Europeana Inside WP3 Iteration 1 Plan

# Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Revision** | **Date** | **Author** | **Organisation** | **Description** |
| V0.5 | 2012-12-24 | Neil Smith;  Rich Bruin | K-INT | Initial draft for partner contribution |
| V0.6 | 2012-01-07 | Sašo Zagoranski | SEM | Added development specifics for SEM |

# Tasks

## T3.1.1 CMS: ECK supporting functionality

### Partners Developing:

SEM, MON, SYS, MOB, SKI, KE, ZET, KUL, K-INT, ADLIB?

### Partners Integrating/ Using:

PS

### Summary of functionality:

This module covers all the functionality that must be provided within the main CMS in order for the CMS to be classed as ECK compliant for iteration 1. It includes functions such as selecting and exporting records in the relevant metadata profile to the core ECK system as well as reporting on the state of record selection and export.

### Technical Design:

#### Interfaces

The code for this section of the system is highly CMS specific with the exception of the interface with the ECK core. Therefore the only interface that is specified here is that the CMS is able to communicate with the interfaces exposed by the ECK core component.

#### Conformance Criteria

A CMS will be deemed to conform to this specification when it implements interfaces to satisfy each of the functional requirements listed in D2.5.

#### Implementation Expectations

Since this functionality is entirely contained within each CMS it is expected that each CMS will implement this section separately.

#### Outstanding requirements

The following requirements must be satisfied before this module can be fully implemented:

Export format for metadata from CMS into ECK must be specified (LIDO profile, etc.)

### Planned Steps - Development:

K-INT: Integrate ECK with CIIM middleware product in this iteration. Integrate with ODA (Culture Grid) at a later stage.

SEM: Develop a middleware layer, which will enable us to easily integrate the ECK into Galis (our CMS).

Other partners to add their development plans here

### Test Plan:

K-INT: Test integration with IWM, supplying via Culture Grid

SEM: Test integration between Galis and the ECK at all stages of development.

Other partners to add their test plans here

## T3.1.2 ECK Core

### Partners Developing:

**K-INT**

### Partners Integrating/ Using:

SEM, MON, PS, MOB, SKI, KE, KUL, ZET, ADLIB?

### Summary of functionality:

This module represents the core of the ECK functionality as it is integrated into the various CMS systems. It is envisaged that this system will be directly included within the CMS itself and will accept data from the CMS before passing it through the various ECK modules as appropriate and then onto consuming aggregators, etc. This module will also perform the stock mappings from the implemented metadata profiles into EDM ensuring consistency throughout the implementations and insulating the core CMS systems from changes to EDM.

### Technical Design:

#### Interfaces

The ECK Core will expose the following methods as interfaces to the CMS as part of iteration 1:

* List available functions (including those from modules)
* Call methods on modules
* Import record into or update existing record in ECK (this is likely to be expanded to allow publication, etc. in later iterations).

See figure 3 of D2.5 for details of interface behaviours.

#### Conformance Criteria

#### In order to conform to these specifications the module must implement each of the mandatory interfaces detailed here. In addition all modules that are to be integrated with the ECK core should be fully exposed by the ECK core for use by the CMS which submitted the request.

#### Implementation Expectations

It is expected that there will be one implementation of the ECK core which will then be wrapped in a RESTful HTTP interface to allow integration with tools written in other languages. This HTTP interface will not implement an event based interface but will rely on the caller polling for updates or on language specific SDK implementations which in turn wrap the HTTP interface if these are required by the various partners.

Future iterations of this component may require that multiple mappings are performed by the ECK in order to transform from the input data format to the output EDM (or possibly other output data formats). In addition it is possible that the mapping to be applied may not be chosen automatically by the software and that it must be specified by the caller. Although these are not requirements for this iteration they should be taken into account when implementing the component to ensure that they are possible for later iterations.

#### Outstanding requirements

The following requirements must be satisfied before this module can be fully implemented:

* Import format for metadata from CMS into ECK must be specified (LIDO and MARC in iteration 1)
* Report formats from the various modules must be specified

### Planned Steps - Development:

K-Int will develop a java implementation of the Core during January 2013. HTTP interfaces will be available for testing as follows:

07 Jan: Stub HTTP interfaces for all three interfaces listed above

14 Jan: Import / update records; List available functions internal to the ECK Core

21 Jan: List available functions provided by other modules

28 Jan: Call functions both internal to the ECK Core and provided by other modules.

Iterative improvements will be made in consultation with testing partners during February. The code base will be made available for porting to other platforms if required at the beginning of March.

Above dates assume implementation of mock external module interfaces by K-INT for initial testing prior to delivery of those modules.

### Test Plan:

K-INT: Internal testing as part of each development release. Testing against the CIIM tool during March.

SEM: Test calls of each functions provided by the ECK core.

Other partners to add their test plans here

## T3.1.3 Metadata profile definition

### Partners Developing:

**K-INT**, SYS

### Partners Integrating/ Using:

Other partners to provide their plans here

### Summary of functionality:

### This module will allow for the provision of multilingual definitions for the fields in the different profiles that are implemented. The caller will be able to request all definitions for a profile in a given language or the definition of a single field in a given language.

### In addition the module will enable the lookup of meaningful multilingual error definitions and guidance on steps to take to avoid them for use when explaining validation and other errors that have occurred elsewhere in the ECK system. For example validation errors can be returned as codes from the validation module and they can then be dereferenced in the appropriate language for the user using this module.

### Technical Design:

#### Interfaces

#### The module will expose the following methods as interfaces for use by other parts of the ECK:

#### List languages (all languages that are available at all in the system);

#### List profiles (all available profiles and the languages in which they are available);

#### Request definitions (all definitions for a profile for a language);

#### Request individual definition / guidance;

#### Request error definitions (all error definitions for a language);

#### Request individual error definition / guidance.

See figure 4 of D2.5 for details of interface behaviours.

#### Conformance Criteria

A CMS will be deemed to conform to this specification when it exposes each of the above defined interfaces for use by other modules and the ECK core / CMS components.

#### Implementation Expectations

It is expected that there will be a single implementation of this module that will be able to be shared by all users of the ECK but there may be more than one installation in order to allow for different deployment scenarios.

#### Outstanding requirements

### The following requirements must be satisfied before this module can be fully implemented:

### Each metadata format profile must be specified;

### The list of spoken languages which are to be supported must be defined;

### For each field in the profiles definitions and examples of use must be generated in each required language;

### For each specified error code (particularly possible validation errors) a meaningful error message and guidance on what it means and how to rectify it in the user’s data and mappings must be generated for each required language.

### It will be necessary to translate the various messages, etc. to be generated as a part of the project. These translations should be shared as much as possible throughout the project. Therefore efficient generation of the translations is very important to ensure that multiple partners do not perform the same translation unless there is a good reason for this to happen. Existing translation tools such as Transifex should be investigated to see if they can be used.

### Planned Steps - Development:

K-INT: Address outstanding requirements in January. Initial reference development in three languages (English, French, Dutch) for initial set of messages. Main development activity from February.

Other partners to add their development plans here if any planned

### Test Plan:

K-INT will work with SYS to test this module.

Other partners to add their test plans here

## T3.1.4 Persistence

### Partners Developing:

KE, SYS, SKI, ZET, ADLIB?

### Partners Integrating/ Using:

SEM, MON, MOB, PS, K-INT, KUL?

### Summary of functionality:

### This module enables the ECK core to communicate with the required data store. This is required for two reasons:

### To enable the ECK to save its state in non-standard database system if required;

### and

### To allow the ECK to use the CMS data store without worrying about conflicts with existing data structures and to allow relevant foreign keys to be linked as appropriate.

### Technical Design:

#### Abstract model

See figure 5 of D2.5 for a representation of the abstract data model. It is expected that there will be some extension required (for example to support revision histories) and possibly implementation specific fields will be added. It is also expected that individual CMS providers may want to link these entities to existing entities in their existing data store although that is not a requirement of these specifications.

#### Interfaces

The following methods are required to be exposed via the module’s interfaces for use within the ECK:

* Record entity:
  + Lookup by ID, CMS ID or PID
  + Create new
  + Save / update / delete
* Publication information entity
  + Lookup by ID, Linked record ID
  + Create new
  + Save / update / delete
  + Link to record
* Validation report
  + Lookup by ID
  + Create new
  + Save / update / delete (with relevant cascades)
* Validation data
  + Lookup by ID, Report ID, Record ID and by Record ID and Report ID combined
  + Create new
  + Save / update / delete
  + Link to validation report
  + Link to record
* Harvest event
  + Lookup by ID, IP Address, Date, Set spec
  + Create new
  + Save / update / delete
  + Link to record

#### Conformance Criteria

In order to conform to these requirements the module should:

* Fully model the data specified in the entity-relationship diagram above (although the precise model need not match the diagram exactly);
* Implement and expose each of the specified interfaces for use by other parts of the ECK.

#### Implementation Expectations

There will be multiple implementations of the persistence module. It is also expected that CMS vendors will either want to extend the standard module to interface with their existing data model or that they will write their own implementation to communicate with their underlying database systems.

#### Outstanding requirements

None

### Planned Steps - Development:

Partners to add their development plans here

### Test Plan:

Partners to add their test plans here

K-INT: Check that the module(s) work as expected with MySQL as reference database implementation

## T3.1.5 PID Generation

### Partners Developing:

**SEM,** MON, K-INT, SYS(?), SKI(?), KE(?)

### Partners Integrating/ Using:

PS, ZET, ADLIB?, KUL?

### Summary of functionality:

This module is in charge of creating PIDs for those records which do not already have identifiers which are globally unique and persistent. It will allow creation of PIDs based on institution URL (or some other identifier if no URL is available), record type and record accession number (or other suitable local identifier), combining these into a PID. Reverse lookup will also be supported allowing a generated PID to be separated and the constituent parts returned.

It is intended that the PID generated here can be used to drive a co-referencing service that allows links to be made between CMS IDs and other external IDs such as those exposed as part of an OAI-PMH server, generated by Europeana or other aggregators, etc.

### Technical Design:

#### Interfaces

The module will expose the following methods as interfaces for use by other parts of the ECK:

* Configure (for “built in” use case)
* Show configuration (for “built in” use case)
* Generate PID
* Reverse lookup PID (return constituent parts)

See figure 6 of D2.5 for details of interface behaviours.

#### Conformance Criteria

Implementations of this module will conform to the specification when they fully expose each of the above specified interfaces for use.

#### Implementation Expectations

It is expected that there will be multiple installations of this module either on an installation by installation basis (the “built in” use case), or per CMS, or shared between the projects (the “remote / shared” use case).

The module should be written in such a way that it is possible to ‘plug in’ external PID generators at a later date. This will allow the use of existing services such as handle or doi. The module should also allow for future cases where the institution involved does not have its own URL, or uses non-unique accession numbers, etc.

It is also expected that the implementation(s) of this module will take into account existing recommendations and standards for ID generation in this field.

#### Outstanding requirements

None

### Planned Steps - Development:

SEM to lead development of a shared module in collaboration with MON and K-INT.

SEM will develop a .NET reference implementation of the PID generation and expose the specified interfaces as RESTful HTTP services. The development will be done in the following stages:

1. Specify and document the PID generation interfaces
2. Develop the reference implementation
3. Create a RESTful HTTP interface

Other partners have indicated that they may be planning to develop on their own. If they are then they should provide plans here and should communicate with SEM to coordinate and standardise efforts.

### Test Plan:

SEM: Internal testing during development of the module.

Collaborative testing: SEM will publish the RESTful HTTP service

## T3.1.6 Preview

### Partners Developing:

**MON,** SEM, K-INT, SYS(?)

### Partners Integrating/ Using:

KE, PS, ZET, ADLIB?

### Summary of functionality:

This module allows for the generation of preview web pages to show how the user’s data will look when imported into Europeana (and optionally other targets e.g. intermediate aggregators, etc.). Given a single or set of metadata records and related media files the module will populate template pages for a sample hit list and record details page including thumbnails, etc. These previews will be packaged into a ZIP archive and returned to the caller or in the case where a web presence for the module is required the preview can be hosted directly and a link to the resource returned to the caller.

### Technical Design:

#### Interfaces

The module will expose the following methods as interfaces for use by other parts of the ECK:

* List preview templates
* Get preview template
* Upload / update preview template
* Apply preview template and return bundle (optional)
* Apply preview template with web presence

See figure 7 of D2.5 for details of interface behaviours.

#### Conformance Criteria

In order to conform to these requirements any implemented preview module must expose each of the mandatory interfaces detailed above. The optional interface can also be implemented, but without the other interfaces the module does not conform to these specifications.

#### Implementation Expectations

It is expected that there may only be one implementation of the preview module, or possibly one per implementation language in order to make the maintenance of the previews themselves simpler. It is also expected that the implementation may act as a façade around existing preview services such as the Europeana content checker if they provide the required functionality.

#### Outstanding requirements

The following requirements must be satisfied before this module can be fully implemented:

* Input format for metadata must be specified (LIDO profile, etc.);
* Preview template format which allows the creation of actual preview templates must be designed, specified and documented;
* The Europeana Content Checker and MINT tools should be fully evaluated before implementation in order to decide whether it can be made use of within the project rather than implementing the module from scratch. Current content checker not suitable for use in real time but code is available and could potentially be reused in a real time system.

### Planned Steps - Development:

MON to lead development of a shared module in collaboration with SEM and K-INT. MON to provide details here.

Other partners have indicated that they may be planning to develop on their own. If they are then they should provide plans here and should communicate with MON to coordinate and standardise efforts.

### Test Plan:

MON to provide details for collaborative testing

## T3.1.7 Validation

### Partners Developing:

**SEM, MON**, KE, K-INT, SYS?, ZET, KUL?

### Partners Integrating/ Using:

PS, ADLIB?

### Summary of functionality:

This module provides validation functionality for the ECK. It receives one or more metadata documents and media files if appropriate. The module then performs the following types of validation on the provided data:

* Schema validation against the specified metadata profiles;
* Checks that media exists if referenced;
* Checks that media is referenced if it exists;
* Check that referenced media is of a suitable size to fit with Europeana guidelines;
* Checks field contents for things that can’t easily be checked by schema validation including URI fields contain URIs, etc.;
* Option – Checks against ‘style guidelines’ for the profile – “rules of thumb” checks that titles aren’t too long, etc. which may suggest that data from the wrong field has been mapped.

A validation report is then returned to the calling system which can be parsed and presented to the user and the persistence layer is used to annotate the record with the validation results. This report will contain validation error codes which can then be dereferenced for each required language using the profile definition module.

### Technical Design:

#### Interfaces

The module will expose the following methods as interfaces for use by other parts of the ECK:

* One by one validation
* Batch validation

See figure 8 of D2.5 for details of interface behaviours.

#### Conformance Criteria

Implementations of this module will be deemed to conform to the specification when they fully implement each of the interfaces detailed above.

#### Implementation Expectations

It is expected that there will be one or two implementations which are then wrapped and exposed using an SDK in each implementation language. This will keep the number of variations to a minimum and will help to ensure that validation is consistent across ECK implementations in general.

#### Outstanding requirements

The following requirements must be satisfied before this module can be fully implemented:

* Input format for metadata must be specified (LIDO profile, etc.);
* XML Schema for the profile must be implemented (for schema validation);
* Conditional and specialist validation must be specified (i.e. validation that cannot be represented using XML Schema, etc.);
* ‘Style guidelines’ must be determined (optional).

### Planned Steps - Development:

MON and SEM to independently implement LIDO validation with KE and K-INT providing support and assistance. MON and SEM to provide details here.

SEM and MON will create the specifications and then develop two reference implementations of the module. SEM will develop the module in .NET and MON in Java(?).

The development will be done in the following stages:

1. Specify and document the Validation module interfaces
2. Develop two independent implementations
3. Make the libraries available as a library/SDK
4. Create a RESTfull HTTP wrapper for the library and expose the module as a web service

Other partners have indicated that they may be planning to develop on their own. If they are then they should provide plans here and should communicate with SEM and MON to coordinate and standardise efforts.

### Test Plan:

SEM: Internal testing during development of the module

Collaborative testing: Testing by all partners, once the SDK and the web service has been published.

SEM and MON to provide details for collaborative testing

## T3.1.8 Supporting tasks

### LIDO profile definition

K-INT to set up a working group of interested parties (including Content Providers and Technical Partners) in collaboration with KMKG.

### MARC profile definition

KUL to develop profile with involvement from any other interested parties.

### Module report format definition

K-INT to provide module interface and report framework in early January.

### Programming language / communications coordination

All partners developing components (including CMS integration) to provide details of the languages they will be using

K-INT: Java / HTTP (REST)

SEM: .NET (for native integration); .NET/HTTP (REST) (for web service integration)

All partners planning to integrate components to provide their language requirements.

K-INT: Java / HTTP (REST)

SEM: .NET