EuDML | The EUROPEAN DIGITAL MATHEMATICS LIBRARY

ICTPSP

DELIVERABLE

Project Acronym:	**EuDML**
Grant Agreement number:	**250503**
Project Title:	**The European Digital Mathematics Library**

Deliverable 4.4 – The EuDML Information Life Cycle Process

Revision: 1.0

Authors:

José Borbinha (IST)

Aleksander Nowinski (ICM)

Gilberto Pedrosa (IST)

Krzysztof Wojciechowski (ICM)

José Delgado (IST)

Project co-funded by the European Commission within the ICT Policy Support Programme		
Dissemination Level		
P	Public	PU
C	Confidential, only for members of the consortium and the Commission Services	

Revision History

Revision	Date	Author	Organisation	Description
0.01	12.03.2012	José Borbinha	IST	First structured draft
0.5	13.03.2012	Aleksander Nowinski, Krzysztof Wojciechowski	ICM	Complete text, no flow, needs redaction
0.6	23.04.2012	Gilberto Pedrosa	IST	Review, formatting
0.7	30.04.2012	José Borbinha	IST	General review
0.8	02.05.2012	Krzysztof Wojciechowski	ICM	Polishing the text. Revised flows
0.9	02.05.2012	Gilberto Pedrosa	IST	General review
1.0	28.05.2012	José Borbinha Krzysztof Wojciechowski	IST ICM	Appendix added (more precise workflow diagram)

The document versions are controlled by the EuDML SVN version control system. The above revisions are for informational purpose only and do not reflect the actual SVN versions.

Statement of originality:

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Contents

1. Executive Summary

This document describes the basic information entities and their lifecycle within the EuDML.

The EuDML system is a XML-document oriented system, with a domain made of multi-part records, allowing storing multiple versions and formats of both content and metadata.

The most relevant entity in this domain is a publication (which can be an article, a proceedings article or a book).This document focuses on the lifecycle of this entity.

Other top-level entities are users and annotations, which lifecycles are described in separate documents (D4.2, D9.3).

One of the key issues for entity lifecycle management is a proper identifier management, to ensure that requirements for proper identification of the data provider and the described works are met. A special identifiers service and a mechanism based on a multi-type identifier preservation is adopted to ensure proper identifier management.

A separate section is dedicated to explain the de-duplication procedures and algorithms, very important issue for the EuDML.

The final part of this document describes the detailed workflow of the publication lifecycle, covering import (harvesting), de-duplication, content management, enhancement, indexing and other relevant operations.

2. Introduction

This document describes the main information entities in EuDML, their representation and lifecycle. It covers the most important entities, describing exactly what and where happens to the information within the system. It represents the current implementation state, and decisions, as well as short term plans for the system development. The status of the implementation is clearly denoted, whenever applicable.

EuDML is a system oriented in a first place for providing information about and access to publication resources to the user. The primary goal is to create a modern and efficient Digital Mathematical Library, which just will be really comfortable in daily use to the mathematicians. Then, on a second purpose, it aims to make this library a place for cooperative work and information sharing, which will allow exploiting the synergy effect and benefiting from the users effort.

EuDML data comes from various data providers, with different levels of quality both in terms of metadata and content (these concerns are addressed in the WP3). In this scenario it is expected to find publications which have duplicates among the sources. In general case this can be not easy to do.

EuDML was designed as a document oriented system (D4.2), which operates mainly in terms of records representing publications. The objects within the EuDML system are represented as XML documents, according the defined schema (EuDML profile of NLM).

The shape of the EuDML profile of the NLM schema has some impact on the representation and lifecycle of the entities. Especially significant is the adoption of virtual representation for higher-level objects, like journals and multi-books.

3. Key concepts

The system information management is based on a few key concepts, which are described below. These concepts explain the basics of the EuDML information management.

3.1. Authoritative and derived data

Two key concepts are authoritative and derived data representations. Within the system a number of data sources is used, each represented by a specialized service. Examples of such services are data storage, browsing service and search index. Each service contains usually relevant information concerning the entities in the system. Therefore, a problem is that the same entity may be represented as an XML NLM file (in the storage), as a set of browsing relations and also as full-text index entry. Therefore, there are the concepts of authoritative data and derived data: an authoritative data representation is the data as it was provided by its original source, while a derived data representation is the view of the data as it is represented in a specific specialized service.

Authoritative data sources are:

- EuDML Storage - for publications in NLM format
- Id Service - for persistent identifiers management
- User Service - for user data
- Annotation Service - for annotations
- Metadata Registry - for the collection and data sources

Derived data is stored within:

- Browsing service
- Full-text indexes
- Math formula indexes
- Semantic similarity service (gensim)

3.2. XML metadata representation

XML is a natural format for metadata exchange and the project has made a decision to store all the metadata in the system in form of XML records. This has important impact on the system architecture. It allows the project to keep really complex metadata and enrich it, if there is a need. The update of the schema is relatively inexpensive, and does not have major impact on the services. Also entity representation may be much more complex, including a lot of fields and sub-relations, which would have to be mapped to a large number of tables in the RDB engine if such model of storing date would have been chosen. Only a few of those relations are actually required during normal operation of the system, and most of them come into play only when record is presented to the user.

This decision has some drawbacks: a separate set of services has to be created and maintained to represent relations in the system. These services are described in D4.2 in details.

3.3. Virtual entities

This is typical, that format used to transfer and store records are self-embedded. It means that for a publication all the information about a publishing path (a journal description, series, volume

and issue) is included in the data describing publication. This makes handling of records simple, but has a drawback, as there is no actual record for higher level entities, like journal. This implies that such entities must be deducted from the lower-level entities, and have no direct representation as authoritative data record. They exist only as derived data and therefore are called virtual entities.

3.4. Duplicate handling and merging

Ii is expected within the data there is a significant amount of duplicated publications. By duplicate, it is considered two or more records representing the same publication coming from different sources.

The duplicate handling is a general procedure of management of duplicates. It includes detection (early and late) of the duplicated items, as well as a merging procedure. The merging procedure has two key aspects: merging the records, while preserving identifiers, and merging the metadata itself.

3.5. Source data preservation

The harvesting is an expensive process – only in terms of the metadata, it can take days to aggregate the whole collection, and the aggregation of the content may consume weeks. Therefore, as the disk space is less and less expensive, EuDML adopts a general policy, that whenever the project gets some data (metadata and content) from somewhere, the original forms are always preserved, along with the transformed, converted or processed data. Processing of the whole corpus takes time and actually most of the tools sometimes fail, also it is expected that during life of EuDML most of used tools will be improved to deduce more information from the original data. For those reasons the policy of preserving any source data that was introduced into the system has been imposed.

4. Key entities

There are a number of the entities within the system, on a number of levels of the abstraction. The following section summarizes most important of them.

4.1. Real entities

The real entities are the ones that have direct representation within the system, and are natural entry points for the systems domain.

4.1.1. Publications

Publications are the most important entities. Each publication has specific, unique and truly persistent (during the whole lifetime of the system) EuDML identifier. This identifier is used to locate the publication within the system. Currently the following types of the publications are defined:

- Journal articles
- Books
- Proceedings articles

The contents of the publications are the true information represented by the publications. The content is a general name for all the types of the publication text. A basic type is a PDF file, but under this term a plain-text representation for the full-text to be indexed, as well as other forms of the content is understood.

4.1.2. Users

Users are representation of the user accounts for the system. The user entity encapsulates the login data (credentials, email etc.), user profile, preferences etc.

4.1.3. Reading lists

Reading lists represent the concept of the collections created and owned by a user. They can be shared among the users.

4.1.4. Annotations

Annotations are entities created by the user to any of the other information entities. They have various types, like comments, corrections, topic suggestion etc.

4.1.5. Repositories and Data Providers

Repositories and Data Providers are entities reflecting sources of the data, as represented within the harvesting system, the REPOX. These entities come along with the transformations, representing necessary conversions needed to ingest data into the system.

4.2. Virtual entities

Virtual entities are entities that have no direct representation within the authoritative data. They are deducted from real entities. The virtual entities may have EuDML identifiers to ensure, in

most cases' also stable, URIs in the system. Following types of the virtual entities are available within the system:

- Journals,
- Multi-books
- Conference proceedings volumes
- Classification subjects

4.3. Implied objects

Within each publication there is a number of entities, which define the publication. All those entities are currently isolated within the publication, but in some cases they are used as a reference to the external world. Those entities are, for example:

- Authors
- Citations
- Affiliations
- Keywords

In case of the citations, they are represented in a complex way, which may contain a reference, within the EuDML, to the cited object (if present).

5. Publication representation

The representation of a publication within the system is quite complex, to ensure that all requirements are met. However, the used solution is very flexible, and it is easy to adopt in case schema is updated to support new kind of information.

Publications within the system are represented as compound records within the EuDML Storage Service. The Storage Service basically is a large hash table, which stores records identified by unique identifiers. It offers only get/put/update and basic iteration over the records (see D4.2 for details).

Each record in the store is identified by unique EuDML identifier. Record is a complex structure, which is composed of a number of parts. Each part is a binary array or a text string (may be interpreted as file), and is identified by partID, unique within a record (and typically shared among the records). Typically partID have path-like form, which allows creating a tree structure of the record parts. For example prefix '/src/' is reserved for source metadata, and '/src/nlm/...' is adopted for source metadata from various providers. This structure allows EuDML to store a number of the data parts, for each publication. Whenever comes a need to store a new type of information, it may be easily added.

First of all, the metadata is stored in the system as text strings (files). For each record there is a source metadata in the original format as provided to the system and the same source metadata in NLM format, created after conversion. This metadata includes ZBMath metadata as well, as this metadata is treated as another format of the source metadata. Then there is one metadata part in NLM format, which is a result of merging of all source metadata into one best possible record, resulting in the so called 'base NLM'. On top of that there is an 'enhanced NLM', which is the base NLM after all enhancement procedures are applied. There is always exactly one 'base NLM' and zero or one 'enhanced NLM'.

For final usage within the system (indexing, presentation, deriving virtual entities etc.) only one representation is used, so called 'used NLM', which is either enhanced NLM (if exists) or base NLM (otherwise). This way whenever there is a need to update the record, as all metadata is stored in the system, all necessary steps can be performed within the system.

The content is stored within the same record together with the metadata. The following types of content are currently stored (the possible multiplicity of content for each resource is also indicated):

- Source full-text (typically pdf-s from provider) [0..*]
- Source full-text for indexing (latex or MathFullText) [0..*]
- Accessible content, which may be served by the system (as generated by accessibility toolkit) [0 or n], where n is defined in WP10.
- Full-text formats, used for enhancements and indexing:
  - Latex (obtained from math-ocr or maxtract) [0..1]
  - MathFulltext (extracted from latex, pdf text extractor or any other tool)[0..1]

Those records are persisted and managed within the store service. Internally, within the system, these records are read from the store and converted to YModel objects, which is a native java representation of ICM BWmeta standard. The YModel objects are abstract enough to ensure proper representation of the NLM data.

6. Identifiers

The EuDML is expected to become an authoritative source for mathematical publications from Europe. Except from collecting the resources, it shall provide reliable and usable methods for referring to its contents. This assumption has the following impact on the EuDML Identifiers (eIDs):

- eIDs must be unique
- eIDs must be persistent
- eIDs must be compact (can be used in paper form), applies only to a citable items (papers)
- URLs must be explicitly derived from the eIDs

Those requirements refer to publications only, as the publications can be cited by other sources (like articles). They do not apply to the annotations, users etc.

Compact eIDs imply that it is not possible to use UUIDs or checksums of specific objects to assign publication identifiers. Therefore, the identifiers for papers are assigned as subsequent integer numbers (starting from 10000, to preserve 'gold' identifiers to be assigned by hand later).

eIDs are URNs with the custom namespace "eudml". The custom string of the URN is composed of two parts:

- Object type (doc for paper, an annotation etc.)
- Object identifier (a string in format dependent on the object type)

Therefore, the format for eIDs is: urn:eudml:<type_id>:<object_id>.

An example may be the paper identifier: urn:eudml:doc:12345. Please note, that other entities do not require compact identifiers (like annotations or users), so it is possible to use UUID or random strings for the identifier part in case of such objects.

The requirement for persistence is understood in the sense that once assigned an eID is never reused, and even after a long period of time a paper will be accessible under this eID.

As the identifiers are short, a special service was introduced which ensures, that each paper will have the same identifier assigned in the harvesting/import process each time such process is performed. The service is based on the document identifiers (provider's internal identifiers, Digital Object Identifier – DOI, ZBMath identifiers etc.).

Whenever a publication is imported, the id service is consulted, and asked whether a publication with its ID(s) has already identifier assigned. If yes, then the same identifier is used, otherwise a new one is assigned. This allows early deduplication, as if identifiers within the document (let's say – DOI) point out that document already exists in the system, then the new document is merged with the previous one instead creating a new instance.

For virtual entities the identifiers are assigned based on specific rules, special for each entity type. These rules exploit whenever possible existing identifiers, so for example journals have identifiers based on ISSN (when available).

7. Annotations

In the EuDML system annotations can be added to any entity. They are various types: notes, suggestion a subject, correction to metadata etc. Detailed description can be found in D9.1. The annotations allow exploiting synergy of users of the system.

The implementation chosen to use in EuDML is based on RDF triples. Therefore there is possible to use semantic inferring when exploiting existing data. Such exploited data can be later presented to the user.

8. Content management

Originally it was assumed that EuDML will not serve the content and the project will not have to manage licensing issues directly (see D4.1). But over the time it comes clear, that due to achievements of the WP10 the project will have to operate on the full-text and serve it to disabled users.

The information in the EuDML has various providers and quality. Most of the publications (>95%) within EuDML are open access, but it does not imply, that we have right to re-serve them to the third parties. Therefore there was need to adopt NLM to allow expressing licensing rights. This has been achieved by adding basic licensing information about the full-text, which clearly states, which operations are permitted on the full-text (analysis, indexing, re-serving accessible version).

8.1. Content types

Content is a general concept, which describes various types of full-text content for the publication, which may appear in the EuDML. The following types of the content are used within the system:

- Full-text document – the content in the document form, which contains publication in the original form. Typically PDF file (or files, in case of book split to chapters). This is used to analysis, text extraction and generating accessible version of the documents.
- Source-text – the source of the document (typically latex file), which was used to generate original document. Barely used at the moment, and hardly ever provided. It may be used for indexing, but varieties of source formats make automated extraction of the metadata difficult.
- Full-text for presentation – occasionally there is a tagged full-text, which is suitable for the presentation, typically in form of XHTML annotated with MathML equations. This type of the content is encapsulated within NLM for transport, and later extracted and kept separately within the system due to performance reasons.
- Accessible full-text document – WP10 toolkit is capable of providing reliable accessible versions of (certain) PDF documents. Those documents are generated out of the original PDF file, stored and served when required.
- Index math full-text – a special type of document, where formatting is lost and only plain text and MathML formulas are present. This document type is used only for full-text and math search indexing. The project assumes that it does not have to be accurate (actually it may be really wrong, as it sometimes comes out of automated OCR), but it is used to improve search capability of the system. Presence of the math formulas is not required, but desired.
- Index latex – in a few situations a latex file may be used in some intermediate stage, especially as the output from the INFTY math OCR engine. Those files differ from the source latex, as they have significantly lower quality, and they are appropriate only for conversion to index math full-text files and indexing.

The creation and usage of each content type is described in the workflow section of this document.

9. De-duplication

This section describes part of the system which is currently under implementation, or will be implemented in the following months.

De-duplication is the process of identifying and handling duplicated documents within the system. It is very important to identify duplicates accurately. Keeping duplicated items significantly worsens user experience and degrades automated analysis results, while false duplicate identification de facto deprives library of some items valuable for the user. Identifying duplicates has potential complexity of $O(N^2)$ and therefore it must involve proper algorithm to avoid this trap. It also has to be understood, that algorithms for the duplicate identification may be improved over time, and may return different results when new information (e.g. corrected item, a new version of the item from ZBMath etc.) appears.

De-duplication is a complex, two-stage process in EuDML. It is also complicated by the fact, that the project wants to ensure, that whenever an EuDML identifier is assigned and (somehow) published, it should always point to the same resource.

9.1. Early de-duplication - identifiers based

First stage is so called 'early de-duplication', where items, which can be easily identified as duplicates are detected before assigning EuDML identifiers. This has to be done with 100% accuracy, so no sophisticated algorithms are needed here. Unique identifiers assigned to the items (DOI, Zentralblatt, Math reviews etc.) are used at this stage, and if identifiers in two items match, then it is assumed that one of the item is a duplicate of other item already existing in the system. In those cases, a new metadata part is just added to the existing item, as one more source for it. This way will ensure that duplicates which are easy to identify do not consume the identifier space, and never pollute the information space of the system.

9.2. Late de-duplication and duplicated item management

In the second stage, 'late de-duplication', runs the duplicate identification algorithm against the whole collection, and the algorithm offers possible duplicates, which are selected and applied to the system. This algorithm is currently tested, and it will be adapted to the EuDML shortly. To avoid relocation of the identifiers, whenever algorithm detects a duplicate, a new item is created, and new EuDML id is assigned. Then duplicated records are marked as replaced by the new item, and all the source data is stored in the source area of the newly created record. Then normal procedure form merging is applied. If one of the former records is referenced from outside (for example user follows the link from the other system), then it is redirected, with proper http redirection code and message to a new record location. This is very important, as the merge decision may be proven to be wrong over time. In such case older records are revived and restored to the original form, and in the wrongfully merged item special record is placed denoting this fact. So whenever user will navigate to the invalid merged record, he or she will see page, which rightfully describes the resource as 'invalid merge', and will present links to the proper resources. This way, whenever EuDML item is cited by outside world a user will be able to navigate to the proper item, and he or she will have clear information about the object.

9.3. Merging algorithm

Whenever duplication occurs, detected by the early or late de-duplication, and if the external metadata (typically ZBMath record) may be obtained for the item, then custom merging algorithm is used. The source parts are stored within the record, and then one base NLM part is

created, which is base operation of the system. The merging algorithm is not a complex from the programming side of view, but it implies some decisions on proper election of the field value based on the record provenience. As this is strictly political issue, an algorithm will be pluggable and it will be possible to adopt different version of the algorithm easily, to achieve consensus among the data providers. This algorithm at the moment has only very simple implementation, which adopts 'first wins' policy, and only missing fields are filled in with the contents of the other record versions.

10. Basic publication processing workflow

This section describes the whole workflow of the processing of a publication (metadata and content) within the EuDML system. That occurs mainly in two steps: the initial harvesting and then the remaining lifetime within the system.

Figure 1 and Figure 2 provide informal views of this workflow.

10.1. Harvesting

Most of the described workflow is currently in the final stage of the implementation, but it is not operational in the current release (1.2) of the EuDML.

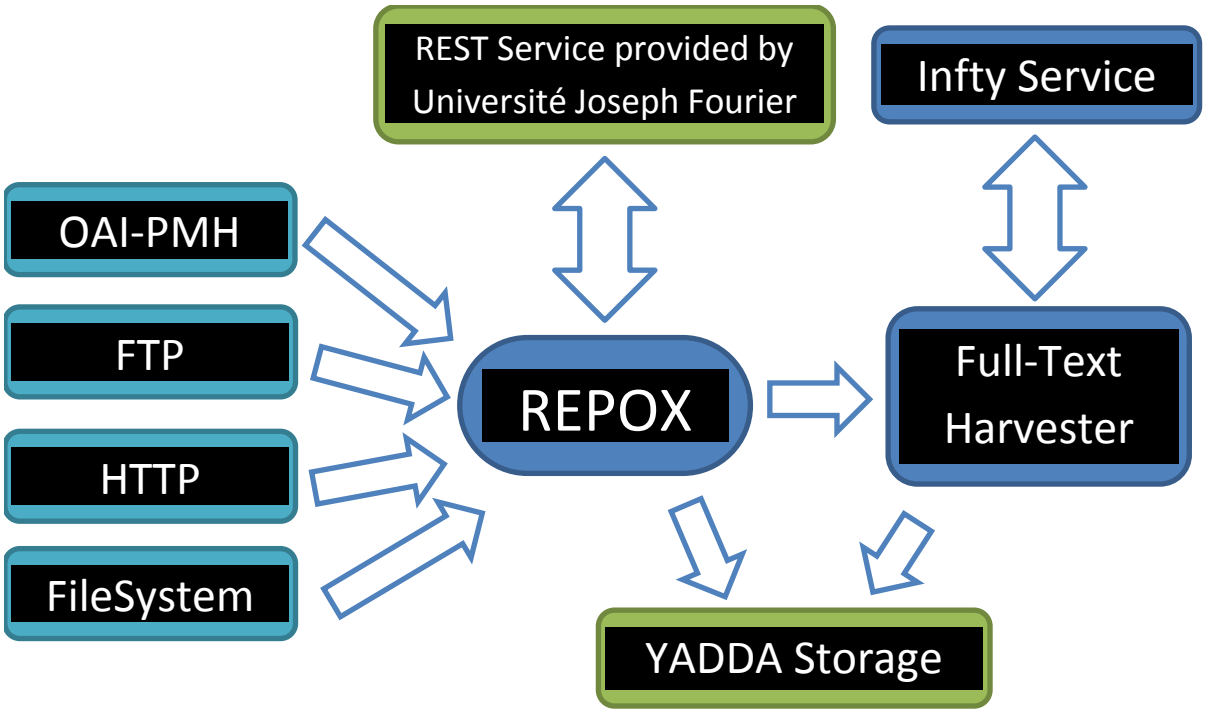

Figure 1: Informal high level view of the workflow for a publication within EuDML

Harvesting is the process of obtaining source data from the providers. The workflow of the harvesting is as follows:

1. Harvesting source records from the data providers using REPOX through one of the following protocols or techniques: OAI-PMH, FTP, HTTP, File System Transfer;
2. Converting those records to the desired NLM format using the REPOX metadata transformation service;
3. Performing initial enhancements (conversion of the math formula etc.) to the records to conform EuDML best practices, as developed in the scope of the WP3 (performed by the REST service developed by the Université Joseph Fourier and used by REPOX);
4. Harvesting the full-text content (Full-Text Harvester), creating the LaTex files (by the Infty Service) and store the results in the EuDML storage system (YADDA Storage)
5. Perform early deduplication and select proper EuDML identifier for the item, which may be:

a. The same as a previous version (if it is an update)
b. The same as other version of this document (if there is an early duplicate detected)
c. A new EuDML identifier otherwise

6. Store the NLM and source record within EuDML Store
7. Fetch the content files, if referred, and licensing allows processing them done by the extensions service for REPOX)
8. If appropriate:
   a. Generate LaTex content by the INFTY math OCR engine for indexing
   b. Store the LaTex content within the EuDML Store.

This concludes the part of the workflow, which is applied during harvesting processing. After this, it is assumed that the proper version of the records, with NLM source matching quality requirements as defined in the WP3 are present within the EuDML Store and are ready for further processing.

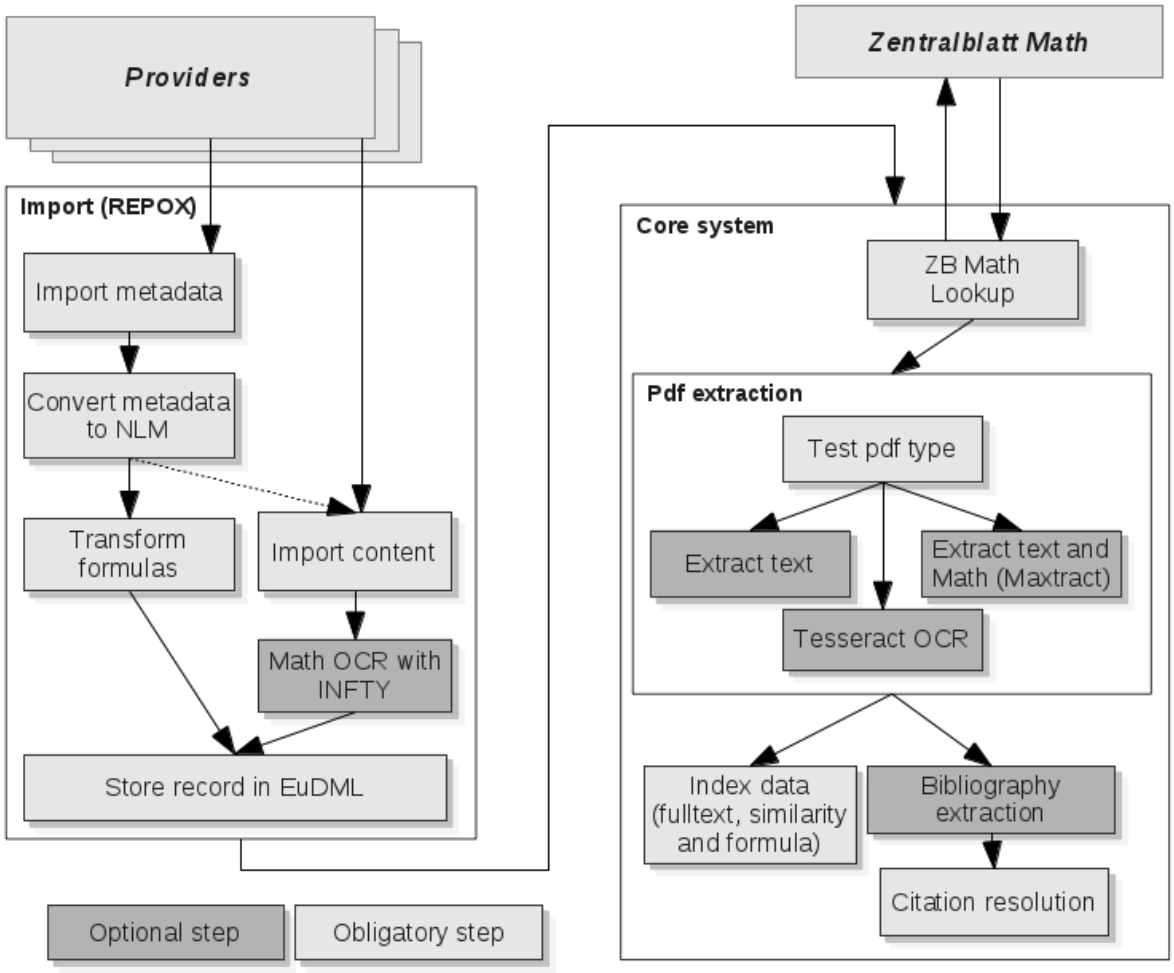

Figure 2: Informal detailed view of the EuDML external data processing workflow

10.2. Record processing within EuDML

Apart from the de-duplication and merging, most of the described workflow is already implemented and operational in current release of the EuDML system (Version 1.2). Applying a de-duplication procedure on the various NLM sources, to obtain base NLM, will be base for further processing.

For that, first the ZBMath lookup step (see D7.3) is performed, and the Zentralblatt record is downloaded for the publication (if found). Next the late de-duplication step is executed. After this, a merging procedure is performed (on various source NLMs and possibly ZBMath metadata), to obtain the base NLM, which is base for further processing. The latter step may be applied multiple times, depending on the needs.

The next workflow step is processing the content to obtain full-text in best possible quality. This step is skipped if:

- there is no content
- a source form or plain full-text is provided
- the INFTY engine already produced a LaTex file

The content file (PDF) is tested with the "pdf tester", and then a proper text extraction method is applied, which may be:

- Simple text extractor from ICM, which extracts text only using PdfBox
- Maxtract from UB, which extracts both text and math formulas
- OCR-ing using Tesseract

After this a normalization of the present text is done, to ensure that proper math full-text file is available in the system.

Next stage is the attempt to recreate the citations from the extracted text and add them to the publication metadata (NLM record).

Finally the identifiers from ZBMath and MR (using the matching references services provided by these entities) are analyzed to add external links to document references (and present them in the UI).

When this workflow is finished the record may be properly analyzed and added into the remaining system services namely:

- Indexing the metadata
- Indexing the content for full-text search
- Adding record to the similarity engine
- Storing record relations in the browse service to enable browsing
- Creation or update of the virtual entities described by the publication (i.e. ensuring journal correctness based on the information in the article metadata)

The last step, which is run periodically on the whole collection, is to (attempt to) resolve the citations for all the items within the system.

The workflow here described covers the publication lifecycle. Basically EuDML assumes that whenever some item is properly incorporated into the system it will stay there forever.

11. Conclusions

As a document oriented virtual library, EuDML is focused on the information representing single publication as the most important resource for the user. Most of the other information entities are derived directly from the publication itself. It is assumed that provided information processing from the raw data, and be able to repeat it when necessary, ensures proper quality and robustness of the system. Proper representation of the publications, which allows storing multiple versions of the same object within the same record, had proven already its value and seems to be a good strategy to maintain a complex document oriented system. At this point most of the described concepts are already implemented, and working, and only minor part has yet to be developed or integrated.

Appendix - Overall EuDML workflow

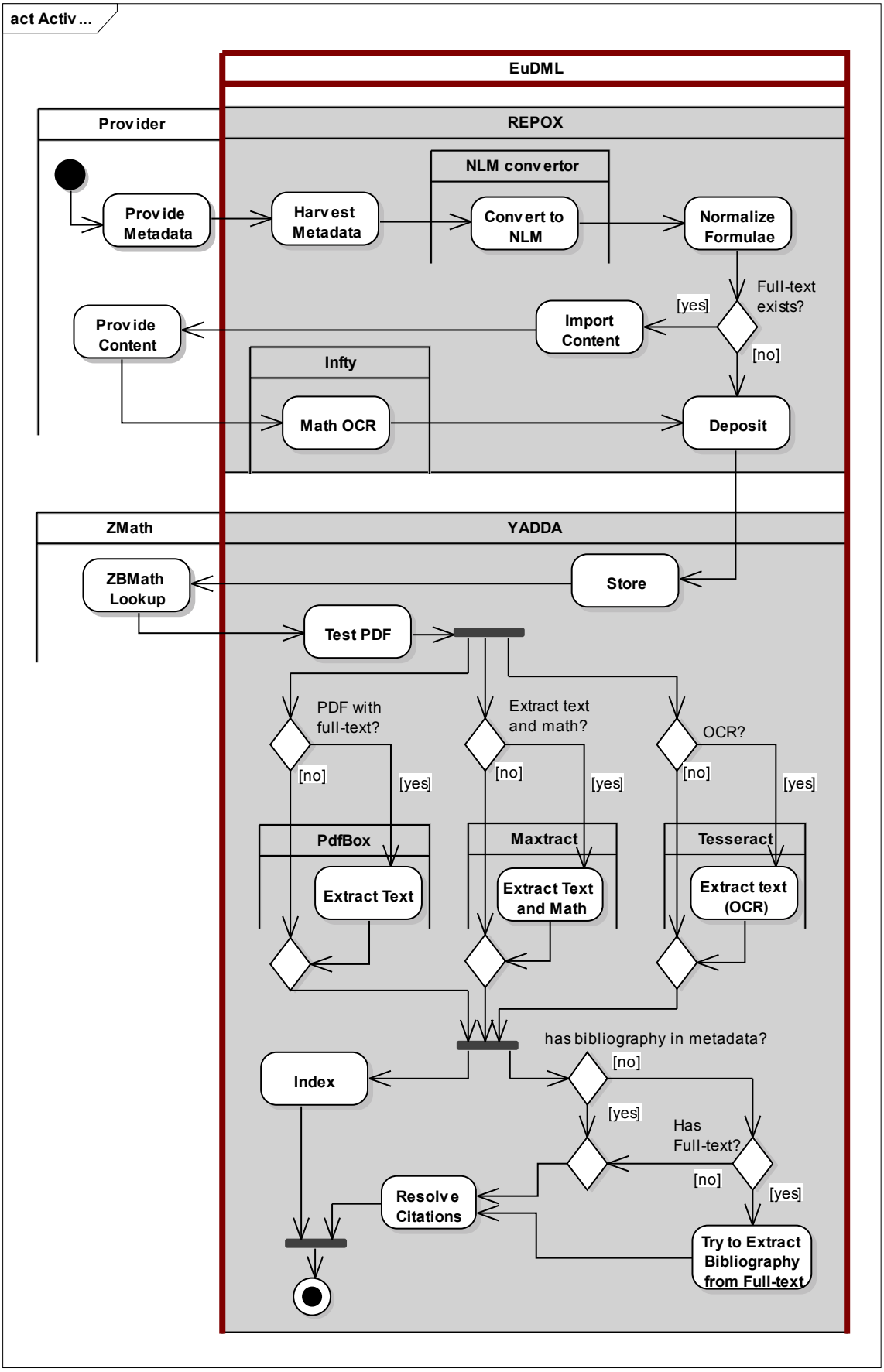

Figure 3: The EuDML external data processing workflow (detailed UML activity diagram)