Not just another portal, not just another digital library: A portrait of Europeana as an application program interface

Cesare Concordia, Stefan Gradmann and Sjoerd Siebinga

IFLA Journal 2010; 36; 61
DOI: 10.1177/0340035209360764

International Federation of
Library Associations and Institutions
36(1) 61–69
© The Author(s) 2010
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0340035209360764
ifla.sagepub.com

Not just another portal, not just another digital library: A portrait of Europeana as an application program interface

Cesare Concordia
Institute of Information Science and Technologies (ISTI-CNR), Pisa

Stefan Gradmann
Humboldt-Universität zu Berlin

Sjoerd Siebinga
Europeana Development, The Hague

Abstract
To the general public, Europeana is primarily perceived as a portal exposing a great amount of cultural heritage information. Even though this perception is not entirely misleading, the main goal of Europeana is rather to build an open services platform enabling users and cultural institutions to access and manage a large collection of surrogate objects representing digital and digitized content via an application program interface (API). The paper covers some details of the overall data space schema, of the API description and of the Europeana Portal implementation; it also discusses use cases and the mental approach that users, in particular cultural institutions, should adopt to completely exploit the potential of the Europeana services platform together with a discussion of related risks. The authors represent key players in the Europeana specification, development and implementation process currently under way.

Keywords
Europeana, application program interfaces, cultural heritage information, open services platforms, web portals, semantic web

What is Europeana?

To the general public, Europeana (http://www.europeana.eu) is primarily perceived as a portal exposing increasingly impressive amounts of cultural heritage from various sources to Europe's citizens. Even though this perception of course is not entirely misleading (and even conforms to most of the European Commission's communication about Europeana) it does not capture the essential characteristics of what we try to build in Europeana. On a very abstract level, Europeana can be seen as a large collection of surrogate objects representing born digital or digitized cultural heritage objects which themselves remain outside the Europeana data space (they need to be accessed by Europeana once, however, for processing with the aim of producing the surrogate representations). In this abstract vision, the surrogates are linked to each other and additionally are contextualized with links to nodes of a semantic network that forms the second data layer in Europeana. These two links

Corresponding author:
Stefan Gradmann is Professor of Library and Information Science with a special focus on Knowledge Management at the Berlin Institute of Library and information Science (B-SLIS) at Humboldt-Universität zu Berlin. Contacts: Unter den Linden 6, 10099 Berlin, Germany. Tel +49 30 2093 4481. Fax +49 30 2093 4335. E-mail: stefan.gradmann@ibi.hu-berlin.de

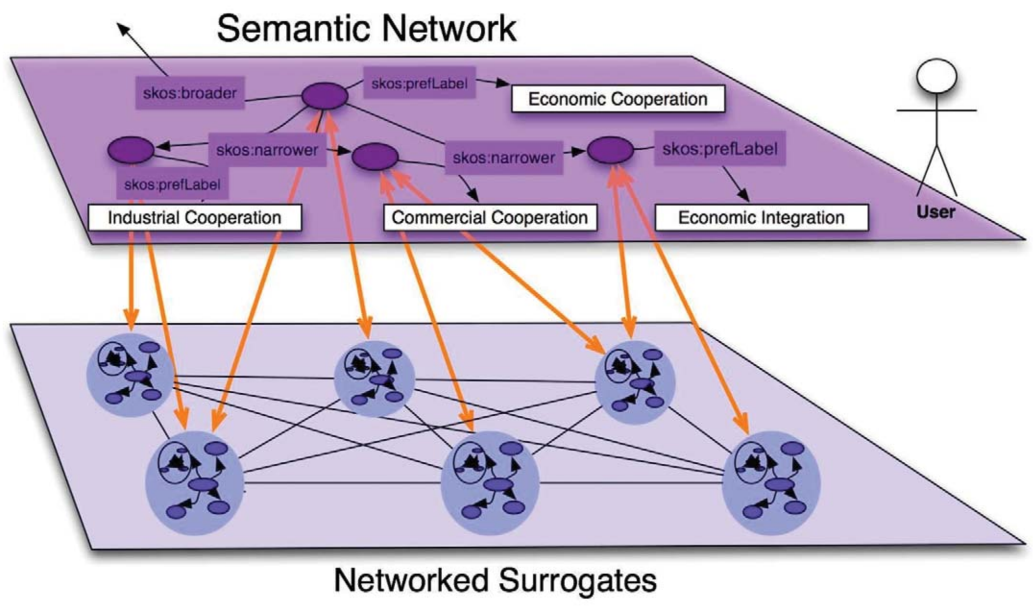

Figure 1. Semantic network and networked surrogates.

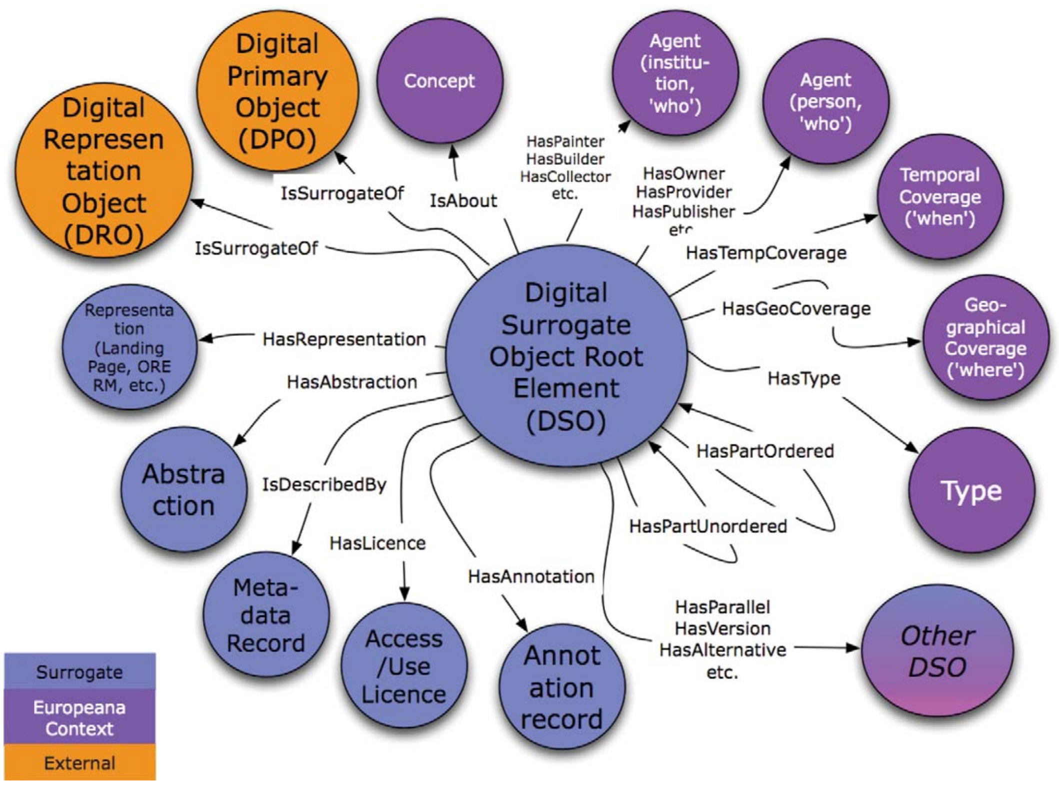
Digital Representation Object (DRO)
Digital Primary Object (DPO)
Concept
Agent (institution, 'who')
Agent (person, 'who')
Temporal Coverage ('when')
Geo-graphical Coverage ('where')
Type
Other DSO
Digital Surrogate Object Root Element (DSO)
Representation (Landing Page, ORE RM, etc.)
Abstraction
Meta-data Record
Access /Use Licence
Annotation record
IsSurrogateOf
IsSurrogateOf
IsAbout
HasPainter HasBuilder HasCollector etc.
HasOwner HasProvider HasPublisher etc.
HasTempCoverage
HasGeoCoverage
HasType
HasPartOrdered
HasPartUnordered
HasParallel HasVersion HasAlternative etc.
HasAnnotation
HasLicence
IsDescribedBy
HasAbstraction
HasRepresentation
Surrogate
Europeana Context
External

Figure 2. Digital surrogate object.

together are used to create rich functionality that is offered on the user interface. This view is illustrated in Figure 1.

Furthermore, as illustrated in Figure 2, these surrogates can have a relatively complex internal structure: the circles in light grey [blue in the online edition – *Ed.*] show constituents of a Digital Surrogate Object (DSO) such as related metadata, licensing information, abstractions (such as tables of contents or color histograms), annotations and representations of the surrogate such as a landing page or ORE resource maps. Furthermore, DSOs may contain other

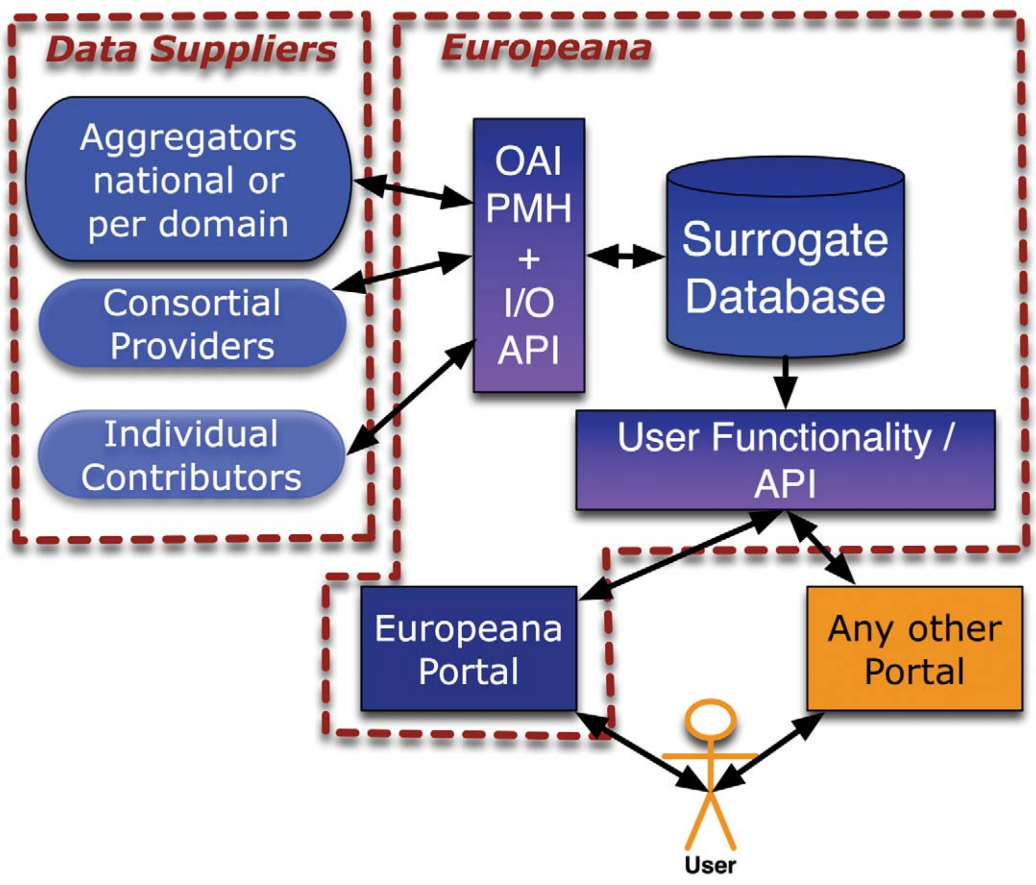

Figure 3. Europeana big picture.

DSOs as parts as in the case of a scanned book with individual surrogates for each page. On the other hand, DSOs have contextual links to other DSOs as well as to concept nodes (the circles in dark grey [purple in the online edition – *Ed*.]) such as those representing time and space entities or abstract concepts.

Both the internal structure of the surrogates and their contextualization build upon the elements provided by the content suppliers, but substantial parts of this structure and context will be created in the course of the Europeana data ingestion routines.

Therefore – and as indicated in Figure 3 – Europeana will not only have an API for end user functionality as further detailed below, but also an I/O-API enabling data flow from and to the content providers – the latter creates the option of re-integrating enriched content in the remote applications of the data providers.

The heart of the Europeana project thus is an endeavour to build an open platform fostering functional, technical and data interoperability.

What is an API?

According to the DELOS DL Manifesto (Candela et al., 2007) we can conceive the Europeana software as a Digital Library System (DLS) which is defined as: "a software system that is based on a defined (possibly distributed) architecture and provides all functionality required by a particular Digital Library". Designing a DLS is a complex task, it requires integrating knowledge and methodologies from various disciplines, such as content management, metadata management, information retrieval, distributed database management and human computer interaction, to mention the most relevant (Gonçalves et al., 2004). Implementing a DLS then means building sophisticated and extensible software that integrates techniques and technologies from the above-mentioned disciplines into a coherent system. The core of such a software system is called the Digital Library Management System (DLMS) (Smith et al., 2003). In general, a DLMS is a software system implementing the business logic and the data access functionalities for creating, operating and managing DLSs; examples of DLMS are DSpace (Smith et al., 2003) and BRICKS (Aloia, Concordia and Meghini, 2007). For the Europeana 1.0 project we decided to build a DLMS by (a) customizing components taken from off-the-shelf solutions, and (b) developing from scratch those software components that offer

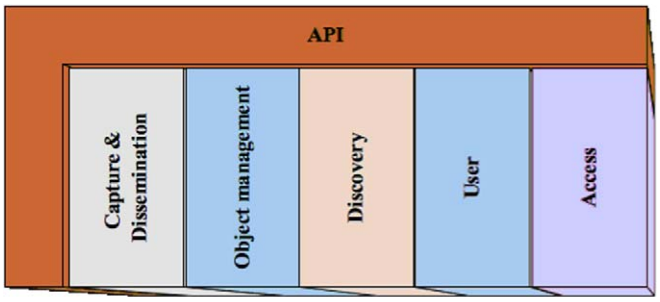

Figure 4. API and DLMS functionality.

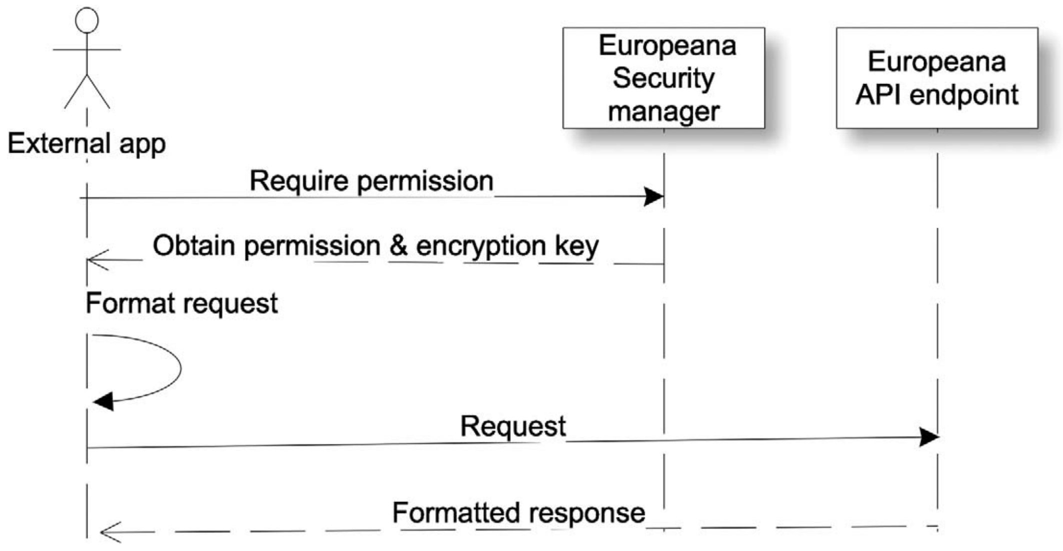

Figure 5. External application interacting with the API.

sophisticated functionalities of the Europeana Digital Library not exhibited by existing solutions.

The functionalities offered by the Europeana 1.0 DLMS can be grouped into five areas:

1. Capture and Dissemination area, offering functionalities for populating Europeana and disseminating its contents.
2. Object Management area, providing functionalities to manage digital content objects, the corresponding surrogates and the associated metadata.
3. Discovery area, supporting the indexing and searching of the Europeana content according to several paradigms.
4. User area, supporting the functionality for managing users, from single persons to institutions, all possibly grouped in dynamic communities.
5. Access area, supporting access to both Europeana services and content as well as to external resources.

Each functional area is implemented by a set of software components. One of the main goals is to make the Europeana DLMS an extensible system: it should be possible to easily add new components offering more sophisticated functionalities or replace existing components when necessary. In order to achieve this form of extensibility, every component in the Europeana architecture is accessible through an Application Program Interface (API) that exposes all the public methods of the component; interactions among components will occur only through their APIs. A subset of the API methods of DLMS components will be published and made available to external applications; these methods will form what we call the Europeana API.

The goal is to enable third party developers to build applications using the functionalities of the Europeana DLMS, and in some cases to extend those functionalities.

To hide the complexity of the underlying system, the Europeana API will be published as a set of callable methods, API endpoints and *calling conventions*. A developer who wants to build an application that uses an exposed Europeana DLMS functionality could write a routine performing three tasks (see section on Use Cases for an example):

1. select a calling convention and according to it format a request specifying a method and its arguments
2. send the request to a specific endpoint
3. receive the relative response.

The calling conventions adopted in Europeana will be initially mapped to three standard techniques for exchanging structured information: Representational State Transfer (REST) (Fielding 2000); XML Remote Procedure Call (XML-RPC (http://www.xmlrpc.com/); and Simple Object Access Protocol (SOAP) (http://www.w3.org/TR/soap/). REST is an HTTP GET or POST action; the method name and parameters are passed as values of defined keywords; in XML-RPC the request is formatted in XML according to a defined schema and posted to a URL; SOAP requests are 'envelopes' of formatted XML data posted to a URL.

One important characteristic of these formats is that they can be used to build distributed applications; this means that a third party developer can build an application running on its own server and use a communication infrastructure (for instance the Web) to interact with the Europeana DLMS.

The response data format depends on the called method and the convention used; typically it would be an XML file. However, for certain methods Europeana DLMS will also provide the standard data interchange-format JavaScript Object Notation (JSON) (http://json.org/), to help for instance the work of developers writing Graphic User Interfaces based on asynchronous JavaScript and XML (AJAX). Another important data format supported is Protocol for Metadata Harvesting defined by the Open Archive Initiative (OAI-PMH) that can be used by external applications to harvest Europeana content. The response data format can be specified as a request parameter.

Note that almost all programming languages support the above-mentioned protocols and techniques and there are several frameworks providing a compatible API; the Europeana DLMS then can easily be extended.

Publishing the Europeana API means that the Europeana DLMS will embed a security framework providing the functionalities of authentication/authorization of API invocations and data encryption/decryption when information should be kept private.

Use cases for Europeana

Use case: external Moodle application

A Europeana external application is an application that uses at least a Europeana service via the Europeana API. In this paragraph a simple use case is shortly described: how to build a plug-in for the Moodle Course Management System (http://moodle.org/) using the Europeana Advanced Search API.

The Moodle software architecture is component based, a number of its main features are implemented by separate components called modules, including themes, activities, interface languages, database schemas and course formats; moreover the system provides an API to enable developers to build new modules. If a teacher wants to add a particular functionality to a course she/he can build a module and add it to the Moodle server according to the specifications. Once the module is correctly installed it can be loaded as a widget in the course Graphic User Interface (GUI) and its functionalities will be available to the course users.

The sequence diagram in Figure 6 shows how a module (in particular an *activity* module*)* that uses the Europeana API to discover objects stored in the digital library could interact with the API endpoint.

To use the Europeana API a Moodle developer should essentially implement the following tasks:

1. Query formatting. As noted in the previous paragraph, every API endpoint can support one or more calling conventions. The developer will know from the API documentation which are supported by the Europeana Discovery manager and must format the query according one of them.
2. Query encoding. It may be the case that a developer wants to build a module searching also for information which is not in the public domain. To implement this kind of search, an encryption/decryption key and/or an identification token are needed; keys and identifiers must be obtained in advance by the developer and used to encode queries sent to API endpoints.
3. Result set parsing. As it is for the calling convention, also the result format of a method invocation will be documented. In the current prototype implementation the result set of an advanced search is formatted as a JSON file.

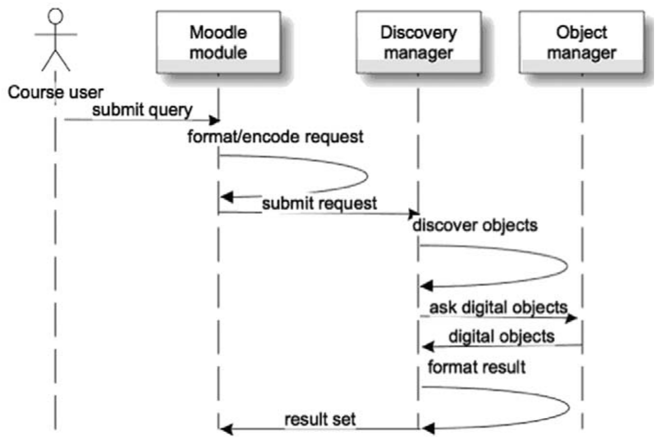

Figure 6. Using the API for connecting Moodle to Europeana.

The Europeana API will provide methods and tools to help developers in building their applications.

Humanities computing

Europeana will contain a wealth of cultural artifacts from all domains of cultural heritage and thus will have an increasing potential for all culturally related scholarship, mainly in the humanities. Our second use case therefore is taken from the humanities, and more specifically from the field of philology. Imagine a scholar working on medieval manuscripts. The manuscript she is working on is represented as a complex surrogate in Europeana with low quality reproductions of the scanned pages as part of the abstractions as well as a representation of the watermarks found in the manuscript. In the metadata part of the surrogate the researcher notices that it is dated 1480 and is supposed to be written in Strasbourg. The watermark found in the manuscript is an anchor – and fortunately one of the contextual resources of Europeana is the 'Wasserzeichen des Mittelalters' (WZMA) database of medieval watermarks at the Austrian Academy of Sciences. In that database our researcher spots two instances of exactly the anchor watermark in her manuscript in two other manuscripts, both supposed to be written in Strasbourg – but both of them dated 1446.

From this combination of information available through Europeana and its context, our researcher concludes that something must be wrong with the dating of the manuscript, as she knows that a given watermark typically was in use only for a few years and definitely not for 34 years. She thus creates an annotation embedding the link to the WZMA resource and making a statement about the supposed incorrectness of the dating.

This probably is when she has touched at the limits of what she can do with Europeana: she has sufficient inferencing elements at her disposal to raise doubts about the correctness of the dating, but in order to establish a new, correct dating she probably needs to go to a site with high quality digitization data or eventually may even have to travel to Vienna where the manuscript can be found at the Austrian National Library.

Europeana portal as a reference implementation

The prototype of the Europeana portal was launched in November 2008, (see http://www.europeana.eu). Its main purpose was to showcase the possibilities of cross-cultural domain interoperability on a pan-European level. Metadata from archives, libraries, museums and audiovisual archives became available through a single interface. The main functionalities offered by the prototype portal were:

- **Inform** the users of the Europeana initiative
- **Browse** through results via a timeline, via search terms provided by other users, and via frequently viewed item shown in the carousel on the home page
- And, finally, **search** via simple and advanced search.

From its inception the portal was designed to be a thin client around the search API. Many of the advanced features described in 'What is Europeana' will only become available in the next major releases planned in mid 2010 and 2011.

So what will be the role of the Europeana portal in the future? Most likely its role will be that of a reference implementation of the Europeana API. New features of the API will first become visible in the Europeana Portal.

Some of the main challenges for the portal/API interaction are tackling multilinguality and finding innovative ways of presenting tremendously large result sets (this is also referred to as info-graphics). Some of the new features which are currently under discussion are geo-temporal presentation, and, contextual grouping of result sets. A query for any location or date will have many thousands of relevant results, however narrow you set your scope. If you search for 'Paris' all dates currently shown in the facets are in excess of 1,000 relevant results. If you select '1888' in the timeline of the portal you will have almost 50,000 results to filter through.

Making all information multilingually available will initially only lead to a combinatorial explosion of relevant results. Obviously, if you are searching for 'painting river' in English and the query is expanded with the translations of 'painting river' in 27 European languages, your result set will dramatically increase. Here is where contextual groupings become important. The user needs to be able to filter a meaningful and manageable result-set out of hundreds of thousands results. Here it is important to note that Europeana suffers from having so much high-quality curated metadata. Any hit for a phrase 'painting river' in Europeana is most likely a very relevant hit to your query. So developing tools for contextual groupings and graphical representations to help users make sense of their search results is of key importance.

Mentality shift: Towards the cultural commons

The approach we are propagating here is based on a strong assumption: we suppose that instead of trying to sustain the digital information silos of the past, cultural heritage communities are ready for an information paradigm of linked data and thus for sharing as much semantic context as possible. Only in such a mental setting does the shift from the portal paradigm to the vision of an API as Europeana's primary incarnation truly make sense.

This mentality shift is a big leap, since it requires cultural heritage institutions to think, not primarily within the boundaries of their particular collections, but in terms of what these collections might add to a bigger, complex and distributed information continuum coupled with various contextual resources enabling European users to turn partial aggregations of this continuum into knowledge that is relevant in their specific context.

The idea thus is not to pre-aggregate information in fixed structures for basically static reuse, but to make it available together with functional primitives for usage scenarios not exclusively defined by Europeana: eScholarship collaboratories, digital libraries of all sorts, the Europeana portal itself ... the basic idea being that all of these use Europeana as some kind of digital cultural commons addressing the API that exposes Europeana data and functionality in a generic manner.

As part of this mentality shift, cultural heritage institutions will also need to increasingly feel part of a larger community sharing a set of generic standards for organizing information and making it available: the standards referred to here will mostly be created by external instances such as the W3C rather than by the cultural heritage communities themselves.

Value proposition

The Europeana API brings value to both stakeholders and users alike, but in different ways. The Europeana service needs to keep the fine balance between meeting user needs and satisfying stakeholders' demands. The following paragraphs contain a selection of added value aspects of the Europeana API.

Increased visibility and coherent branding. Europeana is a high-profile information access point that offers a unique cross-section across digitized European cultural heritage. Content that is part of Europeana will therefore be inherently more visible then just exclusively via the content providers' websites. In addition, the Europeana API will offer a coherent branding strategy for Europeana, national, or domain aggregators and content-providers. This branding strategy will increase the visibility of the content and content-provider alike.

Efficient data-mining and re-use of enriched data. Extracting use-full information from metadata and other indexable data is very computationally intensive and therefore often too expensive for individual institutions. The Europeana infrastructure uses state-of-the-art data-mining and data-enrichment techniques to identify persons, places, events and concepts and align them with existing controlled vocabularies. This extra information is available via the API for re-use into the content-provider's infrastructure.

Multilingual support. One of the biggest obstacles for creating pan-European access to cultural heritage objects is cross-lingual information retrieval. In future releases, the Europeana API will provide cross-lingual services such as query translation and expansion, metadata translation, language dependent spellings of named entities like person names and place names.

Persistent resolvability. Because Europeana aims to make each ingested object into a persistent resolvable URI, individual items can be included into well-known web services such as Wikipedia, Google Scholar, Facebook, etc. Even though the content providers' identifier might change, the Europeana ingestion mechanism will keep track of the correct object. Having a persistent resolvable URI is one of the pre-requisites for making the Europeana surrogates work in semantic web contexts, like linked-data (http://linkeddata.org/) for example.

Reuse of services. The web services offered by the Europeana API can be used to enhance the functionality of institutional web interfaces. For example, information on how to group your own result sets, based on persons, place, concepts, etc. Also the enriched geospatial and temporal information provided by Europeana might be used to provide custom timeline and map applications.

Risk analysis

In some respects the needs of users and stakeholders can be orthogonal and these differences can be considered risks to the success of the Europeana API. These risks can be subdivided into two categories: 1) risks which mitigate the value propositions and, 2) risks which lead to unwanted uses of the data outside the original scope of the stakeholders.

Risks mitigating the value propositions

The biggest risk to the Europeana API approach is the failure of adoption and lack of community buy-in. For an API to be of any use at all people need to start using it. Luckily for Europeana, there will always be at least one user, namely the Europeana portal reference implementation. When new API features will be rolled out, they can be viewed in the portal's thought-lab section.

The success of the API will also depend on how freely available it will be. Initially, the API will be available to Europeana partners only. Access to the API will most likely be moderated by the need for a 'wskey' to use the web service. Many of the web services might be completely freely accessible from the Rhine release (July 2010) onwards. However, the availability of some of the surrogate elements may still be subject to agreements with content providers.

The magnitude of adoption by Europeana partners will largely depend on how relevant the Europeana API will be for them. The most relevant part of the API for partners will be the re-use of enriched data. Thus, in order to stimulate adoption and provide added value to being an API consumer, Europeana needs to provide extensive documentation and tooling support.

Finally, intellectual property rights (IPR) are always a point of contention when any type of content is made available on the Internet. A comprehensive approach to this problem is currently under construction in the wider Europeana network. In order to become truly a part of people's workflow for information gathering, Europeana needs to move beyond largely giving access only to metadata about objects. Users want to have some kind of interactive multimedia experience with the objects of interest. For this to be possible within Europeana, the IPR issues need to resolved.

Risks leading to unwanted data uses

One notable difference between the Europeana Portal and consumers of the Europeana API is that the Europeana Portal is focused on enabling discovery based on stakeholder interests. This might not be the case for other users of the Europeana API. For example, applications making use of the Europeana API can group data in ways that might be considered offensive to our stakeholders. A grouping of artifacts linked to 'genocide' through time is an absolutely valid academic endeavor, but might be politically very undesirable.

Mashups, too, might lead to uses which are unexpected and may be unwanted. For example, information consumed from the Europeana API may become a subordinate part of the mashup instead of being its central component. So the question is how much mashup of information do the Europeana stakeholders want or want to allow?

In addition, the amount of branding that the Europeana data sets will contain needs to be carefully weighted. It is primarily a question of how much do we need and how much do we want to enforce. It is of unquestionable importance that origin and ownership need to be clearly visible, whenever the Europeana API is used. This applies both to Europeana and content providers' branding of the data sets. The most likely scenario will be that all responses from the API contain information about the ownership of the content, but leave the decision on how this is displayed to the API consumer as a recommendation.

Conclusion

As already indicated in the title of this contribution, Europeana thus is much more than a Digital Library: it is a DLS in the sense defined by DELOS, and at same time based on a DLMS as developed in the Europeana V1.0 and EuropeanaConnect projects and which may in turn be used to generate different varieties of Digital Library Systems. The API referred to in this contribution is partly a generic API of the DLMS and partly (foremost as far as surrogatres are concerned) an API of the Europeana DLS. DLMS API functions in this sense will be completely open, whereas DLS API functions may be subject to specific access conditions as mentioned above.

Europeana also is much more than a repository – and at same time much less than that: Europeana will not contain original digital objects (which will continue to be accessible at sites controlled by the rights owners exclusively). However, by creating rich surrogates as representations of these objects (including a pointer to the original) and by creating rich semantic context for these, Europeana will create an added value that can be transferred back to the content providers using the API: data flows between content providers; Europeana should be seen as bidirectional.

Finally, Europeana is much more than a portal: even though offering portal functionality; its main technical incarnation is the Application Programming Interface (API) on which the portal services will be built.

Europeana thus offers cultural heritage institutions a migration path from their current collection silos into a layered, web service-based information architecture and is conceived as an environment facilitating – and requiring – the mentality shift cultural heritage institutions will have to operate in the future, anyway.

In this sense – and being fully aware of the risks associated with our approach – we feel that the API-based model for Europeana as it is still in the course of specification will have been a definite success once major web search engines start to use our API to display European cultural heritage within their retrieval sets together with the Europeana branding of the surrogates: this contribution, among other aims, is intended as an invitation for cooperation in this sense.

References

Aloia, Nicola; Concordia, Cesare; Meghini, Carlo. (2007) Implementing BRICKS, a digital library management system. Proceedings of SEBD 2007, the 15th Italian Symposium on Advanced Database Systems. Torre Canne, Brindisi, Italy. June 2007. pp. 4–15.

Candela, Leonardo; Castelli, Donatella; Pagano, Pasquale; Thanos, Costantino; Ioannidis, Yannis; Koutrika, Georgia; Ross, Seamus; Schek, Hans-Jörg; Schuldt, Heiko. (2007) Setting the foundations of digital libraries. The DELOS Manifesto. *D-Lib Magazine,* 13(3/4) (March/April 2007).

Fielding, Roy. (2000) Architectural styles and the design of network-based software architectures. Irvine / Cal. 2000. http://www.ics.uci.edu/~fielding/pubs/dissertation/ top.htm

Gonçalves, Marcos André; Fox, Edward A.; Watson, Layne T.; Kipp, Neill A. (2004) Stream, Structures, Spaces, Scenarios, Societies (5S): A formal model for digital library. *ACM Transactions on Information Systems (TOIS)*, 22(2) 270–312.

Smith, MacKenzie; Barton, Mary; Bass, Mick; Branschofsky Margret; McClellan, Greg, Stuve, Dave; Tansley, Robert; Walker, Julie Harford. (2003) DSpace: An open source dynamic digital repository. *D-Lib Magazine*, 9(1) (January 2003).

About the authors

Cesare Concordia is a full time researcher at the Networked Multimedia Information Systems Laboratory of the Institute of Information Science and Technologies (ISTI-CNR) in Pisa. Contacts: Area della ricerca CNR, via G. Moruzzi 1, 56124 PISA, Italy. Tel. +39 050 3152970. Fax: +39 050 3152811. E-mail: cesare.concordia@isti.cnr.it

Stefan Gradmann (Corresponding author) is Professor of Library and Information Science with a special focus on Knowledge Management at the Berlin Institute of Library and information Science (B-SLIS) at Humboldt-Universität zu Berlin. Contacts: Unter den Linden 6, 10099 Berlin, Germany. Tel +49 30 2093 4481. Fax +49 30 2093 4335. E-mail: stefan.gradmann@ibi.hu-berlin.de

Sjoerd Siebinga is a senior developer at the Europeana Office located in ther Dutch Royal Library. Contact: Europeana.eu, c/o the Koninklijke Bibliotheek, National Library of the Netherlands, PO Box 90407, 2509 LK The Hague, Netherlands. Tel. +31 70 3140681. E-mail: sjoerd.siebinga@kb.nl