D.3.2 ï Second Year Report

WP3 ï Repository Infrastructure

Version5.0 - FINAL

29 November 2010

Grant Agreement number:	**231809**
Project acronym:	**3D-COFORM**
Project title:	**Tools and Expertise for 3D Collection Formation**
Funding Scheme:	**FP7**
Project co-ordinator name, Title and Organisation:	**Prof David Arnold, University of Brighton**
Tel:	**+44 1273 642400**
Fax:	**+44 1273 642160**
E-mail:	**D.Arnold@brighton.ac.uk**
Project website address:	**www.3d-coform.eu**

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 231809.

Authors: Martin Doerr, Foundation for Research and Technology, Hellas (FORTH-ICS)

Contributing partner organizations:

University of Brighton (UoB)

PIN scrl - Servizi didattici e scientifici per l'Università di Firenze (PIN)

Katholieke Universiteit Leuven (KUL)

Eidgenossiche Technische Hochschule Zurich (ETHZ)

Consiglio Nazionale Delle Ricerche ςISTI (CNR-ISTI)

Technische Universitaet Graz (TU Graz)

Centre de Recherche et Restauration des Musees de France (CNRS-LC2RMF)

University of Glasgow (GU-HATII)

University of Bonn (UBonn)

Media Integration and Communication Centre, University of Florence (MICC)

Victoria and Albert Museum (VAM)

Breuckmann

Spheron

Table of Contents

1 Executive Summary

This document presents the work that has been done in the context of WP3 ςRepository Infrastructure during the second year of activity of the 3D-COFORM project.

WP3 plays a crucial role in the project since it is responsible for the design and the implementation of the 3D-COFORM integrated repository. The 3D-COFORM integrated repository is a distributed cultural Repository Infrastructure (RI) that stores multi-media objects and metadata and is itself composed of different components. The integrated repository system plays a central role to the 3D-COFORM workflow since it is the main integration actor of most of the 3D-COFORM technical outcomes (which interact with the repository as either data producers, data consumers or both).

A well defined Application Programming Interface (API) to the repository functionality allows external tools to access, enhance and/or use the knowledge and the information contained in the repository.

During the first year, the goal of WP3 was to produce the design and the functional specifications of the RI and its components, while in the second year the goal was to implement the specifications and produce a prototype working environment.

In principle the activities of WP3 are on time ςwith no major deviation from the work plan drafted in the project Description of Work (DoW). The activities progressed well, following the work plan and the milestones were accomplished with some deviations in time. All planned partners are contributing to WP3. The Year 2 milestone for WP3, Milestone *MS.3.3 Individual implementations of all components T3.3-T3.8* has been achieved and further work carried out with preliminary integration activities.

The deviations in the release of the RI-beta and RI V1.0 version, were due to the change of implementation platform (from Globus Toolkit to AFS), and the major re-engineering undertaken in the Object Repository (OR). In Month 21 a beta version (basic Java-API) was released for testing (with three months of delay) and has proved the feasibility of the RI (and the basic functionality of its main components OR and Metadata Repository (MR)). Since Month 21 there was a major re-engineering in order to transfer the whole system from a Java application to a Java web application, with a released (in Month 25 with one month of delay) RI Web-service with single-location functionality, and we have planned the full functionality for Month 30, thus meeting the original DoW deadline.

Major activities performed and results obtained during the second year are:

¶ RI alpha versions	delivered Month 14 and 15 (provided for the first Periodic Review)
¶ RI beta version	delivered Month 21
¶ RI version 1.0	delivered Month 24
¶ LTDR tool	delivered Month 24
¶ Watermarking standalone tool	delivered Month 24

The activities will continue in Year 3 according to the plan described in the project contract.

The overall organization of the document is as follows. Section 2 gives a brief presentation of the objectives of WP3 for the reporting period. Sections 3 to 11 present in detail the work done in each of the nine Tasks of WP3 during Year 2 and the results obtained. Section 12 reports on the publications produced so far. Finally, the Repository Infrastructure Architecture is presented in detail in Appendix A, the Fundamental categories and relations in Appendix B and CRMdig, the extension of CIDOC-CRM in Appendix C.

2 Introduction and Objectives

In this report, we present the achievements made in WP3 during the second year of 3D-COFORM. These mainly consist of implementing versions beta and 1.0 of the Repository Infrastructure (RI).

We present the work that was planned for the second year of the 3D-COFORM project and the work that was performed during this second year. The presentation is done per task, following the structure of the original Description of Work (DoW). For each task, a section is devoted to the analysis of the deviations between the work performed and the work planned, and an additional section presents the plans for the third year of the project.

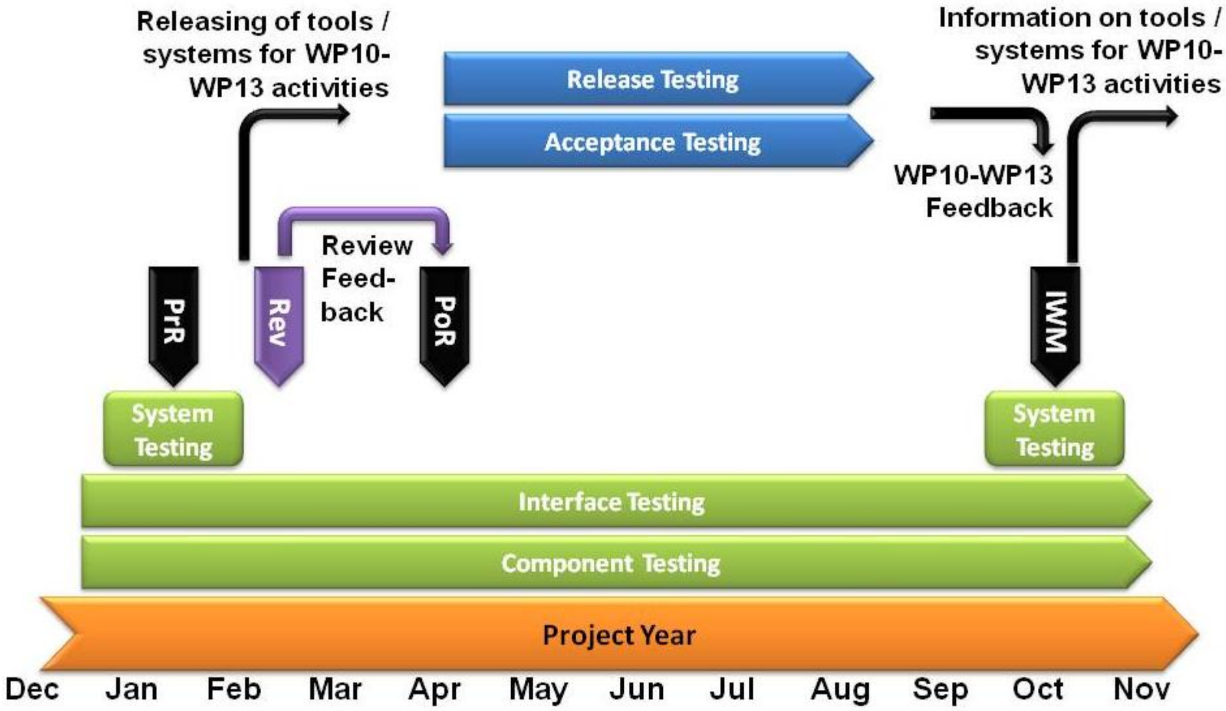
It must be emphasized that the work in WP3 is a collaborative effort of all the involved partners.

3 Task 3.1 Scientific and Technical Coordination

The objective of this task is to coordinate and monitor the progress in the development of tools, including identification of deviations and amendments to the work plan. It also ensures the interrelation between Work Packages.

3.1 Work planned

The work planned for Year 2 for Task 3.1 includes the continuous monitoring and controlling of the technical development of the project. Task 3.1 will constantly measure the progress, in order to recognize deviations and take proper actions. It will continue providing guidelines and coordinating with the structure of the project, in order to guarantee and integrate development. It will also continue with the coordination between the business and technical activities, in order to achieve the objectives of the project. In order to achieve integration and synchronization between the different aspects of the project, a meeting schedule was defined, allowing the partners to be informed about the needs and the progress of the different activities of the project (see following picture).



The Pre-Review meeting (PrR) aims to test the integration of the tool chains, which will be demonstrated and released at the review for user testing and training activities. Between the PrR and the Review (Rev), the technical partners can develop the needed documentation for supporting the user testing and training events. The objective of the Post-Review meeting (PoR) is to synchronize the activities of the project and to evaluate the review report, in order to develop a strategy for implementing the corrective actions. Additionally, it allows for communicating the development plans for the upcoming 6 months. During the Integration Week Meeting (IWM) the current status of the implementation is reported and the possible tool chains for the following year are developed. Between the IWM and the PrR the technical partners are able to further integrate and test the interfaces of the tool chains for different workflows.

Developer testing (including component testing, interface testing and system testing) is performed within the technical WPs and a coordinated evaluation with users is not required, but it could be performed, if it is needed. The technical partners are responsible for testing individual components, as well as the integrating tools (e.g. the Repository Infrastructure - RI - or the Integrated Viewer/Browser - IVB), which require a reliable testing of the interfaces and corresponding communication between the different components. Finally, system testing can be performed with different workflows and for different data, which is mostly provided by Cultural Heritage (CH) institutions.

The overall objective for the technical activities in the second period of the project was to consolidate the development of the integrating tools (e.g. RI and IVB) and the individual tools, including user testing for existing tools. In order to achieve this objective, the following activities were planned, coordinated and executed:

¶ The Pre-Review Meeting (PrR) was held in Darmstadt, Germany from January 18th to January 21st, 2010

¶ The Post-Review Meeting (PoR) was held in Heraklion, Crete, Greece from May 31st to June 4th 2010

¶ The Integration Week Meeting (IWM) was held in Darmstadt, Germany from October 25th to October 28th, 2010

Additionally, Technical Strand reviews were also scheduled as Skype conferences, in order to monitor the progress of the different technical areas of the project and to coordinate needed actions:

¶ General Technical Strand progress review was performed on April 21st, 2010.

¶ 3D capture and processing Technical Strand review was performed on October 4th, 2010

¶ Generative modelling and visualisation Technical Strand review was performed on October 5th, 2010

¶ Integration of metadata and related textual information processing Technical Strand review was performed on October 7th, 2010.

3.2 Work performed

In the context of this task there was a continuous coordination and monitoring of the activities of the technical work packages. This was achieved by a close communication and cooperation of the involved partners through the organization of bilateral and/or general WP meetings, Skype conferences and exchange of e-mails. The organizational structure of the project was used, in order to distribute information, therefore Strand leaders and WP leaders were timely informed and they proceeded to forward the information to the individual partners.

In order to ease the monitoring of the progress of the technical development of the project, the development plans for the components were requested, in order to complement the already provided information regarding the functional specifications of the components. Therefore, the collection and analysis of the development plans of the components has been executed and as a result a consolidated plan was generated, with inconsistencies identified where possible. This information was transmitted to the partners, in order to implement corrective actions. In the context of this task, a set of guidelines was designed for the preparation of the Post Review (PoR) and Integration Week (IWM) meetings and the coordination and monitoring of their preparation was also performed.

The overall objective of this task was successfully achieved, the development of the integrating tools was consolidated (some of them with individual user testing), as well as the development of individual tools and the user testing for the existing tools and the tool chains presented in the first review of the project. Additionally, a mapping between components and tools was developed, thus the monitoring of the development and the schedule of the tools for testing and training purposes are more transparent.

3.3 Deviation from work plan

No deviation from the work plan. The second year work plan has been fulfilled according to the DoW.

3.4 Plans for the next period

The coordination and monitoring of the project technical activities will continue through the whole duration of the project.

This task will continue to support partners toward the objectives of the project. In order to achieve this, guidelines, points for synchronization and clear goals will be defined and provided to the technical partners. The progress of the project will be monitored and it will compare with the needs of the project, in order to adapt and achieve integration at different levels (from a technical and business point of view). In addition, a tied cooperation with the business part of the project will be promoted, in order to test and train CH professionals in the use of the tools developed within the project, aiming to collect feedback, which can be integrated within the development for improving the tools in view of usability, robustness, maturity, benchmarking, interoperability and scalability. This cooperation will be in line with the demonstrations for the end of the project.

The overall objective for the third period of the project is to mature the development of the tools and to increase the interaction with the CH users, by means of testing and training activities.

4 Task 3.2 Design architecture

4.1 Work planned

The objective of this task was to integrate existing open source repository technology in order to implement the platform of the 3D-COFORM integrated repository. This was based on the results of Year 1 definition of the integrated repository which proposed the overall system architecture, its components, and the interfaces between these components.

This task would also extend generic metadata formats to describe and support capture and acquisition process metadata, process and workflow metadata and legacy, annotation and co-reference metadata.

4.2 Work performed

A lot of emphasis was given to the specifics of the Repository Infrastructure (RI) architecture. The RI is composed of several modules which connect and interact with one another (as illustrated in Figure 1). All communication function-signatures between RI components and between RI and external clients are Integrated Repository Architecture & Design Specifications while the Repository Infrastructure Architecture

During this year some of the decisions taken had to be revised and accordingly the design of the RI architecture was revised. In short, the use of GLOBUS Toolkit (GT) in the OR implementation was abandoned and was replaced by the use of AFS. The existence of a web-service on each OR-node was decided and a major re-engineering of the OR service implementation also took place.

The basic components of the RI (metadata and object repositories, query manager and RI-integration mechanism) were delivered to the partners in three deliverables.

The first deliverable (RI-API alpha stand-alone version) was issued at the end of Month 12, as a client side application programming interface (RI-ARI) in the form of Java-library module (jar). It provided limited functionality: ingest/retrieve/query functions while it was based on remote MR installation in conjunction with a local OR installation. That provided 3D-COFORM partners a working framework to develop their code and proceed with the implementation of the IVB prototype user interface.

The second deliverable (RI-API beta-version, internal version number 2.0) was issued at the end of Month 21, as a client side application programming interface (RI-ARI) in the form of Java-library module (jar). It provided full functionality (with only access control and replica-handling missing) while it was based on remote MR installation in conjunction with remote single OR installation. At least two revision deliverables, with bug fixes, were also produced from Month 21 to Month 24, based on bugs reported on RI-bug-tracker (application for receiving bug reports from RI users).

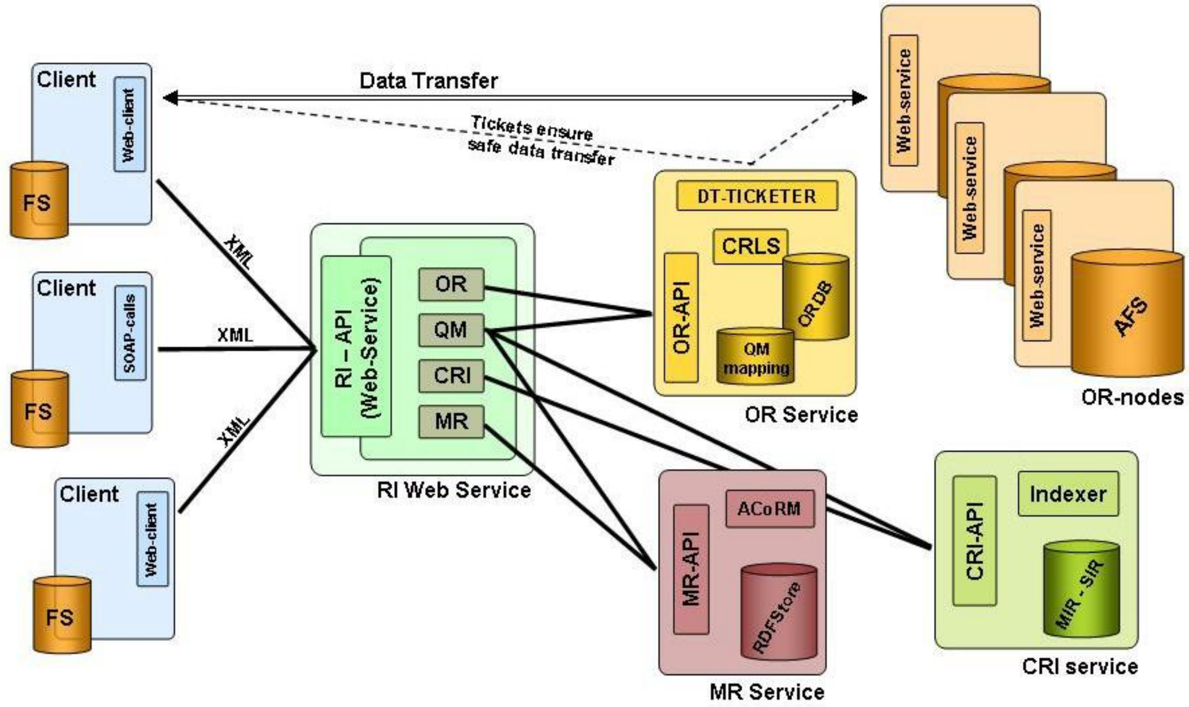


Figure1: Overview of the Repository Infrastructure architecture. The clients communicate with the repository through a central WEB-Service, via XML messages that follow the SOAP standard. This central service is passing the requests to the RI components (e.g. the MR Service, the OR Service that controls the OR-nodes and the CRI Service). Data transfer is performed directly between clients and OR-nodes and it is initiated by the OR Service.

The third deliverable (RI-API v1.0-version, internal version number v3.0) will be issued at the end of Month 24, as a web-service interface and as a Java stub (web-service wrapper). It will provide full functionality based on web-services and it will separate the data transfer mechanism from the rest of the RI interface, thus being more compliant to the revised design Ɩ· ă LJNJŜ ă Ŝ ƴ ű Ŝ Ř ƛ ƴ - Ř Ŝ ű Ɩ· ƛ f Repository Infrastructure Architecture έ Φ will bĿ űased on remote MR installation in conjunction with remote single OR installation. It will still be missing access control and replica-handling, which is envisioned to be provided in the beginning of Year 3.

4.3 Deviation from work plan

The RI-service will be completed in Month 25, with one month delay, due to the change of implementation platform (from Globus Toolkit to AFS), and the major re-engineering undertaken in OR. In Month 21 a beta version (basic Java-API) has been released for testing and has proved the feasibility of RI (and the basic functionality of its main components OR and MR). Since Month 21 there was a major re-engineering in order to transfer the whole system from a Java application to a Java web application, with a released (in Month 24) RI Web-service with single-location functionality. We have planned the full functionality for Month 30, meeting the original work plan.

4.4 Plans for the next period

For the third year we have planned the integration of the CRI existing algorithms into the CRI Web-service and the integration of this service to the RI.

Month 35 - Third Year Milestone

Á MR v1.1 Revised version: Reasoning, and application of reasoning on queries and consistency checking mechanism, full-text search. Documentation of administration, usage, and installation procedure. Full backup restore procedure.

Á OR v1.1 Revised version: Complete functionality for each ORnode-service, access-rights, replica-handling. Documentation of administration, usage, and installation procedure for OR-Service and ORnode-services. Optimization of performance. Full backup restore procedure.

Á QM v1.1 Revised version: Integration with CRI-services, Optimization of performance. Documentation of administration, usage, and installation procedure.

Month 36 - Third Year Deliverable

Á RI v1.1 Revised version: Integration of MR v1.1, OR v1.1, QM v1.1 and CRI-services.

5 Task 3.3 Design and implementation of a metadata repository

5.1 Work planned

The objective of this task for the second year was to provide the design and implementation of the beta and 1.0 versions of the metadata repository, based on the integrated coherent conceptual schema that has been defined during the first year. CRMdig, an extension to CIDOC-CRM, has been defined during the first year, while possible adaptations/revisions to this conceptual schema would be done, if needed, during the second year.

5.2 Work performed

The metadata repository is implemented using Sesame. Sesame is an open source Java framework for storing, querying and reasoning with RDF and RDF Schema. It can be used as a database for RDF and RDF Schema, or as a Java library for applications that need to work with RDF internally. Sesame provides the necessary tools to parse, interpret, query and store all this information, either embedded in an application, or in a separate database or even on a remote server.

During the reporting period, we received several examples of metadata produced by the partners, in order to examine them, correct/modify them and ingest them in the MR. This work helped in identifying some necessary modifications and revisions of the model. One of the problems that we encountered was how to use a set of assertions (triples) as target for an annotation. We decided to use the *Named Graphs*[2] mechanism, a well accepted, standard procedure to deal with multiple RDF graphs in a single document/repository and naming them with URIs. With such an approach, a named graph will be viewed as a *context*of Sesame but in a more standard way. It was also decided to use RDF/XML notation for writing the metadata files and Trig notation for writing annotation metadata files. The details of this approach and examples are presented in Deliverable 6.2 ςSecond Year Report on WP6 ςCreating the 3D Collection Item. A basic, core body of examples is now available in the MR for testing purposes.

In parallel with the on-going implementation and configuration of the MR, experiments were conducted. We investigated the use of Apache Lucene in conjunction with Sesame (the MR platform) in order to have full text search capabilities.

SwiftOWLIM reasoner was also tested and a fundamental reasoning mechanism was implemented to support transitive closure queries on specific properties (e.g. consist_of), and to support forward and backward equivalence of properties.

During the second year we have also defined a set of fundamental categories and relations, required by the IVB tools, which we will describe below. To make the queries upon our semantic web simpler and more generic, we have defined 5 fundamental categories:

- Thing
- Actor
- Place
- Time/Event
- Concept

In each category (except for the Concept), we have defined an extra category to provide the Type of it, so we also have:

- Thing Type
- Actor Type
- Place Type
- Time/Event Type

This distinction is absolutely necessary because it is very common to refer to a category not by an instance of it, but by its type.

The main relationships to be modeled by the proposed fundamental categories and relationships are:

- Identification of real world items by real world names
- Observation and Classification of real world items
- Part-decomposition and structural properties of Conceptual & Physical Objects, Periods, Actors, Places and Times
- t ŀ NJ ű ƛ Ŏ ƛ LJ ŀ ű ƛ ƻ ƴ ƻ Ŧ LJ Ŝ NJ ă ƛ ă ű Ŝ ƴ ű ƛ ű Ŝ Ƴ ă ƛ ƴ - ƒ ű ƛ Ŝ ƴ Ƴ Ŝ LJ ă ƻ NJ ŀ ƒ meeting in space-time
- Location of periods in space-time and physical objects in space
- Influence of objects on activities and products and vice-versa
- Reference of information objects to any real-world item

In the ά ! LJLJ Ŝ ƴ Ƙ ǻ Ƴ Ř ŀ Ƴ Ŝ ƴ ű ŀ ƒ Ŏ ŀ ű Ŝ, we describe these fundamental categories ƴ ă ε and the relationships between them in detail.

5.3 Deviation from work plan

No deviation from the work plan. The second year work plan has been fulfilled according to the DoW.

5.4 Plans for the next period

For the third year we plan to transfer our existing SwiftOWLIM reasoner implementation to BigOWLIM reasoner. In SwiftOWLIM reasoning and query evaluation are performed in-memory. Still, a reliable persistence strategy assures data preservation, consistency, and integrity. While update, reasoning and query evaluation proceed extremely fast even against huge ontologies and knowledge bases, a principle limitation of OWLIM is the relatively slow delete operation.

In contrast to SwiftOWLIM, BigOWLIM performs reasoning and query evaluation directly against the permanent image of the repository. The persistence is implemented through binary files with proprietary format. BigOWLIM is relatively slow on delete ς a limitation typical for the OLAP databases. The upload, the inference, and the query evaluation proceed extremely fast, even against huge ontologies and knowledge bases. The version of TRREE, used in BigOWLIM, features query optimization, which assures optimal evaluation disregarding the syntactic variations of the query. Another important feature of BigOWLIM is the special handling of equality that allows for unmatched efficiency in case of Ř ŀ ű ŀ ă Ŝ ű ă ƛ ƴ Ŏ ƒ dz Ř ƛ ƴ ∃ ƒ ŀ NJ ∃ Ŝ ά Ŝ ǉ dz ƛ Ø ŀ ƒ Ŝ ƴ Ŏ Ŝ owl:sameAs ă ă Ŝ ă έ Σ According to the publicly available results, BigOWLIM is the most scalable RDF repository with OWL reasoning support.

A second advantage of BigOWLIM is that it can easily integrate with Lucene. Lucene is a high-performance, full-featured text search engine library written entirely in Java. Beginning with BigOWLIM version 3.2, full text search capabilities using the Lucene engine are supported. The preliminary version of Lucene integration supports indexing and query evaluation over the entire repository, i.e. all nodes including both URI local names and literals.

For the third year the implementation of an ingestion tool is planned. This tool will support the ingestion of the data of acquisition and processing procedures. The tool will be based on the specifications that will arise from the manual/semiautomatic ingestion of the first set of data that partners will provide and on the scenario and workflow proposed by WP4 ς 3D Artefact Acquisition.

6 Task 3.4 Design and implementation of an object repository

6.1 Work planned

The objective of this task for the second year was to implement a prototype of object repository (OR). This prototype should have these significant functionalities:

1. ! ŏŜƴűNJŀ ſ Řŀűŀōŀ ăŜ attributes (e.g. File name, size, file ID, etc..). Řŀűŀ ăŜűΩă
2. The data transfer between a repository user and repository server should be free of barriers, which means the local physical dataset could be transferred to the OR-node safely and without consideration of other network pre-conditions.
3. OR module should be embedded in Repository Infrastructure (RI) and work correctly.
4. Trying to deploy multiple OR-node ă ŀű Řƛ ŦŦŜNJŜƴű LJŀNJű ƴŜNJΩă ăŜNJǾŜN distributed environment.

6.2 Work performed

At the beginning of the second year, it has been decided to abandon use of GLOBUS Toolkit (GT). The reasons could be summarized as follows:

1. Lack of flexibility and a large amount of useless functional modules of GLOBUS Toolkit. GT has defined many functional modules and the user could only use the functionality with stationary format and customizing the functionality is very difficult.
2. Java has been supported by most fundamental functionalities of GT version 4. However, from GT version 5, Java will not be supported for most functionality.
3. The installation and usage of GT is not humanized enough, it is difficult for the non-expert to operate.
4. The sources and documentation are very limited, there is no mature community or stable user group to consult or get timely support.

According to the above reasons, we made a new decision, using other tools: Andrew File System (AFS) and Java Webservice.

The AFS is a distributed file system with user management, access control lists, Kerberos-5-based user- and service-authentication, efficient caching to reduce bandwidth usage and, as an option, encrypted network traffic. AFS is conceptually mature and reliable in practice. The OpenAFS implementation is available under free IBM Public License, clients exist for Linux, MacOS X, Windows, BSD and Solaris.

The functionalities of OR would be encapsulated into a central OR Web-service (hereafter OR-service, presented in Figure 2 by the yellow rounded-rectangle). The architecture of the Repository

Lყ ŦNJŀ ă űNJdzŎ űdzNJŜ ƛ ă LJNJŜ ă Repository Infrastructure Architecture ɛƛФყ ¢άK!ŜLJLJŀŜwყ Ř service provides the fundamental functions of OR internal database operation and it controls also the trigger of data transfer between client and OR-node. The communication between different OR-nodes and central OR-Service are going to be realized via SOAP, and in this case, Apache Tomcat Servlet Server ǿŀ ă ƛYLJ≥NJűŜŘ ŀ ă űKŜ ăŜNJǾƛŎŜ ă OR-node. Since it is decided to ăŜNJǾŜ ă use Java to implement the whole RI, so, the Tomcat Server might be the most suitable container for our web application, because Java Servlet could be embedded into Tomcat server seamlessly.

The whole OR working cycle can be demonstrated as in Figure 2:

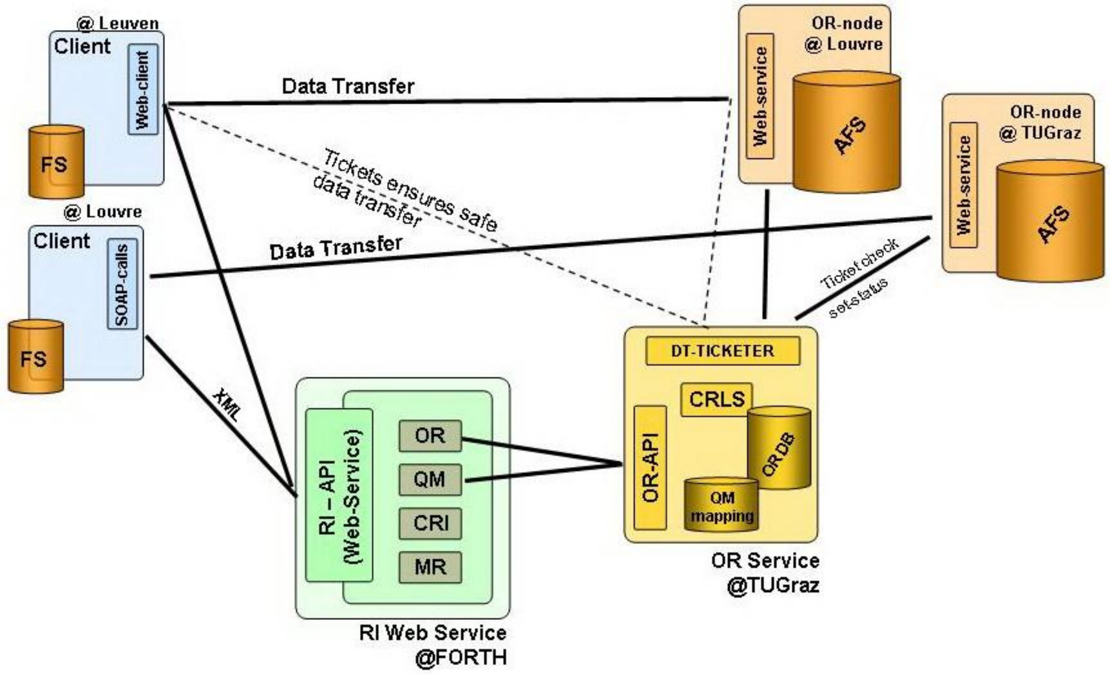


Figure2: OR component and its relevant components

1. Client sends request to RI-service via XML.
2. RI-service forwards the client request to OR-service by calling the functions in OR-service Stub, which has been embedded as Java library in RI-service.
3. OR-service receives the request and executes the internal database (ORDB) operation.
4. The response will be returned to RI-service and,
5. RI-service returns the response to client.
6. If the request is an Ingestion/Retrieval request, OR-service ǿƛ ff ∃ŜყŜNJŀ űŜ ŀ άűƛŎʅŜ will be returned back to client as a pass for starting the data transfer from client to OR-node or vice versa.
7. After transmission, the OR-node will send a signal to OR-service for noticing the internal database to update the table which is responsible for data transfer status. The atomic operation is finished.

At the end of Month 21 we established the first Version of RI and OR, in this release, the OR component (OR API) is a part of RI client library, which would be setup by the client locally. The client communicates with the OR database directly by calling the functions in RI client library and getting the response from the database remotely. Data transfer mechanism was also established in this version. SOAP message is the medium which carries the binary information between client and server. The significant advantage is: SOAP is based on HTTP protocol and can pass through most firewalls conveniently. For this test version, there was only one OR-node, which is located at TUGraz.

The most important achievements of this version were:

Á Distant data transfer for large size file
Á Associated central database for distributed requests
Á Integrated OR API

Figure 3 presents the schema of internal OR database. The purpose is color-coded: the central File table (red), information about the files (green), link tables (blue), dataset grouping (violet), revision Ƴ ŀ ƴ ŀ ∃ Ŝ Ƴ Ŝ ƴ ű ó ƻ NJ ŀ ƴ ∃ Ŝ ú Σ dz ă Ŝ NJ Ƴ ŀ ƴ ŀ ∃ Ŝ Ƴ Ŝ ƴ ű ó è Ŝ ƭ ƭ ƻ ǿ ú Σ Ƴ ƛ foreign key.

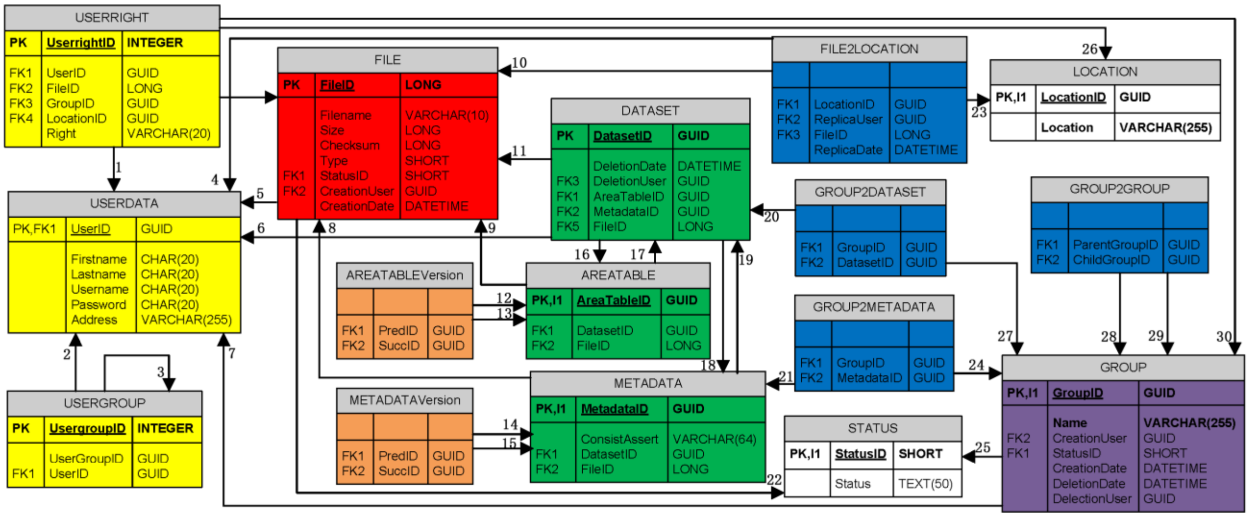


Figure3 Schema of ORDB

Since this release was published for the technical partners, we were going to work on decoupling the OR from RI client library, which means, OR will be a standalone Web-service (see Figure 2) and the only way to access the OR database will be to invoke the RI function first and RI is responsible to redirect the requests from client and calls the corresponding function of OR-service.

We have published the 18 design guidelines and refined 18 internal OR-API functions. They were LJNJ Ŝ ă Ŝ ƴ ű Ŝ Ř ƛ ƴ ű ŃǗ Ʀ Ŝ dzǗ Ŝ ŘŜ Ŋ ō ƻ Ŝ Ŏ ű ƛ ă Ŝ LJ ƻ (see Publications section: / dz ƭ ű dzNJ 3).

6.3 Deviation from work plan

The OR-service was completed in Month 24, with one month delay. The re-engineering of OR and a test prototype of OR have cost about 4 weeks before Integration Week Meeting (Oct. 25 ς28th, 2010).

6.4 Plans for the next period

We will provide during Year 3 each OR-node a special Web-service for data transmission. In the next new release, OR-service will be only a trigger for data transfer, and the real action will be executed directly between client and OR-node, as demonstrated in Figure 2. The most significant advantages are:

- Database operation will be totally encapsulated inside OR-service, port of ORDB towards outside will be closed and this improves the security grad of server. The only way to access the ORDB is calling OR-service through RI-service
- Avoid the data transfer bottleneck between clients and OR-service
- OR-service Ω ă ǿ ≥ NJ ɩ ǿ ƛ f ft much binary data traverse OR-service and OR-service could focus on the internal database operation, which makes the system more stable and efficient
- Ticket, which is generated from OR-service, carries the necessary information for upload and downloading, ensures the authorization of communication between client and OR-node.

The second OR-node will be most probably setup at UBonn. We have arranged to provide a conventional setup program in Month 25/26. And a multiple OR-nodes network will be deployed before Month 30 approximately. User management and privileges management (access-right) have been also taken into account for the beginning of Year 3, these functionalities should be implemented and tested in Month 27 and Month 28 because of the importance and priority.

7 Task 3.5 Design and implementation of a query manager

7.1 Work planned

The objective of this task for the second year was to provide the implementation of a heterogeneous query manager that will combine relational database querying, structured metadata querying and similarity matching.

¢ Ʀ Ŝ ǀj dz Ŝ NJ è ƒ ŀ ƴ Ǝ dz ŀ Ǝ Ŝ ű ƻ ō Ŝ dz ă Ŝ Ř ƛ ă ά { t ! w v [έ Σ ă LJ Ŝ Ŏ ƛ ŀ
retrieve info:

about (Actor, Thing, Event, Place, Date, Type, etc)

related to (Actor, Thing, Event, Place, Date, Type, etc)

The results are presented as a list of entities or as a sub-graph with A-path-B.

In general there are two forms of queries:

a) OR-structure queries are performed on the OR and Semantic queries are performed on the MR. Both use the SPARQL language. The results ƻ Ŧ ă dz Ŏ Ʀ ǀj dz Ŝ NJ ƛ Ŝ ă ŀ NJ Ŝ ŀ ƒ ƛ ă ű ƻ Ŧ h ō where clause.

$$\text{SPARQL} \xrightarrow{returns} \text{list< ObjectID >}$$

b) Index (feature vector) queries are performed on Content Retrieval Indices (CRI) and their result is a ranked list of ObjectIDs.

$$\text{Query object} \xrightarrow{returns} \text{list< ObjectID, rank>}$$

The overall query result is a unified list of the two result sets: list< ObjectID, rank>

7.2 Work performed

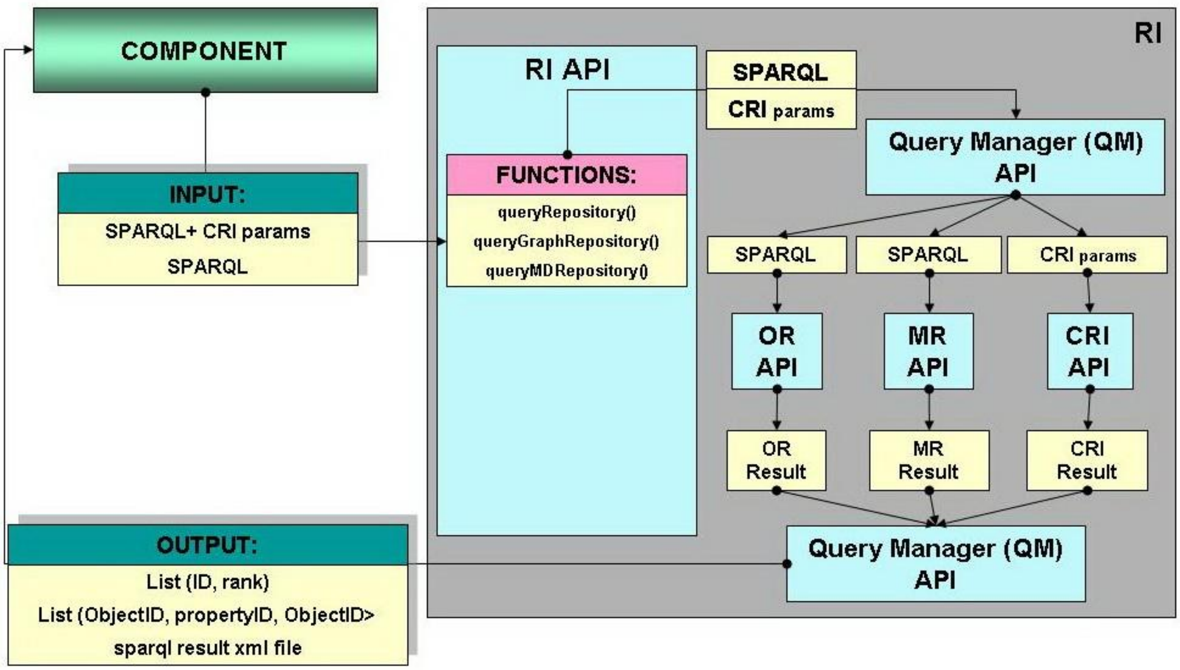


Figure4: The Query Manager (QM) is responsible for redirecting the specific parts of a query to the respective RI services

The Query Manager (QM) module is implemented and constitutes now an integral part of the RI. QM is the module responsible to redirect the specific parts of a query to the respective RI components/services, by splitting it into different query parts (OR-service, MR-service, CRI-service). The way the Query Manager functions is illustrated in Figure 4. It works as follows:

Step 1: QM divides the SPARQL statement into two parts: The first concerning the OR structures is redirected to the OR-service main directory (ORDB) via our D2R-service. OR-service responds with a list of Identifiers.

Step 2: The second part is redirected to the MR-service. MR-service responds with a list of Identifiers.

Step 3: CRI is queried on shape, material features according to arguments passed to the query. Then CRI-service is called and CRI-service responds with a ranked list of Identifiers (list<InstanceID, rank>)

Step 4: QM finally performs an intersection on the resulted lists provided from steps 1, 2 and 3 resulting in a single ranked list of identifiers.

Accessing the OR

Although the OR-service main directory (ORDB) is implemented on a relational database (MySQL) we use D2R Server (a tool for “publishing relational databases on the Semantic Web”) to communicate with

ORDB via SPARQL queries. The D2R server is installed and configured to be accessed by the RI in order to query the ORDB that resides in the OR-service.

CRI material and shape services

We have discussed, re-designed and finalized the architecture and the API of the CRI-services (shape and material).

During the second year we have reviewed state of the art and devised algorithms for feature computation, feature indexing and effective shape retrieval (details of findings and innovation can be found in the Publication section: 4,5 and in particular in the Deliverable D7.2 ςThe second year report of WP7 ςSearching and Browsing 3D Collections).

The CRI components are responsible for indexing and searching all objects stored in the RI, according to similarity of shape and material characteristics, denoted by ĕfeature vectors ἑ

The basic design is concluded in the following:

Á There will be a CRI-service for materials and a CRI-service for shapes. Hereafter we will address them as CRI-services. Each service will be addressed and controlled by the RI-service only. No direct calls from the clients are possible.

Á The functionality of the CRI-service is to (a) calculate a feature vector for a specific object or a region and (b) store it for further use in the CRI-DB, the database of feature-indices (e.g. for retrieval, when invoked by the RI-serviceΩ ă v a Ƴ ƻ Řdz ƒ Ŝ ύ Ӏ· ƴŘ ó Ŏ ύ Ǝ ƛ Ǿ Ŝ ƴ Ӏ· retrieve from the CRI-DB similar objects and return their UUIDs in a ranked list.

We have agreed on the architecture within the CRI-service (the architecture for Shape CRI-service is presented in *Figure5*) and the protocol for its communication with other elements of the system such as the RI. A schematic diagram for this is presented below.

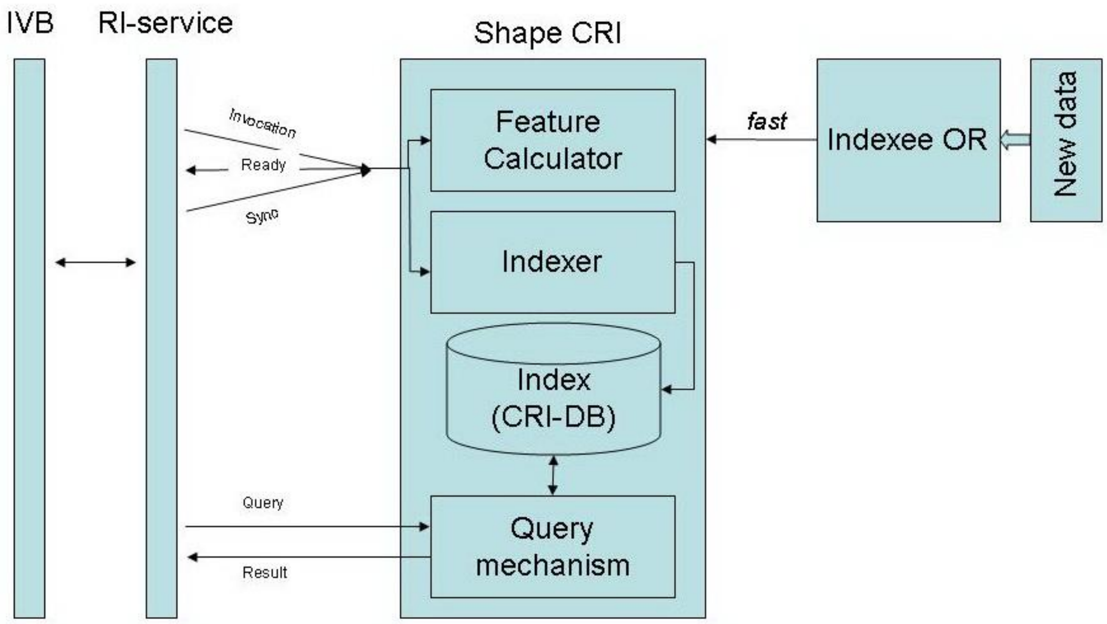


Figure5: Shape CRI-service communication basics

The CRI-service will consist of four parts:

- Feature calculator - This part performs computation of new features (e.g. quantization of shape appearance) on newly acquired data from the Indexee OR. It needs a fast hardware connection to the Indexee OR.
- Indexer - This part indexes the data in the form of features for easy access during the process of querying.
- Index (CRI-DB) - This part is the storage for features that the Feature calculator then calculated on acquired data from the Indexee OR.
- Query Mechanism - This part is the constantly running process that takes requests from the RI-service, parses them, redirects the queries on the CRI-DB and returns the results to the RI-service.

The signals for communication are:

- Update: Upon receiving significant new data or after a specific period, the RI-service sends this message asking the CRI-service to re-compute features and indexing.
- Ready: After completion of features and indexing, the CRI-service returns a signal indicating this is done.
- Sync: The RI-service now asks the CRI-service to use the latest results for future queries.
- Recalculate: Unexpected events may force the CRI-service to recompute its features and index. This may be either a CRI-service initiated or a RI-service initiated signal.
- Query: This is the important final request from the RI-service indicating a query for shape retrieval. It will consist of some fixed bits and some extra bits for internal use of the CRI.

Á Result:This is the result vector of shape IDs and their similarity scores returned to the RI by the CRI-service.

The CRI-service API has been designed to separate the indexer, which preprocesses objects stored in the OR (OR-nodes) for inclusion in the internal CRI databases, from the CRI queries themselves. The indexer can periodically crawl the OR (OR-nodes) for newly added objects, or respond to requests from the RI-service to index specific UUIDs. The choice of indexing policy is under the control of the RI-service so it can properly manage data traffic from the OR-nodes.

Splitting the indexer from the database itself serves two functions. First, it simplifies preprocessing objects (which can be slow) independently of updating the database; this way CRI-services can respond quickly to RI-service "sync" commands that guarantee a consistent set of data between the RI-service, OR-nodes, and all CRI-services because most or all objects in the new sync point have already been preprocessed. Second, it allows the indexer to be physically distributed across OR-nodes, CRI database servers, or independent servers based on convenience of administration, minimizing data traffic, and maximizing available CPU cycles for compute-intensive preprocessing steps.

7.3 Deviation from work plan

No deviation from the work plan. The second year work plan has been fulfilled according to the DoW.

7.4 Plans for the next period

During the third year, we will produce and integrate a web-based service performing the CRI-services that have been described. The feature calculator, indexer and the overall query mechanism will be up and running to be accessed by web-based queries (for both shape and material).

The Query manager will take advantage of the reasoner installed and running on the MR-service.

8 Task 3.6 Design and implementation of an annotation and co-reference manager

8.1 Work planned

The objective of this task for the second year was to refine and finalize the design of the annotation and co-reference model by taking into account the input received from all the partners and to provide an implementation of an annotation and co-reference manager to support a uniform and efficient way of accessing, using, reusing and preserving the available information.

8.2 Work performed

During the reporting period, we received several samples of metadata files produced by the partners (WP4 ς3D Artefact Acquisition, WP5 ς3D Artefact Processing, WP6 ςCreating the 3D Collection Item). We consulted them during the creation of their metadata files and we reviewed and corrected their examples. The consulting/reviewing process resulted in refinements and adaptations of the metadata models (CRMdig ςdefined as an extension of CIDOC-CRM). One major modification was introduced in the way annotations were modelled. The problem that we had to face was how to use a set of assertions (triples) as target for an annotation. A well accepted, standard procedure to deal with such an issue is to use the *Named Graphs*[1] mechanism to treat a bunch of triples as an object in annotation. With such an approach, a named graph will be viewed as a *context* of Sesame but in a more standard way. The new Annotation Model is shown in Figure 6.

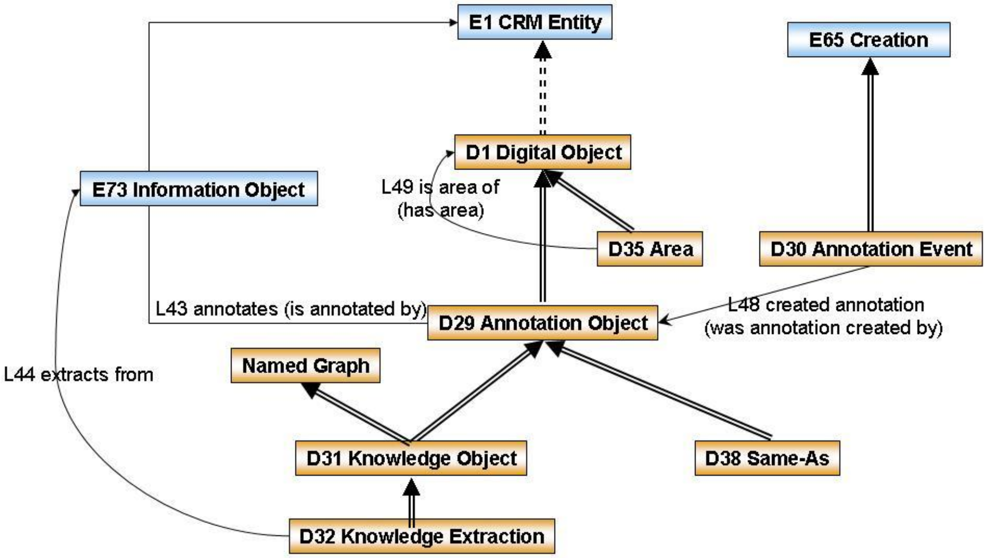


Figure 6: Annotation Model

The Annotation model has two basic entities: the Annotation Event and the Annotation Object. The Annotation Event is the parent event that creates the Annotation Object. The Annotation Object is the entity describing the association between the annotated objects. We define two sub-classes of the Annotation Object: the Knowledge Object modelled as a Named Graph and the Same-As which is used to declare co-reference links. Knowledge Extraction is a specialization of Knowledge Object that is used to model information that will be (semi)automatically extracted from legacy data.

The latest version of the rdf schema for CRMdig is available at: http://www.cidoc-crm.org/rdfs/CRMdig2.4.rdfs and is described in detail in Appendix C- CRMdig.

The work of this task is closely related to work done in WP6 ςCreating the 3D Collection Item and in particular on the Annotation Editor. For details and some examples please refer to D6.2 ςThe second year report of WP6.

8.3 Deviation from work plan

No deviation from the work plan. The second year work plan has been fulfilled according to the DoW.

8.4 Plans for the next period

During the third year, we will integrate the annotation and co-reference managers in the MR-service, the web-based service of the Metadata Repository.

9 Task 3.7 Implementation of a Long term Digital Preservation Management Tool

9.1 Work planned

Taking the results of WP2 ς Requirement Analysis and in particular Task 2.4 on standards and infrastructure, Task 2.5 on provenance, and 2.7 on IPR we will integrate technologies that have been developed in other projects such as Planets, CASPAR and the Digital Curation Centre (to name but three) to underpin the preservation of 3D representations. Given our current knowledge of the work so far done and planned for these projects we know that they will not provide all the components that will be necessary, and as a result of mapping the requirements defined in Task 2.6 ςSpecification of Long Term Digital Preservation Requirements - against the available tools, services and methods we will identify components that will need to be developed in order to deal with 3D Models and their empirical acquisition data. The result will be released in Month 42 as a suite of validated preservation components that can be used by repositories to support the long-term viability of 3D objects.

9.2 Work performed

The LTDP runs as a number of discrete but interconnected services and management applications, broadly defined in terms of the following core high level requirements to be integrated via the RI web service to MR and OR:

- Characterisation of 3D-COFORM Archival Information Object (AIP) according to content, metadata and associated semantic and technical dependencies;
- Application of preservation levels to characterised objects, incorporating corresponding data (including intermediate data), tools and metadata chronology;
- Determination of preservation levels according to object, contextual and dependency characteristics and risks;
- Recording and associating technological and semantic dependencies to inform risk assessments and preservation level recommendations.
- Obsolescence monitoring service prompting administrator of evident preservation risks on the basis of characterisation and risk awareness
- Export and encapsulation of data and metadata into a 3D-COFORM self-describing package format (METS);

9.2.1 Progress

- Requirements definition completed
- Vocabulary for associating risk based on metadata and context established and partially populated on the basis of preservation case studies
- Interface to query obsolescence risks prototyped in Semantic Mediawiki

Á Draft definition of 3D-COFORM METS format for encapsulated content/metadata export completed

Á Progress made towards definition of semantic and technical dependency models

Á Java mock-ups of GUI Interface elements completed

9.2.2 Risk Definition Vocabulary

A vocabulary has been conceived and partially populated (based on legacy digital preservation case studies). Risks are defined in terms of their semantic relationships with Resources, Policy, and Interactions. The *Preserved Object and Repository Risks Ontology* is a novel development conceived to support the recording and traversal of interrelationships between these elements that manifest risk. The (password protected) resource, prototyped using Mediawiki with a number of semantic web extensions is available online at http://porro.hatii.arts.gla.ac.uk/.

Of primary interest are Resources. These are divisible between those things forming the focal point of the preservation activity (*Archival Information Packages*) and those supporting elements required by the preservation process. This distinction enables risks to be represented and comprehended within a common system irrespective of their origins or defining characteristics. Preserved Resources, corresponding to the 3D-COFORM information package, and inclusive of all intrinsic data plus representation information describe the former Resource type. Their structure is considered to be synonymous with a 3D-COFORM AIP structure. Conforming to the National Archives of Australia Ψ t ŜNJŦ ƵNJƳ ŀ ƴŎŜ a Ƶ Ř Ŝ ſ Ω Σ ƛ ƴ Ŧ ƵNJƳ ŀ ű ƛ Ƶ ƴ ƛ ă ŀ ă ădzƳ Ŝ Ř ű Ƶ NJ (representation) to achieve understandability and value.

InformationObject

Á DataObject(s) [raw, intermediate or model data, from OR]

Á Dependencies [encoded behaviours, generally associated representation software code, but also including semantic representation information]

Á TransformationObjectProperties [those characteristics often called significant properties lending information content its value]

Á FixityInformation [MD5 and SHA checksums for encapsulated DataObjects]

Á Metadata [Provenance information, CIDOC CRM, from MR; optionally other preservation metadata such as PREMIS]

Each intrinsic element is collectively or individually associable with risk, or with corresponding Policy or Interactions, which may in turn relate via the Semantic web to risk.

9.2.3 METS 2.0 Profile

The PreservedResource structure described above is represented in a draft METS profile intended to provide a convenient means of packaging, or describing an encapsulated package (as a manifest file), to support offline preservation of identified resources. The respective merits of encapsulating data and code within the METS file are currently being investigated. It is clear that the METS <behavior> element can encapsulate representation dependencies, ensuring the self-evidence of a single exported information object. However, an alternative form of encapsulation using for example .zip or .tar may be preferred in order to limit METS file complexity and encourage content reusability.

The METS <area> element is deployed to indicate divisions within content files, enabling expression of geometric associations between annotations and objects. The extent to which these can survive information manipulations is to be explored.

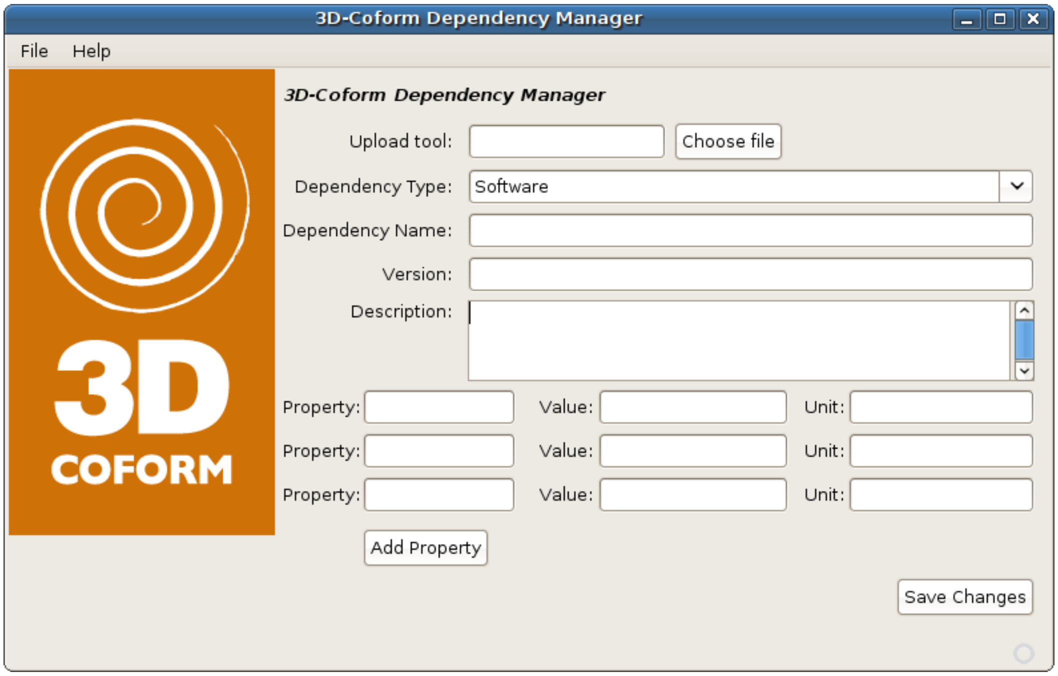
9.2.4 Semantic and Technical Dependencies

A necessary precursor to the creation of a registry/repository of semantic and technical dependencies is a coherent means for their description. Within our implementation such dependencies are based primarily on the OAIS concept of Representation Information ς that which is required to lend a data object meaning and interpretability, its realization as an information object. These are encapsulated within the AIP, with the adopted preservation level influential in determining what is maintained, which may include functionality (in the form of representation/manipulation tools) or interpretative documentation. Each dependency is recorded as a collection of properties, with explicit value and an associated unit of measurement.

Properties are measurable facets of function - *Functional Components* can exist hierarchically, and single functional behaviors may be grouped into wider functions. Any individual dependency may exist in a number of related versions, providing a means for recording variability of rendering, processing and preserved outputs.

9.2.4.1 GUI interface development

Java/Swing mockups have been developed for each administration interface. An example illustrating the dependency manager is included below:



Dependency Type: Software

Figure7: Long Term Digital Preservation Dependency Manager Screenshot

9.2.4.2 Current Activity (to M onth 24)

Á Integration with RI establishment of formal links with the RI is a current priority. As described above, component development to date has been largely repository-agnostic, and validation of their suitability demands imminent integration.

Á Redundancy managementmeans to ensure minimum of three replicated versions of defined AIP across distributed repository is a requirement. At this time this is impeded somewhat by the availability via RI of only a single OR-node, but this is being explored and should be conceptually evaluable.

Á METS Information exportencapsulation of OR and MR content (plus dependency information) in METS/METS manifest-supported package.

Á Obsolescence warning & preservation level recommender systemservice for communicating preservation risk relationships when AIPs are formed or risks emerge based on existing risk ontology is being developed.

9.3 Deviation from work plan

As ű K Ŝ Ŧ Ϩ Ŏdz ă K ŀ ă ō Ŝ Ŝ ƴ -Ğ Ϩ ǀ Y Lă Ϩ Ŝ ǀ Ø Ŝ Ŝ Ǉ ű ŀ ă ƚ Ω Ř ű ʌ Ŭ ă Ŝ Ŝ NJ Ŝ ű ǀ Ŝ ŀ ă Y Y ǀ ƛ Ϩ Ŏ NJ Ϩ ƒ LJ in Month 18 (as had been suggested in the work plan), but a prototype tool will be available in Month 24 illustrating current functionality and illustrating initial RI integration. Current implementations are not integrated with the RI, but work is currently being undertaken to take established information models and apply them more directly, in semantic and structural terms, to the 3D-COFORM object and metadata repositories.

We are satisfied that there is no significant or meaningful divergence from the work plan. Implicit in the work completed to date is the appropriate functional implementation for the period.

9.4 Plans for the next period

There are no significant amendments required for the work plan for the following period. Concisely, the plans remain as follows:

- Continue to develop component for final release in Month 30
- Undertake formal preservation planning process for 3D model data using the Plato Preservation Planning tool, absorbing results into preservation component knowledge base
- Undertake formal preservation capacity and risk evaluation (primarily associated with infrastructural aspects of preservation of 3D materials), absorbing results into knowledge base
- Evaluate 3D-COFORM repository infrastructure for preservation capacity as part of Task 3.9 ς Integration and Testing

10 Task 3.8 Implementation of Watermarking Tool

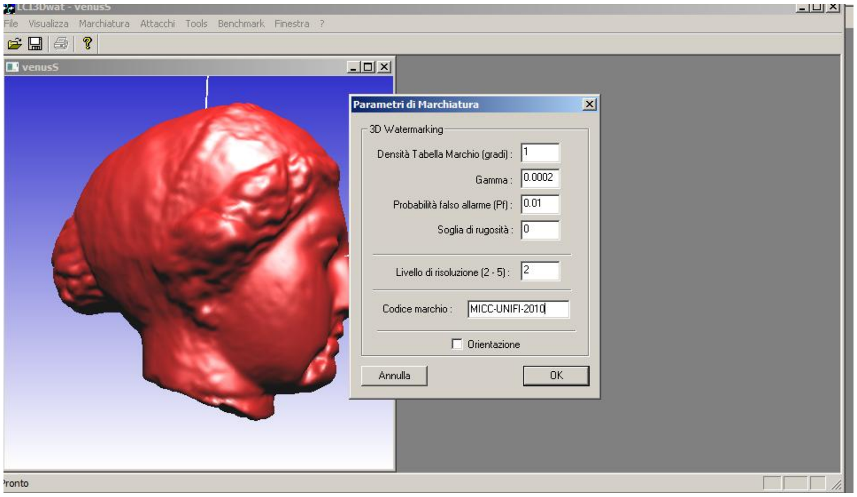
10.1 Work planned

The objective of this task was to investigate specific requirements which, due to the particular type of multimedia objects, can influence watermarking technology design. Furthermore, the plan was to study diverse strategies for the design of watermarking algorithm for 3D models. Robust and blind watermarking algorithms for 3D objects will be researched and security issues will be taken into account.

10.2 Work performed

We studied an ad hoc algorithm for 3D watermark that does not need to alter (remesh) the to-be-marked 3D model.

A specific watermarking signal has been ideated and implemented in order to allow a straightforward integration in MeshLab (for details please refer to D5.2 ςThe second year report of WP5 ς3D Artefact Processing) still maintaining effectiveness and preserving the same perceptual quality of the original mesh. In the following figures we present two screenshots of the first standalone version of the watermarking tool: the encoder (Figure 8) and the decoder (Figure 9).



Parametri di Marchiatura

3D Watermarking

Densità Tabella Marchio (gradi) : 1

Gamma : 0.0002

Probabilità falso allarme (Pf) : 0.01

Soglia di rugosità : 0

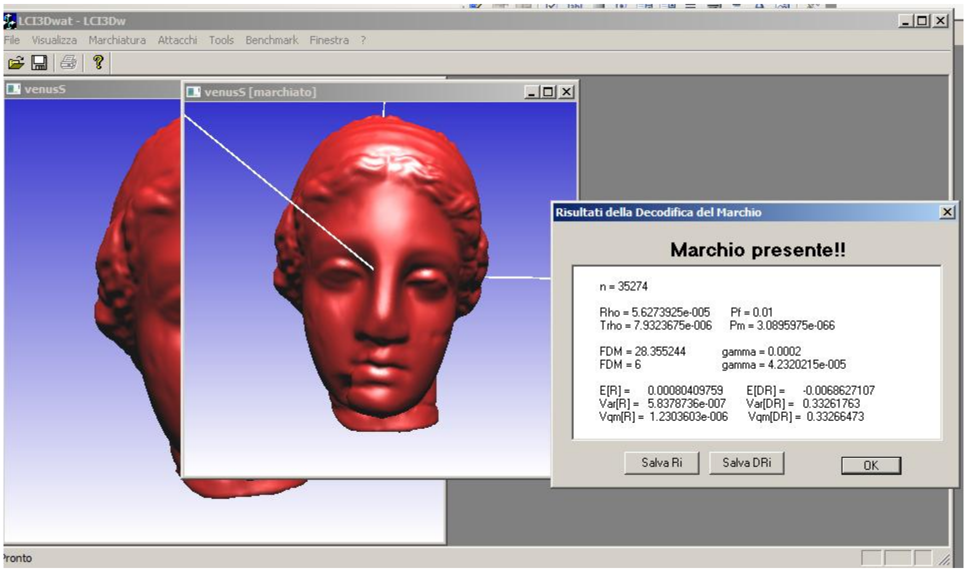
Livello di risoluzione (2 - 5) : 2

Codice marchio : MICC-UNIFI-2010

☐ Orientazione

Annulla OK

Figure 8: Watermark encoder. The user chooses the string to embed in the model



Risultati della Decodifica del Marchio

Marchio presente!!

```
n = 35274

Rho = 5.6273925e-005      Pf = 0.01
Trho = 7.9323675e-006     Pm = 3.0895975e-066

FDM = 28.355244          gamma = 0.0002
FDM = 6                  gamma = 4.2320215e-005

E[R] =     0.00080409759     E[DR] =     -0.0068627107
Var[R] =   5.8378736e-007    Var[DR] =   0.33261763
Vqm[R] =  1.2303603e-006     Vqm[DR] =  0.33266473
```

Figure9: Watermark decoder: the user specifies a string and the system verifies if it is present on the model or not

The watermarking signal is coupled with the 3D model by the spherical coordinates of its vertices, thus relaxing the embedding and detection operations from the knowledge of the mesh topology.

The watermark embedding and detection are thought of as subdividing the mesh vertices coordinates in spherical sectors. Once the mesh is watermarked, as long as inside the same sector, the detector will be able to read the watermark; in this way a controlled (depending on the sectors size) and robust embedding procedure is ensured.

An evaluation phase of the system performances in terms of both missing and false alarm probabilities and perceptual quality has been performed.

We decided to estimate the error probability by collecting the detection results in cases of watermarked and non-watermarked 3D models. With such analysis it will be possible to assess if the analytical theory behind the detector algorithm matches the real case of 3D model of any shape and size.

A visual inspection of the quality of the watermarked model has been performed, and in order to establish the perceived quality we are arranging a set of psycho-perceptual tests.

10.3 Deviation from work plan

No deviation from the plan.

10.4 Plans for the next period

We will proceed with the evaluation of the best theoretical parameter setting in order to ensure the right trade-off between robustness and quality of the watermarked objects in terms of false alarm and missing detection probability and in terms of perceived quality.

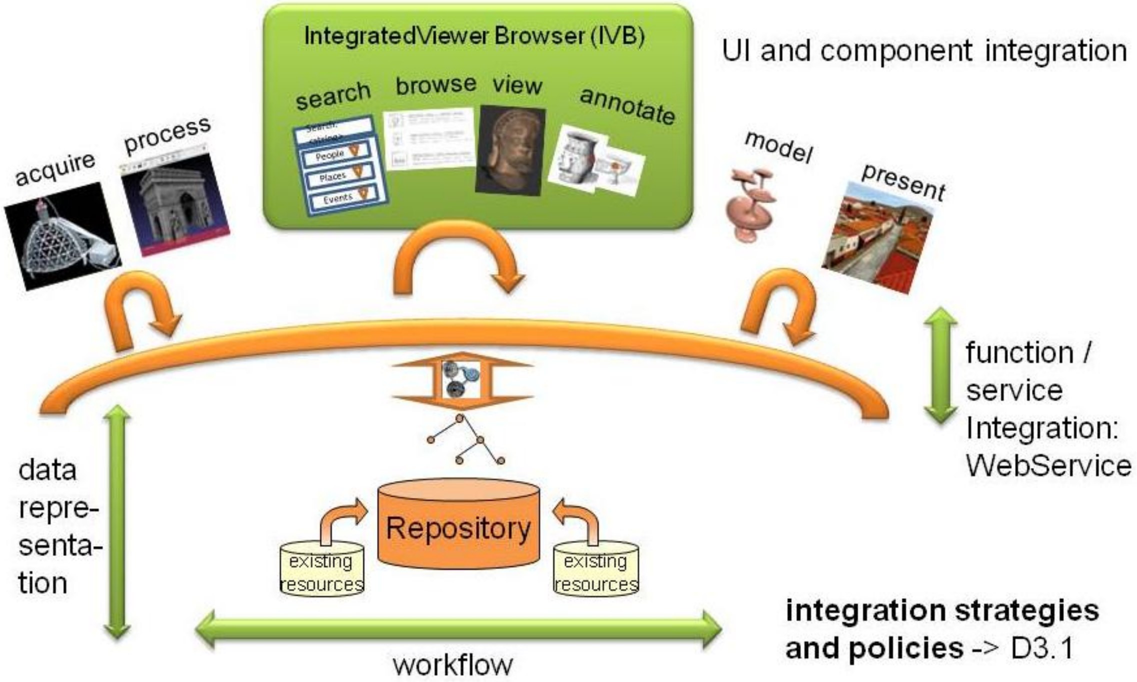
The obtained results will be double checked also in front of an experimental test collecting the detection results in cases of watermarked and non-watermarked 3D models.

11 Task 3.9 Integration and testing

The objective of this task is to coordinate and enable the integration of the different components developed within the project as integrating tools and tool chains, and to test the interfaces and communication between the tools for data and workflows provided by CH partners.

11.1 Work planned

In the second period of the project, Task 3.9 aimed to provide the needed infrastructure for enabling the different types of integration (see Figure below). The major technical integration types within 3D-COFORM are: component integration, service integration and workflow integration. For every type of integration a different strategy is needed. Component integration requires integration at a software level of massively heterogeneous software components. The service integration refers to external communication between different components, which is achieved by means of providing specific functions (services) to other components. The workflow integration deals with the integration of different components into tool chains, representing a CH process with a specific aim and a data set.


IntegratedViewer Browser (IVB)
search
browse
view
annotate
People
Places
Events
UI and component integration
acquire
process
model
present
function / service Integration: WebService
data repre-senta-tion
Repository
existing resources
existing resources
workflow
integration strategies and policies -> D3.1

The overall objective was to provide the needed infrastructure to allow the integration of the different components and to enable the corresponding developer testing at the different levels. This implies setting up an IT software infrastructure, defining clear function interfaces and building common understanding and communication between technical partners and technical partners with CH partners.

11.2Work performed

Component integration and testing

One of the major challenges in 3D-COFORM as an integrating EU project is the integration of several massively heterogeneous software components, consisting of different tools, libraries or applications from many partners, into one homogeneous 3D-COFORM platform. To reach this goal as well as to speed up the development process for all partners, many state-of-the-art technologies and tools were used or developed in the context of Task 3.9. These technologies and tools were primarily needed for IVB and VSL (in the context of WP6 ςCreating the 3D Collection Item, WP7 ςSearching and Browsing 3D Collections and WP9 ςPresenting History), however other tools may benefit as well.

- CMake as cross-platform build system
- FusionForge as collaborative development and source code management environment
- Buildbot as automatic software build and compile test system
- Development of a cross-platform, multi programming language supporting plugin system (For details, please refer to D7.2 ςThe second year report of WP7)

CMake

A cross platform development requires a build system that allows for working with the different programming languages, as is the case in 3D-COFORM, e.g. C++, Java, Flex etc., and which is also able to support different Integrated Development Environments (IDEs) like Visual Studio, Eclipse, KDevelop, QTCreator, Codeblocks or even allows for generating make files and works for different platforms like Windows, Linux or Mac.

FusionForge

In order to promote the effective cooperation among project partners, a well-elaborated collaboration environment that supports bug tracking, shared source code repositories, mailing lists, discussion forums, instant messaging, file release systems for uploading documents and integrated FTP service for data storage is needed. For that purpose, different software solutions were evaluated and it was decided to use FusionForge as the 3D-COFORM collaboration environment, because it is well tested and it is successfully used in many other open source projects.

Buildbot

To speed up the software development process, an open source automatic web-based build system named Buildbot written in Python was adopted. The Buildbot system is used for:

1. Building all external depending libraries from all partners.

   This is an important step, because building only external libraries such as QT, OpenSG and Boost, for instance, which are used by many partners requires in addition to the pure compilation time of 2 days, a large phase of manual adjustment time.

2. Nightly builds to check if the changes in the source code are also working on other platforms. Developing for more than one platform forces the developers to test their changes in other platforms as well. In this case, Buildbot can automatically build code on different platforms overnight.

3. Nightly builds for easy testing and provision of the newest version to other partners within 3D-COFORM.

The above mentioned IT software infrastructure is currently used by several developments within the project, especially the developments requiring integration between partners, e.g. the Visualization Support Library (VSL) for the development of Task 9.1 ς Visualisation Nodes for supported artefact representations, the Integrated Viewer/Browser (IVB) for the development within WP6 ς Creating the 3D Collection Item and WP7 ς Searching and Browsing 3D Collections, and QTGrimaldo for the development of Task 9.3 ς Visualisation and navigation tools for public dissemination. Additionally, a similar infrastructure is used for the development of RI components, which is also undertaken by several partners. A software versioning and revision control system (SVN) was set up, in order to keep track of the code changes and revisions. This is complemented by a bug tracking system for both RI developers and users (e.g. IVB developers), allowing them to report problems with the software and to keep track of their status.

This developing infrastructure is contributing to a more reliable and professional development of tools, where stability, scalability and interoperability is considered. Additionally, in the long term development, it will ease the maturity of the tools and it will enable a rapid modification and integration ƌ Ŧ ŭ K Ŝ / I LJNJ I· Ŏ ŭ ʎ ŭ ʎ ƌ ɣ Ŝ NJ ă Ω Ŧ Ŝ Ŝ Ř ō I· Ŏ ʅ Σ I· Ŏ lj dz ʎ NJ Ŝ Ř Ř dzNJ ʎ ɣ
project.

Service integration and testing

The functional specifications defined in the first period of the project were implemented and preliminary developer testing was performed, in order to evaluate the fulfillment of the requirements for the different needs of the project. This was done during bilateral meetings (e.g. for the RI or the IVB), but also in general meetings like the PrR 2010, PoR 2010 or the IWM 2010. The result of this work is NJ Ŝ Ŧ ƒ Ŝ Ŏ ŭ Ŝ Ř ʎ ɣ ŭ K Ŝ D3.Ø ʎ ς Integrated Ø Repository Architecture & Design ά Specifications έ ø K Ŝ NJ Ŝ ŭ K Ŝ ʎ ɣ ŭ Ŝ NJ Ŧ I· ents are clearly defined. ɣ I· ƒ ƒ ŭ K Ŝ Ŏ ƌ Y LJ
Significant work for the integration of the different components has been going on during the reporting period:

- Á RI-external tools: Long-distance communication, between separate workflow components, is implemented using web services. In particular, the Repository Infrastructure provides its functionality using web services, together with a client-side API with Java bindings (Web-service stub). This interface is used by several tools in the project, e.g. the Long Term Preservation Component for querying, retrieving and encapsulating archival packages for offline storage and to support preservation planning.
- Á RI internal componentsThis approach was also used for the communication of all the internal RI components (MR, OR, CRI).

Á **OR component**: An OR-Webservice stub provides RI possibility to invoke internal OR database operations. The OR-service can activate long-distance data transfer process by means of an ORnode-service stub. Moreover, a standalone OR test tool has been established for testing the feasibility of OR Webservice stub.

Á **MeshLab** The Watermarking tool for 3D meshes has been integrated within MeshLab as a plugin that appears in the filter menu of MeshLab. This has been done according to MeshLab guidelines for software development. This processing tool is composed of two basic parts: an embedder and a detector. Additionally, the texturing of the In-Hand Scanner is also being integrated within MeshLab by means of a common format interface. A new plugin called *3dcoform_io* has been added to MeshLab, in order to retrieve and ingest 3D artifacts and related metadata files into the RI. Specific details regarding maturity of MeshLab and MeshLab components can be found in the Deliverable D5.2 ςSecond Year Report on WP5 ς3D Artefact Processing.

Á **VSL**: a common OpenSG SceneManager was implemented, in order to profit from a common interface, where the different OpenSG nodes for the different 3D representation supported in the project, can be rendered on demand. The Deliverable D9.2 ςSecond Year Report on WP9 ς Presenting History has more information in this regard.

Á **IVB**: a plugin system was developed, enabling the integration of different components (e.g. Query Formulation Interface - QFI, Browsing Interface, Viewers and AnnotationEditor) and the reusability of common services (e.g. there is only one plugin dealing with the communication with the RI).The plugin system is described in the Deliverable D7.2 ςSecond Year Report on WP7 ςSearching and Browsing 3D Collections.

As was mentioned before, the integration was ensured by means of different strategies, according to the specific needs of the development, nonetheless all the tools have been developer tested in different stages, not only by the original developer, but also by other developers who need to interact with the tools. Therefore, the stability, scalability and interoperability of the tools have been assessed and the forthcoming user testing and training activities will provide valuable feedback, in order to achieve the maturity required by the CH practitioners.

Workflow integration and testing

The CH partners received information regarding the capabilities of the 3D-COFORM technology, in order to motivate them to generate CH stories and therefore provide scenarios, where the tools could be employed. These stories were complemented by the corresponding data, thus system testing could successfully be performed during the second period at different venues, e.g. the PrR 2010, PoR 2010 and the IWM 2010. Although different stories have been provided for this purpose, three main stories represented the different areas of the 3D-COFORM pipeline in the second period of the project:

¶ Acquisition ςRepository ςProcessing: Workflow integration between acquisition from images (ARC 3D) and processing of 3D artifacts (MeshLab), including the ingestion into the RI.

- ¶ Metadata Extraction - Repository: A tied integration between the AnnoMAD tool and the Multilingual Support Library (MLSL) for extracting textual metadata and ingesting it into the RI.
- ¶ Modeling ςRepository: A workflow integration enabling the modeling of pictorial maps (Footprint Extractor) as basis for modeling sites (CityEngine) and the corresponding ingestion into the RI.

Additionally, the provided stories and data allowed the developers to remotely perform more reliable developer testing with real data and workflows defined by the CH practitioners. This process has also contributed to the stability, scalability and interoperability of the tools. Different issues were encountered by means of these activities and the corrective actions were planned and performed, in order to improve the capabilities of the tools for CH purposes.

The experience of the involved CH partners in the different areas of the project is very valuable for the integration and testing of the 3D-COFORM technology. For example, VAM has a vast experience with the management of several hundred thousand digital 2D images for all museum purposes. All of this material has been managed and archived with the V&A Digital Asset management System, which acts in a similar way to the RI, therefore the workflow for the ingestion of this large volume of assets is well established. Such experiences from the CH partners provide guidelines and good practices for the development of the project. Thus, the integration and testing of the technology of the project have also profited from the involvement of CH practitioners. For more details, please refer to the work conducted as release and acceptance testing in the context of WP10 ςAssessment and Evaluation, Deliverable D10.2.

11.3 Deviation from work plan

No deviation from the work plan. The second year work plan has been fulfilled according to the DoW.

11.4 Plans for the next period

Task 3.9 will continue providing the needed infrastructure to allow the integration of the 3D-COFORM technology and to enable the developer testing at the different levels (component, interface or system testing). Additionally, it will promote a deeper involvement of the CH partners, by means of providing reliable information regarding the functionality of the tools, which will allow them to build a better dzƴŘŜNJăűŀƴŘƛƴ∃ ƻŦ űKŜ űƻƻƖăΩ ŎŀLJŀōƛƖƛűƛŜă ŀƴŘ űKŜ scenarios, where the tools can successfully be used.

12 Publications

[1] M. Doerr, K. Tzompanaki, M. Theodoridou, Ch. Georgis, A. Axaridou, S. Havemann, A Repository for 3D Model Production and Interpretation in Culture and Beyond, 11th International Symposium on Virtual Reality, Archaeology and Cultural Heritage VAST (2010), Paris, 21-24 September 2010

[2] D. Pitzalis, F. Niccolucci, M. Theodoriou and M. Doerr, LIDO and CRMdig from a 3D Cultural Heritage Documentation Perspective, 11th International Symposium on Virtual Reality, Archaeology and Cultural Heritage VAST (2010), Paris, 21-24 September 2010

[3] X. Pan, P. Beckmann, S. Havemann, K. Tzompanaki, M. Doerr & D. W. Fellner, A Distributed Object Repository for Cultural Heritage, 11th International Symposium on Virtual Reality, Archaeology and Cultural Heritage VAST (2010), Paris, 21-24 September 2010

In this section, we list the publications reported in other workpackages (WP7, WP5) that are relevant to WP3.

[4] Knopp, J., Prasad, M., Willems, G., Timofte, R., Van Gool, L.: Hough transform and 3d surf for robust three dimensional classification. In: ECCV. (2010)

[5] Knopp, J., Prasad, M., Van Gool, L: Orientation Invariant 3D Object Classification using Hough Transform Based Methods. In: ACM Multimedia 3DOR. (2010)

[6] Irene Amerini, Lamberto Ballan, Roberto Caldelli, Alberto Del Bimbo and Giuseppe Serra, ǵGeometric tampering estimation by means of a sift-ō ŀ ă Ŝ Ř Ŧ ƻ NJŜ ƴ ă ƛ Ŏ ŀ ƴ ŀ ƒ è ă ƛ ă έ Σ L 9 9 9 Acoustic, Speech and Signal Processing, ICASSP 2010, March 14-19, 2010, Dallas, TX, USA.

[7] Roberto Caldelli, Irene Amerini, Francesco Picchioni, a ŀ ű ű Ŝ ƻ L ƴ ƴ ƻ Ŏ Ŝ ƴ ű ƛ Σ ά C ŀ ă ű L ¦ ƴ ʅ ƴ ƻ ǿ ƴ { ƻ dzNJŎ Ŝ L Ƴ ŀ Ǝ Ŝ ă έ L 9 9 9 ² ƻ NJ ʅ ă K ƻ LJ ƻ15ƴ DeceLƳŀmber Ǝ Ŝ C ƻ NJŜ 2010, Seattle WA, USA.

13 References

In this section, we list references to works conducted outside 3D-COFORM.

[1] Definition of the CIDOC Conceptual Reference Model
http://www.cidoc-crm.org/docs/cidoc_crm_version_5.0.2.pdf

[2] Named Graphs http://www.w3.org/2004/03/trix/

[3] The TriG Syntax http://www4.wiwiss.fu-berlin.de/bizer/TriG/