REGNET

Cultural Heritage in REGional NETworks

Deliverable D5

The REGNET – System Version 1

Project acronym	REGNET	**Contract nr.**	IST-2000-26336
Type and Number	D5 The REGNET-System, Version 1		
Work package	WP2: Implementation of the System and Preparation of the Demonstration		
Task	T2.2: System Implementation (1. Version)		
Date of delivery	2002-04-19		
Code name	RN_D5v01	**Version** 01 draft ☐ final ☑	
Objective	Report		
Distribution Type	Restricted		
Authors (Partner)	AIT, SR, IMAC, TARX, MOT, ZEUS, SI, CERT, VALT		
Abstract	This deliverable contains the results of Work Area B (Platform Engineering) related to Work Package 2. It describes the actual work done in this area following the results included in deliverable D2 and is related to task 2.2.		
Keywords List	Regnet System, Prototype Implementation		

Table of Contents

Executive summary 5

1 Introduction 7

1.1 Iterations 7

1.2 Artefacts 7
1.2.1 Elaboration (iteration 1) 7
1.2.1.1 Objectives 7
1.2.1.2 Essential activities 8
1.2.1.3 Evaluation Criteria 8
1.2.1.4 Artefacts 9
1.2.2 Construction phase (iterations 2,3) 9
1.2.2.1 Objectives 9
1.2.2.2 Essential Activities 10
1.2.2.3 Evaluation Criteria 10
1.2.2.4 Artefacts 10

2 Core System Design 11

2.1 Architecture overview 11

2.2 Interfaces 11
2.2.1 REGNET Portal 11
2.2.1.1 Data Generation 11
2.2.1.2 Search and Retrieval 12
2.2.1.3 E-Business 13
2.2.2 Cultural Heritage Data Management 14
2.2.2.1 Repository Management 14
2.2.2.2 Reference System 15
2.2.3 E-Business data management 16
2.2.3.1 Catalogue management 16
2.2.3.2 Procurement and Delivery 17
2.2.4 Knowledge Base Access 18
2.2.5 Electronic publisher 20
2.2.5.1 Interface PublicationObjectHome 20
2.2.5.2 Interface PublicationObject 20

2.3 Tools and Platform 22

3 Prototypes 24

3.1 Portal + J2EE/SOAP + PHP/SOAP 24
3.1.1 Portal to J2EE through SOAP prototype 24
3.1.1.1 Scope 24
3.1.1.2 Architecture 24
3.1.1.3 Conclusion 25

3.2 EbXML Message 26
3.2.1 Scope 26
3.2.2 Architecture 27
3.2.2.1 JAXM 27
3.2.2.2 JAXM API 27
3.2.3 Conclusion 28

3.3 TopicMap creation 28
3.3.1 Scope 28
3.3.1.1 XML Topic Maps 28
3.3.1.2 Topic Map Generator Tool 28
3.3.2 Architecture 29
3.3.2.1 Topic Map Generator Tool Architecture 29

3.3.3 Conclusion 31

3.4 TopicMap visualisation 31
3.4.1 Scope 31
3.4.1.1 VRML Definition 31
3.4.1.2 Using VRML to display Topic Maps 32
3.4.1.3 The pros and cons of using the Topic Map To VRML tool 32
3.4.2 Architecture 32
3.4.3 Conclusion 33

3.5 Search and Retrieval 33
3.5.1 Scope 33
3.5.2 Architecture 34
3.5.3 Conclusion 35

3.6 CatXML 35
3.6.1 Scope 35
3.6.2 Architecture 35
3.6.3 Conclusion 36

3.7 Portal 37
3.7.1.1 Main access point 37
1.1.1.1 User registration and authentication 37
1.1.1.2 User and terminal profiles 37
1.1.1.3 Session 37
1.1.1.4 Internationalisation 37
3.7.2 Scope 38
3.7.3 Architecture 38
3.7.3.1 Common Business Logic 39
3.7.4 WEB and WAP Prototypes 40
3.7.5 Conclusion 40

3.8 Publication 41
3.8.1 Scope 41
3.8.2 Architecture 42
3.8.3 Conclusion 43

4 Design Model 44

4.1 Purpose 44

4.2 REGNET Portal 44
4.2.1 Data Generation 44
4.2.1.1 Introduction 44
4.2.1.2 Design model hierarchy 47
4.2.1.3 Diagrams of the design model 48
4.2.1.4 The REGNET off line data entry system 48
4.2.2 Search and Retrieval 54
4.2.2.1 Introduction 54
4.2.2.2 Design model hierarchy 55
4.2.2.3 Diagrams of the design model 56
4.2.3 E-Business 56
4.2.3.1 Introduction 56
4.2.3.2 Design model hierarchy 60
4.2.3.3 Diagrams of the design model 61

4.3 Cultural heritage data management 65
4.3.1 Repository management 65
4.3.2 Reference system 65
4.3.2.1 Introduction 65
4.3.2.2 Design model hierarchy 67
4.3.2.3 Diagrams of the design model 69

4.4 E-Business data management 70

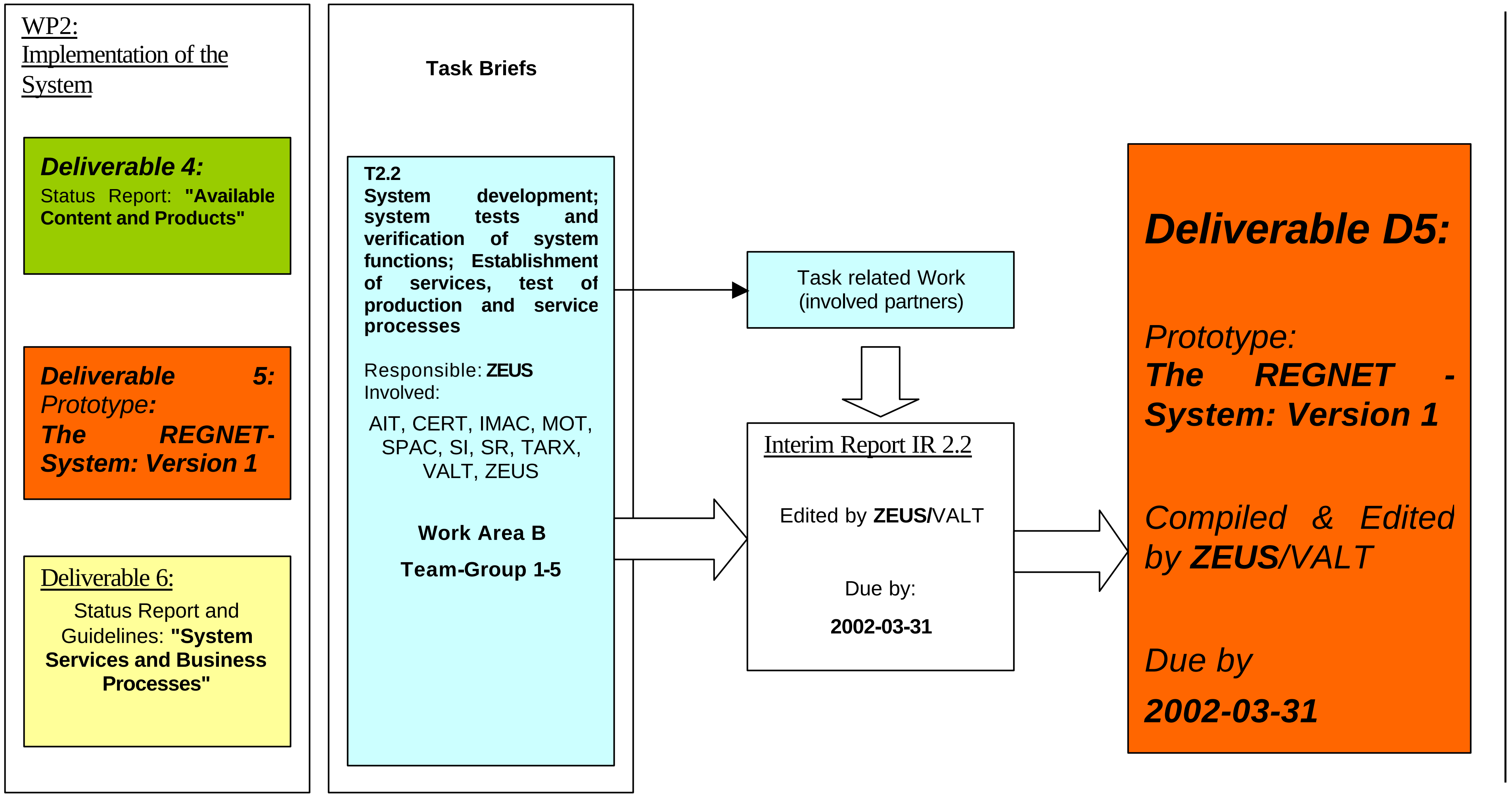
4.4.1 Catalogue management 70
4.4.1.1 Introduction 70
4.4.1.2 Design model hierarchy 74
4.4.1.3 Diagrams of the design model 74
4.4.2 Procurement 78
4.4.2.1 Introduction 78
4.4.2.2 Design model hierarchy 78
4.4.2.3 Diagrams of the design model : REGISTRATION 81
4.4.2.4 Diagrams of the design model : CATALOGUE 85
4.4.2.5 Diagrams of the design model : TRADE 90
The trade is added to the list of the trades of the buyer's account. 92
4.4.2.6 Diagrams of the design model : REPORTS 93
4.4.3 Delivery 94
4.4.3.1 Introduction 94
4.4.3.2 Design model hierarchy 94
4.4.3.3 Diagrams of the design model : WAREHOUSE CALCULATOR 96
4.4.3.4 Diagrams of the design model : SHIPMENT 99
4.5 Knowledge Base Access 105
4.5.1 Introduction 105
4.5.2 Design model hierarchy 105
4.5.3 Diagrams of the design model 106
4.5.3.1 UML diagram of the FilePacket class (part of org.regnet.ontology package) 106
4.5.3.2 UML diagram of the FileReceiver class (part of org.regnet.ontology package) 106
4.5.3.3 UML diagram of the FileSender class (part of org.regnet.ontology package) 107
4.5.3.4 UML diagram of the FileSendInitialize class (part of org.regnet.ontology package) 107
4.5.3.5 UML diagram of the ReceiveFile class (part of org.regnet.ontology package) 109
4.5.3.6 UML diagram of the RemoteFileSendInitialize class (part of org.regnet.ontology package) 109
4.5.3.7 UML diagram of the RemoteReceiver class (part of org.regnet.ontology package) 109
4.6 Electronic publisher 110
4.6.1 Introduction 110
4.6.2 Design model hierarchy 112
4.6.3 Diagrams of the design model 113
5 Implementation Model 116
5.1 Component diagram 116
5.2 Deployment of components 117
6 Deployment Plan 118
6.1 Deployment Planning 118
6.1.1 Responsibilities 118
6.1.2 Schedule 119
6.2 Resources 119
6.2.1 Facilities 119
6.2.2 Hardware 120
6.2.3 The Deployment Unit 120
6.2.3.1 Support Software 120
6.2.3.2 Support Documentation 121
6.2.3.3 Support Personnel 121
6.3 Training 121
7 Glossary 122
8 Bibliography and References 134
Figures 135
List of Tables 137

Executive summary

This document includes the activities from work package 2 that have been carried out within Work Area B (Platform Engineering) of the REGNET project. The main emphasis of the specific document is focused on providing a complete overview of the first version of the REGNET System and the functionalities that are supported by it.

In the first part of the document there is a clear description of the processes that were followed in order to achieve the specific results. This process is based on Unified Process as specified in Work-Package 1.

PLATFORM ENGINEERING



WP2:
Implementation of the System

Deliverable 4:
Status Report: **"Available Content and Products"**

Deliverable 5:
Prototype:
The REGNET-System: Version 1

Deliverable 6:
Status Report and Guidelines: **"System Services and Business Processes"**

Task Briefs

T2.2
System development; system tests and verification of system functions; Establishment of services, test of production and service processes

Responsible: **ZEUS**
Involved:
AIT, CERT, IMAC, MOT, SPAC, SI, SR, TARX, VALT, ZEUS

Work Area B
Team-Group 1-5

Task related Work
(involved partners)

Interim Report IR 2.2

Edited by **ZEUS**/VALT

Due by:
2002-03-31

Deliverable D5:

Prototype:
The REGNET - System: Version 1

*Compiled & Edited by **ZEUS**/VALT*

Due by
2002-03-31

1 Introduction

This document contains artefacts from the task 2.2 "Implementation of the System and Preparation of Content".

As far as process used for Regnet project is based on Unified Process, this document contains iterations and artefacts description.

1.1 Iterations

This task has been split into 3 iterations of 2 months. According to UP the first one is relevant to the ***elaboration*** phase, the second and the third one are relevant to ***construction*** phase.

The inception phase has been finished and elaboration phase has begun in the previous work-package (WP1). So some artefacts are part of deliverables D2 "specification and state of the art".

<u>Fist iteration</u>: Architecture validation

According to the technical risks identified in the previous work-package (WP1), this iteration deals with these validation. The following prototypes has been developed and are described in next part:

- Communication:
  - APACHE/JETSPEED Portal with others J2EE based Java modules through SOAP.
  - APACHE/JETSPEED Portal with others PHP based modules through SOAP.
- EbXML messages communication for B2B interaction.
- TopicMap generation and visualisation.
- Search and retrieval.
- CatXML standard for eBusiness catalogue management.
- Technology validation for Portal based on Apache/Jetspeed and Wireless communication.
- Publisher technology validation.

<u>Second iteration</u>: Portal

This iteration deals with the development of a complete Portal connected to appropriate modules. It is based on some development made in the first iteration complete with Data Generation, EShop and User Interface elaboration.

<u>Third iteration</u>: Fist version

This iteration complete the previous one in order to produce the first version of the Regnet system according to specification established in WP1.

During this iteration, already available modules are complete, Procurement and eBusiness Data Management systems are produced and databases are populated.

1.2 Artefacts

Artefacts from this task are detailed below.

1.2.1 Elaboration (iteration 1)

1.2.1.1 <u>Objectives</u>

The goal of the elaboration phase is to baseline the architecture of the system to provide a stable basis for the bulk of the design and implementation effort in the construction phase. The architecture evolves out of a consideration of the most significant requirements (those that have a great impact on the architecture of the system) and an assessment of risk. The stability of the architecture is evaluated through one or more architectural prototypes.

The primary objectives of the elaboration phase include:

- To ensure that the architecture, requirements and plans are stable enough, and the risks sufficiently mitigated to be able to predictably determine the cost and schedule for the completion of the development. For most projects, passing this milestone also corresponds to the transition from a light-and-fast, low-risk operation to a high cost, high-risk operation with substantial organizational inertia.
- To address all architecturally significant risks of the project
- To establish a baselined architecture derived from addressing the architecturally significant scenarios, which typically expose the top technical risks of the project.
- To produce an evolutionary prototype of production-quality components, as well as possibly one or more exploratory, throw-away prototypes to mitigate specific risks such as:
    - design/requirements trade-offs
    - component reuse
    - product feasibility or demonstrations to investors, customers, and end-users.
- To demonstrate that the baselined architecture will support the requirements of the system at a reasonable cost and in a reasonable time.
- To establish a supporting environment.

In order to achieve these primary objectives, it is equally important to set up the supporting environment for the project. This includes creating a development case; create templates, guidelines, and setting up tools.

1.2.1.2 Essential activities

- **Defining, validating and baselining the architecture** as rapidly as practical.
- **Refining the Vision**, based on new information obtained during the phase, establishing a solid understanding of the most critical use cases that drive the architectural and planning decisions.
- **Creating and baselining detailed iteration plans for the construction phase**.
- **Refining the development case and putting in place the development environment**, including the process, tools and automation support required to support the construction team.
- **Refining the architecture and selecting components**. Potential components are evaluated and the make/buy/reuse decisions sufficiently understood to determine the construction phase cost and schedule with confidence. The selected architectural components are integrated and assessed against the primary scenarios. Lessons learned from these activities may well result in a redesign of the architecture, taking into consideration alternative designs or reconsideration of the requirements.

At the end of the elaboration phase is the second important project milestone, the **Lifecycle Architecture Milestone**. At this point, you examine the detailed system objectives and scope, the choice of architecture, and the resolution of the major risks.

1.2.1.3 Evaluation Criteria

- The product Vision and requirements are stable.
- The architecture is stable.
- Executable prototypes have demonstrated that the major risk elements have been addressed and have been credibly resolved.
- The iteration plans for the construction phase are of sufficient detail and fidelity to allow the work to proceed.

- The iteration plans for the construction phase are supported by credible estimates.
- All stakeholders agree that the current vision can be met if the current plan is executed to develop the complete system, in the context of the current architecture.
- Actual resource expenditure versus planned expenditure is acceptable.

The project may be aborted or considerably re-thought if it fails to reach this milestone.

1.2.1.4 Artefacts

- **Prototypes**: One or more executable architectural prototypes have been created to explore critical functionality and architecturally significant scenarios.
- **Risk list**: Updated and reviewed. New risks are likely to be architectural in nature, primarily relating to the handling of non-functional requirements.
- **Project-specific templates**: The document templates used to develop the document artefacts.
- **Tools**: The tools used to support the work in Elaboration are installed.
- **Software architecture document**: Created and baselined, including detailed descriptions for the architecturally significant use cases (use-case view), identification of key mechanisms and design elements (logical view), plus definition of the process view and the deployment view if the system is distributed or must deal with concurrency issues. *This artefact has been done in WP1.*
- **Design Model**: Defined and baselined. Use-case realizations for architecturally significant scenarios have been defined and required behaviour has been allocated to appropriate design elements. Components have been identified and the make/buy/reuse decisions sufficiently understood to determine the construction phase cost and schedule with confidence. The selected architectural components are integrated and assessed against the primary scenarios. Lessons learned from these activities may well result in a redesign of the architecture, taking into consideration alternative designs or reconsideration of the requirements.
- **Data Model**: Defined and baselined. Major data model elements (e.g. important entities, relationships, tables) defined and reviewed.
- **Implementation model**. Initial structure created and major components identified and prototyped.
- **Software development plan**: Updated and expanded to cover the Construction and Transition phases.
- **Use case model**: A use-case model (approximately 80% complete)—all use cases having been identified in the use-case model survey, all actors having been identified, and **most** use-case descriptions (requirements capture) have been developed.

1.2.2 Construction phase (iterations 2,3)

1.2.2.1 Objectives

The goal of the construction phase is on clarifying the remaining requirements and completing the development of the system based upon the baselined architecture. The construction phase is in some sense a manufacturing process, where emphasis is placed on managing resources and controlling operations to optimise costs, schedules, and quality. In this sense the management mindset undergoes a transition from the development of intellectual property during inception and elaboration, to the development of deployable products during construction and transition.

The primary objectives of the construction phase include:

- Minimizing development costs by optimising resources and avoiding unnecessary scrap and rework.
- Achieving adequate quality as rapidly as practical

- Achieving useful versions (alpha, beta, and other test releases) as rapidly as practical
- Completing the analysis, design, development and testing of all required functionality.
- To iteratively and incrementally develop a complete product that is ready to transition to its user community. This implies describing the remaining use cases and other requirements, fleshing out the design, completing the implementation, and testing the software.
- To decide if the software, the sites, and the users are all ready for the application to be deployed.
- To achieve some degree of parallelism in the work of development teams. Even on smaller projects, there are typically components that can be developed independently of one another, allowing for natural parallelism between teams (resources permitting). This parallelism can accelerate the development activities significantly; but it also increases the complexity of resource management and workflow synchronization. A robust architecture is essential if any significant parallelism is to be achieved.

1.2.2.2 Essential Activities

- Resource management, control and process optimisation
- Complete component development and testing against the defined evaluation criteria
- Assessment of product releases against acceptance criteria for the vision.

At the Initial Operational Capability Milestone, the product is ready to be handed over to the Transition Team. All functionality has been developed and all **alpha** testing (if any) has been completed. In addition to the software, a user manual has been developed, and there is a description of the current release.

1.2.2.3 Evaluation Criteria

The evaluation criteria for the construction phase involve the answers to these questions:

- Is this product release stable and mature enough to be deployed in the user community?
- Are all the stakeholders ready for the transition into the user community?
- Are actual resource expenditures versus planned still acceptable?

Transition may have to be postponed by one release if the project fails to reach this milestone.

1.2.2.4 Artefacts

- **The Regnet system**: The executable system itself, ready to begin "beta" testing.
- **Deployment plan**: Initial version developed, reviewed and baselined.
- **Implementation model**: Expanded from that created during the elaboration phase; all components created by the end of the construction phase.
- **Test model**: Tests designed and developed to validate executable releases created during the construction phase.
- **Training materials**: User Manuals and other training materials. Preliminary draft, based on use cases.
- **Tools**: The tools used to support the work in Construction are installed.
- **Data Model**: Updated with all elements needed to support the persistence implementation (e.g. tables, indexes, object-to-relational mappings, etc.)

2 Core System Design

2.1 Architecture overview

The Software Architecture Document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system.

As explained in deliverable D2 from WP1, the architecture is based on Web Services approach.

2.2 Interfaces

This part explains interfaces of each Regnet building blocks. That is all services provides by a sub-system to other sub-systems.

2.2.1 REGNET Portal

The Regnet Portal had been developed in Java and making use of a Java user customisable portal tool named Jakarta Jetspeed. Jetspeed is an open source tool that provides some very use features such as for example the support for HTML and WML, the local caching of remote content, the User, group, role and permission administration.

Starting form such tool, to obtain a Portal showing all the available tools and applications had been necessary both to realize the integration of the different components and also to register in the right way the components inside the Portal tool.

The Jetspeed Portal provides a framework that allows extensions to be plugged into the portal. These extensions are called *portlets*. In the same way that a servlet is an application within a Web server, a portlet is an application within the portal.

The Regnet components to be integrated into the Portal were substantially of two different kinds:

- ” external and independent” by the Portal;
- “strongly integrated” into the Portal.

The external and independent components provide their own layout (HTML and WML), are easily integrable into the Portal, but allow few processing at Portal level.

Instead, the strongly integrated components provide their own XML interface and the Portal has to manage their layouts (HTML and WML). They require a more strictly collaboration between the applications and the portal developers during the interface definition phase but allow an ease modification of the presentation layer and processing at Portal level.

The “external and independent” components have been integrated into the Portal through simple HTTP Forward while the “strongly” components have been integrated making use of SOAP. In particular in order to integrate the strongly integrated components had been necessary to know the interfaces offered by each component and to developed the suitable SOAP client.

Instead in order to facilitate the definition of the components as portlets of the Portal has been developed a “Regnet Portlet Template”. Thanks to this template to have a portlet representing the component to be integrated is sufficient to define an application property file and follow some very easy steps specified into the Portal Prototype Installation Guide.

2.2.1.1 Data Generation

A 2 step approach has been followed to achieve an integrated version of the Data Generation component into the portal. In the first step an independent Data Input tool was devleoped. This tool acts as prototype and the graphical user interfaces as mockup for the 2nd version. The Data Input tool can be used also as standalone system for single a singel database.

This Data Generation prototype is included into the portal prototype through a link and belongs to the group of "external and independent" components as described above.

The ongoing second step is to build a "strongly integrated" Data Generation component. This user interfaces of this component will be based on the prototype. The component can benefit from the user management and terminal profiling from the portal framework. The component is based on Servlet technology and uses XSL transformation for presentation issues.

The Data Generation component has connection to two other Regnet components:

1. To the Cultural Heritage Data Management component
    a) The DataGenerationService is an Web service located in the Reference System which is dedicated to serve the Data Generation component in the portal. This Web service supports all functionalities needed to manage single records in an repository.
    b) If there are additional components used in the Repository Management component in future an interface for the Data Generation component has to be provided to support the input of digital surrogates of real world objects (images, videos, audiofiles, ...).

2. To the Ontology System

    Data Input has to support different data fields for different repositories. The data structure of an repository (as XML-Schema) and the input masks (as XML stylesheets) for each repository are stored in the Ontology System.

    Also user profiles are stored in the Ontology System. This data is very important for the Data Generation component as creating, editing and deleting data has to support user identification.

2.2.1.2 Search and Retrieval

The Search and Retrival system is strongly integrated into the portal. The to main task of the component are to build a query interface and to present the results for a query. The component is based on Servlet technology and the presentation tasks are driven by XSL stylesheet transformation.

The real search task is delegated to the Reference Systems SearchService. The connection to this Web service is based on SOAP.

The query interface has 2 levels: the Simple Search and the Advanced Search level.

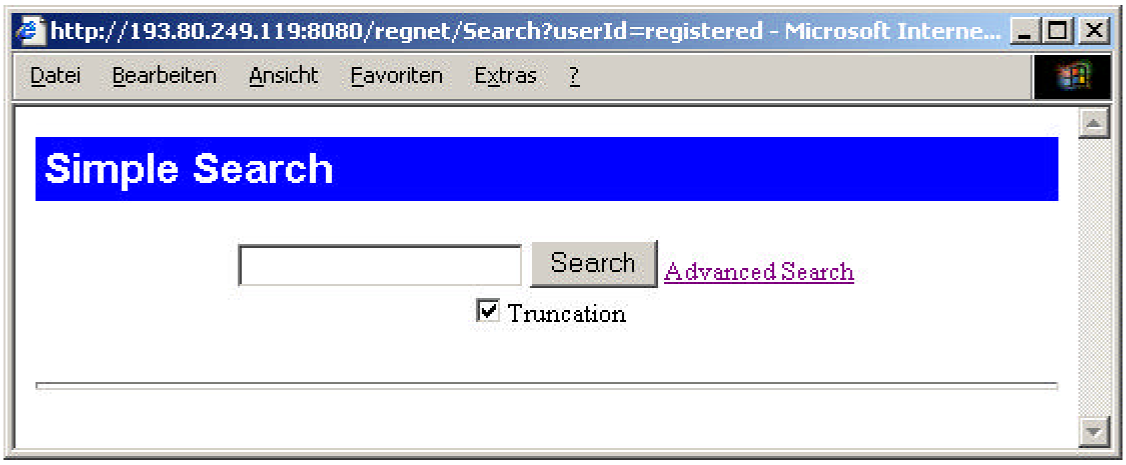


Figure 1: Simple Search GUI

The Simple Search page only provides a single text input line and the option to have your text automaticaly truncated while searching. All the other search limits are set automatically in this level. The limits are set very smuth (all databases are searched, fulltext index is used) to help unskilled users to find something.

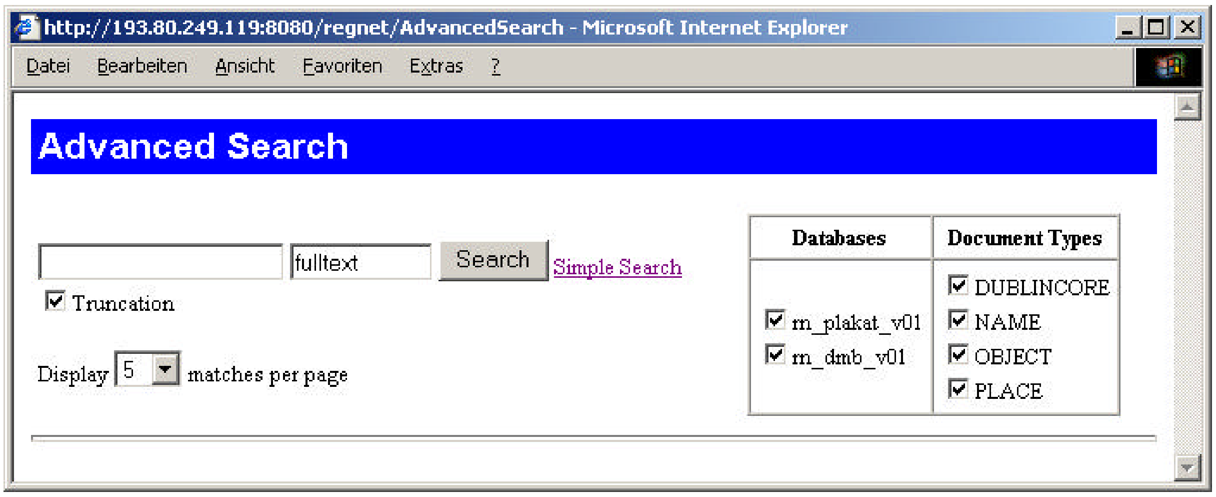

http://193.80.249.119:8080/regnet/AdvancedSearch - Microsoft Internet Explorer
Datei Bearbeiten Ansicht Favoriten Extras ?
Advanced Search
fulltext
Search
Simple Search
Truncation
Display 5 matches per page
Databases
m_plakat_v01
m_dmb_v01
Document Types
DUBLINCORE
NAME
OBJECT
PLACE

Figure 2: Advanced Search GUI

The Advance Search page provides more options to limit the users search. The user can select the databases and documenttypes for searching and in which scope he wants to search (fulltext or some other scope). This user interface represents the first version and will be updated according to user feedback. Also there will be more and more databases integrated into the system. This will be an issue for the interface also to display the search limits in an appropriate and understandable way.

As all the presentation (for query interfaces and result presentation) are done via stylesheet transfomation, changing the user interface is made much easier. There are no changes to be made to the code but only to be developed new stylesheets. For further project phases there can even be an option for the users which stylesheet they want to use for building up the user interface or they can even make their own stylesheets and integrate them into the system.

2.2.1.3 E-Business

The basic concept of the ebusiness subnode is based on the ebxml specifications. Therefore our approach establishes the ebxml configuration of the e-business subsystem, which reflects the special needs of the REGNET project.

Our demonstrator is developed using PHP as developing language. The results are communicating with the portal and the interface of it.

It is important to distinguish the two main components of the E-Business subsystem and these are the systems that implement the e-shop and the B2B services.

A) E-SHOP

The basis is that the E-Business subsystem is responsible for the management and presentation of the different catalogue products and services by the content providers. All this stored information is available to the end-users using RPN queries in order to access the distributed information. More specifically we developed the following:

- Z39.50 server. Special software meant for the receiving queries (RPN) from the z39.50 client and transfers them into the SQL queries to the relational database. Receives the sets of data, as a result of searching from the relational database, convert them into the XML format and sent via the z39.50 protocol to the z39.50 client;

- Z39.50 client based on the PHP/YAZ module. Meant for the receiving the text queries from the user then convert them into the RPN queries and sent them to the chosen z39.50 servers. Receives the results from the z39.50 servers, parse them from the XML format and presents these results to the presentation level.

B) B2B

B2B extension was built on SOAP technology. The main aspect is the independence of the B2B scenario implementation, which just provides input and output functionalities based on the ebxml specifications. Special repositories are used in order to describe the provided services of the REGNET group and the different aspects of the B2B scenario.

This (b2b) subsystem interconnects with PCM (product catalogue management), in order to find the necessary products to buy and the necessary description of the provided services.

2.2.2 Cultural Heritage Data Management

The Cultural Heritage Data Management is based on the use of the TEXTML-Server from IXIASOFT. Nevertheless the interfaces to this subsystem are designed to be open to integrate other databases into the Regnet System.

The TEXTML-Server is only available for Windows platform and the access is done via an API based on COM technology. Therefore the new .NET Framework was used to establish the connections to the database. The interoperability with the Regnet System, which is mainly implemented using the J2EE platform, is guaranteed by providing a SOAP interface to the Cultural Heritage Data Management component.

2.2.2.1 Repository Management

The Repository Management component is responsible for managing the storage and retrieval of all digital surrogates of primary ('real world') objects (e.g. images, audivisual and audio files, text files, ...).

As there can be large files (e.g. movies or high resolution pictures) and also there may be many files (e.g. more than a million images) the storage of these files is done on a dedicated Linux machine. Depending on how the digital surrogates should be used there are 2 principal types of storage and retrieval:

a) Storage directly on the filesystem:
   - storage can be a simple copy process from any other storage medium or can be done via file transfer (FTP)
   - a webserver enables retrieval of the stored files.
   - appropriate for small and medium resolution pictures, text files

b) Storage via an specialized application:
   - high resolution pictures, large media files or data which needs special protection mechanisms have to be stored in specialized applications to provide appropriate access mechanisms.

ad a)

The images from one database are already stored and available at http://csc002.cscaustria.at/plakat/ . As Web server Apache v1.3.20 is used. Images might be stored in different resolutions (e.g. full resolution and thumbnail) in different directories. Linkage between the pictures and data associated to them is done via the meta data descriptions in the Reference System.

ad b)

An image server (XLimage®server 2.0) was evaluated. This image server is capable of storing high resolution pictures in an proprietary format which allows very fast access to parts (zooming) of the

pictures and to thumbnails. Also on-line watermarking is possible to protect the images against unauthorized access.
Other applications may optimise access to resources via streaming technology or may even provide enhanced search capabilities within pictures, sounds or movies (e.g. Visual Information Retrieval).

2.2.2.2 Reference System

The Reference System provides access to the databases holding cultural heritage meta data. It is split into to main parts:

a) Data Generation Service (provides data generation functionalities)

b) Search Service (provides search and retrieval functionalities)

Both subcomponents are accessible through an ASP.NET web service. The code behind the web service is done in C#. This implies that the .NET environment must be installed on the machine running the web services described below. Also the access to a TEXTML-Server is restricted to the machine running the web service. The interfaces provided for the 1st prototype are:

for a)

- Version
  Return Version of the Web Servi ce.

- ListRepositories
  Return a list of repositories connected to the Data Generation Service.

- ListDoctypes
  Returns a list of document types for each repository.

- Edit
  Locks a record for one user. True if successful, false if record is already locked by another user or an error occurred. One can use this method instead of EditRecord if the record data is has been retrieved by the use of another component.

- EditRecord
  Same as Edit but returns the record data instead of boolean. Record data if successful, "" if record is already locked by another user or an error occurred.

- GetLockUser
  Get identifier from user that has currently locked the record.

- Save
  Saves the record and removes the lock from this record. True if successful, false if record is not locked for this user or an error occurred.

- Delete
  Deletes a record. The record must be unlocked or locked for this user. True if successful, false if record is already locked by another user or an error occurred.

The Data Generation Service is available via:

http://193.80.249.119/regnet/data/DataGenerationService.asmx

There is a SOAP specification for each method available when browsing the URL with and ordinary web browser. The full WSDL specification of the web service can be obtained via:

http://193.80.249.119/regnet/data/DataGenerationService.asmx?WSDL

Version, ListRepositories, ListDoctypes and GetLockUser are "helper" method for building up the user interface. The important methods of this web service are Edit, Save and Delete which allow to manipulate single records in one database.

for b)

- Version
  Return Version of the Web Service.
- ListRepositories
  Return a list of repositories connected to the Search Service.
- ListDoctypes
  Returns a list of document types for each repository.
- ListAccessProfiles
  Returns a list of index definitions (as XML document) for each repository.
- ListQueryTypes
  Lists the query types (query formats) that can be used with this service.
- SearchRetrieve
  The main method of the Search Service. Returns query hits and the records found respectively.
- GetRecords
  Returns the records specified in a record list filtered by the optional doctypes list.

The Search Service is available via http://193.80.249.119/regnet/search/SearchService.asmx. There is a SOAP specification for each method available when browsing the URL with and ordinary web browser. The full WSDL specification of the web service can be obtained via http://193.80.249.119/regnet/search/SearchService.asmx?WSDL .

Only the last 2 methods are actually returning records (in XML format). The first 5 methods provide information about the Search Service. They can be used to build up a user interface and to build up a network of connected Search Services.

2.2.3 E-Business data management

2.2.3.1 Catalogue management

Product Catalogue Management (PCM) is an information system, which allows the users (participants) to manage their catalogues. Their catalogues contain information about the warehouses and items in them.

The most important aspect of the PCM is usage of the SOAP protocol in the interconnections between the central (administrative database) and the remote (distributed) catalogues. Scheme of the interconnections between the parts of the PCM is presented on the following figure.

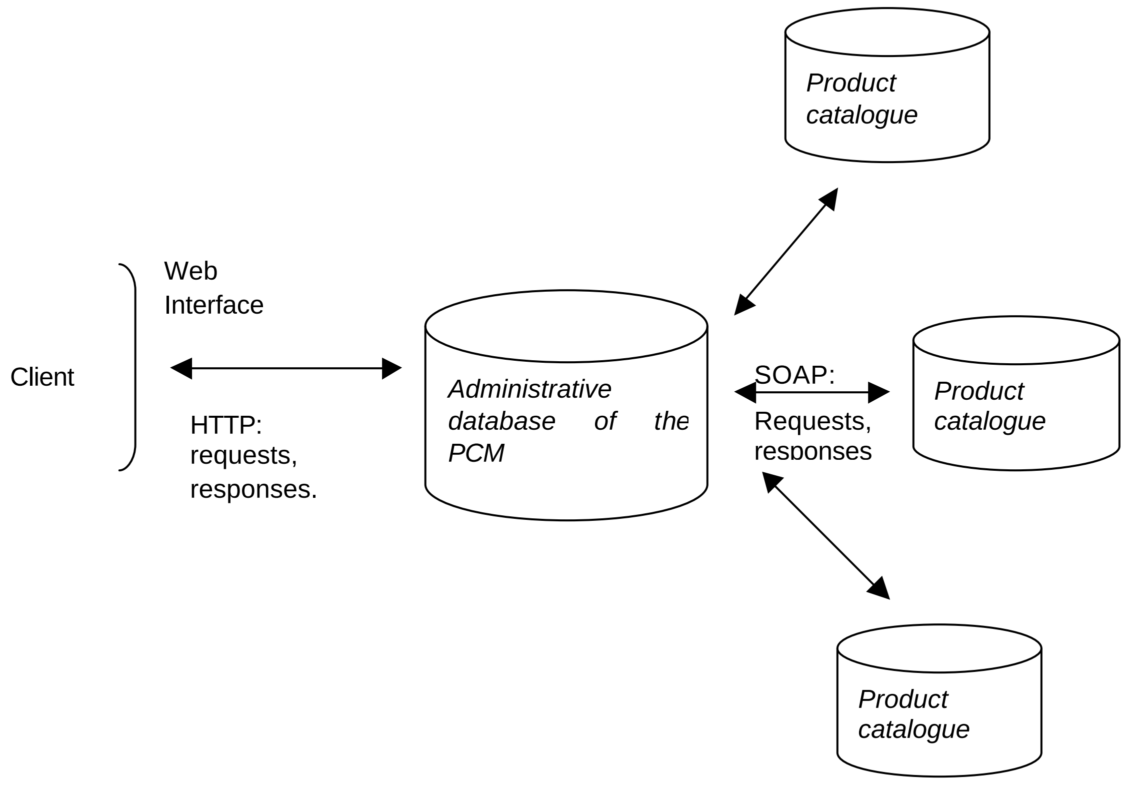


Figure 3: Scheme of the interconnections between the parts of the PCM.

Structure of the SOAP requests, responses is available by the references, provided on the each page of the PCM, which uses SOAP. To look on them it is necessary to click on the references:

Show SOAP request

Show SOAP response

2.2.3.2 Procurement and Delivery

The demonstrator uses Apache Soap implementation of SOAP protocol. Soap server publishes business interface that different clients are able to call if they support soap.

- searchProducts
- getUSerCommand
- createCommand
- getProduct
- getAllProducts
- addProductToCatalog
- removeProductFromCatalog
- getSupplier
- getAllCommands
- getCommand
- getArtistCommand getAllSuppliers

- searchSuppliersToBeValidated
- subscribeNewSupplier
- getListOfType
- getListOfCategories

2.2.4 Knowledge Base Access

The Knowledge Base System is the metadata secondary database of the REGNET System, and at the same time the core of the Ontology. Specifically, in accordance with the data format schema defined by the REGNET consortium, it consists of the REGNET *secondary data*, such as *Topic Maps*, *User Profiles*, *stylesheets* and so on. This data is auxiliary; it helps all the other nodes connected, to function properly. As an example, the Portal needs the *User Profiles* in order to perform user authentication, and the electronic Publisher requires stylesheets*, all stored in the Ontology Knowledge Base.*

The core of the **Ontology** can be summarized as a system that comprises of the **Knowledge Base** and a **set of programming classes** to perform essential functions against the Knowledge Base data, such as sending the data to the other nodes (with or without using the Java RMI protocol), or checking the validity of the documents stored in the correct XML format.

So, in order to describe the Core Ontology system one has to describe the implementation behind the Knowledge Base (section 3.2.4.1), and the software tools developed to add the functionalities required by the specific requirements of the Ontology System (section 3.2.4.2).

3.2.4.1 Knowledge Base implementation

From a technological point of view, Knowledge Base is an *XML database*, a more specialized kind of database with extra functionality provided for XML files, such as XPATH queries. The technology selected was **Xindice** (http://www.dbxml.org), an open-source XML database, reliable enough for the purpose of manipulating the secondary data of Regnet. It is implemented in Java, which makes it convenient to integrate to other software systems used for REGNET, such as the *tm4j* (a Java set of packages that handle the Topic Map format).

Some tests that were carried out showed that when correctly indexed, Xindice's performance was sufficient for the system requirement of the Regnet System; its problem lies in its inability to handle very large documents, or extreme quantities of files, but neither case is likely to occur when dealing with REGNET secondary data.

3.2.4.2 Tools

Since Xindice cannot connect to the other REGNET nodes by itself, or perform sophisticated actions, a set of software Tools were designed, which provide the main functionality of the Ontology System. The next section is dedicated to analysing those tools.

The following are intended for the REGNET programmers, who among other tasks implement the various interconnections through the system.

3.2.4.2.1 RMI interconnection

RMI is used to remotely execute methods and Java classes. Ontology provides a package, *org.regent.rmi*, which contains generic classes to be used for file transfer (uploading and downloading) between the Knowledge Base and the various nodes. The classes contained in the package are:

FileReceiver.class	Java interface for a class that acts as the RMI Server for the machine that requires files from a remote node. This class runs in the background, and its task is to receive files from the remote machine.
RemoteReceiver.class	The class implementing FileReceiver.
FileSender.class	Java class that sends the file specified to a remote machine.

FileSendInitialize.class	Java interface for a class that acts as the RMI Server for the machine that transmits files via RMI. It also provides methods to alter the transfer settings.
RemoteFileSendInitialize.class	The class for implementing FileSendInitialize.
ReceiveFile.class	Java class that requests and receives a file from a remote machine.

3.2.4.2.2 Ontology-Portal interconnection

The classes analysed in 3.2.4.1.1 provide a general file transfer system based on the RMI protocol. However, the interconnection with the portal requires further, more specialized classes. For this purpose, *org.regent.portal* was created, providing the necessary methods. There is an interface, *org.regnet.portal*, and an implementing class, *org.regent.portalimpl*. The methods contained are:

File getUserProfile (String userId)	Portal retrieves a specific User Profile from the Knowledge Base.
File getUserProfiles ()	Portal retrieves all of the User Profiles stored in the Knowledge Base.
boolean setUserProfile (File UserProfile, String name)	Portal sends a User Profile to Ontology.
boolean setUserProfiles (File LocalDir)	Portal sends all of its User Profiles to Ontology.
String getNewUserId ()	Generates a unique user Id to be used by portal.
File setLocalDirectory (String localdir)	Sets the local Portal directory.
void synchronizationEqualtoOntology () **void synchronizationEqualtoPortal ()**	Performs synchronization between the Portal and Ontology.

In the following a set of Ontology tools is described intended mainly for the REGNET.

XTM Loader

This is a web tool written in Java for designing and generating Topic Maps to be used by the REGNET content providers. Its presence in the system makes REGNET independent of expensive commercial solutions, such as Ontopia Navigator or Empolis K-42.

A demo of this tool can be found at: http://160.40.50.22:8800/servlet/org.regnet.ontology.XTMLoader

Knowledge Base Management System

This is a web tool developed with PHP. It provides a graphical user interface to manage and browse the contents of the Knowledge Base (the Xindice database). The user has the options of adding and deleting collections as well as documents, and performs XPATH queries to collections. A demo of this tool can be found at: http://160.40.50.22/arhs/regnet/dbxml/default_test.php

RMI Client

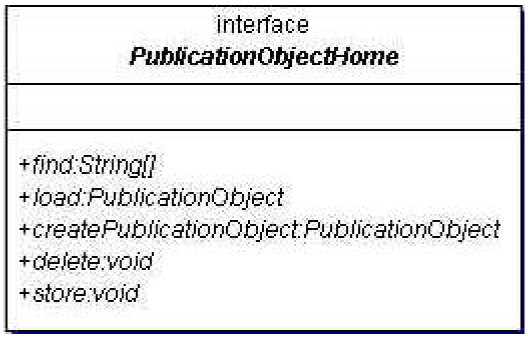
This is another web tool developed with PHP, designed for simple uploads of files from the Ontology Knowledge Base to a local directory using the powerful RMI protocol. A graphical user interface is provided, in which the user has the ability to select the type of document he/she wants to retrieve, and type its name. This system assumes that the user knows the name of the document he/she wishes to collect. A demo of this tool can be found at: http://160.40.50.22/arhs/client

2.2.5 Electronic publisher

The external interface of the Electronic Publisher is formed by the classes `PublicationObject` and `PublicationObjectHome`. The functionality of the interfaces is used by the Servlet / Applet of the Electronic Publisher. The current version of the Electronic Publisher is dependent on the Servlet / Applet (which is the visual representation of the Electronic Publisher) to gather the needed parameter information for publishing objects from the user. Due to that the external interface is not represented via a SOAP interface as for now the communication exclusively runs over the provided Servlet / Applet. However, the interfaces defined use standard types only which can easily be mapped to the xsd datatypes.

The two most important interfaces for external access, `PublicationObjectHome` and `PublicationObject`, are summarised below.

2.2.5.1 Interface PublicationObjectHome

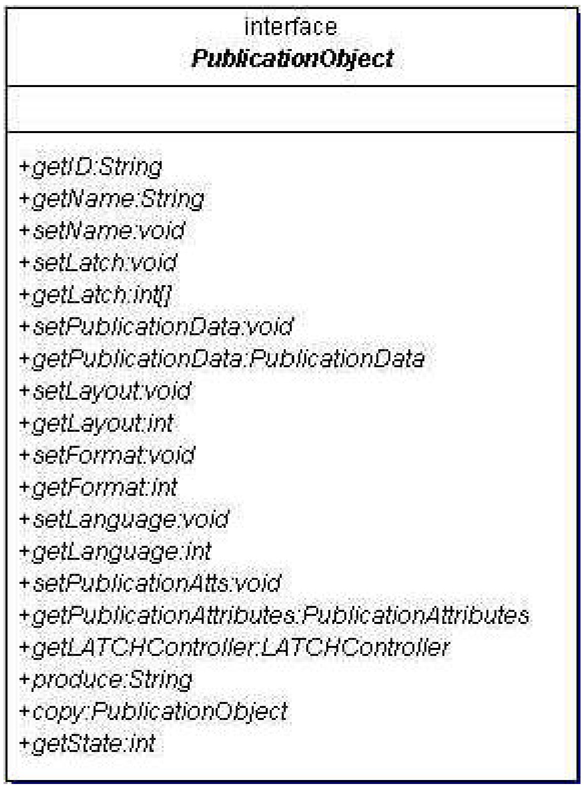


public interface **PublicationObjectHome**

This is the main interface for administrating publication data.

Method Summary	
`PublicationObject`	`createPublicationObject(String id)` Creates a new publication object
`void`	`delete(String id)` Deletes the specified publication object
`String[]`	`find(String[][] params)` Search for a specific publication
`PublicationObject`	`load(String id)` Returns a PublicationObject
`void`	`store(PublicationObject po)` Stores the specified publication object

2.2.5.2 Interface PublicationObject



public interface **PublicationObject**

PublicationObject is the central interface for creating a publication. It defines methods for selecting a layout, a format, the language and the sorting criteria by the LATCH principle.

Method Summary	
`PublicationObject`	`copy()` Produces a copy of this publication
`int`	`getFormat()` Returns the format of the publication
`String`	`getID()` Returns the ID of a PublicationObject
`int`	`getLanguage()` Returns the language of the publication
`int[]`	`getLatch()` Returns the selected LATCH principles
`LATCHController`	`getLATCHController()` Returns the LATCHController assigned to the PublicationObject
`int`	`getLayout()` Returns the layout of the publication
`String`	`getName()` Returns the name of the PublicationObject
`PublicationAttributes`	`getPublicationAttributes ()` Returns the PublicationAttributes at once
`PublicationData`	`getPublicationData()` Returns the PublicationData
`int`	`getState()` Returns the state of the publication
`String`	`produce()` Produces the publication according to the selected criteria (layout, format, LATCH principle, language
`void`	`setFormat(int format)`

	Sets the format of the publication
`void`	`setLanguage(int language)` Sets the language of the publication
`void`	`setLatch(int[] principles)` Sets the LATCH principle by which the publication should be sorted
`void`	`setLayout(int layout)` Sets the layout of the publication
`void`	`setName(String name)` Sets the name of the PublicationObject
`void`	`setPublicationAtts(PublicationAttributes atts)` Sets the PublicationAttributes at once
`void`	`setPublicationData(PublicationDate data)` Sets the PublicationData which is the base for the publication

2.3 Tools and Platform

The Regnet system has been developed using both J2EE, Microsoft/ASP and PHP technologies and making use of the following tools:

- UML modelisation:
  - Rational Rose;
- Java Environment:
  - JDK 1.3 and Servlet v.2.2;
- Java IDE:
  - Borland JBuilder v.5;
- Http Server and Servlet Engine:
  - Jakarta Tomcat v.3.2.2;
  - Apache Server.
- Portal Front-End Development:
  - Jakarta Jetspeed v.1.3.a and Portlet API
  - Jakarta Ecs v.1.4.1
- XML Parser/Validator:
  - Apache Xerces v. 1.2.0;
  - JDOM v.1.0b7;
- Version Control:
  - CVS and Tortoise CVS v.0.43
- WAP Emulator:
  - Motorola MADK 2.0, OpenWave v.4.1
- Standard Web Browsers.
- Databases:
  - Relational databases Management System: MySQL
  - XML databases: TextML server and dbXML (ontology).
- SOAP middleware:
  - Apache Implementation.

- Operating System:
  - Linux.

3 Prototypes

Prototypes are used in a directed way to reduce risk.

Evolutionary prototypes, as their name implies, evolve from one iteration to the next. While not initially production quality, their code tends to be reworked as the product evolves. In order to keep rework manageable, they tend to be more formally designed and somewhat formally tested even in the early stages. As the product evolves, testing becomes formalized, as usually does design.

3.1 Portal + J2EE/SOAP + PHP/SOAP

The objective of this prototype is to validate the integration architecture. The Portal has been developed using JetSpeed, a J2EE technology, and its components both with J2EE and with PHP. In order to integrate all of them different approach can be used:

- HTML portlet for loose coupling components;
- Protocol such as SOAP. SOAP is completely neutral with respect to operating system, programming languages and distributed computing platform.

3.1.1 Portal to J2EE through SOAP prototype

A prototype has been defined in order to validate communication between the portal and a J2EE based component. In order to do this, the portal has been simulated by a Jakarta/Tomcat container.

3.1.1.1 Scope

The prototype implements a simple procurement system for artist and supplier collaboration. Its aim is to validate technical architecture based on simple functionalities.

- **Supplier:**
  - o Subscribe in order to become a REGNET member with the role supplier. The subscription must be validated by the REGNET administrator.
  - o Describe catalogue in terms of products: type, category, reference, name, description, price.
- **Administrator** (user name *admin)***:**
  - o Validates new request for registration of suppliers.
- **Artists:**
  - o Must be registered.
  - o Have access to their selection.
  - o Navigate through supplier's catalogues to buy materials (an email is sent to the corresponding supplier).
  - o Look for a given product by name. They can add a quantity to a virtual cart (an email is sent to the corresponding supplier).

3.1.1.2 Architecture

The prototype is based on the following architecture.

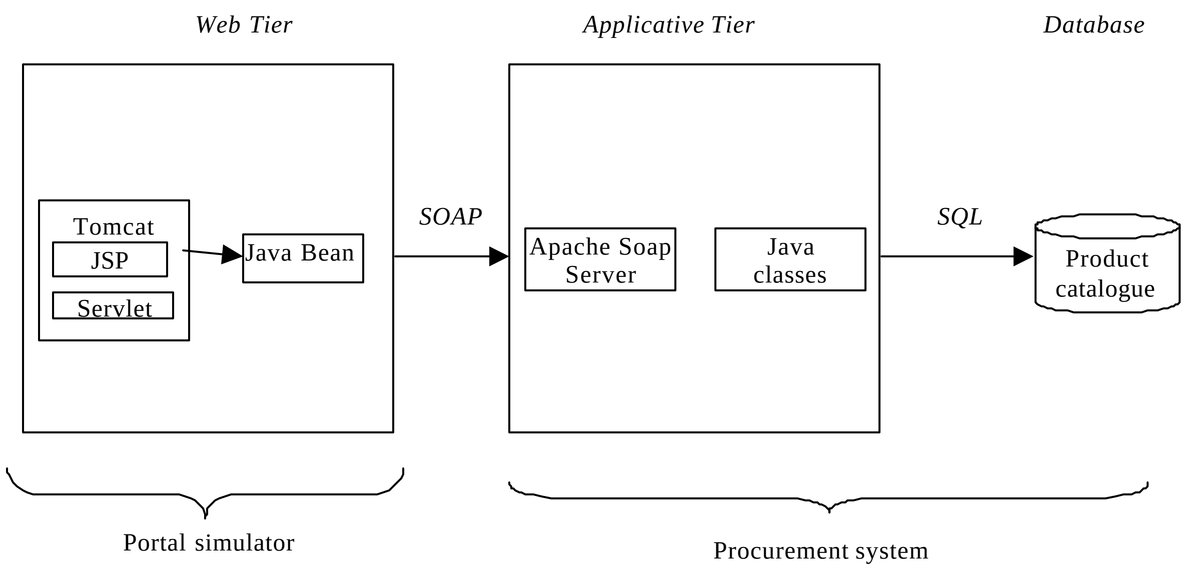


Figure 4: Prototype Architecture (Example: Procurement)

Connections between Presentation and applicative logic are based on a model 2 architecture that allows easy implementation of different communication protocols.
It is based on the following software:

- Apache/LOG4J : With log4j it is possible to enable logging at runtime without modifying the application binary. The log4j package is designed so that these statements can remain in shipped code without incurring a heavy performance cost. Logging behavior can be controlled by editing a configuration file, without touching the application binary.
- Apache/SOAP : Apache SOAP is an open-source implementation of the SOAP v1.1 and SOAP Messages with Attachments specifications in Java. Soap server is implemented as a web application in the tomcat server. The soap server deployment is very easy: only a jar file to deploy to tomcat application server.
- Apache/Xerces: The Xerces Java Parser 1.4.3 supports the XML 1.0 recommendation and contains advanced parser functionality, such as support for the W3C's XML Schema recommendation version 1.0, DOM Level 2 version 1.0, and SAX Version 2, in addition to supporting the industry-standard DOM Level 1 and SAX version 1 APIs.
- MySql: Open source relational database. *Take care that the system needs corresponding JDBC drivers*.
- Sun/JavaMail: The JavaMailTM 1.2 API provides a set of abstract classes that model a mail system.
- Sun/JSDK 1.3. The Java virtual machine.

3.1.1.3 Conclusion

The REGNET Portal has been deployed using J2EE technology. The application server used is TOMCAT V3.2.1. The REGNET Portal runs JSP and servlets for presentation facilities (dynamic generation of HTML pages). REGNET portal is responsible for running applicative logic including:

- Session data management
- Business logic invocation
- JSP execution

These dynamics pages extract data to show from a soap server. JSP and Servlet are soap client.

3.2 EbXML Message

As stated in the architecture document, communication between REGNET nodes is based on ebXML messaging. The aim of this prototype is to validate communication via this protocol.

3.2.1 Scope

In ebXML, an entire message is called a *message package*. A message package has one *header container* and zero or more *payload containers*. Only the header container is a SOAP envelope; the payload containers are SOAP attachments to the header container SOAP envelope.

As such, most of the ebXML-specific elements exist in the header container, which includes the elements that describe how the message will route, what type of operations it will perform, the unique identifier for the message, and so on.

The payload container(s) hold the message package's content, which can range from XML content (such as an invoice) to a TIFF image (such as a scanned-in legal contract). You refer to the payload container in the header container with an xlink reference in the header container's body.

<u>EbXML header elements</u>

To and From header elements

The To and From header elements in an ebXML message refer to a unique identifier for the sender (From) and a unique identifier for the receiver (To). It's recommended that both be URIs.

CPAId header element

The CPAId header element usually refers to an ID defined according to the Collaboration Protocol Profile and Agreement Specification. This ID defines how the two parties interact, both from a messaging perspective and from a business-process perspective.

ConversationId header element

A ConversationId header element uniquely identifies a set of related message exchanges. The ID must be unique.

Service header element

The Service header element denotes the service that processes the message at the destination (the To element).

Action header element

The Action header element, a subtype of the Service element, specifies what action the service should take.

MessageId header element

The MessageId header element uniquely identifies this particular message, usually in some form of a GUID (Globally Unique Identifier).

Timestamp header element

The Timestamp header element's timestamp field marks the message package's creation time.

3.2.2 Architecture

The prototype is built on top of the following software:

- Tomcat 3.2.2
- JAXM early access implementation

3.2.2.1 JAXM

The JAXM specification's stated mission is to provide a SOAP (Simple Object Access Protocol)-message-oriented API for Java. JAXM uses DOM4J and JDOM as the encapsulation mechanism for XML documents (which are messages).

JAXM clients can assume two basic roles in JAXM: *sender* and/or *receiver.* The sender role sends messages to other JAXM clients acting as receivers. The receiver role receives messages sent by other sender JAXM clients. A JAXM client can play either one or both of these roles.

A JAXM provider is a product or package that provides an actual implementation of the JAXM APIs, allowing different packages to offer a JAXM layer over an existing product.

JAXM supports both the SOAP specification and the SOAP Attachment specification. JAXM assumes this base level of support. In addition, although SOAP messaging standards such as ebXML or BizTalk are not yet specified, message profiles layer them onto JAXM.

The core includes client-side libraries to generate SOAP messages using the JAXM API. This client-side runtime library can be used to send messages to remote parties using either a *local provider* or a *remote provider*. Messages can be received synchronously (using a request-response model) or asynchronously.

Local Provider

A local provider is a pure library implementation of the JAXM API that lets you send SOAP messages directly to a remote party. A local provider by definition is simple to get started but has limited possibilities for reliability and message delivery guarantees. For instance, a local provider relies largely on the reliability of the underlying transport for delivering a message. APIs to work with the local provider are in the **javax.xml.soap** package of the JAXM API.

Remote Provider

A remote provider is akin to a messaging server. It takes messages from an application and holds on to the application until the message is delivered. The application itself may be active or quiescent, but the remote provider continues to try to send messages from the application and to receive messages on the application's behalf. When an application comes up and establishes a connection to a remote provider, messages received by the provider for that application are delivered to it. All messages sent through a remote provider are logged for perusal later. APIs to work with the remote provider are in the **javax.xml.messaging** package of the JAXM API.

3.2.2.2 JAXM API

The API is broken into a messaging package and a soap package.

javax.xml.messaging

The messaging package holds the classes, interfaces, and so on required for sending and receiving SOAP messages. Moreover, it contains the Connection class to connect to the JAXM provider and to send messages, as well as the MessageListener class for receiving messages.

javax.xml.soap

The soap package provides the classes necessary for encapsulating SOAP messages. You'll find classes for SOAP header, body, elements, and so on. Many classes in the current implementation extend DOM4J and JDOM classes to achieve their functionality.

3.2.3 Conclusion

The prototype validates simple communication between two distributed servers implemented as Servlets.

3.3 TopicMap creation

3.3.1 Scope

3.3.1.1 XML Topic Maps

The recently introduced technology of Topic Maps provides a knowledge representation of real-world subjects, collected in indices, catalogues and taxonomies. Specifically for the REGNET project, Topic Maps provide a way to create metadata for the museum items as well as general information concerning the method followed for organizing the data. This data will be stored in the central node of the REGNET System, the Ontology, and will be readily available for use by the interconnecting sub-systems (e-publishing, business, portal, search system).

A Topic Map is constructed of various elements that describe real-world items like a painter or a sculpture (a “topic” element), and the association/relation between them (an “association”). Further elements include occurrences, subject descriptors, variants of the original name, instances of a topic, and scopes of various elements. These will be further mentioned in the subsequent sections, along with the description of the Topic Map Authoring Tool.

The programming format used to map the Topic Map concept to a programming language is XTM, which stems from XML – it is a well-formed XML with a specific DTD applied. Its detailed specifications and current status can be found in http://www.topicmaps.org/xtm/index.html

An XTM file can be thought of as consisting of just three components:

Topics

which describe a subject in the real world. A subject may be either addressable by a machine (e.g. a web-page with a URL); or not addressable (such as a person) or even an abstract concept (such as 'Music').

Associations

which each describe a relationship between a set of topics. The association construct may include additional typing information, which specifies the nature of the relationship between the topics, and also what role each topic plays in the relationship. Thus, association can be used to represent such common concepts as a hierarchy, or a group but also more complex relationships such as a meeting (with topics representing people and playing roles such as 'chairman', 'scribe' and 'attendee').

Occurrences

which connect a topic to some information resource related to the topic. The occurrence can be used to specify both the information resource (or the information itself) and the nature of the relationship between the information resource and the topic. For example, a topic could represent a person and that topic could have an occurrence, which could be a URL, which points to the 'homepage' of the person.

In addition to defining these basic structures, the topic map standards also define the way in which two or more topic maps may be combined or merged. Topic map merging is a fundamental and highly useful characteristic of topic maps as it allows a modular approach to the creation of topic maps and it provides a means for different users to share and combine their topic maps in a controlled manner.

3.3.1.2 Topic Map Generator Tool

This authoring tool gives to the content providers an easy way to produce XTM files and store them automatically in the Knowledge Base for immediate use by all REGNET nodes. The Topic Map Generator tool is web-based and it is implemented with the Java programming language, as a servlet. Many users can simultaneously access the features of the tool (concurrent requests). The concurrent requests are realised by server sessions. Every file produced by a user, is available to other users as it is stored to the Knowledge Base. Every XTM file contains multiple topics. The Topic Map Generator

Tool helps users to extract those topics and uses them in order to create new XTM files by combining them properly.

3.3.2 Architecture

3.3.2.1 Topic Map Generator Tool Architecture

The Topic Map Generator Tool is based on the TM4J, which is a Topic Map engine for Java. TM4J is an open-source project to develop topic map processing tools and applications. The current focus of the TM4J project is on the development of a topic map "engine" which processes files conforming to the XML Topic Maps (XTM) specification and stores them either in memory or in a persistent store, providing access via a Java API. In REGNET the persistent store is Xindice (a native XML database), which is also free.

The architecture of the Topic Map Generator Tool is as follows:

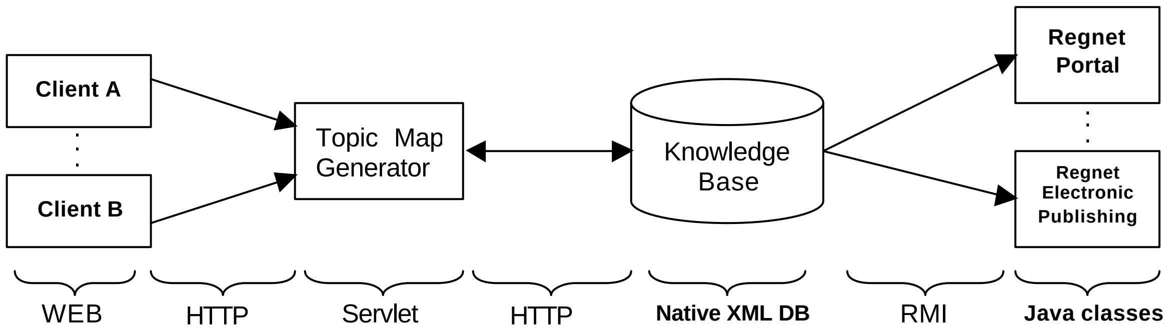


Figure 5: Architecture of the Topic Map Generator Tool

- Content providers connect to the Topic Map Generator Tool through an Internet Browser using the HTTP protocol.
- The web pages are produced by the Jakarta Tomcat Server (Servlet) and sent back to clients.
- The Topic Map Generator communicates with the Knowledge Base (Xindice in our case) to store and retrieve XTM files.
- Every REGNET node has access to the Knowledge base through RMI.

The java class XTMLoader implements the Topic Map Generator Tool. It is a public class that extends the javax.servlet.http.HttpServlet and it is a part of the org.regnet.ontology package. A more detailed architecture diagram is shown in figure 2:

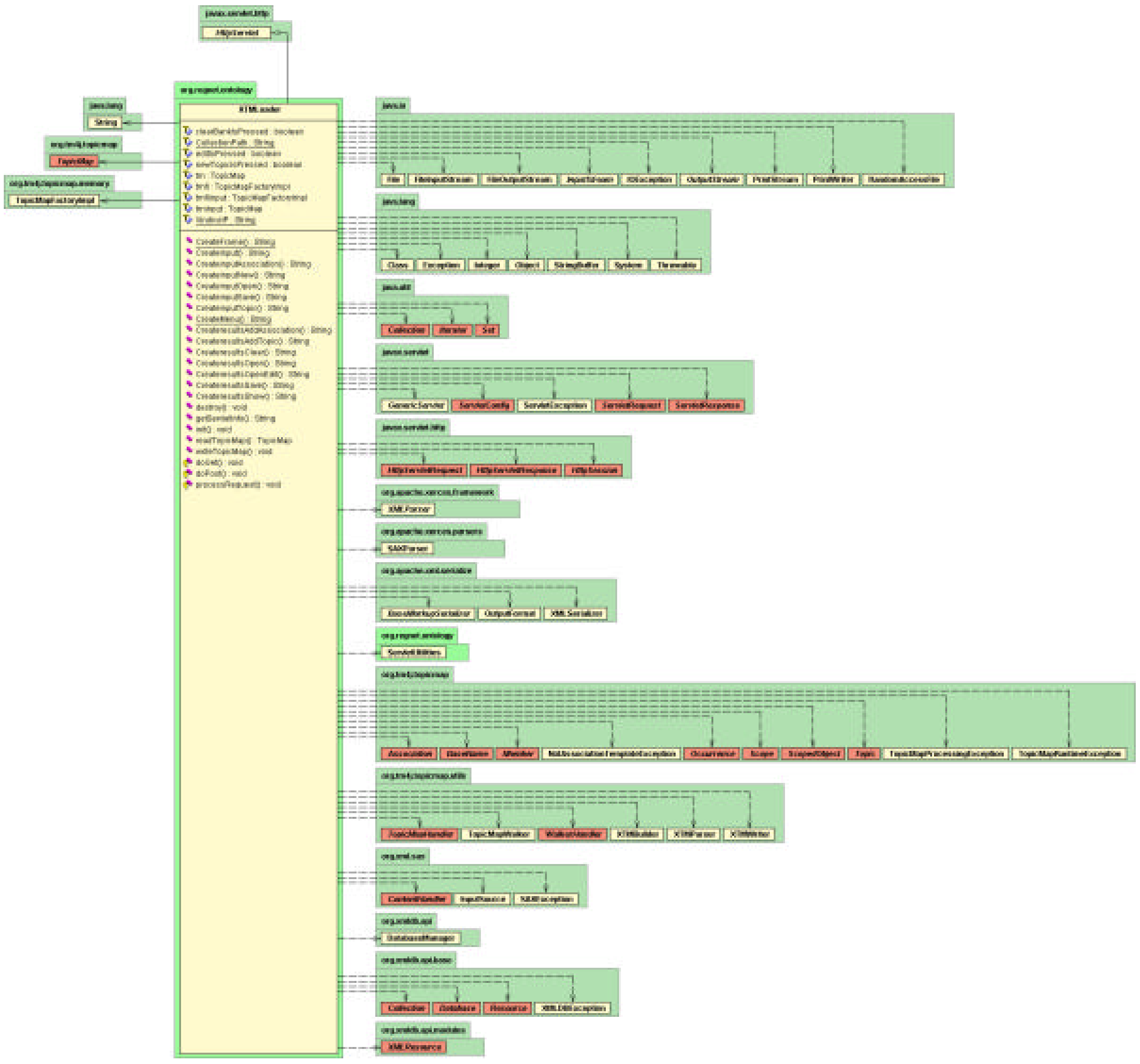


Figure 6: XTMLoader class

As concluded from the above diagram, XTMLoader class provides the methods necessary for creating the interface of the tool.

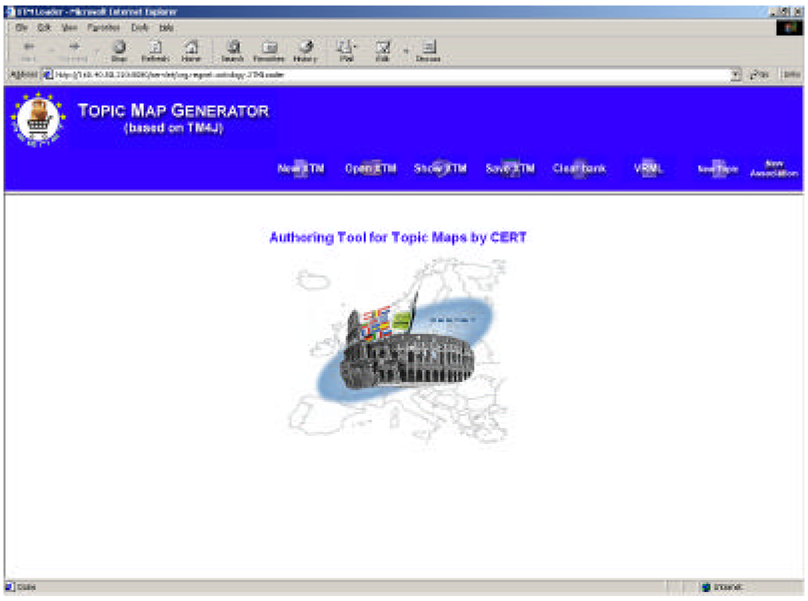


Figure 7: Main page of Topic Map Generator Tool

Figure 3 shows the web interface of the Topic Map Generator Tool, which can be accessed through an Internet browser. It is developed to be used by the content providers of REGNET.

3.3.3 Conclusion

A tool for creating Topic Maps was essential for the REGNET content providing partners, since they have the responsibility to create XTM documents and insert them into the Ontology System. Alternatively to the use of the tool the users could either create a Topic Map file manually (typing the whole XTM code in a word processor) or buy an off-the-shelf commercial system (Omnigator, k42, etc). Both solutions are cumbersome: the first requires increased effort from the user's part, while the second is considered out of the scope of REGNET. The Topic Map Generator offers some of the functions and features of a commercial system, specifically developed for REGNET and provides custom tools for direct interconnection with the Knowledge Base.

3.4 TopicMap visualisation

3.4.1 Scope

A tool that allows the transformation of a Topic Map file (xtm) to VRML format (wrl) has been developed. The use of VRML allows to create a 3D visual interface on the web that can display large amounts of data (topics). The VRML output of Topic Maps is based on the Cone Trees methodology for displaying large datasets. The end-user is able to interact with the 3D model. This technology provides the user with unlimited viewpoints and is an efficient way to visualize Topic Maps.

The tool is entitled Topic Maps to VRML. The current version does not support all functionalities provided by the Topic Maps standard. The purpose was to develop a tool that will cover the REGNET needs rather developing a tool that will cover the whole concept of Topic Maps.

3.4.1.1 <u>VRML Definition</u>

VRML stands for Virtual Reality Modelling Language. Technically speaking VRML is neither virtual reality nor a modelling language. Virtual reality typically implies an immensive 3D experience (such as head-mounted display) and 3D input devices (such as digital gloves). VRML neither requires nor preludes immersion. Furthermore, a true modelling language would contain much richer geometric modelling primitives and mechanisms. VRML provides a bare minimum of geometric modelling features and contains numerous features far beyond the scope of a modelling language.

VRML was designed to create a more "friendly" environment for the World Wide Web. It provides the technology that integrates three dimensions, two dimensions, text, and multimedia into a coherent model.

3.4.1.2 Using VRML to display Topic Maps

Data visualization is an area where VRML can be effectively used. Data visualization can be used to understand large databases of information, such as in financial trading, Web traffic analysis, database marketing analysis, and scientific research.

Topic Maps is an international standard providing a language expressed in SGML and HyTime to construct a layer of topics and relations aimed at classifying and semantically tagging a collection of documents.

The aim of using VRML in terms of information access is to facilitate navigation through dynamic, renewable textual resources, and to make this navigation adaptable to different users' viewpoints.

Navigation on the basis of a pre-defined category taxonomy is inadequate in the case of dynamically assembled text collections as these collections are heterogeneous and do not correspond to circumscribed domains. Furthermore, a pre-defined taxonomy imposes a single, fixed way of representing the content of a text collection, which is not necessarily aligned to the user's viewpoint.

One of the aims of the REGNET Topic Maps navigation interface is therefore to take into account the user's point of view and orientation.

The approach of creating different views on the text collection instead of imposing a certain perspective on it through a fixed, pre-defined taxonomy allows more flexibility and control to the user for accessing information. This can be achieved by using the VRML standard.

3.4.1.3 The pros and cons of using the Topic Map To VRML tool

Pros

- VRML file size is very small.
- Easy to generate VRML files.
- A concrete way to visualize TM since there are not many available tools in the market and their cost is high.
- Possible to have the tool online in a server and dynamically generate VRML Topic Maps.
- VRML viewers are free to use.
- Compatible with most common used Web browsers.

Cons

- The VRML files are not binary.
- Difficult to integrate fully the Topic Map standard.
- Difficult to navigate in a large topic Map file.

3.4.2 Architecture

The following diagram displays the components from which the TopicMap To VRML tool consists of. The tool is part of the Ontology node and is connected with the Topic Maps database. Once a topic map file is being produced in the Ontology node then this will be able for retrieval from this tool. Then the tool will produce the visual interface for the topic map file (3D and 2D) and the output will be stored in a new database (3D/2D database). The produced result will then be available to the client via the web browser.

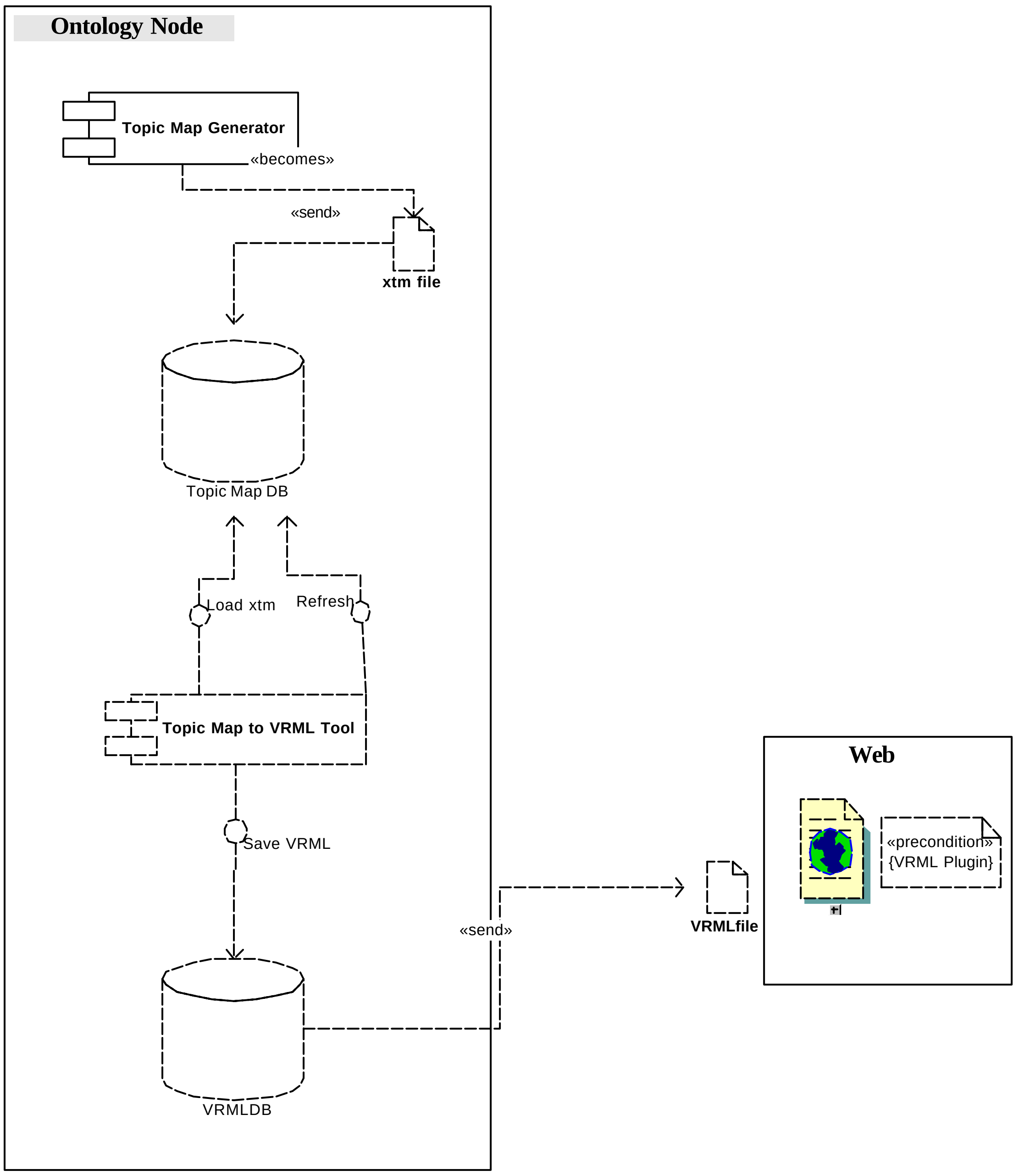


Figure 8: Component Diagram – Ontology Node

3.4.3 Conclusion

It is desirable to have a 3D and 2D visual interface for displaying topic maps. At this stage the 3D interface has been developed, while the 2D option has not been fully developed yet. It is expected that the 2D interface will be fully integrated to the Ontology node and it will be available until the end of the development phase.

3.5 Search and Retrieval

3.5.1 Scope

The Search and Retrieval part of the REGNET System can be split into 2 parts. One part is responsible for presentation issues and is located and integrated into the portal. The other part resides in the Cultural Heritage Data Management subsystem. The component is called Reference System and provides access to the repositories.

The search and retrieval steps that can be performed with the prototype can be split as follows:

- Build up user interface (= query interface).
- Process query (= SOAP request to Search Service)
- Presentation of Result
- Navigation trough the Result

3.5.2 Architecture

The prototype is based on the following architecture.

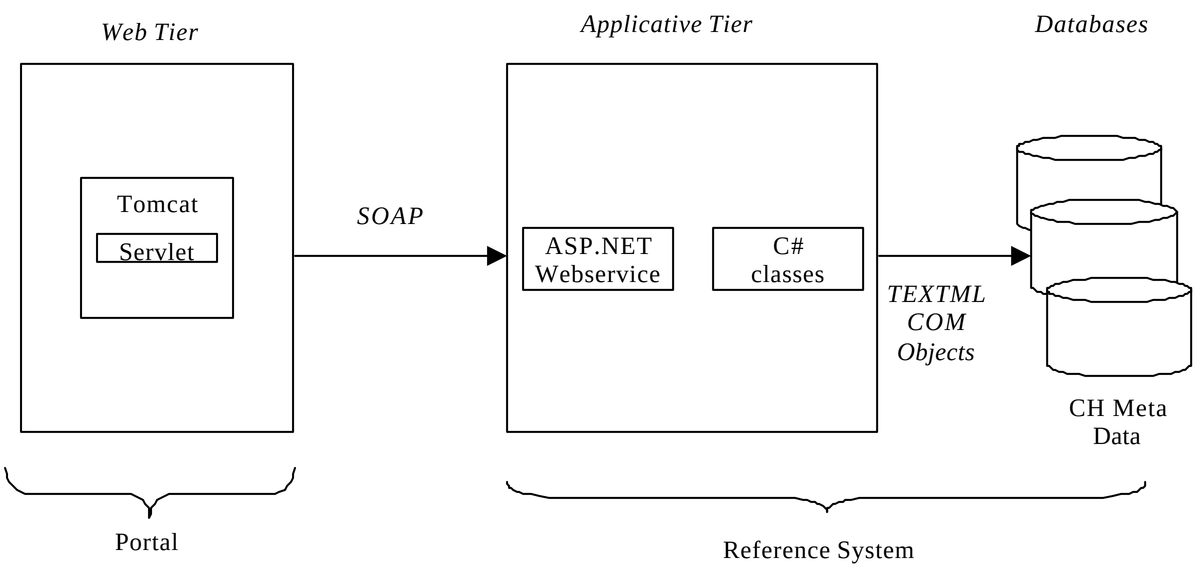


Figure 9: Search and Retrieval Architecture

Connection between presentation and applicative logic is based on a model 2 architecture that allows easy implementation of different communication protocol.

It is based on the following software:

- Apache/LOG4J : With log4j it is possible to enable logging at runtime without modifying the application binary. The log4j package is designed so that these statements can remain in shipped code without incurring a heavy performance cost. Logging behaviour can be controlled by editing a configuration file, without touching the application binary.

- .NET Framework : Microsoft .NET is the Microsoft XML Web services platform. For developing .NET applications and Web services you have to install the .NET Framework SDK that can be downloaded from the Microsoft homepage. Additionally the IIS from Microsoft has to be installed to work with a .NET web service. .NET Web services are compliant with the SOAP v1.1 specification. The SDK includes also tools for compiling, debugging and deploying Web services.

- Apache/Xerces : The Xerces Java Parser 1.4.3 supports the XML 1.0 recommendation and contains advanced parser functionality, such as support for the W3C's XML Schema recommendation version 1.0, DOM Level 2 version 1.0, and SAX Version 2, in addition to supporting the industry-standard DOM Level 1 and SAX version 1 APIs.

- Apache/Xalan : The Xalan 2 XSLT processor is a Java API for transforming XML documents into HTML, text, or other XML document types. It implements the W3C Recommendations for XSL Transformations (XSLT) and the XML Path Language (XPath). Xalan is compliant to Sun's JAVA API for XML Processing 1.1 (JAXP) since version 2.

- Mindelectric/GLUE : The GLUE 2.3 standard edition is a JAVA API for SOAP messaging. The GLUE standard edition is free for most commercial uses.

- TEXTML-Server : Native XML database system. It is accessible via an AIP based on COM technology and defines an own query DTD. A subcomponent is responsible to transform queries into the TEXTML query format.

- Sun/JSDK 1.4 . The Java virtual machine.

3.5.3 Conclusion

REGNET Portal has been deployed using TOMCAT on the WebTier side. The Portal runs Servlets for presentation (dynamic generation via XSL transformation) and for establishing connection to the Search Web Service by the means of the GLUE API. The Web service in this prototype can now only access TEXTML-Server databases but the interfaces are designed to be open for other database systems also. The next step will be to implement a Search Web services accessing a Xindice database. In this way performance optimised Web services can be developed for different database systems.

The WebTier part of this prototype does not take care of user- and terminal-profiling which is part of the real REGNET portal.

The prototype validate communication between two distributed servers one implemented as Servlets and one implemented as ASP.NET Web service. It also validates cross-domain searching on native XML databases and shows the way to integrate other database systems as well.

3.6 CatXML

3.6.1 Scope

The CatXML specification is the basic infrastructure that supports the Product Catalogue Management.

The main issue is the usage of the interoperable standard in order to be compliant with the most recent standards. A flexible 'smorgasbord' approach allows industry partners to implement matching profiles quickly. Both inbound and outbound, completely scalable, dynamic information exchanges are supported using the information server approach as its foundation.

It is important to emphasise the fact that the PCM uses the CatXML specifications and provide all necessary information in XML format.

The main scope of the CatXML technological support is the validation of the interconnection of the Catalogue Management with distributed sources of information.

3.6.2 Architecture

The basic architecture was established in the development of the PCM according to the CatXML standards, is the structure of the PCM schema, which consists of three parts (see next figure).

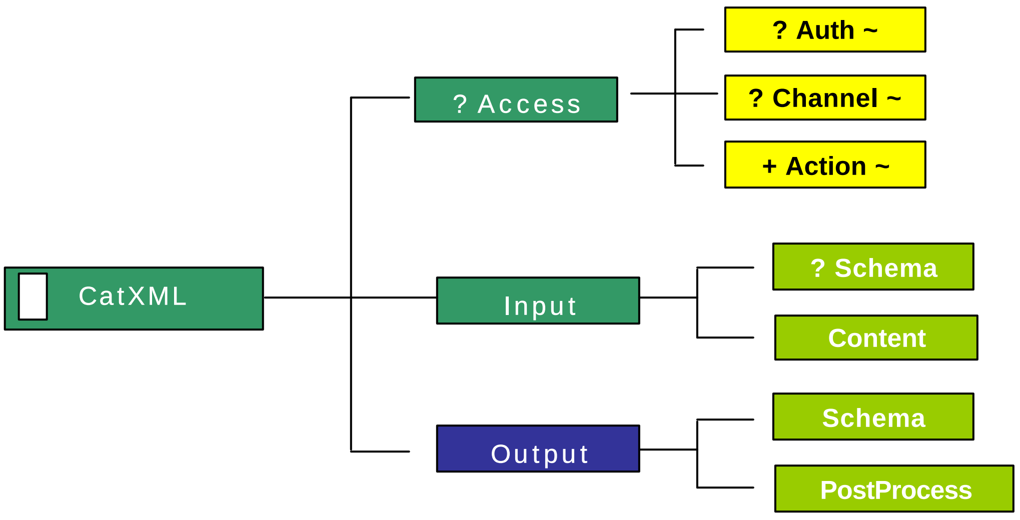


Figure 10: Components of the CatXML interchange within the PCM

Within the Access profile, the Authentication (Auth), Channel and required action (Action) options (Read, Search, Match, Update, Delete, or Insert) are specified. This allows control over the type of interchanges a specific user can attempt.

The Input section defines the information associated with the Action (default action is 'read/search'). In the search mode the Content itself contains the value, or values, that are to be matched on, otherwise it is the replacement information in an update operation.

An optional additional schema can be provided. The primary need for this is to supply associations between the provided information, and the CatXML base definitions, when they are proprietary or different.

The Output section can be used to customize the behaviour of the response received from a CatXML server system. By default CatXML provides a base schema that is designed to cover most catalogue information combinations. Figure 3 shows the structure of this base schema catalogue definitions.

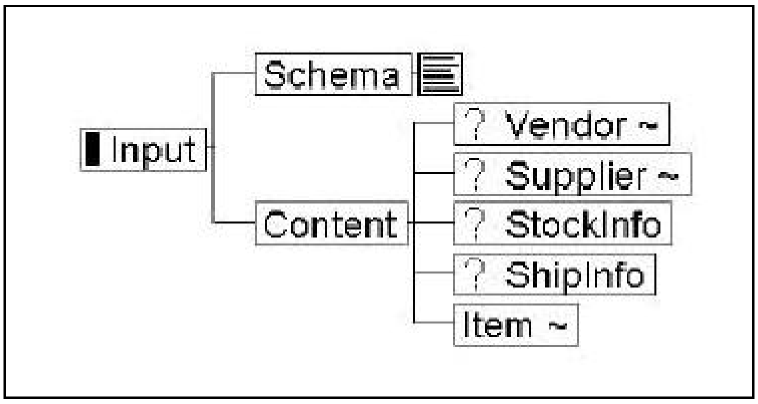


Figure 11: Components of the PCM – CatXML Catalogue Definitions

3.6.3 Conclusion

The most important aspect of the CatXML integration in the PCM is the provision of information regarding the vendors and the suppliers to the E-Business subsystem. The interconnection of the two specific subsystems is adequate to provide the B2B scenario available to the REGNET members. All necessary input and output format are also available in XML.

It is important to note that we have selected the basic elements of the CatXML specifications and in the next version we will be able to integrate the added value elements if it will be required by the REGNET project.

3.7 Portal

The main tasks that must be provided by the Portal are the following:

- Provide the main access point to all the subsystems (e.g. Search and Retrieval, eBusiness, Data Entry);
- Provide the users' registration and authentication;
- Manage users' and terminal profiles;
- Track session during user's navigation;
- Manage internationalisation.

Let's see more details of each of these items.

3.7.1.1 Main access point

The Portal must be the access point to the entire REGNET infrastructure and to all its components. In order to do that the Portal had to harmonize and integrate all the available components and tools developed by the different partners with different technologies and on different platforms.

The three main subsystems to include into the Portal were the following:

- Data Generation;
- Search And Retrieval;
- E-Business.

The integration had been conducted starting from the interfaces supplied by each component and the overall communication infrastructure based on SOAP.

1.1.1.1 User registration and authentication

The Portal must be able to allow the users to register them into the REGNET system. Every time they want to access to protected features the Portal must be able to give them the access only if they are authorised.

The tool used to realise the Portal (Jakarta Jetspeed) already provides its own registration and authentication features. This mechanism had been revised and modified in order to support the REGNET security requirements.

1.1.1.2 User and terminal profiles

The Portal must be able to manage user and terminal profiles. This means that the Portal must be able to adapt its contents according to the preferences specified by the users (user profiles) and to the features of the terminal with which the user will access the REGNET Portal.

1.1.1.3 Session

The Portal must be able to manage the session and the information associated to the user during each HTTP session.

1.1.1.4 Internationalisation

The Portal must be able to manage the following internationalisation aspects:

- As default the user obtains all REGNET information shown in the language specified during the registration phase;
- The Portal should offer the user the possibility to choose a language at runtime (eventually different to that specified during the registration) to access the REGNET information.

At the moment there are two Portal prototypes available: A Web Portal prototype and a Wap Portal prototype. The last one offers the wireless REGNET user a limited set of functionalities while the Web Portal allows access to all the REGNET functionalities with any Web browser. The server side business logic is common for the two prototypes while the front-end has been specifically developed for each of them.

3.7.2 Scope

The scopes of the Portal prototypes are to test the components integration, to harmonize all the available features and to make all the components already developed available. The last version of the Portal prototype meets the following requirements:

- Availability of tools and applications developed by the various partners;
- Integration of the Search And Retrieval module through SOAP;
- Integration of the Data Entry module through SOAP;
- Integration of external applications through HTTP Forward;
- Users' registration and authentication;
- User profiles synchronization with the Ontology repository;
- Support for registered users and system administration.

3.7.3 Architecture

The Portal architecture is constructed by as a set of different, overlapping layers. At the base there are the Web Server (Jakarta Tomcat) and the portal development tool (Jakarta Jetspeed). Upon that there are the layers responsible to manage the portal core functionalities. At the top the layers suitable to manage more specific tasks such as for example all the presentation aspects are situated.

The overall architecture is shown in the following figure:

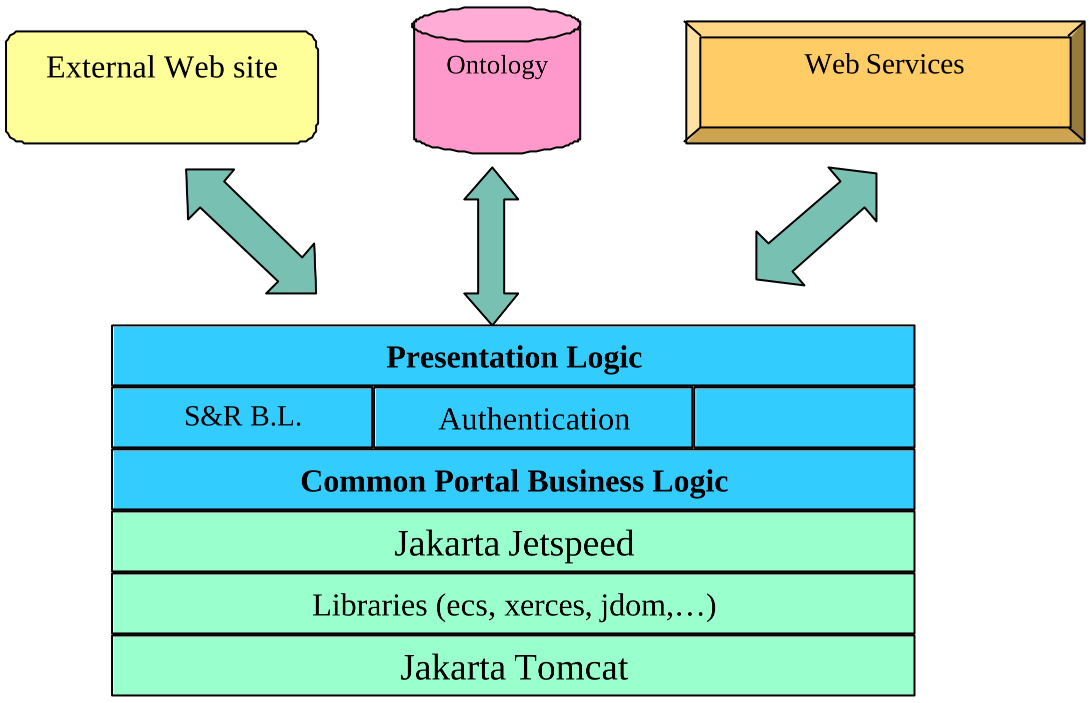


Figure 12: Portal Architecture

The top layers are able to manage the interactions with the external components such as the Ontology repository, Web sites offering REGNET tools and Web services (such as those provided by the Search And Retrieval component).

3.7.3.1 Common Business Logic

As stated previously the Portal is constituted by different components. Some of them are responsible to manage the server-side business logic and others to manage the front-end. More in detail the server-side layers are the following:

1. User registration;
2. User authentication;
3. User profile management;
4. User terminal management;
5. Session management.

The available prototype contains the first three modules. The fourth will be available until the next release planned for the end of June while the last one is realised by Jakarta Jetspeed. In the next release possible extensions and necessary modifications to such modules are planned.

User Registration

The User registration module is responsible for providing the visual interface necessary to input user information. After that this module must be able to register the user in the REGNET system.

In order to satisfy both requirements specified above, the steps performed when a new user chooses to register himself are the following:

1. The user accesses the registration interface provided by the Portal and provides his information (name, surname, email address, phone number etc…);

1. The portal gathers all the user information, and then asks the Ontology to associate the user_id to the new user. This id must be unique in the entire REGNET system;
2. The Ontology generates an unique userId and sends it back to the Portal;
3. The Portal registers the user inside the Portal database;
4. the Portal builds the user profile file (the structure of such a file is described in the subsequent paragraph) and sends it to the Ontology;
5. the Ontology adds the new user into the central repository.

After that the user is registered for the entire system.

User Authentication

The user authentication module is responsible for checking the access to the system. Only authorized users and also to make available to the user only the applications and services associated to his profile. The authentication system is based on the single sign on principle, i.e. that the user can access all applications with the same password.

In order to satisfy the requirements specified above the authentication mechanism has been realized in the following way:

1. during the registration mechanism is associated with the user a unique userId and password;
2. when the user wants to access a protected application, the Portal requires a login;
3. the Portal will check the user name and password;
4. if the information specified at point 3 is correct the Portal will give access to the protected application, otherwise it will ask the user to re enter his data;
5. every time the user would wants to access another protected application the Portal will send all necessary information to the application as described in the following paragraph;
6. the user doesn't need to re enter his login and password.

User Profile management

The user profile management module is responsible to manage the profiles associated to each REGNET user. The information contained in the profile is personal (such as user name and surname) and also REGNET specific.

At the moment the user profile is stored in an XML file structured in the following way:

```
<?xml version="1.0">
<userProfile>
<userName> user_login<userName>
<Surname>user_surname</Surname>
<Name> user_name </Name>
<Sex> user_sex</Sex>
<Age> user_age</Age>
<Occupation> user_occupation</Occupation>
<Language>user_language</Language>
<Terminal>user_terminal_number</Terminal>
<userId>userId_generated_by_the_Ontology</userId>
<password>user_password</password>
<email>user_email_address</email>
<address>user_address</address>
<phone>user_phone_number</phone>
<RegnetUser>user_role</RegnetUser>
</xml>
```

Beyond that information it is necessary to provide and maintain a set of applications associated to each role defined in the REGNET system. That means that it is necessary to store and maintain the set of applications and tools available for example to all the REGNET registered users, to the administrators, to the suppliers, etc...

The user profiles (which is the XML profile file) must be known and maintained both by the Portal and by the Ontology. The Ontology must act as the common repository for all the information related to the user. Such information could be used by the application activated by the Portal.

In the available prototypes the user profiles are managed in the following way:

- as stated before the userId is generated by the Ontology repository. The userId will be used as key to uniquely identify the user inside the REGNET system;
- every time the user accesses the Portal interface and chooses to access one application the Portal has to launch the chosen application passing the userId associated to the user;
- after that, the launched application could use the userId to access the Ontology repository to extract all the information available into the user profile.

For example an application such as the eShop could need the user address. Thanks to the userId passed by the Portal the eShop can extract the needed information from the Ontology subsystem.

3.7.4 WEB and WAP Prototypes

The Web Portal prototype available at the moment is able to manage the various aspects described in the previous paragraph. The WAP Prototype on the other hand, cannot provide all the functionalities of the Web-based Portal due to the limitation of the Wap environment (display size, memory, connection speed). In order to provide a usable tool special care has to be taken in the area of the GUI design and the data volume transferred from the server.

3.7.5 Conclusion

As stated in the previous sections the two available prototypes allow the access to the tools and applications already developed and are able to manage a set of user profiles (registered users and administrator). Layout aspects such as accessibility and cosmetic enhancements will be addresses in the next release.

Other aspects that will be treated in the next releases are the following:

- Complete the user profile management;
- Develop the terminal profile management;
- Develop an enhanced WAP Portal;
- Give the support for the internationalisation and localization;
- Enrich the Web Portal Prototype.

3.8 Publication

3.8.1 Scope

The Publication Prototype allows the authoring and publishing of search results returned by the S&R subsystem. The process of defining publications is described below.

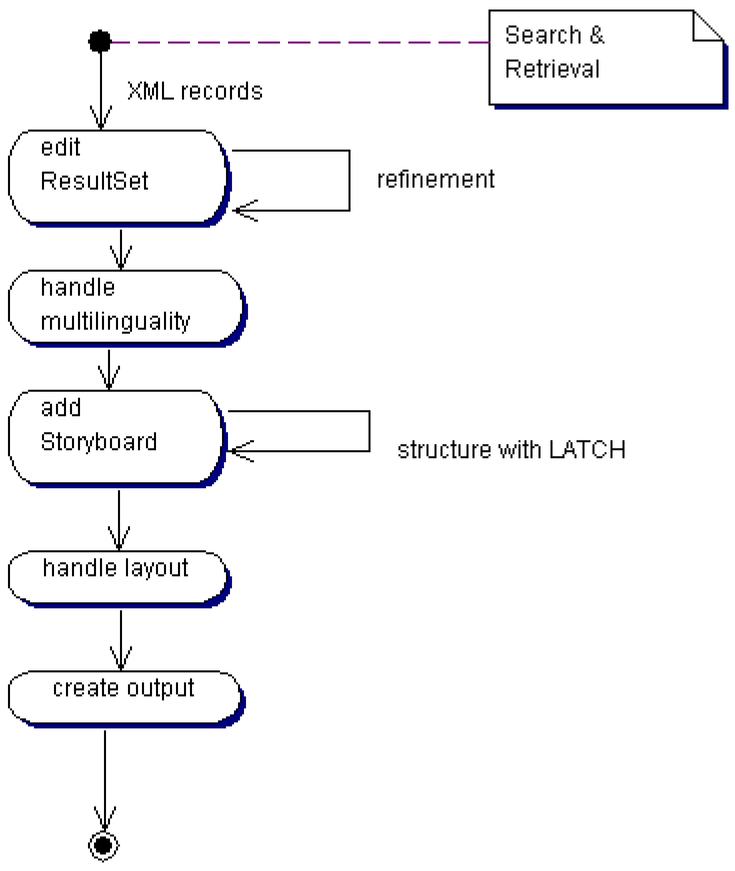


Figure 13: Activity Diagram - Publication

The first version implements the key activities "add storyboard", "handle layout" and "create output". In the following, each of these steps is described briefly. A more comprehensive description can be found in the users manual of the authoring tool.

Search & Retrieval

The input of the EP Prototype is a search result from Search & Retrieval in XML format according to the REGNET dtd.

Edit ResultSet

The first prototype does not organise the input Resultset by the means of editing it.

Handle Multilinguality

As the first version of the REGNET system is implemented in English only, this functionality is currently not implemented. This functionality will be realised with multilingual lookups using stylesheets. It will be possible to associate multilingual labels with particular fields in a record set.

Add Storyboard

The Electronic Publisher allows to process data in a location-based manner.

Handle Layout

Layouting is performed through XSL transformation. The user can select different layouts by selecting a stylesheet to be applied. The layouting depends on the publication type selected, i.e. a catalogue and a virtual gallery have distinct layout stylesheets.

Create output

In this step the user can choose the format for the publication. In the first version PDF, HTML and SMIL are supported. The applicable formats depend on the chosen layout, i.e. a catalogue might be formatted in HTML or PDF but cannot be formatted in SMIL whereas a Virtual Gallery can be represented in HTML and SMIL. The dependencies of layout and output are primary determined by the degree of user interactivity.

3.8.2 Architecture

The following diagram shows the architecture of the Electronic Publishing subsystem.

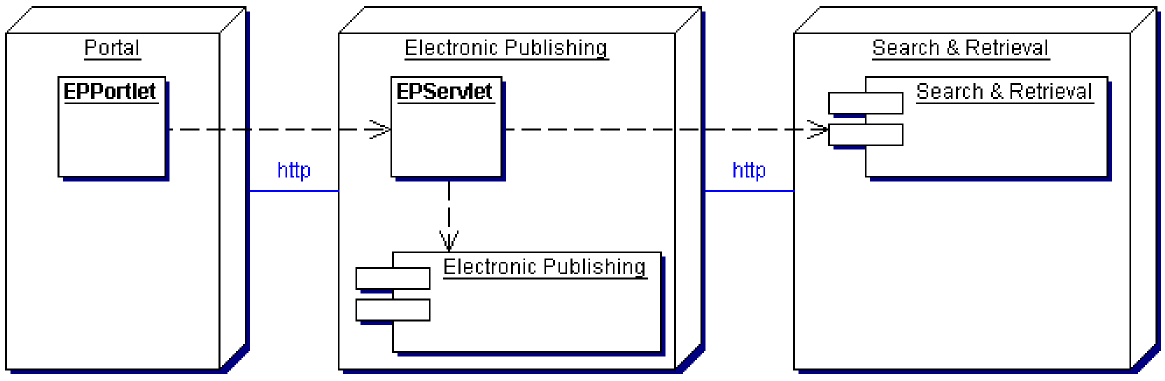


Figure 14: Electronic Publishing Architecture

The EPPortlet component is integrated in the Portlet environment, which connects to the EPServlet on the Electronic Publishing node. The EPServlet controls the Electronic Publishing component and uses the Search & Retrieval subsystem to get the data for publishing.

Within the publication prototype the following tools where used:

a) Xerces

Xerces is a publicly available XML parser from the Apache Software project. This parser can convert text XML files into DOM representations that can be further queried or manipulated by other XML-aware programs.

Fully validating parsers are available for both Java and C++, implementing the W3C XML and DOM (Level 1 and 2) standards, as well as the de facto SAX (version 2) standard. The parsers are highly modular and configurable. Initial support for XML Schema (draft W3C standard) is also provided.

The Xerces Java Parser 1.4.4 supports the XML 1.0 recommendation and contains advanced parser functionality, such as support for the W3C's XML Schema recommendation version 1.0, DOM Level 2 version 1.0, and SAX Version 2 in addition to supporting the industry-standard DOM Level 1 and SAX version 1 APIs.[http://xml.apache.org/xerces-j/index.html]

Other XML parsing tools evaluated:

- XML4J from IBM: IBM is a major contributor to Apache's Xerces-J code base. Version 1.4.2 of Xerces-J forms the basis for XML4J 3.2.1. IBM is a pioneer in XML technologies, with parsers that have been consistently highly rated since XML4J version 1.0 was released in 1998. The XML Parser for Java is a validating XML parser written in 100% pure Java.
- expat - XML Parser Toolkit: Expat is an XML 1.0 parser written in C. It aims to be fully conforming. It is currently not a validating XML processor
- XP: XP is an XML 1.0 parser written in Java. It is fully conforming: it detects all non well-formed documents. It is currently not a validating XML processor. However it can parse all external entities: external DTD subsets, external parameter entities and external general entities.

b) Xalan

Xalan is a publicly available XSLT engine from the Apache Software project. This engine can transform XML files into other formats such as other XML files, XHTML, WML, etc., according to the specifications of an accompanying XSL file.

c) Cocoon

Apache Cocoon is an XML publishing framework that raises the usage of XML and XSLT technologies for server applications to a new level. Designed for performance and scalability around pipelined SAX processing, Cocoon offers a flexible environment based on the separation of concerns between content, logic and style. A centralized configuration system and sophisticated caching top this all off and help you to create, deploy and maintain rock-solid XML server applications.

Even if the most common use of Cocoon is the automatic creation of HTML through the processing of statically or dynamically generated XML files, Cocoon is also able to perform more sophisticated formatting, such as XSL:FO rendering to PDF files, client-dependent transformations such as WML formatting for WAP-enabled devices, or direct XML serving to XML and XSL aware clients.

To do this, the Cocoon model divides the development of web content into three separate levels:

- ***XML creation:*** The XML file is created by the *content owners*. They do not require specific knowledge on how the XML content is further processed - they only need to know about the particular chosen "DTD" or tagset for their stage in the process. (As one would expect from a fully generic XML framework, DTDs are not required in Cocoon, but can be used and validated against). This layer is always performed by humans directly, through normal text editors or XML-aware tools/editors.
- ***XML processing:*** The requested XML file is processed and the logic contained in its logicsheet(s) is applied. Unlike other dynamic content generators, the logic is separated from the content file.
- ***XSL rendering:*** The created document is then rendered by applying an XSL stylesheet to it and formatting it to the specified resource type (HTML, PDF, XML, WML, XHTML, etc.)

3.8.3 Conclusion

The first version shows that the structuring and tools used for the Publishing Prototype are well suited for generating publication products. The performed tests have shown, that the setup and functionality of the prototype are capable of producing the publications in a reasonable timeframe.

However, there are still open questions like providing an easy to use user interface for the definition and application of LATCH based structuring, especially the category and hierarchy principles, and visualisation of highly structured data.

4 Design Model

The design model is an object model describing the realization of use cases, and serves as an abstraction of the implementation model and its source code. The design model is used as essential input to activities in implementation and test.

4.1 Purpose

This part describes the design model comprehensively, in terms of how the model is structured into packages and what classes are in the model. If you are using packages, the document shows the model structure hierarchically. The report can be used to describe the entire design model at different stages:

- During elaboration, such as when you have identified the first classes and their objects.
- During construction, when the design is complete.

This part can be used by various people interested in the design model, such as the software architect, use-case designers, designers, testers, reviewers, and managers.

4.2 REGNET Portal

4.2.1 Data Generation

4.2.1.1 Introduction

The Data Generation componet has to deal with 2 different kinds of usage:

1. importing external data (metadata and/or digital surrogates of real objects)
2. managing internal data (insert, edit, delete metadata)

ad 1)

The first task is now done via external tools. There are tools for transformation into proper data formats, for verifying data integrety and for storing data into databases/repositories. These tools were developed on demand and are dedicated to special input data and to the repositories where the data is stored.

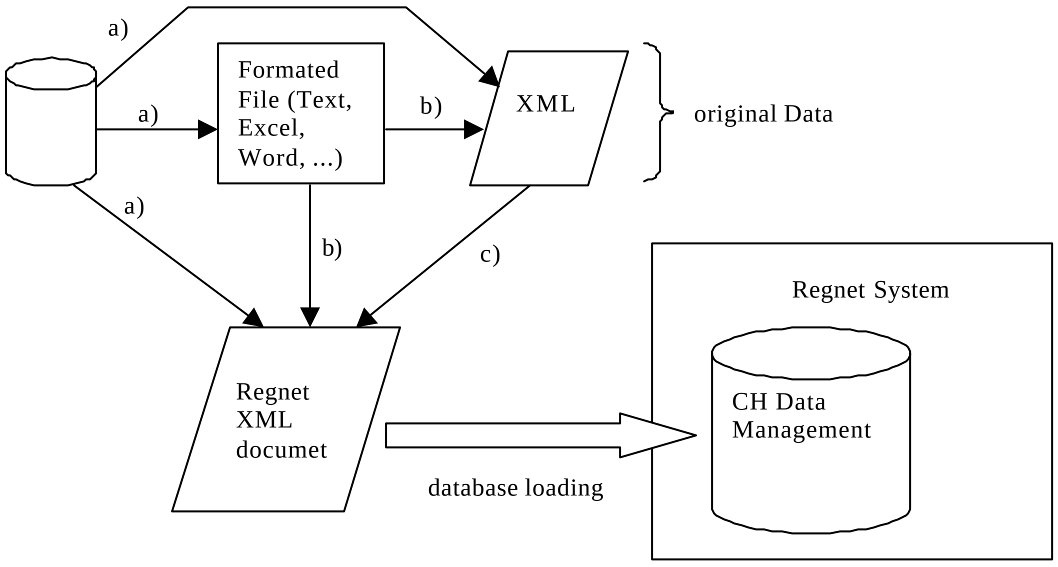


Figure 15: Data Transformations

All the arrows mark transformations which can be neccessary to lead content provider data over to the Regnet System:

a) Export Tools provided by database products and JAVA applications (e.g. jXTransform from DataDirect®) accessing the database via JDBC were used. With jXTransform you can specify SQL queries and transform the output into XM L structures.

b) Simple JAVA applications and VB scripts were used to transform formated files into XML structures.

c) Transformation between different XML structures was done by XSL stylesheet transformation.

ad 2)

This component is integrated into the portal site. The following 2 diagrams show activities for insert/edit and delete records from repositories.

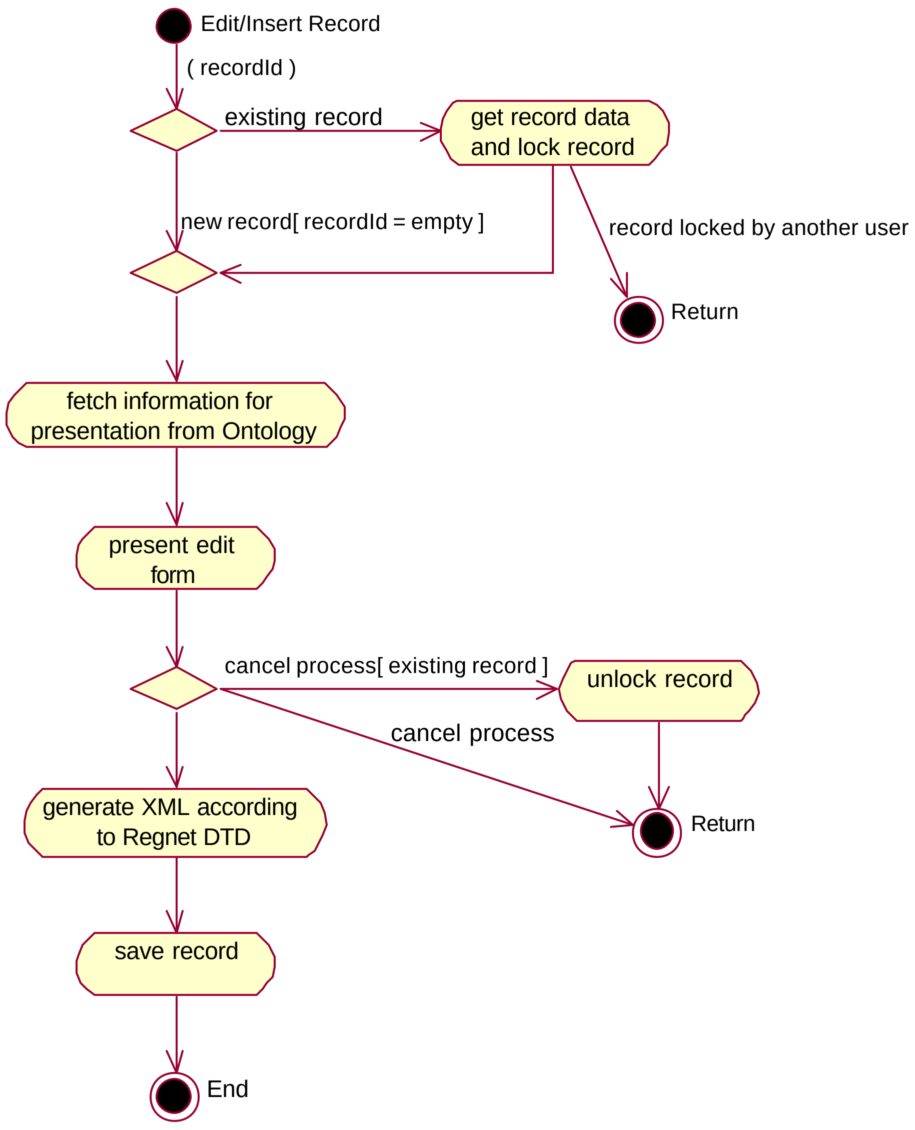

Edit/Insert Record
(recordId)
existing record
get record data and lock record
new record[recordId = empty]
record locked by another user
Return
fetch information for presentation from Ontology
present edit form
cancel process[existing record]
unlock record
cancel process
generate XML according to Regnet DTD
Return
save record
End

Figure 16: Activity Diagram – Insert / Edit Record

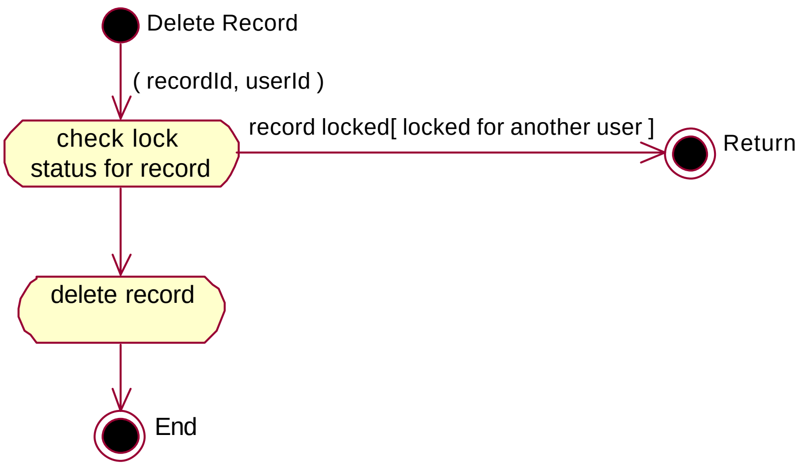

Delete Record
(recordId, userId)
check lock status for record
record locked[locked for another user]
Return
delete record
End

Figure 17: Activity Diagram – Delete Record

Note that the insert / edit record process is a sumptuous one. Beside the handling of concurrent access to records there has to be massive support from the Ontology System. The Data Entry From depends from the underlying database. In a first step the Data Generation Component has to fetch meta data about fields, more exact editable fields in a database. After building the Input form the data has to be transformed into the real database format because there can be fields which are not editable and fields which are dependent from other fields (calculated from other fields). Meta data for all these tasks (XSL stylesheets, XML Schemas for validation) have to be stored in the Ontology System.

4.2.1.2 Design model hierarchy

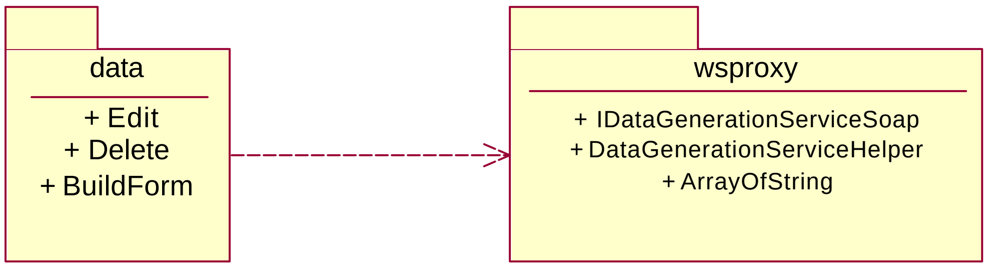


Figure 18: Package Diagram – Data Generation

The packages for the first prototype are very simple. One package handles the data generation tasks and the other one helps to connect to the DataGeneration Web service in the Reference System via SOAP protocol.

- `data`:
  The main package of the component. Responsible for building up the user interface for inserting / editing records.
- `wsproxy`:
  These are proxy-classes automatically generated with a tool from the GLUE API. The tool examines the WSDL description of a Web service and automatically generates proxy classes out of it. The package includes the proxy classes for the DataGeneration Web service located in the Reference System. The proxies and the GLUE API allow very easy access to the Web service via the SOAP protocol.

4.2.1.3 Diagrams of the design model

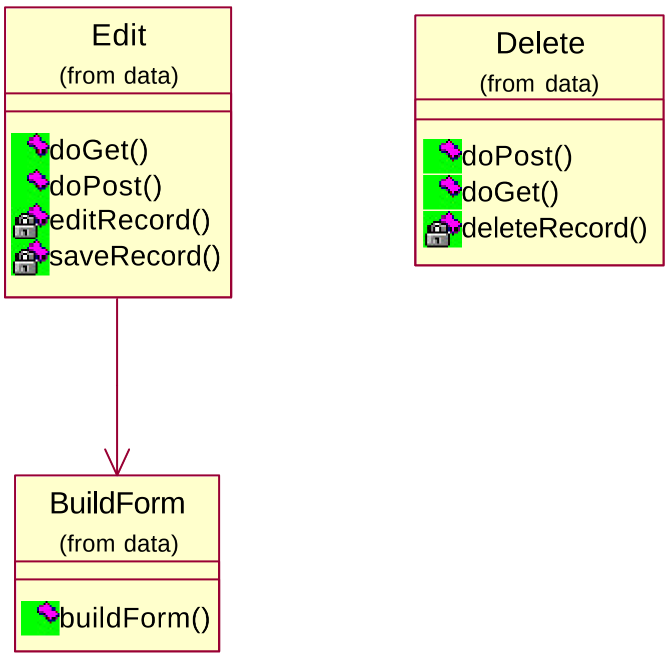


Figure 19: Class Diagram – Data Generation

The `Edit` class builds up the user interface for editing existing data or an empty form when a new records should be created. The `BuildForm` class handles the different user interfaces for different content from separate databases. It uses Meta-data files in XML format where information about database fields are stored.

The `Delete` class just redirects the delete requests to the Reference System.

4.2.1.4 The REGNET off line data entry system

The needs for the development of an off line data entry system, based on templates for the production of the meta data of cultural heritage content (files), are multiple, the main ones being:

- The collection of content examples at an early stage of the project (until the availability of on line tools)
- The introduction of new parameters corresponding with the REGNET-objectives related to thematic descriptions and e-Business.
- The possibility to set up a kind of collection management for those partners not disposing of a standard collection management system.
- The availability of data input facilities for those not possessing or not willing to use, on line connections.
- The offering of a local back up solution for the data delivered to the system.

Basics.

The spreadsheet approach.

In order to create a tool with a sufficient level of user friendliness for the content providers, not requiring new specific experience and at the same time offering a well-defined structure for the automated conversion of the generated data into the desired internal data structure, the spreadsheet paradigm was chosen. The wide spread use of spreadsheets and the possibility to integrate programming (macros) guarantee a sufficient level of flexibility to cope with the specific REGNET objectives.

The templates.

Different templates were defined containing the necessary fields and field names that correspond with the different data structure types used within REGNET. In the current version this pertains to four separate templates representing respectively the catalogue meta data, image meta data, object text meta data and thematic text meta data. Every template type resides in a separate sheet and all sheets belong to one workbook. One workbook contains only data of one content provi der.

The fields to be filled in are "vertically tagged" with a start tag and an end tag. This is similar to XML-tagging and eases the conversion from the spreadsheet to an XML-structure. At the same time this tagging-approach allows the insertion of multiple entries between the tags and the creation of new "tagged" fields without programming efforts. The "vertical tagging" was chosen in order to display all fields of one template on one full screen. The naming of the fields is as far as possible based on the AMICO-standard.

Data structure types.

Catalogue description (cd): this template allows the user to enter the most relevant meta data about a piece of art, including those meta data not foreseen in AMICO (such as price, etc.).

Image description (im): this template allows the user to enter the most relevant meta data concerning a digital image of a piece of art.

Object description (od): this template allows the user to enter the most relevant meta data concerning a (longer than a few lines) descriptive text concerning a piece of art. The smaller texts can be incorporated in the catalogue template.

Thematic description (td): this template allows the user to enter the most relevant meta data concerning a descriptive text treating thematic and/or contextual aspects of pieces of art.

Language.

A translation of original content (text files) into another language necessitates the production of a new corresponding meta data template expressed in that language. Although images remain the same in different languages, their meta data need also to be translated because they contain different "displayable" fields. E.g. the title can be used as a caption or the description how the image was produced can be made available to the user.

Guidance.

Every workbook has, besides the four template sheets, a fifth sheet, which contains a short explanation how to fill in the different fields with their respective constraints.

XML-conversion.

A spreadsheet-to-XML converter is developed to create the XML-structure of the catalogue, image, object and thematic descriptions to be integrated into the REGNET data structures.

Lay out.

In the following tables the general layout of the four templates is represented. They contain already real samples of the content of the SAINTS theme.

<catalogue_description>							
<identifier>	<language>						
tarx_angela01_cd_en	en						
</identifier>	</language>						
<title>	<object_type>	<mat_techn>	<dimension>	<dim_value>	<dim_unit>		
Saint Angela	glass door panel	glass	width	50	cm		
</title>	original	etch	height	200	cm		
	</object_type>	</mat_techn>	depth	0,2	cm		
			</dimension>	</dim_value>	</dim_unit>		
<description>							
</description>							
<location>	<author>	<author_org>	<creation_date>	<contributor>	<contrib_org>	<comment>	
Convent of the Ursulines; Tildonk Belgium	unknown	unknown	end 19th century				
</location>	</author>	</author_org>	</creation_date>	</contributor>	</contrib_org>	</comment>	
<relation_link>	<primary_keyw>	<secondary_keyw>	<rights>	<price_excl_vat>	<vat>	<currency>	
(1): tarx_angela01_od_cl2_sl1_en	saint	glass	Vereniging van de religieuzen Ursulinen van Tildonk vzw	N/A		Euro	
(2): tarx_angela01_im_rl2_sl3_en	angela	etch	</rights>	</price_excl_vat>	</vat>	</currency>	
</relation_link>	</primary_keyw>	</secondary_keyw>					

Original documenting and validation	<doc_author>	<doc_auth_org>	<doc_date>	<doc_validator>	<doc_val_org>	<d_val_date>	<d_comment>
	Sermeus, Rosette	Tarx nv Belgium	2002-02-16	Haesaerts, Vic	Tarx nv Belgium	2002-02-18	
	</doc_author>	</doc_auth_org>	</doc_date>	</doc_validator>	</doc_val_org>	</d_val_date>	</d_comment>
Document modification and validation	<mod_author>	<mod_auth_org>	<mod_date>	<mod_validator>	<mod_val_org>	<m_val_date>	<m_comment>
	</mod_author>	</mod_auth_org>	</mod_date>	</mod_validator>	</mod_val_org>	</m_val_date>	</m_comment>
</catalogue_description>							

<catalogue_description>

new template

</catalogue_description>

<object_description>							
<identifier>	<content_level>	<size_level>	<language>	<encoding>	<file_size>	<file_size_unit>	
tarx_angela01_od_cl2_sl1_en	cl2	sl1	en	doc	19	kb	
</identifier>	</content_level>	</size_level>	</language>	</encoding>	</file_size>	</file_size_unit>	
<title>	<description>						
Saint Angela							
</title>	</description>						
<resource_locator>	<author>	<author_org>	<creation_date>	<contributor>	<contrib_org>	<comment>	
tarx_angela01_od_cl2_sl1_en.doc	Sermeus, Rosette	Tarx nv Belgium	2001-12-10				
</resource_locator>	</author>	</author_org>	</creation_date>	</contributor>	</contrib_org>	</comment>	
<relation_link>	<primary_keyw>	<secondary_keyw>	<rights>	<price_excl_vat>	<vat>	<currency>	
(1): tarx_angela01_cd_en	saint	glass	Tarx nv Belgium	N/A		Euro	
(2): tarx_angela01_im_en	angela	etch	</rights>	</price_excl_vat>	</vat>	</currency>	
</relation_link>	</primary_keyw>	</secondary_keyw>					
Original documenting and validation	<doc_author>	<doc_auth_org>	<doc_date>	<doc_validator>	<doc_val_org>	<d_val_date>	<d_comment>
	Sermeus, Rosette	Tarx nv Belgium	2002-02-16	Haesaerts, Vic	Tarx nv Belgium	2002-02-18	
	</doc_author>	</doc_auth_org>	</doc_date>	</doc_validator>	</doc_val_org>	</d_val_date>	</d_comment>
Document modification and validation	<mod_author>	<mod_auth_org>	<mod_date>	<mod_validator>	<mod_val_org>	<m_val_date>	<m_comment>
	</mod_author>	</mod_auth_org>	</mod_date>	</mod_validator>	</mod_val_org>	</m_val_date>	</m_comment>

```
</object_description>
<object_description>
new template
</object_description>
```

4.2.2 Search and Retrieval

4.2.2.1 Introduction

The following activity diagram shows the steps during a users search request. The mechanism is simple. More sophisticated add-on's like 'search history', including previous search results and choosing different presentation formats are strongly related to user management. These features will be added in the next phase of the project and can be coverd with the term 'personalisation of search'.

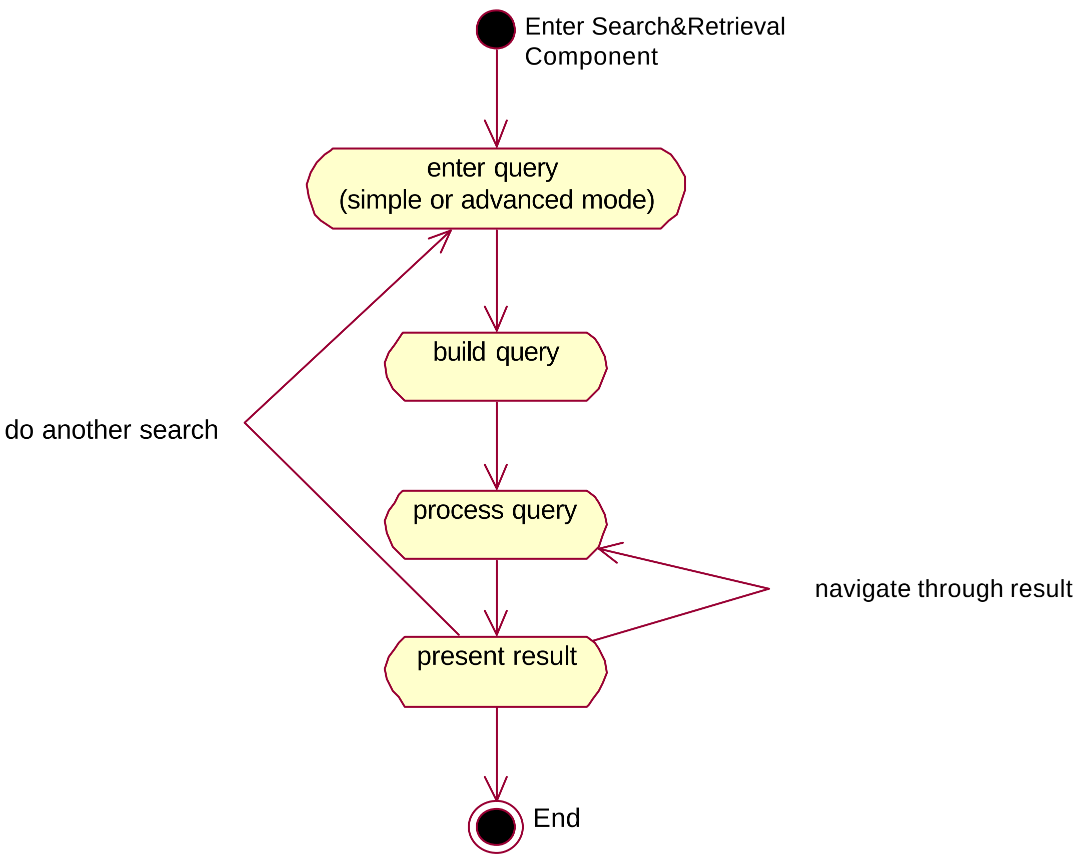


Figure 20: Activity Diagram – Search & Retrieval

After entering the Search & Retrieval component the user has to enter a query. Now there are 2 levels of query interfaces available: simple and advanced. But in future there can be several levels of query interfaces adapted to the users needs.

The query is transformed into a query format that the Reference System can understand in the 'build query' activity and afterwards sent to the Reference System. The results from the Reference System are transformed by the means of XSL stylesheet transformation and presented to the user. The presentation includes navigation through the current result set.

4.2.2.2 Design model hierarchy

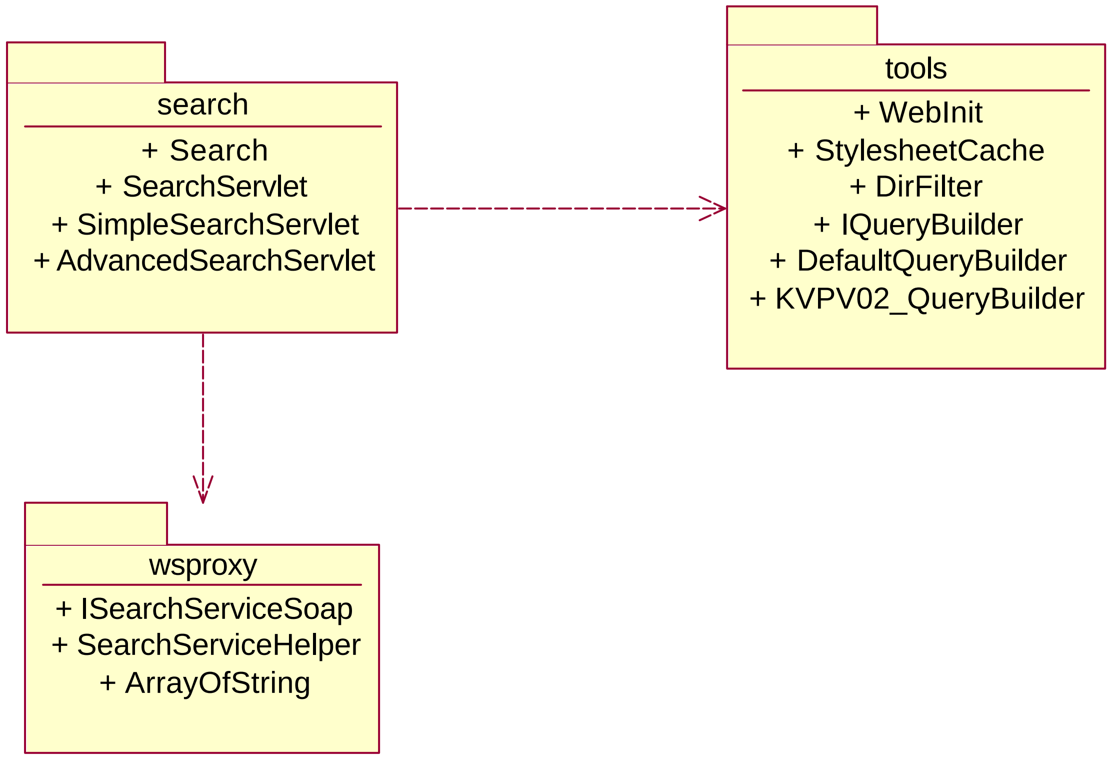


Figure 21: Package Diagram – Search & Retrieval

Package description:

- `search`:
  The main package of the component. Responsible for building up the user interface for queries and results.
- `wsproxy`:
  These are proxy-classes automatically generated with a tool from the GLUE API. The tool examines the WSDL description of a Web service and automatically generates proxy classes out of it. The package includes the proxy classes for the SearchService Web service located in the Reference System. The proxies and the GLUE API allow very easy access to the Web service via the SOAP protocol.
- `tools`:
  This package contains useful support classes for the `search` package and for other packages as well. It includes Initialisation support, a stylesheet cache which is loaded at application startup and classes responsible for building up queries.

4.2.2.3 Diagrams of the design model

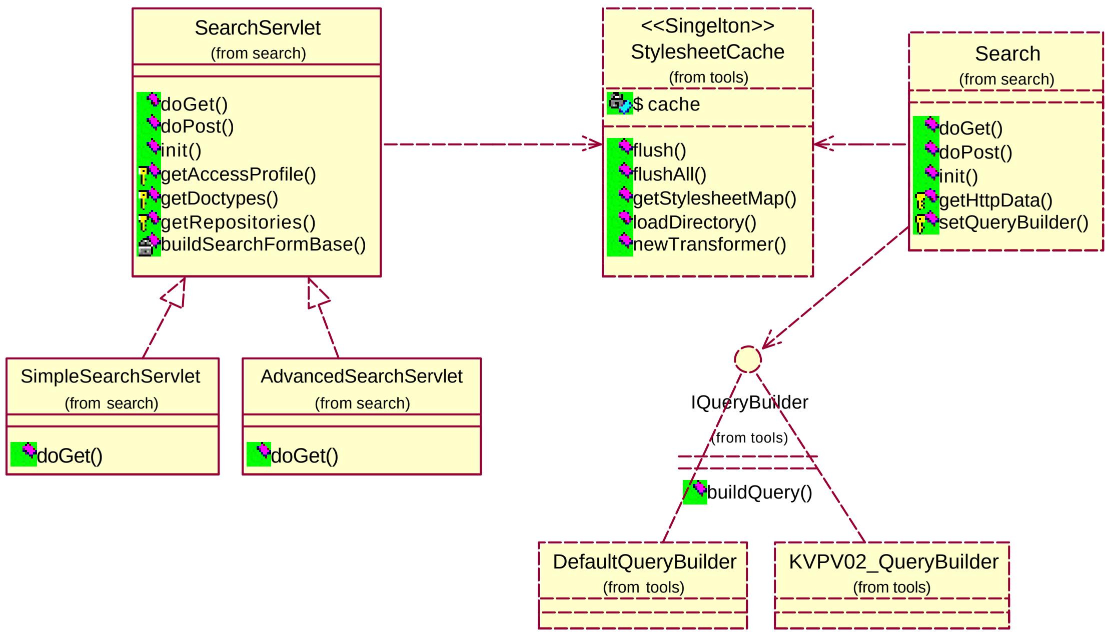


Figure 22: Class Diagram – Search & Retrieval

The `SearchServlet` class is responsible for building the query interface for the user. After submitting a query the `Search` class takes over the task of sending the search to the Reference System and presenting the results. Both classes use the `StylesheetCache` which holds preprocessed XSL stylesheets for their presentation tasks. The `StylesheetCache` is loaded during web server startup. It reads all stylesheets from a specific directory.

For building up the query before sending it to the Reference System a specialized `QueryBuilder` class is used.

4.2.3 E-Business

4.2.3.1 Introduction

Functional architecture was developed according to the optimised set of the functional components, which the total system has to have.

The total system has three levels of the components:

- Presentation level
- Core level
- Remote processing and data storing level.

Each of the mentioned level includes some subsystems, which have their own functionality and dedicated to fulfil different actions.

Scheme of the components and interconnections between them is presented on the next figure.

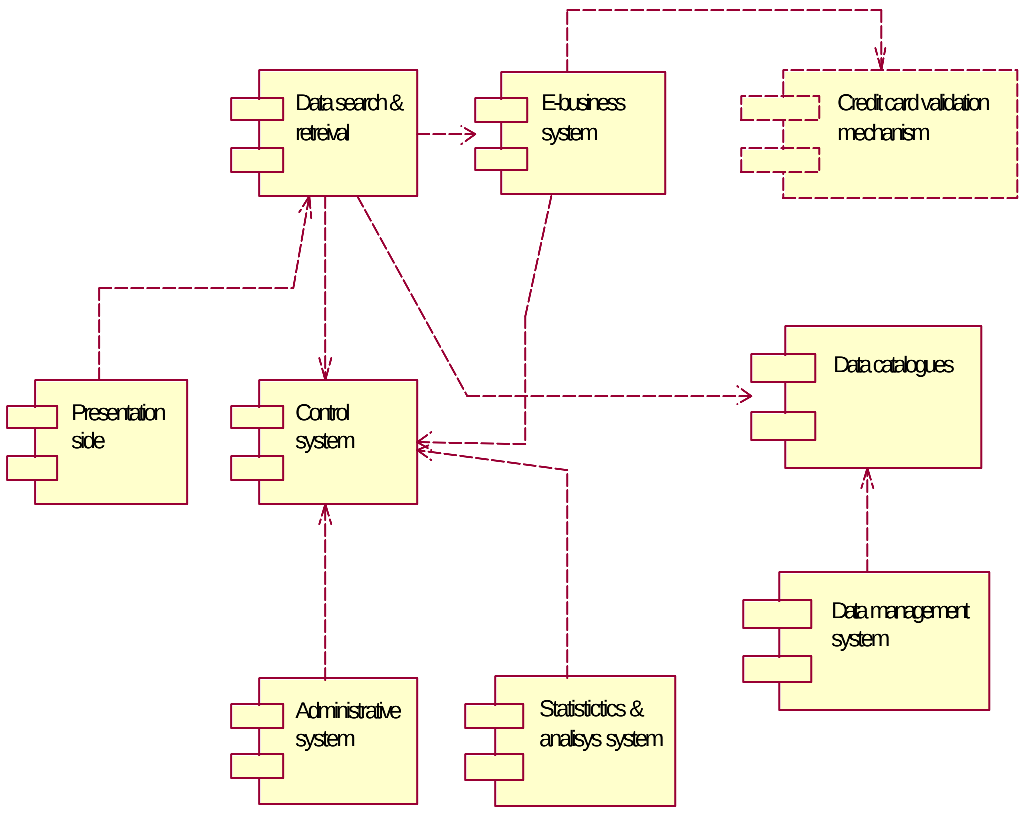


Figure 23: Scheme of components and interconnections between them.

Presentation level,

has the only one subsystem, named

Presentation side.

This subsystem meant for the present all the necessary text and graphical information to the user (customer). The PHP methods and the standard features of the browsers implement these functions. This subsystem also provide the customer all the necessary interactive input forms and interactive interfaces.

Core components (subsystems),

contain all the necessary facilities to execute successful business transactions. This level contain the following subsystems:

Data search & retrieval (DSR).

This component is one of the most important. It is meant for searching the required information in the remote sources of data. This component includes YAZ client, which is prepare and send the RPN query to the remote databases. It is also receives the responses from them and presents received information to the customer.

After receiving the words for the search, DSR interconnects with a control system to receive all the available targets (URLs) to search.

This subsystem transfers the chosen products to the E-business system.

E-business system (EB).

This component is meant for the execution of the business transactions. It allows user to create the orders, to change them, choose the method and accomplish the payment for the products. It is strongly related with a mechanism of the credit card validation.

It is also necessary to appoint that all the input information about the credit card should be done in a protected order, using the SSL protocol. This allows us to protect the customers from the unapproved permission to the privet information of his credit card i.e. its number, expiration date etc.

EB allows having a permanent control of the conditions of the processing the orders. This action includes the full control of the dates of the order creation, sending it to the customer and the date of receiving it by the client.

EB system also related with a Control system. It puts the all information about the transaction to the database **shop_admin**, then this information will be used to execute analysis of the customer's preferences and other useful parameters.

Control System (CS).

CS mostly meant for collecting all kind of data, which appeared in the total system. This system contains the database **shop_admin**, which has some different tables. Each of the table contains information for the different subsystems. As it was mentioned earlier, as an example we can say that tables BASKED and ORDERS, which will be described in full later, meant for the SA system.

The same situation is with other tables.

Administrative System (AS).

AS helps the responsible person to control the condition and actuality of the reference information, which stores in the database. This information mostly concerns to the suppliers and targets (URLs) to search. This system allows us to add, edit or delete information about the supplier. The other useful feature of this subsystem is that it gives possibility to the supplier to has not only one database of products but as many as he wants, and edit, add or remove this separate targets (URLs of the remote databases).

Statistics & Analysis system (SA).

Last system in the Core level is the SA system. This system is not vitally important to the total system but it could help the organization to construct the successful commercial policy. This system will allow managers to view and analyse the statistics of the business transactions and customer's preferences.

Remote processing and data storing level.

This level mainly located on the remote hosts and interconnects with a Core layer components by the z39.50 protocol. The only exception is a credit cards validation mechanism.

Data Catalogues (DC).

Main system of this layer is a DC. It contains the database, which stores the information of the product, such as information of its name, price, description etc. DC also contains web server, which stores the images of the products. Table ITEMS from the **zgoods** database contains the references on the images of the products. These references are transferred to the client's browser, which will load these images on one's own.

The most important thing is to associate the right place of the images storage with references on them.

Data Management system (DM).

DM has the similar functions as AS of the Core Level. The main difference between them is that DM meant for the managing the information about products. Administrator also has an ability to add, edit and delete the information about the product. As a necessary option the function that allows administrator to upload the files with images of the product was developed. It is also possible to replace the image of the product, new one will has the same name as a previous.

These components can fulfil different functions, but in general we can define the most common and group them by the components. It means that not only one component has an ability to execute exact function, some other also have this ability.

That is the reason to collect these functions in the few amounts of schemes to represent them in the report.

The following table presents the all-main use cases for the total system.

Use Case ID	Description
UC1	Receiving the query from the user.
UC2	Transfer query words into the RPN query.
UC3	Searching via the remote product catalogues.
UC4	Return results of searching to the presentation layer.
UC5	Send results of UC2 to UC4. Preparation for the sending the RPN query.
UC6	Transform XML result of the query to the HTML format.
UC7	Creation of the order to purchase the chosen products.
UC8	Choose the products, which was received from the UC4.
UC9	Send already made order to suppliers e-mail.
UC10	Process of update information, which stores in the shop_admin and zgoods databases. Ability to edit the user's profile and wish list.
UC11	Input the new information instead of the already existed.
UC12	Output message that update was successful.
UC13	Inserting of the new information into every database, or registration a new user.
UC14	Request new information for the chosen database, or new information about the user.
UC15	Output message that insertion was successful.
UC16	Process of delete information, which stores in the shop_admin and zgoods databases. Ability to delete the user's wish list.
UC17	Request information to delete from the chosen database, or information about the user's wish list.
UC18	Output message that information was removed.

Table 1: e-Business Main Use Cases

4.2.3.2 Design model hierarchy

The following model provide a thorough understanding of the packages that are created in the E-Business subsystem.

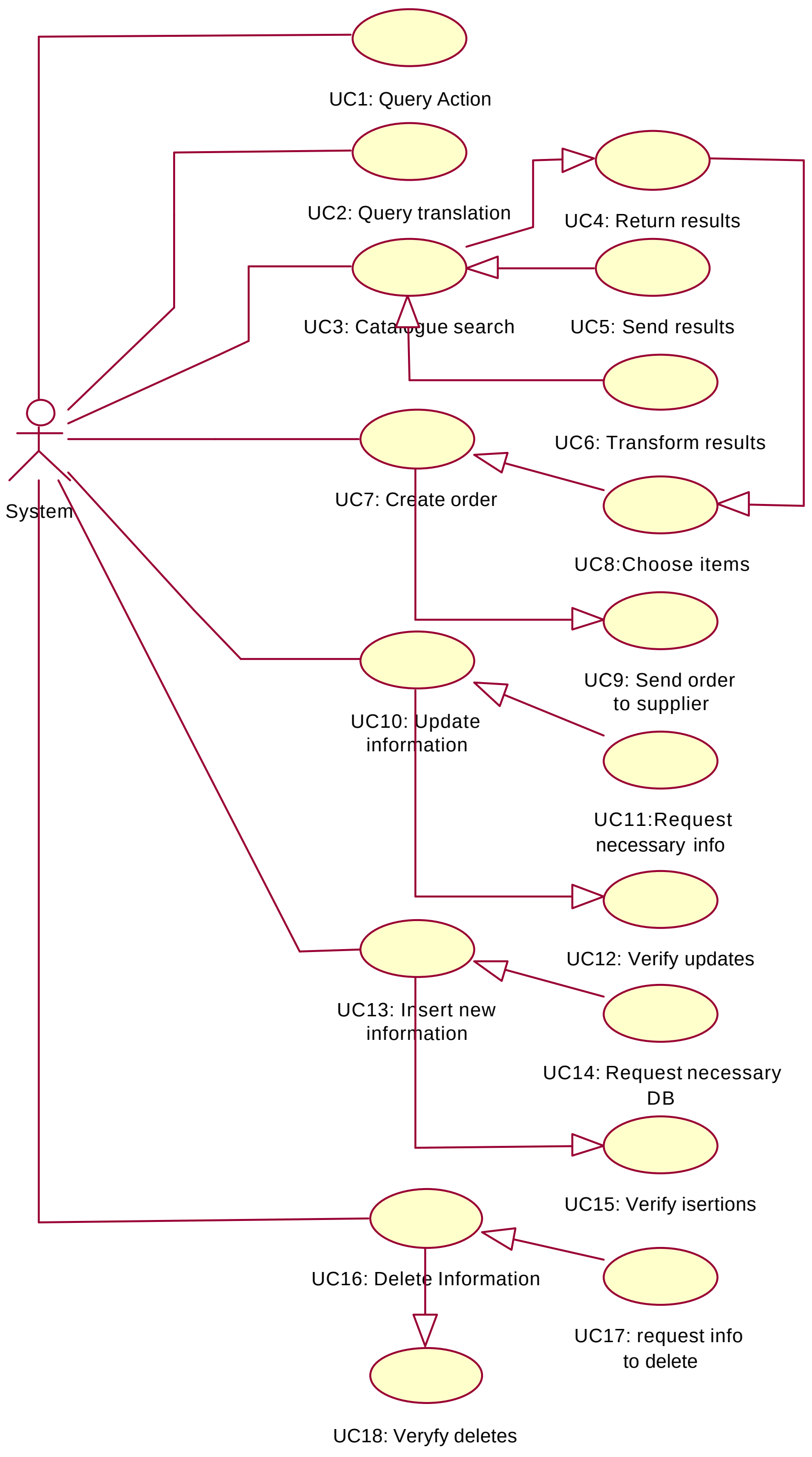


Figure 24: Use Case Diagram – e-Business

The E-Business subsystem is not differed to the normal procedures that are used to justify the overall implementation phase. It is more convenient for the reader to understand the functionalities that the E-Business subsystem supports and the exact procedure that the interactions are realised.

We consider the specific hierarchy as interactions of the system with the end-user and we are in line with the main concept of the specific document.

4.2.3.3 Diagrams of the design model

The diagrams of the design model are functional flows of the work processes that the subsystem supports and they also represent the exact flow that messages are exchanged between the user and the system. In order to justify the aforementioned issues, specific figures are presented that detail the basic diagrams of the design model.

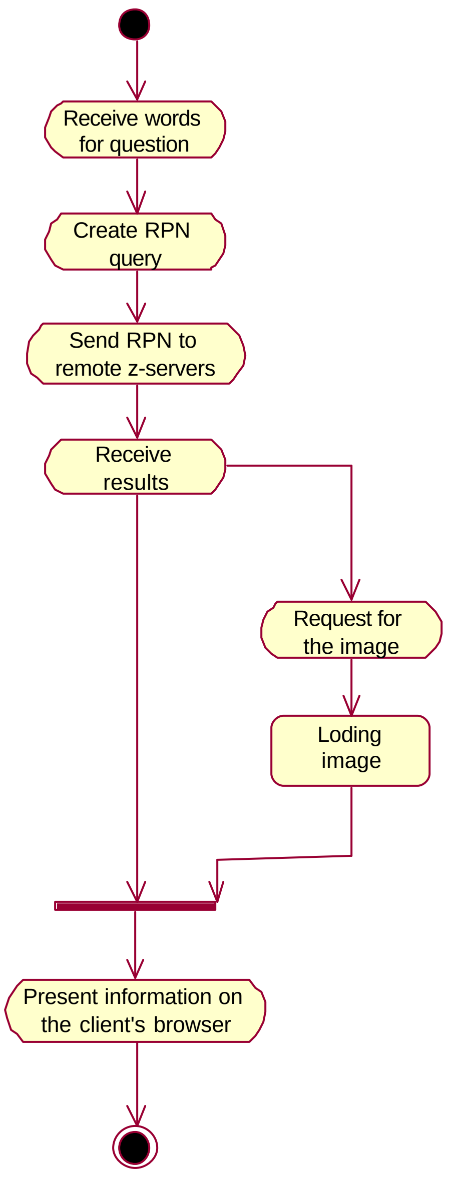

Receive words for question
Create RPN query
Send RPN to remote z-servers
Receive results
Request for the image
Loding image
Present information on the client's browser

Figure 25: Activity Diagram – e-Business search functionality

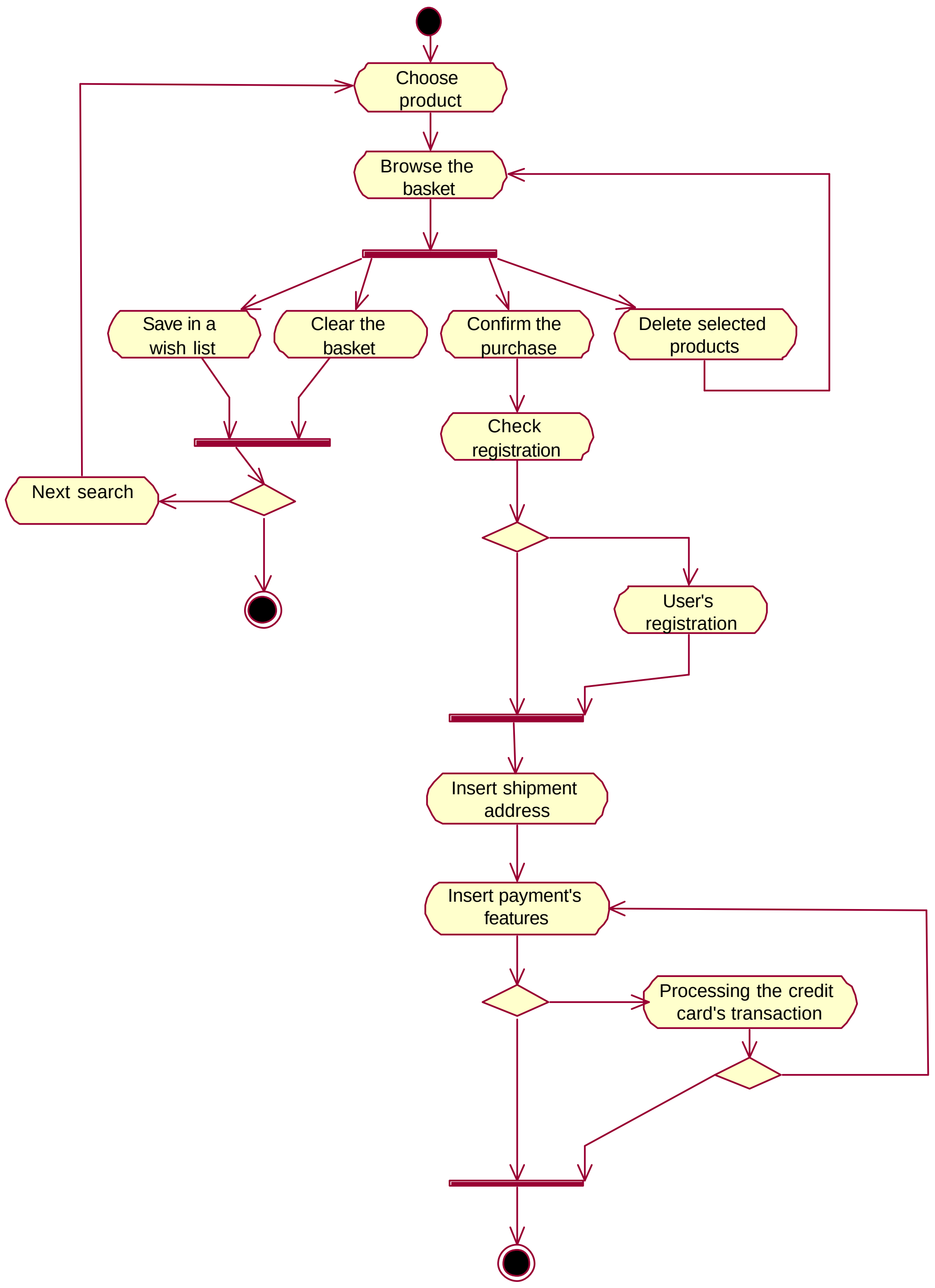


Figure 26: Activity Diagram – e-Business Order

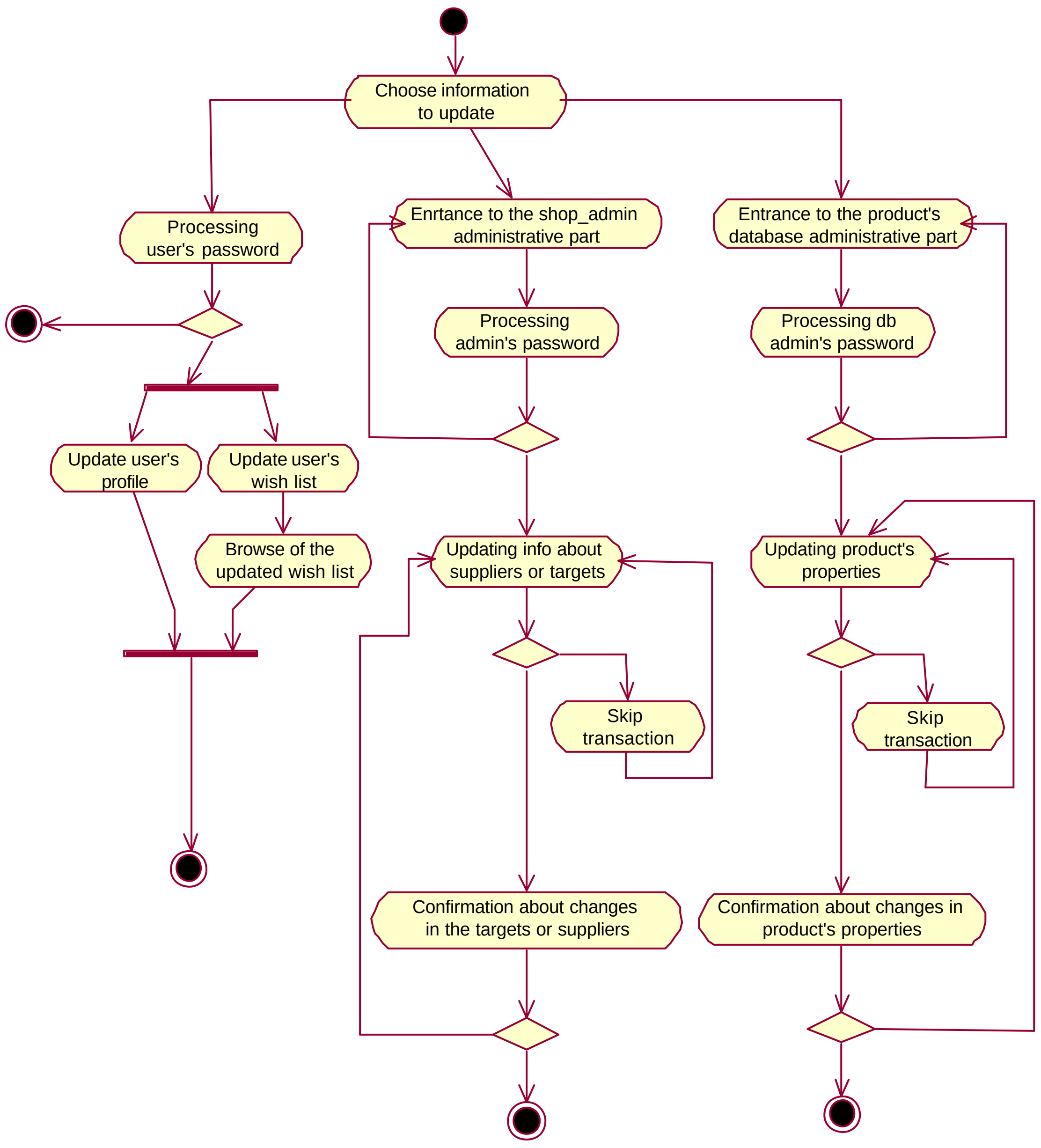

Choose information to update
Processing user's password
Enrtance to the shop_admin administrative part
Entrance to the product's database administrative part
Processing admin's password
Processing db admin's password
Update user's profile
Update user's wish list
Browse of the updated wish list
Updating info about suppliers or targets
Updating product's properties
Skip transaction
Skip transaction
Confirmation about changes in the targets or suppliers
Confirmation about changes in product's properties

Figure 27: Activity Diagram - Update Catalogues

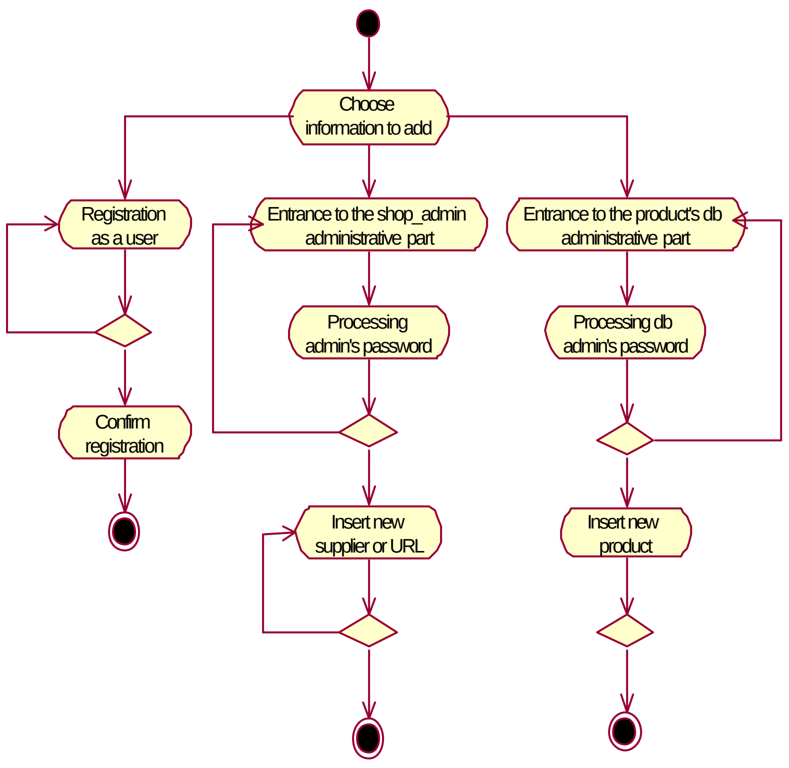

Choose information to add
Registration as a user
Entrance to the shop_admin administrative part
Entrance to the product's db administrative part
Processing admin's password
Processing db admin's password
Confirm registration
Insert new supplier or URL
Insert new product

Figure 28: Activity Diagram – Insert Information

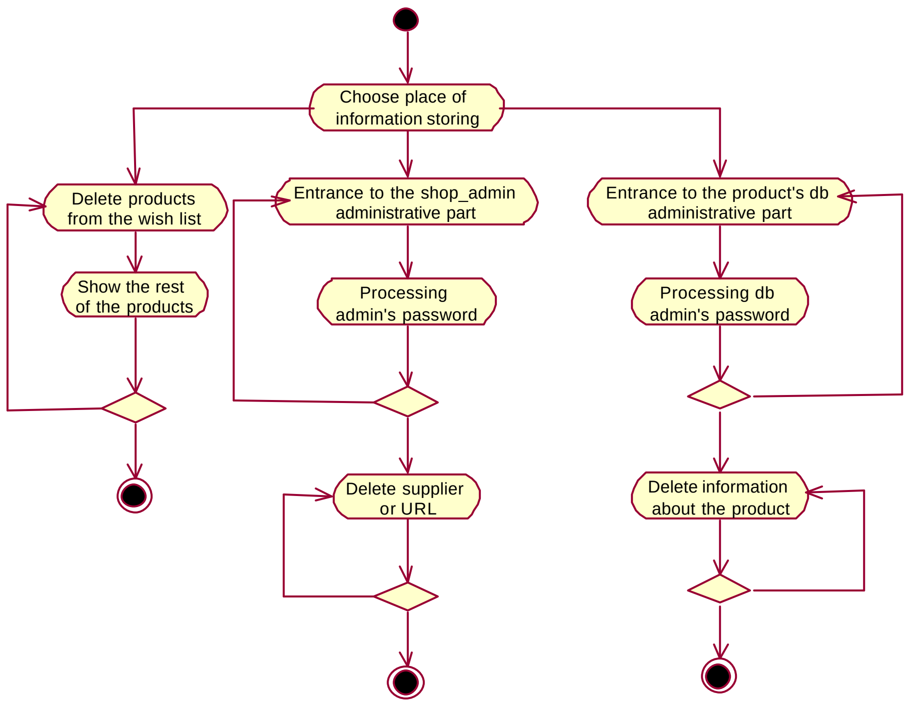

Choose place of information storing
Delete products from the wish list
Entrance to the shop_admin administrative part
Entrance to the product's db administrative part
Show the rest of the products
Processing admin's password
Processing db admin's password
Delete supplier or URL
Delete information about the product

Figure 29: Activity Diagram - Delete Information

The above provided diagrams describe in great detail all the necessary implemented procedures that support the exchange of documents and relevant information regarding content and E-Business services.

It is important to emphasise on the fact that the demonstrator is able to provide all the aforementioned activities and the elaboration of the integration of the two main E-Business relevant systems (E-Business and PCM)

4.3 Cultural heritage data management

4.3.1 Repository management

The Repository Management uses only existing tools and is accessable via common protocols (FTP and HTTP). Up to now no additional components have to be developed for this subsystem.

4.3.2 Reference system

4.3.2.1 Introduction

The 2 parts of the Reference System are

a) Data Generation sub-component

b) Search sub-component

Following are activity diagrams showing basic activities for these 2 subcomponents:

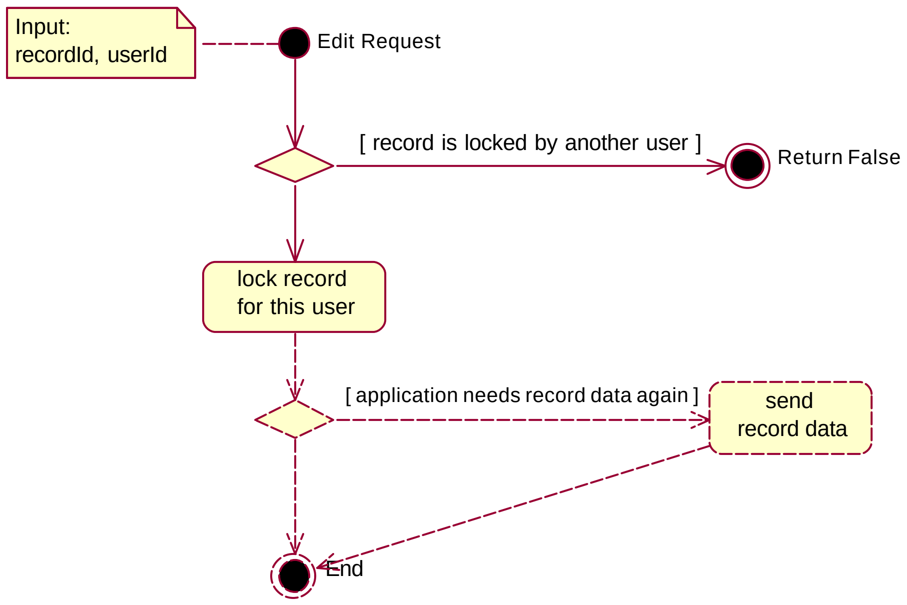

Input:
recordId, userId
Edit Request
[record is locked by another user]
Return False
lock record for this user
[application needs record data again]
send record data
End

Figure 30: Activity Diagram – Edit Record

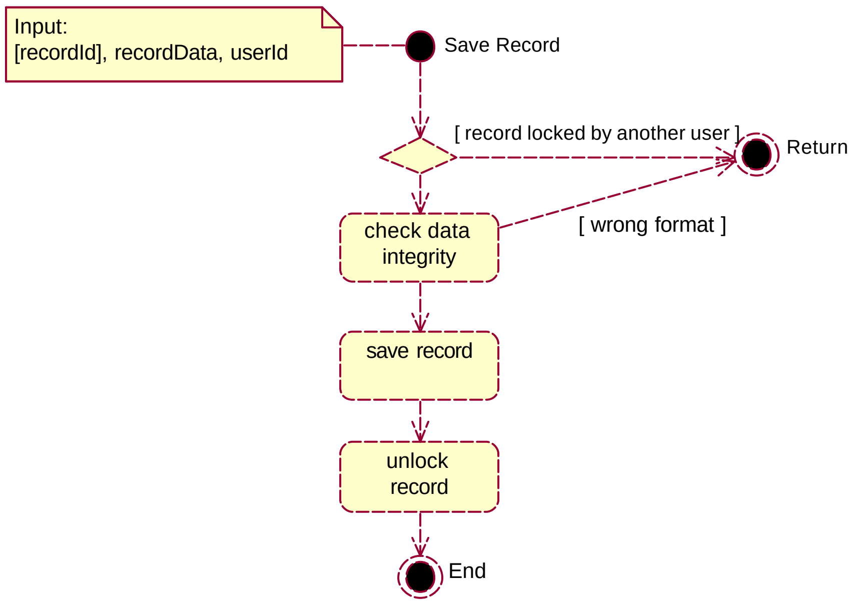


Figure 31: Activity Diagram – Save Record

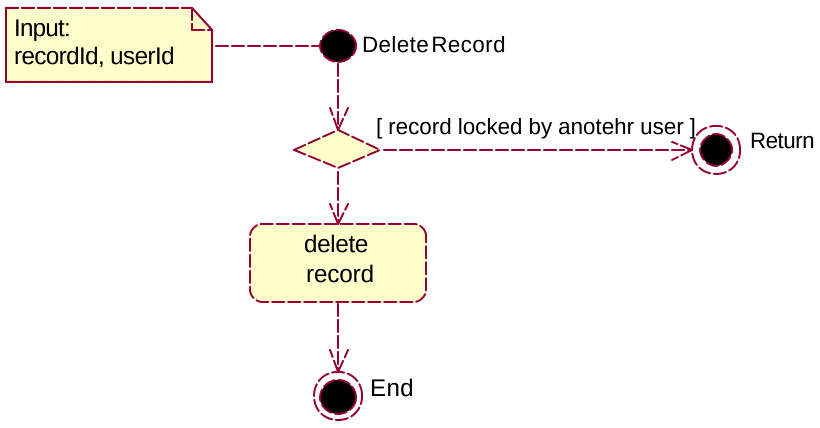


Figure 32: Activity Diagram – Delete Record

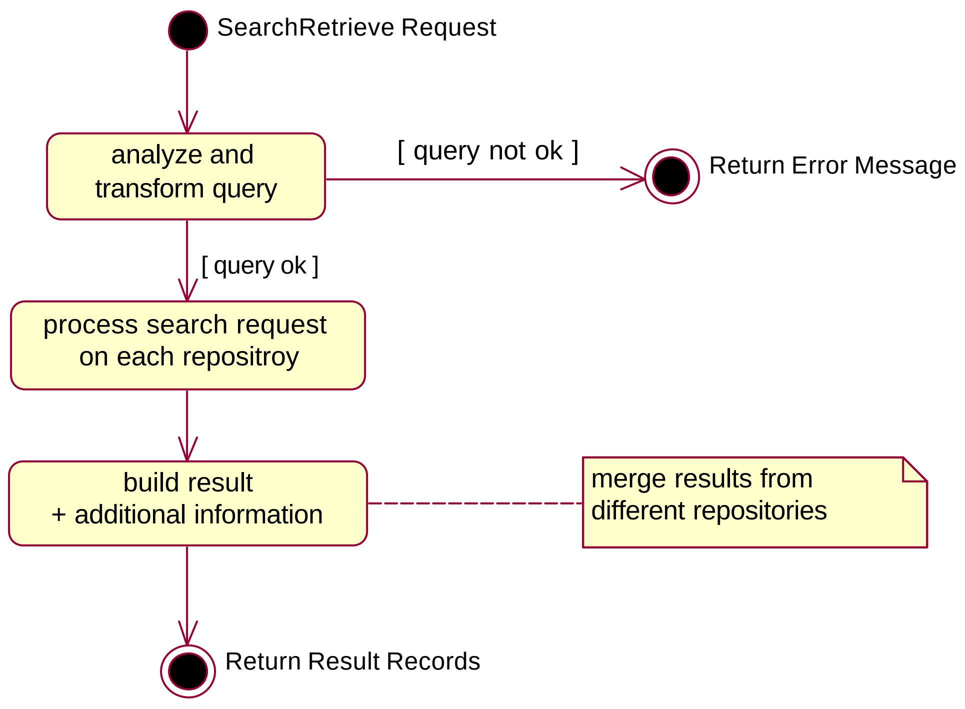

SearchRetrieve Request
analyze and transform query
[query not ok]
Return Error Message
[query ok]
process search request on each repositroy
build result + additional information
merge results from different repositories
Return Result Records

Figure 33: Activity Diagramm – Search

4.3.2.2 Design model hierarchy

The package structure of the Reference System is as follows:

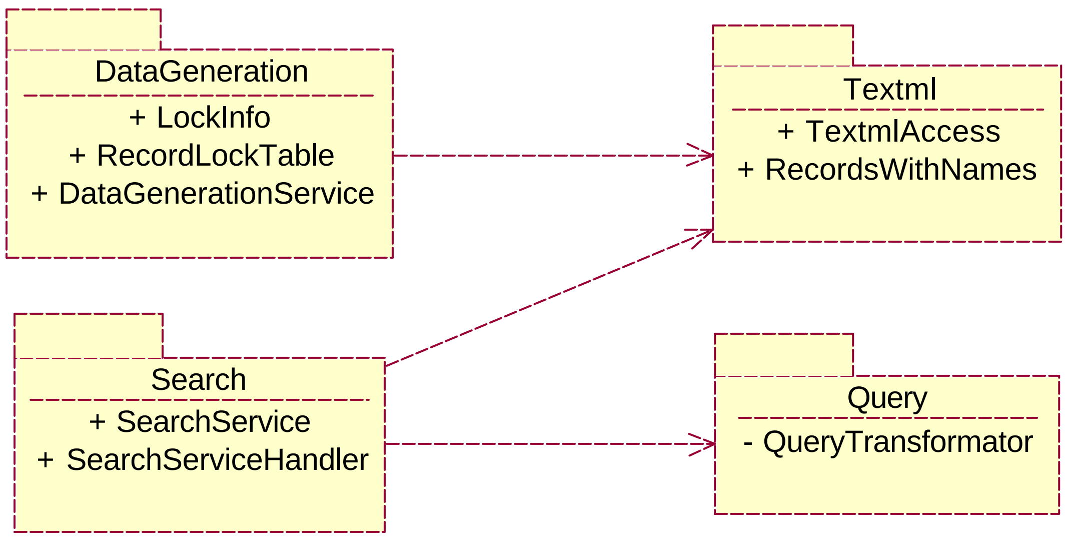


Figure 34: Package Diagram – Reference System

<u>DataGeneration:</u>

The DataGeneration package contains the Web service class (`DataGenerationService`) and two helper classes which are used to manage locks on single records and to link them to users known within the Regnet System.

<u>Search:</u>

The Search package contains the Web service class (`SearchService`) and the `SeachSearviceHandler`. The `SeachSearviceHandler` is responsible for handling the single TextmlAccess classes and therefore controls the whole search process.

Textml:

The Textml package contains only one class, the `TextmlAccess` class. This class can be seen as wrapper class. One `TextmlAccess` class establishes the connection to one Textml database and uses Textml API's to achieve this.

Query:

The `QueryTransformator` class transforms the query sent from the client into an query format which is compliant to the Textml Query-DTD.

4.3.2.3 Diagrams of the design model

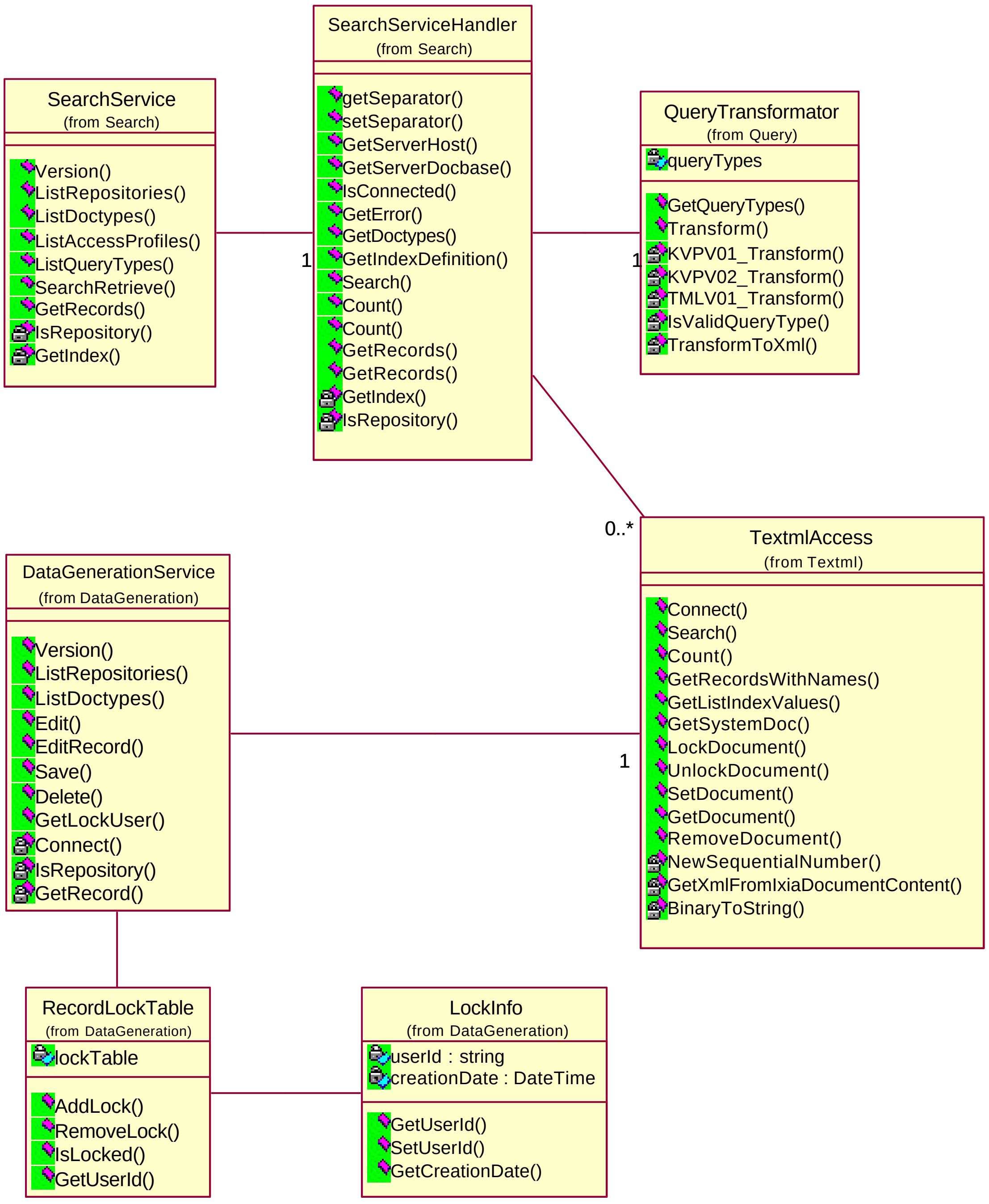


Figure 35: Class Diagram – Reference System

The central classes are the 2 Web services, `SearchService` and `DataGenerationService` on the one hand and the `TextmlAccess` class on the other hand. The Web services are the SOAP interfaces to the Reference System and they are described in more detail in section 2.2.2.2 Reference System. The `TextmlAccess` builds the interface to the Textml-Server which is the only database management system used in the 1st prototype version of Regnet.

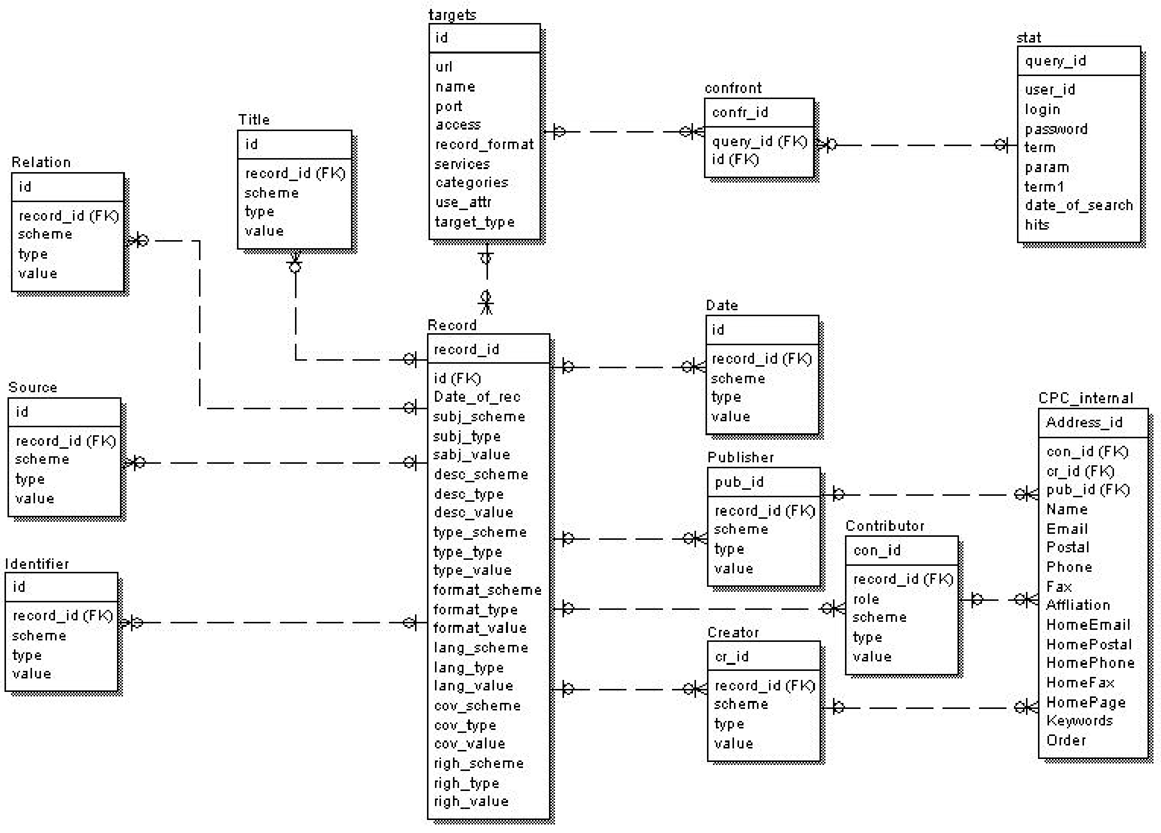
From the searching point of view the SearchServiceHandler is the class which enables the distributed seach facilities of the system. Search requests are distributed to the databases that are connected to this Web service and the results are merged together to one response there.

4.4 E-Business data management

4.4.1 Catalogue management

4.4.1.1 Introduction

The design model that we used for the warehouse that support the Catalogue Management is the following:



The final step before the building of the information model of the PCM is to define the other additional (not the elements of the DC) elements will be stored in the warehouse.

First what is needed is an information abut remote targets. This information is very useful for the system. It is needed to create the RPN query to the distributed catalogues (targets). The most new information about the remote targets has been provided by Reference System (RS). But to enhance the process of search PCM has to have this information inside its own database, to not call to remote subsystem, because it requires additional time. Next necessary information is information about the user. System has to have an ability to recognize the user as permanent or new to provide him/her the most convenient format of output, and to has necessary statistics about each user and each query.

In the process of modelling all the DC elements was analysed together with theirs qualifiers. There are some elements, which might be met more than one time in one DC record. These elements were separated to the different tables. This helped to avoid superfluity in the database. Table confront bind the information about the targets and already made queries, it will help to define which records of the DC metadata was received from the exact target, and from which target was received exact record.

More over by this information structure it is possible to check all the queries from the exact user and define the responses on these queries.

Information about the user is received from the Search and Retrieval (SR) subsystem. If there is no information about the user, the response of the query will be marked as *anonymous*.

Another aspect is a query to the databases, which contain information about the commercial products. This information will be received from the targets defined as *business* in the field *target_type* in table targets. This information may changes very quickly, that is why it is no reason to store the cached commercial data therefore each query from the e_Business subsystem will be immediately sent to remote targets.

All the DC elements in table Record are single elements. That means that each of these elements will be met only ones in each DC record. Other DC elements, which was picked out into the different tables may has many values. Each value of the exact record is bind with the rest values of the DC record by the unique identifier *record_id.*

Input and Output interfaces of the PCM

?	Name of the element	Input/Output element	Description
1	Identifier	Output	Unique identifier, for permit to the RS.
2	RepositoryURL	Input	URL or the remote Product Catalogue. Needed to perform distributed search.
3	RepositoryTrgetName	Input	Element should be used in the administrative tasks, to make the work with Repository's names more convenient. (Not to work with an IP addresses).
4	RepositoryDBName	Input	Defines the exact database name.
5	RepositoryDBPort	Input	Defines the exact database port.
6	RepositoryAccess	Input	Shows us the level of access to the target (repository).
7	RepositoryRecordFormat	Input	Displays the available format of the records.
8	RepositoryServices	Input	Displays the available services on the z39.50-server
9	RepositoryCategories	Input	If repository has the shared categories of the stored information.
10	RepositoryUseAttributes	Input	Displays which Use attributes it contain

Table 2: Interface of PCM & RS

?	Name of the element	Input/Output element	Description
1	Login	Output	User login. Meant to find his private information.
2	Password	Output	User password. Meant to find his private information.
3	UserName	Input	User name
4	TypeOfTerminal	Input	Type of the terminal, on which we have to send the result of search.
5	PreferableFormat	Input	Preferable format of the records to be presented to the user.

Table 3: Interface of PCM & KB

?	Name of the element	Input/Output element	Description
1	UserId	Input	User who made a query
2	QueryId	Input	Query number (Unique)
3	Term	Output	Search was executed by this term(s)
4	DateOfSearch	Output	Date of the query execution
5	RepositoryId	Output	Id of the repository where the information was found.
6	RepositoryDBName	Output	Names of the databases where the information was found.
7	RepositoryDBPort	Output	Port number of the database, mentioned in line 6.
8	NumberOfHits	Output	Number of hits, received in the response.

Table 4: Interface of PCM & DG

?	Name of the element	Input/Output element	Description
1	User_id	Input	User who have done the query
2	Login	Input	User's login
3	Password	Input	User's password
4	Field	Input	Search by which field.
5	Term	Input	Term for search (might be one or two)
6	ParameterField	Input	Logical parameters AND, OR
7	ShowHits	Input	Number of hits, user wish to see.
8	PresentationType	Input	Type of the presentation (brief or full)
9	Creator	Output	DC creator or producer from RDBMS
10	Title	Output	DC title or name of the resource
11	Publisher	Output	DC publisher
12	Identifier	Output	Reference to the resource.

13	Subject	Output	DC subject
14	Description	Output	DC Description.
15	Contributor	Output	DC contributor.
16	Date	Output	DC date
17	Type	Output	DC type
18	Format	Output	DC format
19	Source	Output	DC source
20	Language	Output	DC language
21	Relation	Output	DC relation
22	Coverage	Output	DC coverage
23	Rights	Output	DC rights

Table 5: Interface of PCM & SR

?	Name of the element	Input/Output element	Description
1	ItemName	Input	Item name for the new search
2	ItemId	Input	Item Id for the already known item
3	TargetURL	Input	URL where the information about item stored
4	TargetPort	Input	Port of the z3950 server.
5	DBName	Input	Name of the database, where the information about the product stored.
6	Price	Output	Price of the found product
7	PossibleDiscount	Output	Discount of the found product
8	Producer	Output	Producer of the product
9	Destributor	Output	Distributor of the product

Table 6: Interface of PCM & EB

The aforementioned interfaces and data exchanges between the subsystems and the PCM provide the exact overview of the already developed demonstrator that extend the management of the electronic catalogues in a powerful tool that collaborate also with the B2B scenario that is designed and implemented for the REGNET project.

4.4.1.2 Design model hierarchy

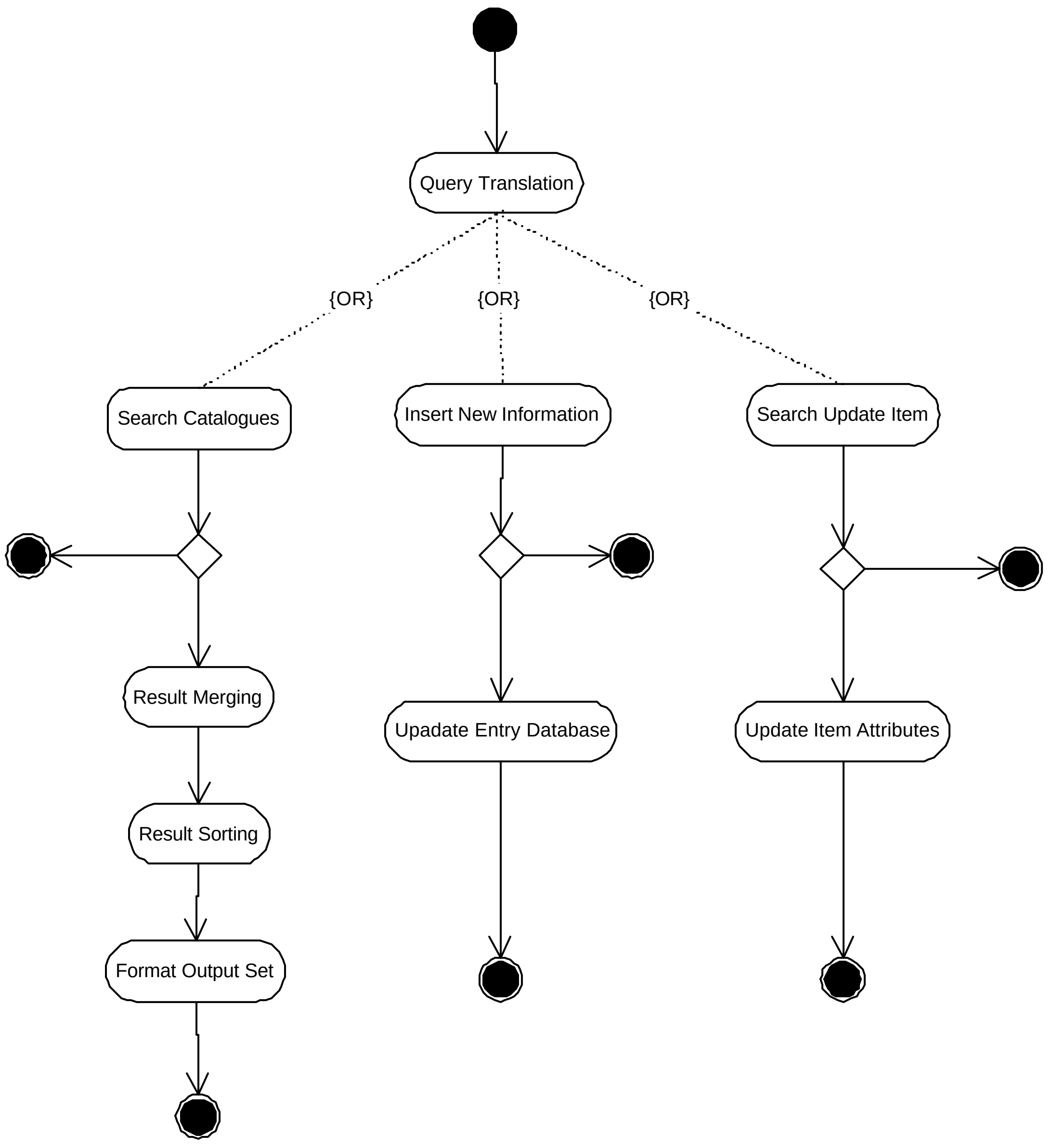


Figure 36: Product Catalogue Management model of the activities and the components

The necessary elements and procedures of the aforementioned model are clearly seen in the above figure and we are able to distinguish the three main functionalities of the components.

1) Provide access to the distributed catalogues and inform the interested user about the services and the products that are offered by the providers.

2) Provide the ability to each supplier to manage the content of the catalogues automatically.

2a) Update the attributes of the catalogues

2b) Insert new products or services to the stored catalogues.

4.4.1.3 Diagrams of the design model

The activity diagrams of these processes are presented in this section.

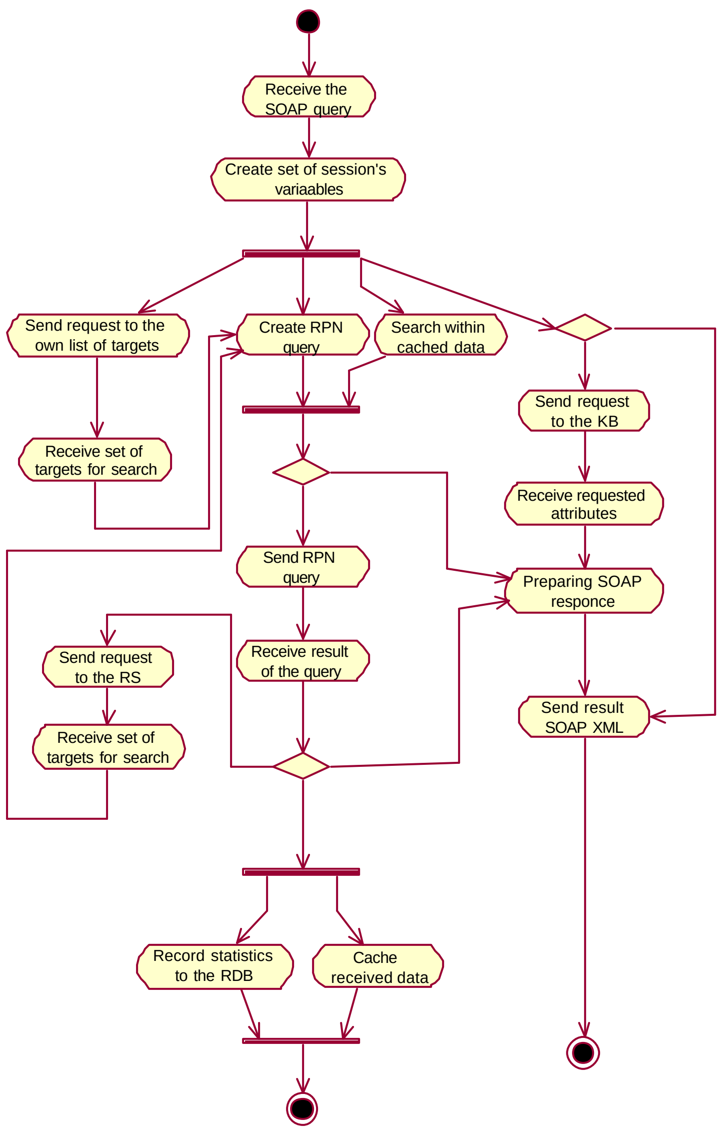

Receive the SOAP query
Create set of session's variaables
Send request to the own list of targets
Create RPN query
Search within cached data
Send request to the KB
Receive set of targets for search
Receive requested attributes
Send RPN query
Preparing SOAP responce
Send request to the RS
Receive result of the query
Send result SOAP XML
Receive set of targets for search
Record statistics to the RDB
Cache received data

Figure 37: Activity Diagram - Search and retrieval CH data

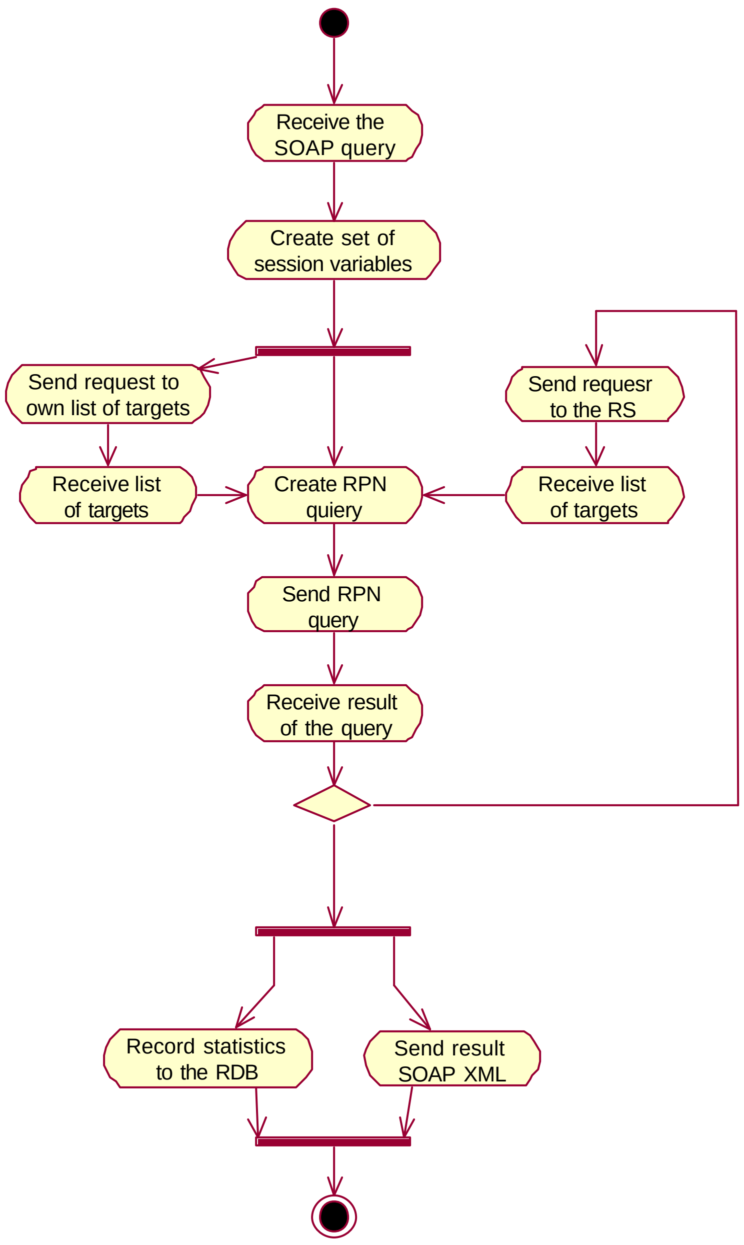


Figure 38: Activity Diagram - Search and retrieval commercial data

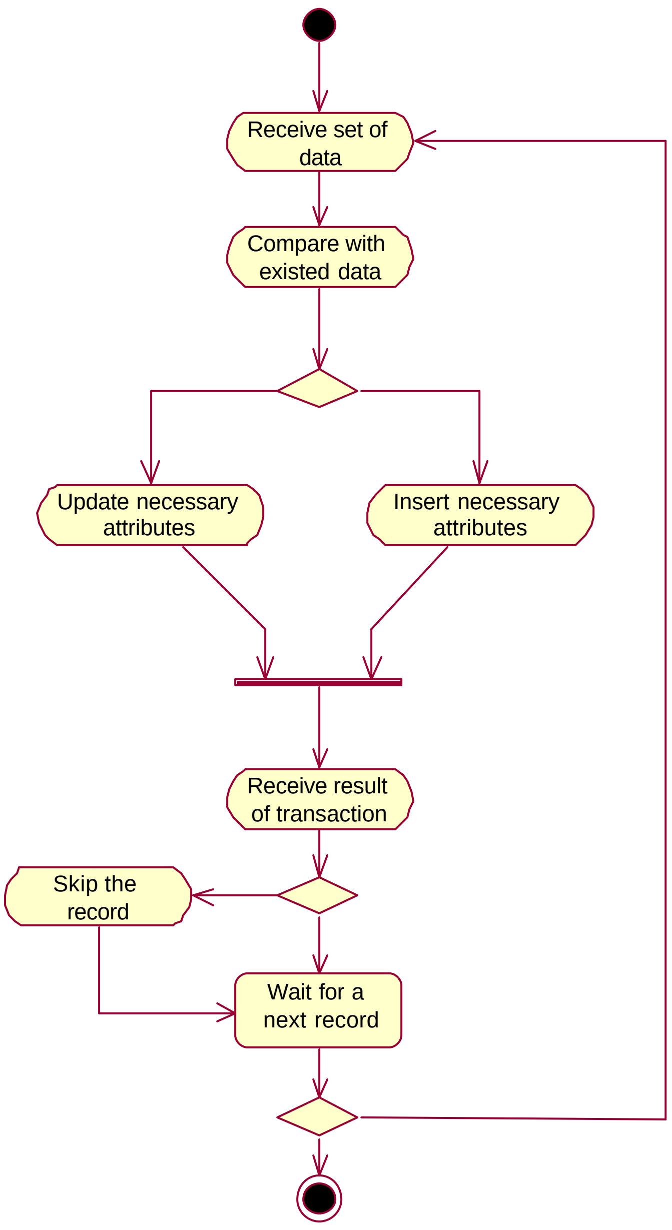

Receive set of data
Compare with existed data
Update necessary attributes
Insert necessary attributes
Receive result of transaction
Skip the record
Wait for a next record

Figure 39: Activity Diagram - Update and insert data

The last function of the PCM is to form and send some statistic data to the Data Generation (DG) subsystem. This information might be used by the total system to determine the effectiveness of the each server. It might be possible after the analysis of the stored statistics data. As was mentioned in chapter 2 the request to the PCM contains such elements as *user_id* and *query_id*. Element *user_id* might be undefined, because not all the users are registered and they have an ability to make searches just for their own interest. These users are stored by name *anonymous.* Element *query_id* must be unique for the each search, because all the received for the response information has to be bind with only one query, which contain the information about the *term* of the query and the *targets*.

PCM collect all the information related to the unique *query_id* (elements 3-8 in table 6) and send all this information to the DG subsystem.

The activity diagram of these processes is presented on the next figure.

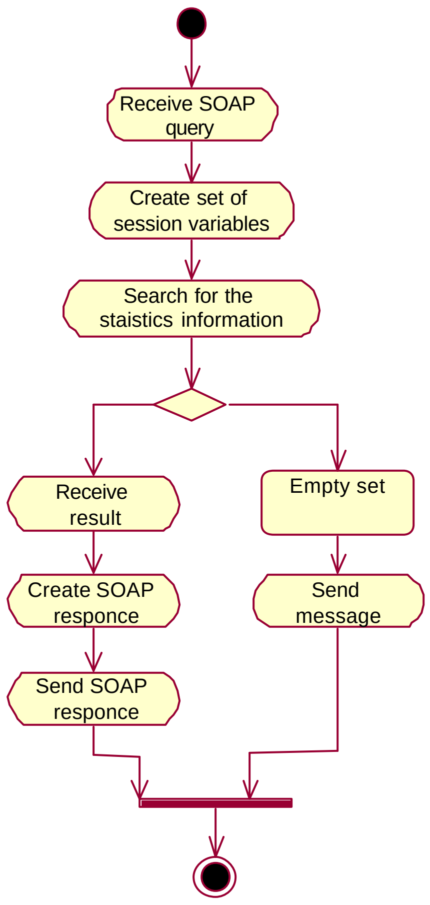


Figure 40: Activity Diagram – Collecting and sending statistics data

4.4.2 Procurement

4.4.2.1 Introduction

This is the continuation of the Procurement business model developed in the WP1.

In the WP1, to understand the structure, the dynamics and the boundaries of the Procurement Market Place System, we formalized and defined the processes (use cases), roles, and responsibilities (actors).

In the WP2, on the basis of the business model (input), we're going to develop the Procurement object oriented design model ; we have to identify the classes and the collaborations between them. We have to ensure that the class provides the behaviour the use-case realizations require.

4.4.2.2 Design model hierarchy

The model is composed of 4 packages :

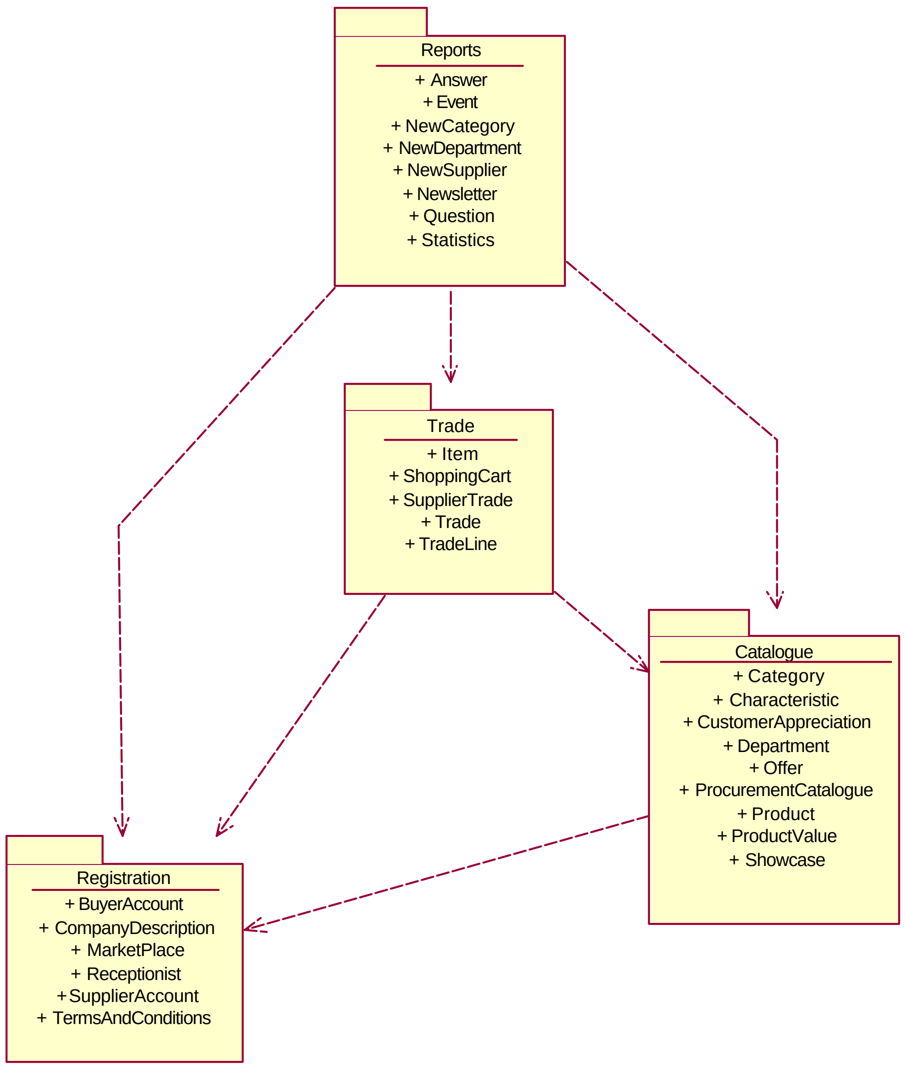


Figure 41: Package Diagram - Procurement

NAME	DESCRIPTION	CLASSES	
REGISTRATION	Object business concepts of Presentation and Registration to the Market Place of Procurement processes. 1st package to be developed.	MARKET PLACE	Virtual Place where identified suppliers can trade with identified buyers.
		TERMS AND CONDITIONS	The rules of Market Place functioning that every supplier or buyer commits himself to respect.
		BUYER ACCOUNT	Expert object of the operations that a buyer can do in the market place.
		SUPPLIER ACCOUNT	Expert object of the operations that a supplier can do in the market place.
		RECEPTIONIST	Person who will receive all the requests for trade with the supplier.
		COMPANY DESCRIPTION	Presentation of his company by the supplier.

NAME	DESCRIPTION	CLASSES	
		A USER	Actor, authorized person to use the market Place.
		AN ADMINISTRATOR	Actor, he manages all users of the market place, pre-qualifies all the suppliers, manages the structure of the catalogue.
		A SUPPLIER	Actor, he is the responsible of the Supplier's Account in the market place, he manages all information about and his showcase.
		A BUYER	Actor, everyone authorized to trade with the suppliers of the market place.
CATALOGUE	Object business concepts of management processes of the Procurement Catalogue. 2nd package to be developed.	PROCUREMENT CATALOGUE	Structured catalogue of all products traded in the market place. It is managed by the administrator of the market place.
		DEPARTMENT	Division of catalogue (ex book, arts materials, multimedia, reproduction, services, …)
		CATEGORY	Division of department (ex for book department, categories could be arts books, history books, children books,…)
		CHARACTERISTIC	Each category is defined by a set of characteristics. for example : the category poster must have the characteristic "dimension" the category artist book must have the characteristics "author"
		SHOWCASE	Supplier's catalogue on market Place. Each supplier manages his showcase.
		PRODUCT	Each good or service referenced on the market place. The supplier registers products in his showcase, simultaneously each registered product is ranged in the procurement catalogue and its characteristics are increased in value.
		PRODUCT VALUE	For a product, Value of a characteristic of the product's category. Each characteristic of the product's category has to be increased in value.

NAME	DESCRIPTION	CLASSES	
		OFFER	For each product of his showcase, the supplier can have one or many offers, an offer is a price for a quantity or an interval of quantity.
		CUSTOMER APPRECIATION	A buyer can give an appreciation about a product.
TRADE	Object business concepts of trade between buyer and supplier processes. 3rd package to be implemented.	SHOPPING CART	Virtual cart where the buyer places the items he chose.
		ITEM	Choice of the buyer : product and offer (quantity)
		TRADE	Pre-order; the buyer informs the supplier(s) that he is interested with some products for an announced price, but the complete term of the order remains to be arranged : shipping, timeframe, payment terms.
		SUPPLIER TRADE	Subdivision by supplier of a trade.
		TRADE LINE	The elementary part of a trade; product's level.
REPORTS	Object business concepts of information, measurements of the Market Place processes. 4th package to be implemented	QUESTION	A User can question about procurement functioning
		ANSWER	The administrator has to answer of each question
		NEWSLETTER	Periodically, the administrator selects last events to be presented to the Market Place users.
		EVENTS	Important fact, which concerns the market place.
		NEW SUPPLIER	Sort of event
		NEW DEPARTMENT	Sort of event
		NEW CATEGORY	Sort of event
		STATISTICS	Measurements of market place.<

Table 7: Procurement

4.4.2.3 Diagrams of the design model : REGISTRATION

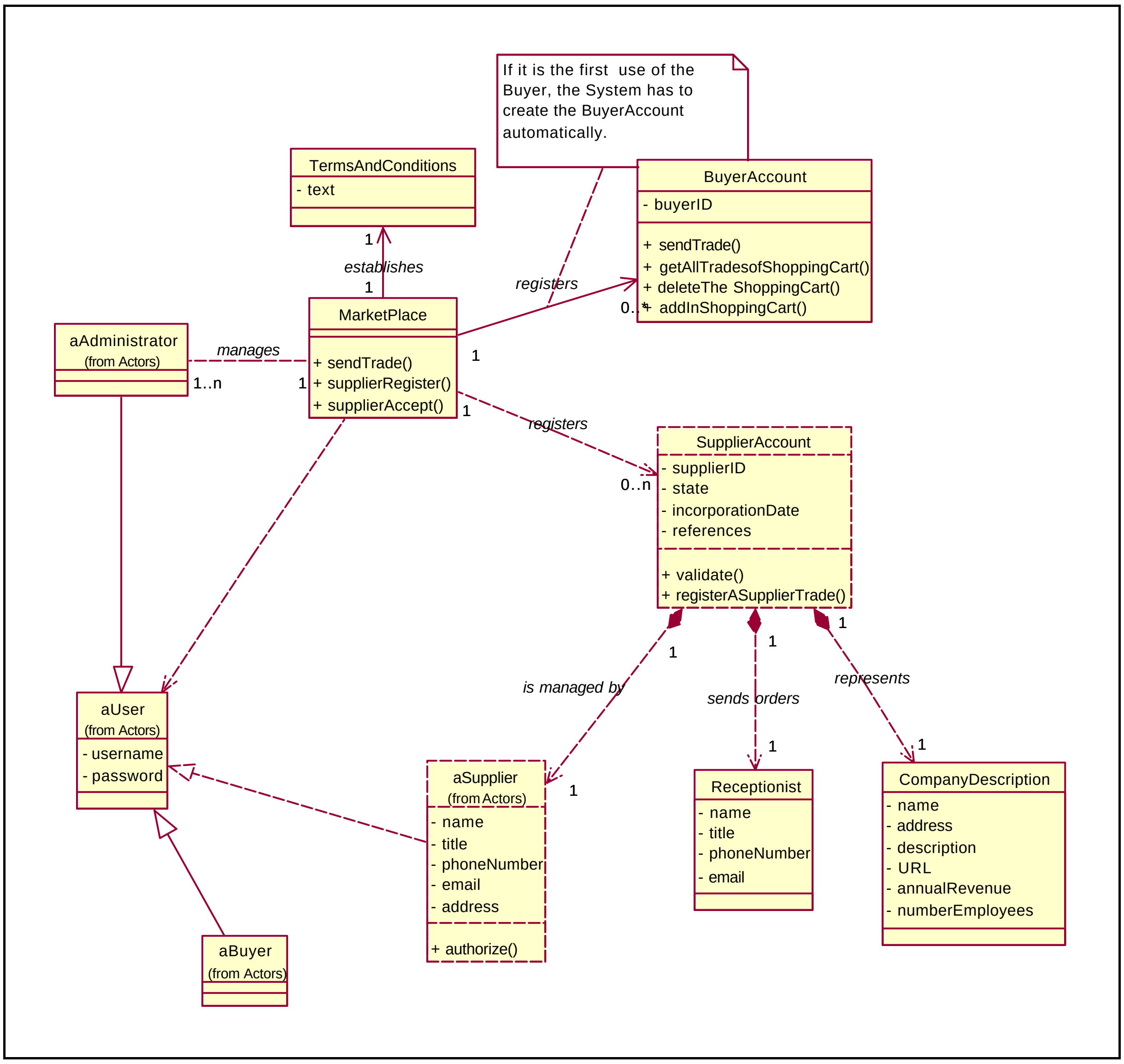


Figure 42: Class Diagram - Registration

☞ The Procurement Market Place is managed by the CSC administrator.

☞ Every REGNET member can be a buyer on the procurement market place. The first time he trades, his account is created (BuyerAccount) if he accepts terms and conditions of the market place.

☞ A supplier of the procurement has to be "validated" by the administrator before to be authorized.
For that, he has to complete a registration form that the system registers as a SupplierAccount which is composed by the Supplier (it is the responsible), the receptionist and a company description. When it is created the SupplierAccount is in state "to be validated" and the administrator is informed of the demand.
Of course, the supplier has to accept terms and conditions of the market place.
The administrator verifies the references of the supplier, and can accept (the SupplierAccount is in state "validated") or refuses (the SupplierAccount is in state "refused") the supplier's application.

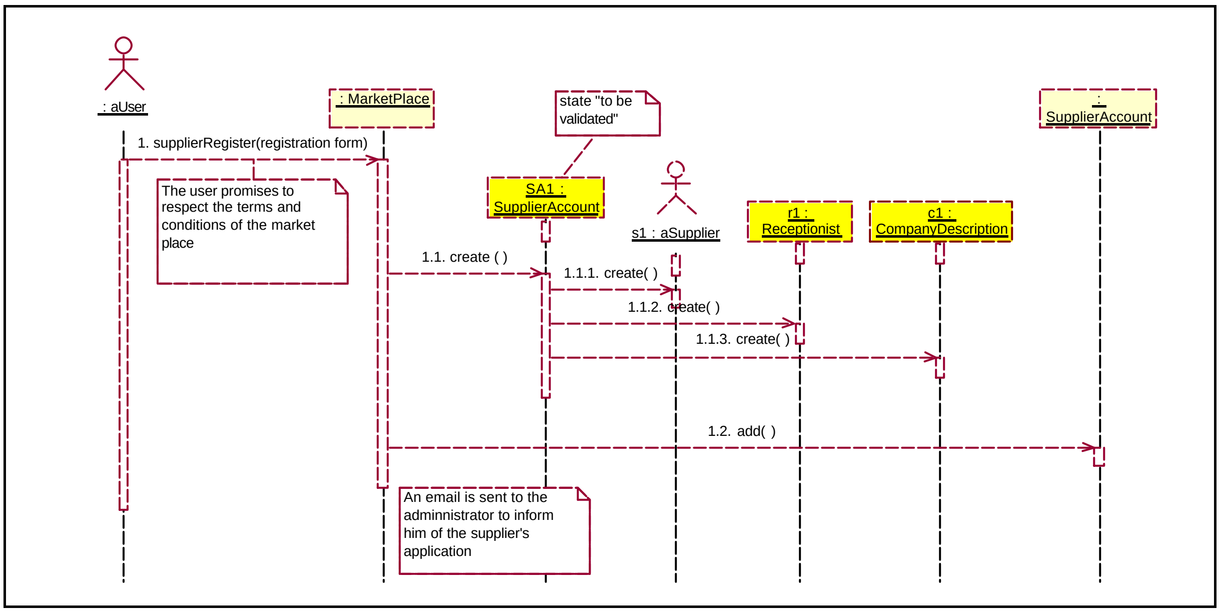


Figure 43: Sequence Diagram - Supplier's registration

A user complete **a registration form** with all necessary information :

→ The company description : name, address, description, URL, annual revenue and number of employees

→ The references: the future supplier must provide trade references (minimum 2); what's company, what is the trade's relationship.

→ The responsible of the company for the market place, this person will be the only one authorised to manage the supplier's showcase and information.

→ The receptionist of the company; name, number and email of the person who will receive the orders.

The market place creates the supplier's account with a state "to be validated"; it is waiting for the acceptance or the rejection by the market place administrator.

It creates too the bound instances of the supplier-responsible, the receptionist and the company's description.

It adds the supplier's account to the list of all suppliers' accounts registered in the market place.

An email is sent to the administrator of the market's place to inform him of the new application.

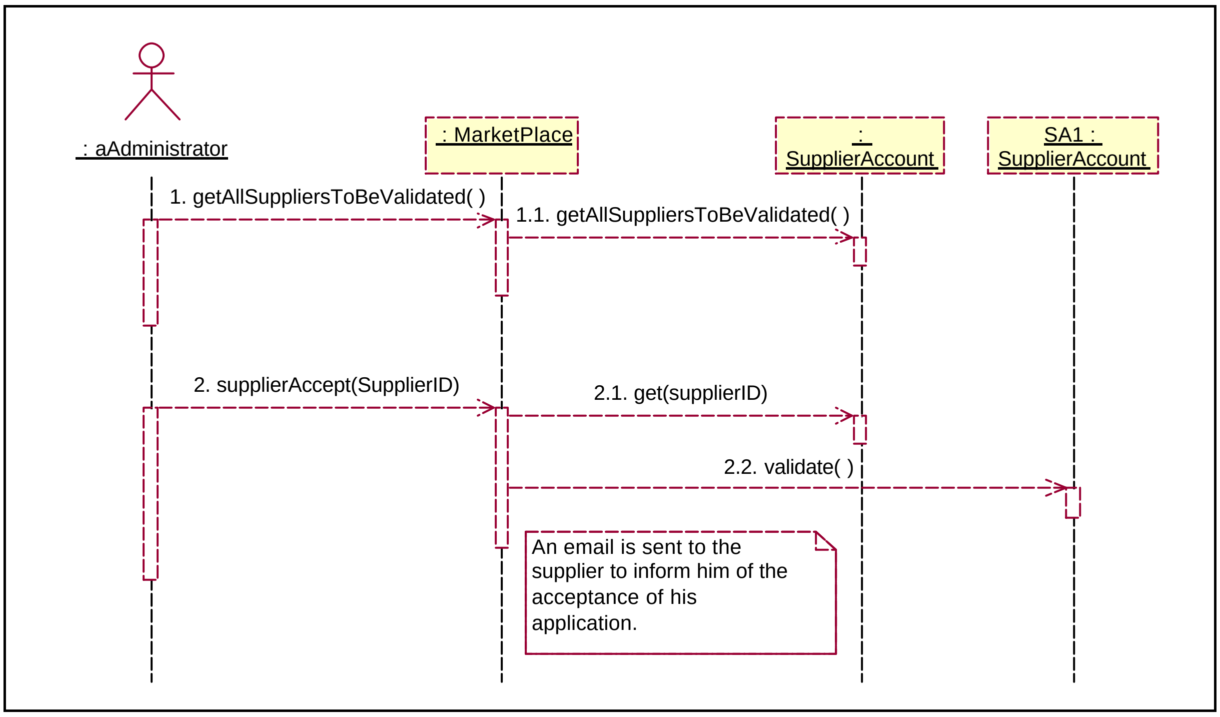

: aAdministrator
: MarketPlace
: SupplierAccount
SA1 : SupplierAccount
1. getAllSuppliersToBeValidated( )
1.1. getAllSuppliersToBeValidated( )
2. supplierAccept(SupplierID)
2.1. get(supplierID)
2.2. validate( )
An email is sent to the supplier to inform him of the acceptance of his application.

Figure 44: Sequence Diagram - Supplier's acceptation

The administrator can visualise all the suppliers, which are waiting for registration's acceptance.

For each new supplier, the administrator has to verify the references provided.

If they suit him, he accepts the registration : the supplier's account becomes "validated".

An email is sent to the supplier (responsible) to inform of the acceptance.

4.4.2.4 Diagrams of the design model : CATALOGUE

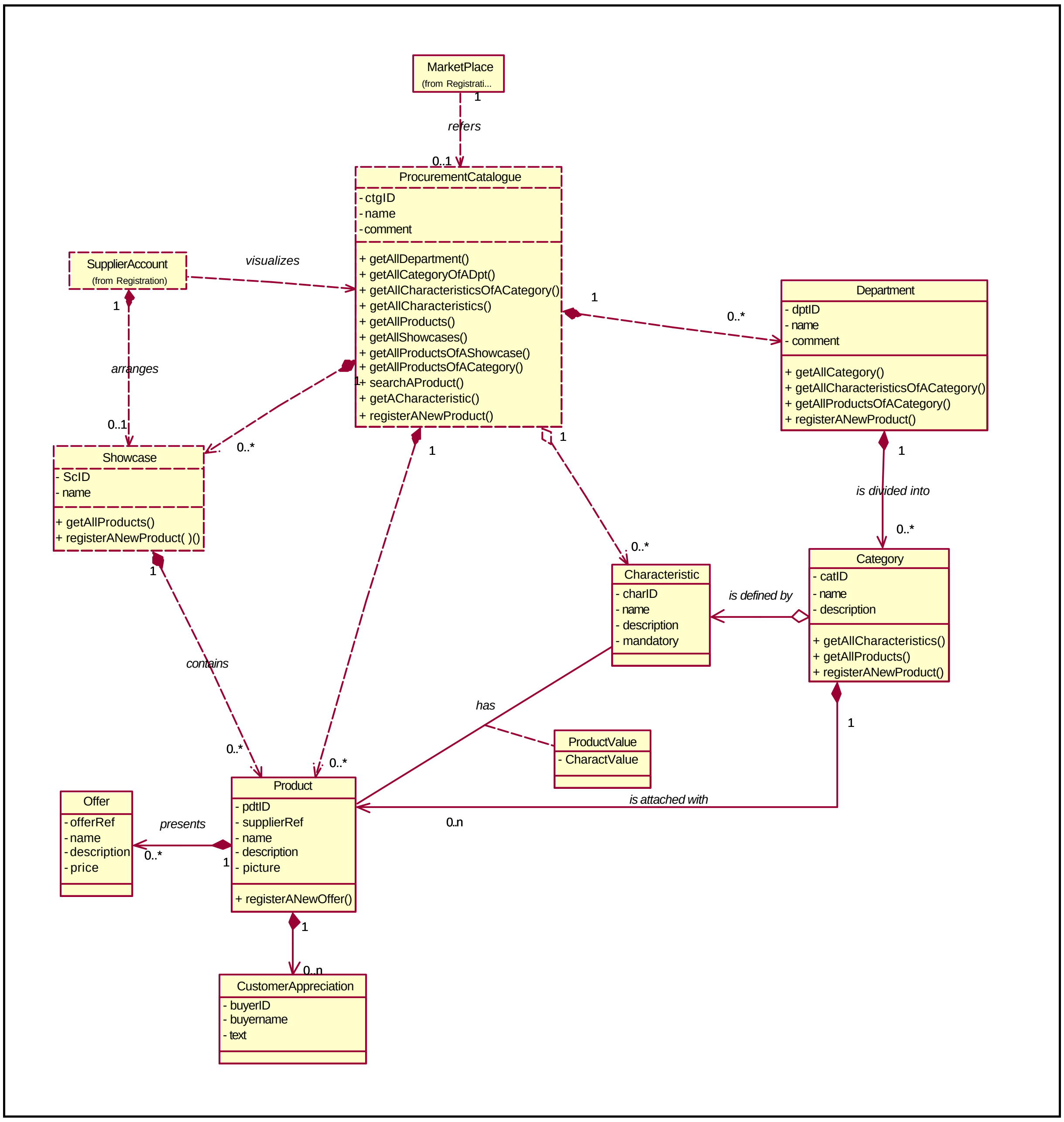


Figure 45: Class Diagram - Catalogue

- The Structuration of the Procurement Catalogue is managed by the administrator; the catalogue is structured in departments, each department is divided in Categories. The administrator creates the departments and the categories.
- The administrator defines each category by a set of characteristics. So when a supplier creates a product, he increases the value of each characteristic of the category the product belongs to.
- In parallel, the procurement catalogue is composed by the suppliers showcases. Each supplier manages his showcase, which corresponds to his own catalogue. The showcase allows the supplier to create, present his products and offers.

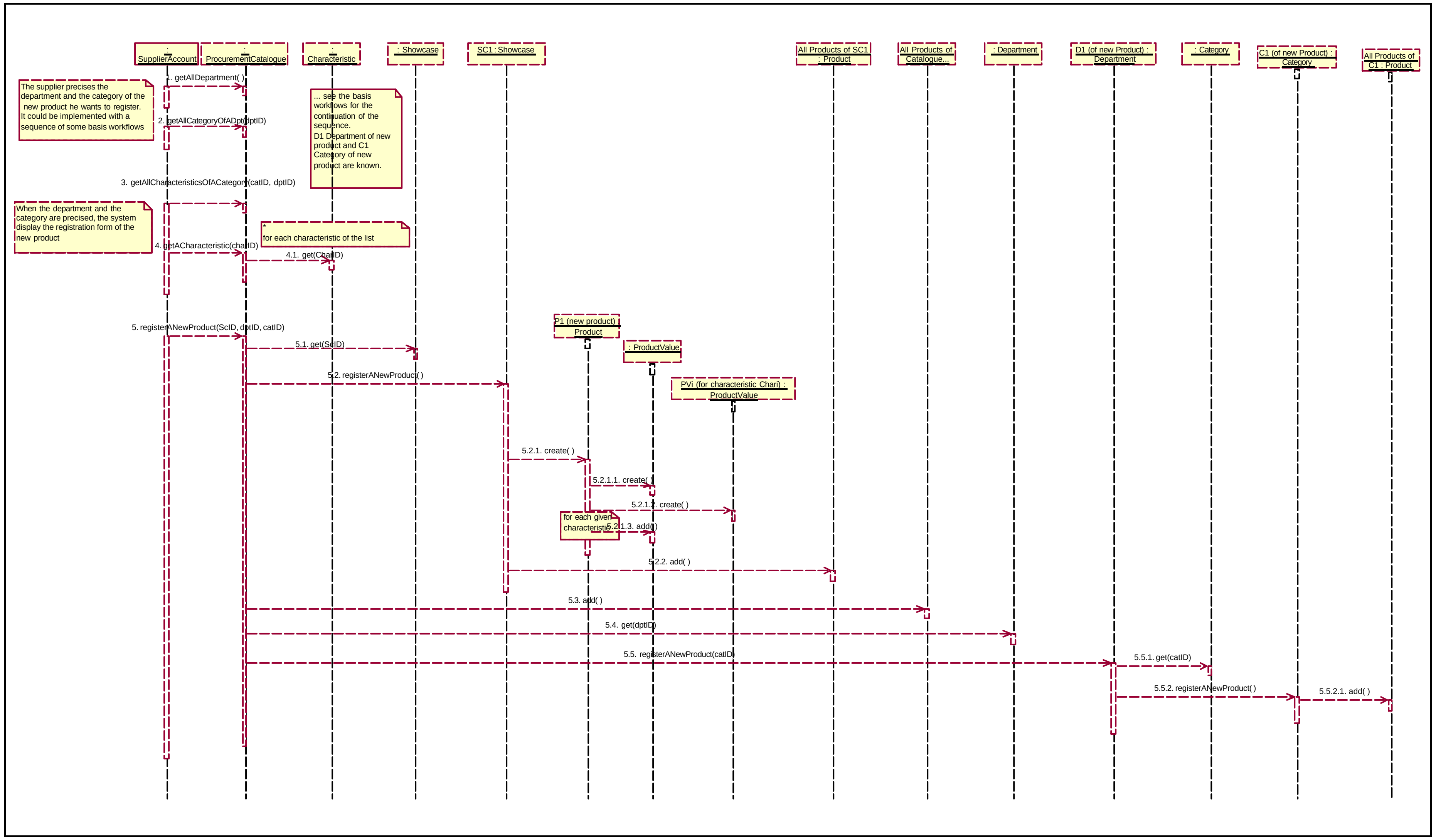

: SupplierAccount
: ProcurementCatalogue
: Characteristic
: Showcase
SC1 : Showcase
All Products of SC1 : Product
All Products of Catalogue...
: Department
D1 (of new Product) : Department
: Category
C1 (of new Product) : Category
All Products of C1 : Product
The supplier precises the department and the category of the new product he wants to register. It could be implemented with a sequence of some basis workflows
1. getAllDepartment( )
2. getAllCategoryOfADpt(dptID)
... see the basis workflows for the continuation of the sequence. D1 Department of new product and C1 Category of new product are known.
3. getAllCharacteristicsOfACategory(catID, dptID)
When the department and the category are precised, the system display the registration form of the new product
* for each characteristic of the list
4. getACharacteristic(charID)
4.1. get(CharID)
P1 (new product) : Product
: ProductValue
PVi (for characteristic Chari) : ProductValue
5. registerANewProduct(ScID, dptID, catID)
5.1. get(ScID)
5.2. registerANewProduct( )
5.2.1. create( )
5.2.1.1. create( )
5.2.1.2. create( )
for each given characteristic
5.2.1.3. add( )
5.2.2. add( )
5.3. add( )
5.4. get(dptID)
5.5. registerANewProduct(catID)
5.5.1. get(catID)
5.5.2. registerANewProduct( )
5.5.2.1. add( )

Figure 46 - sequence diagram - new supplier's product

The supplier (responsible) wants to register a new product in his showcase.

He is in his account and asks for a registration of a new product.

1st step

The procurement catalogue provides all the departments.
The supplier selects among the list, the **department to which the new product belongs**.
Automatically, the procurement catalogue provides all the categories of the selected department.
The supplier selects among the list, the **category to which the new product belongs**.
Automatically, the procurement catalogue provides a **product's registration with all the characteristics** to complete which are the defined characteristics of the category of the department.

The supplier completes the new product's registration.

2nd step

The procurement catalogue creates the **new product** in the **showcase** of the supplier (it creates the showcase and the list of products of showcase if it's the first product).
For the new created product, first the **list of product's values** is created, then for each characteristic of the product, an **instance product value** is created.
The new product is added to the list of products of the supplier's showcase.

3rd step

The procurement catalogue adds the new product to the list of all products of the market place.
The category adds the new product to the list of all products of the category of the department.

4th step (not detailed in sequence diagram)

For each product, the supplier has to complete at least one offer, which presents a price for a quantity or a segment of quantities. The supplier can have some offers for the same product.

Search Workflows

These workflows specify the different ways to search a product in the market place.

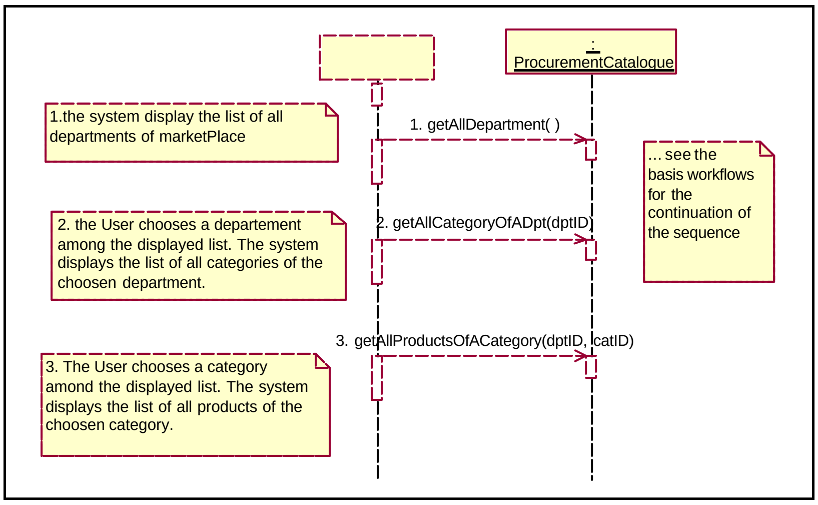

: ProcurementCatalogue
1.the system display the list of all departments of marketPlace
1. getAllDepartment( )
... see the basis workflows for the continuation of the sequence
2. the User chooses a departement among the displayed list. The system displays the list of all categories of the choosen department.
2. getAllCategoryOfADpt(dptID)
3. getAllProductsOfACategory(dptID, catID)
3. The User chooses a category amond the displayed list. The system displays the list of all products of the choosen category.

Figure 47: Sequence Diagram – Navigation by department and category

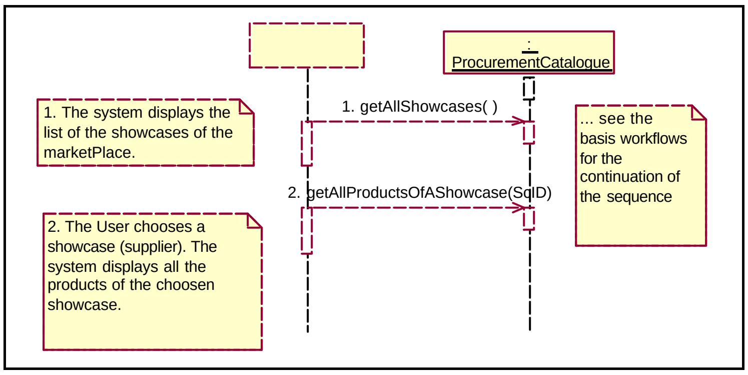

:
ProcurementCatalogue
1. The system displays the list of the showcases of the marketPlace.
1. getAllShowcases( )
... see the basis workflows for the continuation of the sequence
2. getAllProductsOfAShowcase(SqID)
2. The User chooses a showcase (supplier). The system displays all the products of the choosen showcase.

Figure 48: Sequence Diagram – Navigation by supplier

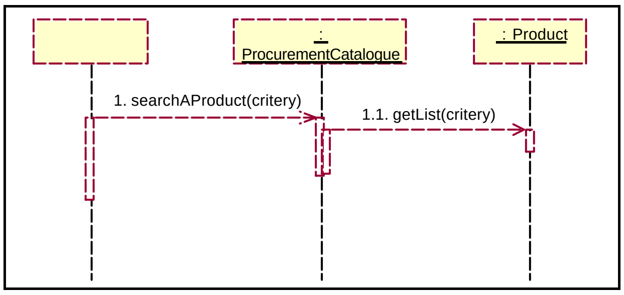


Figure 49: Sequence Diagram - Search a product

Basis Workflows

The basis workflows are workflows, which are, used in others complex workflows.

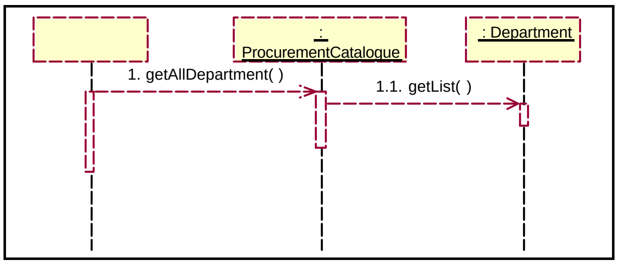


Figure 50: Sequence Diagram - List of departments

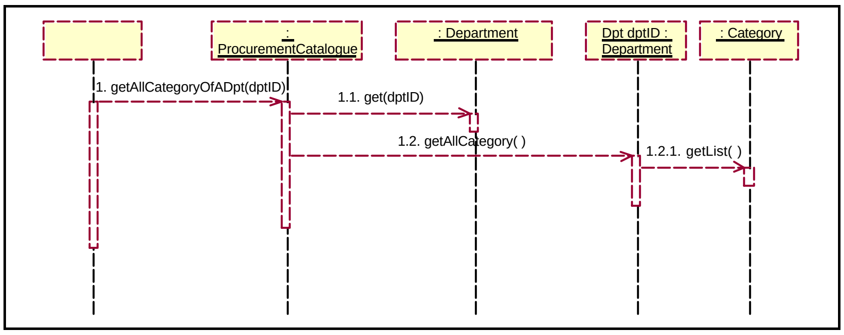


Figure 51: Sequence Diagram - List of categories of a department

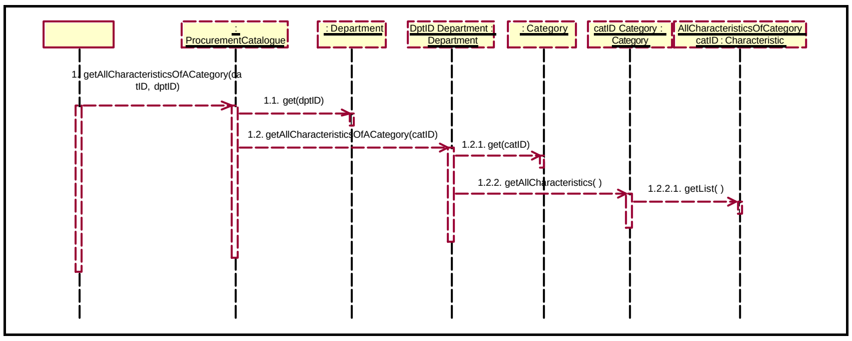

: ProcurementCatalogue
: Department
DptID Department : Department
: Category
catID Category : Category
AllCharacteristicsOfCategory catID : Characteristic
1. getAllCharacteristicsOfACategory(datID, dptID)
1.1. get(dptID)
1.2. getAllCharacteristicsOfACategory(catID)
1.2.1. get(catID)
1.2.2. getAllCharacteristics( )
1.2.2.1. getList( )

Figure 52: Sequence Diagram - List of characteristics of a category

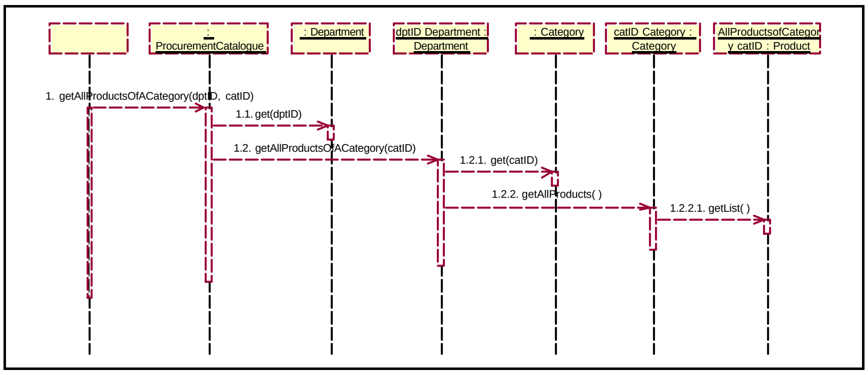

: ProcurementCatalogue
: Department
dptID Department : Department
: Category
catID Category : Category
AllProductsofCategory catID : Product
1. getAllProductsOfACategory(dptID, catID)
1.1. get(dptID)
1.2. getAllProductsOfACategory(catID)
1.2.1. get(catID)
1.2.2. getAllProducts( )
1.2.2.1. getList( )

Figure 53: Sequence Diagram -List of products of a category

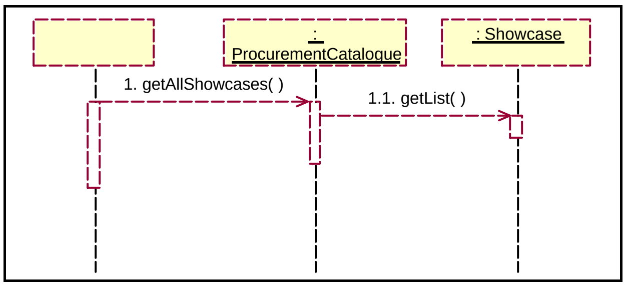


Figure 54: Sequence Diagram - List of showcases

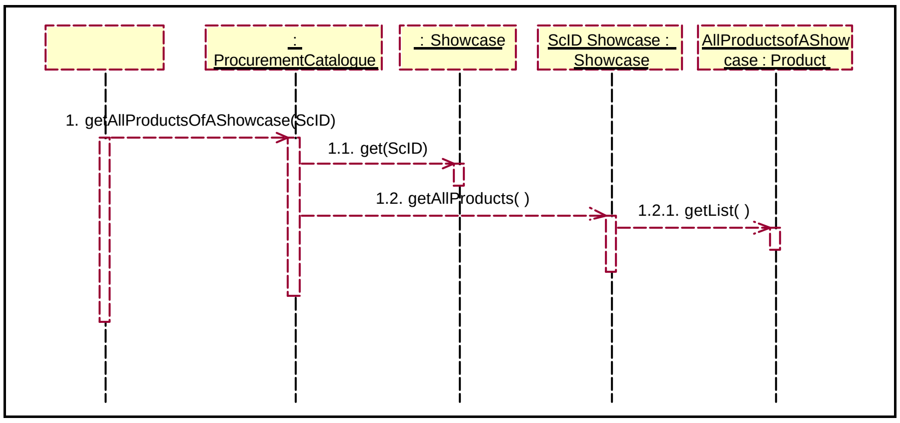


Figure 55: Sequence Diagram - List of products of a showcase

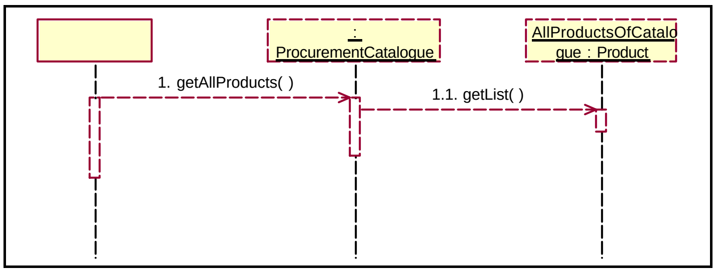


Figure 56: Sequence Diagram - List of products

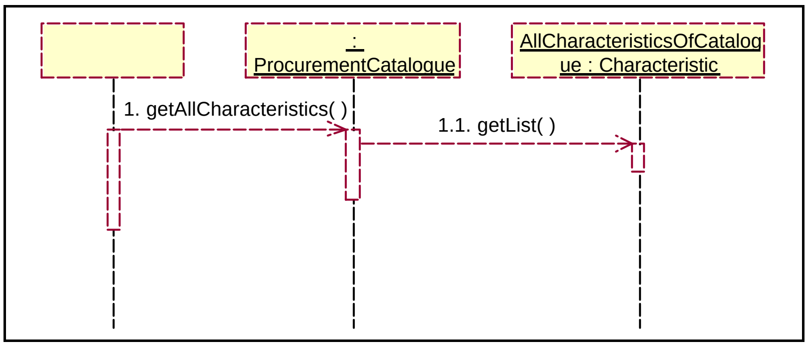

: ProcurementCatalogue
AllCharacteristicsOfCatalogue : Characteristic
1. getAllCharacteristics( )
1.1. getList( )

Figure 57: Sequence Diagram - List of characteristics

4.4.2.5 Diagrams of the design model : TRADE

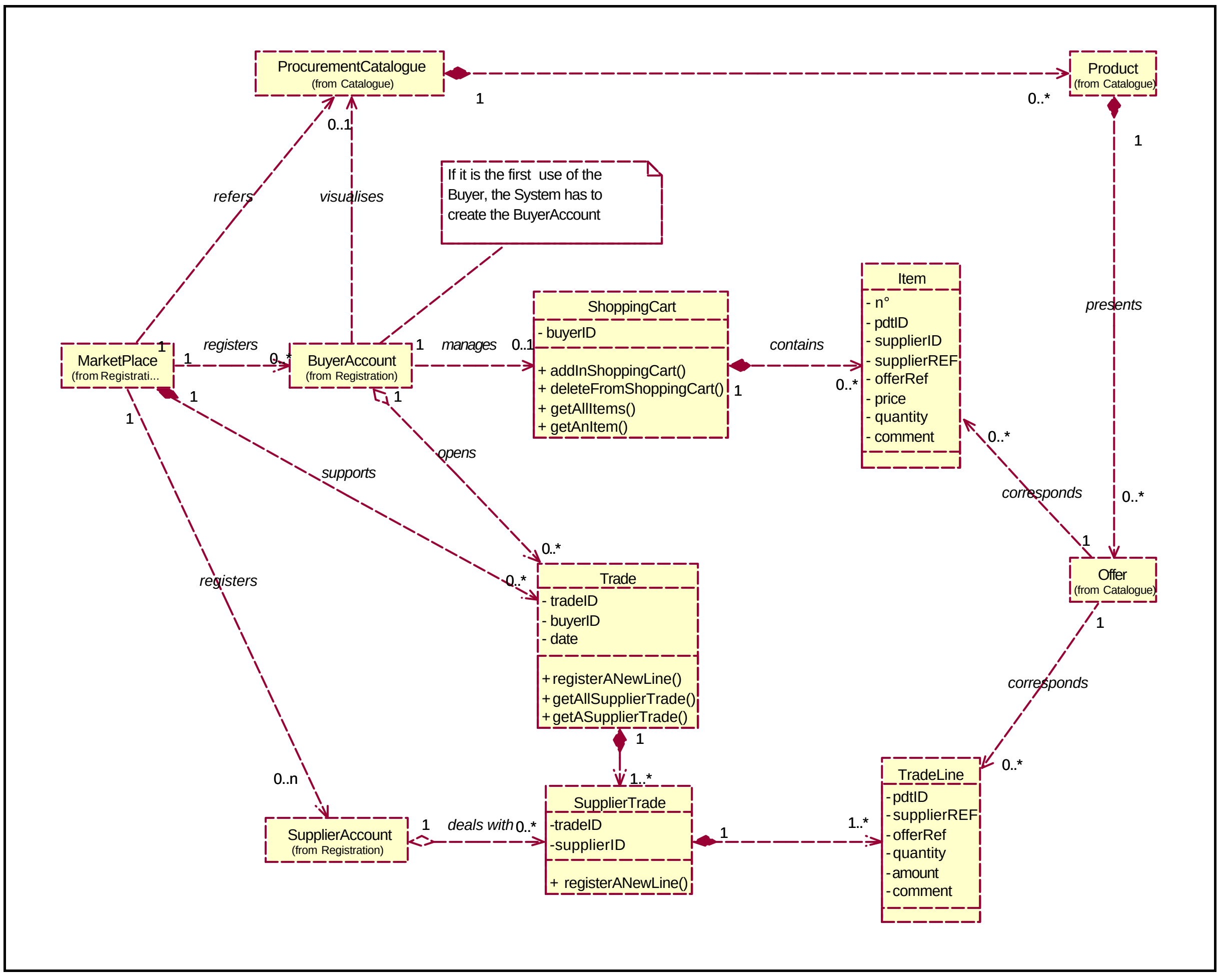


Figure 58: Class Diagram - Trade

☞ The Buyer searches the products of market place he wants to buy; he can navigate by department, category, by supplier, or search product by criteria.
When an offer suits him, he places it in his shopping cart.
At any time, he can visualize the items of his shopping cart, he can remove any item.
The selected offers can be from different suppliers.

☞ When a buyer finished his shopping, he sends his trade.
The content of his shopping cart is translated in trade-line : one trade-line for one item. The trade-lines are grouped by supplier. Each supplier's trade is sent to the receptionist of the supplier by mail.
The shopping cart and its content are deleted.
The trade is registered; the buyer and the concerned suppliers could consult the whole or the part concerning them.

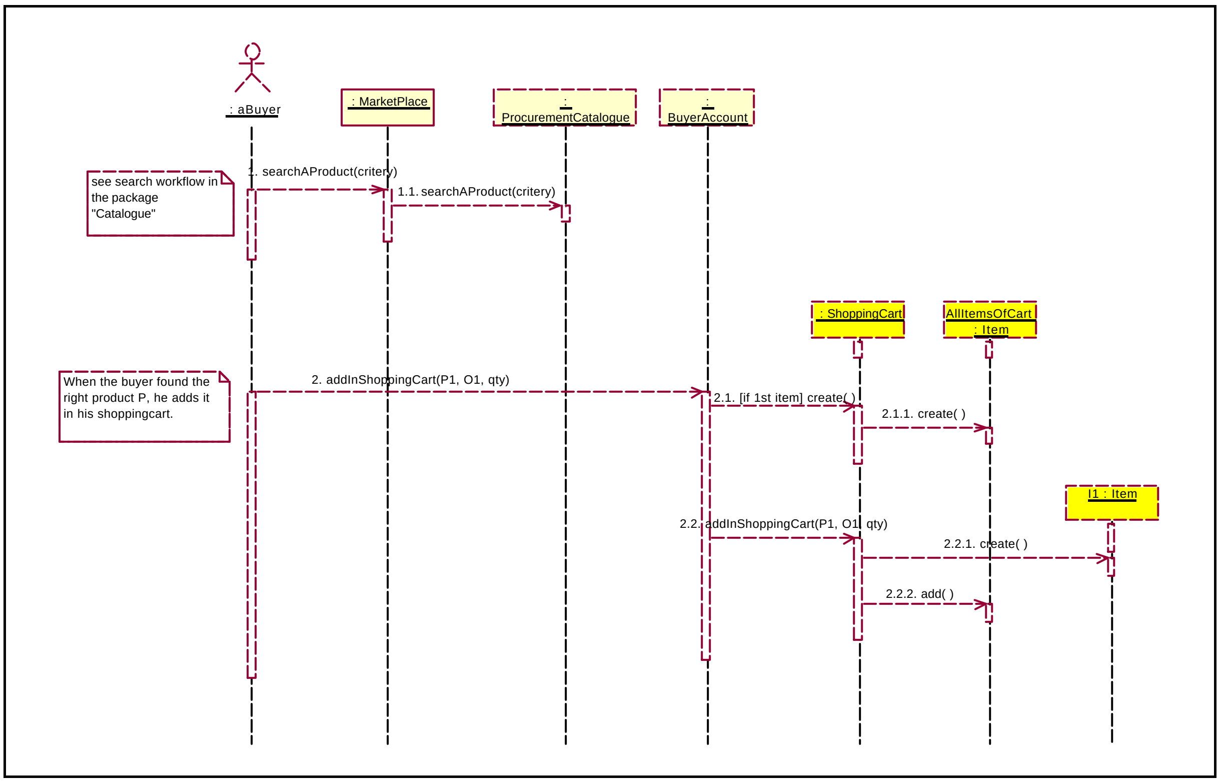

: aBuyer
: MarketPlace
: ProcurementCatalogue
: BuyerAccount
see search workflow in the package "Catalogue"
1. searchAProduct(critery)
1.1. searchAProduct(critery)
: ShoppingCart
AllItemsOfCart : Item
When the buyer found the right product P, he adds it in his shoppingcart.
2. addInShoppingCart(P1, O1, qty)
2.1. [if 1st item] create( )
2.1.1. create( )
I1 : Item
2.2. addInShoppingCart(P1, O1, qty)
2.2.1. create( )
2.2.2. add( )

Figure 59: Sequence Diagram - place an item in a cart

The buyer searches for a product in the market place as detailed in the search workflows.

When he displays a product, he can select it for his shopping cart.

If it is the first selected product, the shopping cart is created and the list of items in cart too. Then, the corresponding item is created, and added to the list of items.

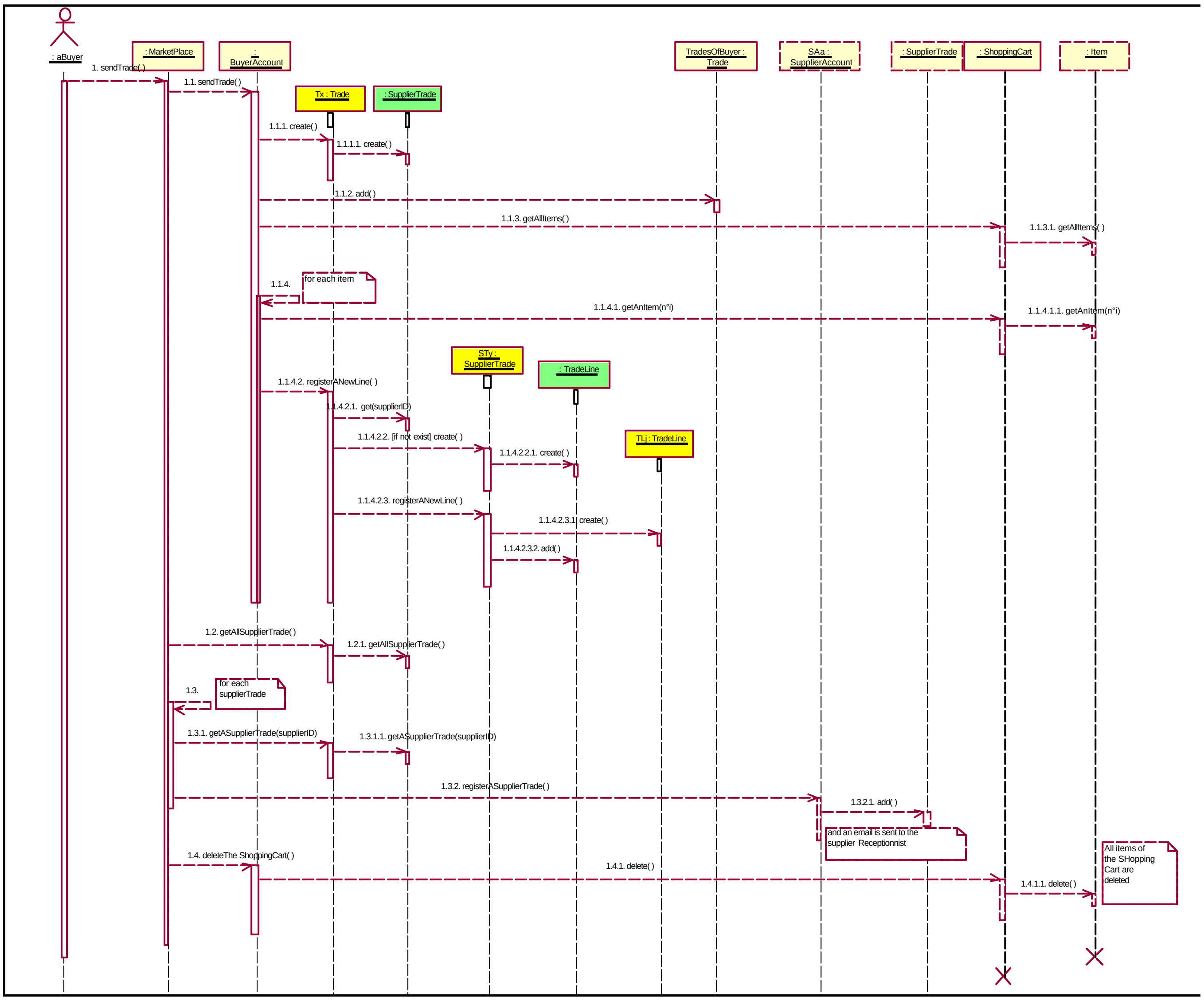


Figure 60: Sequence Diagram - send the trade

When he has finished his shopping, the buyer validates his shopping cart.

1st step

A **trade** is created with a **list of supplier's (sub-)trade**.

The trade is added to the list of the trades of the buyer's account.

2nd step

For each item in the shopping cart,

The trade searches the corresponding supplier's trade (supplierID) among its list. If it doesn't exist, a **supplier's trade** is created with a **list of trade's lines**.

A trade line is created, corresponding to the considered item; a number is automatically affected to the line (increment +1), the information of item are saved (productID, supplierID, supplierRef, offerRef, price and quantity). It is added to the list of the trade's lines of the concerned supplier's trade.

3rd step

Each created supplier's (sub-)trade of the trade is added to the list of supplier's trade of the supplier's account.

And an email is sent to the supplier's receptionist with the buyer's references and all trade's line of the supplier's trade.

4th step

The shopping cart and its list of items are deleted.

4.4.2.6 Diagrams of the design model : REPORTS

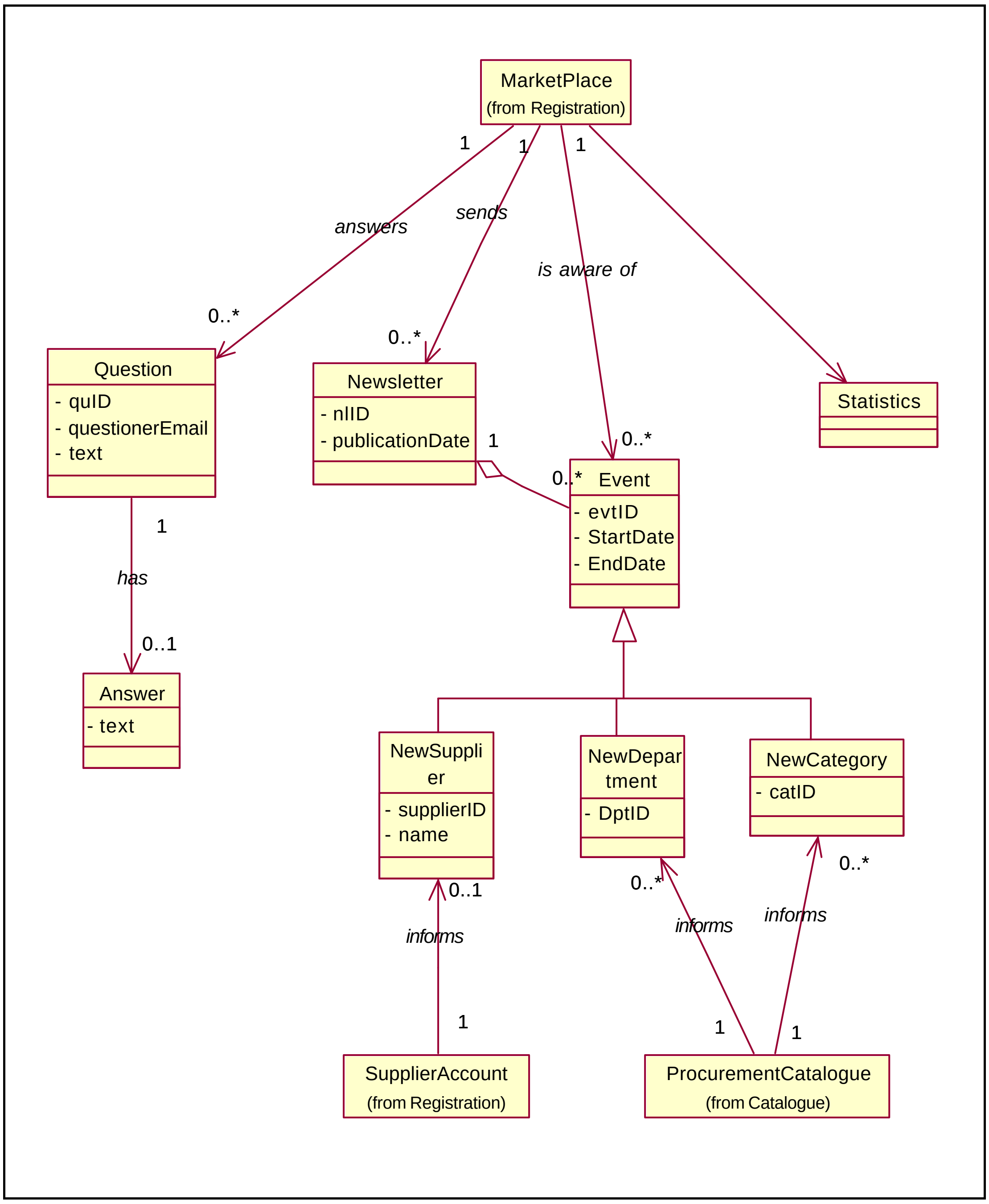


Figure 61: Class Diagram - Reports

- a User can have questions about the market place functioning. The administrator answers all questions.
- Periodically, the administrator presents last events of market place in a newsletter, which is sent to all users; buyers and suppliers of market place.

☞ Periodically, statistics could be calculated to measure the market place.

4.4.3 Delivery

4.4.3.1 Introduction

This is the continuation of the delivery and invocation services business model developed in the WP1.

In the WP1, to understand the structure, the dynamics and the boundaries of the Delivery and Invocation System, we formalized and defined the processes (use cases), roles, and responsibilities (actors).

In the WP2, on the basis of the business model (input), we're going to develop the Delivery and Invocation object oriented design model; we have to identify the classes and the collaborations between them. We have to ensure that the class provides the behaviour the use-case realizations require.

This system depends on the e-Shop system (catalogue and order) developed by our partner ZEUS.

The e-Shop System has to provide all data and information necessary for functioning of Delivery and Invocation System.

HYPOTHESIS:

we can have one e-shop for one warehouse OR several e-shops for one warehouse (never one e-shop for several warehouses)

4.4.3.2 Design model hierarchy

The model is composed of 2 packages, which depend on Packages Catalogue and Order of the e-shop.

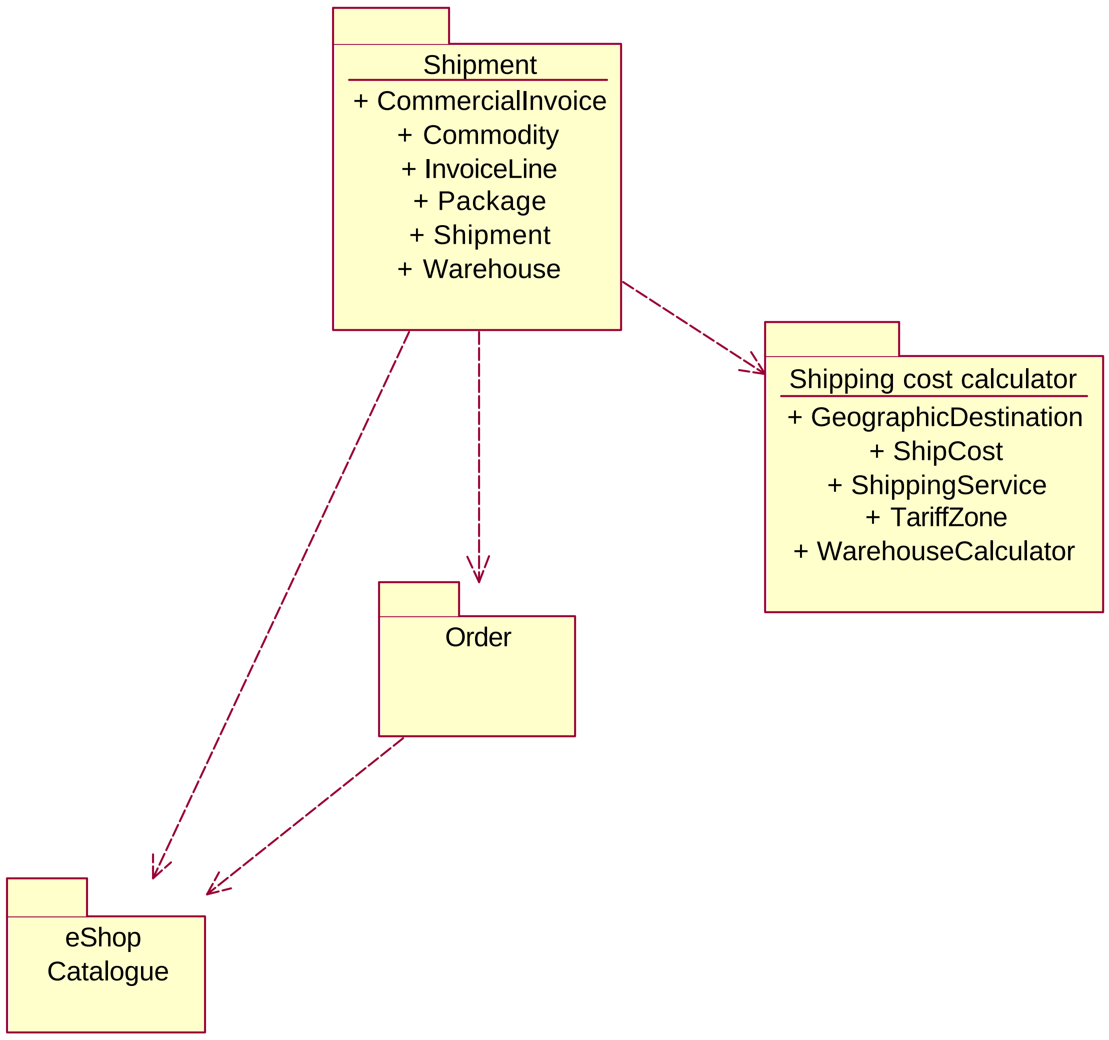


Figure 62: Package Diagram - Delivery

NAME	DESCRIPTION	CLASSES	
SHIPPING COST CALCULATOR	Object business concepts of Loading of data of transport rates and calculation. 1st package to be developed.	WAREHOUSE CALCULATOR	Calculator of the shipping cost. It is attached to the warehouse (defines the origin of all the deliveries).
		SHIPPING SERVICE	Company, which takes care of the transport of the ordered commodities (DHL UPS Fedex Chronopost ...). We can propose a choice among several companies, each one is represented by a shipping service.

NAME	DESCRIPTION	CLASSES	
		GEOGRAPHIC DESTINATION	It is the littlest geographic destination zone defined and registered. It could be a country or a city in a country. A geographic destination could be delivered by several services type or none.
		TARIFF ZONE	For each service level of a transport company, it is defined an arbitrary division of the worldwide destinations. Each referenced geographic destination belongs to one and only one tariff zone according of the shipping service.
		SHIP COST	price of the transport defined according to : the shipping service the tariff zone the weight of the shipment (max limit)
SHIPMENT	Object business concepts of Shipment's preparation. 2nd package to be developed.	WAREHOUSE	The e-Shop transmits to the warehouse each order to be fulfilled. Each detailed and complete order could be received in XML format (defined by ZEUS)
		SHIPPING CLERK	Actor, The shipping clerk prepares the parcels; he collects the items related to each order, he packs them, he weighs the parcel in order to calculate the shipping rate, he prints the invoice and the customs formalities for the export and encloses with the parcel, he labels the parcel, and puts it on the collection quay
		SHIPMENT	Each order is fulfilled by one shipment, which could present one or more packages and a corresponding commercial invoice.
		PACKAGE	Real parcel : each parcel is labelled, the first one "the principal" presents the waybill for the transport company.
		COMMODITY	real ordered good
		COMMERCIAL INVOICE	All information about the order and the shipment. The commercial invoice is printed and enclosed with the principal parcel. For international transport, 5 copies of the commercial invoice have to be provided to the transport company with the packages of the shipment.
		INVOICE LINE	Each line corresponds to an ordered referenced product.

Table 8: Delivery

4.4.3.3 Diagrams of the design model : WAREHOUSE CALCULATOR

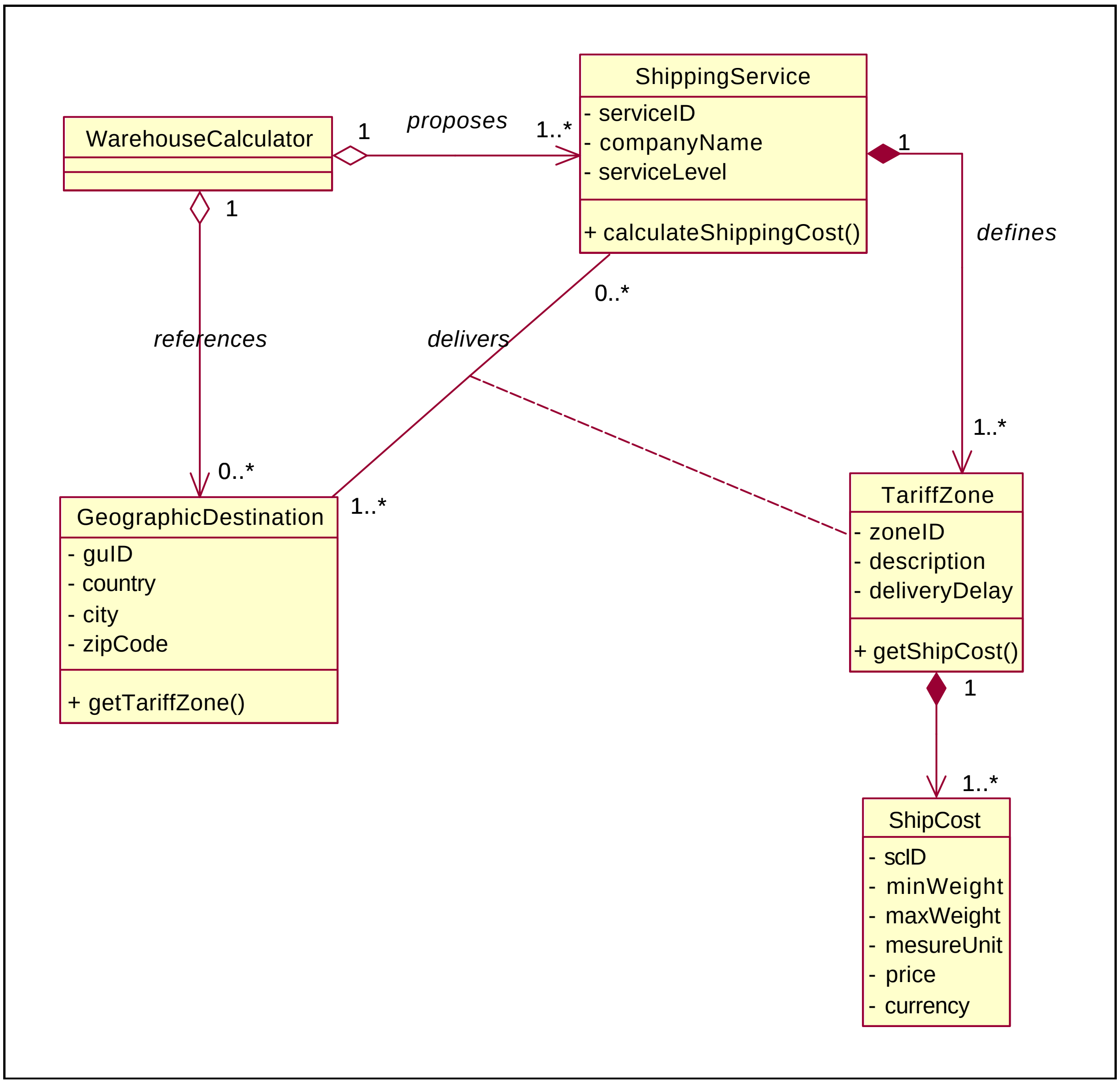


Figure 63: Class Diagram : Shipping cost calculator

- ☞ The Warehouse Calculator could be called (used) by any other object, which needs to calculate a shipping cost; it is a processus of query/answer with input's information to be provided.
- ☞ The Warehouse Calculator references all possible destination: country, city, zip code.
- ☞ The Warehouse Calculator can propose several shipping services if REGNET want to propose the choice among services of several type (standard and express for example) or several transport companies (UPS, DHL for example).
- ☞ Each Shipping service can delivers all destinations or not. Each delivered destination belongs to a tariff zone characteristic of the shipping services. This tariff zone allows getting the ship cost depending on the delivery's weight.

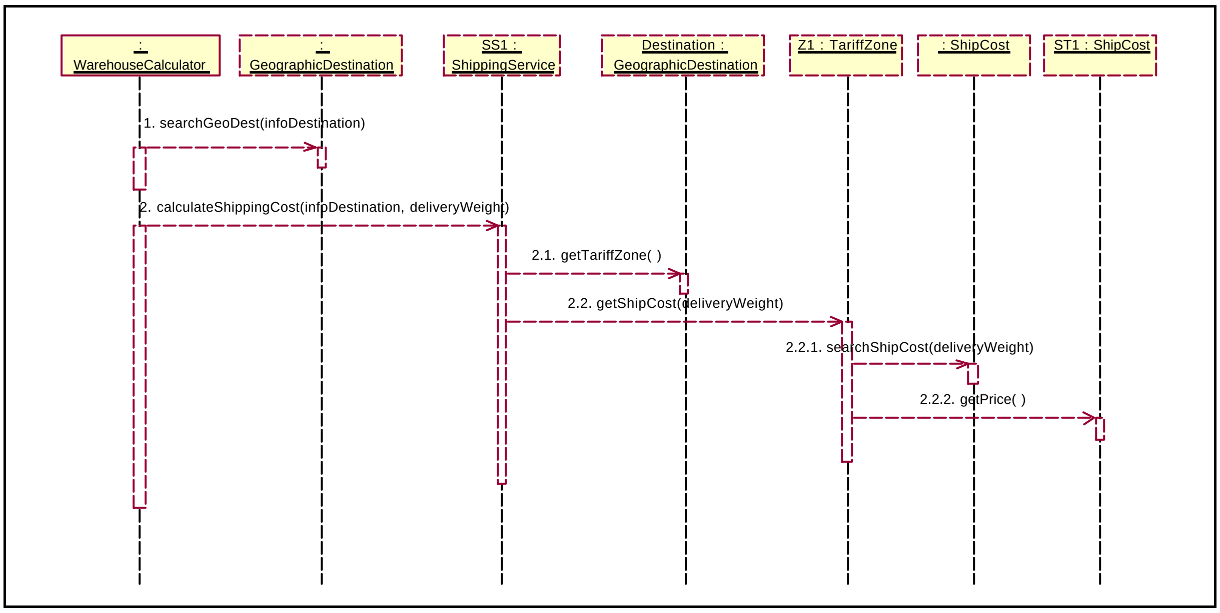

: WarehouseCalculator
: GeographicDestination
SS1 : ShippingService
Destination : GeographicDestination
Z1 : TariffZone
: ShipCost
ST1 : ShipCost
1. searchGeoDest(infoDestination)
2. calculateShippingCost(infoDestination, deliveryWeight)
2.1. getTariffZone( )
2.2. getShipCost(deliveryWeight)
2.2.1. searchShipCost(deliveryWeight)
2.2.2. getPrice( )

Figure 64: Sequence Diagram -Calculation of shipping cost

1st step

The calculator searches the destination among the complete list of referenced geographic destinations.

2nd step

The calculator verifies that the retrieved destination is delivered by the chosen shipping service. If it is right, the geographic destination corresponds to one tariff zone of the shipping service.

3rd step

Finally, the corresponding tariff zone allows giving a delay and the ship cost depending on the weight of the delivery.

4.4.3.4 Diagrams of the design model : SHIPMENT

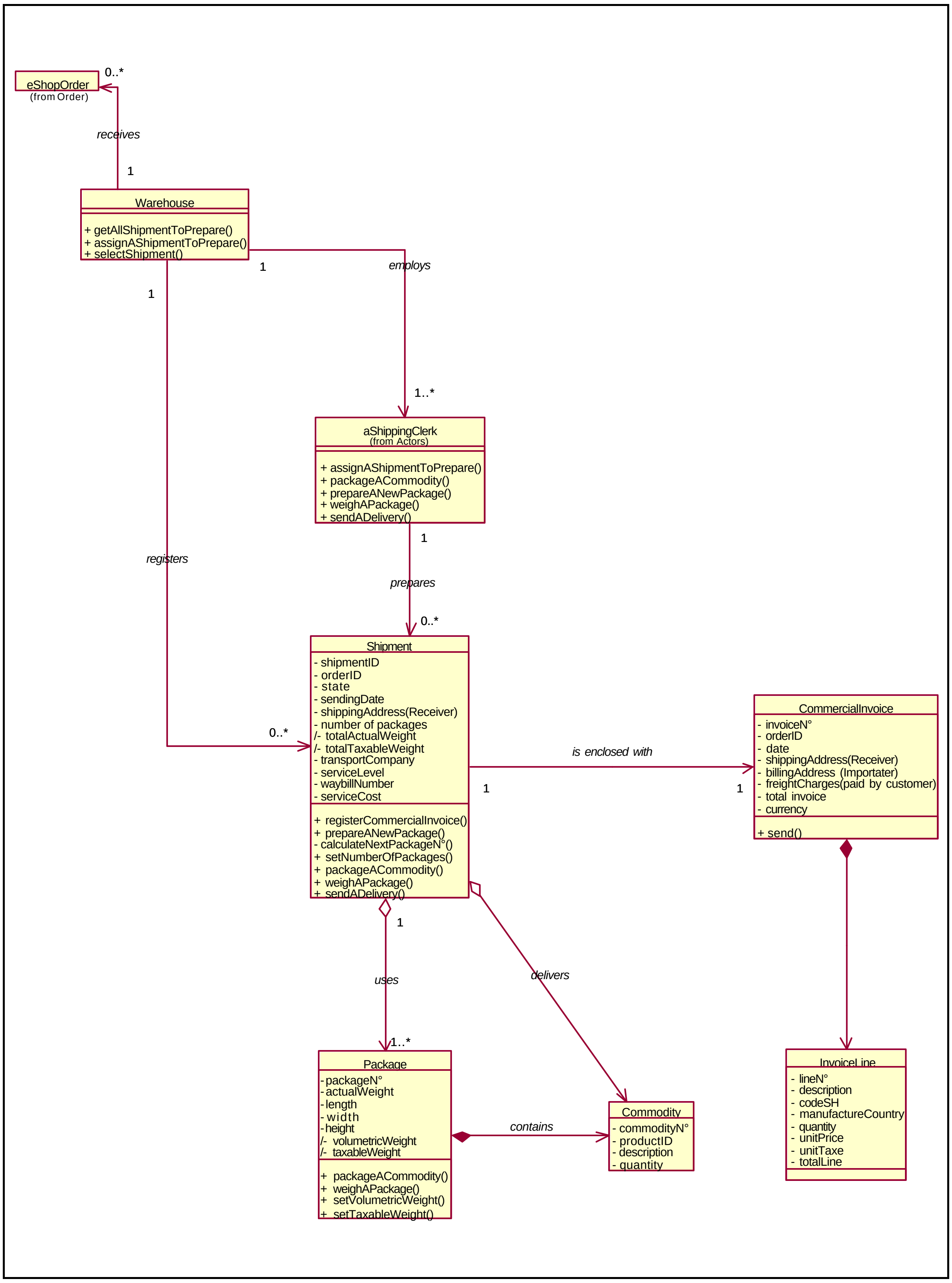


Figure 65: Class Diagram - Shipment

- The Warehouse employs one or more shipping clerks who prepare the shipments.
- Each shipment corresponds to an order (sent by e-shop). It is referred in the warehouse and is attached with the corresponding invoice. When the warehouse receives a new order, automatically a shipment and an invoice are created. The shipment is in state “to be prepared”.
- When a shipping clerk ask for a new delivery to be prepared, automatically the warehouse assigns to him the first shipment with state “to be prepared” (ordered by ID), The state of the shipment becomes “in preparation”.
- The shipment presents the list of the commodity to be delivered, which is used by the shipping clerk to search the different goods in the warehouse.
- As he prepares the packages, the shipping clerk registers them in the shipment.

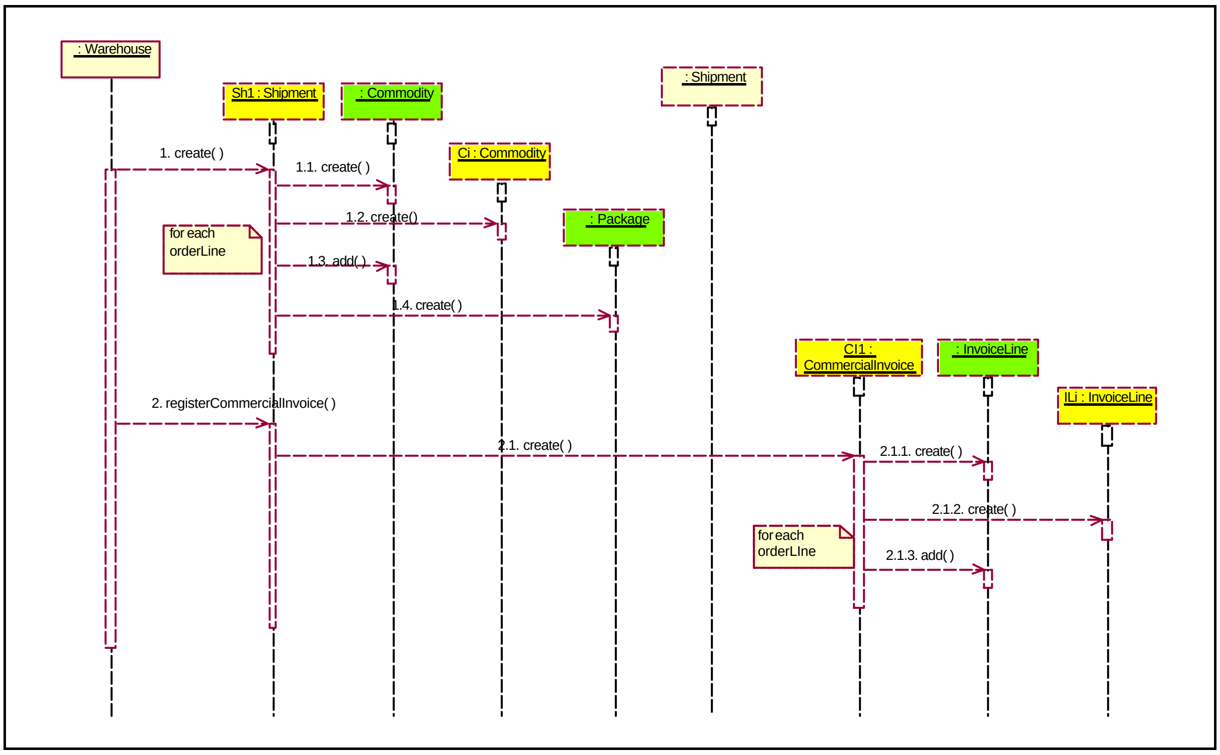


Figure 66: Sequence Diagram - Delivery's Registration

The warehouse the warehouse receives eshopOrder in XML format.

Automatically, for each received order, the warehouse pre-creates the shipment and its commodities and the invoice, which are corresponding.

1st step

The **shipment** in created with a state "to prepare", just the above information are completed,

→ shipmentID : automatic number

→ orderID : from eShopOrder

→ state : "to prepare"

→ destinationAddress : from eShopOrder

The ***list of commodities*** is created

Each commodity to be delivered by the shipment is created.

- → productID : from eShopOrderLine
- → description : from eShopOrderLine
- → quantity : from eShopOrderLine

The ***list of packages*** is created; at least, it will be one package.

2nd step

The **commercial invoice** is created, it is attached to the shipment.

- → Invoice N° : automatic number
- → Destination Address : from eShopOrder
- → Billing Address : from eShopOrder
- → Freight Charges : from eShopOrder
- → Total Invoice : from eShopOrder
- → currency : from eShopOrder

The ***list of invoice's lines*** is created.

Each invoice line is created.

- → Line N° : automatic number
- → description : from eShopOrderLIne
- → code SH ????
- → manufacture Country : from eShopOrderLine
- → quantity : from eShopOrderLine
- → unit Price : from eShopOrderLine
- → unit Taxe : from eShopOrderLine
- → total Line : from eShopOrderLine

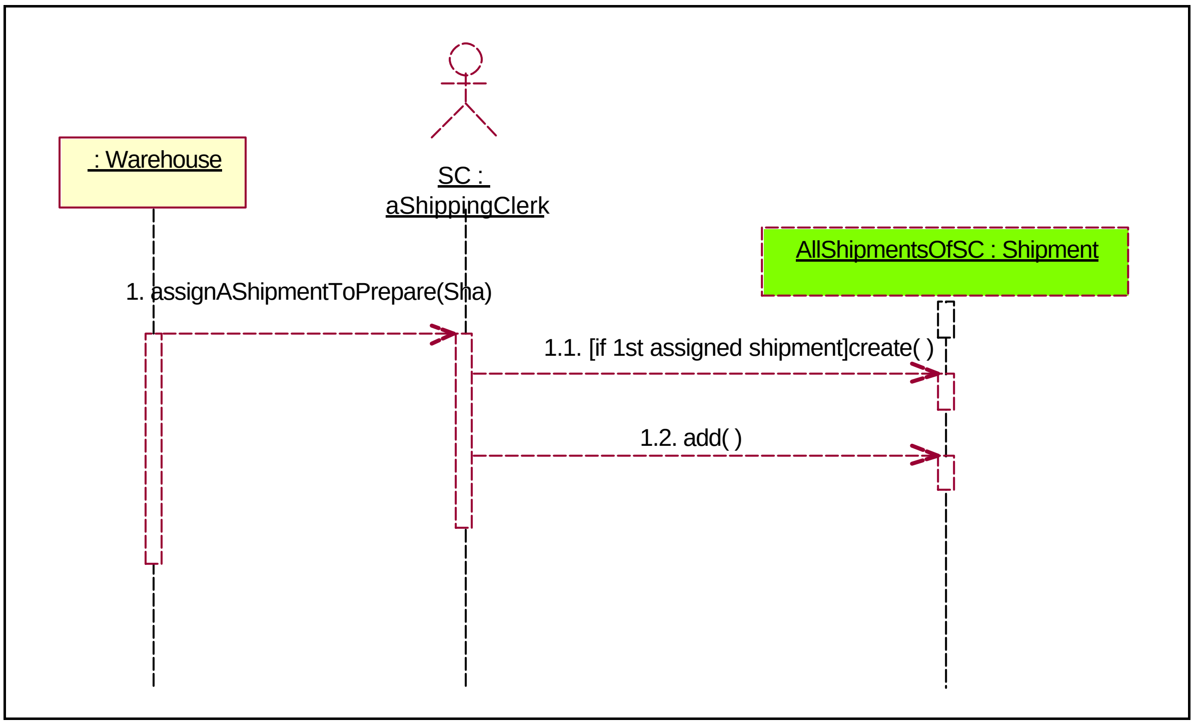

: Warehouse
SC : aShippingClerk
AllShipmentsOfSC : Shipment
1. assignAShipmentToPrepare(Sha)
1.1. [if 1st assigned shipment]create( )
1.2. add( )

Figure 67: Sequence Diagram - Shipment's assignation

The warehouse assigns a shipment "to be prepared" to a shipping clerk; the shipment is added to the list of shipments of the shipping clerk, its state becomes "in preparation".

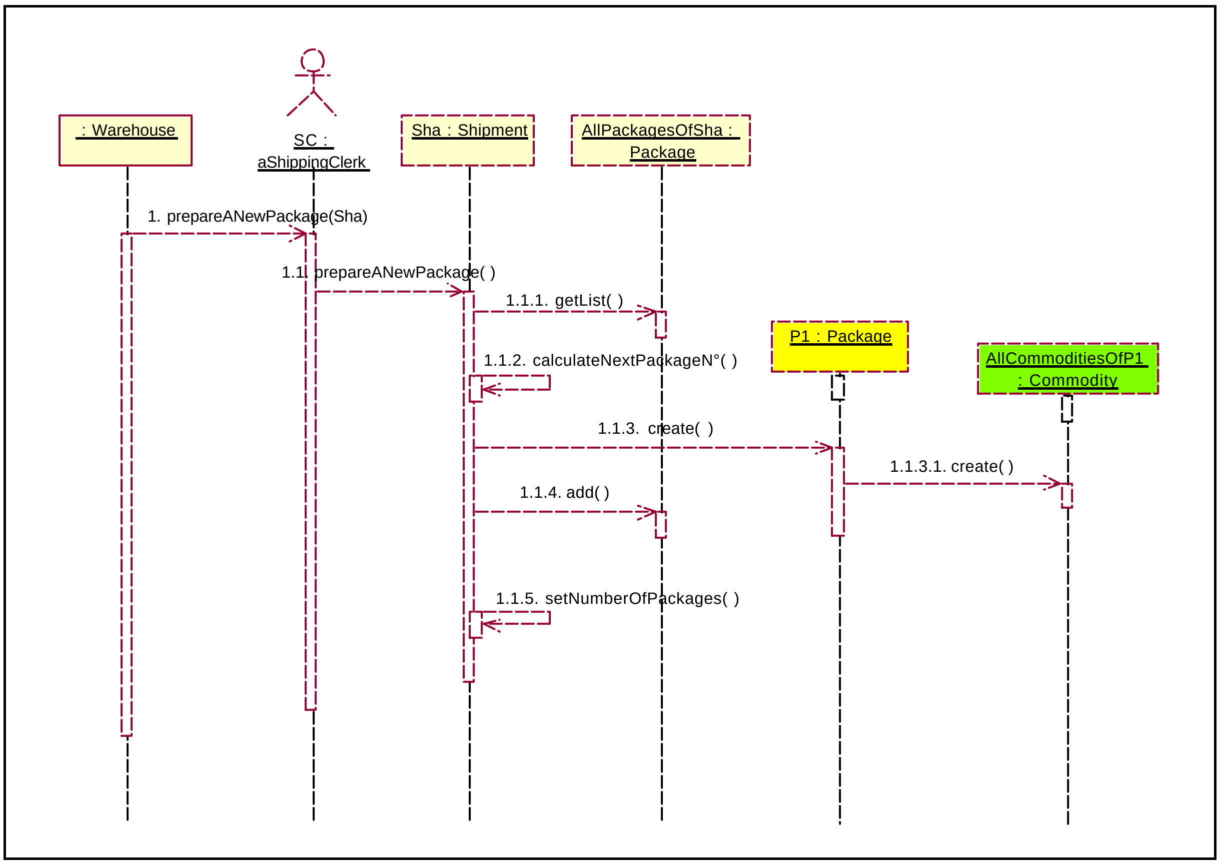

: Warehouse
SC : aShippingClerk
Sha : Shipment
AllPackagesOfSha : Package
P1 : Package
AllCommoditiesOfP1 : Commodity
1. prepareANewPackage(Sha)
1.1. prepareANewPackage( )
1.1.1. getList( )
1.1.2. calculateNextPackageN°( )
1.1.3. create( )
1.1.3.1. create( )
1.1.4. add( )
1.1.5. setNumberOfPackages( )

Figure 68: Sequence Diagram - Package's registration

For each new package, the shipping clerk registers it in the shipment.

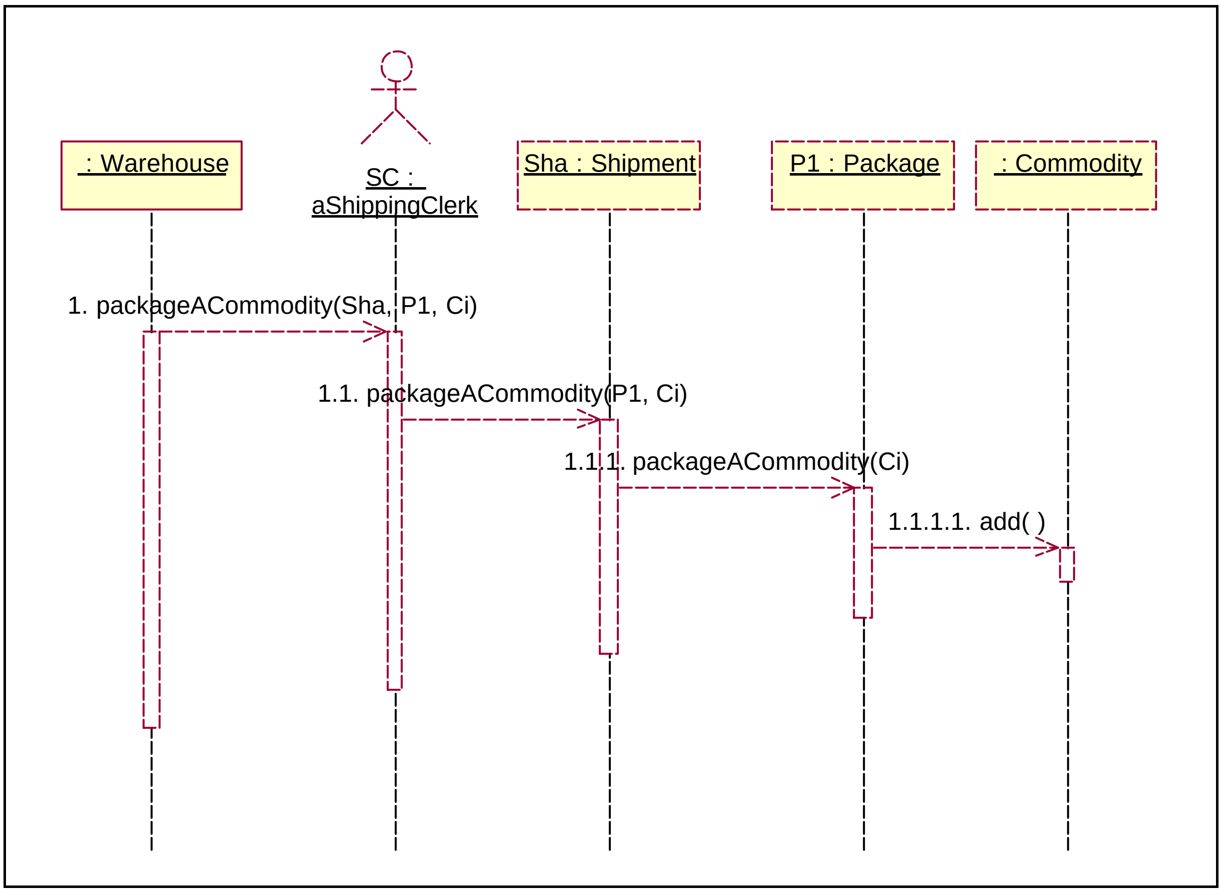

: Warehouse
SC : aShippingClerk
Sha : Shipment
P1 : Package
: Commodity
1. packageACommodity(Sha, P1, Ci)
1.1. packageACommodity(P1, Ci)
1.1.1. packageACommodity(Ci)
1.1.1.1. add( )

Figure 69: Sequence Diagram - commodity's packing

Then, the shipping clerk adds each packed commodity to the list of commodities of the package.

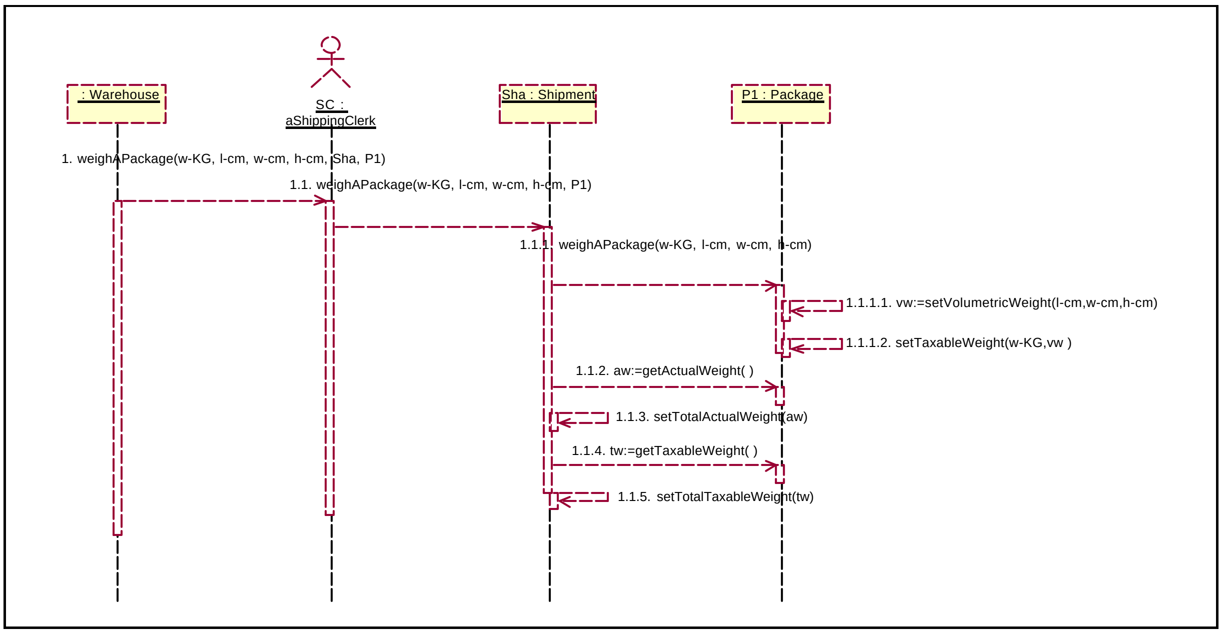

: Warehouse
SC : aShippingClerk
Sha : Shipment
P1 : Package
1. weighAPackage(w-KG, l-cm, w-cm, h-cm, Sha, P1)
1.1. weighAPackage(w-KG, l-cm, w-cm, h-cm, P1)
1.1.1. weighAPackage(w-KG, l-cm, w-cm, h-cm)
1.1.1.1. vw:=setVolumetricWeight(l-cm,w-cm,h-cm)
1.1.1.2. setTaxableWeight(w-KG,vw)
1.1.2. aw:=getActualWeight( )
1.1.3. setTotalActualWeight(aw)
1.1.4. tw:=getTaxableWeight( )
1.1.5. setTotalTaxableWeight(tw)

Figure 70: Sequence Diagram - package's weighing

Then the shipping clerk weighs the package and registers the weight.

The taxable weight is calculated = maximum {actual weight, volumetric weight}.

The total of actual weight and the total of taxable weight of the shipment are calculated.

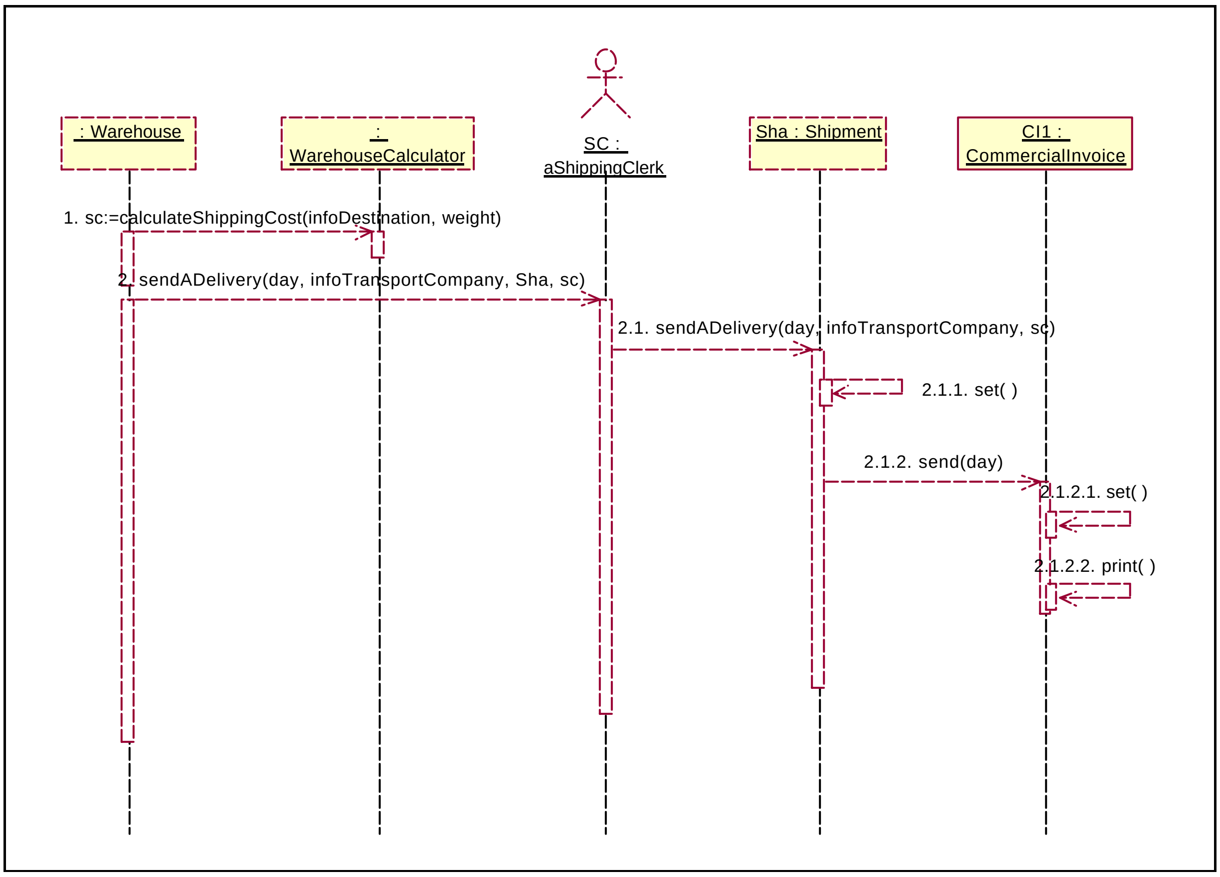

: Warehouse
: WarehouseCalculator
SC : aShippingClerk
Sha : Shipment
CI1 : CommercialInvoice
1. sc:=calculateShippingCost(infoDestination, weight)
2. sendADelivery(day, infoTransportCompany, Sha, sc)
2.1. sendADelivery(day, infoTransportCompany, sc)
2.1.1. set( )
2.1.2. send(day)
2.1.2.1. set( )
2.1.2.2. print( )

Figure 71: Sequence Diagram - delivery's sending

Finally, the shipping cost is calculated by the warehouse calculator.

The information of shipment and invoice is completed by sending date, shipping cost.

The invoice is printed.

4.5 Knowledge Base Access

4.5.1 Introduction

Ontology System is the central node of the REGNET infrastructure, as it connects to all other REGNET nodes exchanging useful metadata and information with them. This data exchange is accomplished through the use of the Java RMI protocol, which enables the transfer between remotely located systems. The data is stored in Knowledge Base, the XML metadata database of Ontology. Apart from the Knowledge Base, the server should provide functions for sending and receiving files, as well as classifying them according to their XML content.

4.5.2 Design model hierarchy

Ontology system implemented at **org.regnet.ontology** package consists of the following classes:

Package: org.regnet.ontology	
Class Name	**Description**
FilePacket	Class to represent a File object that can be sent on another system.
FileReceiver	Interface that represents a server that allows the uploading of a file.
FileSender	Class that implements the uploading of a file to the remote server.
FileSendInitialize	Initialises the interface. Extends Remote (rmi).
RemoteFileSendInitialize	Initialises the remote interface. An implementation of FileReceiver.
RemoteReceiver	Implements FileReceiver in order to receive files.
XTMLoader	Implements Topic Map Generator Tool.

Package: org.regnet.ontology
Relationships with other packages: Electronic Publisher, Portal, E-business data management, CH data management
Packages owned by org.regnet.ontology: None

4.5.3 Diagrams of the design model

4.5.3.1 UML diagram of the FilePacket class (part of org.regnet.ontology package)

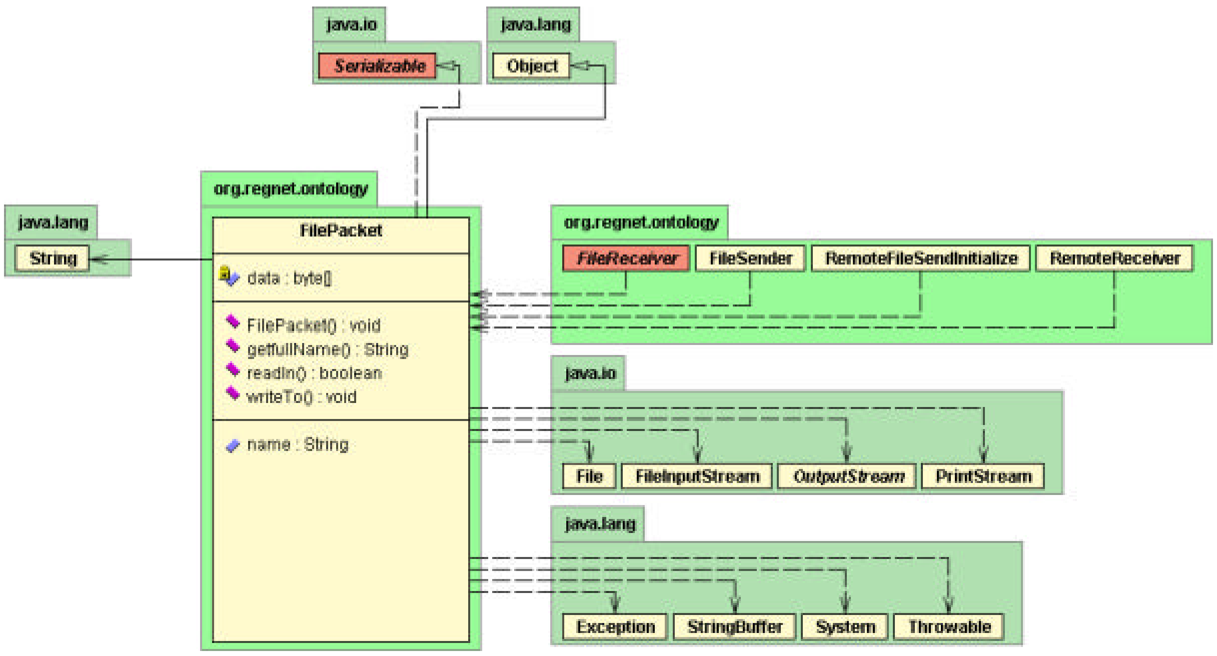


Figure 72: FilePacket class UML diagram.

4.5.3.2 UML diagram of the FileReceiver class (part of org.regnet.ontology package)

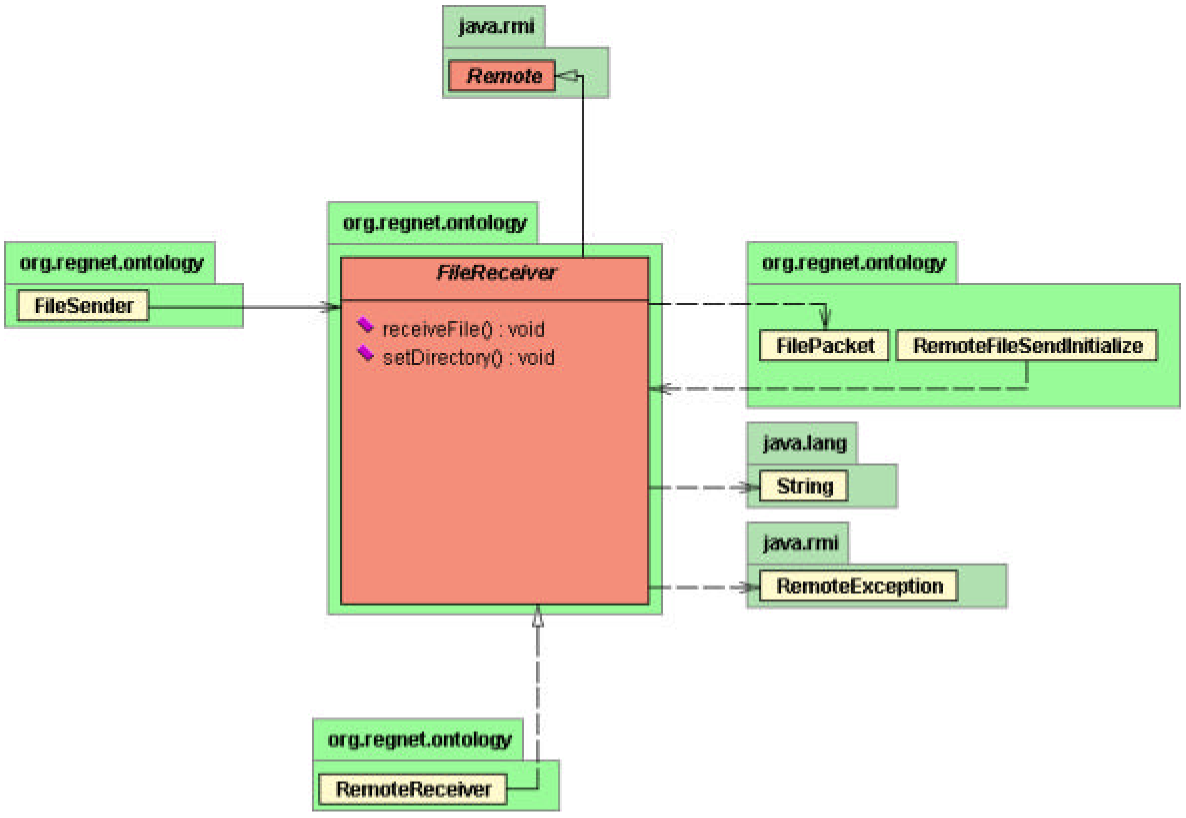


Figure 73: FileReceiver class UML diagram.

4.5.3.3 UML diagram of the FileSender class (part of org.regnet.ontology package)

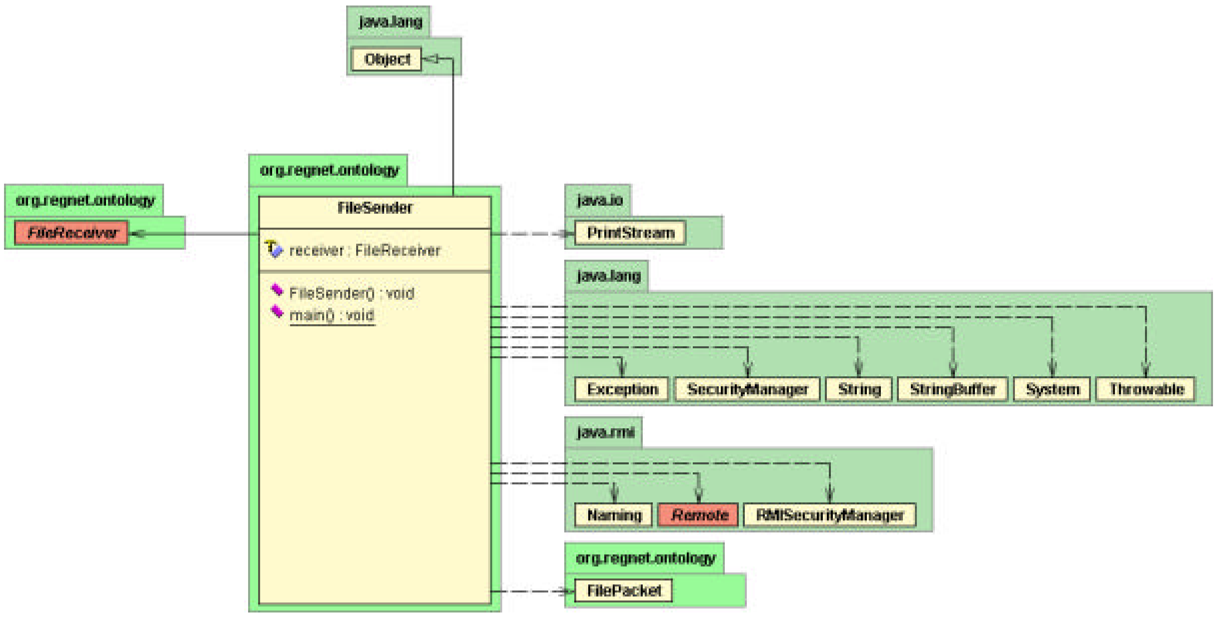


Figure 74: FileSender class UML diagram.

4.5.3.4 UML diagram of the FileSendInitialize class (part of org.regnet.ontology package)

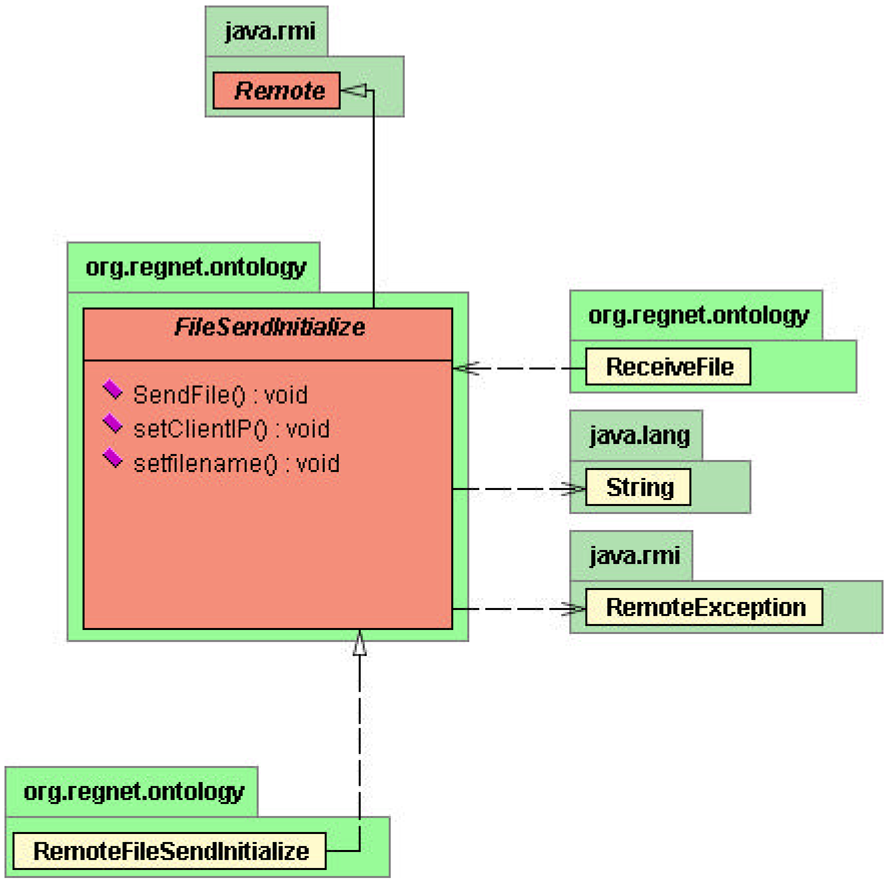


Figure 75: FileSendInitialize class UML diagram.

4.5.3.5 UML diagram of the ReceiveFile class (part of org.regnet.ontology package)

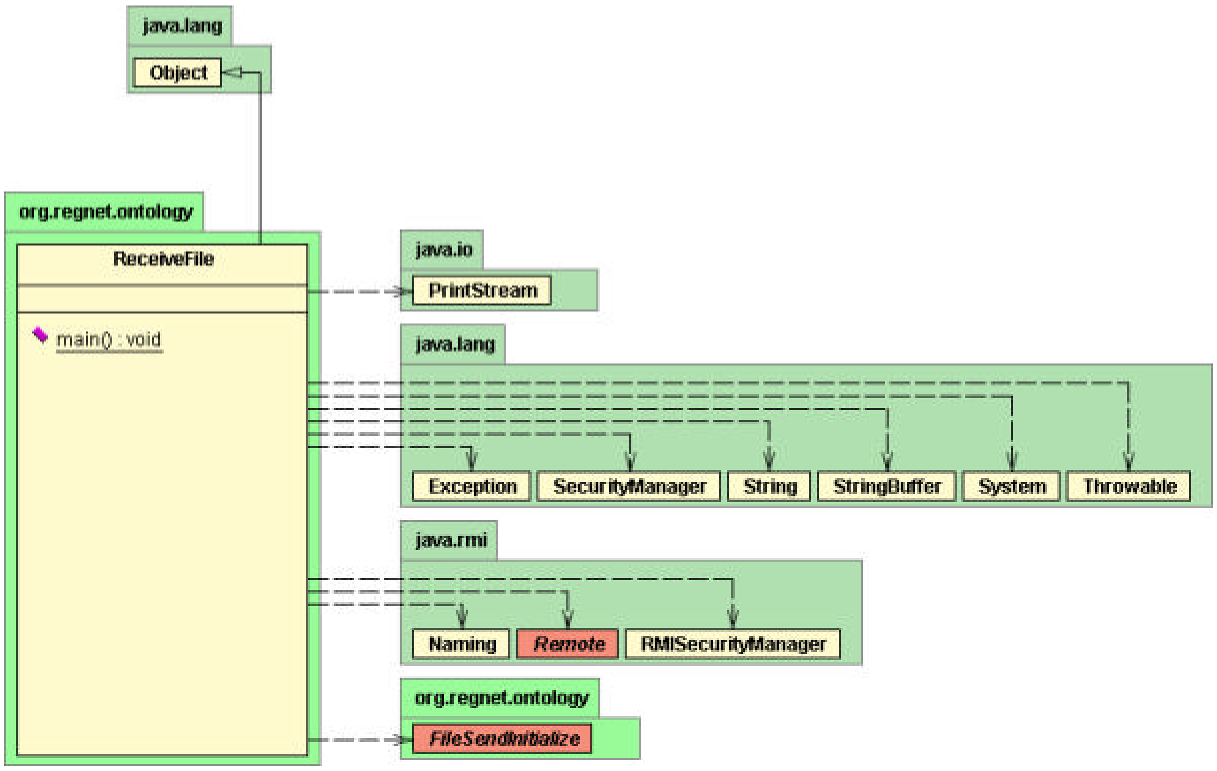


Figure 76: ReceiveFile class UML diagram.

4.5.3.6 UML diagram of the RemoteFileSendInitialize class (part of org.regnet.ontology package)

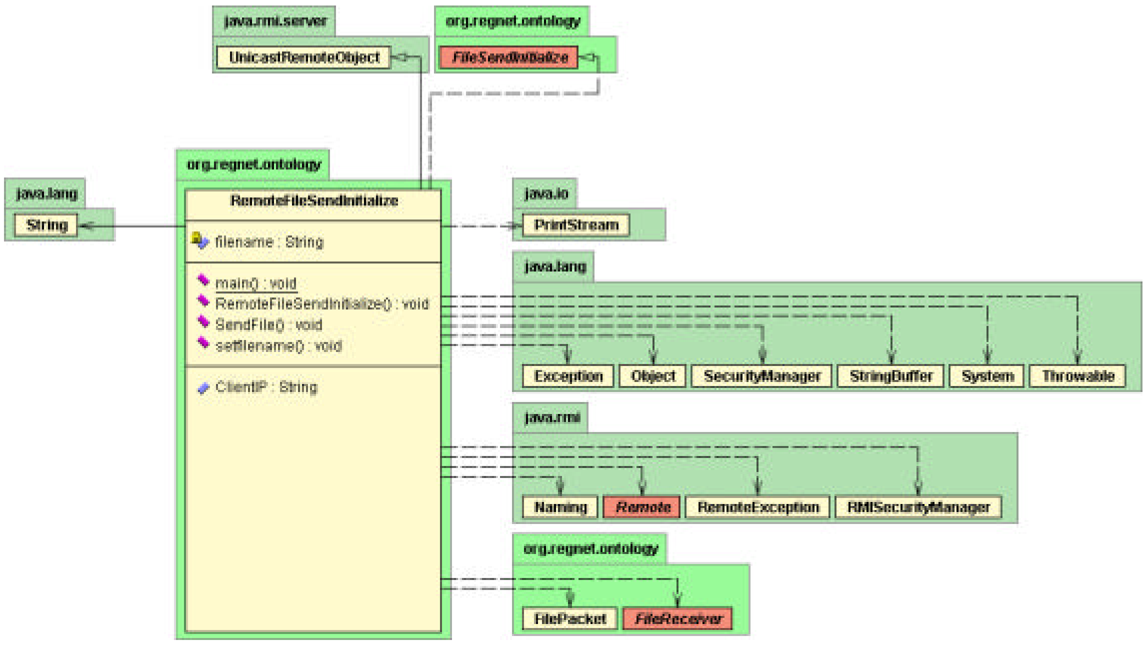


Figure 77: RemoteFileSendInitialize class UML diagram.

4.5.3.7 UML diagram of the RemoteReceiver class (part of org.regnet.ontology package)

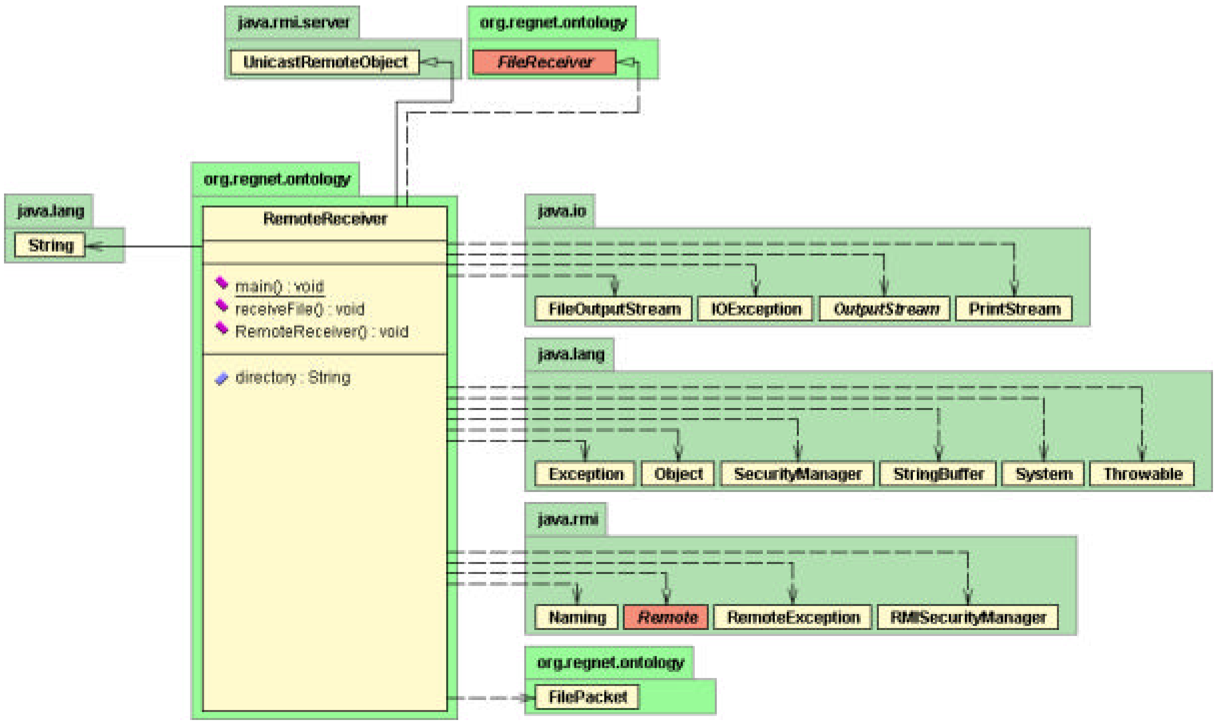


Figure 78: RemoteReceiver class UML diagram.

4.6 Electronic publisher

4.6.1 Introduction

The design model of the electronic publisher follows the activity diagram introduced in the prototype section of this document:

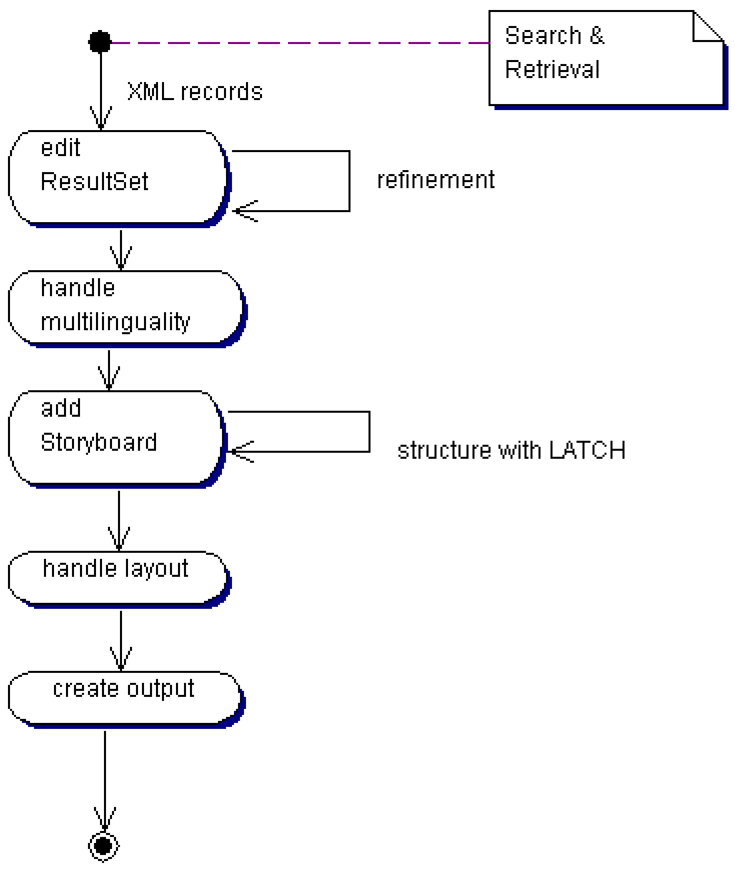


Figure 79: Activity Diagram - Publisher

The activities are a detailed view of the activity diagram of D2 (deliverable 2). In a technical sense they are structured in the following steps: input (via Search & Retrieval), multilinguality (handle multilingualtiy), layout (handle layout), organization (edit ResultSet and add Storyboard) and output (create output).

The following table shows the mapping between the Use Cases of D2 and the processing steps shown in the above activity diagram and whether they are realised in the current implementation.

UC Identifier	Use Case	Processing Step	Implemented
UC 5.5	edit resultset	input	no
UC 5.5.1	delete record	input	no
UC 5.5.2	edit single record	input	no
UC 5.6	add storyboard	organization	implemented for location based information
UC 5.7	create output	layout, output	yes
UC 5.7.1	create PDF output	output	yes
UC 5.7.2	create SMIL output	output	yes
UC 5.7.3	create HTML output	output	yes
UC 5.8	create printer-friendly version	output	yes

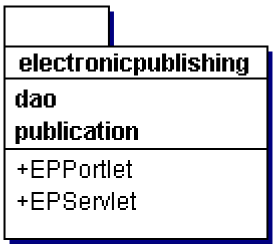
UC 5.9	generate product	layout	no
UC 5.10	publish a theme	layout	no
UC 5.10.1	use theme storyboard	organization	no
UC 5.11	create an exhibition	layout	implemented for location based information
UC 5.12	create a virtual gallery	layout	implemented for location based information
UC 5.13	create catalogue	layout	yes

Table 9: Use Cases – Electronic Publisher

4.6.2 Design model hierarchy

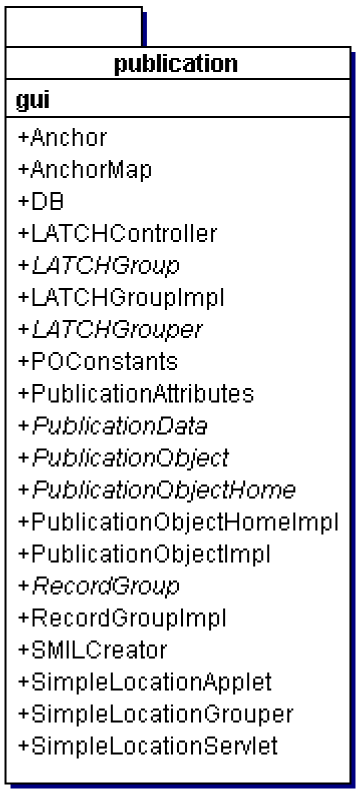
The following diagram shows the main classes / packages of the Electronic Publisher.

The package `electronicpublishing` contains all classes and packages related to the Electronic Publishing process, excluding the "edit resultset" step.

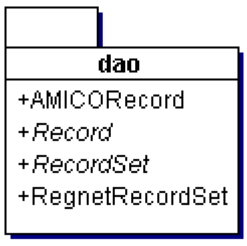


- `EPPortlet`: The portlet forms the connection between the Portal and the Electronic Publishing component. So far the portlet just defines the link to the server running the `EPServlet`.
- `EPServlet`: The `EPServlet` makes the functionality of the Electronic Publisher accessible via a HTML / Applet interface. It controls the flow of information within the publishing process.

The package `publication` contains all classes needed for restructuring and formatting the input data into the desired output format. It therefore defines classes for the realisation of the LATCH principles as well as classes for the realisation of the SMIL formatting, etc. Details to the classes of the `publication` package can be found in the subsequent section.



The package `dao` realises the access to different sources of data and the transformation of the data into the internal data structures of the Electronic Publisher. Details to the classes of the `dao` package can be found in the subsequent section.



4.6.3 Diagrams of the design model

The following diagram shows the structure of the `publication` package.

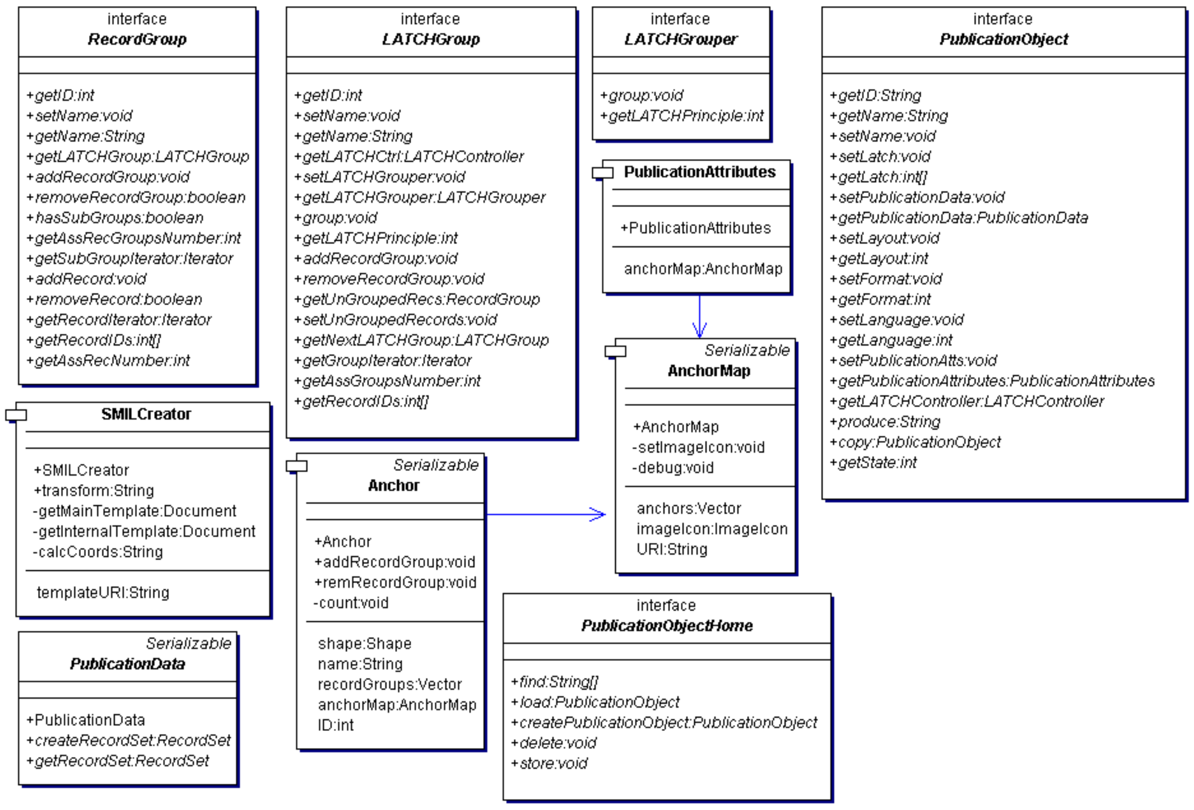


Figure 80: Class Diagram – Electronic Publisher

Class	Description
PublicationObject	Interface defining methods for access to publication data and attributes of a publication.
PublicationObjectHome	Interface defining methods for accessing and managing publication objects
PublicationAttributes	Class defining the selected criteria for a publication object
AnchorMap	Class storing a map and assigned anchors
Anchor	Class which holds a reference to its visual representation (Rectangle) and to assigned RecordGroups
LATCHGrouper	Class which performs grouping dependent on the underlying LATCHPrinciple
LATCHGroup	Interface, which is implemented by a class, which represents all Records, which are grouped by the same principle. Chaining principles will require several LATCHGroups
RecordGroup	Interface, which is implemented by a class, which is intended to bundle some records.
PublicationData	Interface, which is implemented by a class, which retrieves the records and creates the internal representation (RecordSet) out of it.
SMILCreator	Class which performs transformation out of a publication object into a SMIL document

The following diagram shows the structure of the `dao` package.

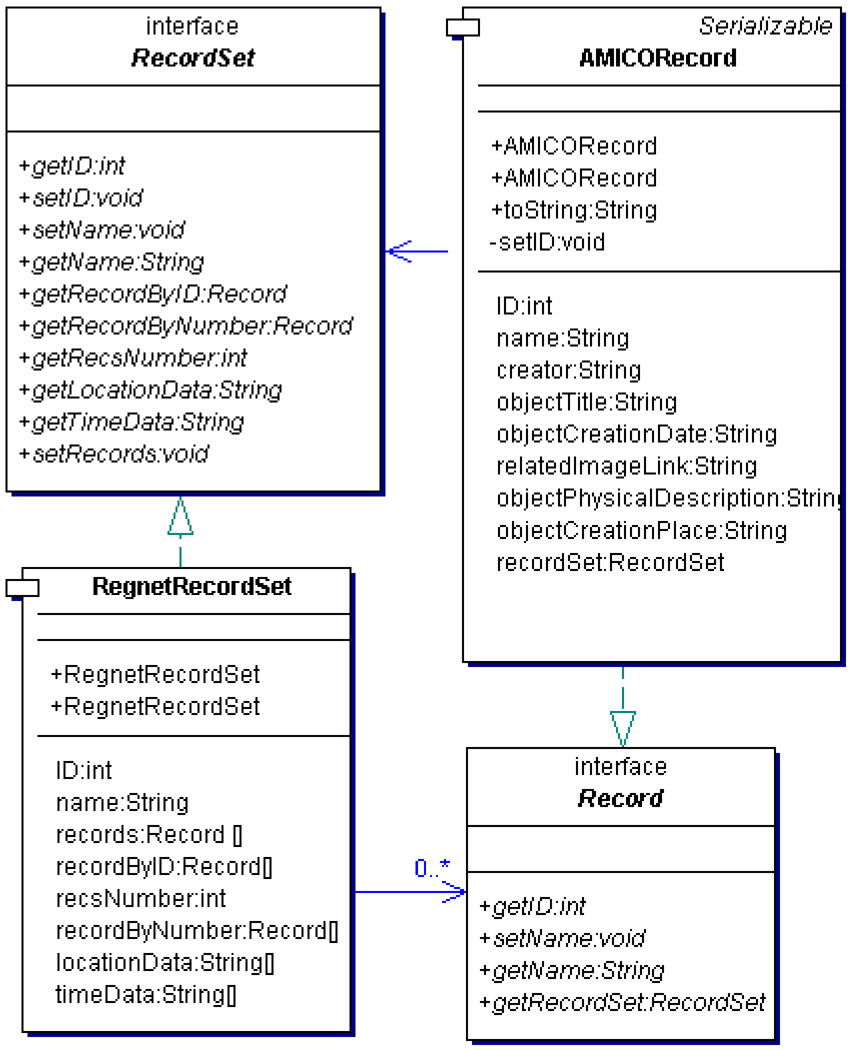


Figure 81: Class Diagram – Electronic Publisher

Class	Description
RecordSet	Interface, which is implemented by the class, which is the internal representation of the retrieved data.
RegnetRecordSet	Class implementing the RecordSet interface.
Record	Interface providing methods for access to records.
AMICORecord	Wrapper class for AMICO based Records

5 Implementation Model

The **implementation model** is a collection of components, and the implementation subsystems that contain them. Components include both deliverable components, such as executables, and components from which the deliverables are produced, such as source code files.

The implementation model is a composite, comprehensive artefact, which encompasses all artefacts needed to build and manage the system in the run-time environment.

5.1 Component diagram

As describes in the previous Work-Package, the interconnection between different REGNET components are presented in the following figure:

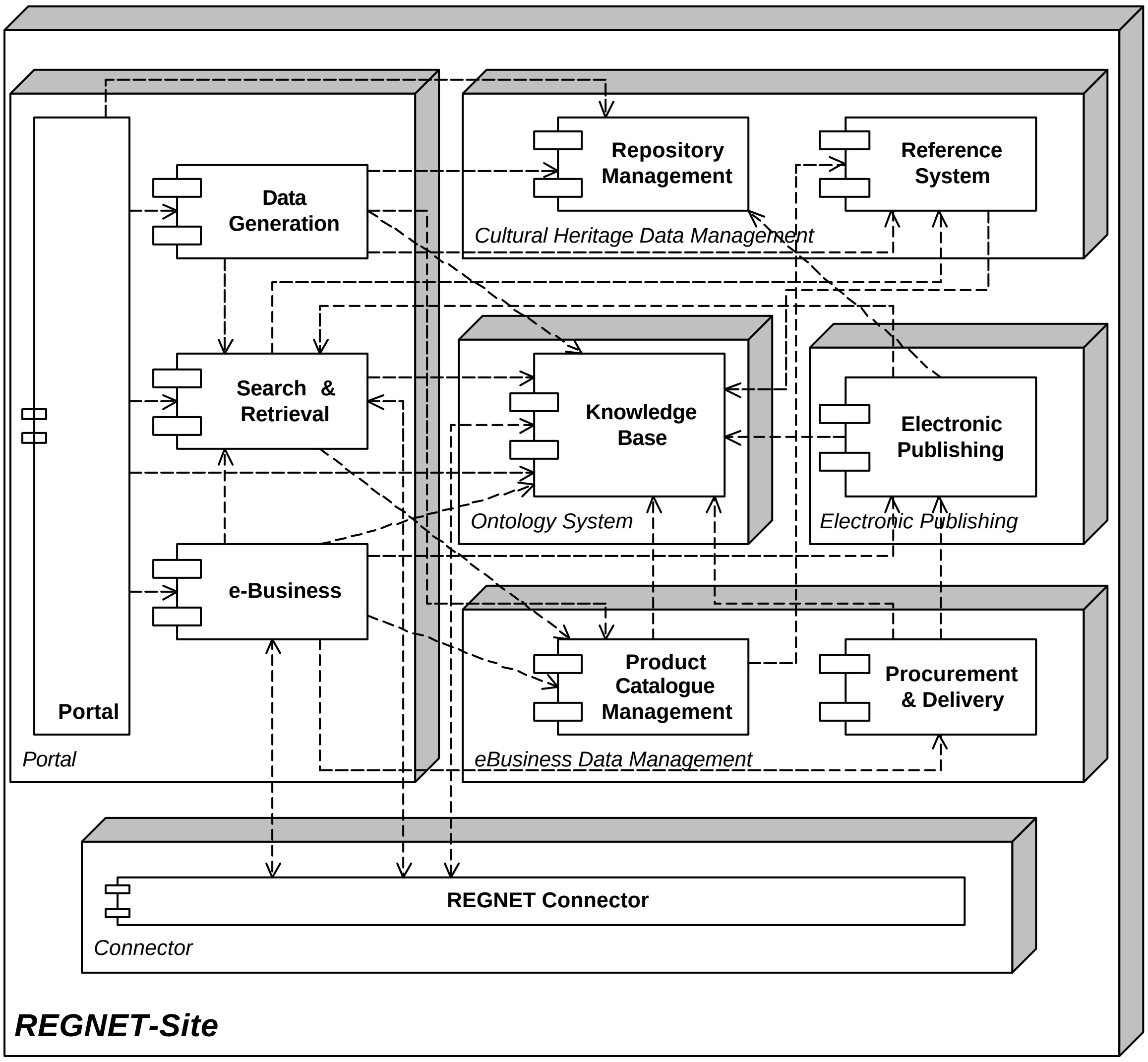
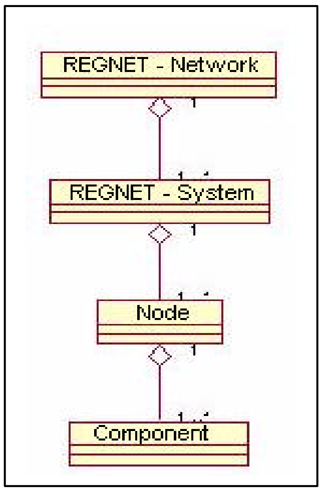


Figure 82: The REGNET-Site



A REGNET-Site can include all REGNET nodes (Portal, etc), but it is possible that a site contains only a subset of nodes, e.g. if a centre is not providing e-Business functions the e-Business related components will not constitute this site.

The conceptual framework for a REGNET Network is presented in the left figure:

- The REGNET Network consists of one or more
- REGNET Systems (Sites), which include one ore more
- REGNET Nodes. A REGNET Node can include one or more
- REGNET Components.

In the first version of the Regnet system we have developed a standalone system. It means that the system must be completely deployed in the CSC local area network. As states before, a proof of concept prototype has been developed for the ebXML based Regnet connector, but this component is not useful now.

5.2 Deployment of components

A detail description of the deployment of each component is given into technical annex named “components deployment and installation”.

6 Deployment Plan

The Deployment Plan describes the set of tasks necessary to install and test the developed product such that it can be effectively transitioned to the user community.

6.1 Deployment Planning

This part describes all activities performed in deploying the product to the customer. Activities include planning, beta testing, preparing items to be delivered, packaging, shipping, installation, training, and support.

6.1.1 Responsibilities

General groups are already outlined in the Annex 1 "Description of Work" – Appendix A of the contract. Content Providers (C), Developers (S) and Regional Poles (B) and their relations are described there. This structure will be reflected in the next table:

C	…	ONB, LMG, NRM, KVA, ALI, MECH, GRAN
S	…	SR, SI, CERT, VALT, MOT, AIT, TARX, ZEUS
B	…	SUL, MUS, CC, IAT, IMAC, TARX, SPAC, ZEUS, ICCS, AIT

User Group	Task	Partner(s)
General System Users	select users for beta testing (testing functionalities for unregistered and registered users)	IMAC
	provide and analyse questionnaires	IMAC
Content Providers	provide and update content	C
	provide and update additional information (news, events, ...)	C
	test functionalities in relation with their own content	C
	test functionalities in relation with their special domain (museum, archive, ...)	C
Developers	prototype integration + integration tests	VALT
	provide a bug reporting/tracking system	VALT
	operate development sites	AIT
	beta tests on development sites	VALT (+ S)
	test connection between different sites	VALT
	provide installation, update, de-installation scripts for all system components.	VALT
	provide correct documentation corresponding to system version.	VALT (+S)
	training and support for system operators	VALT
	convert databases (from content providers)	AIT
Operators	operate deployment sites	AIT, IMAC, TARX, ICCS
	test installation, update, de-installation scripts	AIT, IMAC, TARX,

		ICCS
	announce deployed systems (e-mail, press, ...)	AIT
	training for supporters	AIT, IMAC, TARX, ICCS, SUL, CC
Supporters	support system after final deployment	AIT, IMAC, TARX, ICCS, SUL, CC
	plan and provide training for new customers / users	AIT, IMAC, TARX, ICCS, SUL, CC

Table 10: Deployment responsibilities

There are 4 service centers (deployment sites) planned in the following regions (see Annex 1 “Description of Work” – Appendix A, page 3):

- Region 1 – Middle and Northern Europe

  a) Operator: AIT, Content Provider: ONB

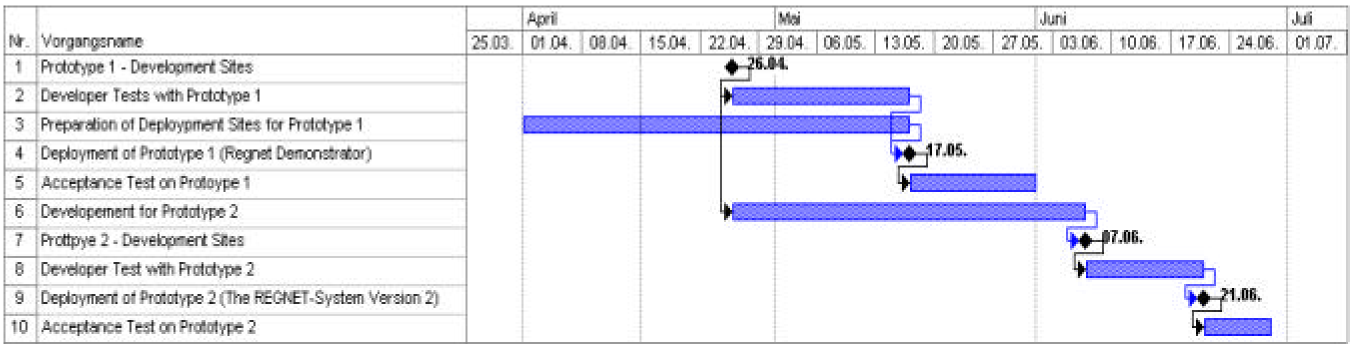
  b) Operator: IMAC, Content Provider: SUL, LMG, NRM, KVA
- Region 2 – Western Europe

  Operator: TARX, Content Provider: MUS, MECH
- Region 6 – Eastern Europe

  Operator: ICCS, Content Provider: SUSU

6.1.2 Schedule

Deployment Activities and Milestones:



Nr.	Vorgangsname
1	Prototype 1 - Development Sites
2	Developer Tests with Prototype 1
3	Preparation of Deployment Sites for Prototype 1
4	Deployment of Prototype 1 (Regnet Demonstrator)
5	Acceptance Test on Protoype 1
6	Developement for Prototype 2
7	Prottpye 2 - Development Sites
8	Developer Test with Prototype 2
9	Deployment of Prototype 2 (The REGNET-System Version 2)
10	Acceptance Test on Prototype 2

Figure 83: Deployment Schedule

6.2 Resources

This part lists the resources and their sources required to carry out the planned deployment activities.

6.2.1 Facilities

No special facilities are needed in order to deploy the Regnet system. The CSC need to provide an external Internet access in a securised DMZ (DeMilitarized Zone) if possible.

6.2.2 Hardware

Example:

Service Center Austria: running on Windows 2000 (except Repository Management)

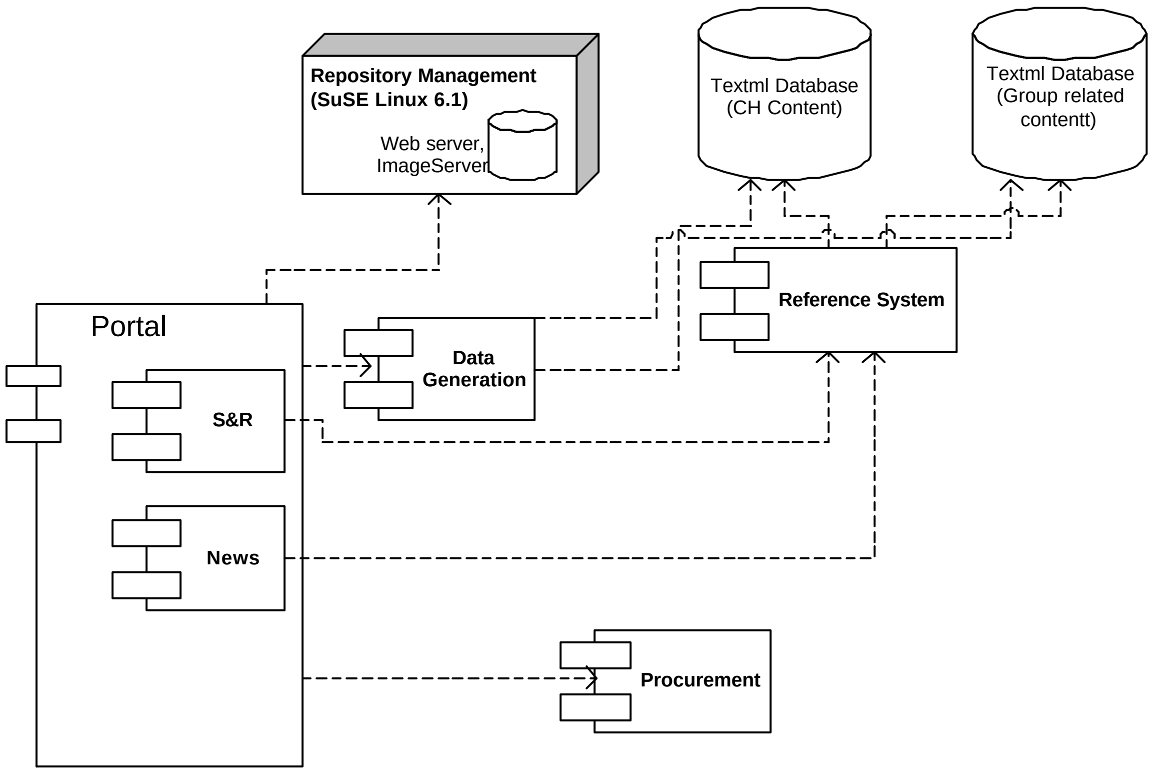

Repository Management (SuSE Linux 6.1)
Web server, ImageServer
Textml Database (CH Content)
Textml Database (Group related contentt)
Reference System
Portal
Data Generation
S&R
News
Procurement

Figure 84: Example Deployment for Prototype

6.2.3 The Deployment Unit

6.2.3.1 Support Software

Regnet Server requirements (see technical annex for software version details):

Operating System	Linux
Hardware (minimum)	Pentium processor 200 MHz, 128 MB RAM, 1 GB disk space
Software	Apache/Tomcat Java Virtual Machine PHP framework Apache/Jetspeed Apache/SOAP Apache/Xerces JDOM DbXML server

TEXTML Server requirements:

Operating System	Windows NT4, Windows 2000, Windows XP
Hardware	Pentium processor 200 MHz, 128 MB RAM, 40 MB disk space
Software	Internet Explorer 5 (version 5.00.3xxx.xxxx) Service Pack 1. Microsoft XML Parser version 3 (Service Pack 1 recommended) for NT4 only: Service Pack 6a Microsoft Transaction Server (MTS), from Option Pack 4. Microsoft Management Console 1.2 (MMC), from Option Pack 4.

.NET Framework requirements:

Operating System	Microsoft Windows NT® 4.0 (SP 6a required) Microsoft Windows® 2000 (SP 2 recommended) Microsoft Windows XP Professional
Hardware	Pentium 133 MHz, 128 MB RAM
Software	see http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconnetframeworksystemrequirements.asp

6.2.3.2 Support Documentation

Installation documentation for each module is needed in order to install the software.

6.2.3.3 Support Personnel

Installation is supervised by technical experts from Regnet consortium. Valtech is provided such personnal.

6.3 Training

Training material will be provided by the work-package 3.

7 Glossary

Acronym	Description	Link
A2A	Application to Application Integration	
AAT	Arts and Architecture Thesaurus	http://www.getty.edu/research/tools/vocabulary/aat/index.html
ADO	ActiveX Data Object	
AF	Architectural Forms (HyTime)	http://xml.coverpages.org/hytime.html
AHDS	Arts and Humanities Data Service	http://ahds.ac.uk/
AITF	Art Information Task Force	http://www.getty.edu/research/institute/standards.html
AMICO	Art Museums Image Consortium	http://www.amico.org/
ANSI	American National Standards Institute	http://www.ansi.org/
Apache JServ	Servlet Engine fully Servlet API 2.0 compliant	http://java.apache.org/jserv/
Apache Tomcat	Reference Implementation of JAVA Servlet and JAVA Serverpages Technologie	http://jakarta.apache.org/tomcat/
API	Application Programming Interface	
ASN.1	Abstract Syntax Notation One	http://www-sop.inria.fr/rodeo/personnel/hoschka/asn1.html
ASP.NET (.NET framework)	New development framework from Microsoft.	http://www.microsoft.com/net/
B2B	Business to Business	
B2C	Business to Consumer	
B4B	Business for Business	
BATIK	Toolkit for developing SVG-Applications	http://xml.apache.org/batik/
BEA Weblogic	The #1 Web and wireless application server	http://www.bea.com/products/weblogic/server/index.shtml
BizTalk	Resources for Learning XML	http://www.biztalk.org
BOV	Business Operational View - One View of UMM	
BUC	Business Use Case	
C2C	Consumer to Consumer	
CatXML	Open interoperable standard for catalogue information exchanges	http://www.catxml.org/
CDWA	Categories for the Description of Works of Art	

CDWF	Categories for the Description of Works of Art	http://www.getty.edu/research/institute/standards/cdwa/
CEN/ISSS	European Committee for Standardization/ Information Society Standardization System	http://www.cenorm.be/isss/
CGI	Common Gateway Interface	
CH	Cultural Heritage	
CIDOC	Le Comité international pour la documentation du Conseil international des musées	http://www.cidoc.icom.org/
CIMI	Consortium Museum Intelligence	http://www.cimi.org/
CIO	Cultural Institutions and Organisations	
COCOON	Publishing Framework	http://xml.apache.org/cocoon/
CommerceNet		http://www.commercenet.com
Component-X	XML- and JAVA-based Web services platform	http://www.enterprise-component.com/
CORBA	Common Object Request Broker Architecture	
COVAX	Contemporary Culture Virtual Archives in XML	http://www.covax.org/
CPP	Collaboration Protocol Profile	http://www.ebxml.org
CRM	Conceptual Reference Model	http://cidoc.ics.forth.gr
CS	Coding Schemes (MPEG- 7)	
DBD	Data Base Driver	
DBI	Data Base Interface	
dbXML	native XML Database	http://www.dbxml.org/
DC	Dublin Core	http://dublincore.org/
DCES	Dublin Core Element Set	http://dublincore.org/documents/dces/
DCMES	Dublin Core Metadata Element Set	http://dublincore.org/documents/dces/
DCMI	Dublin Core Metadata Initiative	http://dublincore.org/
DCOM	Distributed Component Object Model	
DDL	Description definition language (MPEG- 7)	
DESIRE	Project focussed on enhancing existing European information networks for research users across Europe	http://www.desire.org/
DG	Data Generation: Part of the REGNET System	
DHTML	Dynamic HTML	http://builder.cnet.com/Resources/Info/Glossary/Terms/dhtml.html
DMZ	De- militarized Zone	

DOM	Document Object Model	http://www.w3.org/DOM/
DS	Description Scheme (MPEG- 7)	
DSMCC	Digital Storage Media Command and Control	
DTD	Document Type Definition	
DVD	Digital Versatile Disc	
EAD	Encoded Archival Description	http://www.loc.gov/ead/
EAI	Enterprise Application Integration	
EB	E- Business	
EBNF	Extended Backus-Naur Form	http://www.cl.cam.ac.uk/~mgk25/iso-ebnf.html
EBU	European Broadcasting Union	http://www.ebu.ch/
EBU P/META	Metadata Project of EBU	http://www.ebu.ch/pmc_meta.html
ebXML	Goal: To provide an open XML-based infrastructure enabling the global use of electronic business information in an interoperable, secure and consistent manner by all parties.	http://www.ebxml.org
EC	Electronic Components	
ECMA	European Computer Manufacturers Association Standardizing Information and Communication Systems	http://www.ecma.ch/
EDI	Electronic Data Interchange	
EDIFACT	EDI standard	http://www.edifact-wg.org
EJB	Enterprise JavaBeans	http://java.sun.com/products/ejb/
ELISE	Electronic Library Image Service for Europe	http://nile.dmu.ac.uk/elise/e2_intro.html
EP	Electronic Publishing: Part of the REGNET System	
ERP	Enterprise Resource Planning	
EULER	The aim of the project is to provide strictly user-oriented, integrated network based access to mathematical publications.	http://www.emis.de/projects/EULER/
FR	Functional Requirements	
FSV	Functional Service View	http://www.ebxml.org
GLUE	A Java API for SOAP messaging and XML processing.	http://www.themindelectric.com
GPRS	General Packet Radio Service	
GRINS	A Smil Editor	http://www.oratrix.com/GRiNS/
GUI	Graphical User Interface	

HDML	Handheld Device Markup Language	
HL7	Health Level 7 - a medical Standard	http://www.hl7.org/
HTML	Hyper Text Markup Language	
HTML+TIME	Timed Interactive Multimedia Extensions for HTML, a w3c submission by Microsoft	http://www.w3.org/TR/NOTE-HTMLplusTIME
HTTP	Hyper Text Transfer Protocol	
HTTPS	Hyper Text Transfer Protocol over SSL	
HyTime	The HyTime standard actually defines a number of related but largely independent architectures and facilities	http://www.hytime.org/
IAI	Internet Application Integration	
IASA	International Association of Sound and Audiovisual Archives	http://www.iasa.org/
ICA	International Council of Archives	http://www.ica.org/
ICOM	International Council of Museums	http://www.icom.org/
IDL	Interface Description Language	
IETF	The Internet Engineering Task Force	http://www.ietf.org/
IFLA	International Federation of Library Associations and Institutions	http://www.ifla.org/
IIOP	Internet Inter-ORB Protocol	
INTERMARC	a MARC format	
IONA/NETFISH	Company/product	http://www.iona.com/
ISAD(G)	General International Standard Archival Description	http://lettere.unipv.it/obc/add/infap/archdes/isad(g)e.html
ISBD	International Standard Bibliographic Description	http://www.alice.it/library/law.lib/isbd.htm
ISO 12207	This standard describes the major component processes of a complete software life cycle, their interfaces with one another, and the high-level relations that govern their interactions	http://www.software.org/quagmire/descriptions/iso-iec-12207.asp
ISO 23950	Information retrieval protocol standard (=ANSI/NISO Z39.50)	http://www.loc.gov/z3950/agency/ http://www.niso.org/z3950.html
ISO 2907	Defines a record structure (= ANSI/NISO Z39.2)	http://www.loc.gov/marc/concise/concise.html
ISO/IEC JTC 1/SC 29	Sub Committee of the Joint Technical Committee 1 of the ISO/IEC dealing with coding of Audio, Picture, and Multimedia and Hypermedia Information	http://www.jtc1.org/

J2EE	JAVA 2 Platform, Enterprise Edition	http://java.sun.com/j2ee/
J2ME	JAVA 2 Platform, Micro Edition	http://java.sun.com/j2me/
J2SE	JAVA 2 Standard Edition	http://java.sun.com/j2se/
JAVA 2D	JAVA 2D API	http://java.sun.com/products/java-media/2D/
JAVA 3D	JAVA 3D API	http://java.sun.com/products/java-media/3D/
JAVA Applet	JAVA Applets	http://java.sun.com/applets/?frontpage-spotlight
JAVA Plugin	JAVA Plugin	http://java.sun.com/products/plugin/
JAVA Servlet	JAVA Servlet Technology	http://java.sun.com/products/servlet/
JAVA WebStart	JAVA Application Deployment Technology	http://java.sun.com/products/javawebstart/
JBOSS	Open Source Enterprise JavaBeans Application Server	http://www.jboss.org/
JDBC	JAVA DataBase Access API	http://java.sun.com/products/jdbc/
JetSpeed	Open source Implementation of an Enterprise Information Portal using JAVA and XML	http://jakarta.apache.org/jetspeed/site/index.html
Jext	A JAVA Text Editor	http://www.jext.org/
JMS	JAVA Messaging Service	
JNDI	JAVA Naming and Directory Interface	
JRMP	JAVA Remote Method Procedure call	
JSP	JAVA Server Pages	
JTA	JAVA Transaction API	
JTS	JAVA Transaction Service	
JVM	JAVA Virtual Machine	
JZKit	JAVA toolkit for building distributed information retrieval systems	http://www.k-int.com/jzkit/
k42	Knowledge Server	http://k42.empolis.co.uk/home.html
KB	Knowledge Base Access: Part of the REGNET System	
LATCH	Location, Alphabet, Time, Category, Hierarchy	
LDAP	Lightweight Directory Access Protocol	
MALVINE	Manuscripts and Letters via Integrated Networks in Europe	http://www.malvine.org/
MARC	The MARC formats are standards for the representation and communication of	http://www.loc.gov/marc/

	bibliographic and related information in machine-readable form.	
MARC 21	A MARC subformat	http://lcweb.loc.gov/marc/umb/um07to10.html
MDF	Metadata Processing Framework	http://www.techquila.com/mdf.html
Mercator	E- Business Integration Broker	http://www.mercator.com/products/index.html
MHEG-1	MHEG object representation-base	http://www.km.giti.waseda.ac.jp/WG12/
MHEG-5	Support for base-level interactive applications	http://www.km.giti.waseda.ac.jp/WG12/
MHEG-6	Support for enhanced interactive applications	http://www.km.giti.waseda.ac.jp/WG12/
MHEG-7	Interoperability and conformance testing	http://www.km.giti.waseda.ac.jp/WG12/
MHEG-8	Standard to provide XML encodings for ISO/IEC 13522-5 (MHEG-5)	http://www.km.giti.waseda.ac.jp/WG12/
Microsoft SQL Server	Fully Web-enabled database product, providing core support for Extensible Markup Language (XML) and the ability to query across the Internet and beyond the firewall.	http://www.microsoft.com/sql/default.asp
MIDlet	A MIDlet is a JAVA application that conforms to the specifications set out by the Connected, Limited Device Configuration (CLDC) and the Mobile Information Device Profile (MIDP).	http://developer.java.sun.com/developer/products/wireless/midp/articles/intro/
MOM	Message Oriented Middleware	
MOSAIC	Museums Over States And vIrtual Culture	http://mosaic.infobyte.it/
MPEG- 7	Will be a standardized description of various types of multimedia information	http://www.darmstadt.gmd.de/mobile/MPEG7/Documents/ w4006.htm
MPEG-4	The MPEG-4 format is meant to become the universal language between broadcasting, movie and multimedia applications.	http://www.iis.fhg.de/amm/techinf/mpeg4/index.html
MySql	Open Source Database	http://www.mysql.com/
NCSU	North Carolina State University	http://www.csc.ncsu.edu/
NEDLIB	NEDLIB is a collaborative project of European national libraries It aims to construct the basic infrastructure upon which a networked European deposit library can be built.	http://www.kb.nl/coop/nedlib/
Netfish	see IONA/NETFISH	
NWI	Nordic Web Index	http://nwi.dtv.dk/nwi_info.html
OASIS	Organization for the Advancement of Structured Information Standards	http://www.oasis-open.org/

OBI	Open Buying on the Internet	http://www.openbuy.org/
OCR	Optical Character Recognition	
ODBC	Open Database Connectivity	
OMG	Object Management Group	http://www.omg.org/
ONTOLOGY	An ontology is a specification of a conceptualization.	http://www-ksl.stanford.edu/kst/what-is-an-ontology.html
Ontopia Navigator	Tool for navigating Topicmaps	http://www.ontopia.net/
OODBMS	Object Oriented Database Management Systems	
OPAC	Online Public Access Catalog	
OpenEDI	B2B Transaction Engine	http://www.pentagon2000.com/p_edi.html
ORB	Object Request Broker	
OTM	Object Transactional Monitor	
PCM	Product Catalogue Management: Part of the REGNET System	
PCM	Pulse Code Modulation	
PD	Procurement and Delivery: Part of the REGNET System	
PDA	Personal Digital Assistant	
PDF	Portable Document Format	
PHP/YAZ	extension to PHP that implements Z39.50 origin (client) functionality.	http://www.indexdata.dk/phpyaz/
PIP	Partner Interface Process. Defined by RosettaNet	
PSTN	Public Switched Telephone Network	
PT	Portal: Part of the REGNET System	
QT	Query Translator (see SOAP)	
Rational Unified Process	Web-enabled software engineering process that enhances team productivity and delivers software best practices to all team members.	http://www.rational.com/products/rup/index.jsp
RDBMS	Relational Database Management System	
RDF	Resource Description Framework	http://www.w3.org/RDF/
REGNET	Cultural Heritage in REGional NETworks	http://www.REGNET.org/
RM	Repository Management: Part of the REGNET System	
RMI	Remote Method Invocation	http://java.sun.com/products/jdk/rmi/
RNIF	RosettaNet's Implementation Framework	http://www.rosettanet.org
RPC	Remote Procedure Call: a call to a routine that results in code being	

	executed on a different system from the one where the request originated. An RPC system allows calling procedures and called procedures to execute on different systems without the programmer needing to explicitly code for this.	
RS	Reference System: A subsystem of the REGNET System.	
S&R	Search and Retrieval: A subsystem of the REGNET System.	
SAX	SAX is a standard interface for event-based XML parsing, developed collaboratively by the members of the XML-DEV mailing list and OASIS. Currently in version 2.0.	
SCHEMAS	EU project that provides a forum for metadata schema designers involved in projects under the IST Programme and national initiatives in Europe.	http://www.schemas-forum.org
Schmunzel	Interactive SMIL 1.0 compliant player developed by Salzburg Research Forschungsgesellschaft m.b.H., member of the REGNET consortium	http://www.salzburgresearch.at/
SET	Secure Electronic Transactions is a system for ensuring the security of financial transactions on the Internet developed by VISA and Master Card.	
SGML	Standard Generalised Markup Language (ISO 8879). A generic markup language for representing documents. SGML is a system for defining structured document types, and markup languages to represent instances of those document types.	
Shiloh	Product extension for the MS SQL-Server for the mapping of XML structure to RDBMS	
SHOE	Simple HTML Ontology Extensions which allows web page authors to annotate their web documents with machine-readable knowledge.	http://www.cs.umd.edu/projects/plus/SHOE/
Simple-EDI	Object oriented extension for EDI	
SM	Semiconductor Manufactoring	
SME	Small and Medium sized Enterprises.	
SMIL	Synchronised Multimedia Integration Language is a specification from the w3c allowing the integration of a set of independent multimedia objects into a synchronised multimedia	http://www.w3.org/TR/REC-smil/

	presentation, currently in version 1.0	
SMPTE	Society of Motion Picture and Television Engineers	http://www.smpte.org/
SMTP	Simple Mail Transfer Protocol according to RFC 821	
SOAP	Simple Object Access Protocol providing a simple and lightweight mechanism for exchanging structured and typed information between peers in a decentralised, distributed environment using XML proposed under the w3c.	http://www.w3.org/TR/SOAP/
SOAP	Simple Object Access Protocol	
SOAP4J	A JAVA implementation supporting SOAP, developed under the APACHE project.	http://xml.apache.org/soap/index.html
SOJA	SMIL compliant player from HELIO.	http://www.helio.org/products
SQL	Structured Query Language: ISO, ANSI standard user front end to a relational database management system.	
SR/Z39.50	see Z39.50	
SSL	Secure Sockets Layer is a transaction security standard developed by Netscape Communications to enable commercial transactions to take place over the otherwise notoriously non-secure Internet.	
SSL	Secure Socket Layer	
SUTRS	Simple Unstructured Text Record Syntax is part of the Z39.50 standard.	
SWIFT	Society for Worldwide Interbank Financial Telecommunication founded in 1973 to support world-wide financial transactions	http://www.swift.com/
Swing	Software library for building user interfaces, based on AWT.	http://java.sun.com/products/jfc/
Tamino	A XML database product of the Software AG	http://www.softwareag.com/tamino/
Tcl/Tk	Tool Command Language (Tcl) whitch is associated user interface toolkit (Tk) for quickly creating cross-platform applications with graphical user interfaces, developed by John Ousterhout.	http://www.pconline.com/~erc/tcl.htm
Textml	XML database product from ixiasoft.	http://www.ixiasoft.com/
TGN	Getty Thesaurus of Geographic	http://www.getty.edu/research/tools/voca

	Names.	bulary/tgn/
TM4J	The topic map engine for JAVA is a product of Tequila	http://www.techquila.com/tm4j.html
Tmproc	Is an open source Python topic map engine.	http://www.ontopia.net/software/tmproc/
Tomcat	Servlet Engine developed by the Apache-Jakarta project.	http://jakarta.apache.org/tomcat/
TR&P	Transport Routing and Packaging	
UC	Use Case	
UDDI	Universal Description, Discovery and Integration is an industry initiative for creating a platform-independent, open framework for describing services, discovering businesses, and integrating business services using the Internet.	http://www.uddi.org/
UID	Unique Identifier	
ULAN	Union List of Artist Names	
UML	Unified Modelling Language from Rational encompassing the object-oriented analysis and design methodologies of Booch, Rumbaugh, and Jacobson.	
UMM	UN/EDIFACT Modelling Methodology	
UMTS	Universal Mobile Telecommunications System	http://www.umts-forum.org/
UN/EDIFACT	United Nations Rules For Electronic Data Interchange for Administration, Commerce and Transport	http://www.disa.org/international/edif.htm
UNICODE	Character encoding standard which is part of the ISO-10646.	
URI	Uniform Resource Identifier is a generic set of all names/addresses that are short strings that refer to resources. Defined in RFC 2396.	http://www.ietf.org/rfc/rfc2396.txt
URL	Uniform Resource Locator	
VoiceXML	Voice eXtensible Markup Language is designed for creating audio dialogs that feature synthesized speech, digitized audio, recognition of spoken and DTMF key input, recording of spoken input, telephony, and mixed-initiative conversations. Submitted to the w3c.	http://www.w3.org/TR/voicexml/
WAE	WAP Application Environment	
WAIS profile	Wide Area Information Servers profile based on ANSI/NISO Z39.50 Version 2.	http://ftp.std.com/obi/Standards/TEXT/WAIS-Profile-of-Z3950-V2.html

WAP	Wireless Application Protocol	http://www.wapforum.org/
WML	Wireless Markup Language	http://www.wapforum.org/what/technical.htm
WSDL	Web Services Description Language is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information.	http://www.w3.org/TR/wsdl
WSP	Wireless Session Protocol	
WTLS	Wireless Transport Layer Security Protocol	
WTSL	Wireless Transport Security Layer	
WYSIWYG	What You See Is What You Get	
X12	EDI standard	
XALAN	XSLT processor of the apache project	http://xml.apache.org/xalan-j/index.html
XER	XML Encoding Rules	http://asf.gils.net/xer/standard.html
XERCES	XML parser of the apache project	http://xml.apache.org/xerces-j/index.html
XM	eXperimentation Model (MPEG- 7)	
XMI	XML Metadata Interchange defined by the OMG for file export/import of models, and a transfer format specification for unique identification of the version of the MOF meta-metamodel and any metamodels referenced but not included in an SMIF-compliant transfer.	
XML	EXtensible Markup Language	
XML/EDI	EDI encoding using XML	
XPath	XML Path Language is a language for addressing parts of an XML document, designed to be used by both XSLT and XPointer.	http://www.w3.org/TR/xpath
XSL	Extensible Stylesheet Language. An XSL stylesheet specifies the presentation of a class of XML documents by describing how an instance of the class is transformed into an XML document that uses the formatting vocabulary.	http://www.w3.org/TR/xsl/
XSLT	XSL Transformations is a language for transforming XML documents into other XML documents.	http://www.w3.org/TR/xslt
XSU	XML Schema Upgrade service	http://www.w3.org/2001/03/webdata/xsu

XSV	XML Schema Version checking service	http://www.w3.org/2000/09/webdata/xsv
XTM	XML Topic Map	http://www.topicmaps.org/xtm/1.0/
Z39.2	American National Standard for Information Interchange (= ISO 2907)	http://www-it.hr.doe.gov/standards/stdsdesc.cfm?Id=11
Z39.50	Information retrieval protocol standard (= ISO 23950)	http://www.niso.org/z3950.html http://www.loc.gov/z3950/agency/
ZAP!	A Z39.50 apache module	http://www.indexdata.dk/zap/

8 Bibliography and References

- Bluetooth: The Bluetooth Specification v. 1.1 core, Bluetooth, 2000.
- ebXML: ebXML Technical Architecture Specification v1.0.4, ebXML, 2001. http://www.ebxml.org/specs/ebTA.pdf
- ebXML: Business Process Specification Schema v1.01, ebXML, 2001.
- Gamma, Erich et.al.: Design Patterns - Elements of Reusable Object Oriented Software, Addison Wesley, 1995.
- Grady Booch, Jim Rumbaugh, and Ivar Jacobson: Unified Modeling Language—Reference Manual, Addison-Wesley, 1999.
- International Standardisation Organisation: ISO 23950 Information and documentation -- Information retrieval (Z39.50) -- Application service definition and protocol specification, 1998.
- International Standardisation Organisation: ISO/IEC 13250 Topic Maps Information Technology
- Document Description and Processing Languages, International Standardisation Organisation, 1999.
- Ivar Jacobson, Grady Booch, and Jim Rumbaugh, Unified Software Development Process, Addison-Wesley, 1999.
- MPEG Comittee: ISO/IEC WD 15938-1 MPEG-7 Systems Specification, MPEG Comittee, 2001.
- Sun Microsystems: Enterprise JavaBeans(TM) 2.0 Specification Final Release, Sun Microsystems, 2000.
- Sun Microsystems: JAVA(TM) 2 Platform, Enterprise Edition Final Release 1.3, Sun Microsystems, 1999.
- Sun Microsystems: Mobile Information Device Profile (MIDP) Specification 1.0, Sun Microsystems, 2000.
- Sun Microsystems: J2ME(TM) Connected Limited Device Configuration Specification 1.0a, Sun Microsystems, 2000.
- Susanne Boll, Wolfgang Klas, Utz Westermann: Multimedia Document Models – Sealed Fate or Setting Out for New Shores?, Databases and Information Systems (DBIS), 1999.
- Thomas, Anne: Enterprise JavaBeans Technology - Server Component Model for JAVA Platform, Patricia Seybold Group, 1998. http://java.sun.com/products/ejb/pdf/white_paper.pdf
- Topicmaps Consortium: XML Topic Maps (XTM) 1.0, Topicmaps Consortium, 2001.
- UDDI: UDDI Version 2.0 Data Structure Reference, UDDI, 2001. http://www.uddi.org/pubs/DataStructure-V2.00-Open-20010608.pdf

List of Figures

Figure 1: Simple Search GUI 12
Figure 2: Advanced Search GUI........ 13
Figure 3: Scheme of the interconnections between the parts of the PCM. 17
Figure 4: Prototype Architecture (Example: Procurement)........ 25
Figure 5: Architecture of the Topic Map Generator Tool 29
Figure 6: XTMLoader class........ 30
Figure 7: Main page of Topic Map Generator Tool........ 31
Figure 8: Component Diagram – Ontology Node 33
Figure 9: Search and Retrieval Architecture........ 34
Figure 10: Components of the CatXML interchange within the PCM 36
Figure 11: Components of the PCM – CatXML Catalogue Definitions 36
Figure 12: Portal Architecture........ 38
Figure 13: Activity Diagram - Publication........ 41
Figure 14: Electronic Publishing Architecture 42
Figure 15: Data Transformations 45
Figure 16: Activity Diagram – Insert / Edit Record........ 46
Figure 17: Activity Diagram – Delete Record 47
Figure 18: Package Diagram – Data Generation 47
Figure 19: Class Diagram – Data Generation 48
Figure 20: Activity Diagram – Search & Retrieval 54
Figure 21: Package Diagram – Search & Retrieval 55
Figure 22: Class Diagram – Search & Retrieval........ 56
Figure 23: Scheme of components and interconnections between them. 57
Figure 24: Use Case Diagram – e-Business........ 60
Figure 25: Activity Diagram – e-Business search functionality 61
Figure 26: Activity Diagram – e-Business Order 62
Figure 27: Activity Diagram - Update Catalogues 63
Figure 28: Activity Diagram – Insert Information 64
Figure 29: Activity Diagram - Delete Information........ 64
Figure 30: Activity Diagram – Edit Record........ 65
Figure 31: Activity Diagram – Save Record 66
Figure 32: Activity Diagram – Delete Record 66
Figure 33: Activity Diagramm – Search........ 67
Figure 34: Package Diagram – Reference System 67
Figure 35: Class Diagram – Reference System........ 69
Figure 36: Product Catalogue Management model of the activities and the components........ 74
Figure 37: Activity Diagram - Search and retrieval CH data........ 75

Figure 38: Activity Diagram - Search and retrieval commercial data....76
Figure 39: Activity Diagram - Update and insert data....77
Figure 40: Activity Diagram – Collecting and sending statistics data....78
Figure 41: Package Diagram - Procurement....79
Figure 42: Class Diagram - Registration....82
Figure 43: Sequence Diagram - Supplier's registration....83
Figure 44: Sequence Diagram - Supplier's acceptation....84
Figure 45: Class Diagram - Catalogue....85
Figure 46 - sequence diagram - new supplier's product....86
Figure 47: Sequence Diagram – Navigation by department and category....87
Figure 48: Sequence Diagram – Navigation by supplier....88
Figure 49: Sequence Diagram - Search a product....88
Figure 50: Sequence Diagram - List of departments....88
Figure 51: Sequence Diagram - List of categories of a department....88
Figure 52: Sequence Diagram - List of characteristics of a category....89
Figure 53: Sequence Diagram -List of products of a category....89
Figure 54: Sequence Diagram - List of showcases....89
Figure 55: Sequence Diagram - List of products of a showcase....89
Figure 56: Sequence Diagram - List of products....90
Figure 57: Sequence Diagram - List of characteristics....90
Figure 58: Class Diagram - Trade....91
Figure 59: Sequence Diagram - place an item in a cart....91
Figure 60: Sequence Diagram - send the trade....92
Figure 61: Class Diagram - Reports....93
Figure 62: Package Diagram - Delivery....95
Figure 63: Class Diagram : Shipping cost calculator....97
Figure 64: Sequence Diagram -Calculation of shipping cost....98
Figure 65: Class Diagram - Shipment....99
Figure 66: Sequence Diagram - Delivery's Registration....100
Figure 67: Sequence Diagram - Shipment's assignation....102
Figure 68: Sequence Diagram - Package's registration....103
Figure 69: Sequence Diagram - commodity's packing....104
Figure 70: Sequence Diagram - package's weighing....104
Figure 71: Sequence Diagram - delivery's sending....105
Figure 72: FilePacket class UML diagram....106
Figure 73: FileReceiver class UML diagram....107
Figure 74: FileSender class UML diagram....107
Figure 75: FileSendInitialize class UML diagram....108
Figure 76: ReceiveFile class UML diagram....109

Figure 77: RemoteFileSendInitialize class UML diagram. 109
Figure 78: RemoteReceiver class UML diagram. 110
Figure 79: Activity Diagram - Publisher 111
Figure 80: Class Diagram – Electronic Publisher 114
Figure 81: Class Diagram – Electronic Publisher 115
Figure 82: The REGNET-Site 116
Figure 83: Deployment Schedule 119
Figure 84: Example Deployment for Prototype 120

List of Tables

Table 1: e-Business Main Use Cases 59
Table 2: Interface of PCM & RS 71
Table 3: Interface of PCM & KB 72
Table 4: Interface of PCM & DG 72
Table 5: Interface of PCM & SR 73
Table 6: Interface of PCM & EB 73
Table 7: Procurement 81
Table 8: Delivery 96
Table 9: Use Cases – Electronic Publisher 112
Table 10: Deployment responsibilities 119