of interests between OGC and ISO TC211. In practice, the two work closely together, principally through a Coordination Group established formally in a 1999 joint agreement. A number of OGC specifications are incorporated in the ISO TC211 program of work. These include the OGC's Geography Markup Language (ISO 19136) and Web Map Server interface (ISO 19128).

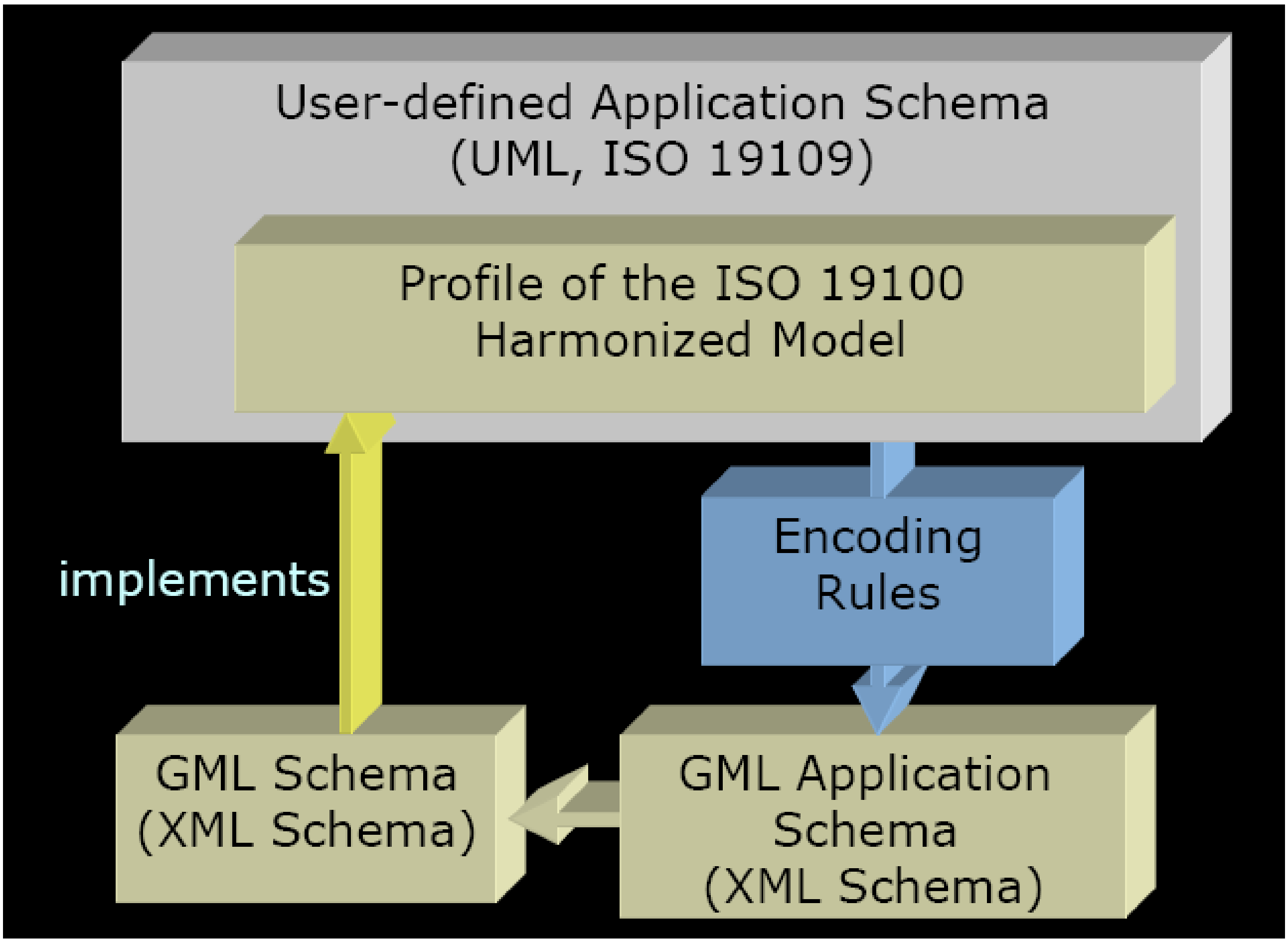
GML and WMS.

The Geography Markup Language (GML) is a reference XML schema encoding that implements the conceptual schemas of the ISO 19100 series (parts of ISO 19107, 19108, 19111 and ISO 19123). It will be compliant with the encoding rules of ISO 19118. In February 2004 GML 3.1 has been harmonised with ISO CD 19136.

GML provides a number of integrated XML schemas for feature types, geometry (0, 1, 2 and 3-dimensional primitives and aggregates), coordinate reference systems, topology, temporal information, dictionaries, units of measure, directions, observations and coverages. In shorts, it supports both spatial and non spatial properties of objects.

The language supports the definition of profiles, is open and vendor neutral. In addition it is also extensible. For all these reasons it is therefore likely that GML will continue to evolve as a Working Draft in the committee stage before release as an International Standard.

The GML Schema is horizontal and not focused on a specific application domain, but it provides common constructs and concepts which may be used by all the different application domains



The OGC Web Map Service Implementation Specification (WMS) defines a web accessible interface for requesting rendered maps of spatial data. The functionality is similar to that provided by the Live Access Server used in the climate sciences. A WMS request embodies the following parameters:

- required information to be rendered (one or more map 'layers')
- styling of layers
- a bounding box specifying a region of the Earth
- coordinate reference system or map projection to be used
- output image size, format, background colour and transparency

WMS servers can be 'cascaded' to aggregate maps from multiple sources, or to perform processing such as format conversions or coordinate transformations.

The Web Feature Service Implementation Specification (WFS) provides a means to obtain GML-encoded 'simple features' from a geo-database.

Transactional updates are also supported. Simple features are those that have geometry-valued properties limited to points, lines or polygons, or collections of these.

Semantic Transformation from AGILE International Conference

OWS allow for syntactic interoperability but do not allow for semantic interoperability:

- conceptual schemas of source systems hidden from target systems
- semantic transformation not supported yet

4. Requirements Synthesis

From[*http://www.ec-gis.org/Workshops/12ec-gis/presentations/Plenary%20room/INSPIRE%20I/portele.pdf*](http://www.ec-gis.org/Workshops/12ec-gis/presentations/Plenary%20room/INSPIRE%20I/portele.pdf)

We have to consider data harmonisation requirements on different levels:

- the schema level (use of common application schemas independent of the data model of the base data)
- the data level (e.g. edge matching in border areas)
- the information product level (e.g. integration on the level of raster maps hosted by Web Map Services or derived reports)

Rules for Application Schemas

- Based on ISO 19109 and ISO/TS 19103 –with some constraints on the GFM and the use of UML

ISO 19100 Profile

- Identification of a profile of ISO 19107/19108/19123/etc. not possible due to the wide range of spatial information in the themes
- Recommendation to use the Simple Feature profile of the spatial schema (ISO 19125-1) when possible
- Potential SDIC/LMO consultation: spatial object types and interpolation types that are in use

Conceptual model may be based on ISO/TS 19139

The

Enterprise Architect has been selected as the UML tool for DT work based on an analysis of the available tools.

From: [*www.sistemapiemonte.it/serviziositad/dwd/INSPIRE_Data_Spec.pdf*](www.sistemapiemonte.it/serviziositad/dwd/INSPIRE_Data_Spec.pdf)

- INSPIRE will not set off an extensive programme of new spatial data collection in the Member States. Instead, it is designed to optimise the scope for exploiting the data that are already available.... INSPIRE will pave the road for a progressive harmonisation of spatial data in the Member States.
- As regards harmonisation, INSPIRE will address only those aspects needed to achieve cross-level and cross-thematic consistency of spatial data and to make them available to support Community policies. For instance, INSPIRE does not require Member States to change the format of their spatial data holdings; instead, Member States can provide interfaces that transform heterogeneous data to a uniform model.

Proposal article 2

The Inspire program addresses identifiable collections of spatial data, hereinafter “spatial data sets”, which fulfil the following conditions:

- they are related to an area under the jurisdiction of a Member State or to its exclusive economic zone/search and rescue region, or equivalent;
- they are in electronic format;
- they relate to one or more of the themes listed in Annexes I, II or III

1. ”spatial data” means any data with a direct or indirect reference to a specific location or geographical area;
2. “spatial object” means an abstract representation of a real-world entity related to a specific location or geographical area;

Harmonisation of data will be progressive, and is not intended to trigger extensive spatial data collection efforts. The emphasis of the proposal is on achieving interoperability by leveraging existing data through transformation, rather than requiring new data collection efforts or wholesale re-engineering.

Key assumptions

- All nations & organisations (within a nation) start from different positions in terms of data models etc
- Due to different political, economic, cultural and organisational drivers, we will not achieve total harmonisation across every nation during the INSPIRE process
- We need to agree on a mechanism that provides a common language to support needs at EU and other large-scale cross-border and cross-sector levels
- It is clear that in the first step we should concentrate on a limited set of aspects rather than trying to solve every problem (in every country across every kind of dataset)
- “harmonisation” through interoperability in a service-based architecture rather then (full) harmonisation of the underlying data models
- We have to consider data harmonisation requirements on different levels:
- the schema level (use of common application schemas independent of the data model of the base data)
- the data level (e.g. edge matching in border areas)
- the information product level (e.g. integration on the level of raster maps hosted by Web Map Services or derived reports)

An INSPIRE Data Specification is based on a data product specification (DPS) according to ISO 19131. These DPS includes the following sections:

Mandatory specifications:

- Overview
- Specification scopes
- Data product identification
- Data content and structure
- Reference systems
- Data quality
- Data product delivery
- Metadata

Optional specifications:

- Data capture
- Data maintenance
- Portrayal
- Additional information
- Data harmonisation within the ESDI means that all countries use a common set of coordinate reference systems, data model, classification system, etc.
- Interoperability within the ESDI means that each country maintains their own infrastructure, but adopts a framework that enables existing datasets to be linked up from one country to another (e.g. via transformation or translation)

INSPIRE Requirements, according to [4]:

1 The definitions of the GCM (and therefore ISO 19100) shall be applied to the final application schemas.

2 Spatial datasets shall be structured according to the previously defined application schema.

3 New requirements (concepts) shall not conflict with existing ones in GCM and must not change directly the application schema. They must rather lead to a change proposal for the CGM.

4 Terms and definitions used in all the data specifications shall be drawn from the INSPIRE Glossary.

5 Data specifications must refer to the reference model specified in standard ISO 19101

6 INSPIRE application schemas shall conform to the General Feature Model as specified in ISO 19109 Clause 7.

7 The common concepts related to spatial object types, attribute types, association types and coded values shall be recorded in a dictionary as specified in ISO 19126[18] and maintained for INSPIRE in an ISO 19135 conformant register.

8 The spatial object types of an INSPIRE application schema shall be expressed in a feature catalogue as specified in ISO 19110 (now in Committee Draft stage).

9 Every feature catalogue shall contain the information for all spatial object types that are relevant for the particular application schema.

10 Each spatial object in a spatial data set shall be described in an application schema.

11 An application schema shall contain a complete and precise description of the semantic content of its spatial object types following the General Feature Model.

12 INSPIRE application schema included in every INSPIRE data specification shall be modelled according to ISO 19109 8.2.

13 The spatial object types and their properties specified in an application schema shall be drawn from the common feature concept dictionary.

14 Spatial object types shall be modelled according to ISO 19109 7.1-7.2, 8.1, 8.5-8.9 and according to the additional rules in Clauses 9, 10, 11, 13, and 22 of document [4].

15 The profile of the conceptual schema defined in the ISO 19100 series that is used in the application schema shall conform to ISO 19109 8.4.

Document [4] also holds recommendations that can be summarised as follows:

[18] Including the name, definition and description of all themes given by the Directive.

1 The validity of the GCM should be first tested on a subset of data specifications, i.e. before the specification of all the possible themes covered by the INSPIRE directive.

2 The implementation of the common feature concept dictionary register should wait that ISO 19126, currently in Committee Draft stage in the ISO standardisation process, should reach the Draft International Standard stage.

3 INSPIRE should comprise an ISO 19135 conformant register and registry for INSPIRE feature catalogues in order to support the ESDI.

Regarding requirement 5 it should be underlined that no explicit reference model has been yet defined, therefore the model described in ISO 19101 has been used as the reference model for the development of data specifications.

Regarding requirement 6, some elements of the Generic Conceptual Model may not have a direct representation on the implementation platforms relevant for the application schema and, therefore, the use of these elements will lead to additional complexity on the implementation level.

Regarding requirements 9, a feature catalogue is a repository for a set of definitions used to classify significant real-world phenomena. The catalogue provides a means for organising into categories the data that represent these phenomena, in order to make information as much comprehensible and useful as possible and to avoid ambiguity. According to this definition, the application schema may contain information not necessarily represented in the corresponding feature catalogue and vice versa.

Regarding requirements 13, the process of defining spatial object types is presently detailed in document [5] describing the methodology for developing data specification. In short, different actions take place

5. State of Play

Different products have been taken into account according to their usefulness for the INSPIRE technical chapter issues of data interoperability and harmonisation.

Since the major activity with the applicability of the INSPIRE requirements involves the validation of the data abstract model and/or the generated application schema against the standards, the look-up in the present software scenario has mainly considered the capabilities of existing software to perform analysis and cross check validation on the UML (XML) models.

Just to give an idea of the wide spectrum of software available on the web, a list of UML tools is provided below, grouped according to the involvement of a license, copyright or patent law (proprietary) or not. The list has been taken from Wikipedia[19].

Non-proprietary UML tools

- Acceleo: Eclipse and EMF template-based system for source-code generation from UML models.
- ArgoUML: a Java-based UML engineering tool, closely follows the UML standard, BSD license.
- Astade: a platform-independent UML-tool based on wxWidgets.
- ATL - a QVT-tool which can transform UML models into other models. Available from the Eclipse GMT project (Generative Modeling Tools).
- BOUML: multi-platform UML 2.0 toolbox, reverse/generates C++/Java/IDL. Very high performances (written in C++, on Qt). Licensed under the GNU GPL.
- Dia: a GTK+/GNOME diagramming tool that also supports UML (licensed under the GNU GPL)
- Eclipse: with Eclipse Modeling Framework (EMF) and UML 2.0 (meta model without GUI) projects.
- Fujaba: Acronym for "From UML to Java and back". Allows modeling behaviour using story diagrams.
- Gaphor: a GTK+/GNOME UML 2.0 modeling environment written in Python
- Kivio: part of the KOffice project
- MetricView Evolution: a tool for metrics-based quality-analysis and better comprehension of UML models
- MonoUML: based on the latest Mono, GTK+ and ExpertCoder.
- NetBeans: with NetBeans IDE 5.5 Enterprise Pack
- Omondo: Eclipse 3.2 plugin. Implements UML2.1, uses JDK 5.
- Papyrus: an open-source UML2 tool based on Eclipse and licensed under the EPL
- StarUML: a UML/MDA platform for Microsoft Windows, licensed under a modified version of GNU GPL, mostly written in Delphi
- Taylor: model-driven architecture "on rails" (licensed under the GNU LGPL)
- Topcased: open source model editors, transformation and formal verification tools, modelling languages (www.topcased.org)
- Umbrello UML Modeller: part of KDE
- UML Pad: a UML modeller written in C++/wxWidgets (licensed under the GNU GPL)

[19] Browse the web page http://en.wikipedia.org/wiki/List_of_UML_tools

- UML Pad (PalmOS): a UML tool for PalmOS
- UMLet: a Java-based UML tool (licensed under the GNU GPL)
- Use Case Maker: a use cases management tool (licensed under the GNU LGPL)
- Violet UML Editor: an easy-to-use Java-based UML Editor; fully integrated into Eclipse; licensed under the GNU GPL
- Xholon: an open-source tool that transforms, simulates and executes models developed using third-party UML 2.0 modelers

Proprietary UML tools

Potential users can freely download versions of most of the following tools; such versions usually impose limits in capability and/or by a time-period.

- AgileJ StructureViews: custom reverse-engineered class-diagrams — Java/Eclipse/XP. (Formerly marketed as "Modelistic".)
- Altova UModel: GUI UML editor, supports UML 2.1, able to export as XMI
- Apollo for Eclipse:supports UML 2.0 and Java 5. Integrates with the Eclipse IDE
- ARTiSAN Studio: supports UML 2.0 and SysML
- Blueprint Software Modeler: An integrated software-modeling environment with UML 2.1 modeling, OCL 2.0, meta-modeling and profiles; based on Eclipse
- Borland Together: UML modelling tool, integrated with Eclipse and with MS VS.NET 2005. Supports UML 2.0 and MDA, OCL, MOF
- Cadifra UML Editor: UML diagram editor for Windows
- ConceptDraw 7: diagramming tool for Windows and Mac, supports UML 2.0
- eRequirements: free web-based requirements-management tool
- GatherSpace: Online/On-Demand Use Case and Requirements Management
- Gliffy: Desktop application feel in a web-based UML diagramming solution
- JUDE: object-oriented analysis and design with UML and Mindmap. JUDE/Community, though free to use, does not provide open source
- Konesa: Canyon Blue's collaborative modelling tool
- MacA&D: UML and requirements-management for Mac OS X
- MagicDraw UML: UML 2.0 tool with forward- and reverse-engineering and support for many plugin products for MDA. Integrates with many IDEs, including Eclipse and NetBeans. Supports SysML
- MasterCraft (software): a suite of tools from Tata Consultancy Services Limited which support object-oriented analysis and design using UML for development of MDA-based application-software. The tool-suite consists of IDE-based modelers which allow for UML-based modeling
- Metamill: a round-trip UML tool for C++, C# and Java. Runs under Windows and Linux.
- Microsoft Visio:– a diagramming tool that also supports UML
- MyEclipse: An Eclipse-based IDE. Professional Edition includes UML solutions
- Objecteering: provides complete coverage of model-driven development (UML 2.0, MDA)
- OmniGraffle: for Mac OS X
- OptimalJ: a model-driven development environment for Java
- Poseidon for UML: commercial version of ArgoUML - supports UML 2.0
- PowerDesigner: by Sybase; supports UML 2.0, data-modeling, business-process modeling - round trip engineering
- Rational Rose: by Rational Software (sold to IBM in 2003); supports UML 1.x.

- Rational Rose XDE: an "eXtended Development Environment" in the tradition of Rational Rose; supports UML 1.x
- Rational Software Architect: Eclipse-based UML 2.0 tool by the Rational Division of IBM
- SDMetrics: a UML-design quality-measurement and design-rule-checking tool
- Select Architect: a UML/MDA platform for Microsoft Windows, running on a scalable repository it integrates with Eclipse and VS.NET
- SmartDraw: UML-diagram tool for Microsoft Windows
- Sparx Enterprise Architect: supports UML 2.1 and SysML
- Telelogic Rhapsody: supports UML 2.0 and SysML for embedded and real-time systems markets
- Telelogic TAU: supports UML 2.0 and SysML
- TextUML Toolkit: a tool for creating UML 2.1 models using a textual notation
- Use Case Studio: a use-case authoring tool by Rewritten Software. Free for educational use
- Visustin: reverse-engineers UML activity-diagrams and flow-charts
- Visual Paradigm for UML: supports UML 2.1, data modeling and business modeling
- WinA&D: UML and requirements management for Microsoft Windows
- yalips: modeling tool that also supports brainstorming and gantt project management

Addressing the compatibility issues raised in sharing data between collaborative organizations that employ different approaches to representing data becomes a major concern in today's net-centric computing environment.

Effective information exchange requires not only an agreement on the syntax and semantics to be established between data producers and consumers, but also a common understanding of the pragmatics, namely the intended use of the data in specific situational contexts. It is the development of a generic ontology framework

The next paragraphs describe the following software products or architectures solutions:

- Hitachi Software – Any*GIS
- ArcGIS Data Interoperability 9.2
- UML Analyzer Tool
- Executable UML (xUML)
- INTERLIS Studio
- 1Spatial Radius Studio
- Domain Solutions - CodeGenie RULES
- KAVI organisation
- UGAS – UML to GML

5.1 Hitachi Software – Any*GIS[20]

Any*GIS is an enterprise GIS platform that enables users to edit and update any GIS data anywhere. This platform is interesting for our study because it offers great interoperability features.

Users can access, edit and update multiple data formats using this GIS platform which is compliant with OGC standards. This platform is interoperable, allowing users to integrate multiple legacy data formats including Autodesk, ESRI, Intergraph, MapInfo and Smallworld.

Utility and telecommunication companies, local and central government agencies, as well as firms of all sizes can reduce their dependency on "closed" legacy GIS data formats while taking advantage of each application's feature and format strengths.

By means of an Internet connection and a browser, users can access, modify and update information without being concerned about data location, type, and format.

For example, a large electric utility needs to provide access to several spatial data formats for its employees and migrate mainframe-based GIS data to current RDBMS technology. Any*GIS provides access to the legacy spatial and CAD data, as well as a new data store for the mainframe data using Oracle Spatial.

The data can be accessed through the Any*GIS Professional Client or Web Client using LAN, WAN, and wireless networks. Engineering and drafting groups can access and update facility information through the AutoCAD Client on their existing CAD workstations. So, via the Internet, Any*GIS can serve the relevant data to the customers.

The design of Any*GIS Architecture (Figure 5.1) features three distinct tiers:

- Data Storage,
- Application Services and
- Client Applications.

This three-tiered design is ideal for enterprise systems because it enables the construction of portable and expandable systems.

It allows Any*GIS to simplify the integration of disparate data sources by providing a practical framework for publishing and querying data and maps over the web.

Whether data is generated in the main office, the satellite office, or in the field, Any*GIS allows seamless integration of all formats and sources through a common web interface.

Example in a local council

Within a local council, the property appraisal department has created an ESRI ArcInfo parcel basemap that the public works department uses for its CAD construction drawings.

Now, the local council wants to make its GIS data available throughout the organization and to the public.

Any*GIS enables the departments to share data of existing GIS software and data investments, without translating or reprojecting.

CAD users within the public works department can now access current GIS data and update information back to the GIS source. Any*GIS also allows the public to view relevant data through the municipal website.

[20] Web site: http://www.hitachisoft-gis.com/content-en/GWSENHomePage.htm

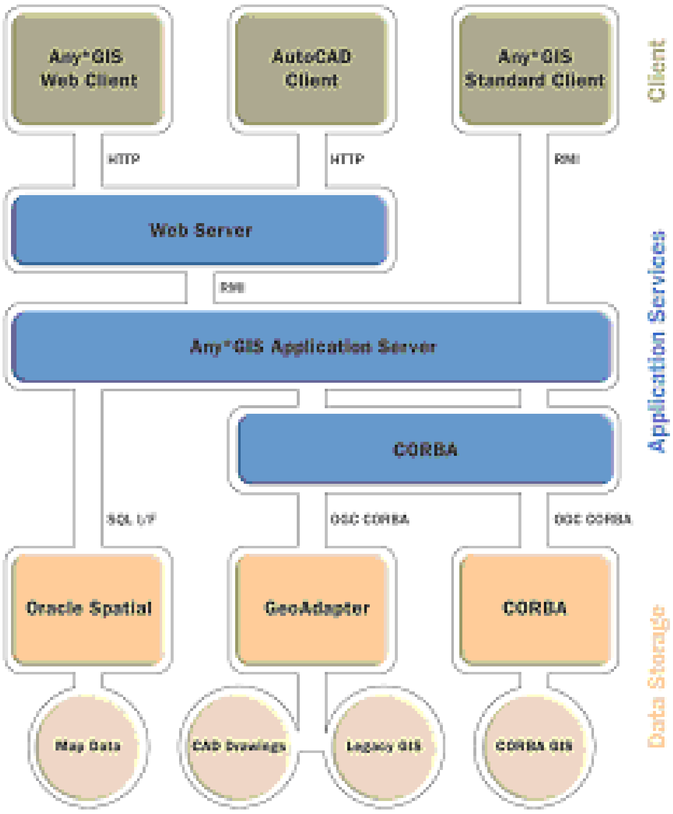


Figure 5.1 Any*GIS Architecture

The platform has three unique client interfaces that users can configure to access to shared data:

- Any*GIS Web Client
- AutoCAD Client
- Any*GIS Professional Client

5.1.1 Application Services Tier

The Application Services tier comprises of the Any*GIS Application Server and the Web Server, which are responsible for accessing, processing and serving data to the end-user via the client applications.

Web Server

The Any*GIS Web Server allows users to interface with standard internet servers such as Apache and Microsoft IIS.

Any*GIS Application Server

The Any*GIS Application Server is the functional backbone of Any*GIS, providing services for integration and transformation of spatial data.

5.1.2 Data Storage Tier

The Data Storage tier represents the various spatial and non spatial data repositories within an organization. These data repositories usually contain information in several non-compatible data formats. Data exchange between the application servers and data storage tier is accomplished with the use of GeoAdapters and CORBA components.

Oracle Spatial

Any*GIS provides reliable spatial object storage based on Oracle Spatial. This system provides full-functioned read/write access to spatial data including geometry and attributes management. Oracle Spatial provides transactional and locking control for enterprise use.

GeoAdapter

The Any*GIS GeoAdapters enable read/write functionality for the many varieties of supported native data formats, such as .DWG, .SHP and ArcInfo coverages. The GeoAdapters also provide connectivity to other legacy GIS systems and public digital maps.

5.2 ESRI - ArcGIS Data Interoperability 9.2

ArcGIS Data Interoperability[21] from ESRI[22] provides state-of-the-art direct data access, transformation, and export capabilities. This extension enables ArcGIS Desktop users to easily use and distribute data in many formats.

With ArcGIS Data Interoperability it is possible to:

- Directly read more than 75 spatial data formats including GML, XML, Autodesk DWG/DXF, MicroStation Design, MapInfo MID/MIF and TAB, Oracle and Oracle Spatial, and Intergraph GeoMedia Warehouse.
- Export to more than 50 spatial data formats.
- Model and diagram custom spatial data formats using a semantic data translation engine with 150 specialized transformers.
- Integrate with the geoprocessing framework including ModelBuilder to add data format manipulations within GIS models.

ArcGIS Data Interoperability can be used for:

- Directly access all standard data formats without the need to convert between formats.
- Share data with anyone by exporting to various formats.
- Create custom data formats and transformations that support complex data models.
- Improve productivity by working within a single GIS that integrates data from any source.

ArcGIS Data Interoperability enables ArcView, ArcEditor, and ArcInfo users to directly read and import more than 65 spatial data formats and export to more than 50. Users also have the flexibility to define custom data formats within an interactive visual diagramming environment. ArcGIS Data Interoperability is built on Safe Software's industry-standard Feature Manipulation Engine (FME) product.

Users can perform custom semantic translations to manipulate data as it moves from a source to a destination format. Quick Import and Quick Export tools allow automated conversion between different formats. Customized import and export tools can be used to refine the translation of data and enable schema redefinitions, giving the user full control of the translation/transformation process through a graphical data flow environment called Workbench.

This solution addresses a wide audience of users, as outlined in the following list

- Government departments sharing information
- GIS service providers
- Engineering consultants
- Data conversion companies
- Data publishers/vendors
- Utility companies accepting data from contractors
- Government organizations with the responsibility to deliver data
- Consultants who require integration/combination of data from different sources
- Enterprise organizations with multiple database GIS technologies

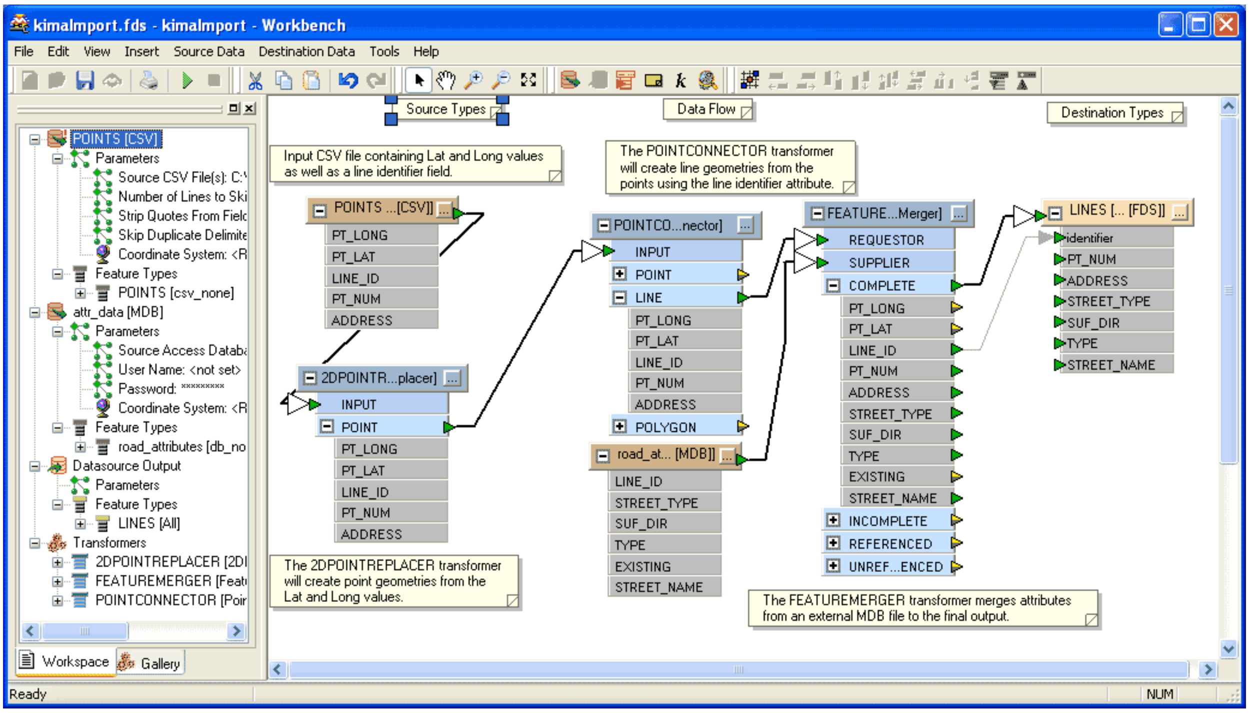
[21] Main product web page: http://www.esri.com/software/arcgis/extensions/datainteroperability/index.html

[22] Main web page: http://www.esri.com/index.html

- GIS data portals/clearinghouses

The most important features exposed by the software are the translation and transformation functionalities. With the first, users can set up import and export operations automating data translation between source and destination formats. With the latter, called Workbench, it is possible to exploit the integrated semantic translation engine in order to create or process custom formats and supporting specific data models.

An example of the visual capabilities of this engine is given in the Figure 5.2 below.



Input CSV file containing Lat and Long values as well as a line identifier field.

The POINTCONNECTOR transformer will create line geometries from the points using the line identifier attribute.

The 2DPOINTREPLACER transformer will create point geometries from the Lat and Long values.

The FEATUREMERGER transformer merges attributes from an external MDB file to the final output.

Figure 5.2 – Workbench Feature for Data Transformation

The Workbench application included with the ArcGIS Data Interoperability extension provides a visual diagramming environment with more than 150 transformers that enable you to transform both geographic and attribute information. Workbench can be used in two ways.

- Create custom formats to transform data on-the-fly.
- Create custom data tools for transforming data as you create new data sources.

Users can create new custom formats from any of the supported base formats by applying transformers to manipulate the attribute and geometric information stored within the file. It is possible to use a single data source as the input or combine a multitude of formats. This data transformation takes place on-the-fly, getting data integration without a preliminary conversion.

Figure 5.3 shows how to create new custom tools in order to help the data transformation process: these tools can perform custom semantic translations to manipulate data as it moves from source to destination formats. These tools use Safe Software's feature manipulation language of their Feature Manipulation Engine (FME). They allow for schema redefinitions and give the user full control of the translation/transformation process.

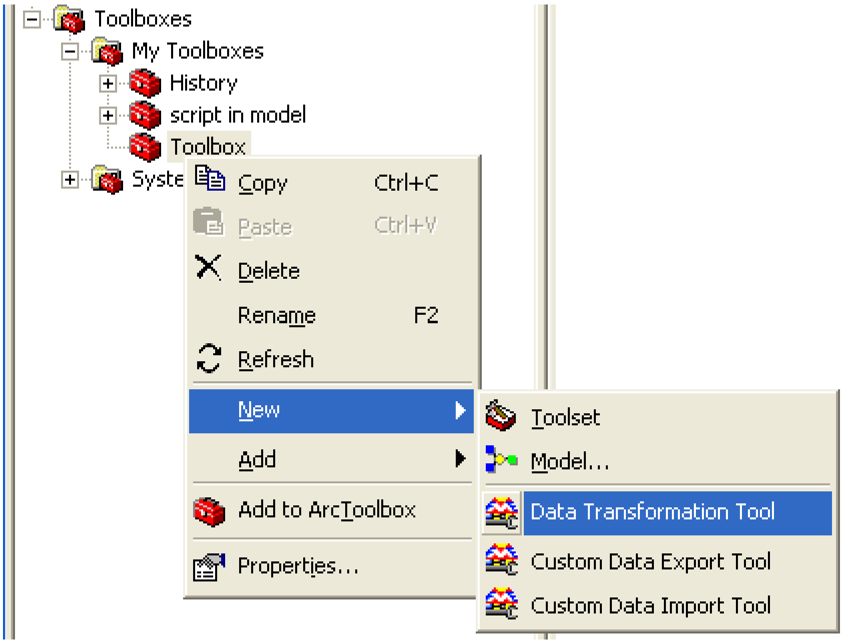


Figure 5.3 – Definition of a Data Transformation Tool

As showed in the figure 5.3 above, users can create:

- Data Transformation Tool, to transform data schema and geometry of feature classes within the geoprocessing environment.
- Custom Data Import Tool, to transform data as it is translated to a personal geodatabase.
- Custom Data Export Tool, to transform data as it moves from a source format to a destination format.

The list of all supported data formats recognised by the import/export facilities can be found in a Acrobat .pdf file available for download[23].

[23] Page address: http://www.esri.com/library/fliers/pdfs/data-interop-formats.pdf

5.3 UML/Analyzer Tool

The UML/Analyzer is a research tool that has been developed for defining and analyzing the conceptual integrity of UML models.

This tool was implemented in Java (MSJVM) and it is integrated with IBM Rational Rose, reusing the code from the Argo/UML project (developed by the University of California, Irvine). This tool is owned by the University of Southern California (USC).

The UML/Analyzer[24] tool is a software for defining and analyzing the conceptual integrity of UML models. UML/Analyzer describes and identifies causes of architectural and design mismatches across UML views as well as outside views represented in UML (e.g., C2 style architectures).

A GUI example of this tool is given in Figure 5.4 below.

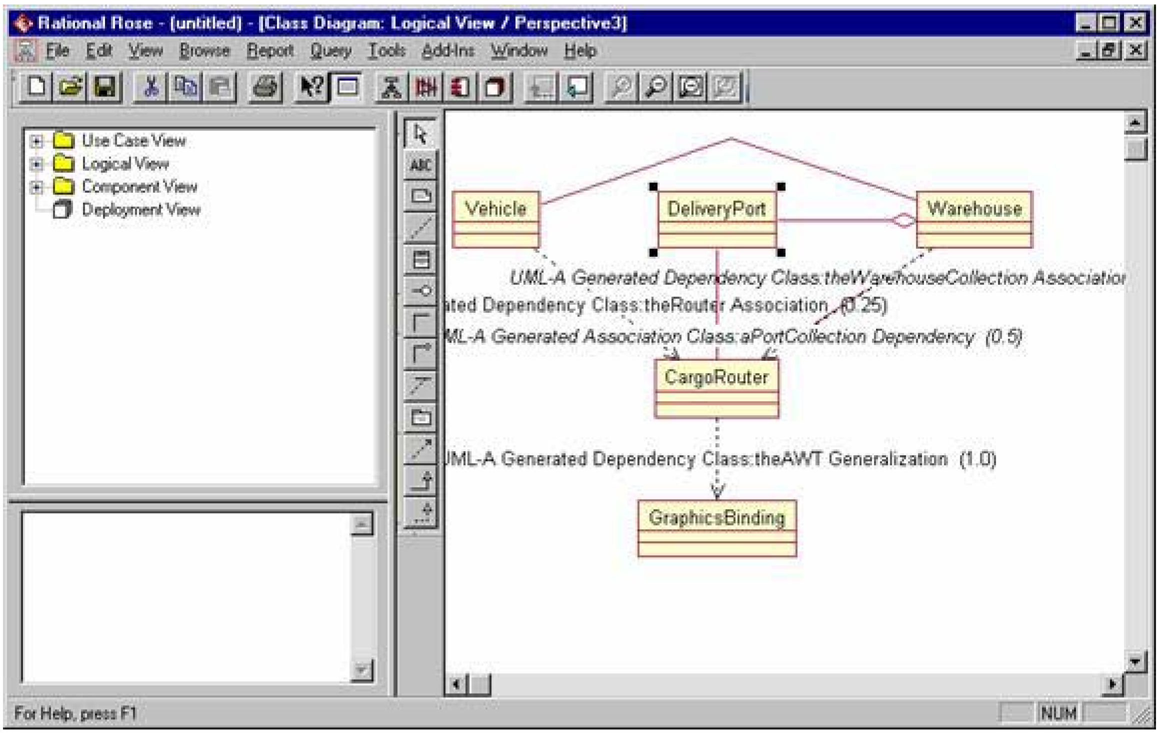

Rational Rose - [untitled] - [Class Diagram: Logical View / Perspective3]
File Edit View Browse Report Query Tools Add-Ins Window Help
Use Case View
Logical View
Component View
Deployment View
Vehicle
DeliveryPort
Warehouse
UML-A Generated Dependency Class:theWarehouseCollection Associatior
ated Dependency Class:theRouter Association (0.25)
ML-A Generated Association Class:aPortCollection Dependency (0.5)
CargoRouter
JML-A Generated Dependency Class:theAWT Generalization (1.0)
GraphicsBinding
For Help, press F1
NUM

Figure 5.4 UML/Analyzer Client

Typical users of this tool are the software designer and architects. Software development is more and more about modelling a real problem, solving the model problem, and interpreting the model solution in the real world. In doing so, a major emphasis is placed on mismatch identification and reconciliation within and among system views (such as diagrams).

The tool implements a generic view integration framework which supports automated model transformation and consistency checking within UML object and class diagrams as well as the C2SADEL architectural description language.

[24] Product web page at: http://www.alexander-egyed.com/tools/uml_analyzer_tool.html

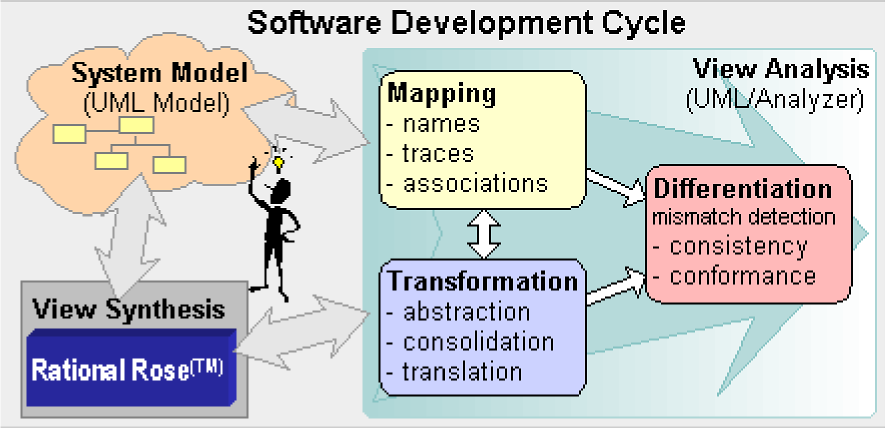


Figure 5.5 – View Integration Framework[25]

A general approach to modelling should include the following activities

1. Mapping: identifies and cross-references related modelling elements that describe overlapping and thus redundant pieces of information. Mapping is often done manually via naming dictionaries or traceability matrices (e.g., trace matrices). Mapping assists consistency checking by defining what to compare.
2. Transformation: converts modelling elements or diagrams into intermediate models in such a manner that they (or pieces of them) can be understood easier in the context of other diagram(s). Transformation assists consistency checking by defining how to compare.
3. Differentiation: compares model elements and diagrams with intermediate models that were generated through transformation where differences indicate inconsistencies.

The role of mapping is to ease transformation and differentiation by specifying what information needs to be exchanged and what information needs to be compared. The role of transformation is to enable more direct comparison by converting modelling elements into similar types and thus defining how modelling elements can be compared. Transformation also extends the model in that new (automatically generated) modelling elements are derived from the user-defined ones.

A further examination of this tool is also given in the chapter dedicated to the testing.

[25] The picture has been taken from http://sunset.usc.edu/available_tools/UML_analyzer/index.html

5.4 Executable UML (xUML)

Reference [15] states that “xUML process involves the creation of platform independent, Executable UML models with the UML diagrams supported by the action semantics-compliant Action Specification Language (ASL). The resulting models can be independently executed, debugged, viewed and tested”.

Multiple Executable UML models can be assembled together to form complex systems with the mappings between them being expressed using ASL.

Both the individual executable models and multi-model assemblies can then be translated into target implementations. The execution rules of the xUML formalism mean that the same models can be translated into a wide variety of target architectures without changes to the models.

A cornerstone of the translation approach is that the mappings to target implementations are themselves expressed using executable Models. These translation models are then used to code generate a code generator for the target system architecture.

In the language of the OMG's MDA approach, each Executable UML model is a Platform Independent Model (PIM). The mappings between such models (specified using ASL) are PIM-to-PIM mappings. The translation approach makes use of PIM to Platform Specific Model (PSM) and Platform Specific Implementation (PSI) mappings.

5.4.1 The xUML Formalism

xUML is a subset of the Unified Modelling Language incorporating a complete Action Language that allows system developers to build executable domain models and then use these models to produce high quality code for their target systems.

5.4.2 The xUML Development Process

The xUML process is a rigorous object-oriented system development method based upon experience with executable modelling that has been applied to many pioneering projects in sectors such as telecommunications, automotive, aerospace and defence, since 1994. It is founded on the principle of building a set of precise, testable models of the system to be developed, executing specific tests on these models and defining a systematic strategy by which the models will be used to produce code for the desired target system. The xUML process embodies these distinctive characteristics:

- Precise, Complete Models that can be subjected to rigorous testing by simulation. Thus, xUML provides a strong analysis completion criterion, which is: "The model is complete when it successfully executes the tests designed for it".
- Simple Notations, using the Unified Modelling Language organised into a coherent set of layers. This means that normal human beings, such as customers, hardware engineers and managers can read, understand and provide helpful feedback on the emerging models.
- An Understandable and Repeatable Partitioning Strategy, based upon the idea of Separation of Subject Matters. This means that both small and large system development projects can be partitioned in a way that avoids, for example, fruitless arguments about what constitutes analysis and what constitutes design.
- A Conceptually Coherent Process, that provides a small but sufficient set of techniques to address all subject matters, including "design", in a uniform way.
- Usable Models, which can be utilised by system designers and coders without the need for unreliable "interpretation" of the meaning of the models.

- Implementation by Translation, in which the entire system can, if desired, be automatically generated from the models, using a set of rigorously specified rules that deliver a system with the required performance characteristics.
- Large-Scale Reuse, in which entire sets of classes are reused as a single component.

5.4.3 The xUML Standard

Executable UML uses the Unified Modelling Language (UML) which is standardised by the Object Management Group (OMG). The OMG has recently issued a request for proposal (RfP) for a standardised Executable subset with rigorously defined semantics.

Executable UML is a major innovation in the field of software development. It is designed to produce a comprehensive and understandable model of a solution independent of the organization of the software implementation. It is a highly abstract thinking tool that aids in the formalization of knowledge, and is also a way of describing the concepts that make up abstract solutions to development problems.

Executable UML has been founded by E2E, a 10-year old firm that, as systems integrator, had to figure a way to join several retail banking networks together as part of the initial UBS merger in 1998. It adopted UML (Unified Modelling Language) to rationalize development of the connective code. But as it did so, E2E realized that it could save development time by making the UML model executable. The result is the E2E Bridge.

E2E’s approach can be considered as the emerging model-driven development in most respects. Users start with a requirements phase and process design platforms, such as IDS Scheer’s ARIS modelling tool, import the process model into UML, where the architecture of the application is developed. The last step is compilation to generate a UML byte code which is used to execute the application.

The first model is designed using a UML class diagram. The abstraction process requires that each object be subject to and conform to the well-defined and explicitly stated rules or policies of the subject matter under study, that attributes be abstractions of characteristics of things in the subject matter under study, and that relationships similarly model associations in the subject matter.

Next, the objects (the instances of the classes) may have lifecycles (behaviours over time) that are abstracted as state machines. These state machines are defined for classes, and expressed using a UML state-chart diagram. The abstraction process requires that each object be subject to and conform to the well-defined and explicitly stated rules or policies of the world under study, so each object is known to exhibit the same pattern of behaviour.

The behaviour of the system is driven by objects moving from one stage in their lifecycles to another in response to events. When an object changes state, something must happen to make this new state be so. Each state machine has a set of procedures, one of which is executed when the object changes state, thus establishing the new state.

Executable UML is a single language in the UML family, designed for a single purpose: to define the semantics of subject matters precisely. Executable UML is a particular usage, or profile, the formal manner in which we specify a set of rules for how particular elements in UML fit together for a particular purpose.

Executable UML is one pillar supporting the Model-Driven Architecture (MDA) initiative announced by the Object Management Group (OMG) in early 2001, the purpose of which is to enable specification of systems using models.

Model-driven architecture depends on the notion of a Platform-Independent Model (PIM), a model of a solution to a problem that does not rely on any implementation technologies. A PIM is independent of its platform(s).

A model of a online bookstore, for example, is independent of the user interface and messaging services it employs.

A PIM can be built using an executable UML.

Some proponents of MDA hold that a specification of the interface in a language such as the OMG's Interface Description Language (IDL), plus some constraints, is sufficient to specify without over-specifying. The views of these two camps are not contradictory, but complementary. There is no technical reason why a PIM specified using an executable UML cannot be bridged to one specified in terms of interfaces and constraints. One is just a more complete version of the other.

It is because an executable model is required as a way to specify PIMs completely that we view an executable UML as a foundation of model-driven architectures.

MDA also defines the concept of a Platform-Specific Model (PSM): a model that contains within it the details of the implementation, enough that code can be generated from it. A PSM is produced by weaving together the application model and the platforms on which it relies. The PSM contains information about software structure, enough information, possibly, to be able to generate code. Executable UML views the PSM as an intermediate graphical form of the code that is dispensable in the case of complete code generation.

At the time of writing this document, MDA is still being defined. However, some variation of the concepts of executable UML will, in our opinion, be required to support MDA. We offer our view on executable UML concepts here. Describing and defining MDA is another project and another book.

5.5 INTERLIS Studio

INTERLIS is a data exchange mechanism for Land-Information-Systems. This mechanism consists of a conceptual description language and a transfer format which in particular takes into account spatially related data (shortly geodata), thus permitting compatibility among various systems and long-term availability, i.e. depositing in archives and documentation of data.

INTERLIS is a standard which has been especially composed in order to fulfil the requirements of modelling and the integration of geodata into contemporary and future geographic information systems. The current version is INTERLIS version 2 (English). INTERLIS version 1 remains a Swiss standard. With the usage of unified, documented geodata and the flexible exchange possibilities the following advantage may occur:

- the standardized documentation
- the compatible data exchange
- the comprehensive integration of geodata e.g. from different data owners.
- the quality proofing
- the long term data storage
- the contract-proof security and the availability of the software

INTERLIS tools are a suit of products that support the INTERLIS standard. The author has found different references to the INTERLIS-Toolbox on the web and this solution seems interesting for UML test purposes.

The toolbox main features can be summarised in the following list:

- UML/ili-Editor*
- Compiler*
- jEdit*
- ili-Conversion-System ICS & Checker**
- ili-Viewer**

The INTERLIS-tools are available both in freeware or test version as in the following:

- *freeware[26]
- **test-versions[27]

UML/XMI-Interoperability of the INTERLIS-Toolbox is summarised hereafter:

- Rational Rose export XMI 1.1/UML 1.3
- Import to UML/ili-Editor > UML/ili > Werkzeuge/Importieren/XMI/Rose
- Display the grafics > UML/ili Datei/Öffnen/EuroRoadS_RR_mDiagramm.uml
- Export INTERLIS > Werkzeuge/Interlis/Exportieren
- Visual control ili-model > jEdit > Open EuroRoadS_RR_mDiagramm.ili
- Check the model > Werkzeuge/Interlis/Modell prüfen
- Optimizing the model by UMLI/ili-Editor/jEdit > EuroRoadS-RR-DATATYPES.uml/ili
- Check the model again > Werkzeuge/Interlis/Modell prüfen

[26] Web page at: www.interlis.ch

[27] Web pages at www.infogrips.ch or www.geocom.ch

- Export XML-schema / export GML-schema > GMLschema-D/ModelDef3.xsd + EuroRoadS_submodel.xsd

Examples of complex models – Swiss profile of ISO19115 /Swiss survey (central model and 26 canton models) > UML/ili GM03_V16.uml/ili/xsd.

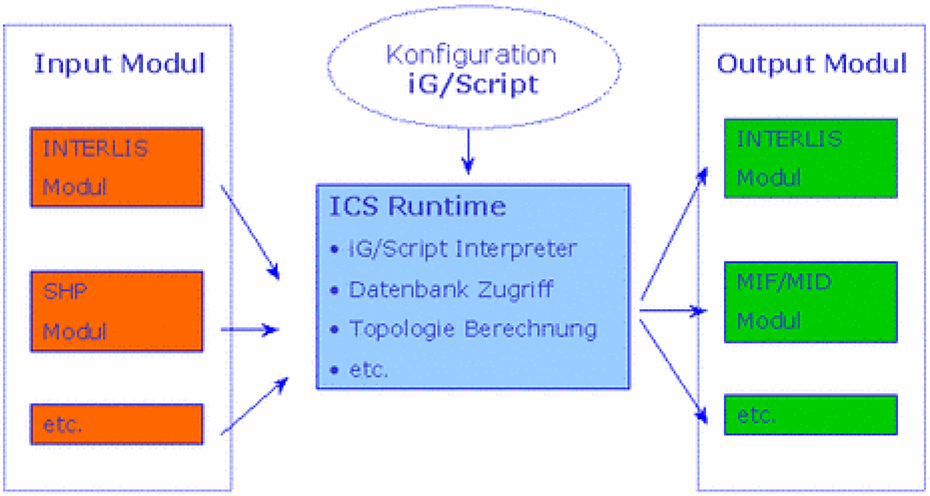


Figure 5.5 – INTERLIS Model Transformation

Compare UML/ili-model and import/export options with EnterpriseArchitect/Visio (only XMI1.0 for ESRI)

- The different tools don't understand each other because of different versions

⇨ need of a agreement within INSPIRE/CEN-community on a set of „usable versions“ of UML, XMI, etc. + establishment of a test environment

⇨ corresponding demand from the INSPIRE/CEN-community to the tool-manufacturers to follow the corresponding standards strictly

⇨ keep common models as „simple“ as possible

5.6 1Spatial Radius Studio

1Spatial is a world leader in the certification, management and the ongoing maintenance of spatial and spatially related data. 1Spatial's Radius products facilitate effective master data management, enabling data quality throughout the supply chain. These products are interoperable and independent of GIS; they can help you to collect, audit, store, manage, modify and reuse spatial data.

Radius Studio™

Radius Studio™ is a spatial processing, analysis and compliance engine that makes comprehensive use of the Oracle technology stack. It provides a data certification platform that ensures the quality and consistency of spatial data located in Oracle 9i or 10g Databases. This solution is delivered in a J2EE architecture that offers web access through the Oracle 10g Application Server and makes effective use of Oracle MapViewer.

Radius Studio enables users to rapidly analyse scattered spatial data to establish its operational purpose and facilitate its reuse by providing data mining, rules-based conformance checking and data cleaning capabilities. This rigorous control of data quality provides significant return on investment by automating these traditionally time-consuming, ongoing and expensive tasks.

Spatial data can be certified to enable critical business decisions to be made, safe in the knowledge that it is fit for a particular purpose because it is tested and documented to be so.

Radius Studio is currently being used within Central Government and Mapping organisations.

Product site: http://www.1spatial.com/products/radius_studio/index.php

Radius Studio™ is a spatial processing, analysis and compliance engine that ensures the quality and consistency of spatial data located in Oracle 9i or 10g Databases. It provides a data certification platform and makes comprehensive use of the Oracle technology stack. This solution is delivered in a J2EE architecture that offers web access through the Oracle 10g Application Server and makes effective use of Oracle MapViewer.

Radius Studio enables users to rapidly analyse scattered spatial data to establish its operational purpose and facilitate its reuse by providing data mining, rules-based conformance checking and data cleaning capabilities. This rigorous control of data quality provides significant return on investment by automating these traditionally time-consuming and expensive tasks.

Your spatial data can be certified to enable critical business decisions to be made, safe in the knowledge that it is fit for a particular purpose because it is tested and documented to be so.

5.7 Domain Solutions - CodeGenie RULES

Domain Solutions[28] was founded in 1995 to provide software automation tools and supporting professional services to increase software production efficiency.

Based on industry standards, Domain Solutions' CodeGenie toolset delivers proven reduction in software development costs and increase in levels of (model-based) software reuse.

CodeGenie RULES[29] is the first MDA model analyser designed to enforce compliance with your specified UML/MDA standards. Built on proven CodeGenie MDA technology, CodeGenie RULES uniquely combines:

- Application (PIM) modelling standard enforcement
- Architecture (PSM) modelling standard enforcement
- PIM model verification prior to PM transformation
- PSM model validation prior to code generation

The product delivers to users the following benefits:

- Find critical UML modelling errors quickly and easily
- Locate re-factoring opportunities in complex architecture models
- Raise model maintainability
- Increase solution performance by detecting inefficient models
- Enforce company-wide naming standards
- Ensure modelling best practice across all UML artefacts
- Report model deficiencies against an "executable" UML profile

In figure 5.6 below there is the architectural schema of the CodeGenie product.

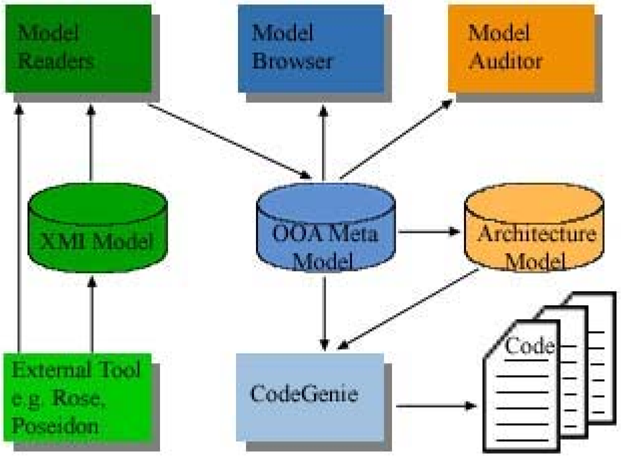


Figure 5.6 – CodeGenie Architecture

CodeGenie RULES exposes a great flexibility for static model analysis and rule implementation. Key features include:

[28] Official web site at http://www.domainsolutions.co.uk/
[29] Product main page at http://www.domsols.com/index.php?id=211

- User-defined rule categorisation (e.g. Naming Rules, Structure Rules)
- Rule and Rule Category Viewer
- Java-based rule expression
- Rule Editor
- User-configurable Rule Severity
- Architecture-specific (PSM) Rules
- Application-specific (PIM) Rules
- User-specified rule violation messaging
- Rule linking (e.g. Package to Class to Attribute, PSM to PIM)
- Interpreted and Compiled Rule options

5.8 KAVI

During this study the Author has found different organisations that deliver testing services or functionalities. One of these organisations is KAVI[30], a leading provider of enterprise-class solutions for standard-setting organizations. The figure below displays the activity areas where KAVI services can be of help.

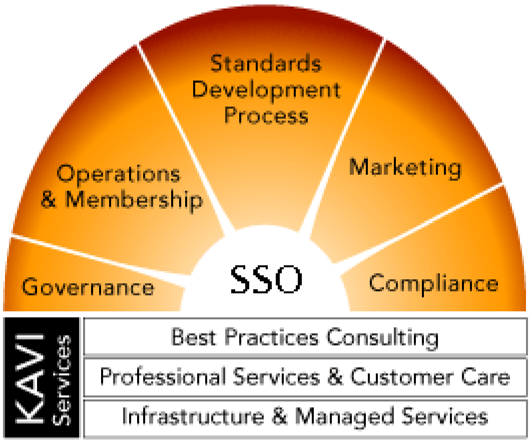

Standards Development Process
Operations & Membership
Marketing
Governance
SSO
Compliance
KAVI Services
Best Practices Consulting
Professional Services & Customer Care
Infrastructure & Managed Services

Figure 5.7: KAVI Services Platform

Kavi standard

According to Kavi, the Interoperability Testing can be used to know whether a product correctly and completely implements the standard, i.e. testing whether the implementation will interoperate with other implementations that are using the same standard.

The Conformance Testing is a different method to check that a product correctly and completely implements the standards. The product is tested against a list of items to see whether it performs as the standard specifies.

There are pros and cons of each of these types of testing. Interoperability testing is usually quite simple but wouldn't necessarily test the product against the entire standard. It may only test the most commonly used but simplest parts of the standard.

Interoperability tests should be designed to exercise as many of the capabilities of the implementation as possible.

Conformance testing requires considerably more effort, both in designing and conducting the test. Even a simple standard could have hundreds of test cases.

Given the advantages and disadvantages with both interoperability test and conformance testing it may be wise to do both, as these two types of tests are very complimentary with each other.

Compliance and Certification Information Solutions

Compliance and certification are a core business process for mature standard setting organizations – organizations that have built a community, developed and approved a specification, successfully promoted adoption of that specification. For many organizations at this stage, compliance becomes their primary reason for existence. Their compliance program is essential because:

[30] Organisation web page at http://www.kavi.com/home/

- A standard is a “brand”
- The brand promise is interoperability
- Certification is a “guarantee” of interoperability
- The credibility of the brand/standard is at stake

For organizations that administer standards that are aimed at interoperability in the consumer electronics industry, a credible certification program is an absolute necessity. Consumers must believe that a certification mark means that the device will interoperate with other products with the mark or they will not buy.

KAVI provides tools and processes that help organizations manage the information that flows into and out of their compliance and certification processes and present appropriate views into that information based upon stakeholders' roles and privileges.

KAVI participates in partnership with compliance program developers, test labs, and other experts who will drive the actual testing or certification activities. This organisation provides the tools and processes that help organizations manage the information that flows into and out of their compliance and certification processes and present appropriate views into that information based upon stakeholders' roles and privileges.

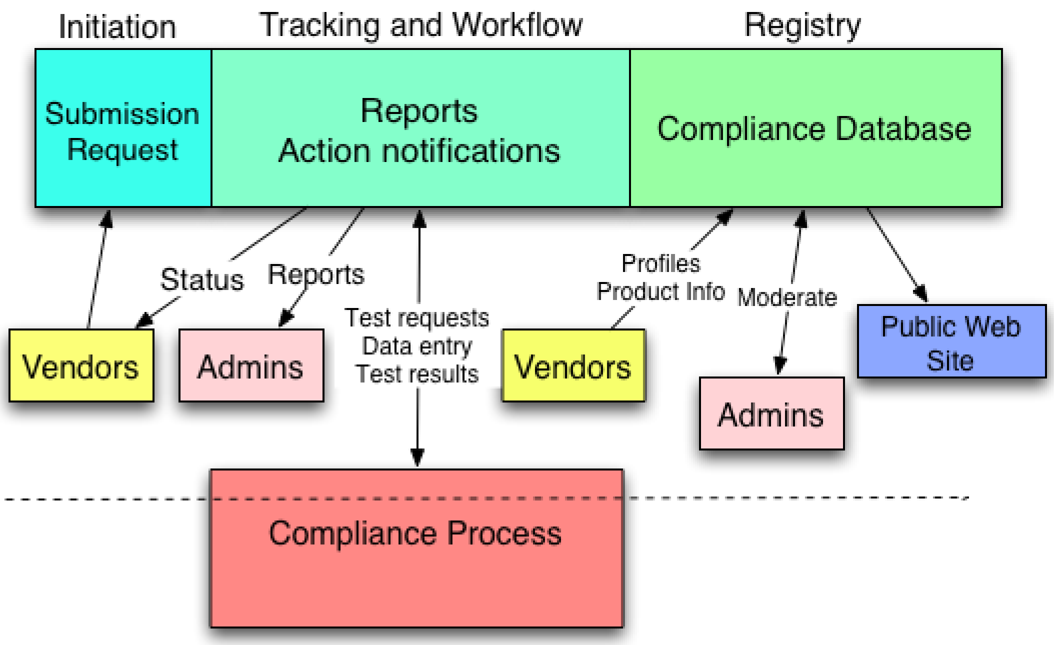


Figure 5.8 – KAVI’s role in Compliancy and Certification

In the previous figure 5.8 there is a schema of KAVI’s role in the compliancy process.

KAVI has the domain knowledge with which to help orgs design a compliance information management solution and facilitate the overall program information flow.

5.9 Interactive Instruments – ShapeChange

There are a set of tools presently on the market that deliver conversion functionalities of UML schemas into GML schema. These applications are categorised as *UGAS* tools.

One of the most prominent tools in this category is *ShapeChange*, a software that can generate a valid GML Application Schema when provided an UML model that follows a well-defined set of guidelines. The tool is based on the following documents of ISO/TC 211[31] and the Open Geospatial Consortium Inc:

- GML 3.0, GML 3.1 (ISO/CD 19136), GML 3.2 (ISO/DIS 19136)
- ISO 19118:2005
- ISO/TS 19103:2005
- ISO 19109:2005.

ShapeChange accepts UML models as input in XMI 1.0 format, recognizes and generates diagnostic error messages for incorrectly constructed UML models and XMI files and generates valid GML 2.1.2, 3.0.1 or GML 3.1.1 application schemas, given correctly constructed input.

Figure 5.9 displays the relations of involved parts in the ShapeChange software.

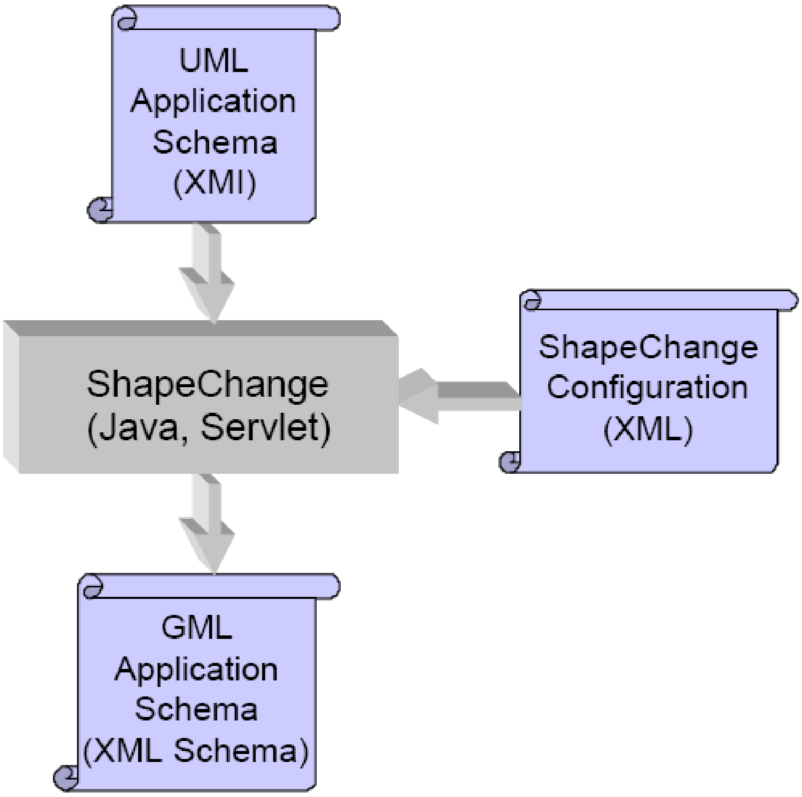

UML Application Schema (XMI)
ShapeChange (Java, Servlet)
ShapeChange Configuration (XML)
GML Application Schema (XML Schema)

Figure 5.9 – ShapeChange architecture

ShapeChange can query Web Registry Services for existing XMI documents provided that the schema metadata conforms to the ShapeChange requirements. It is a partial encoding service in the sense of ISO 19118:2005 chapter 9 and implements the “generateXMLSchema” operation.

[31] Browse the web page http://www.isotc211.org

ShapeChange configuration is stored in a XML file that contains the following information:

- Aliases for well-known stereotypes
- Namespaces
- Map entries for well-known classes to XML Schema elements and types

A stereotype alias is the alias of a known stereotype and it is represented by the following syntax:

```
<StereotypeAlias wellknown="FeatureType" alias ="Feature"/>
```

Stereotypes include elements of type FeatureType, DataType, Enumeration, etc.

A Namespace element specifies namespaces (XML Attribute "ns"), their abbreviations (XML Attribute "nsabr"), the corresponding GML version (XML Attribute "gmlVersion"), and the location of the corresponding XML Schema document (XML Attribute "location").

Examples:

```
<Namespace nsabr="xlink" ns ="http://www.w3.org/1999/xlink" gmlVersion="3.1"
location="http://schemas.opengis.net/gml/3.1.1/xlink/xlinks.xsd"/>

<Namespace nsabr="gml" ns=" http://www.opengis.net/gml " gmlVersion="3.1"
location="http://schemas.opengis.net/gml/3.1.1/base/gml.xsd"/>
```

Map entries are grouped in three categories:

1. Type Maps, used to map types of UML attributes and UML association roles to types of the corresponding GML property elements.
2. Element Map, used to map UML types to global XML elements (usually GML objects).
3. Base Map, used to map UML types to named XML complex types, for example to map inheritance relationships from the UML to the GML application schema.

Example of the configuration file:

```
<?x ml version="1.0" encoding="UTF-8"?>
<ShapeChangeConfiguration xmlnsabr=" http://www.interactive-instruments.de/ShapeChange"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"                              xsi:schemaLocation="
http://www.interactiveinstruments.de/ShapeChange ShapeChangeConfiguration.xsd">
<stereotypeAliases >
<StereotypeAlias wellknown="DataType" alias="Request"/>
<StereotypeAlias wellknown="DataType" alias="Response"/>
<StereotypeAlias wellknown="FeatureType" alias ="Feature"/>
</stereotypeAliases>
<namespaces >
<Namespace nsabr="sc" ns=" http://www.interactive-instruments.de/ShapeChange" gmlVersion="*"
location="http://www.interactive-instruments.de/ShapeChange/ShapeChangeAppinfo.xsd"/>
<Namespace nsabr="xlink" ns ="http://www.w3.org/1999/xlink" gmlVersion="2.1"
location="http://schemas.opengis.net/gml/2.1.2/xlinks.xsd"/>
<Namespace nsabr="gml" ns=" http://www.opengis.net/gml " gmlVersion="2.1"
location="http://schemas.opengis.net/gml/2.1.2/feature.xsd"/>
<Namespace nsabr="xlink" ns ="http://www.w3.org/1999/xlink" gmlVersion="3.0"
location="http://schemas.opengis.net/gml/3.0.1/xlink/xlinks.xsd"/>
<Namespace nsabr="gml" ns=" http://www.opengis.net/gml " gmlVersion="3.0"
location="http://schemas.opengis.net/gml/3.0.1/base/gml.xsd"/>
<Namespace nsabr="xlink" ns ="http://www.w3.org/1999/xlink" gmlVersion="3.1"
location="http://schemas.opengis.net/gml/3.1.1/xlink/xlinks.xsd"/>
<Namespace nsabr="gml" ns=" http://www.opengis.net/gml " gmlVersion="3.1"
location="http://schemas.opengis.net/gml/3.1.1/base/gml.xsd"/>
<Namespace nsabr="scXML" ns ="http://www.isotc211.org/scXML" gmlVersion="3.1"
location="http://www.isotc211.org/scXML/scXMLbase.xsd"/>
<Namespace nsabr="smXML" ns="http://www.isotc211.org/smXML" gmlVersion=" 3.1"
location="http://www.isotc211.org/smXML/metadataEntity.xsd"/>
</namespaces >
<mapEntries >
<!-- ISO/TS 19103 -->
<BaseMapEntry type="Record" gmlVersion=" *" rule=" direct" mapTo="Record_Type" nsabr="scXML"/>
<BaseMapEntry type="RecordType" gmlVersion="*" rule ="direct" mapTo="RecordType_Type"
nsabr="scXML"/>
<ElementMapEntry type="Record" gmlVersion="*" rule=" direct" mapTo ="Record" nsabr="scXML"/>
<ElementMapEntry type="RecordType" gmlVersion=" *" rule=" direct" mapTo="RecordType"
nsabr="scXML"/>
<TypeMapEntry type="Angle" gmlVersion="3.0" rule ="direct" mapTo="AngleType" nsabr="gml"/>
<TypeMapEntry type="Angle" gmlVersion="3.1" rule ="direct" mapTo="AngleType" nsabr="gml"/>
<TypeMapEntry type="Area" gmlVersion="3.0" rule ="direct" mapTo="AreaType" nsabr="gml"/>
<TypeMapEntry type="Area" gmlVersion="3.1" rule ="direct" mapTo="AreaType" nsabr="gml"/>
……………………………………………………………………………………………………………………………………………
……………………………………………………………………………………………………………………………………………
```

Behaviour of ShapeChange can be also modified by the following parameters:

- reportLevel: The level for the output of diagnostic messages, e.g. "WARNING". Further levels are " INFO", "ERROR", "FATAL ERROR"
- maxSizeOfRequest: maximum in bytes for data being processed to deny DoS attack, e.g. "20000"
- session-timeout: time in minutes a session will be valid, e.g. "180"

In order to run ShapeChange from the command line, the following components must be installed:

- Java (tested with version 1.5 [http://java.sun.com/j2se/1.5/index.jsp])
- Xerces2-J (tested with version 2.6.2 and 2.7.1 [http://xml.apache.org/xerces2-j/])
- Xalan-J (tested with version 2.6.0 and 2.7.0 [http://xml.apache.org/xalan-j/])

ShapeChange has been tested with Tomcat version 5.5.9. It also requires the following component:

- Jakarta Commons Net API (tested with version 1.4.0[32], JAR installed in the Java Runtime Environment)

With respect to the command line version, the web interface provides several further features for the user:

- Search Web Registry Services (WRS) for appropriate schemas
- Select a UML model found in the WRS for conversion
- Choose the conversion settings
- Store the conversion result files in a repository
- Download the result files from the repository
- Register the conversion result in one or more WRS for publication

The result of the mapping process is documented in a file ShapeChangeResult.xml in the output directory, including a sequence of Messages (fatal errors, errors, warnings, informational messages) and a list of XML Schema files that were created by the tool.

More information can be found in [11] together with an exhaustive description of the configuration and usage.

[32] Available at web page: http://jakarta.apache.org/site/downloads/downloads_commons-net.cgi

6. Critical Review

The solutions described in the previous paragraph have been chosen in the context of a general market survey using a simple criterion: they must expose at least one (or more) functionality addressing the INSPIRE data requirements expressed in paragraph 4.

In general this criterion has been centred more on the UML capabilities of the software, like modelling, schema validation at syntax and semantic levels, translation, etc. The list of the most known tools (both proprietary and not) for UML management, reported at the beginning of the previous paragraph, is enough proof of the wide market availability on the UML side.

The State of Play paragraph has also included solutions that can address other aspects of the INSPIRE Data Technical Chapter, like the ability to implement interoperability features in order to exchange data with different formats. Or, again, the capability to perform semantic translation of schemas from a source platform to a destination one.

The analysed solutions can suit in different way the needs of implementing INSPIRE-complaints data infrastructures. ArcGis Data Interoperability, for example, is a powerful and robust platform for the sharing of GIS data across different systems, delivering to users read, translation, and transformation capabilities. One of the major issues related to the requirements of data technical chapter is the possibility to translate from one format/model to different ones: the ArcGis Data Interoperability can be of help in this type of requirements.

A different software that supports a conversion feature of UML models is the ShapeChange product from Interactive Instruments. This tools conforms to the ISO/TC211 standard and performs validation and translation of UML models into GML application schemas. The users can benefit from this features in that he can create application schemas with the correct (i.e. compliant) abstract model. Validation is performed according to the selected standard and a configuration XML file can be used to generate the final GML result in a flexible way.

Similar to the previous one is the Domain Solutions' product, CodeGenie RULES, even if this software is more indicated for MDA architectures. CodeGenie can be used to design, analyse and enforce compliancy with PIM and PSM models. MDA technology is not really strictly related to INSPIRE requirements, but it's interesting enough for our survey. As a matter of fact, what more interests this study is the state of play of existing technologies coping with data model validation, translation and transformation. It can be stated that all this issues can be found in MDA PIMs and PSMs approaches to modelling.

The interoperability and conformance testing can also be a service with fee delivered by some IT firms inside the software market. The advantage of this approach is the possibility to free an organisation from the burden of purchasing (or using) a product with the unavoidable effort of getting skilled with it, thus investing in time and human resources. In other words, any check on data compliancy and interoperability can be delegated to others.

KAVI is one of these companies, taken into account during this survey and described in the previous chapter. KAVI has pointed out different services to support both the implementation and the test activities of any company involved with data sharing. Of course Kavi was just an example, meaning that all the actions required to conform to the INSPIRE data technical chapter can be supported not only by given tools but also by firms delivering the same results through services with fee.

A different point of view has been presented in the paragraph 5.4 dedicated to xUML. Along with real tools or services there are also some technologies that can be of help in the conformance and interoperability testing of data. An executable UML is a platform independent model implemented with ASL so that the final model can be executed, tested, viewed, etc

The xUML architecture can be therefore a useful compromise between a tool with given (and unchangeable) features and an external firm delivering consulting services. Thanks to xUML formalism, any executable model can be translated in different application schemas without making any changes to the beginning abstract models. The translation (mapping) process also ensures conformance to given rules ensuring compliancy and consistency at the syntax/semantic level.

The previous chapter has detailed other examples that can fit in the State of Play evaluation study (see UML Analyser Tool or Hitachi Any*Gis). They have been added to complete the picture of existing solutions. Generally speaking, all this examples address some of (but predictably not all) the INSPIRE requirements. What they lack, with exception of xUML, is the ability to deliver conformance and interoperability features in a flexible enough way. This flexibility would ensure them the ability to adjust to changes in the standard requirements.

Fortunately the market is rich of countless tools, platforms and more or less complex solutions focused on the web sharing of data.

The final outcome of INSPIRE implementing rules related to the data technical chapter, scheduled for the end of this year, will also complete the picture of the European programme requirements. It will be then possible to better understand what is the right approach in order to comply with the INSPIRE specifications.

7. Test Cases

Model syntax verification

Static validation is used to check whether a model is syntactically valid, i.e. whether the model is an instance of the UML meta-model including the well-formalness rules given by OCL. OCL is a query and expression language for UML that is an integral part of the UML standard.

This method works by the use of static validation i.e. by checking that the models conform to accurate syntax rules. It checks that the models do not violate the rules. This method can be run both automatically and manually depending on the tool that implements it. The goal with this method is to identify problems before they impact the software.

Test cases defined with UML 2 Testing Profile

In 2004 the UML 2 Testing Profile (U2TP) became a standard for testing the UML.

U2TP provides concepts to develop test specifications, and test models mainly for black-box testing. U2TP was developed because the OMG belief was that also the testing part of MDA should be specified with UML. U2TP was developed with TTCN-3 and JUnit in mind. The UML testing profile was introduced to bridge the gap between designers and test designers by providing a means to test with UML.

The profile allows for reuse of UML design documents and test development in an early system development phase. U2TP introduces four logical concept groups covering different aspects:

1. Test Architecture. Concepts introduced to specify structural aspects of the test system.
2. Test Behaviour. Specifies the behaviours and the objectives that are necessary to evaluate the test objective.
3. Test Data. Defines the test data and templates used in test procedures.
4. Test Time. Time quantified definition of test procedures.

To specify tests with U2TP activity, sequence, state machines, and interaction diagrams can be used. The most common way to specify a test case with U2TP is through sequence diagrams [42]. The sequence diagrams are used to specify the interactions between the test component and the SUT. The outcome of the test cases can be pass (SUT behaves as expected), fail (SUT does not behave as expected), inconclusive, and error (the test could not be concluded).

Pathfinder PathMATE

Pathfinder PathMATE makes it possible for developers to execute their design model at an early stage before coding or code generation begins. Pathfinder PathMATE integrates with Rational Rose and Rational Software Architect. It also has Eclipse 3.0 plugin architecture.

It works by transforming the PIM to C, C++ or Java software. PathMATE then executes the models, and lets the developers see what happens inside the system while running. In Pathfinder PathMATE small elements as well as whole systems can be tested [48].

By the use of Pathfinder PathMATE a tester can execute, and debug the PIM model in both the host and target environment. It has no editing capabilities or UML display but makes use of the editing environment used for development.

When using Pathfinder PathMATE the tester can insert break and trace points so that the program stops executing and collects information. These can later be saved in a driver file. The tester can also step through different events manually and stop the execution whenever it is appropriate.

When using Pathfinder PathMATE the tester can turn animation on to see were in the active state the system is while it is executing. The tester can choose both the appropriate speed of the animation as well as which classes to animate. In Pathfinder PathMATE the tester can assert/inject signals, messages, and interrogate and modify system parameters and variables to see what happens. Pathfinder PathMATE can therefore be used for protocol testing. It can also run different test scenarios created by 39 the tester. Finally Pathfinder PathMATE can also generate sequence diagrams when the system is executed.

There are different ways to test products developed in an MDA compatible way. Possible ways are to validate the design models, execute the design models, test the code that has been generated from the design models, and generate test cases from either test models or the design models.

The tools that have been presented in this chapter belong to 4 different categories:

- Tools that validate models
    - I-Logix Rhapsody
    - Telelogic TAU G2.
- Tools that test the code that has been generated from the design models
    - OptimalJ
    - Eclipse TPTP
    - RSA
    - Titan
    - Telelogic TAU G2.
- Tools that execute design models and display what happens inside them
    - Pathfinder PathMATE
    - I-Logix Rhapsody
    - Telelogic TAU G2.
- Tools that generate test cases from models
    - Conformiq
    - Cow SUITE
    - I-Logix Rhapsody
    - T-Vec RAVE.

7.1 OGC Compliancy & Interoperability Testing Related to GML

The OGC Compliance & Interoperability Testing & Evaluation (CITE) Initiative[33] is an ongoing initiative of the OGC that is building tests for OGC specifications. CITE-1, the first in the series, developed a planning and feasibility study that helped to refine the existing Compliance Testing Program. CITE-1 developed a scriptable compliance testing engine and tests for WMS 1.1.1 and WFS 1.0.0, as well as a validation capability for GML 2.1.2. The CITE-1 initiative also developed reference implementations for WMS and WFS and a portal resource dedicated to the subject.

OWS-2 developed a compliance test suite and reference implementation for WCS 1.0.0.

OWS-4 integrated a new open source engine that can be run from the command line or web interface, developed a new Compliance Test Language (CTL), migrated the existing WFS 1.0 and WMS 1.1.1 test suites to the new engine, and developed new test suites for CSW 2.0.1, WFS 1.1, WMS 1.3. In addition, the new engine will validate Web Map Context (WMC 1.1.0) instance documents. A GeoRSS feed validator was also created during this timeframe.

The GML 2.1.2 test engine supports the validation of GML 2.1.2 schemas and instance documents. There is currently one test suite version for Geography Markup Language validation available online via the Legacy CITE Engine[34]. As a stand-alone tool, it is not currently associated with any type of official compliance certification.

These tools available in the CITE portal pages and useful for the conformance test are described in the next sub-paragraphs.

7.1.1 TEAM Engine

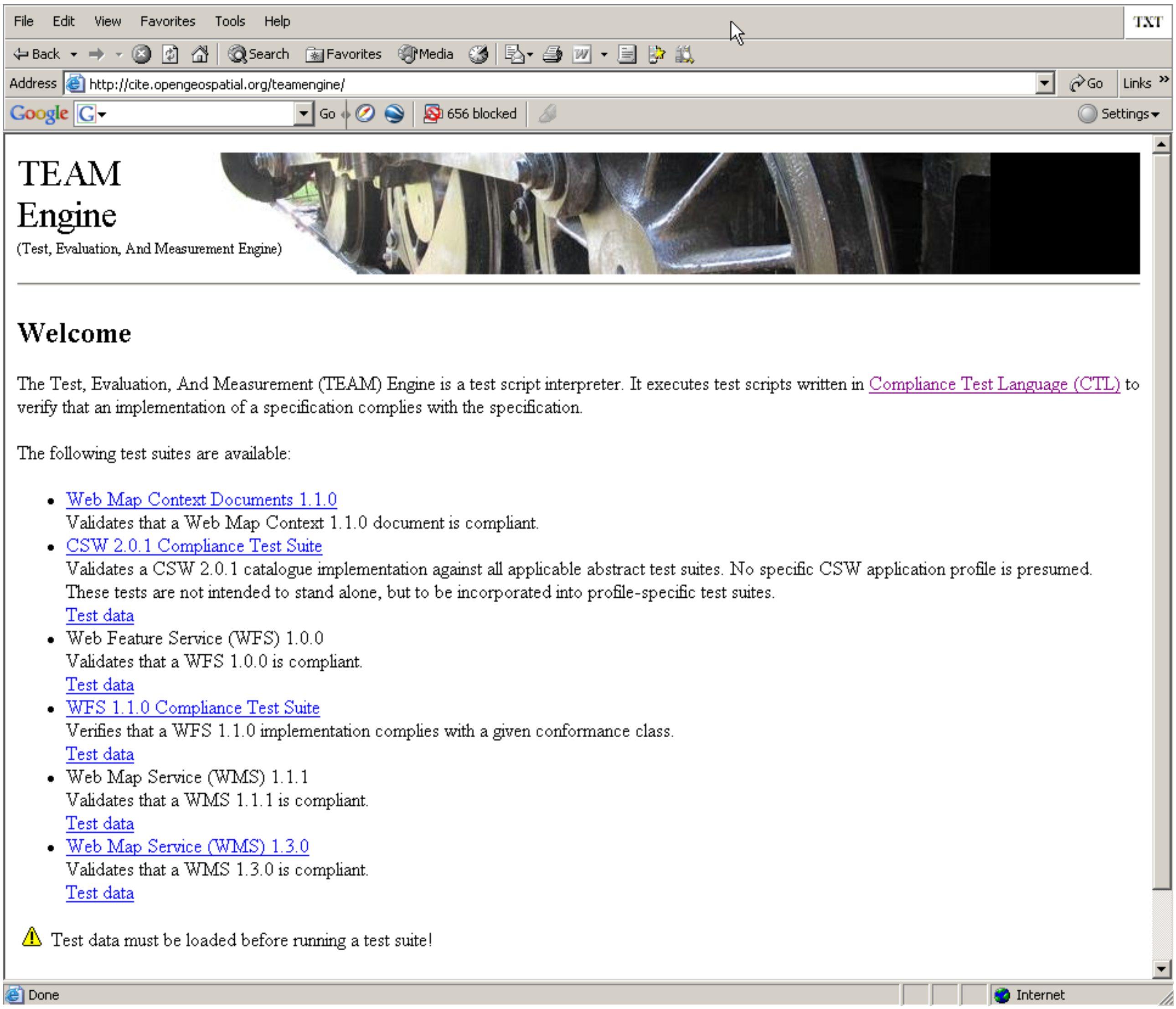
The Test, Evaluation, And Measurement (TEAM) Engine is a test script interpreter. It executes test scripts written in Compliance Test Language (CTL) to verify that an implementation of a specification complies with the specification.

The following test suites are available:

- Web Map Context Documents 1.1.0: validates that a Web Map Context 1.1.0 document is compliant.
- CSW 2.0.1 Compliance Test Suite: validates a CSW 2.0.1 catalogue implementation against all applicable abstract test suites. No specific CSW application profile is presumed. These tests are not intended to stand alone, but to be incorporated into profile-specific test suites.
- Web Feature Service (WFS) 1.0.0: validates that a WFS 1.0.0 is compliant.
- WFS 1.1.0 Compliance Test Suite: verifies that a WFS 1.1.0 implementation complies with a given conformance class.
- Web Map Service (WMS) 1.1.1: validates that a WMS 1.1.1 is compliant.
- Web Map Service (WMS) 1.3.0: validates that a WMS 1.3.0 is compliant.

The figure 7.1 below displays the TEAM engine web page[35].

[33] Web page at: http://cite.opengeospatial.org/cite
[34] Web page at: http://cite.opengeospatial.org/test_engine/gml/2.1.2
[35] Web page at: http://cite.opengeospatial.org/teamengine/



TEAM
Engine
(Test, Evaluation, And Measurement Engine)

Welcome

The Test, Evaluation, And Measurement (TEAM) Engine is a test script interpreter. It executes test scripts written in Compliance Test Language (CTL) to verify that an implementation of a specification complies with the specification.

The following test suites are available:

- Web Map Context Documents 1.1.0
  Validates that a Web Map Context 1.1.0 document is compliant.
- CSW 2.0.1 Compliance Test Suite
  Validates a CSW 2.0.1 catalogue implementation against all applicable abstract test suites. No specific CSW application profile is presumed. These tests are not intended to stand alone, but to be incorporated into profile-specific test suites.
  Test data
- Web Feature Service (WFS) 1.0.0
  Validates that a WFS 1.0.0 is compliant.
  Test data
- WFS 1.1.0 Compliance Test Suite
  Verifies that a WFS 1.1.0 implementation complies with a given conformance class.
  Test data
- Web Map Service (WMS) 1.1.1
  Validates that a WMS 1.1.1 is compliant.
  Test data
- Web Map Service (WMS) 1.3.0
  Validates that a WMS 1.3.0 is compliant.
  Test data

⚠ Test data must be loaded before running a test suite!

Figure 7.1 – Team Engine Home Page

CTL is a language from OGC used to develop tests for several specifications. In detail, (reference [10]), CTL is an XML grammar for documenting and scripting suites of tests for verifying that an implementation of a specification complies with the specification.

A suite of CTL files is typically installed in a compliance test engine, which executes the scripts and determines whether the implementation being tested passes or fails. The CTL files can also be used to generate user documentation for the tests in the suite.

A test suite in CTL consists of a set of objects. The initial object is a suite, which identifies a starting test. The starting test contains instructions that may call other tests or use functions and parsers as outlined in Figure 7.2.

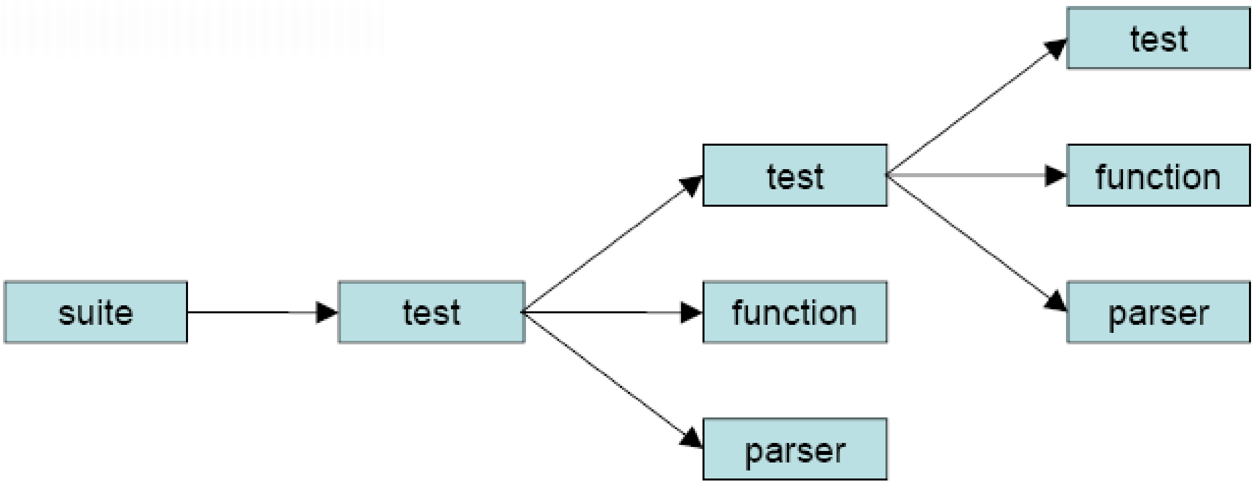


Figure 7.2 – CTL Test Suite

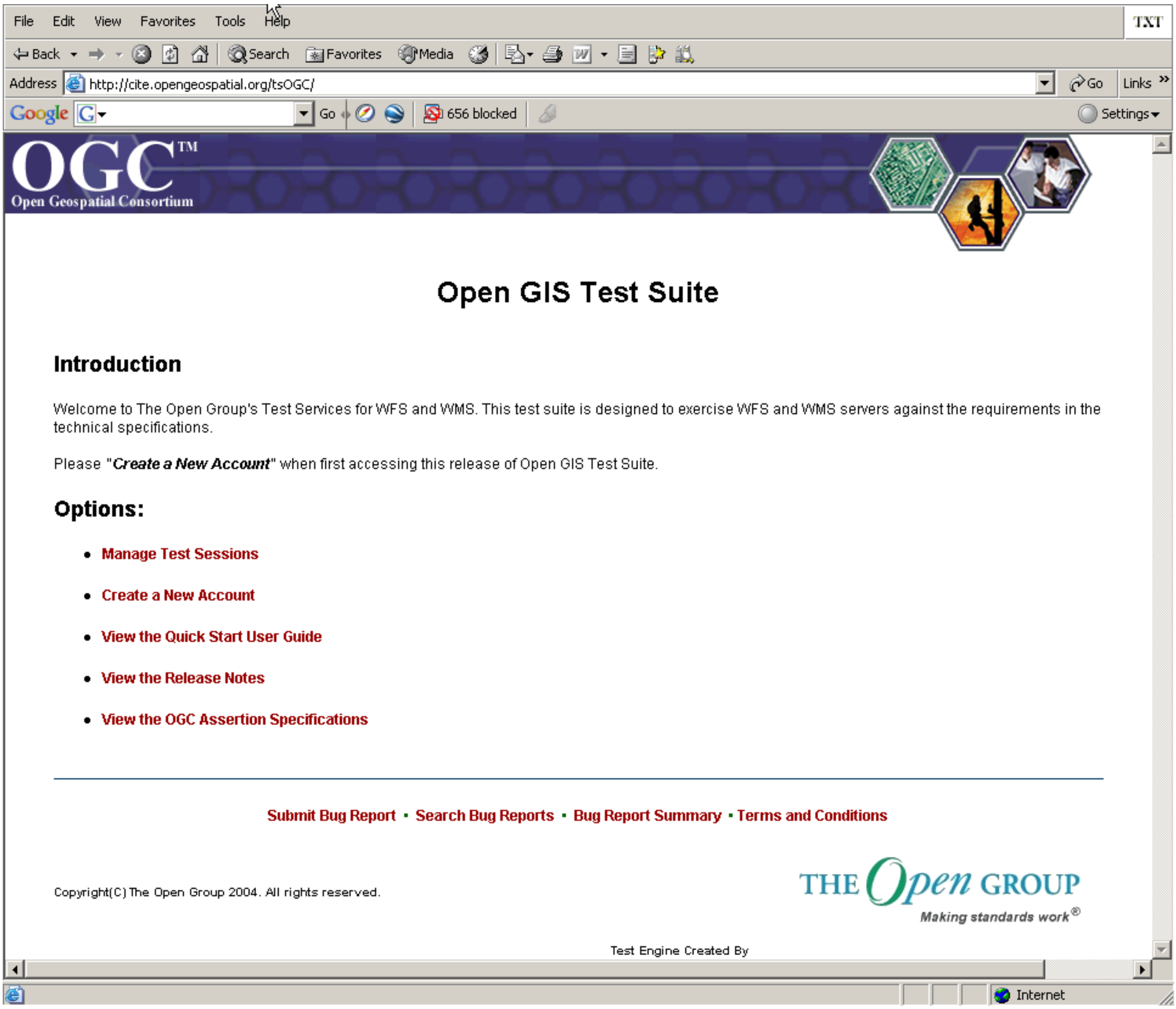
A CTL file is an XML file that contains a CTL object or a package element as its root element. The package element is a container for multiple CTL objects. Each CTL object is identified by a unique, namespace qualified name, so the set of objects in a test suite may span several files.

Test objects contain programmatic code that consists of XSL instructions and/or CTL instructions.

7.1.2 WCS Compliance Testing

This web service makes possible the test of WCS and WCF servers against the requirements in the technical specifications. The page dedicated to this service (see Figure 7.3) presents a link to the available assertions specifications and listed below:

- CSW ebRIM 1.0.0
- Catalogue Services 2.0.0
- Open Location Directory Service 1.0.0
- Open Location Gateway Service 1.0.0
- Open Location Presentation Service 1.0.0
- Open Location Route Service 1.1.0
- Open Location Utility Service 1.0.0
- Web Coverage Server 1.0.0
- Web Feature Service
- Web Feature Server 1.1.0
- Web Map Server
- Web Map Server 1.3.0


File Edit View Favorites Tools Help
TXT
Back Search Favorites Media
Address http://cite.opengeospatial.org/tsOGC/
Go Links
Google Go 656 blocked Settings
OGC™ Open Geospatial Consortium
THE Open GROUP Making standards work®
Internet

Open GIS Test Suite

Introduction

Welcome to The Open Group's Test Services for WFS and WMS. This test suite is designed to exercise WFS and WMS servers against the requirements in the technical specifications.

Please "***Create a New Account***" when first accessing this release of Open GIS Test Suite.

Options:

- Manage Test Sessions
- Create a New Account
- View the Quick Start User Guide
- View the Release Notes
- View the OGC Assertion Specifications

Submit Bug Report ▪ Search Bug Reports ▪ Bug Report Summary ▪ Terms and Conditions

Copyright(C) The Open Group 2004. All rights reserved.

Test Engine Created By

Figure 7.3 – WCS Testing Home Page

The user must first create an account giving his agreement on the evaluation license conditions. After the login it is fundamental to create a test session and setting the session options from the HTML browser with the following steps[36]:

- Select the options and enter details applicable to your device and then click the *Configure Test Session* button.
- Input the variables and areas that the server under test supports.
- Either create the test session, or get variables and scopes from the capabilities document of the server under test.
- Select the *Add Capabilities Values* button, and then the values are gained from the capabilities document.
- Select the *Create Test Session* button to create the actual test session.
- The browser now reloads the initial page *Test Session Management Menu for Account <username>*, where a new test session will be found in the Session table.
- Test ID's are numbered from 0001. The user can also provide an arbitrary description for the test session.

[36] For a detailed description refer to http://cite.opengeospatial.org/tsOGC/ViewQuickStartGuide

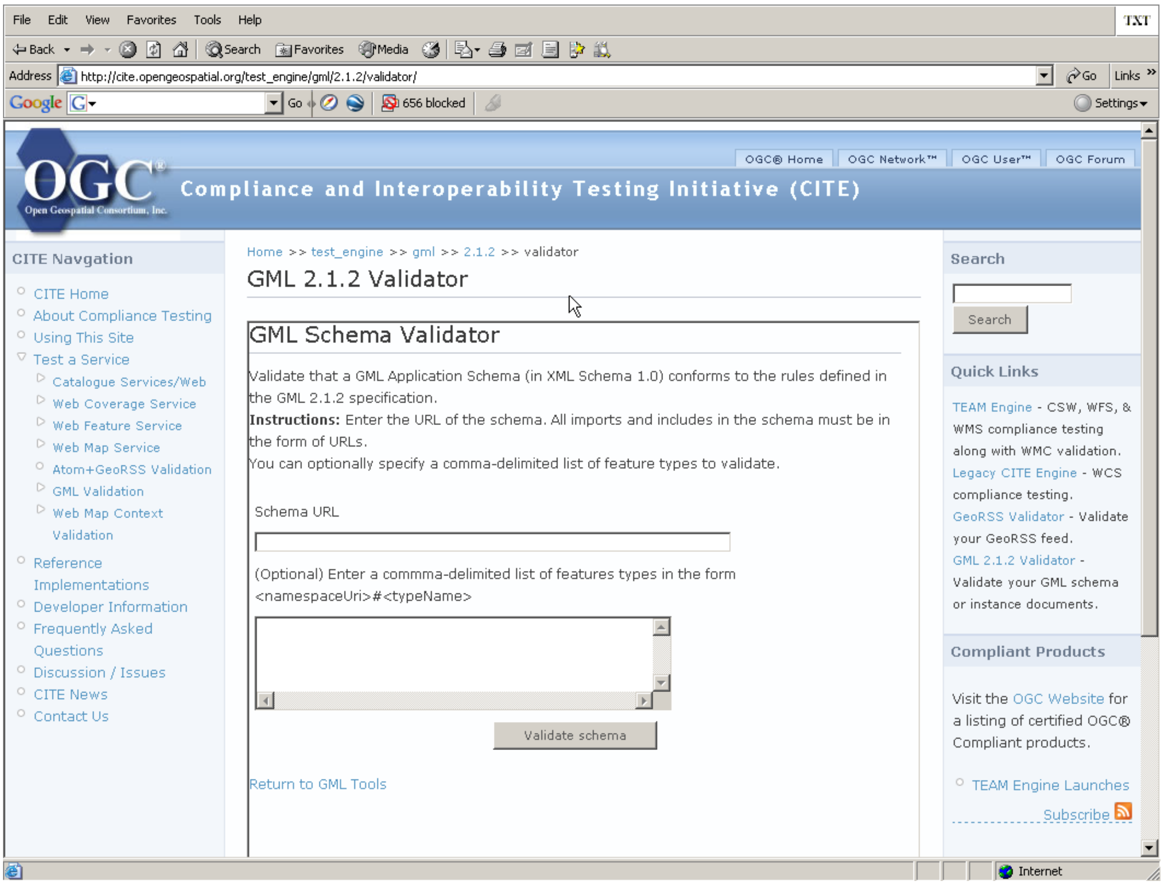
The final step is the execution of the session test accomplished with the following steps:

- Select the *View Test Results* option for the test session you wish to test. This will take you to the *Summary of Test Session* page. Select the *Execute Test* link next to the test from which point you wish your test session to begin execution. Test execution will automatically sequence through the tests in the order they are listed in the summary table.
- Select the *Execute Tests* option from the test session management menu. This will carry out the test execution as for the previous method except that test execution will start from the starting test. This defaults to the first test in newly created test sessions.

Upon completion of the test execution the user can browse the test results reloading the *Summary of Test Session* page. Customised test session options used to create the session can be viewed at any time selecting the *View the Configuration Settings* hyperlink.

7.1.3 GML 2.1.2 Validator

The GML application schema validator checks that the schema is valid according to the XML Schema 1.0 rules and the GML 2.1.2 rules. The schema must be available on a web server via a HTTP URL.



GML 2.1.2 Validator

GML Schema Validator

Validate that a GML Application Schema (in XML Schema 1.0) conforms to the rules defined in the GML 2.1.2 specification.
Instructions: Enter the URL of the schema. All imports and includes in the schema must be in the form of URLs.
You can optionally specify a comma-delimited list of feature types to validate.

Schema URL

(Optional) Enter a commma-delimited list of features types in the form <namespaceUri>#<typeName>

Validate schema

Return to GML Tools

TEAM Engine - CSW, WFS, & WMS compliance testing along with WMC validation.
Legacy CITE Engine - WCS compliance testing.
GeoRSS Validator - Validate your GeoRSS feed.
GML 2.1.2 Validator - Validate your GML schema or instance documents.

Visit the OGC Website for a listing of certified OGC® Compliant products.

Figure 7.4 – GML Schema Validator

The following checks are performed.

- XML Schema 1.0 Validity (Users Xerces schema validator).
- All feature type definitions must extend xmlns (gml=http://www.opengis.net/gml) AbstractFeatureType or from a complex type that extends that type.

- A GML feature definition must not have a direct child element that derives from xmlns (gml=http://www.opengis.net/gml)gml:AbstractFeatureType.
- A GML feature definition must not have a direct child element that derives from xmlns(gml=http://www.opengis.net/gml)gml:AbstractGeometryType.

Note: The schemas must define the schemaLocation for all import and include statements that are resolvable from the source schema URL.

7.2 Interactive Instruments – ShapeChange

As outlined in paragraph 5.9, ShapeChange maps an UML (ISO 19109 conformant) Application Schema to the corresponding GML Application Schema. The modelling guidelines have been well documented in [12] and prior to use the tool the author suggests the reading of this document in order to know the required encoding rules with respect to UML and XMI.

This paragraph is dedicated to the experimental ShapeChange web interface[37] publicly available for testing purposes. Please note that the web page anyway may not be always accessible. ShapeChange is © 2002-2005 interactive instruments GmbH and the source code is available under the GNU Public License.

The input UML Application Schema must be conformant to ISO 19109:2005 and follow some additional modelling rules described in [12] and summarised in the sequel of this paragraph.

UML requirements

The UML Application Schema is a valid input into a mapping if it conforms to GML 3.1 (ISO/CD 19136) Annex E. In some cases, experimental modelling elements are supported, too, but must be activated using non-standard switches. Now GML 3.2 (ISO/DIS 19136) has recently been published for review, therefore support for this version will be added in the near future.

XMI requirements

The XMI representation of the UML Application Schema is a valid input into the mapping process if it conforms to the following conditions:

- The UML model containing the application schema and all other required model elements shall be stored in a single XMI document.
- The XMI document shall be well-formed.
- The XMI document shall conform to XMI version 1.0.
- The XMI document shall be valid, i.e. contain a DOCTYPE declaration and the document must validate against this document type definition. This DTD must be the normative DTD that is part of UML 1.3.
- Only the contents of the <XMI.header> and the <XMI.content> elements are be used by the UGAS Tool. All other elements will be ignored.

Encoding Rules

The mapping from an ISO 19109 conformant UML Application Schema to the corresponding GML Application Schema is based on a set of encoding rules. These encoding rules are identical with those specified in GML 3.1 (ISO/CD 19136) Annex E plus experimental extensions. Only the deviations from GML 3.1 and experimental extensions are specified in this document.

The schema encoding rules are based on the general idea that the class definitions in the application schema are mapped to type and element declarations in XML Schema, so that the objects in the instance model can be mapped to corresponding element structures in the XML document.

The target version of GML is by default 3.1.1.

The schema conversion rules define how to produce XML Schema documents (XSDs) according to an ISO 19109 application schema expressed in UML.

In addition to the rules specified in GML 3.1 (ISO/CD 19136) the following rules exist:

[37] Available at: http://services.interactive-instruments.de/ShapeChange/servlet/ConverterServlet

- All tagged values that are implicitly or explicitly associated with ShapeChange are mapped to appinfo elements of the corresponding model element.

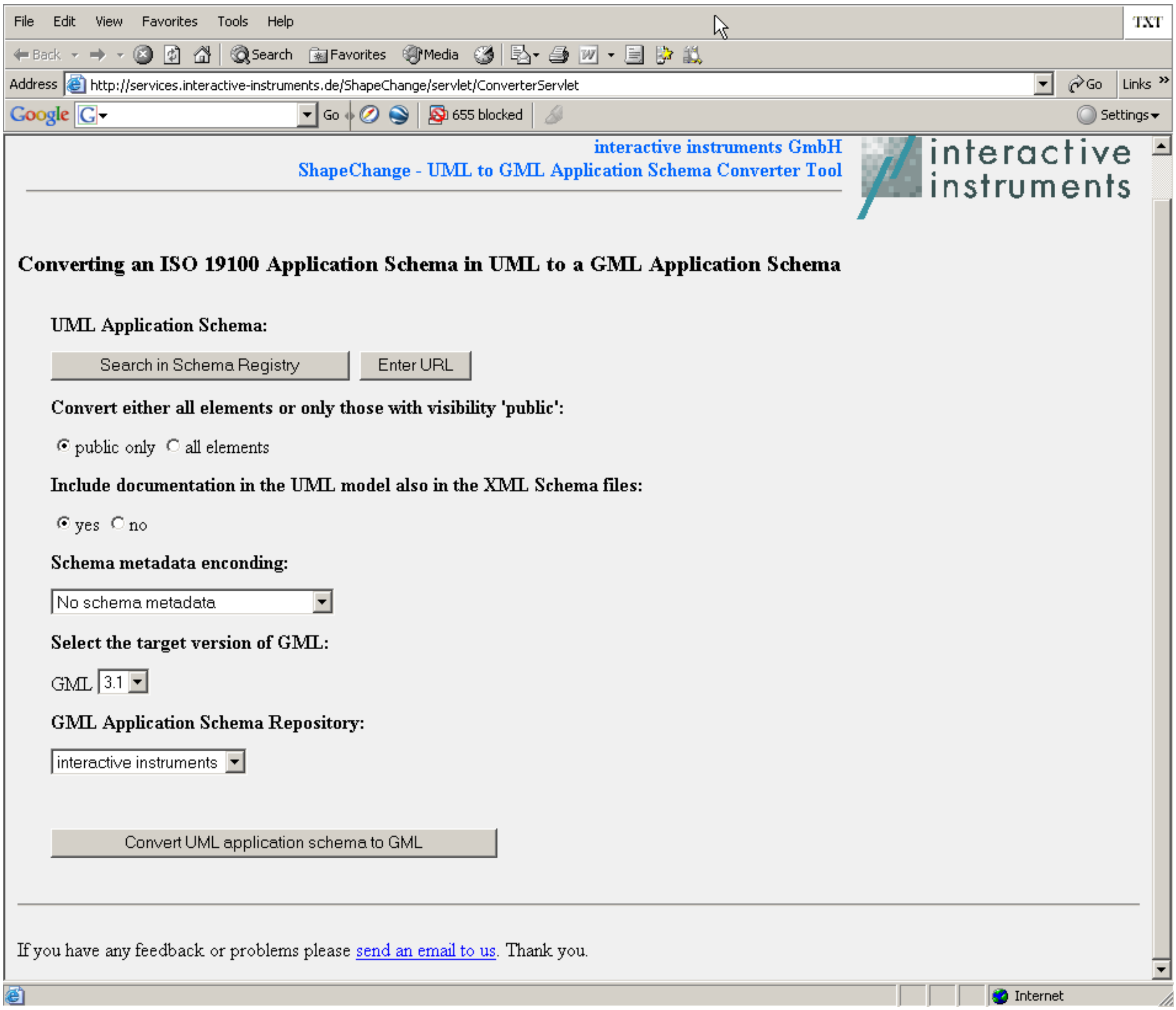
Example:

```
<complexType name="PAA010Type">
    <annotation>
        <documentation>Mine: An excavation made in the earth for the purpose of extracting natural deposits.</documentation>
        <appinfo>
            <sc:taggedValue tag="name">Mine</sc:taggedValue>
            <sc:taggedValue tag="code">PAA010</sc:taggedValue>
        </appinfo>
    </annotation>
…
</complexType>
```

In addition to previous rules, the following options also exist (often experimental):

- It can be selected whether documentation elements are included in annotation elements in the XML Schema files.
- <<Type>> classes may be mapped to GML objects or features.
- <<Union>> classes may be mapped to a complexType with a choice or to a group containing a choice.
- Properties may be mapped to local elements or global groups (to support restrictions across namespaces - not recommended).
- Property types for all complexTypes may be added or not.
- Property-by-value types for all complexTypes may be added or not.
- <<Enumeration>> classes may be mapped to a named simpleType or to an anonymous simpleType.
- <<CodeList>> classes may be mapped to an extendable enumeration or to a GML Dictionary
- Restrictions redefining properties may be allowed or prohibited (recommended).
- Parameterized types Set<T> and Sequence<T> may be supported.
- Either only packages carrying a tagged value 'xsdDocument' are mapped to an XML Schema document or all packages are mapped to an XML Schema document.
- Schema metadata may be added in an appinfo element of the <xs:schema> element.
- Currently only the experimental schema metadata of the OWS-3 inititative is supported (using the DDMS schema).

When the user navigates to the ShapeChange web address, the browser opens the page displayed in the figure 7.5.


File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://services.interactive-instruments.de/ShapeChange/servlet/ConverterServlet
Go Links
Google Go 655 blocked Settings
Internet

interactive instruments GmbH
ShapeChange - UML to GML Application Schema Converter Tool

Converting an ISO 19100 Application Schema in UML to a GML Application Schema

UML Application Schema:

Search in Schema Registry
Enter URL

Convert either all elements or only those with visibility 'public':

● public only ○ all elements

Include documentation in the UML model also in the XML Schema files:

● yes ○ no

Schema metadata enconding:

No schema metadata

Select the target version of GML:

GML 3.1

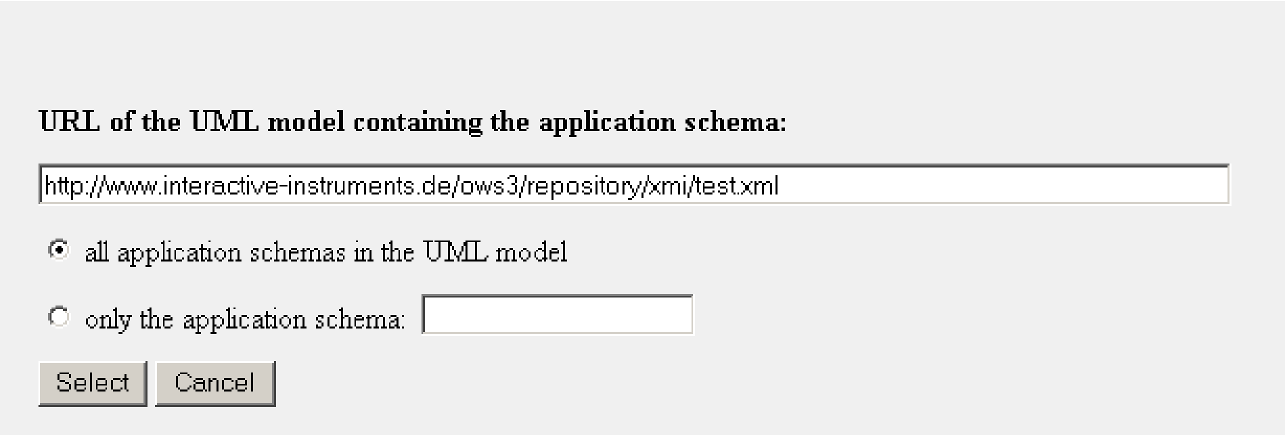
GML Application Schema Repository:

interactive instruments

If you have any feedback or problems please send an email to us. Thank you.

Figure 7.5 – ShapeChange Web Page

The user can search the application schema in the schema registry or providing the URL to a specific schema. In the latter case, the user can choose if the tool should consider all the application schemas contained in the model or only the one specified by the user (figure 7.6).



URL of the UML model containing the application schema:

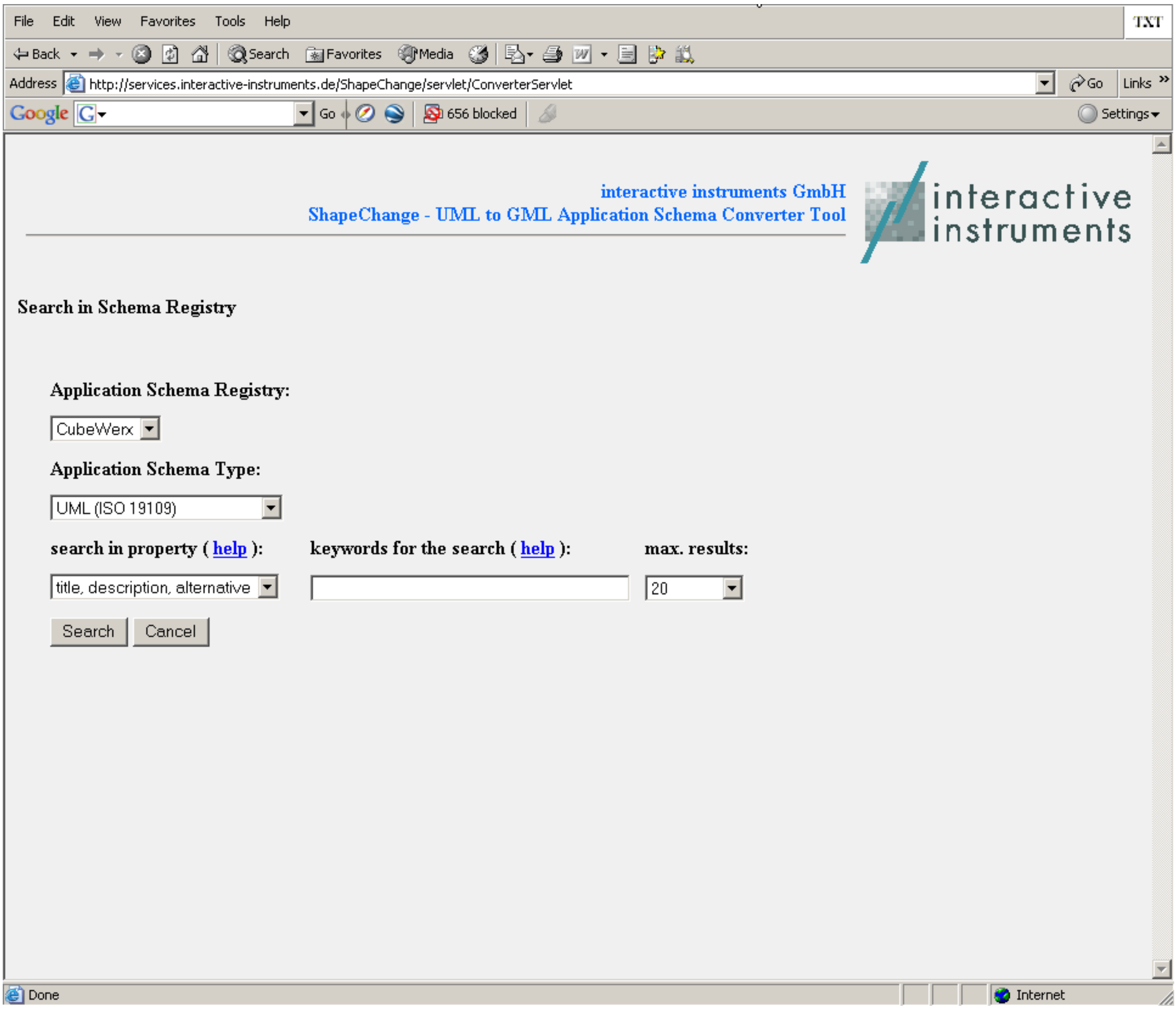
http://www.interactive-instruments.de/ows3/repository/xmi/test.xml

● all application schemas in the UML model

○ only the application schema:

Figure 7.6 – Defining the Application Schema

When the user presses the *Search in the Registry* button, the browse open a new page as displayed in figure 7.8.



File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://services.interactive-instruments.de/ShapeChange/servlet/ConverterServlet
Go Links
656 blocked
Settings
interactive instruments GmbH
ShapeChange - UML to GML Application Schema Converter Tool
interactive instruments

Search in Schema Registry

Application Schema Registry: CubeWerx

Application Schema Type: UML (ISO 19109)

search in property (help): title, description, alternative

keywords for the search (help):

max. results: 20

Search
Cancel
Done
Internet

Figure 7.8 – ShapeChange Registry Lookup

Users must choose the application schema type, define the filter criteria and start the look-up.

Other options can be set in the tool main page (Figure 7.5). When all the configuration settings have been defined, the user can start the conversion clicking on the existing button.

7.3 UML Analyzer Tool

The following summarizes the usage of UML/Analyzer in the context of a Disaster Relief Scenario. This demo was created for the EDCS project (Evolution and Design of Complex Systems) to demonstrate the integration of USC-CSE technologies (University of Southern California-Center for Software Engineering) that span requirements capture, architecture, and design.

UML/Analyzer was used in this scenario to ensure consistency between an architectural diagram (called C2) and the corresponding UML class diagram following the C2 style.

The architecture shows a high level view of a Cargo Router Simulator (a piece of the EDCS demo). The major C2 components of that architecture are Clock, Warehouse, Delivery Port, Vehicle, Cargorouter, and GraphicsBinding. The elements labelled ClockConn, RouterConn, and GraphicsConn represent C2 connectors. The role of connectors is to bridge the interaction of components.

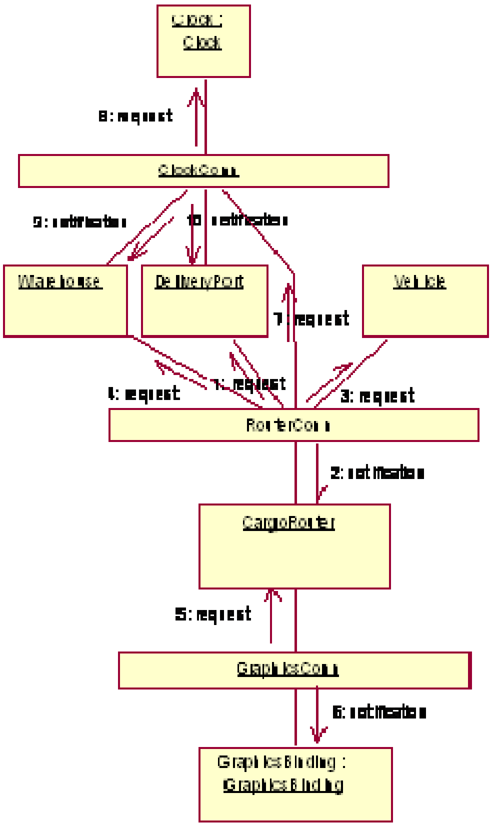


Figure 7.1 – C2 Architecture in UML

The C2 architectures impose a number of configuration constraints:

- C2 components can only interact via connectors (e.g., Cargorouter may not talk to Warehouse directly but must instead use RouterConn to bridge the interaction).
- C2 connectors can interact with both connectors and components.
- C2 requests are upwards and C2 notifications are downwards. This is similar to the layered architecture style with the exception that components of the same layer may not talk to one another (e.g., Warehouse and Vehicle cannot interact in any way; not even through RouterConn or ClockConn).

The C2 architecture was build to increase the ability of run-time dynamic component exchange. This means that if someone wishes to adopt C2 to describe the Cargorouter architecture then one also wishes to ensure that the design and subsequent source code reflect that architecture faithfully.

If, for instance, consistency between the architecture and design are not ensured then there is no way of telling whether desirable architectural properties were carried over into the design. As mentioned above, the C2 architecture supports the dynamic exchange of components. Thus, if the design faithfully reflects the architecture then the design will continue to have that property. It is therefore our goal to provide means of ensuring faithful refinement via consistency checking.

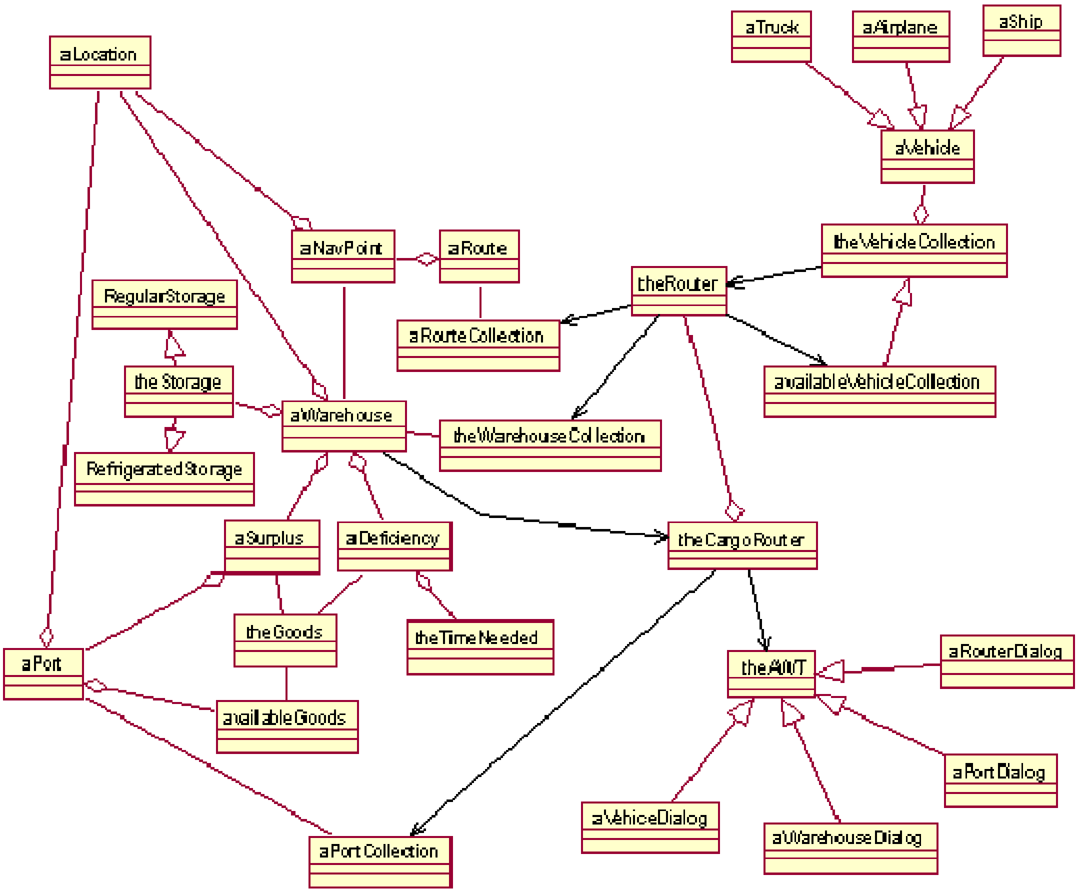


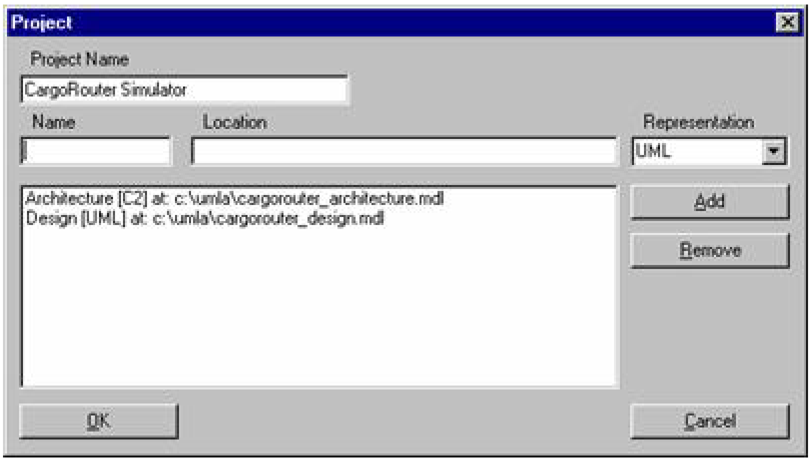
Figure 7.2 – Design in UML class

Figure 7.2 displays the Cargorouter Simulator from a UML class diagram point of view. Although this is a seemingly simple application with only about 30 classes, it is not straightforward to tell whether or not this design is consistent with/conforms to the architecture in Figure 7.1. In order to properly take into account consistency and conformance, we must consider that:

- if the architecture defines some properties (e.g., components) for which there are no associated counterparts in the design, then this may indicate potential conformance mismatches.
- If the design contains properties that contradict architectural ones then this may indicate potential consistency mismatches.

7.3.1 UML/Analyzer and Mismatches

UML/Analyzer can be used to locate mismatches of both consistency and conformance types. In the CargoRouter example it is used to abstract a UML class diagram in such a manner that it closer resembles the C2 architectural diagram.



Project

Project Name: CargoRouter Simulator

Name:

Location:

Representation: UML

Architecture [C2] at: c:\umla\cargorouter_architecture.mdl
Design [UML] at: c:\umla\cargorouter_design.mdl

Figure 7.3 – UML/Analyzer Project Creation

The popup of Figure 7.3 is used to provide the project name as well as the names, locations, and types of representation of all your input files. The default is already set towards the Cargorouter application. The first item shown in the Project list therefore specifies the input file of the C2 architecture cargorouter_architecture.mdl stored in the tool installation folder (same as Figure 7.1) – a Rational Rose model file.

Although Rational Rose and UML do not support C2 other USC works have focused on how UML's extensibility mechanisms can be used to represent C2. This is done via Stereotypes and OCL. The C2 architecture in above mdl file was automatically generated from a tool called SAAGE (also developed at USC) and the architecture can be displayed in and modified with Rational Rose.

The second item shown in the Project list specifies a UML design file (cargorouter_design.mdl) that resembles design from Figure 7.2 and is located in the same folder. Upon pressing OK, UML/Analyzer opens two instances of Rational Rose in order to load and display both models (architecture and design).

Users must choose Download from the Model menu to download architectural and design elements into UML/Analyzer. This may take a few seconds. The UML/Analyzer main window will then display all loaded items. Note that the design is downloaded four times. This is a prototype feature to allow different experimentations. The design copies are called DesignCopy1, DesignCopy2, and DesignCopy3 and are identical with Design).

8. Conclusions

This study has evaluated the availability of solutions that can suit demands and requirements of the INSPIRE Data technical chapter.

The study anyway is far from being exhaustive, mainly for the continuous evolution of technologies, improvement and diffusion of standards and also for the fast changing demands of the market, especially on the business side.

Few considerations should be made:

- Stability is a key issue if services are to be successful. Legacy data and legacy systems will form an important part of any general application, and that stability must include the ability to deal with constant technology and sensor data changes. This leads to the concept that any widely used approach must be based on a "system of systems" and not constrained by a single architecture.
- There is the need to include measures of statistical uncertainty in data (especially for interpolated data) as the need for standardized test data and test algorithms.

Part of the problem may be that the concept of interoperability must be more clearly defined. Interoperability should be achieved at syntax, content, and business level. The main results of initiatives to promote interoperability are at syntax level for the moment. Interoperability issues at business level related to IPR and data policy are not resolved – indeed they are not yet widely recognised.

The following sections of this chapter try to consider also the status of alternative technologies on the interoperability side of data.

Ontologies and semantic Web

The semantic Web and ontologies are gaining more and more importance and in a near future they will be used in dedicated versions of the data infrastructures to organise access to services and (meta)data in a multi-lingual environment.

There is however no consensus on how these technologies may best be used, and further research remains to be done in this area. Although several EC projects in different EC DGs have performed research in this area, it was felt by some participants that exchange of research results among projects up to now is not sufficient to ensure wide endorsement of the proposed solutions.

Grid technology and SOA

The expected convergence between grid technology and SOA is taking longer than hoped, and it is not progressing at the desired pace.

ESA presented its use of Grid to solve real problems in the EO and physics domain, based on a new paradigm of taking the processing to the data and not vice-versa. This has clear benefits given the amount of data involved. The initial results appeared to be encouraging.

As yet, there are no agreed GRID standards, and the challenge is not only how to deal with massive volumes of distributed data but also how to move to a new paradigm [13]

Concluding, the present list of available solutions, platforms and software packages will need more time to adjust to the INSPIRE final requirements and get in tune with the involved standards.

Annex A: Acronyms

Acronyms for organisations

BADC	British Atmospheric Data Centre
BODC	British Oceanographic Data Centre
DMTF	Distributed Management Task Force
DIS	Draft International Standard
DT	INSPIRE Drafting Team
EDR	Environmental Data Registry
EPA	US Environmental Protection Agency
ESDI	European Spatial Data Infrastructure
EU	European Union
GCM	Generic Conceptual Model (INSPIRE D2.5)
INSPIRE	Infrastructure for Spatial InfoRmation in Europe
IR	INSPIRE Implementing Rules
ISO	International Standards Organization
LMO	Legally Mandated Organisation
NERC	Natural Environment Research Council
OASIS	Organization for the Advancement of Structured Information Standards
OGC	Open Geospatial Consortium, also referred to as OpenGIS®
OMG	Object Management Group
OSI	Open Systems Interconnection (ISO/OSI)
RISE	Reference Information Specifications for Europe
SDI	Spatial Data Infrastructure
SDIC	Spatial Data Interest Community
W3C	World Wide Web Consortium
WS-I	Web Services Interoperability Organization

Acronyms for technologies (protocols, languages, etc.)

ASL	Action Specification Language
CSL	Conceptual Schema Language
CSML	Climate Science Modelling Language
CWM	Common Warehouse Metamodel
EDOC	Enterprise Distributed Object Computing
HTTP	Hyper Text Transfer Protocol
MDA	Model Driven Architecture (http://www.omg.org/mda/)
MOF	Meta-Object Facility (http://www.omg.org/mof/)
MOLES	Metadata Objects for Links in Environmental Science
NDG	NERC DataGrid (http://ndg.badc.rl.ac.uk/)
ORCHESTRA	Open Architecture and Spatial Data Infrastructure for Risk Management
OCL	Object Constraint Language
OWL	Web Ontology Language (http://www.w3.org/TR/owl-features/)
PDM	Platform Definition Model
PIM	Platform-Independent Model
PSM	Platform-Specific Model
QVT	Queries/Views/Transformations
RDF	Resource Description Framework (http://www.w3.org/TR/rdf-schema/)
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SPEM	Software Process Engineering Metamodel
UML	Unified Modelling Language
WCF	Windows Communication Foundation
WS	Web Service
XMI	XML Metadata Interchange
XML	eXtensible Markup Language

Annex B: References

The following table lists the sources referenced in the present document.

	Title	Author	Notes
1	Metadata State of Play	TXT e-Solutions	First survey on Inspire Metadata
2	Services State of Play	TXT e-Solutions	Second survey on Inspire Services
3	INSPIRE Work Programme Transposition Phase 2007-2009	INSPIRE	Available at: http://www.ec-gis.org/inspire/reports/transposition/INSPIRE_IR_WP2007_2009_en.pdf
4	Generic Conceptual Model	INSPIRE	Available at: http://www.ec-gis.org/inspire/reports/ImplementingRules/inspireDataspecD2_5v2.0.pdf
5	Methodology for the development of data specifications	INSPIRE	Available at: http://www.ec-gis.org/inspire/reports/ImplementingRules/inspireDataspecD2_6v2.0.pdf
6	INSPIRE Drafting teams Data Specifications	Clemens Portele – OGC	Available at http://www.ec-gis.org/inspire/reports/13ecgigis/4_inspire_progress_portele_2.pdf
7	Spatial Data Infrastructures in Australia, Canada and the United States	Margaret Hall	Available at: http://www.ec-gis.org/inspire/reports/stateofplay/rpAUCAUSAv5.pdf
8	Model Driven Architecture - Test Methods and Tools	Renas Reda Yusuf Tözmal	Document available at: http://www.bth.se
9	OWL Web Ontology Language Overview	W3C	Available at: http://www.w3.org/TR/2004/REC-owl-features-20040210/
10	Compliance Test Language (CTL)	OGC	Available at: http://portal.opengeospatial.org/files/?artifact_id=16860
11	Mapping UML to GML Application Schemas: ShapeChange - Architecture and Description	C. Portele Interactive Instruments	Document available at: http://www.interactive-instruments.de/ugas/ShapeChange.pdf
12	Mapping UML to GML Application Schemas: Guidelines and Encoding Rules	C. Portele Interactive Instruments	
13	Joint Operability Workshop Report	Y. Coene, R. Gasser	ftp://ftp.cordis.europa.eu/pub/ist/docs/environment/sise-workshop-report_en.pdf
14	Supporting MDA with Executable UML	Kennedy Carter	Available at: http://www.kc.com/xuml.php
15	OMG Unified Modelling Language Specification	OMG	ftp://ftp.omg.org/pub/docs/ad/99-06-08.pdf

European Commission

EUR 23045 EN – Joint Research Centre – Institute for Environment and Sustainability
Title: Data State of Play, Compliance Testing and Interoperability Checking
Author(s): S. Viganó, M. Millot
Luxembourg: Office for Official Publications of the European Communities
2007 – 88 pp. – 21 x 29.7 cm
EUR – Scientific and Technical Research series – ISSN 1018-5593
ISBN 978-92-79-07817-0
DOI 10.2788/55339

Abstract

The document an inventory of existing solutions for compliance testing and interoperability checking for data taking into account the draft INSPIRE data specifications conceptual model (D2.5), the first draft of the INSPIRE Methodology for the development of data specifications (D2.6) and the first draft of the data Specifications Guidelines for the encoding of spatial data (D2.7). Even if the emphasis is on spatial and geographical data, the document investigates applicable solutions outside the geographical Information System domain, with a particular attention paid to checking compliance with “application schemas” as defined in the previously mentioned documents.

How to obtain EU publications

Our priced publications are available from EU Bookshop (http://bookshop.europa.eu), where you can place an order with the sales agent of your choice.

The Publications Office has a worldwide network of sales agents. You can obtain their contact details by sending a fax to (352) 29 29-42758.

The mission of the JRC is to provide customer-driven scientific and technical support for the conception, development, implementation and monitoring of EU policies. As a service of the European Commission, the JRC functions as a reference centre of science and technology for the Union. Close to the policy-making process, it serves the common interest of the Member States, while being independent of special interests, whether private or national.

LN NA 23045 EN C

JRC
EUROPEAN COMMISSION

Publications Office
Publications.eu.int

ISBN 978-92-79-07817-0

9 789279 078170