Grant Agreement No.	ICT-2009-270082
Project Acronym	PATHS
Project full title	**Personalised Access To cultural Heritage Spaces**

D 3.3 Second Prototype and Documentation

Author: **Stein Runar Bergheim (AVINET)**

Contributors: **Mark Hall (USFD)**
Eneko Agirre (UPV/EHU)
Kate Fernie (MDR Partners)
Alok Singh (AVINET)

Project funded under FP7-ICT-2009-6 Challenge 4 – "Digital Libraries and Content"	
Status	Final
Distribution level	Public
Date of delivery	03.05.2013
Type	Software, documentation
Project website	http://www.paths-project.eu
Project Coordinator	Dr. Mark Stevenson University of Sheffield

Change Log

Version	Date	Amended by	Changes
0.1	01/04/2013	Stein Runar Bergheim	TOC
0.2	30/04/2013	Stein Runar Bergheim	Draft
0.9	03/05/2013	Stein Runar Bergheim	Revised draft, inputs from Kate Fernie, Mark Hall
1.0	03/05/2013	Stein Runar Bergheim	Incorporating inputs from Kate Fernie, Mark Hall, Jillian Griffiths and Mark Stevenson

Table of Contents

1 Executive Summary

In May 2012, the first PATHS prototype and Web Service API was released. The system was subject to intensive testing in the period up to September 2012. During the testing period, feedback was gathered through combination of user trials, demonstrations and professional assessments. Specifications for the final prototype were finalized during the last quarter of 2012 and the development of the final prototype described in this document took place from January to April 2013.

Building on the experiences from the first prototype, the project is continuing its user-centric approach to design and development. The second prototype is an incremental build of the first; retaining all features that were assessed as functioning well and useful by end-users; enhancing issues that were identified during the trials; and refining the overall functionality.

The final prototype introduces several new and innovative features including but not limited to map based and thesaurus based navigation.

The final prototype is released in May 2013. This document provides an overview of the PATHS system including hardware, software, applications and interfaces, details about the development process itself and comprehensive data model and API documentation.

The deliverable consists of three major parts:

1. A logical data model, the *PATHS Database*
2. A web service API, the *PATHS Web API*
3. A web application, the *PATHS Prototype User Interface* (UI)

The data model is implemented as a combination of SQL and non-SQL databases and indexes combining the power of the PostgreSQL RDBMS with that of Apache Solr inverted index and Virtuoso RDF triple store. All relationships and references in the data model are implemented using persistent URIs as foreign keys. This allows for flexible integration between the three data stores.

The web service API is implemented on top of a wide range of server components and controls all data I/O operations towards the data layer. It consists of more than 30 different web methods grouped into seven Web Services. The Web Services communicate over the HttpGet, HttpPost, Soap and Soap 1.2 protocols. The default return format is JSON but the services are also capable of delivering XML.

The web application is implemented with a number of sophisticated end-user interfaces that rely and data received through Web Service requests for their operation. The application is user centric and emphasizes good interaction design as well as innovative modes of exploration.

This prototype is designed to demonstrate the core functionality of the system and the potential of the navigation, information retrieval and content enrichment methodology proposed by the project.

Deliverable D3.3 is based on the user requirements defined in D1.5 “Functional Specification of Final Prototype” and D4.2 “Final Prototype Interface Design”. The application builds on D3.2 “First prototype and documentation” and publishes data processed and made available from D2.3 “Processing and Representation of Content for Second Prototype”.

In the next months, the final prototype will be evaluated by users and, together with the laboratory trials, will lead to further refinements and enhancements feeding into the long-term sustainability and market planning for PATHS project results.   This report provides an overview of the different parts of the system and seeks to provide a platform for conducting system, technical and end-user testing; and to provide technical reference documentation for third parties who are interested in implementing services on the comprehensive PATHS Web API.

2 Introduction

This deliverable, "D3.3 Second Prototype and Documentation" provides an introduction and thorough description of the second version of the web API and prototype user interface of the PATHS system. The second prototype is an incremental improvement upon the first which was released in May 2012, providing several new modules and a vast array of enhancements to both the data layer, the API and in particular the prototype user interface.

The technical reference information required to use the API is included with this deliverable, however the high-level conceptual and architectural diagrams that were included in D3.2 are not repeated as they are still valid and remain unchanged.

Instead, the document highlights new features and enhancements that are of interest either from a technical and/or an end-user perspective. A section that describes the development methodology is included.

2.1 Second PATHS Prototype Overview

The second prototype is an incremental build of the first prototype. All functionality that was present in the first prototype has been retained and/or enhanced in the second.

The prototype conceptually consists of five different elements in a quad-layer structure. The development methodology is parallel to the software architecture and is described in a separate chapter.

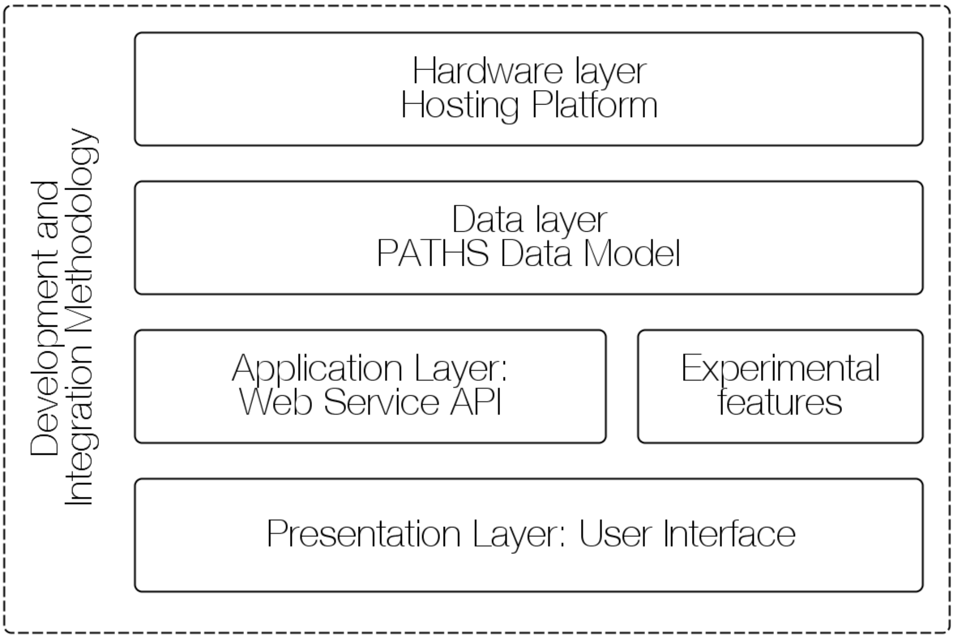

Development and Integration Methodology
Hardware layer Hosting Platform
Data layer PATHS Data Model
Application Layer: Web Service API
Experimental features
Presentation Layer: User Interface

Figure 1: Overview of hardware, software and process in the PATHS system

2.2 Relationship to other deliverables

This deliverable builds upon work reported in six previous PATHS deliverables:

- D1.1 “User requirements analysis”
- D1.5 “Functional Specification of Final Prototype”
- D3.1 “Specification of System Architecture”
- D3.2 “First Prototype and Documentation”
- D4.2 “Final Prototype Interface Design”
- D5.1 “Evaluation of First Prototype”

The prototype also provides access to enriched data described in:

- D2.2 “Processing and Representation of Content for Second Prototype”

Copies of these deliverables are available from http://www.paths-project.eu/eng/Resources

3 Development Methodology

This section describes the methodology applied in the planning and execution of the development tasks of the second prototype.

3.1 Team

The development has been carried out by a decentralized set of developers sourced from four partners with committed resources to the technical objectives of the project.

- Developers from the University of Sheffield (UK) have implemented the prototype user interface. The technical specifications for the Web Service API have largely been derived from the specifications and prototypes.
- Developers from the University of the Basque Country (Spain) have contributed code to the recommender system and prototype enrichment functions in the Web Service API.
- Developers from Asplan Viak Internet (Norway) have been the main responsible party for the development of the Web Service API, data loading. AVINET is also responsible for setting up and running the platform with its various components.
- Developers from iSieve Technologies (Greece) have contributed code for real-time content-enrichment based on their sentiment analysis technology.

3.2 Working methodology

With a decentralized team, the following three issues become critical to a successful outcome: time-planning, communication and clear division of roles and responsibilities.

Time-planning

While the overall project time-frame provides ample time for developing all the components of the PATHS system, the number of dependant tasks confines the major development efforts into a concentrated space of time.

During this interval of time, partners have guaranteed the availability of their development resources through careful time-planning.

Communication

When developers are not working from the same physical environment it is challenging to ensure synchronization between dependent tasks as well as up-to-date knowledge of each other’s activities at all times.

To mitigate this issue, the technical teams have taken part in online technical conferences every two weeks throughout the development phase. In these conferences, issues concerning the progress of the implementation has been discussed and resolved through direct clarifications and/or follow-up actions.

Furthermore, a development management system, RedMine, has been installed at USFD to handle user requirements specifications, development tasks, feature requests and bug-fix tickets.

Source code and pre-processed data from D2.3 is managed in a source code repository based on the Subversion SVN software.

Roles and responsibilities

Each partner has, as far as possible, been given self-contained tasks that may be developed independently up to the time of integration.

This ensures that development resources are used at their full capacity and that time is not lost due to waiting for dependant tasks.

While self-contained tasks have been possible for the individual modules, the final integration task relies heavily on direct communication between USFD and AVINET as the main responsible parties for the prototype user interface and the Web Service API respectively.

This task has been limited to only two parties in order to reduce the risk of lengthy delays

3.3 Testing

The decentralized development methodology and software architecture based on a Web Service API makes testing particularly critical.

A call is issued from the client application with a set of parameters. The Web API must be aware of all incoming parameters and must be able to respond according to a format that is known and documented to the client application.

With the client application being developed in one place and the API in another, developers run the risk of having to interrupt their work while awaiting feedback or input from their counterparts who may be on a different schedule. The process of testing is therefore closely linked to time-planning and is conducted intensively during a very concentrated period of time with a high frequency of exchanges between the key developers.

Most of this communication has been handled through RedMine where each issue has been followed up with questions and clarifications, allowing not only for a convenient way of keeping track of requirements but also provides an audit-trail to see when and how a feature was introduced and implemented. For an example of how a development ticket may look, please see “Appendix D – RedMine ticket example“

3.4 Documentation

With a continuously evolving code-base and a data model that cannot be frozen until all data processing has been completed, documentation also becomes an issue.

The development teams have taken on this challenge by observing the following practices:

- To use inline XML Documentation Comments for classes, methods and method parameters.
- To use inline SQL comments on tables and fields at all times throughout the evolution of the data model.

By following these two principles, the majority of the data model and Web Service API reference documentation can be auto-compiled, minimizing the subsequent need for manual editing upon completion of the development period.

As the development is a highly dynamic process it is important to be able to quickly compile new versions of the documentation without having to revisit all aspects of the application.

XML Documentation Comments

The inline-comments follow the standard notation for XML-comments. The two examples below shows comments for a web service element. Comments are available for the method, for each input parameter(s), for the return value(s) as well as any remarks.

```
/// <summary>
/// The Usr web service contains methods for authenticating users,
/// creating and modifying users, logging user behavior and issuing
/// reminder e-mails upon forgetting passwords. The service is
/// fundamental to web services which require authentication.
/// </summary>
```

```
/// <summary>
/// Returns information about the user identified by the specified ID
/// </summary>
/// <param name="uri">uri of user</param>
/// <returns>User data object</returns>
/// <remarks></remarks>
```

Per service WSDL documentation

Because the Web Services are implemented using C#.NET web service framework, machine-readable documentation is auto-generated based on the function and parameter names as well as data types defined in the C# application code classes. This information is serialized into WSDL (Web Service Description Language) XML and made available to the client interfaces.

A WSDL file can be quite long, the below XML-fragment shows a small extract from a more comprehensive file.

```
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:t
m="http://microsoft.com/wsdl/mime/textMatching/"xmlns:soapenc="http://schema
s.xmlsoap.org/soap/encoding/" xmlns:mime="http://schemas.xmlsoap.org/wsdl/mi
me/" xmlns:tns="http://paths-
project.eu/"xmlns:s="http://www.w3.org/2001/XMLSchema" xmlns:soap12="http://
schemas.xmlsoap.org/wsdl/soap12/" xmlns:http="http://schemas.xmlsoap.org/wsd
l/http/"xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" targetNamespace="http:
//paths-project.eu/" slick-uniqueid="3">
   <wsdl:types>
          <s:schema elementFormDefault="qualified" targetNamespace="http://p
          aths-project.eu/">
                 <s:element name="Current">
                        <s:complexType/>
                 </s:element>
                 <s:element name="CurrentResponse">
                        <s:complexType>
                               <s:sequence>
                                      <s:element minOccurs="0" maxOccurs="1" n
                                      ame="CurrentResult" type="s:string"/>
                               </s:sequence>
                        </s:complexType>
                 </s:element>
          <s:element name="LogPage">
                 <s:complexType>
                        <s:sequence>
                               <s:element minOccurs="0" maxOccurs="1" name="d
                               c_title" type="s:string"/>
                               <s:element minOccurs="0" maxOccurs="1" name="d
                               c_source" type="s:string"/>
                        </s:sequence>
                 </s:complexType>
          </s:element>
          ...
```

Using the XML Documentation Comments in combination with the Web Service Description Language (WSDL) information for each Web Service, it is possible to create a complete documentation set for the Web Service API by means of a simple XSLT-transformation. The result is the API reference documentation included in the appendices to this deliverable, see “Appendix C – PATHS Web Service API”.

4 Hardware Layer

Although the hardware layer is an essential prerequisite, the platform is entirely nevertheless entirely transparent to the end-user. It is therefore necessary to provide a brief description of the main elements that together constitute the PATHS hosting platform. This information is important if the platform is to be made deployed to other collections and organizational contexts as envisaged in the PATHS dissemination and exploitation strategies.

4.1 Server

For the purpose of portability, the entire system is installed in a single server machine. To increase flexibility, the platform is hosted within a virtual machine so that it is easy to create new instances of the system, or move existing instances between hosting providers.

Table 1: Specifications of server hardware used in hosting environment

Item	Specification
CPU	4 x 2GHz
RAM	16 GB
Disk space	2 TB

4.2 Server management

In addition to the server machine itself, the hosting environment includes a number of technical and practical measures to guarantee the uninterrupted availability of the PATHS system according to a professional service level agreement.

Table 2: Security measures and service level agreement items for application hosting

Measure	Description
Support	The data centre is serviced with instant response during normal working hours 08:00-.16:00 Mon-Fri. Outside of these hours, support is available on two hours’ notice (from 16:00 to 22:00) and on four hours’ notice (from 22:00 to 08:00).
Cooling	Several cooling aggregates are installed in the data centre. See also sensors.
Backup	Data are backed up to a file server as well as to a tape streamer.
Hubs, switches	Only gigabit switches are being used in the data centre.
Firewall	Public access over the Internet is restricted to port 80 (http) and port 21 (ftp).
Cabling	Cables are clearly marked, have new connectors and are attached to each other using cable strips.
Sensors	Sensors to detect fluctuations in temperature as well as smoke are installed in the data centre

Measure	Description
Remote access	Remote access to the data centre is possible through Windows Remote Desktop and TeamViewer subject to double authentication and use of a proxy server with a fixed IP address within the AVINET (Asplan Viak) enterprise network.

The final requirement for the hardware part of the system is Internet connectivity and bandwidth. The hosting environment is presently providing a 50 Megabit connection to the Internet.

5 Data Layer

This section describes the data layer of the PATHS application. The data layer is entirely virtualized from the perspective of the end-user interface. All data I/O operations are conducted exclusively through the Web Service API subject to calling the authentication methods.

5.1 Data model

The paths data model defines the available tables/entities, attributes, primary and foreign keys as well as other constraints. The model is designed using the Datanamic Dezign software and is developed with PostgreSQL 9.2 as target system.

The data model makes use of all SQL 92 features as well as additional geometric data types defined by the spatial extension to PostgreSQL - PostGIS. This extension introduces a number of advanced spatial data representation and processing techniques in compliance with the OGC Simple Features Specification and its associated geometrical operators.

The data model defines the following 12 distinctive data types as well as a number of supporting tables to facilitate specific functionality in the end-user interface.

- Usr: contains user details such as username, password etc.
- Path: contains metadata about paths
- Node: contains metadata about any object identifiable by a URI, in the present implementation, nodes primarily refer to items
- Item: ESE data imported from Europeana enriched with relevance metrics, keywords etc.
- Background link: ESEpaths links to Wikipedia and other web content
- Similarity link: ESEpaths links between items
- Topic: a multi-hierarchical topic/concept hierarchy
- Map point: spatial data representing semantic space
- Map polygon: spatial data representing semantic space
- Rating: contains rating for any object identifiable by a URI
- Comment: contains comments for any object identifiable by a URI
- Tag: contains tags for any object identifiable by a URI.

The illustration below shows an E-R diagram of the data model. The full-size diagram as well as comprehensive documentation of the various tables and attributes is included in Appendix B, C and D are independent documents attached to this report in the order specified below. Each document is separated by a cover page stating the identity of the document. Page numbering is individual to each document.

Appendix B – PATHS data model documentation.

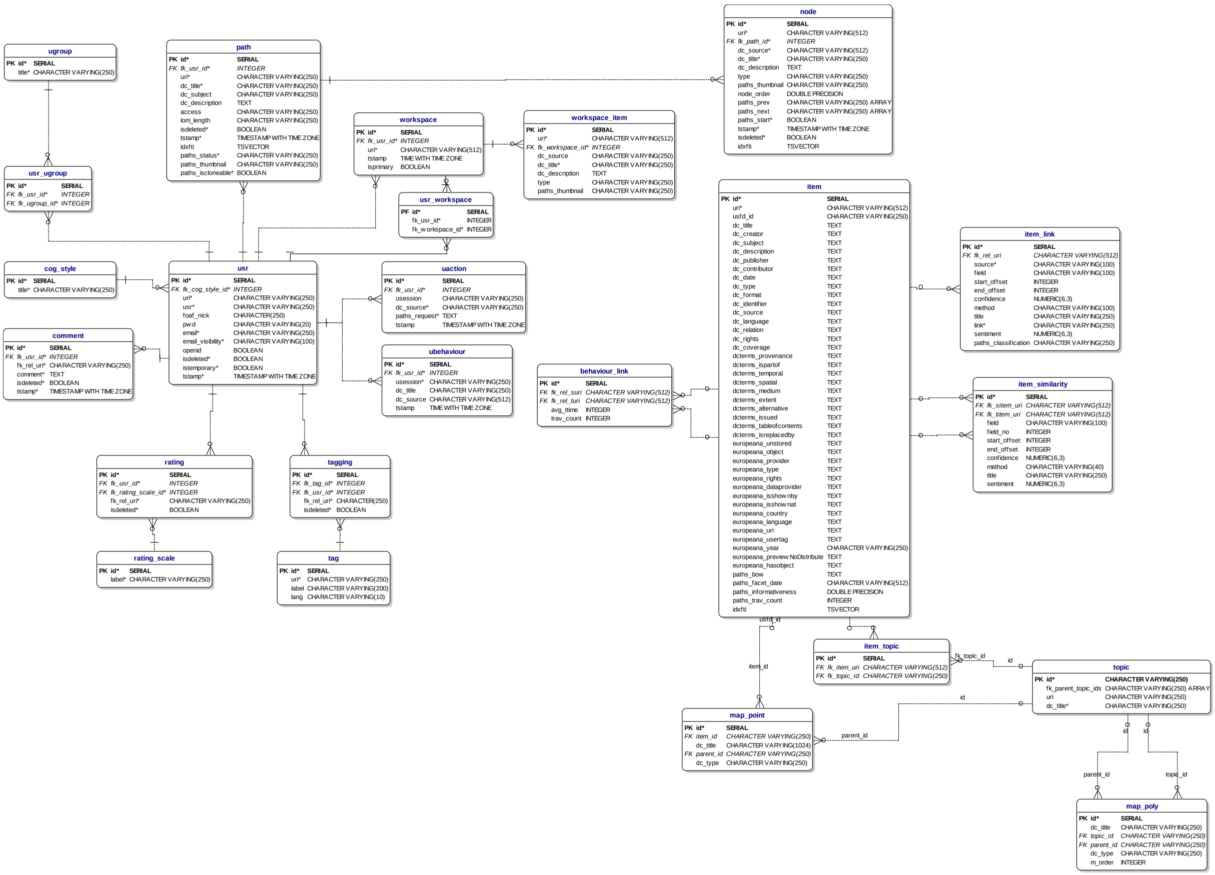

ugroup
PK id* SERIAL
title* CHARACTER VARYING(250)
usr_ugroup
PK id* SERIAL
FK fk_usr_id* INTEGER
FK fk_ugroup_id* INTEGER
path
PK id* SERIAL
FK fk_usr_id* INTEGER
uri* CHARACTER VARYING(250)
dc_title* CHARACTER VARYING(250)
dc_subject CHARACTER VARYING(250)
dc_description TEXT
access CHARACTER VARYING(250)
lom_length CHARACTER VARYING(250)
isdeleted* BOOLEAN
tstamp* TIMESTAMP WITH TIME ZONE
idxfti TSVECTOR
paths_status* CHARACTER VARYING(250)
paths_thumbnail CHARACTER VARYING(250)
paths_iscloneable* BOOLEAN
workspace
PK id* SERIAL
FK fk_usr_id* INTEGER
uri* CHARACTER VARYING(512)
tstamp TIME WITH TIME ZONE
isprimary BOOLEAN
usr_workspace
PF id* SERIAL
fk_usr_id* INTEGER
fk_workspace_id* INTEGER
workspace_item
PK id* SERIAL
uri* CHARACTER VARYING(512)
FK fk_workspace_id* INTEGER
dc_source CHARACTER VARYING(250)
dc_title* CHARACTER VARYING(250)
dc_description TEXT
type CHARACTER VARYING(250)
paths_thumbnail CHARACTER VARYING(250)
node
PK id* SERIAL
uri* CHARACTER VARYING(512)
FK fk_path_id* INTEGER
dc_source* CHARACTER VARYING(512)
dc_title* CHARACTER VARYING(250)
dc_description TEXT
type CHARACTER VARYING(250)
paths_thumbnail CHARACTER VARYING(250)
node_order DOUBLE PRECISION
paths_prev CHARACTER VARYING(250) ARRAY
paths_next CHARACTER VARYING(250) ARRAY
paths_start* BOOLEAN
tstamp* TIMESTAMP WITH TIME ZONE
isdeleted* BOOLEAN
idxfti TSVECTOR
cog_style
PK id* SERIAL
title* CHARACTER VARYING(250)
comment
PK id* SERIAL
FK fk_usr_id* INTEGER
fk_rel_uri* CHARACTER VARYING(250)
comment* TEXT
isdeleted* BOOLEAN
tstamp* TIMESTAMP WITH TIME ZONE
usr
PK id* SERIAL
FK fk_cog_style_id* INTEGER
uri* CHARACTER VARYING(250)
usr* CHARACTER VARYING(250)
foaf_nick CHARACTER(250)
pwd CHARACTER VARYING(20)
email* CHARACTER VARYING(250)
email_visibility* CHARACTER VARYING(100)
openid BOOLEAN
isdeleted* BOOLEAN
istemporary* BOOLEAN
tstamp* TIMESTAMP WITH TIME ZONE
uaction
PK id* SERIAL
FK fk_usr_id* INTEGER
usession CHARACTER VARYING(250)
dc_source* CHARACTER VARYING(250)
paths_request* TEXT
tstamp TIMESTAMP WITH TIME ZONE
ubehaviour
PK id* SERIAL
FK fk_usr_id* INTEGER
usession* CHARACTER VARYING(250)
dc_title CHARACTER VARYING(250)
dc_source CHARACTER VARYING(512)
tstamp TIME WITH TIME ZONE
behaviour_link
PK id* SERIAL
FK fk_rel_suri CHARACTER VARYING(512)
FK fk_rel_turi CHARACTER VARYING(512)
avg_ttime INTEGER
trav_count INTEGER
rating
PK id* SERIAL
FK fk_usr_id* INTEGER
FK fk_rating_scale_id* INTEGER
fk_rel_uri* CHARACTER VARYING(250)
isdeleted* BOOLEAN
tagging
PK id* SERIAL
FK fk_tag_id* INTEGER
FK fk_usr_id* INTEGER
fk_rel_uri* CHARACTER(250)
isdeleted* BOOLEAN
rating_scale
PK id* SERIAL
label* CHARACTER VARYING(250)
tag
PK id* SERIAL
uri* CHARACTER VARYING(250)
label CHARACTER VARYING(200)
lang CHARACTER VARYING(10)
item
PK id* SERIAL
uri* CHARACTER VARYING(512)
usfd_id CHARACTER VARYING(250)
dc_title TEXT
dc_creator TEXT
dc_subject TEXT
dc_description TEXT
dc_publisher TEXT
dc_contributor TEXT
dc_date TEXT
dc_type TEXT
dc_format TEXT
dc_identifier TEXT
dc_source TEXT
dc_language TEXT
dc_relation TEXT
dc_rights TEXT
dc_coverage TEXT
dcterms_provenance TEXT
dcterms_ispartof TEXT
dcterms_temporal TEXT
dcterms_spatial TEXT
dcterms_medium TEXT
dcterms_extent TEXT
dcterms_alternative TEXT
dcterms_issued TEXT
dcterms_tableofcontents TEXT
dcterms_isreplacedby TEXT
europeana_unstored TEXT
europeana_object TEXT
europeana_provider TEXT
europeana_type TEXT
europeana_rights TEXT
europeana_dataprovider TEXT
europeana_isshownby TEXT
europeana_isshownat TEXT
europeana_country TEXT
europeana_language TEXT
europeana_uri TEXT
europeana_usertag TEXT
europeana_year CHARACTER VARYING(250)
europeana_preview NoDistribute TEXT
europeana_hasobject TEXT
paths_bow TEXT
paths_facet_date CHARACTER VARYING(512)
paths_informativeness DOUBLE PRECISION
paths_trav_count INTEGER
idxfti TSVECTOR
item_link
PK id* SERIAL
FK fk_rel_uri CHARACTER VARYING(512)
source* CHARACTER VARYING(100)
field CHARACTER VARYING(100)
start_offset INTEGER
end_offset INTEGER
confidence NUMERIC(6,3)
method CHARACTER VARYING(100)
title CHARACTER VARYING(250)
link* CHARACTER VARYING(250)
sentiment NUMERIC(6,3)
paths_classification CHARACTER VARYING(250)
item_similarity
PK id* SERIAL
FK fk_sitem_uri CHARACTER VARYING(512)
FK fk_titem_uri CHARACTER VARYING(512)
field CHARACTER VARYING(100)
field_no INTEGER
start_offset INTEGER
end_offset INTEGER
confidence NUMERIC(6,3)
method CHARACTER VARYING(40)
title CHARACTER VARYING(250)
sentiment NUMERIC(6,3)
usfd_id
item_id
item_topic
PK id* SERIAL
FK fk_item_uri CHARACTER VARYING(512)
FK fk_topic_id CHARACTER VARYING(250)
fk_topic_id
id
topic
PK id* CHARACTER VARYING(250)
fk_parent_topic_ids CHARACTER VARYING(250) ARRAY
uri CHARACTER VARYING(250)
dc_title* CHARACTER VARYING(250)
id
map_point
PK id* SERIAL
FK item_id CHARACTER VARYING(250)
dc_title CHARACTER VARYING(1024)
FK parent_id CHARACTER VARYING(250)
dc_type CHARACTER VARYING(250)
parent_id
id
id
parent_id
topic_id
map_poly
PK id* SERIAL
dc_title CHARACTER VARYING(250)
FK topic_id CHARACTER VARYING(250)
FK parent_id CHARACTER VARYING(250)
dc_type CHARACTER VARYING(250)
m_order INTEGER

Figure 2: E-R diagram showing the entities, attributes and relationships of the data model

5.2 Data processing

The “interesting” part of the data processing takes place in Work Package 2 and is illustrated in a simplified way consist of the following steps:

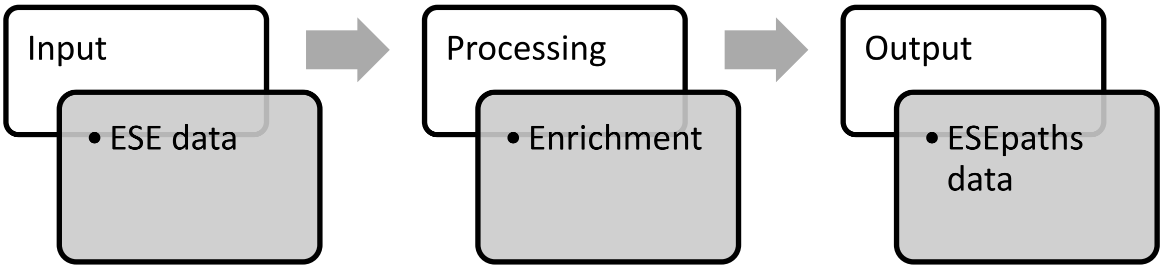


Figure 3: Simplified view of the PATHS enrichment process (for details, see D2.3)

The ESEpaths data are delivered as a number of different XML-files that must subsequently be processed into DDL scripts suitable for loading into the relational database. Several of the ESEpaths files are around a Gigabyte in size.

5.3 Data definition and loading

This section describes the process of transforming ESEpaths XML data into SQL and loading it into the database.

Items

Items are the base information type in the PATHS system and are derived from Europeana as ESE and then further enriched into ESEpaths in D2.3. A typical ESE record may look like the XML-fragment below:

```
<record>
   <dc:identifier>http://www.beamishcollections.com/collections/display.asp?
   ItemID=1</dc:identifier>
   <europeana:uri>http://www.europeana.eu/resolve/record/09405/8BBFE1B9EC70E
   EA34651852DD27A3C0F2532624C</europeana:uri>
   <dc:title>Enamel Advertisement</dc:title>
   <dc:source>Beamish Treasures</dc:source>
   <dc:description>Enamel Advertisement "Spillers Balanced Rations and
   UVECO"/ "For Cattle, Sheep, Pigs&amp; Poultry"/ "We Sell Them" Height:
   1280mm x 795mm.</dc:description>
   <dcterms:isPartOf>Beamish Treasures</dcterms:isPartOf>
   <dc:subject>Advertising</dc:subject>
   <dc:subject>Enamels</dc:subject>
   <dc:type>Image</dc:type>
   <europeana:object>http://www.peoplesnetwork.gov.uk/dpp/resource/2060233/s
   tream/thumbnail_image_jpeg</europeana:object>
   <europeana:provider>CultureGrid</europeana:provider>
   <europeana:isShownAt>http://www.beamishcollections.com/collections/displa
   y.asp?ItemID=1</europeana:isShownAt>
   <europeana:hasObject>true</europeana:hasObject>
   <europeana:country>uk</europeana:country>
   <europeana:type>IMAGE</europeana:type>
   <europeana:language>en</europeana:language>
</record>
```

Background links

Part of the enrichment process is to match ESE content to related articles in Wikipedia. This yields between 8 and 15 background links on average for each ESE item.

A typical ESEpaths background link record may look like the XML-fragment below:

```
<?xml version="1.0" encoding="UTF-8"?>
<metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:paths="http://www.paths-project.eu">
   <record>
      <dc:identifier>http://www.beamishcollections.com/collections/display.a
      sp?ItemID=1</dc:identifier>
      <paths:background_link source="wikipedia" start_offset="0"
      end_offset="6" field="dc:title" field_no="0" confidence="0.021"
      method="wikipedia-miner-1.2.0" title="Enamel
      paint">http://en.wikipedia.org/wiki/Enamel%20paint</paths:background_l
      ink>
      <paths:background_link source="wikipedia" start_offset="0"
      end_offset="6" field="dc:description" field_no="0" confidence="0.017"
      method="wikipedia-miner-1.2.0" title="Vitreous
      enamel">http://en.wikipedia.org/wiki/Vitreous%20enamel</paths:backgrou
      nd_link>
   </record>
</metadata>
```

Intra-links

Another step of the content enrichment process is to establish internal relationships between items created from ESE content. This also yields around 10 links on average per processed ESE record something which together with the background links contributes to a significant growth of data, ranging towards 50 million records. This poses a challenge to the built-in indexing system of the relational database in terms of performance and capacity.

A typical ESEpaths intra-link may look like the XML-fragment on the following page:

```
<?xml version="1.0" encoding="UTF-8"?>
<metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:paths="http://www.paths-project.eu">
   <record>
          <dc:identifier>http://viewfinder.english-
          heritage.org.uk/search/detail.asp?calledFrom=oai&amp;imageUID=1247
          59</dc:identifier>
          <paths:related_item confidence="0.912582" method="lda-
          vector">http://viewfinder.english-
          heritage.org.uk/search/detail.asp?calledFrom=oai&amp;imageUID=1241
          07</paths:related_item>
          <paths:related_item confidence="0.892687" method="lda-
          vector">http://viewfinder.english-
          heritage.org.uk/search/detail.asp?calledFrom=oai&amp;imageUID=1240
          67</paths:related_item>
   </record>
</metadata>
```

Keywords

A third content enrichment process seeks to extract the most important keywords from the content for the purpose of powering the tag-cloud based navigation paradigm of the end-user interface. This yields a vast number of key-words per item. These are stored as comma separated values in a field on the item table to avoid costly sub-queries when retrieving the info. Instead, the field is split when generating the inverted index in Solr so that queries may be made against specific keywords.

A typical ESEpaths keyword record may look like the XML-fragment below:

```
<?xml version="1.0" encoding="UTF-8"?>
<metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:paths="http://www.paths-project.eu">
   <record>
          <dc:identifier>http://www.kirkleesimages.org.uk/frontend.php?keywo
          rds=Ref_No_increment;EQUALS;k014611&amp;pos=2&amp;action=zoom</dc:
          identifier>
            <paths:event source="wordnet">motorbike</paths:event>
            <paths:event source="wordnet">collapse</paths:event>
   </record>
</metadata>
```

Dates

In order to establish unified facets that may be exploited for content exploration, the content enrichment process also identifies date values contained within the content and reformats it to a unified date format across the collections. Where possible, these values are extracted and written to ESEpaths.

A typical ESEpaths date facet file may look like the XML-fragment below:

```
<?xml version="1.0" encoding="UTF-8"?>
<metadata xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:paths="http://www.paths-project.eu">
   <record>
          <dc:identifier>http://viewfinder.english-
          heritage.org.uk/search/detail.asp?calledFrom=oai&amp;imageUID=18</
          dc:identifier>
          <paths:normalized_date>[??:1/1/1903:??.??:??]</paths:normalized_da
          te>
          <paths:normalized_date>[??:31/12/1903:??.??:??]</paths:normalized_
          date>
    </record>
</metadata>
```

Topics

One of the new features of the second prototype is the thesaurus navigation paradigm. This feature is based on content being associated with a multi-hierarchical concept hierarchy. This information is not provided as ESEpaths but rather follows a simple tabulator-separated format suitable for parsing and loading into the database.

A typical line from a topic file formatted as <item.uri><tab><topic.id>:

```
http://viewfinder.english-
heritage.org.uk/search/detail.asp?calledFrom=oai&imageUID=46668<tab>513701e3
abf1e109fd6ce786
```

Spatial data

Another new function in the second prototype is the map based navigation. This is not maps as in geography – but maps representing semantic space. This information is stored as PostGIS Point and Polygon geometries that conform to the Open Geospatial Consortium's Simple Feature Specification (SFS). This standard defines both textual and binary representations of geometry.

The SFS Well-Known Text (WKT) format for a point-geometry may look like this:

```
POINT (1 2)
```

The SFS Well-Known Text format for a polygon-geometry may look like this:

```
POLYGON ((1 1, 1 2, 2 2, 2 1))
```

Both of the data types are stored in binary form in the database. A geometry field may be accessed, modified and queried using a wide range of geo-processing instructions as defined by PostGIS.

PATHS data loader

To facilitate the transformation of the XML files into SQL suitable for loading into the PATHS data model, a simple "PATHS Data Loader" application has been developed.

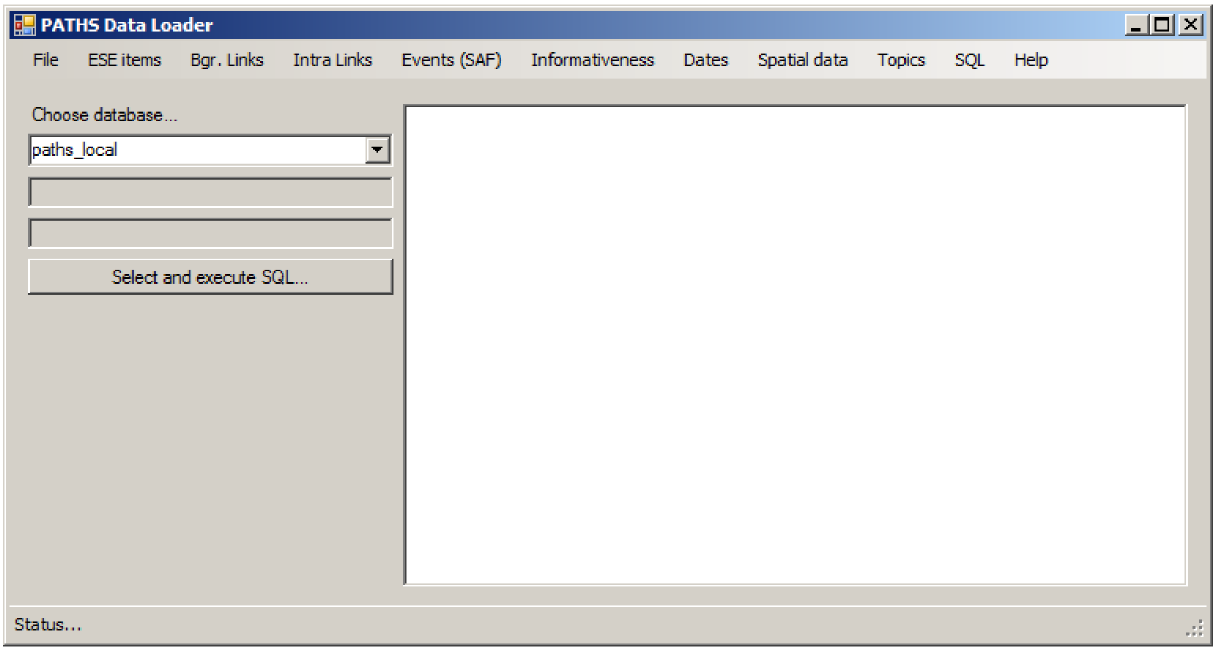


Figure 4: Main application window of PATHS Data Loader

The application has a menu strip with entries for each of the data types. Each entry has a one or more sub-entries that permit the user to parse the data and subsequently load it into the database in transaction batches of between 1,000 and 50,000 records of SQL statement per commit depending on the complexity of the data loading or updating task.

The application also has functions to drop and reload indices and constraints to facilitate quicker bulk loading of data.

5.4 Additional indexing using Solr

In addition to the built-in indexing mechanisms of the PostgreSQL relational database management system, an inverted index is generated on top of the information types: item, path and node. This is used for a number of complex and costly queries that would otherwise not be possible to conduct with an acceptable level of performance.

These queries are issued against a proxy Web Service that forwards permissible requests to a Solr SELECT end-point.

6 Application Layer

This section describes the PATHS platform operating system, server side application environment as well as version 2.0 of the PATHS Web Service API. The API will go live and be accessible on the URL below on the 8th of May 2013

↙ URL for PATHS Web Service API: http://api2.paths-project-eu/

6.1 Operating system

The PATHS system is hosted on a Windows Server. The Web Service API is implemented in C#.NET that requires the .NET framework or Mono.

The prototype user interface and the enrichment services have been developed based on Linux based servers using technologies such as Python, Perl and Java. All services will however run from a single server after an initial operational period during which minor bug-fixes and enhancements are anticipated.

6.2 Server components

The platform includes the following server components that form the foundation for the various Web Service API methods and properties.

Table 3: Overview of server components

Category	Component	Description
Web server	IIS	Used to deliver the Web Service API, web server running on port 80
	Apache Tomcat	Used to deliver Java servlets to the platform, mainly powering Solr but also the platform on which the recommender service is built.
	Python	Used to serve the user interface. Runs as Windows Service within dedicated local web server.
Database, data store	PostgreSQL	Main data store
	PostGIS	Spatial extension for main data store
	Virtuoso	Triple-store for resolving queries on graph databases
Language runtimes	.NET 4.0 (C#)	Development language for Web Service API
	Python	Server-side development language for user interface. All client code delivered as HTML and JavaScript.
	PERL	Development language for prototype content enrichment service.
	Java	Development language for recommender

Category	Component	Description
		engine.
Source control	Visual SVN	Concurrent versioning system used during development and integration of PATHS code.

6.3 API Web Services

The API consists of a set of web services each of which expose a number of methods. Each web service is structured around the principal object types in the data layer, logically grouping together related functionality.

Comprehensive documentation for the API is included in Appendix C – PATHS Web Service API reference and only a brief description of each of the services is included in this section.

Usr

The Usr web service contains methods for authenticating users, creating and modifying users, logging user behavior and issuing reminder e-mails upon forgetting passwords. The service is fundamental to web services which require authentication.

Path, Node

The Path web service contains methods for creation, editing and deletion of paths and path nodes. Furthermore, it has functions to transfer work space items to nodes in a path and to query paths and nodes. Paths and nodes are the core dynamic objects in the PATHS Web Service API. A path consist of one or more nodes, a node references an item (or another object) via a URI.

Workspace

The Workspace web service contains methods for creating, managing, querying and deleting workspace items. A workspace item can be considered a node which has not yet been completed and/or assigned to a Path. Workspace items can refer to any object identifiable by a URI and most commonly references records from the Items table.

Comment, Tag, Rating

The web service Social contains all functionality associated with user generated content which may be attached to paths, nodes and items. UGC elements are associated with resources via a URI and may in principle be attached to any web resource. This reduces the amount of tables required for the connections and simplifies the data management.

Item

The Item web service contains methods for querying and retrieving information about items. PATHS items are information derived from Europeana and Alinari and include most attributes defined by the Europeana Semantic Elements. Items have been enriched with (1) background links, (2) topic links and (3) item similarity links.

Topic

The topic web service contains methods for querying, traversing and interacting with multi-hierarchies of concept labels as well as their related path, node and item objects.

SolrProxy

The SolrProxy web service is, as the name suggests, merely a transparent proxy that allows secure access to a Solr search server end-point. Only the select method is permitted in order to prevent 3rd parties form modifying the index.

6.4 Examples API calls

This section shows examples of how the PATHS Web API may be invoked to audit its functionality and return data. This section is of a technical instructive nature and is with minor modifications identical to the one included in D3.2. The reason for repeating it here is to provide a stand-alone document by which developers can get a quick introduction to the system.

HTTP Header of Post Request

By default, these web services will return the response JSON wrapped in an XML element named "string". The encoding will be UTF-8. To get pure JSON, the Content-Type parameter is passed as part of the HTTP/POST request:

```
Content-Type: application/json; charset=utf-8
```

Users invoking the methods of the PATHS Web API are likely to use a cross-browser AJAX/HTTP library like jQuery. Such libraries enable developers to specify the format of the return data type as shown above.

JQuery.ajax request

```
.ajax({
   type: 'POST',
   url: '/Usr.asmx/CreateUser',
   data: “{
        ‘cognitiveStyle':'1',
        'usr':'user',
        'foaf_nick':'Nick Name',
        'pwd':'password’,
        'email':’user@domain.tld',
        'openid':'true'}",
   contentType: "application/json; charset=utf-8",
   dataType: ‘json’,
   success: done,
   error: cstatus
);
```

The JSON result of any web service request will be wrapped in an additional top-level object "d". Take this into account when parsing the response. This is a security feature of the .NET Framework.

On the next level of the object, the value "code" states whether the request was successful and the object data is an array of values.

Response JSON from Web Service Request

```
{
  "d":{
    "code":"2",
    "data":[
      {
        "id":"1",
        "fk_usr_id":"1",
        "fk_rel_uri":"http://www.bergheim.dk",
        "comment":"This is a third comment",
        "isdeleted":"0",
        "tstamp":"04/04/2012 23:56:21"
      }
    ]
  }
}
```

To return the value of "fk_rel_uri" in JavaScript, you would type

```
var uri = d.data[0].fk_rel_uri;
```

When a JSON result yields more than one return item, i.e. a result set from a query, items are accessible through a zero-based Array.

Response JSON from Web Service Request yielding more than one item

```
{
  "d":{
    "code":"2",
    "data":[
      {
        "id":"3",
        "fk_usr_id":"1",
        "fk_rel_uri":"http://www.bergheim.dk",
        "comment":"A comment",
        "isdeleted":"0",
        "tstamp":"04/04/2012 23:56:21"
      },
      {
        "id":"2",
        "fk_usr_id":"1",
        "fk_rel_uri":"http://www.bergheim.dk",
        "comment":"Another comment",
        "isdeleted":"0",
        "tstamp":"04/04/2012 23:56:21"
      },
      {
        "id":"1",
        "fk_usr_id":"1",
        "fk_rel_uri":"http://www.bergheim.dk",
        "comment":"A third comment",
        "isdeleted":"0",
        "tstamp":"04/04/2012 23:56:21"
      }
    ]
  }
}
```

An example of how to iterate through the array of comments contained in the JSON object is found below:

```
for (var i = 0; i < jsonData.d.data.length; i++) {
   var title = d.data[i].comment;
}
```

6.5 Service status codes

The following return codes are used for PATHS web services and can be used to validate the results.

```
NoSuchUser = -1
AuthenticationFailed = 1
OperationCompletedSuccessfully = 2
OperationFailed = 3
AuthenticationSucceeded = 4
OperationRequiresAuthentication = 5
LogoutSuccess = 6
DatabaseSQLError = 7
QueryDidNotReturnRecords = 8
FailedToCreateTemporaryUser = 9
SpecifiedObjectDoesNotExist = 10
NotImplementedYet = 99
```

Most of the service codes are self-explanatory. The latter one, 99, is used during development of new functionality but is not present in the published version of the API. All functions documented in the API, (see Appendix C – PATHS Web Service API), are fully implemented and operational.

6.6 Authentication

Most of the services require the user to be authenticated. Authentication is maintained between requests through a session cookie which is sent along with the HTTP-request from the Client application.

A call to the web service "Authenticate" with the credentials as parameters will set session variables letting other web services know that the user is authenticated - as well as store the usr_id for use in user profile related functions.

```
URI: http://development.paths-project.eu/Usr.asmx/Authenticate
```

Unless a cookie container is sent along with the web request, there is no mechanism to exchange session variables between requests to the Web Services; therefore, developers implementing applications on top of the API must take care to fit their HTTP requests with a cookie container.

7 Presentation Layer

This section describes the second prototype user interface. The user interface will go live and be accessible on the URL below on the 8th of May 2013

↙ URL for PATHS user interface: http://explorer.paths-project.eu/

7.1 Key improvements over D3.2

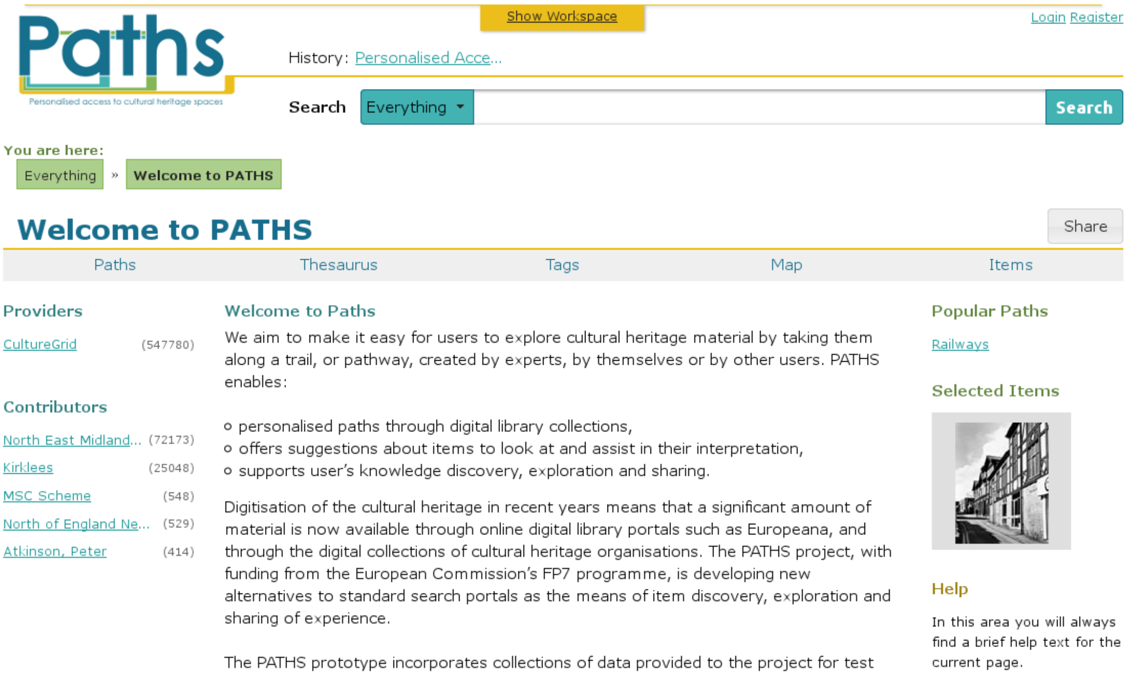
This section describes the major improvements to the user interface over the one that was available in D3.2, the first prototype.

Following the results of the user trials as well as constructive input from the two project reviews, a number of enhancements have been made to the first generation of the interface. Additionally, some modules, i.e. the recommender system and map-based exploration system, were not planned to be included until the second prototype.

7.1.1 Front page

The new front page provides the end-user with a context for understanding what the PATHS system is and what it can be used for. In the previous prototype, the user landed directly into a complex set of semantics that could not be assumed to be known in advance. An online instruction video shows users how they can get started with the system and progress to more advanced usage.

Additionally, the navigation framework has been modified to conform to common good practices with a “white-box” text search available at all times on the top of the page. A horizontal menu provides access to the various modes of exploration and information retrieval at all times.


Paths
Personalised access to cultural heritage spaces
Show Workspace
Login Register
History: Personalised Acce...
Search
Everything
Search
You are here:
Everything » Welcome to PATHS
Share
Paths
Thesaurus
Tags
Map
Items

Welcome to PATHS

Providers

CultureGrid	(547780)

Contributors

North East Midland...	(72173)
Kirklees	(25048)
MSC Scheme	(548)
North of England Ne...	(529)
Atkinson, Peter	(414)

Welcome to Paths

We aim to make it easy for users to explore cultural heritage material by taking them along a trail, or pathway, created by experts, by themselves or by other users. PATHS enables:

- personalised paths through digital library collections,
- offers suggestions about items to look at and assist in their interpretation,
- supports user’s knowledge discovery, exploration and sharing.

Digitisation of the cultural heritage in recent years means that a significant amount of material is now available through online digital library portals such as Europeana, and through the digital collections of cultural heritage organisations. The PATHS project, with funding from the European Commission’s FP7 programme, is developing new alternatives to standard search portals as the means of item discovery, exploration and sharing of experience.

The PATHS prototype incorporates collections of data provided to the project for test

Popular Paths

Railways

Selected Items

Help

In this area you will always find a brief help text for the current page.

Figure 5: The above screen shows the enhanced front-page

7.1.2 Paths and Nodes

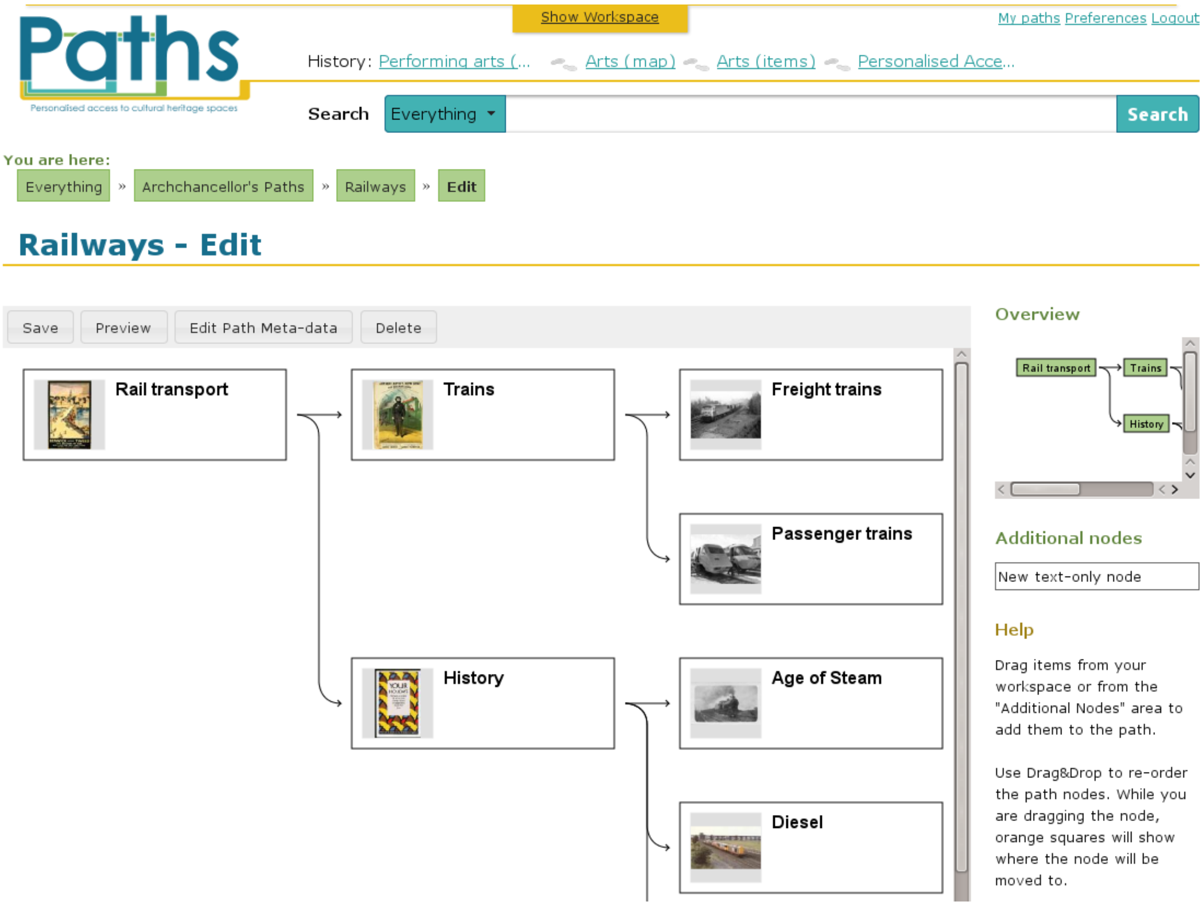
The PATHS system consists of two parallel volumes of data: (1) the paths system consisting of paths, nodes, related tags, comments and ratings and; (2) the enriched ESEpaths content with background links to Wikipedia, intra-links to other content items etc.

The two data volumes are connected through URI references from the dc_source field in the node table referencing the URI field on the item table. However, this reference could be to any object identifiable by a persistent URI – i.e. any web content – not necessarily only instances of item objects.

Enhanced PATH creation and editing

In the first prototype, paths were constrained to linear, non-branching, non-merging sequences of nodes. This was experienced as too limiting from the perspective of path authors who would like to express more complex graphs of interrelated content.

For this reason, the second prototype user interface introduces a set of new features that supports authoring of complex paths consisting of multiple branches.



Drag items from your workspace or from the "Additional Nodes" area to add them to the path.

Use Drag&Drop to re-order the path nodes. While you are dragging the node, orange squares will show where the node will be moved to.

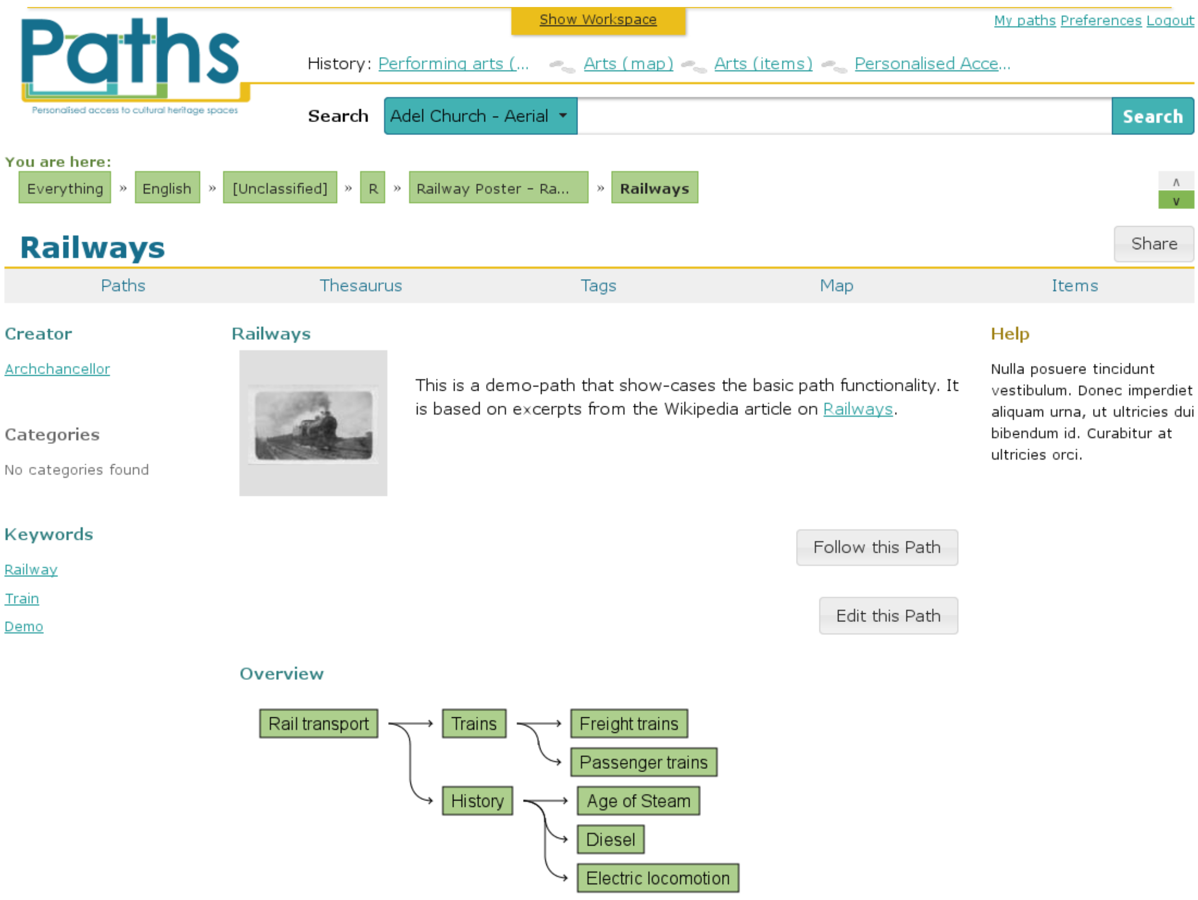
Figure 7: The above screen shows the enhanced front-page

7.1.3 Navigation

The other key area where major enhancements have been made to the system is with respect to navigation. The existing features have been redesigned to comply with input from end-users (collected during user trials) as well as professional assessments from the consortium and reviewers.

Enhanced PATH navigation

With more complex path structures, the second prototype also introduces a new and more user friendly visualization of the path itself, conveying in a more effective way the way forward, backward as well as the context of the currently displayed prototype.



Paths
Personalised access to cultural heritage spaces
Show Workspace
My paths Preferences Logout

History: Performing arts (... Arts (map) Arts (items) Personalised Acce...

Search Adel Church - Aerial Search

You are here: Everything » English » [Unclassified] » R » Railway Poster - Ra... » Railways

Railways

Share
Paths Thesaurus Tags Map Items

Creator

Archchancellor

Categories

No categories found

Keywords

Railway
Train
Demo

Railways

This is a demo-path that show-cases the basic path functionality. It is based on excerpts from the Wikipedia article on Railways.

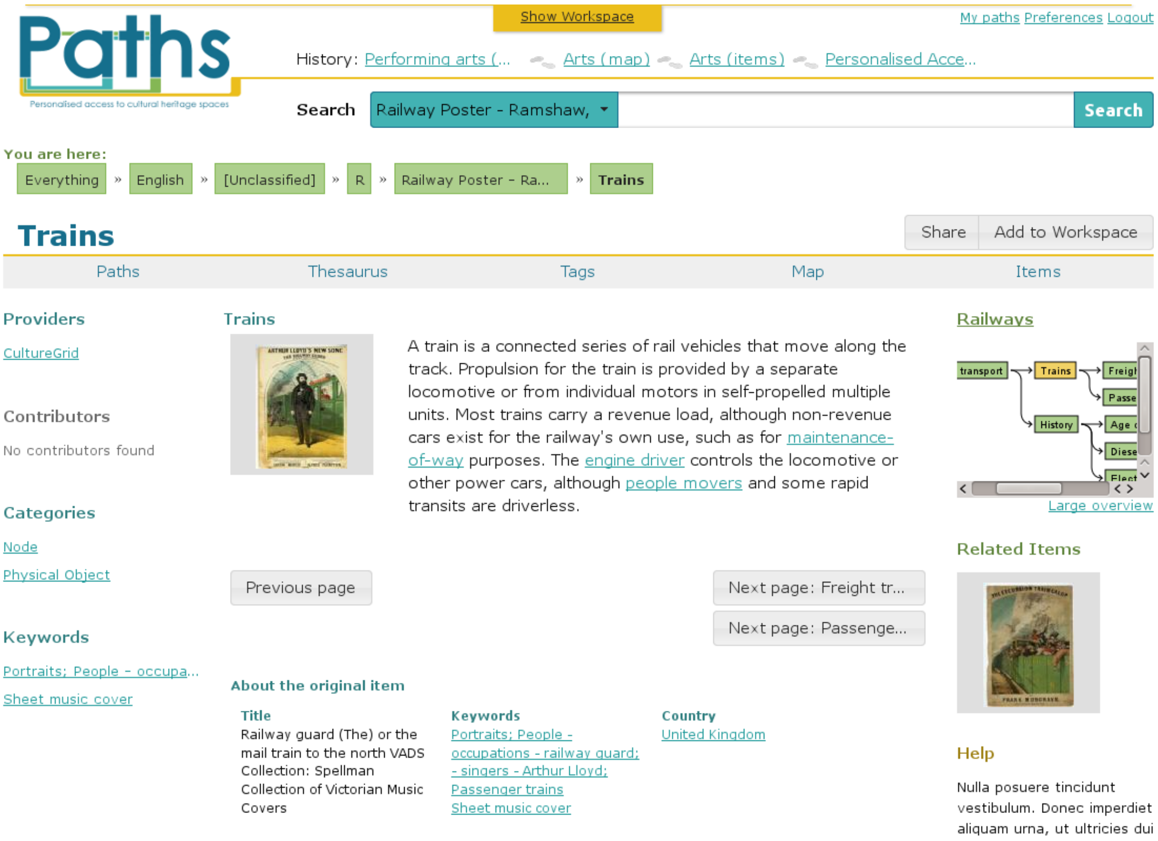
Overview

Rail transport → Trains → Freight trains, Passenger trains
Rail transport → History → Age of Steam, Diesel, Electric locomotion

Help

Nulla posuere tincidunt vestibulum. Donec imperdiet aliquam urna, ut ultricies dui bibendum id. Curabitur at ultricies orci.

Figure 8: The above screen shows the path overview with the graphical representation of the path's nodes


Paths
Personalised access to cultural heritage spaces
Show Workspace
My paths Preferences Logout
transport
Trains
Freigh
Passe
History
Age c
Diese
Elect

History: Performing arts (... Arts (map) Arts (items) Personalised Acce...

Search Railway Poster - Ramshaw, Search

You are here: Everything » English » [Unclassified] » R » Railway Poster - Ra... » Trains

Trains

Share Add to Workspace

Paths Thesaurus Tags Map Items

Providers

CultureGrid

Contributors

No contributors found

Categories

Node

Physical Object

Keywords

Portraits; People - occupa...

Sheet music cover

Trains

A train is a connected series of rail vehicles that move along the track. Propulsion for the train is provided by a separate locomotive or from individual motors in self-propelled multiple units. Most trains carry a revenue load, although non-revenue cars exist for the railway's own use, such as for maintenance-of-way purposes. The engine driver controls the locomotive or other power cars, although people movers and some rapid transits are driverless.

Previous page

Next page: Freight tr...

Next page: Passenge...

About the original item

Title
Railway guard (The) or the mail train to the north VADS Collection: Spellman Collection of Victorian Music Covers

Keywords
Portraits; People - occupations - railway guard; - singers - Arthur Lloyd; Passenger trains Sheet music cover

Country
United Kingdom

Railways

Large overview

Related Items

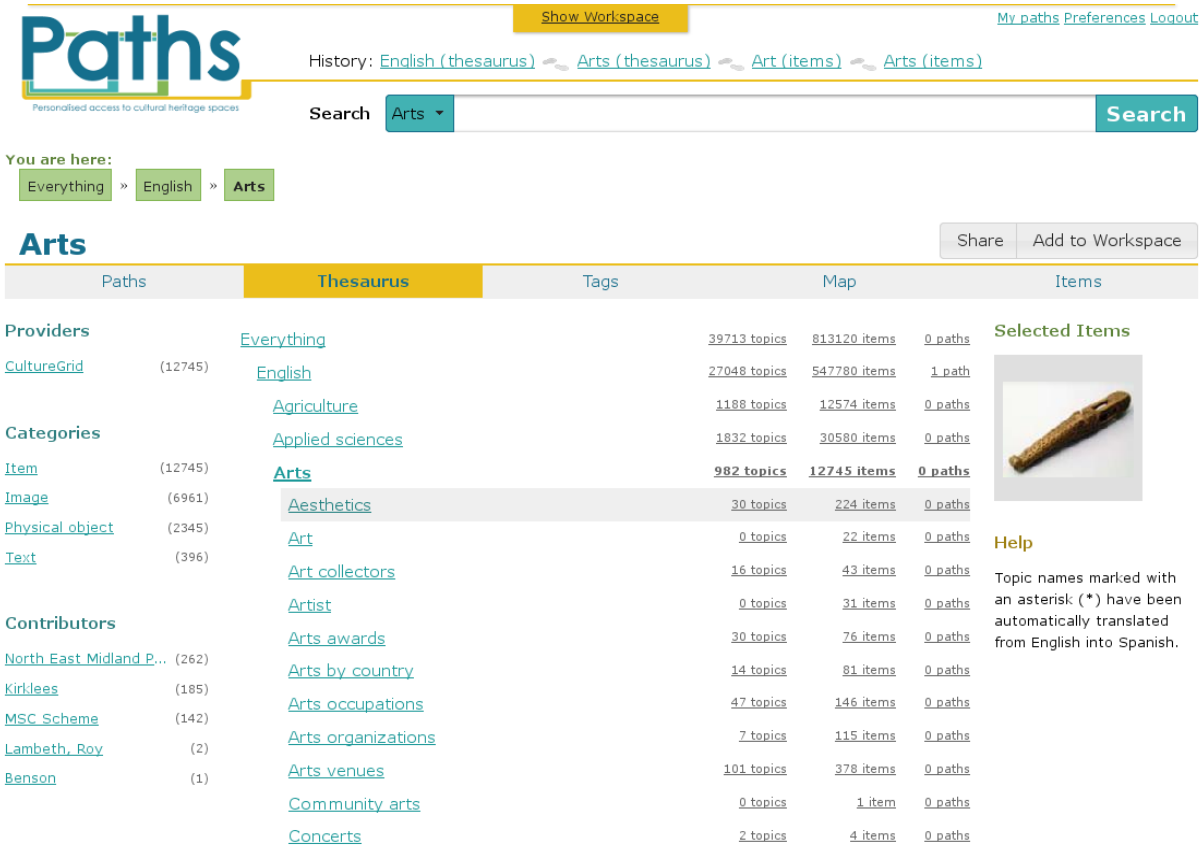
Help

Nulla posuere tincidunt vestibulum. Donec imperdiet aliquam urna, ut ultricies dui

Figure 9: The above screen shows a path node, with the visual path overview on the right-hand side

Thesaurus navigation

Through the enrichment process, ESEpaths content stored in the item table have been connected to a multi-hierarchical thesaurus. This relationship as well as the internal relationship between topics has been exploited to offer a thesaurus based exploration mode in the second prototype user interface. By incrementally browsing through the hierarchy, users can “drill down” into the greater levels of detail by narrowing the selection of content as the user expands the various sub-topics.


Paths
Personalised access to cultural heritage spaces
Show Workspace
My paths Preferences Logout
History: English (thesaurus) Arts (thesaurus) Art (items) Arts (items)
Search
Arts
Search
You are here:
Everything » English » Arts
Share
Add to Workspace
Paths
Thesaurus
Tags
Map
Items

Arts

Providers

CultureGrid	(12745)

Categories

Item	(12745)
Image	(6961)
Physical object	(2345)
Text	(396)

Contributors

North East Midland P...	(262)
Kirklees	(185)
MSC Scheme	(142)
Lambeth, Roy	(2)
Benson	(1)

Everything	39713 topics	813120 items	0 paths
English	27048 topics	547780 items	1 path
Agriculture	1188 topics	12574 items	0 paths
Applied sciences	1832 topics	30580 items	0 paths
Arts	**982 topics**	**12745 items**	**0 paths**
Aesthetics	30 topics	224 items	0 paths
Art	0 topics	22 items	0 paths
Art collectors	16 topics	43 items	0 paths
Artist	0 topics	31 items	0 paths
Arts awards	30 topics	76 items	0 paths
Arts by country	14 topics	81 items	0 paths
Arts occupations	47 topics	146 items	0 paths
Arts organizations	7 topics	115 items	0 paths
Arts venues	101 topics	378 items	0 paths
Community arts	0 topics	1 item	0 paths
Concerts	2 topics	4 items	0 paths

Selected Items

Help

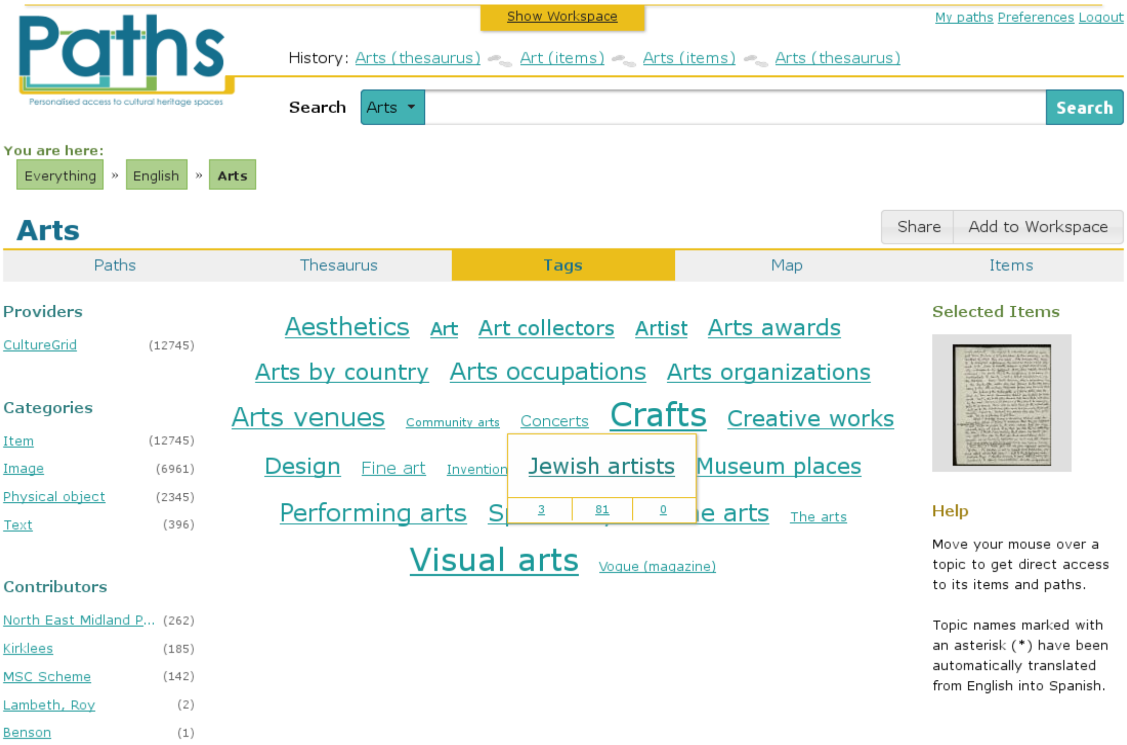
Topic names marked with an asterisk (*) have been automatically translated from English into Spanish.

Figure 10: Thesaurus based navigation

Tag cloud navigation

In the first prototype, the auto-generated tag cloud navigation feature was welcomed during the user trials but its efficiency as a means of navigating the content was complicated by the co-existence of Spanish and English terms in the underlying keyword data – as well as over-representation of a small number terms that on account of their high frequency were assigned too much weight.

In this generation of the system, the keywords are derived from the thesaurus, creating cleaner tag clouds and a closer integration between the three exploration visualisations (thesaurus, tag-cloud, map).



Providers

CultureGrid	(12745)

Categories

Item	(12745)
Image	(6961)
Physical object	(2345)
Text	(396)

Contributors

North East Midland P...	(262)
Kirklees	(185)
MSC Scheme	(142)
Lambeth, Roy	(2)
Benson	(1)

Help

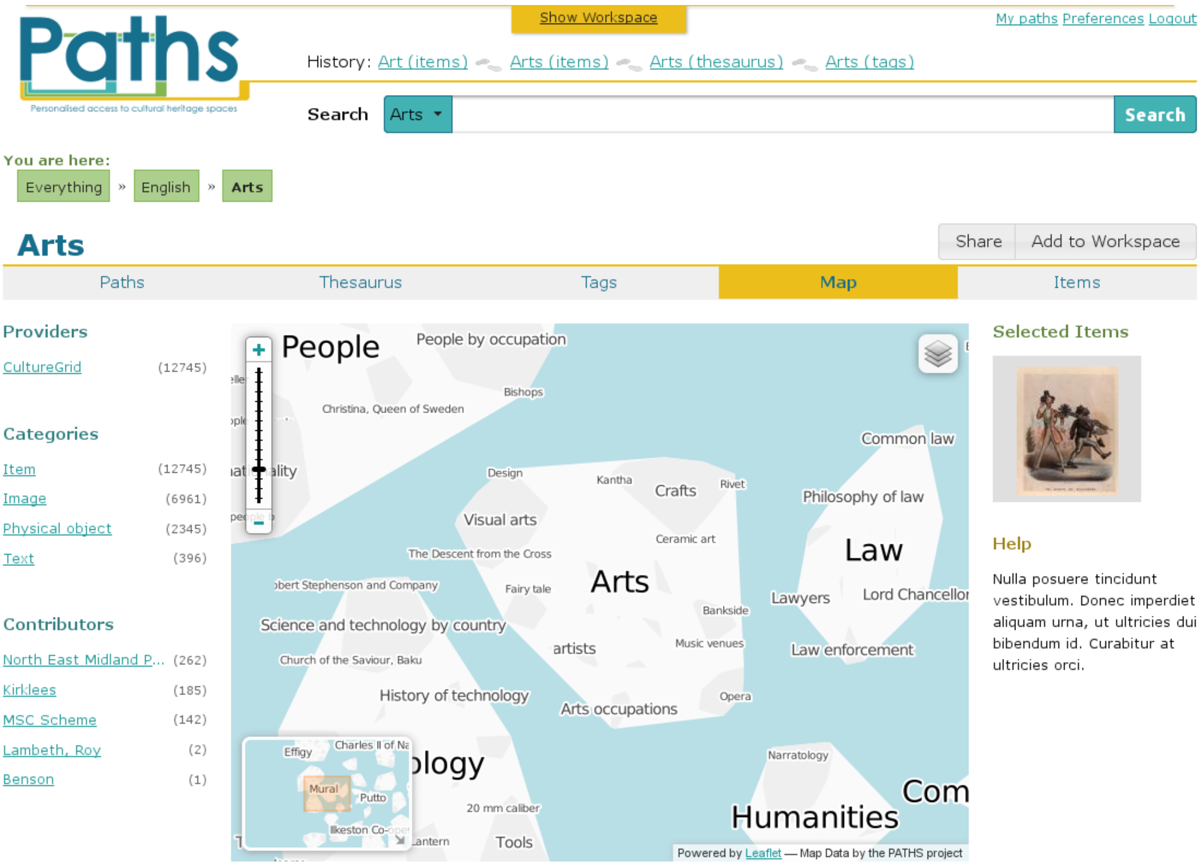
Move your mouse over a topic to get direct access to its items and paths.

Topic names marked with an asterisk (*) have been automatically translated from English into Spanish.

Figure 11: Tag cloud based navigation

Map based navigation

New, and perhaps also the most innovative feature of the second user interface is the map based navigation. Using a clustering of topics as a starting point, USFD has generated a 2D semantic space that is based on the hierarchical representation of the thesaurus. As you zoom further into the map, greater resolution of information appears, all the way down to the “leafs” of the thesaurus, including the individual items.



Providers

CultureGrid (12745)

Categories

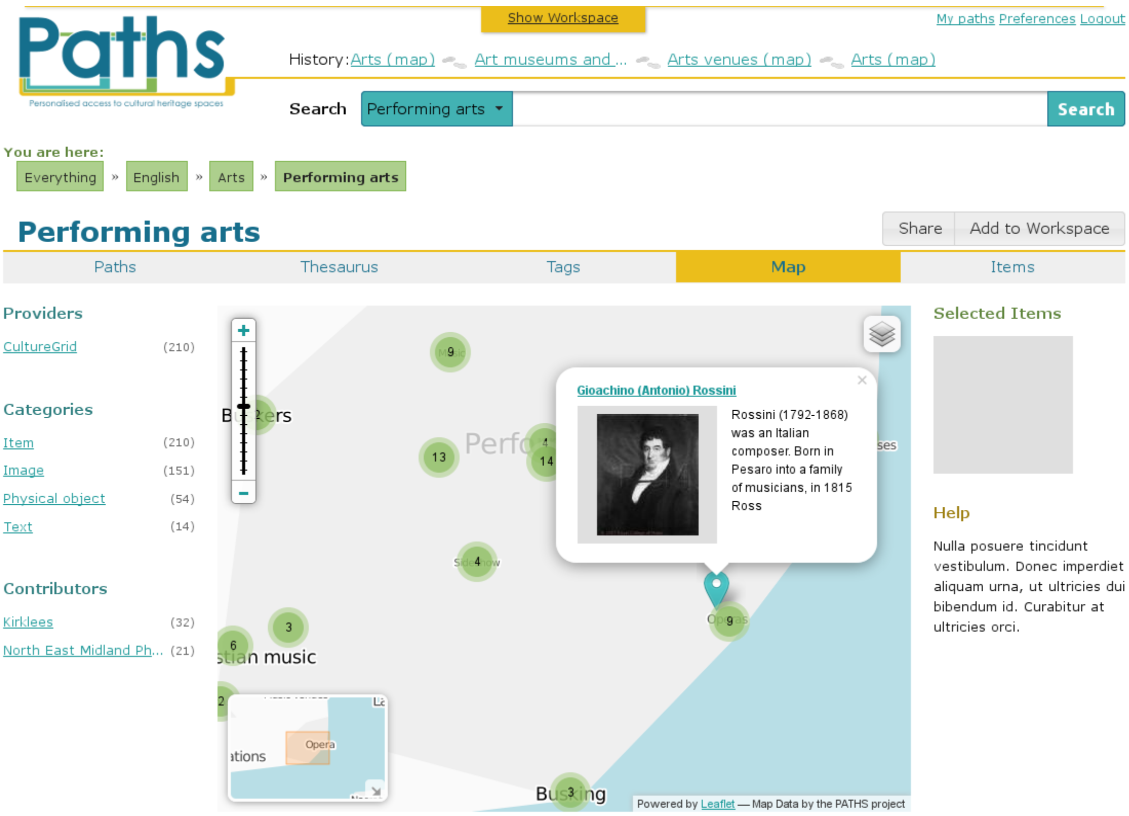
Item (12745)
Image (6961)
Physical object (2345)
Text (396)

Contributors

North East Midland P... (262)
Kirklees (185)
MSC Scheme (142)
Lambeth, Roy (2)
Benson (1)

Nulla posuere tincidunt vestibulum. Donec imperdiet aliquam urna, ut ultricies dui bibendum id. Curabitur at ultricies orci.

Figure 12: Map based navigation


Paths
Personalised access to cultural heritage spaces
Show Workspace
My paths Preferences Logout
History: Arts (map) Art museums and ... Arts venues (map) Arts (map)
Search Performing arts Search
You are here:
Everything » English » Arts » Performing arts
Performing arts
Share Add to Workspace
Paths Thesaurus Tags Map Items
9
Bakers
13
Perfo
4
14
4
3
6
stian music
2
Opera
ations
3
Busking
9
Powered by Leaflet — Map Data by the PATHS project
Selected Items
Help

Providers

CultureGrid (210)

Categories

Item (210)
Image (151)
Physical object (54)
Text (14)

Contributors

Kirklees (32)
North East Midland Ph... (21)

Gioachino (Antonio) Rossini

Rossini (1792-1868) was an Italian composer. Born in Pesaro into a family of musicians, in 1815 Ross

Nulla posuere tincidunt vestibulum. Donec imperdiet aliquam urna, ut ultricies dui bibendum id. Curabitur at ultricies orci.

Figure13: Map based navigation with an item selected

8 Experimental Features

This section describes experimental features included in the second prototype user interface and API. The experimental features will be accessible on the URL below from the 8th of May 2013.

↙ URL for experimental features lab: http://labs.paths-project.eu/

8.1 Recommender System

The recommender system takes as a starting point:

1. previous user behavior that is recorded through scanning of the log-files
2. incremental collection of present user behavior through built-in logging mechanisms in the data layer and API
3. commonly viewed content

Based on this information, it extracts relevant links including their titles and returns these to the requesting client.

The recommender system is implemented in Java and runs on Apache Tomcat. It is presently hosted from the "labs" platform and will be migrated after an initial test-run phase of 1-2 months during which the platform will be subject to intensive external testing and refinements.

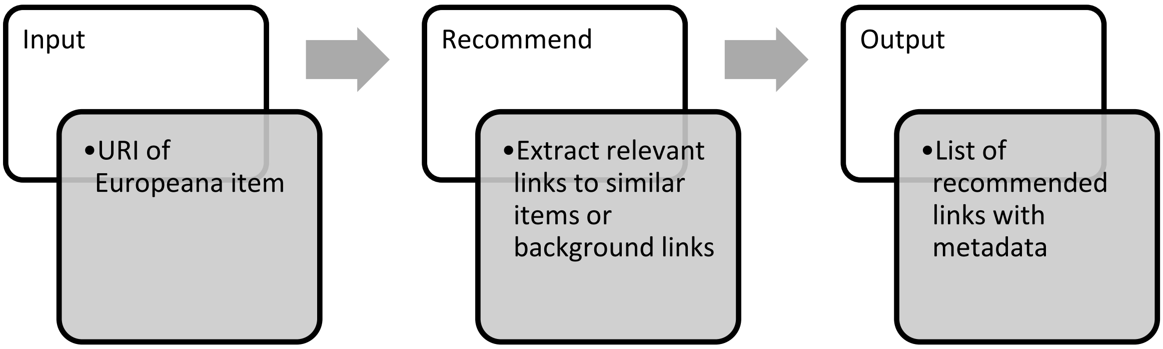

Input
•URI of Europeana item
Recommend
•Extract relevant links to similar items or background links
Output
•List of recommended links with metadata

Figure 14: Simplified internal workflow of the recommender system

8.2 Metadata Enrichment Service

The metadata enrichment service takes as a starting point a record of Europeana data structured as XML according to the Europeana Semantic Elements (ESE) application profile. In essence, this service does in real-time what the pre-processing in D2.3 does in batch for the entire volume of data.

This has a number of interesting applications, especially if seen in conjunction with the Europeana API where any ESE record may be retrieved.

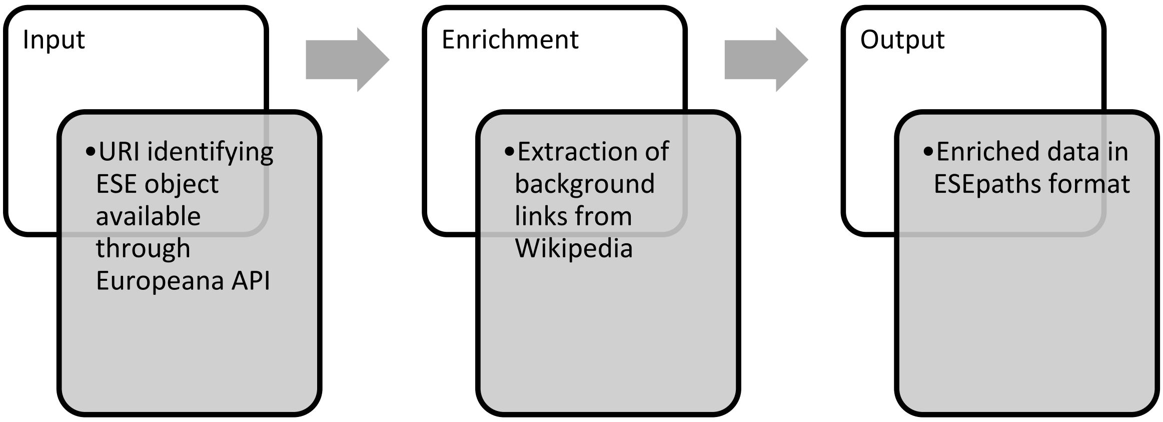

Input
•URI identifying ESE object available through Europeana API
Enrichment
•Extraction of background links from Wikipedia
Output
•Enriched data in ESEpaths format

Figure 15: Simplified internal workflow of the enrichment service

9 References

D1.1 User Requirements Analysis - Paula Goodale, Mark Hall, Kate Fernie, Phil Archer

D1.5 Functional Specification for second prototype - George Chrysochoidis, Phil Archer, Kostas Chandrinos, with Mark Hall, Paul Clough, Paula Goodale, Mark Stevenson, Eneko Agirre, Aitor Soroa, Iñaki Alegria, Jillian Griffiths, Kate Fernie

D2.2 Processing and representation of Content for the second prototype;- Eneko Agirre, Arantxa Otegi and ;Aitor Soroa with, Nikos Aletras, Constantinos Chandrinos, Samuel Fernando, Aitor Gonzalez-Agirre

D3.1 System architecture specification - Stein Runar Bergheim and Idar Thoresen Kvam with Phil Archer, Kate Fernie, Paul Clough, Tor Gunnar Øverli and Mark Stevenson

D3.2 First PATHS Prototype Stein Runar Bergheim, Mark Hall, Eneko Agirre, Aitor Soroa, Antonis Kukurikos, Kate Fernie, Tor Gunnar Øverli

D4.2 Final Prototype Interface Design - Mark Hall, Paula Goodale, with Paul Clough, Eneko Agirre, Kate Fernie, Jillian Griffiths, Mark Stevenson

D5.1 Evaluation of the first PATHS prototype - Jillian Griffiths, Paula Goodale, with Sam Minelli, Andrea de Pollo, Rodrigo Agerri, Aitor Soroa, Mark Hall, Stein Runar Bergheim, Konstantinos Chandrinos, George Chryssochoidis, Kate Fernie, Tom Usher

Appendices

Appendix A – Acronym List and Glossary

Term	Description
API	Application Programming Interface
HTML	Hyper-Text Mark-up Language
HTTP	Hyper-Text Transfer Protocol
IP	Internet Protocol
JavaScript	See: ECMA Script
JDBC	Java DataBase Connectivity
JSON	JavaScript Object Notation
KML	Keyhole Mark-up Language
ODBC	Open DataBase Connectivity
OGC	Open Geospatial Consortium
OMG	Object Modelling Group
RDBMS	Relational Database Management System
REST	REpresentational State Transfer
SDLC	System Development Life Cycle
SMB	Server Message Block. A protocol for file sharing on Windows and Unix based systems
SOA	Service-Oriented Architecture
SPARQL	Simple Protocol And RDF Query Language
SQL	Structured Query Language
TCP	Transmission Control Protocol
UML	Unified Modelling Language
WFS	Web Feature Server. A protocol for on-the-fly generation of map images using http requests.
WMS	Web Map Server. A protocol for query and download of vector maps using http requests.
WP	Work Package
WS	Web Service
WSDL	Web Service Description Language
XML	eXtensible Mark-up Language
SFS	Simple Features Specification

CVS	Concurrent Versioning System
WAI	Web Accessibility Initiative
WCAG	Web Content Accessibility Guidelines
JSON	JavaScript Object Notation

Appendix B, C and D are independent documents attached to this report in the order specified below. Each document is separated by a cover page stating the identity of the document. Page numbering is individual to each document.

Appendix B – PATHS data model documentation

Appendix C – PATHS Web Service API reference

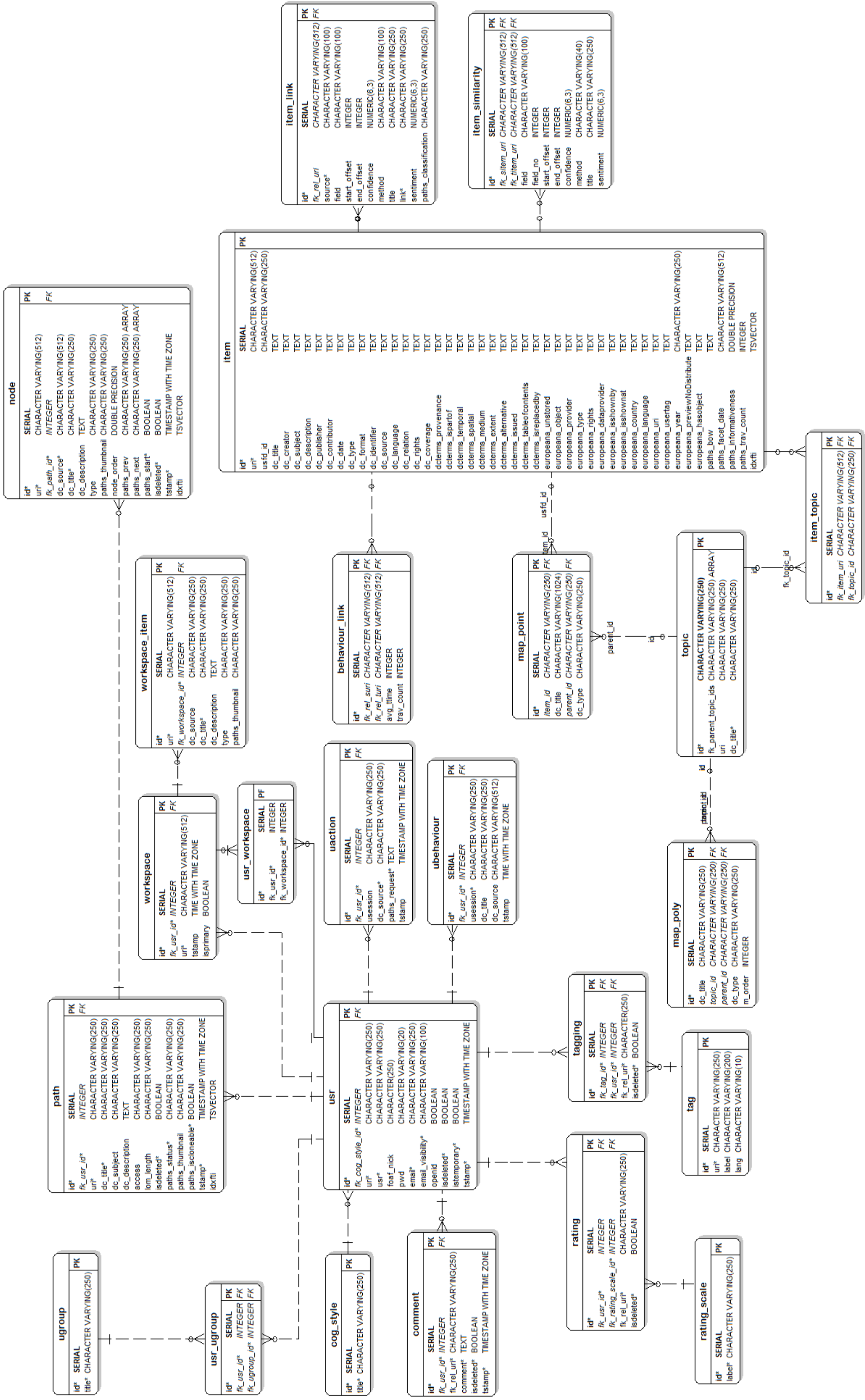
Appendix D – RedMine ticket example

Appendix B: PATHS Data Layer

Project details

Project name	Appendix B: PATHS Data Layer
Description	Deliverable: - (v1) D 3.2 First Prototype and Documentation - (v2) D 3.3 First Prototype and Documentation Contributors: Mark Hall (USFD) Kate Fernie (MDR Partners) Eneko Agirre, Aitor Soroa (UPV/EHU) Antonis Kukurikos (iSIEVE) Alok Singh, Tor Gunnar Øverli, Idar Thoresen Kvam (AVINET)
Author	(Stein) Runar Bergheim, Asplan Viak Internet A/S (Ed.)
Copyright	ICT-2009-270082 - PATHS - Personalised Access To cultural Heritage Spaces
Target DBMS	PostgreSQL 9.3
Created	2012-03-11
Modified	2013-05-03

ER diagram



item_link
id* SERIAL PK
fk_rel_uri CHARACTER VARYING(512) FK
source* CHARACTER VARYING(100)
field CHARACTER VARYING(100)
start_offset INTEGER
end_offset INTEGER
confidence NUMERIC(6,3)
method CHARACTER VARYING(100)
title CHARACTER VARYING(250)
link* CHARACTER VARYING(250)
sentiment NUMERIC(6,3)
paths_classification CHARACTER VARYING(250)
item_similarity
id* SERIAL PK
fk_sitem_uri CHARACTER VARYING(512) FK
fk_titem_uri CHARACTER VARYING(512) FK
field CHARACTER VARYING(100)
field_no INTEGER
start_offset INTEGER
end_offset INTEGER
confidence NUMERIC(6,3)
method CHARACTER VARYING(40)
title CHARACTER VARYING(250)
sentiment NUMERIC(6,3)
node
id* SERIAL PK
uri* CHARACTER VARYING(512)
fk_path_id* INTEGER FK
dc_source* CHARACTER VARYING(512)
dc_title* CHARACTER VARYING(250)
dc_description TEXT
type CHARACTER VARYING(250)
paths_thumbnail CHARACTER VARYING(250)
node_order DOUBLE PRECISION
paths_prev CHARACTER VARYING(250) ARRAY
paths_next CHARACTER VARYING(250) ARRAY
paths_start* BOOLEAN
isdeleted* BOOLEAN
tstamp* TIMESTAMP WITH TIME ZONE
idxfti TSVECTOR
item
id* SERIAL PK
uri* CHARACTER VARYING(512)
usfd_id CHARACTER VARYING(250)
dc_title TEXT
dc_creator TEXT
dc_subject TEXT
dc_description TEXT
dc_publisher TEXT
dc_contributor TEXT
dc_date TEXT
dc_type TEXT
dc_format TEXT
dc_identifier TEXT
dc_source TEXT
dc_language TEXT
dc_relation TEXT
dc_rights TEXT
dc_coverage TEXT
dcterms_provenance TEXT
dcterms_ispartof TEXT
dcterms_temporal TEXT
dcterms_spatial TEXT
dcterms_medium TEXT
dcterms_extent TEXT
dcterms_alternative TEXT
dcterms_issued TEXT
dcterms_tableofcontents TEXT
dcterms_isreplacedby TEXT
europeana_unstored TEXT
europeana_object TEXT
europeana_provider TEXT
europeana_type TEXT
europeana_rights TEXT
europeana_dataprovider TEXT
europeana_isshownby TEXT
europeana_isshownat TEXT
europeana_country TEXT
europeana_language TEXT
europeana_uri TEXT
europeana_usertag TEXT
europeana_year CHARACTER VARYING(250)
europeana_previewNoDistribute TEXT
europeana_hasobject TEXT
paths_bow TEXT
paths_facet_date CHARACTER VARYING(512)
paths_informativeness DOUBLE PRECISION
paths_trav_count INTEGER
idxfti TSVECTOR
item_topic
id* SERIAL PK
fk_item_uri CHARACTER VARYING(512) FK
fk_topic_id CHARACTER VARYING(250) FK
workspace_item
id* SERIAL PK
uri* CHARACTER VARYING(512)
fk_workspace_id* INTEGER FK
dc_source CHARACTER VARYING(250)
dc_title* CHARACTER VARYING(250)
dc_description TEXT
type CHARACTER VARYING(250)
paths_thumbnail CHARACTER VARYING(250)
behaviour_link
id* SERIAL PK
fk_rel_suri CHARACTER VARYING(512) FK
fk_rel_turi CHARACTER VARYING(512) FK
avg_time INTEGER
trav_count INTEGER
map_point
id* SERIAL PK
item_id CHARACTER VARYING(250) FK
dc_title CHARACTER VARYING(1024)
parent_id CHARACTER VARYING(250) FK
dc_type CHARACTER VARYING(250)
topic
id* CHARACTER VARYING(250) PK
fk_parent_topic_ids CHARACTER VARYING(250) ARRAY
uri CHARACTER VARYING(250)
dc_title* CHARACTER VARYING(250)
workspace
id* SERIAL PK
fk_usr_id* INTEGER FK
uri* CHARACTER VARYING(512)
tstamp TIME WITH TIME ZONE
isprimary BOOLEAN
usr_workspace
id* SERIAL PF
fk_usr_id* INTEGER
fk_workspace_id* INTEGER
uaction
id* SERIAL PK
fk_usr_id* INTEGER FK
usession CHARACTER VARYING(250)
dc_source* CHARACTER VARYING(250)
paths_request* TEXT
tstamp TIMESTAMP WITH TIME ZONE
ubehaviour
id* SERIAL PK
fk_usr_id* INTEGER FK
usession* CHARACTER VARYING(250)
dc_title CHARACTER VARYING(250)
dc_source CHARACTER VARYING(512)
tstamp TIME WITH TIME ZONE
map_poly
id* SERIAL PK
dc_title CHARACTER VARYING(250)
topic_id CHARACTER VARYING(250) FK
parent_id CHARACTER VARYING(250) FK
dc_type CHARACTER VARYING(250)
m_order INTEGER
path
id* SERIAL PK
fk_usr_id* INTEGER FK
uri* CHARACTER VARYING(250)
dc_title* CHARACTER VARYING(250)
dc_subject CHARACTER VARYING(250)
dc_description TEXT
access CHARACTER VARYING(250)
lom_length CHARACTER VARYING(250)
isdeleted* BOOLEAN
paths_status* CHARACTER VARYING(250)
paths_thumbnail CHARACTER VARYING(250)
paths_iscloneable* BOOLEAN
tstamp* TIMESTAMP WITH TIME ZONE
idxfti TSVECTOR
usr
id* SERIAL PK
fk_cog_style_id* INTEGER FK
uri* CHARACTER VARYING(250)
usr* CHARACTER VARYING(250)
foaf_nick CHARACTER(250)
pwd CHARACTER VARYING(20)
email* CHARACTER VARYING(250)
email_visibility* CHARACTER VARYING(100)
openid BOOLEAN
isdeleted* BOOLEAN
istemporary* BOOLEAN
tstamp* TIMESTAMP WITH TIME ZONE
tagging
id* SERIAL PK
fk_tag_id* INTEGER FK
fk_usr_id* INTEGER FK
fk_rel_uri* CHARACTER(250)
isdeleted* BOOLEAN
tag
id* SERIAL PK
uri* CHARACTER VARYING(250)
label CHARACTER VARYING(200)
lang CHARACTER VARYING(10)
rating
id* SERIAL PK
fk_usr_id* INTEGER FK
fk_rating_scale_id* INTEGER FK
fk_rel_uri* CHARACTER VARYING(250)
isdeleted* BOOLEAN
rating_scale
id* SERIAL PK
label* CHARACTER VARYING(250)
ugroup
id* SERIAL PK
title* CHARACTER VARYING(250)
usr_ugroup
id* SERIAL PK
fk_usr_id* INTEGER FK
fk_ugroup_id* INTEGER FK
cog_style
id* SERIAL PK
title* CHARACTER VARYING(250)
comment
id* SERIAL PK
fk_usr_id* INTEGER FK
fk_rel_uri* CHARACTER VARYING(250)
comment* TEXT
isdeleted* BOOLEAN
tstamp* TIMESTAMP WITH TIME ZONE
usfd_id
item_id
parent_id
id
fk_topic_id
topic_id

List of entities

Entity name	Primary key attributes	# Attributes	Description
behaviour_link	id	5	Information on which Items a user has traversed between.
cog_style	id	2	Codelist of different cognitive styles. A user may have one cognitive style.
comment	id	6	Comments added to objects identifiable by a URI
item	id	48	Information on resources imported from Alinari and Europeana, corresponding to the Europeana Semantic Elements specification.
item_link	id	12	Links between Items and external background resources (e.g. Wikipedia) as derived from semantic processing.
item_similarity	id	11	Information on similarity between Items as derived from semantic processing.
item_topic	id	3	Many to many table between item and topic. One topic may have many items, one item may have many topics.
map_point	id	5	This table also contains the column "geom" that holds WKT POINT geometries. This is added using the PostGIS AddGeomColumn function after creation of the table.
map_poly	id	6	This table also contains the column "geom" that holds WKT POLYGON geometries. This is added using the PostGIS AddGeomColumn function after creation of the table.
node	id	15	Information about path nodes such as title, description, node_order etc.
path	id	14	Information about paths such as title, subject, description etc.
rating	id	5	Assigns a rating to any resource identifiable by a URI. Rating is linked to a rating scale and a user. A user is only allowed to rate a URI resource once.
rating_scale	id	2	Rating scale for paths and other resources identifiable by a URI. 1 = dislikes, 2 = likes.
tag	id	4	Tags: keywords and keyphrases assigned to URI resources. Tags may be language specific and are identifiable by a URI.

tagging	id	5	Association between tags, users and resources identifiable by a URI. A user can only add the same keyword to a resource once.
topic	id	4	Information about topic hierarchies
uaction	id	6	
ubehaviour	id	6	Information on the way users navigate through information in the PATHS database.
ugroup	id	2	Codelist of user groups used to distinguish what privieges each user has in the PATHS system. New users by default are members of the 'user' group (id=1). Administrator users are members of the 'admin' group (id=2). New groups may be added to further differentiate privileges.
usr	id	12	Information about users such as username, password, nickname etc.
usr_ugroup	id	3	Many-to-many relationship table between users and user groups.
usr_workspace	id	3	
workspace	id	5	Temporary storage table for half-baked nodes and items that a user wants to add to PATHS at a later stage after working on them.
workspace_item	id	8	Temporary storage table for half-baked nodes and items that a user wants to add to PATHS at a later stage after working on them.

Entity details

Entity: behaviour_link

Entity details:

Description	Information on which Items a user has traversed between.
Primary key constraint name	PK_behaviour_link

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	Unique identifier
FK	fk_rel_suri	CHARACTER VARYING	No	Source URI resource (the URI of the resource the user came from)
FK	fk_rel_turi	CHARACTER VARYING	No	Target URI resource (the URI of the resource the user browsed to)
	avg_ttime	INTEGER	No	Average time at target URI in seconds
	trav_count	INTEGER	No	Number of times the link has been traversed.

Entity: cog_style

Entity details:

Description	Codelist of different cognitive styles. A user may have one cognitive style.
Primary key constraint name	PK_cog_style

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	Unique identifier
	title	CHARACTER VARYING	Yes	Name of cognitive style

Entity: comment

Entity details:

Description	Comments added to objects identifiable by a URI
Primary key constraint name	PK_comment

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	Unique identifier
FK	fk_usr_id	INTEGER	Yes	Id of user creating comment
	fk_rel_uri	CHARACTER VARYING	Yes	URI of resource which comment is assigned to
	comment	TEXT	Yes	Comment text
	isdeleted	BOOLEAN	Yes	Flag indicating whether the entry is deleted (true=deleted)
	tstamp	TIMESTAMP WITH TIME ZONE	Yes	Timestamp for the time of creation of the record

Entity: item

Entity details:

Description	Information on resources imported from Alinari and Europeana, corresponding to the Europeana Semantic Elements specification.
Primary key constraint name	PK_item

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	Unique numeric identifier
	uri	CHARACTER VARYING	Yes	Unique URI of object. This is the original URI value derived from Europeana.
	usfd_id	CHARACTER VARYING	No	Internal ID used by USFD to relate items to topics and maps
	dc_title	TEXT	No	As per ESE
	dc_creator	TEXT	No	As per ESE
	dc_subject	TEXT	No	As per ESE
	dc_description	TEXT	No	As per ESE
	dc_publisher	TEXT	No	As per ESE
	dc_contributor	TEXT	No	As per ESE
	dc_date	TEXT	No	As per ESE
	dc_type	TEXT	No	As per ESE
	dc_format	TEXT	No	As per ESE
	dc_identifier	TEXT	No	As per ESE
	dc_source	TEXT	No	As per ESE
	dc_language	TEXT	No	As per ESE
	dc_relation	TEXT	No	As per ESE
	dc_rights	TEXT	No	As per ESE
	dc_coverage	TEXT	No	As per ESE
	dcterms_provenance	TEXT	No	As per ESE
	dcterms_ispartof	TEXT	No	As per ESE
	dcterms_temporal	TEXT	No	As per ESE
	dcterms_spatial	TEXT	No	As per ESE
	dcterms_medium	TEXT	No	As per ESE
	dcterms_extent	TEXT	No	As per ESE
	dcterms_alternative	TEXT	No	As per ESE
	dcterms_issued	TEXT	No	As per ESE
	dcterms_tableofcontents	TEXT	No	As per ESE
	dcterms_isreplacedby	TEXT	No	As per ESE
	europeana_unstored	TEXT	No	As per ESE
	europeana_object	TEXT	No	As per ESE. This field contains the filename to a thumbnail representation for items with europeana_type=image.For other europeana_types, this may be a sound snippet, a short video or another representative item.
	europeana_provider	TEXT	No	As per ESE
	europeana_type	TEXT	No	As per ESE
	europeana_rights	TEXT	No	As per ESE
	europeana_dataprovider	TEXT	No	As per ESE
	europeana_isshownby	TEXT	No	As per ESE
	europeana_isshownat	TEXT	No	As per ESE
	europeana_country	TEXT	No	As per ESE
	europeana_language	TEXT	No	As per ESE

	europeana_uri	TEXT	No	As per ESE
	europeana_usertag	TEXT	No	As per ESE
	europeana_year	CHARACTER VARYING	No	As per ESE
	europeana_previewNoDistribute	TEXT	No	As per ESE
	europeana_hasobject	TEXT	No	As per ESE
	paths_bow	TEXT	No	This column contains all keywords extracted from the *.events file. This is used to suppert the tag-cloud based navigation in the user interface.
	paths_facet_date	CHARACTER VARYING	No	This contains date facets extracted from the data through the enrichment process.
	paths_informativeness	DOUBLE PRECISION	No	A number indicating the informativeness of an item based on the completeness of the metadata, amount of text etc.
	paths_trav_count	INTEGER	No	Number of times an item has been selected or viewed.
	idxfti	TSVECTOR	No	An index field including keyword information from main metadata fields to be used by PostgreSQLs internal full-text search functions. Can be ignored for other purposes.

Entity: item_link

Entity details:

Description	Links between Items and external background resources (e.g. Wikipedia) as derived from semantic processing.
Primary key constraint name	PK_item_link

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	Unique numeric identifier, primary key
FK	fk_rel_uri	CHARACTER VARYING	No	URI of similar item from the item table.
	source	CHARACTER VARYING	Yes	The data source from where the item link has been extracted, i.e. Wikipedia
	field	CHARACTER VARYING	No	The name of the field where the match was found
	start_offset	INTEGER	No	The start-location of the occurence within the field
	end_offset	INTEGER	No	The end-location of the occurence within the field
	confidence	NUMERIC	No	A number indicating the confidence that the similarity match is relevnt.
	method	CHARACTER VARYING	No	The method by which the link was extracted.

	title	CHARACTER VARYING	No	The title of the linked item
	link	CHARACTER VARYING	Yes	The URI/link to the similar item, i.e. link to Wikipedia article
	sentiment	NUMERIC	No	A number that indicates the sentiment expressed about the source item in the target item. Only relevant for data extracted by iSieve's sentiment analysis technique.
	paths_classification	CHARACTER VARYING	No	The type/classification of bacground link

Entity: item_similarity

Entity details:

Description	Information on similarity between Items as derived from semantic processing.
Primary key constraint name	PK_item_similarity

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	Unique numeric identifier, PK
FK	fk_sitem_uri	CHARACTER VARYING	No	The URI of the source item from the item table
FK	fk_titem_uri	CHARACTER VARYING	No	The URI of the target item in the item table
	field	CHARACTER VARYING	No	The name of the field where the match was found
	field_no	INTEGER	No	The number of fields
	start_offset	INTEGER	No	The start of the occurence within the field
	end_offset	INTEGER	No	The end of the occurence within the field
	confidence	NUMERIC	No	A number indicating the confidence that the similarity is relevant
	method	CHARACTER VARYING	No	A string indicating the method that was used to extract the similarity link
	title	CHARACTER VARYING	No	The title of the target item.
	sentiment	NUMERIC	No	A number that indicates the sentiment expressed about the source item in the target item. Only relevant for data extracted by iSieve's sentiment analysis technique.

Entity: item_topic

Entity details:

Description	Many to many table between item and topic. One topic may have many items, one item may have many topics.
Primary key constraint name	PK_item_topic

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
FK	fk_item_uri	CHARACTER VARYING	No	
FK	fk_topic_id	CHARACTER VARYING	No	

Entity: map_point

Entity details:

Description	This table also contains the column "geom" that holds WKT POINT geometries. This is added using the PostGIS AddGeomColumn function after creation of the table.
Primary key constraint name	PK_map_point

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
FK	item_id	CHARACTER VARYING	No	
	dc_title	CHARACTER VARYING	No	
FK	parent_id	CHARACTER VARYING	No	
	dc_type	CHARACTER VARYING	No	

Entity: map_poly

Entity details:

Description	This table also contains the column "geom" that holds WKT POLYGON geometries. This is added using the PostGIS AddGeomColumn function after creation of the table.
Primary key constraint name	PK_map_poly

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
	dc_title	CHARACTER VARYING	No	
FK	topic_id	CHARACTER VARYING	No	
FK	parent_id	CHARACTER VARYING	No	
	dc_type	CHARACTER VARYING	No	
	m_order	INTEGER	No	

Entity: node

Entity details:

Description	Information about path nodes such as title, description, node_order etc.
Primary key constraint name	PK_node

Attributes:

Key	Attribute name	Data type	Not null	Description

PK	id	SERIAL	Yes	Unique numeric identifier, PK
	uri	CHARACTER VARYING	Yes	Automatically generated uri at the time of creating a new record
FK	fk_path_id	INTEGER	Yes	The unique numeric identifier of the path the node belongs to, FK
	dc_source	CHARACTER VARYING	Yes	The URI of the item (or other object identifiable by a URI) that is referenced by the node
	dc_title	CHARACTER VARYING	Yes	Title of the node
	dc_description	TEXT	No	Descriptive text about the node
	type	CHARACTER VARYING	No	Type of node, i.e. image, text etc.
	paths_thumbnail	CHARACTER VARYING	No	Filename of a thumbnail explicitly chosen for the respective node. Not derived from the item.
	node_order	DOUBLE PRECISION	No	Deprecated (in the first version of the prototype indicated the sequential ordering of linear paths)
	paths_prev	CHARACTER VARYING[]	No	Array of node URIs for nodes that are "before" the present node within a path. Introduced to support branching paths.
	paths_next	CHARACTER VARYING[]	No	Array of node URIs for nodes that "follows" the present node within a path. Introduced to support branching paths.
	paths_start	BOOLEAN	Yes	A boolean value that is set to true on any node where a path can start, i.e. where there are no node identifiers in paths_prev.
	isdeleted	BOOLEAN	Yes	A boolean value that indicates whether a node has been deleted. True = deleted, false = not deleted.
	tstamp	TIMESTAMP WITH TIME ZONE	Yes	An automatically generate timestamp at the time when the record is created.
	idxfti	TSVECTOR	No	An index field including keyword information from main metadata fields to be used by

				PostgreSQLs internal full-text search functions

Entity: path

Entity details:

Description	Information about paths such as title, subject, description etc.
Primary key constraint name	PK_path

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	Unique numeric identifier, PK
FK	fk_usr_id	INTEGER	Yes	Id of user who created path
	uri	CHARACTER VARYING	Yes	Automatically generated uri at the time of creating a new record
	dc_title	CHARACTER VARYING	Yes	Title of path, taken from Dublin Core namespace
	dc_subject	CHARACTER VARYING	No	Subject of path, taken from Dublin Core namespace. Multiple values are separated by semi-colon ";"
	dc_description	TEXT	No	Description of path, taken from Dublin Core namespace.
	access	CHARACTER VARYING	No	Any access restrictions associated with path
	lom_length	CHARACTER VARYING	No	Approximate time required/duration of path, taken from Learning Object Model namespace.
	isdeleted	BOOLEAN	Yes	A boolean value indicating whether the resource has been marked for deletion or not. True = deleted, false = not deleted.
	paths_status	CHARACTER VARYING	Yes	A boolean attribute that indicates whether the path is public or not.
	paths_thumbnail	CHARACTER VARYING	No	The complete URI of a thumbnail specifically chosen for this path - not derived from the items
	paths_iscloneable	BOOLEAN	Yes	Boolean value (true/false) that determines whether the path may be cloned or not
	tstamp	TIMESTAMP WITH TIME ZONE	Yes	An automatically created timestamp at the time of creating a new record
	idxfti	TSVECTOR	No	An index field including keyword information from main metadata fields to

				be used by PostgreSQLs internal full-text search functions

Entity: rating

Entity details:

Description	Assigns a rating to any resource identifiable by a URI. Rating is linked to a rating scale and a user. A user is only allowed to rate a URI resource once.
Primary key constraint name	PK_rating

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
FK	fk_usr_id	INTEGER	Yes	
FK	fk_rating_scale_id	INTEGER	Yes	
	fk_rel_uri	CHARACTER VARYING	Yes	
	isdeleted	BOOLEAN	Yes	

Entity: rating_scale

Entity details:

Description	Rating scale for paths and other resources identifiable by a URI. 1 = dislikes, 2 = likes.
Primary key constraint name	PK_rating_scale

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
	label	CHARACTER VARYING	Yes	

Entity: tag

Entity details:

Description	Tags: keywords and keyphrases assigned to URI resources. Tags may be language specific and are identifiable by a URI.
Primary key constraint name	PK_tag

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
	uri	CHARACTER VARYING	Yes	Automatically generated uri at the time of creating a new record
	label	CHARACTER VARYING	No	
	lang	CHARACTER VARYING	No	

Entity: tagging

Entity details:

Description	Association between tags, users and resources identifiable by a URI. A user can only add the same keyword to a resource once.
Primary key constraint name	PK_tagging

Attributes:

Key	Attribute name	Data type	Not null	Description

PK	id	SERIAL	Yes	
FK	fk_tag_id	INTEGER	Yes	
FK	fk_usr_id	INTEGER	Yes	
	fk_rel_uri	CHARACTER	Yes	
	isdeleted	BOOLEAN	Yes	

Entity: topic

Entity details:

Description	Information about topic hierarchies
Primary key constraint name	PK_topic

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	CHARACTER VARYING	Yes	
	fk_parent_topic_ids	CHARACTER VARYING[]	No	This is an array field...
	uri	CHARACTER VARYING	No	Automatically generated uri at the time of creating a new record
	dc_title	CHARACTER VARYING	Yes	

Entity: uaction

Entity details:

Description	
Primary key constraint name	PK_uaction

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
FK	fk_usr_id	INTEGER	Yes	
	usession	CHARACTER VARYING	No	
	dc_source	CHARACTER VARYING	Yes	
	paths_request	TEXT	Yes	
	tstamp	TIMESTAMP WITH TIME ZONE	No	

Entity: ubehaviour

Entity details:

Description	Information on the way users navigate through information in the PATHS database.
Primary key constraint name	PK_ubehaviour

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
FK	fk_usr_id	INTEGER	Yes	
	usession	CHARACTER VARYING	Yes	
	dc_title	CHARACTER VARYING	No	URI of object user is navigating from
	dc_source	CHARACTER VARYING	No	

	tstamp	TIME WITH TIME ZONE	No	Time spent at source in seconds

Entity: ugroup

Entity details:

Description	Codelist of user groups used to distinguish what privieges each user has in the PATHS system. New users by default are members of the 'user' group (id=1). Administrator users are members of the 'admin' group (id=2). New groups may be added to further differentiate privileges.
Primary key constraint name	PK_user_group

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
	title	CHARACTER VARYING	Yes	

Entity: usr

Entity details:

Description	Information about users such as username, password, nickname etc.
Primary key constraint name	PK_usr

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
FK	fk_cog_style_id	INTEGER	Yes	The id of the cognitive style associated with the user
	uri	CHARACTER VARYING	Yes	Automatically generated uri at the time of creating a new record
	usr	CHARACTER VARYING	Yes	In the first prototype, the usr field stored a username that could be distinctive from the e-mail address of the user. This was counter-intuitive and has been replaced in the final prototype. The field still stores the user name for backwards compatibility but upon registering, the field is set to the same value as the e-mail address.
	foaf_nick	CHARACTER	No	The nick-name that the user will be displayed as in the paths prototype user interface
	pwd	CHARACTER VARYING	No	The users chosen password
	email	CHARACTER VARYING	Yes	The e-mail address of the user. This is also copied to the usr field for backwards compatibility. In web services this value is also referred to as "foaf_mbox".
	email_visibility	CHARACTER VARYING	Yes	Value that indicates whether the e-mail address of the user shall be publicly displayed in the user

				interface. In web services also referred to as "foaf_mbox_visibility"
	openid	BOOLEAN	No	Boolean value indicating whether the account belongs to an OpenID provider
	isdeleted	BOOLEAN	Yes	Boolean value stating whether the user has been deleted or not. True = deleted, false = not deleted
	istemporary	BOOLEAN	Yes	Boolean value indicating whether this is a temporary user. True = temporary, false = permanent.
	tstamp	TIMESTAMP WITH TIME ZONE	Yes	An automatically generated time-stamp at the time the record is created.

Entity: usr_ugroup

Entity details:

Description	Many-to-many relationship table between users and user groups.
Primary key constraint name	PK_usr_ugroup

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
FK	fk_usr_id	INTEGER	Yes	
FK	fk_ugroup_id	INTEGER	Yes	

Entity: usr_workspace

Entity details:

Description	
Primary key constraint name	PK_usr_workspace

Attributes:

Key	Attribute name	Data type	Not null	Description
PK, FK	id	SERIAL	Yes	
	fk_usr_id	INTEGER	Yes	
	fk_workspace_id	INTEGER	Yes	

Entity: workspace

Entity details:

Description	Temporary storage table for half-baked nodes and items that a user wants to add to PATHS at a later stage after working on them.
Primary key constraint name	PK_workspace

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
FK	fk_usr_id	INTEGER	Yes	
	uri	CHARACTER VARYING	Yes	
	tstamp	TIME WITH TIME ZONE	No	
	isprimary	BOOLEAN	No	

Entity: workspace_item

Entity details:

Description	Temporary storage table for half-baked nodes and items that a user wants to add to PATHS at a later stage after working on them.
Primary key constraint name	PK_workspace_item

Attributes:

Key	Attribute name	Data type	Not null	Description
PK	id	SERIAL	Yes	
	uri	CHARACTER VARYING	Yes	
FK	fk_workspace_id	INTEGER	Yes	
	dc_source	CHARACTER VARYING	No	
	dc_title	CHARACTER VARYING	Yes	
	dc_description	TEXT	No	
	type	CHARACTER VARYING	No	
	paths_thumbnail	CHARACTER VARYING	No	

Appendix C

PATHS Web Service API reference

Date: 30.04.2013

Edited by (Stein) Runar Bergheim (AVINET) with major contributions from Mark Hall (USFD) & Alok Singh (AVINET)

1. Web Service: Usr

Summary: The Usr web service contains methods for authenticating users, creating and modifying users, logging user behavior and issuing reminder e-mails upon forgetting passwords. The service is fundamental to web services which require authentication.

1.1 Web Method: Current

Summary: Gets information about the currently authenticated or temporary user.

1.1.1. Current Request Parameters

N/A (this web method does not accept any calling parameters)

1.1.2. Current Response

Data type	Description
s:string (JSON)	User data object for current user

1.1.3. Example of Current HttpGet Request

```
Request:
http://api2.paths-project.eu/Usr.asmx/Current?
```

1.2 Web Method: LogPage

Summary: Adds the given title and url to the current user's paths_breadcrumbs.

1.2.1. LogPage Request Parameters

Parameter	Data type	Description
dc_title	s:string	Title of web page to log
dc_source	s:string	URI of web page to log

1.2.2. LogPage Response

Data type	Description
s:string (JSON)	Success Message

1.2.3. Example of LogPage HttpGet Request

```
Request:
http://api2.paths-project.eu/Usr.asmx/LogPage?dc_title=s:string&
dc_source=s:string
```

1.3 Web Method: LogAction

Summary: Log every request of user that is processed

1.3.1. LogAction Request Parameters

Parameter	Data type	Description
dc_source	s:string	The URL that was requested by the user
paths_request	s:string	Any request parameters formatted as a JSON structure and serialised as a string

1.3.2. LogAction Response

Data type	Description
s:string (JSON)	Returns OperationCompletedSuccessfully (code=2)

1.3.3. Example of LogAction HttpGet Request

```
Request:
http://api2.paths-project.eu/Usr.asmx/LogAction?dc_source=s:string&
paths_request=s:string
```

1.4 Web Method: UserGet

Summary: Returns information about the user identified by the specified ID

1.4.1. UserGet Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	uri of user

1.4.2. UserGet Response

Data type	Description
s:string (JSON)	User data object

1.4.3. Example of UserGet HttpGet Request

```
Request:
http://api2.paths-project.eu/Usr.asmx/UserGet?paths_identifier=s:string
```

1.5 Web Method: Paths

Summary: Fetches all of the current user's paths

1.5.1. Paths Request Parameters

N/A (this web method does not accept any calling parameters)

1.5.2. Paths Response

Data type	Description
s:string (JSON)	Path data object

1.5.3. Example of Paths HttpGet Request

```
Request:
http://api2.paths-project.eu/Usr.asmx/Paths?
```

1.6 Web Method: UserLogin

Summary: Perform user login

1.6.1. UserLogin Request Parameters

Parameter	Data type	Description
foaf_mbox	s:string	The user's e-mail address to use as the login
paths_password	s:string	The user's password

1.6.2. UserLogin Response

Data type	Description
s:string (JSON)	AuthenticationSucceeded (code=4) on success, AuthenticationFailed (code=1) on wrong username/password, OperationFailed (code=3) on error.

1.6.3. Example of UserLogin HttpGet Request

```
Request:
http://api2.paths-project.eu/Usr.asmx/UserLogin?foaf_mbox=s:string&
paths_password=s:string
```

1.7 Web Method: UserLogout

Summary: Logs the current user out of the system by erasing user information from the session

1.7.1. UserLogout Request Parameters

N/A (this web method does not accept any calling parameters)

1.7.2. UserLogout Response

Data type	Description
s:string (JSON)	Always returns LogoutSuccess (code=6)

1.7.3. Example of UserLogout HttpGet Request

```
Request:
http://api2.paths-project.eu/Usr.asmx/UserLogout?
```

1.8 Web Method: UserRegister

Summary: Create a new user

1.8.1. UserRegister Request Parameters

Parameter	Data type	Description
foaf_nick	s:string	Nickname/display name
foaf_mbox	s:string	E-mail address, will be used as user name
paths_password	s:string	Password

1.8.2. UserRegister Response

Data type	Description
s:string (JSON)	Returns OperationCompletedSuccessfully (code=2) and the user data for the created user

1.8.3. Example of UserRegister HttpGet Request

```
Request:
http://api2.paths-project.eu/Usr.asmx/UserRegister?foaf_nick=s:string&
foaf_mbox=s:string&paths_password=s:string
```

2. Web Service: Item

Summary: The Item web service contains methods for querying and retrieving information about items. PATHS items are information derived from Europeana and Alinari and includes most attributes defined by the Europeana Semantic Elements. Items have been enriched with (1) background links, (2) topic links and (3) item similarity links.

2.1 Web Method: Get

Summary: Get a single item by its URI

2.1.1. Get Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	URI of item

2.1.2. Get Response

Data type	Description
s:string (JSON)	JSON String: Single item information

2.1.3. Example of Get HttpGet Request

```
Request:
http://api2.paths-project.eu/Item.asmx/Get?paths_identifier=s:string
```

2.2 Web Method: Bbox

Summary: Web service returns items based on a bounding box and a limit parameter

2.2.1. Bbox Request Parameters

Parameter	Data type	Description
bbox	s:string	string: a comma separated list of values
limit	s:int	integer: a number of items to return

2.2.2. Bbox Response

Data type	Description
s:string (JSON)	

2.2.3. Example of Bbox HttpGet Request

```
Request:
http://api2.paths-project.eu/Item.asmx/Bbox?bbox=s:string&limit=s:int
```

3. Web Service: Topic

Summary: The topic web service contains methods for querying, traversing and interacting with multi-hierarchies of concept labels as well as their related path, node and item objects.

3.1 Web Method: GetTopicHierarchy

Summary: Returns the parent hierarchy of a topic by its ID

3.1.1. GetTopicHierarchy Request Parameters

Parameter	Data type	Description
topic_id	s:string	Unique database identifier of topic

3.1.2. GetTopicHierarchy Response

Data type	Description
s:string (JSON)	JSON String: Topic hierarchy

3.1.3. Example of GetTopicHierarchy HttpGet Request

```
Request:
http://api2.paths-project.eu/Topic.asmx/GetTopicHierarchy?topic_id=s:string
```

3.2 Web Method: GetTopicByUri

Summary: Get parent topic hierarchy of topic by its URI

3.2.1. GetTopicByUri Request Parameters

Parameter	Data type	Description
topic_uri	s:string	URI of topic

3.2.2. GetTopicByUri Response

Data type	Description
s:string (JSON)	JSON String: Topic hierarchy

3.2.3. Example of GetTopicByUri HttpGet Request

```
Request:
http://api2.paths-project.eu/Topic.asmx/GetTopicByUri?topic_uri=s:string
```

3.3 Web Method: GetTopicById

Summary: Get a topic by its ID

3.3.1. GetTopicById Request Parameters

Parameter	Data type	Description
topic_id	s:int	Unique database identifier of topic

3.3.2. GetTopicById Response

Data type	Description
s:string (JSON)	JSON String: Single topic information

3.3.3. Example of GetTopicById HttpGet Request

```
Request:
http://api2.paths-project.eu/Topic.asmx/GetTopicById?topic_id=s:int
```

4. Web Service: Workspace

Summary: The Workspace web service contains methods for creating, managing, querying and deleting workspace items. A workspace item can be considered a node which has not yet been completed and/or assigned ot a Path. Workspace items can refer to any object identifiable by a URI and most commonly references records from the Items table.

4.1 Web Method: Item

Summary: Fetches a single item from the workspace

4.1.1. Item Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	Unique database identifier of workspace item to be retrieved.
paths_item_identifier	s:string	Items URI.

4.1.2. Item Response

Data type	Description
s:string (JSON)	JSON String: Single workspace item information

4.1.3. Example of Item HttpGet Request

```
Request:
http://api2.paths-project.eu/Workspace.asmx/Item?paths_identifier=s:string&
paths_item_identifier=s:string
```

4.2 Web Method: Items

Summary: Get all workspace items for the current authenticated or temporary user.

4.2.1. Items Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	Unique identifier of the workspace item.

4.2.2. Items Response

Data type	Description
s:string (JSON)	JSON String: List of workspace items information

4.2.3. Example of Items HttpGet Request

```
Request:
http://api2.paths-project.eu/Workspace.asmx/Items?paths_identifier=s:string
```

4.3 Web Method: Add

Summary: This function will first check for the given uri in workspace table if record presents it will insert new record in workspace_item with exiting workspace id. Otherwise will insert new record in both table (workspace,workspace_item)

4.3.1. Add Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	URI reference to the Workspace
dc_title	s:string	Title of workspace item
dc_description	s:string	Description of workspace item (optional)
dc_source	s:string	Any URI, but commonly a reference to the URI of a PATHS Item
paths_thumbnail	s:string	The item's thumbnail
paths_type	s:string	Type of workspace item

4.3.2. Add Response

Data type	Description
s:string (JSON)	JSON String: Workspace item

4.3.3. Example of Add HttpGet Request

```
Request:
http://api2.paths-project.eu/Workspace.asmx/Add?paths_identifier=s:string&
dc_title=s:string&dc_description=s:string&dc_source=s:string&
paths_thumbnail=s:string&paths_type=s:string
```

4.4 Web Method: Update

Summary: Updates the information about an item in the users workspace

4.4.1. Update Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	Unique identifier of the workspace item.
paths_item_identifier	s:string	URI of referenced object in workspace_item table
dc_title	s:string	Title of workspace item
dc_description	s:string	Description of workspace item

4.4.2. Update Response

Data type	Description
s:string (JSON)	JSON String: Single workspace item information

4.4.3. Example of Update HttpGet Request

```
Request:
http://api2.paths-project.eu/Workspace.asmx/Update?paths_identifier=s:string&
paths_item_identifier=s:string&dc_title=s:string&dc_description=s:string
```

4.5 Web Method: Delete

Summary: Deletes an item from the workspace.

4.5.1. Delete Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	Unique identifier of the workspace item.
paths_item_identifier	s:string	URI of referenced object in workspace_item table

4.5.2. Delete Response

Data type	Description
s:string (JSON)	JSON String: OperationCompletedSuccessfully (code=2) on success, DatabaseSQLError (code=7) on error.

4.5.3. Example of Delete HttpGet Request

```
Request:
http://api2.paths-project.eu/Workspace.asmx/Delete?paths_identifier=s:string&
paths_item_identifier=s:string
```

5. Web Service: Path

Summary: The Path web service contains methods for creation, editing and deletion of paths and path nodes. Furthermore, it has functions to transfer work space items to nodes in a path and to qury paths and nodes. Paths and nodes are the core dynamic objects in the PATHS Web Service API. A path consist of one or more nodes, a node references an item (or another object) via a URI.

5.1 Web Method: Get

Summary: Get a single path identified by its URI

5.1.1. Get Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	URI of path to be retrieved

5.1.2. Get Response

Data type	Description
s:string (JSON)	Path data object

5.1.3. Example of Get HttpGet Request

```
Request:
http://api2.paths-project.eu/Path.asmx/Get?paths_identifier=s:string
```

5.2 Web Method: Create

Summary: Create a new path

Remark: Methods requires a user to be authenticated

5.2.1. Create Request Parameters

N/A (this web method does not accept any calling parameters)

5.2.2. Create Response

Data type	Description
s:string (JSON)	Path data object for created path

5.2.3. Example of Create HttpGet Request

```
Request:
http://api2.paths-project.eu/Path.asmx/Create?
```

5.3 Web Method: Delete

Summary: Delete a path identified by its URI

Remark: Method requires authentication

5.3.1. Delete Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	URI of path to be deleted

5.3.2. Delete Response

Data type	Description
s:string (JSON)	JSON String

5.3.3. Example of Delete HttpGet Request

```
Request:
http://api2.paths-project.eu/Path.asmx/Delete?paths_identifier=s:string
```

5.4 Web Method: Update

Summary: Updates path data for a path identified by its URI

Remark:

5.4.1. Update Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	URI of path to be modified
dc_title	s:string	Modified title of path (string, optional)
dc_description	s:string	Modified description of path (optional)
paths_duration	s:string	Modified length/duration of path (optional)
paths_access	s:string	Modified access information for path (optional)
paths_clone	s:string	True false (optional)
paths_thumbnail	s:string	Thumbnail URL (optional)
dc_subject	s:string	Modified subject of path (optional) separater multiple entries by a semicolon ";"
paths_start	s:string	Modified rights statement of path (optional)
paths_node_changes	s:string	Change in Node order

5.4.2. Update Response

Data type	Description
s:string (JSON)	JSON object on success

5.4.3. Example of Update HttpGet Request

```
Request:
http://api2.paths-project.eu/Path.asmx/Update?paths_identifier=s:string&
dc_title=s:string&dc_description=s:string&paths_duration=s:string&
paths_access=s:string&paths_clone=s:string&paths_thumbnail=s:string&
dc_subject=s:string&paths_start=s:string&paths_node_changes=s:string
```

5.5 Web Method: Update_Node

Summary: Update information of a node identified by its URI

Remark: Method requires authentication

5.5.1. Update_Node Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	URI of path
paths_node_identifier	s:string	URI of node to be updated
dc_title	s:string	Title of node
dc_description	s:string	Description of node

5.5.2. Update_Node Response

Data type	Description
s:string (JSON)	JSON String: OperationCompletedSuccessfully (code=2) on success

5.5.3. Example of Update_Node HttpGet Request

```
Request:
http://api2.paths-project.eu/Path.asmx/Update_Node?paths_identifier=s:string&
paths_node_identifier=s:string&dc_title=s:string&dc_description=s:string
```

5.6 Web Method: Delete_Node

Summary: Delete a node identified by its URI

Remark: Method requires authentication

5.6.1. Delete_Node Request Parameters

Parameter	Data type	Description
paths_identifier	s:string	URI of path
paths_node_identifier	s:string	URI of node to be deleted

5.6.2. Delete_Node Response

Data type	Description
s:string (JSON)	JSON String: Single node information

5.6.3. Example of Delete_Node HttpGet Request

```
Request:
http://api2.paths-project.eu/Path.asmx/Delete_Node?paths_identifier=s:string&
paths_node_identifier=s:string
```

6. Web Service: Social

Summary: The web service Social contains all functionality associated with user generated content which may be attached to paths, nodes and items. UGC elements are associated with resources via a URI and may in principle be attached to any web resource. This reduces the amount of tables required for the connections and simplifies the data management.

6.1 Web Method: GetCommentsForUri

Summary: Get comments for a web resource with specified URI

6.1.1. GetCommentsForUri Request Parameters

Parameter	Data type	Description
fk_rel_uri	s:string	URI of web resource for which comments should be retrieved.

6.1.2. GetCommentsForUri Response

Data type	Description
s:string (JSON)	OperationCompletedSuccessfully (code=2) + list of comment data objects on success.

6.1.3. Example of GetCommentsForUri HttpGet Request

```
Request:
http://api2.paths-project.eu/Social.asmx/GetCommentsForUri?fk_rel_uri=s:string
```

6.2 Web Method: AddComment

Summary: Add new comment to web resource identified by URI

Remark: Web method requires user to be authenticated

6.2.1. AddComment Request Parameters

Parameter	Data type	Description
fk_rel_uri	s:string	URI of web resource to be commented upon
comment	s:string	Comment text

6.2.2. AddComment Response

Data type	Description
s:string (JSON)	OperationCompleteSuccessfully (code=2) + single comment data object

6.2.3. Example of AddComment HttpGet Request

```
Request:
http://api2.paths-project.eu/Social.asmx/AddComment?fk_rel_uri=s:string&
comment=s:string
```

6.3 Web Method: DeleteComment

Summary: Deletes comment with specified identifier

Remark: Method requires a user to be authenticated.

6.3.1. DeleteComment Request Parameters

Parameter	Data type	Description
comment_id	s:int	Unique database identifier of comment to be deleted

6.3.2. DeleteComment Response

Data type	Description
s:string (JSON)	OperationCompletedSuccessfully (code=2) on success.

6.3.3. Example of DeleteComment HttpGet Request

```
Request:
http://api2.paths-project.eu/Social.asmx/DeleteComment?comment_id=s:int
```

6.4 Web Method: AddTag

Summary: Adds a tag (keyword) to a resource identified by a URI

Remark: Method requires a user to be authenticated

6.4.1. AddTag Request Parameters

Parameter	Data type	Description
fk_rel_uri	s:string	URI of resource which tag should be added to
tag	s:string	Any keyword or keyphrase to be used as tag

6.4.2. AddTag Response

Data type	Description
s:string (JSON)	Tag data object and OperationCompletedSuccessfully (code=2) on success

6.4.3. Example of AddTag HttpGet Request

```
Request:
http://api2.paths-project.eu/Social.asmx/AddTag?fk_rel_uri=s:string&tag=s:string
```

6.5 Web Method: DeleteTag

Summary: Delete tag with specified URI

Remark: Method requires a user to be authenticated

6.5.1. DeleteTag Request Parameters

Parameter	Data type	Description
tag_uri	s:string	URI of the tag to be deleted

6.5.2. DeleteTag Response

Data type	Description
s:string (JSON)	OperationCompletedSuccessfully (code=2) on success

6.5.3. Example of DeleteTag HttpGet Request

```
Request:
http://api2.paths-project.eu/Social.asmx/DeleteTag?tag_uri=s:string
```

6.6 Web Method: GetTagsForUri

Summary: Get list of tags associated with a specific resource identified by its URI

6.6.1. GetTagsForUri Request Parameters

Parameter	Data type	Description
fk_rel_uri	s:string	URI of resource for which tags should be retrieved

6.6.2. GetTagsForUri Response

Data type	Description
s:string (JSON)	QueryDidNotReturnRecords (code=8) if no tags are found, OperationCompletedSuccessfully (code=2) and list of tag data objects on success

6.6.3. Example of GetTagsForUri HttpGet Request

```
Request:
http://api2.paths-project.eu/Social.asmx/GetTagsForUri?fk_rel_uri=s:string
```

6.7 Web Method: AddRating

Summary: Add rating to a resource identified by its URI

Remark: Requires an authenticated or temporary user session

6.7.1. AddRating Request Parameters

Parameter	Data type	Description
fk_rating_scale_id	s:int	Unique database identifier for rating_scale table. 1 = dislikes, 2=likes
fk_rel_uri	s:string	URI of resource which rating should be added to

6.7.2. AddRating Response

Data type	Description
s:string (JSON)	QueryDidNotReturnRecords (code=8) if no rating values exist; OperationCompletedSuccessfully (code=2) and count of ratings

6.7.3. Example of AddRating HttpGet Request

```
Request:
http://api2.paths-project.eu/Social.asmx/AddRating?fk_rating_scale_id=s:int&
fk_rel_uri=s:string
```

6.8 Web Method: DeleteRatingForUri

Summary: Deletes a rating assigned to a resource identified by its URI

Remark: Method requires user to be authenticated. Only one rating is permitted per resource per user.

6.8.1. DeleteRatingForUri Request Parameters

Parameter	Data type	Description
fk_rel_uri	s:string	URI of resource for which rating should be deleted

6.8.2. DeleteRatingForUri Response

Data type	Description
s:string (JSON)	OperationCompletedSuccessfully (code=2) on success

6.8.3. Example of DeleteRatingForUri HttpGet Request

```
Request:
http://api2.paths-project.eu/Social.asmx/DeleteRatingForUri?fk_rel_uri=s:string
```

Appendix D

RedMine ticket example

Paths Prototype - Feature #238
WS call: paths.update_node

08 April 2013 18:17 - Mark Hall

Status:	In Progress	**Start date:**	08 April 2013
Priority:	Normal	**Due date:**	
Assignee:	Mark Hall	**% Done:**	90%
Category:	Web API	**Estimated time:**	0.00 hour
Target version:	0.2		

Description

Updates an individual node's text.

IN:
paths_identifier: The identifier of the path that includes the node to update,
paths_node_identifier: The identifier of the node to update,
dc_title: The new dc_title for the node,
dc_description: The new dc_description for the node,

OUT:
Updated node as described in #232.

History

#1 - 16 April 2013 13:30 - Alok Singh

Hi Mark,

Can you please explain why we are passing "paths_identifier" and "paths_node_identifier" both.
Only "paths_node_identifier" is enough for updating the node.

Thanks,
Alok

#2 - 16 April 2013 14:51 - Mark Hall

Because the paths_node_identifier does not have to be globally unique, only unique within the path. Thus both are necessary in the same way as for the workspace calls.

#3 - 21 April 2013 14:31 - Alok Singh

- Status changed from New to In Progress

- % Done changed from 0 to 30

#4 - 27 April 2013 10:34 - Alok Singh

- % Done changed from 30 to 90

#5 - 27 April 2013 12:25 - Alok Singh

- Assignee changed from Alok Singh to Mark Hall

Hi Mark,

Please test this service.

Thanks,
Alok

Paths Prototype - Feature #240
WS call: workspace.item

08 April 2013 18:22 - Mark Hall

Status:	Resolved	**Start date:**	08 April 2013
Priority:	Normal	**Due date:**	
Assignee:	Alok Singh	**% Done:**	90%
Category:	Web API	**Estimated time:**	5.00 hours
Target version:	0.2		

Description

Fetches a single item from the workspace.

IN:
paths_identifier: The identifier of the workspace to get the item from
paths_item_identifier: The identifier of the item to get

OUT:

```
{
  paths_identifier: The workspace item's unique identifier,
  dc_title: The workspace item's title,
  dc_description: The workspace item's description (optional),
  dc_source: The workspace item's source data (optional),
  paths_thumbnail: The workspace item's thumbnail(s) (optional),
  paths_type: The type of workspace item (item, search, topic, ...)
}
```

Related issues:

Related to Feature # 241: WS call: workspace.items	**Closed**	**08 April 2013**

History

#1 - 08 April 2013 18:27 - Mark Hall

- Description updated

#2 - 08 April 2013 18:29 - Mark Hall

- Subject changed from WS call: paths.workspace_item to WS call: workspace.item

#3 - 13 April 2013 10:30 - Alok Singh

For the thumbnail, there is only one accessible image for each workspace item / i.e. the isShownAt and isShownBy fields in the Europeana items table that can be traversed via the associated item uri. If you want to return a thumbnail for the workspace item, this is the only place we can get it. Are we understanding you correctly? :-)

#4 - 13 April 2013 10:34 - Alok Singh

Please disregard the question above... worked on the requirements in the wrong order for a few minutes there... :-)

#5 - 13 April 2013 16:40 - Alok Singh

- Status changed from New to In Progress

- % Done changed from 0 to 60

- Estimated time set to 4.00

#6 - 15 April 2013 12:36 - Alok Singh

- Assignee changed from Alok Singh to Mark Hall

- % Done changed from 60 to 90

- Estimated time changed from 4.00 to 5.00

Please review it and let me know your suggestion.

#7 - 22 April 2013 14:33 - Mark Hall

- Description updated

Updated the schema to clarify that the attributes returned are to be those of the workspace item and not of whatever the workspace item refers to.

#8 - 30 April 2013 10:55 - Mark Hall

- Status changed from In Progress to Resolved

- Assignee changed from Mark Hall to Alok Singh

Tested and works.

Paths Prototype - Feature #243
WS call: workspace.update

08 April 2013 19:10 - Mark Hall

Status:	Closed	**Start date:**	08 April 2013
Priority:	Normal	**Due date:**	
Assignee:	Alok Singh	**% Done:**	80%
Category:	Web API	**Estimated time:**	2.00 hours
Target version:	0.2		

Description

Update a node in the workspace

IN:
paths_identifier: The identifier of the workspace to which to add the item
paths_item_identifier: The identifier of the item to update
dc_title: The item's new title,
dc_description: The item's new description (optional),

OUT:
The updated workspace item in the format defined in #240.

History

#1 - 08 April 2013 19:11 - Mark Hall

- Description updated

#2 - 13 April 2013 17:10 - Alok Singh

- Status changed from New to In Progress
- % Done changed from 0 to 60
- Estimated time set to 2.00

#3 - 15 April 2013 12:37 - Alok Singh

- Assignee changed from Alok Singh to Mark Hall
- % Done changed from 60 to 80

Please review it and let me know your suggestion.

#4 - 22 April 2013 17:54 - Mark Hall

- Assignee changed from Mark Hall to Alok Singh

The dc_description parameter must accept HTML values. Currently a "A potentially dangerous Request.Form value was detected from the client" error is returned.

#5 - 23 April 2013 09:10 - Alok Singh

- Assignee changed from Alok Singh to Mark Hall

Now dc_description parameter is accepting HTML values

#6 - 23 April 2013 12:56 - Mark Hall

- Status changed from In Progress to Resolved

- Assignee changed from Mark Hall to Alok Singh

Tested and works.

#7 - 23 April 2013 14:32 - Alok Singh

- Status changed from Resolved to Closed