

LEAF Newsletter

In this issue:

- [Editorial](#)
- [Presentation of the LEAF prototype](#)
- [LEAF central system](#)
- [The LEAF maintenance suite](#)
- [Encoded Archival Context \(EAC\) – The Story Continues](#)



Linking and Exploring Authority Files

Editorial

The LEAF consortium is pleased to present the second issue of the project newsletter. It has been a long time since the first issue was published and since then many new developments have taken place. This current Newsletter focuses on the most important aspects of last year's project work.

First of all, and in order to present a general overview, we describe the most important aspects of the LEAF prototype: the infrastructure that is a framework for the services the consortium will offer to its various user groups.

In deeper detail we take a look at the core of the development, with a view on technical details of the LEAF central system, comprising the main data repository, web user interfaces and facilities for converting records between formats.

Another important component is the LEAF Maintenance Suite, responsible for managing administrative information about LEAF data providers.

Finally we conclude with the second part of the EAC story started in the first issue of this Newsletter. Recent developments in the world of EAC are briefly described, the importance of EAC in LEAF as a common exchange format between different record formats is elaborated on.

As usual, we thank you for your interest in LEAF. Comments or questions are very welcome, please send them to: co-ordinator@sbb.spk-berlin.de.

In order to keep up-to-date in between Newsletter issues, please make sure to visit the regularly updated LEAF website at <http://www.leaf-eu.org/>.

Victor Herreros, Complutense University, Madrid.

Presentation of the LEAF prototype

Kurt Majcen, JOANNEUM RESEARCH, Graz

The purpose of the LEAF system is to provide access to different (national) authority files via a common access point and to show cross references between those authority files. Several kinds of services are part of this overall scenario. These are applicable to different user groups: public end users may want access to information about persons or specifically refine search arguments required for other applications; professional users (e.g. librarians, archivists) may want access to rich authority information in order to improve the quality of their own authority data; other users, like service providers (e.g. manuscript dealers), may want to provide information about their holdings in conjunction with the person descriptions available in the LEAF system.

To cover all these different needs, the system foresees several modules (see figure 1 below) allowing a variety of operations: modules to harvest local authority files, to link records describing the same person and to store the data in a central repository (referred to as offline components); user interfaces which allow browsing, searching and annotating separate or linked records which can be found in the LEAF system (referred to as online components); a conversion unit for the conversion between local formats and the common XML exchange format used in the LEAF system ([EAC](#)); a Maintenance

Suite to administer the organisations providing data to the LEAF system and to monitor the data providers' server systems. Interfaces for external systems are offered to make LEAF an attractive service for other systems like [MALVINE](#).

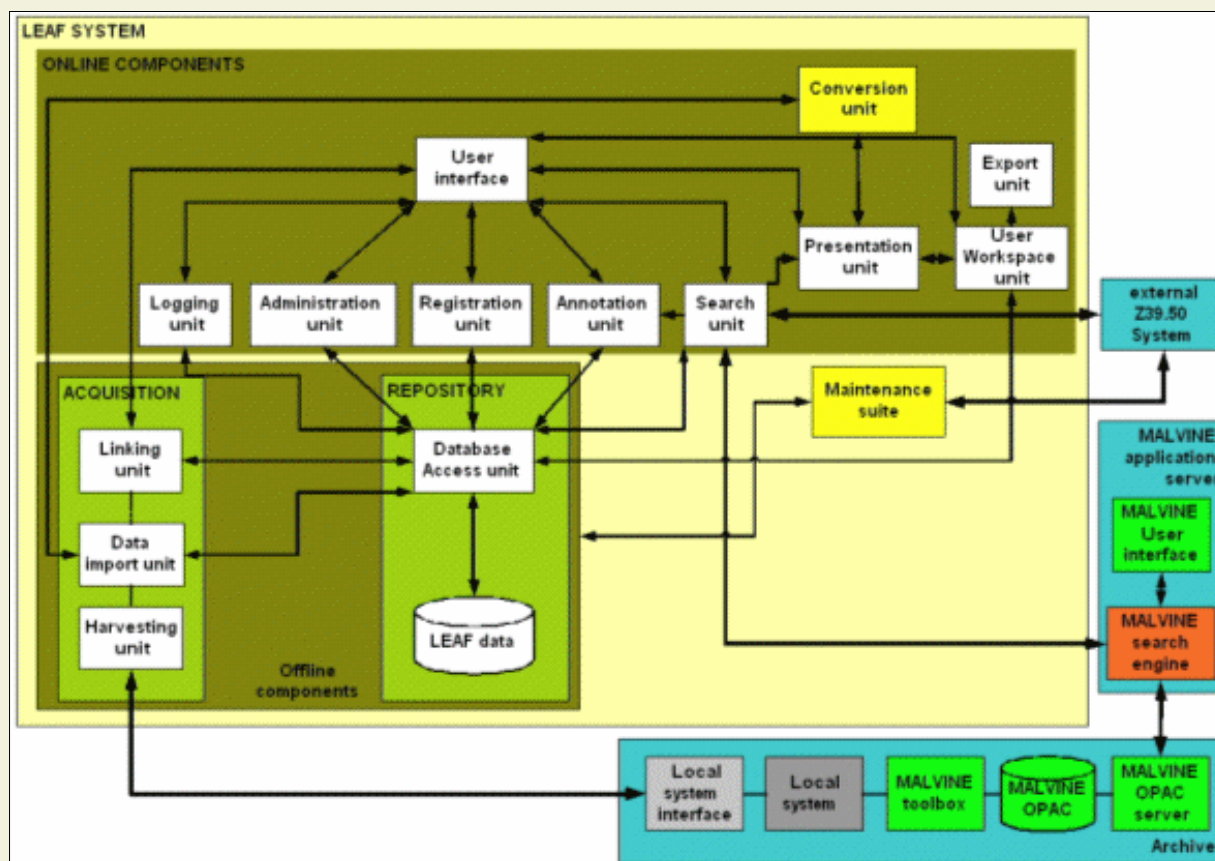


Figure 1: Architecture of LEAF

Since different organisations are involved in the technical development tasks, it is not surprising that different technologies are in use. The central system is mostly focused around the [Java](#) technology, the conversion utilities are, on the one hand, scripts written in [Python](#) and [Perl](#) and XSL style sheets on the other. The Maintenance Suite consists of a web application ([PHP](#) scripting and a relational database) and some Java programs for the part continuously monitoring servers. As a consequence of the different technologies in use various interfaces between the separate system components exist. These interfaces provide a rather loose coupling of the different parts thus allowing for a very modular and independent approach towards the development work on all sites; at the same time the system is expected to be flexible when there is a need to change one or more components in the future.

LEAF Central System

Kurt Majcen, JOANNEUM RESEARCH, Graz

The central LEAF system comprises a repository with appropriate linking and access mechanisms, the harvesting mechanisms to fill the repository, a web user interface being the public entrance to the system and facilities for converting records between formats. All aforementioned components are hosted on a PC with a LINUX operating system.

The main functions of the system available to the different user groups (public users, data providing institutions and their staff members, service providers like research projects and memory institutions not providing data to LEAF) via web user interfaces are:

- Search for person descriptions
- Search for linked person descriptions
- Annotate records
- Search the annotations
- Access to a user workspace
- User registration
- Search for users
- Login / logoff

All retrieved search results receive, behind the scene, the status indication of a Central Name Authority Record thus allowing for the identification of records that were relevant for users.

Additionally numerous processes are necessary to keep the data in the system up-to-date. This means that data are regularly harvested from local data providers; the harvested data are converted into one common exchange format (EAC XML), the EAC records are inserted into the LEAF database where they are then linked. Insertion of data providers' records into the database can mean either of 3 things: insert, update or delete. Regarding the update of records in the database a few actions have to take place because of the reasons previously mentioned. These actions assure consistency between modified data and other information in the system like users' annotations or IDs of records extracted from the system when being used via the web user interface.

Several technologies are used for the production of the central LEAF system:

- The repository and its mechanisms for importing the harvested data make use of an [ORACLE](#) RDBMS and a range of PL/SQL stored procedures.
- The [JRun](#) application server is acting as the web server which hosts the LEAF system web site.
- On top of the application server the [Cocoon](#) XML publishing framework controls the flow between the various requested URIs and also provides conversions via style sheets for output of the web pages. Working with Cocoon includes mainly using technologies like Java, XML and XSL.

For the communication between the modules of the LEAF system and the systems of the LEAF data providers and of potential future participants (i.e. the harvesting processes) some communication protocols (FTP, [OAI](#), [Z39.50](#)) will be used.

The FTP protocol is used by several partners in the LEAF consortium to provide their data to the central LEAF system. The means for the data transfer is offered either from a local FTP-Server at the provider site (the central system pulls the data from there) or the FTP server at the central server (thus the provider pushing his data for further processing).

The OAI protocol for metadata harvesting allows data providers to offer their data via an HTTP interface. The central LEAF system queries an OAI server at the provider's site to request bulk transfer of data to the central server. This mechanism further allows to receive only those records which are newly available or updated at the local sites.

The Z39.50 protocol for search and retrieval is used within the users' search for records. Results from such searches are added to the overall result from the central system and visited data are stored in the central database thus allowing for later updates to keep the central repository up-to-date.

An XML interface grants external access to other systems which want to query LEAF for person descriptions and their details.

The software layers and components used in the central LEAF system are illustrated in the following figure 2.

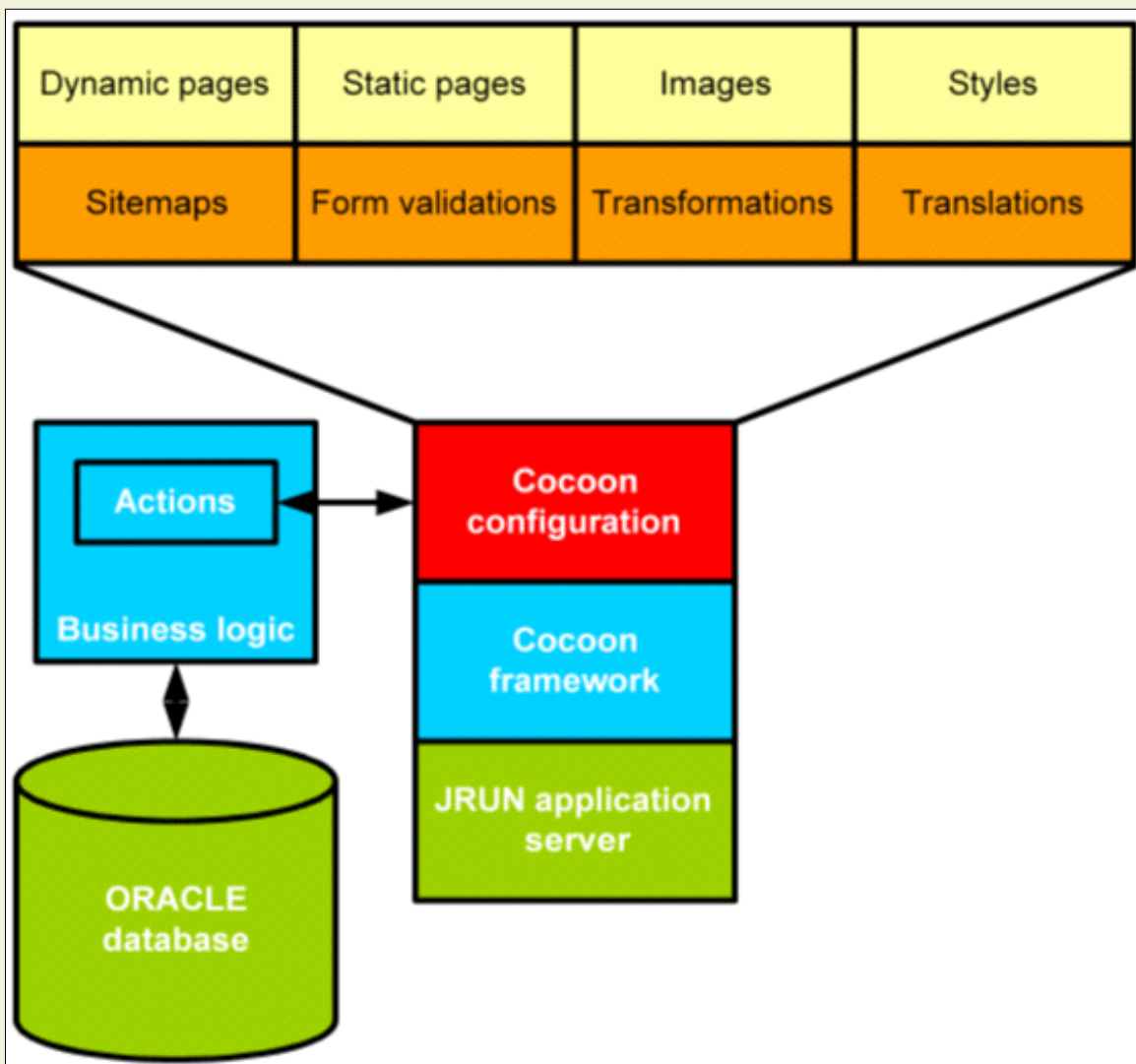


Figure 2: Software layers & components of the central LEAF system

- The [JRun](#) application server accepts the incoming HTTP requests.
- The [Cocoon](#) XML publishing framework controls the flow between the various requested URIs and also provides conversions via style sheets for output of the web pages.
- Interactions with the LEAF database and external sites are done via the business logic including actions (Java classes) which are used by the Cocoon configuration.
- The configuration utilises a variety of different file types as listed in the following table.

Configuration elements	File types
Actions	Java classes
Business logic	Java classes
Dynamic pages	XSP
Form validations	XML
Images	GIF, JPG
Sitemaps	XML
Static pages	XHTML
Styles	CSS
Transformations	XSL
Translations	XML

Table 1: File types used in the Cocoon configuration

The Cocoon framework, an MVC implementation, provides several basic mechanisms for processing web requests and XML

formatted inputs (shown in figure 3, taken from <http://xml.apache.org/cocoon/userdocs/concepts/index.html>):

- Dispatching LEAF requests based on Matchers
- Generation of XML documents (from content, logic, relational DB, objects or any combination) through Generators
- Transformation of XML documents into other XML documents through Transformers
- Aggregation of XML documents through Aggregators
- Rendering XML through Serializers

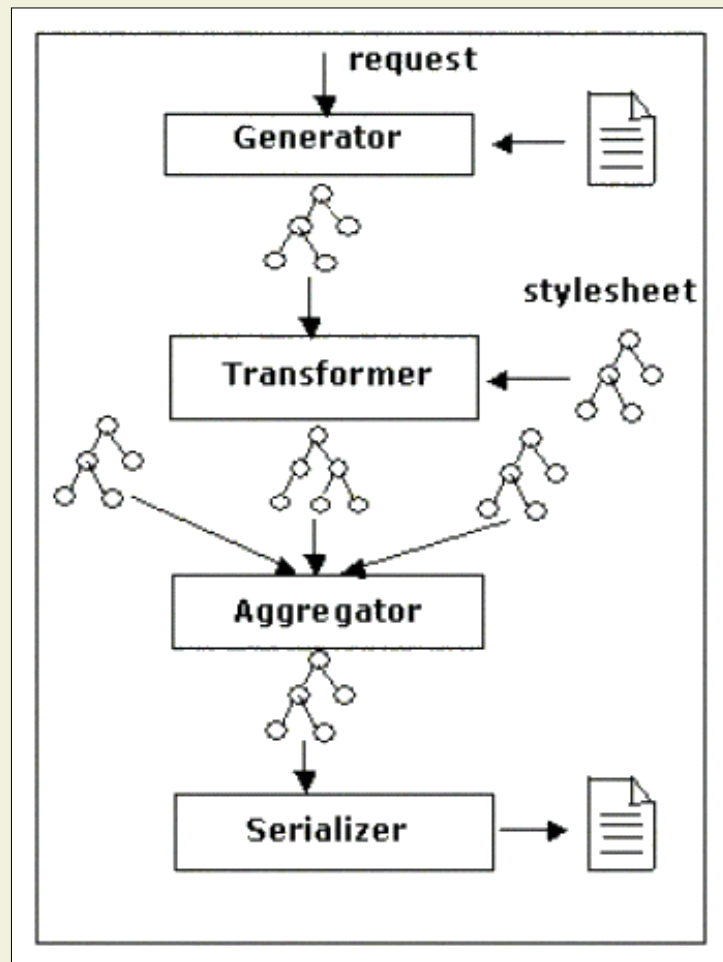


Figure 3: Major components of the Cocoon pipeline

Figure 4 (taken from <http://xml.apache.org/cocoon/userdocs/concepts/index.html>) shows the interaction between the client's web browser and Cocoon.

- The web browser sends an HTTP request regarding LEAF to the Cocoon servlet
- The servlet forwards the request to the sitemap
- The sitemap selects a pipeline to be used and initiates the execution of that pipeline
- The pipeline is executed as defined to produce the HTTP response
- The HTTP response from LEAF is returned to the client's web browser

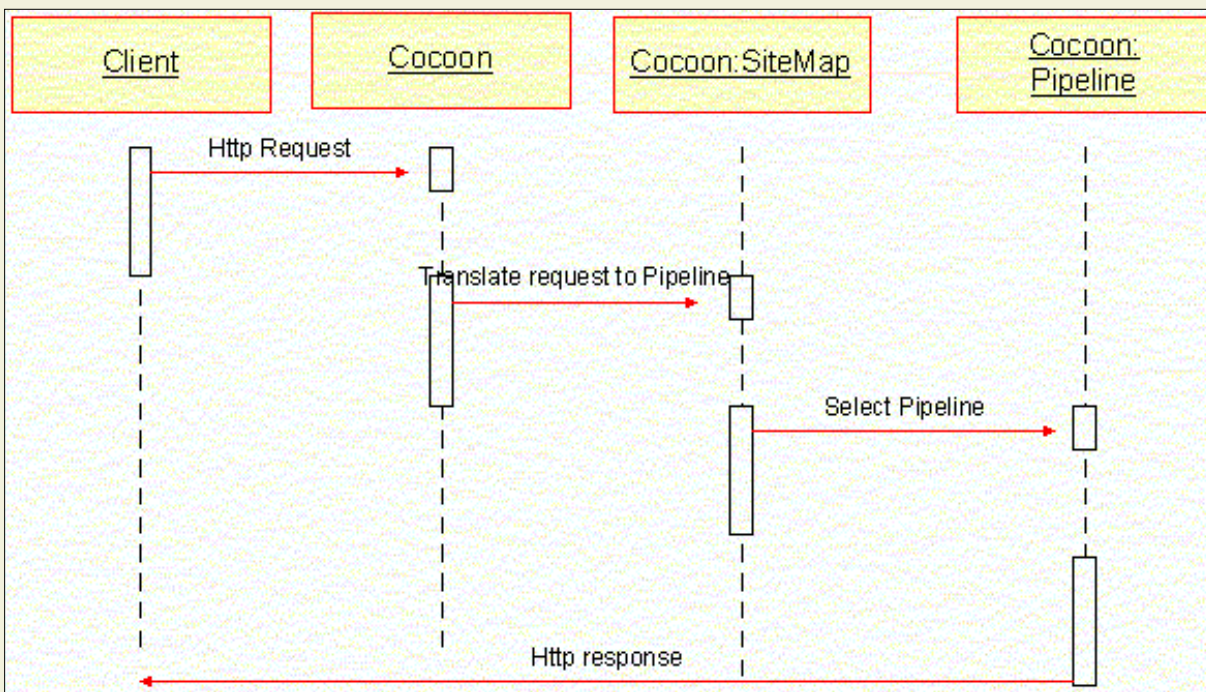


Figure 4: Interaction with a Cocoon based site

The general structure of a Cocoon pipeline can be seen in figure 5 (also taken from <http://xml.apache.org/cocoon/userdocs/concepts/index.html>):

- The request is handled by a Generator which produces SAX events for an [XML](#) document (e.g. the simplest generator – the default generator – reads an XML file from the file system, but a large range of other generators exist which can be also enhanced if necessary)
- The SAX events from the Generator are processed by a sequence of zero or more transformers which provide transformations into other nearly arbitrary XML structures (as well a lot of transformers exist like XSLT using XSL style sheets, i18n for internationalisation or filter for limiting the number of results returned)
- A Serializer concludes the pipeline producing binary or character streams from SAX events for final client consumption (e.g. the simplest serializer – the XML – provides an XML document from the SAX events and the default serializer produces HTML for direct consumption in the client browser)

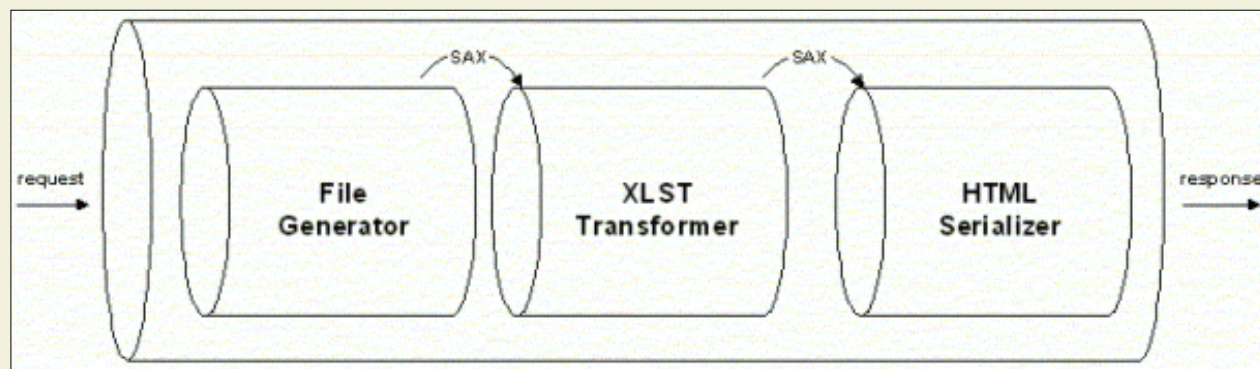


Figure 5: Pipeline principles

The LEAF Maintenance Suite

Rob Bull, CNS

The LEAF System administration will be complimented by a web based maintenance suite.

The objectives of this are to compliment the technical and administrative management of the LEAF distributed system technology. It is often the case when using distributed technology systems that the technical administrative aspects become difficult to maintain and manage due to the numbers of players and stakeholders involved. In addition, there is often an amount of technology learning to address by the data stakeholders involved.

With the LEAF Maintenance Suite, data providers will be able to register their administrative details and the technical details of their data systems to the Maintenance Suite registry.

The Maintenance Suite will notify the LEAF System when any technical administrative factors are required or changed. The user interface of the Maintenance Suite will be multilingual and will contain examples of how the data provider can register their system complimented by on-screen help.

The software will also contain verification utilities that will check the connectivity to the data providers' systems and will automatically issue notifications to administrators if connectivity fails.

Encoded Archival Context (EAC) – The Story Continues

Max Kaiser (Austrian National Library) & Hans-Jörg Lieder (Berlin State Library)

Readers of the first LEAF Newsletter have been aware of the EAC format for a while (cf. Per-Gunnar Ottosson: Encoded Archival Context (EAC) - Recent Developments. In: [LEAF Newsletter, Issue No. 1, March 2002](#)). The Alpha EAC-DTD was developed to answer the widely recognized need for a standardized way of encoding archival context information. The ICA Committee on Descriptive Standards (ICA/CDS, <http://www.hmc.gov.uk/icacds/eng/home.htm>), LEAF and other projects have commented on it and suggested a number of changes.

In September this year a core group of the EAC developers, consisting of Daniel Pitti (University of Virginia, USA), Dick Sargent (National Register of Archives, National Archives of the UK) and LEAF project member Per-Gunnar Ottosson (National Archives of Sweden), met in Stockholm to review the suggested changes.

This resulted in a pre-release of the EAC Beta version in early October this year (see: <http://jefferson.village.virginia.edu/eac/>). This pre-release was made available to a limited number of specialists who were encouraged to test and experiment with it and report possible errors.

Currently the EAC Beta Tag Library is being completed: a few elements and attributes remain to be defined, the most important missing piece in the Tag Library being comprehensive examples. This work is well underway, drawing on ISAAR(CPF) examples as well as other examples submitted by LEAF and others. Additionally a draft introductory overview is now available, taken for the most part from a paper Daniel Pitti delivered at an International Authority Control Conference in Florence in February of this year (see: <http://www.unifi.it/universita/biblioteche/ac/en/home.htm>). The full paper is available both in English and Italian at: <http://eprints.rclis.org/archive/00000316/>.

EAC in LEAF

The local authority data that is uploaded to the central LEAF system is originally encoded in different formats. In order to be able to compare individual records and thus make them available for further operations one common exchange format needed to be identified into which all records, independently of their native format, can be converted. LEAF has adapted EAC for this purpose. The conversion module of the central LEAF system consists of data conversion routines for each local data structure which convert the uploaded or harvested local records into EAC XML and the different character sets into Unicode (UTF-8). The converted data are then further processed in the LEAF system. In addition to the converted form records are saved in their local formats as provided by the LEAF Data Providers.

For presentation of the records by the LEAF user interfaces XSLT scripts are employed which transform the EAC XML data into HTML representations according to specific requests of the users. Besides a view in the specific "LEAF Presentation Format" users can view the records in "MARC21-like", "UNIMARC-like" and "MAB2-like" formats. These presentations are accomplished by XSLT transformations of the EAC representations of single records (these transformations can, of course, not be done for aggregated, i.e. linked records). The results of the transformations are not, however, "true" MARC (MACHINE-READABLE CATALOGUING, <http://www.loc.gov/marc/>), UNIMARC (UNIVERSAL MARC, <http://www.ifla.org/VI/3/ubcim.htm>) or MAB (MASCHINELLES AUSTAUSCHFORMAT FÜR BIBLIOTHEKEN – AUTOMATED LIBRARY EXCHANGE FORMAT, http://www.ddb.de/professionell/mab_e.htm). records: creating "true" MARC or MAB records would be highly complicated not only due to incompatibilities between cataloguing rules and data formats but also because of idiosyncratic character sets. The results may nevertheless be useful for further (manual) processing within the users' local systems.

Registered LEAF users can import selected records into their individual online LEAF user workspace from where they can be manually edited, stored and downloaded. Users will also have the possibility to download records in their original local formats or in the EAC XML format and then further process them locally.

Newsletter created by Crossnet Systems Ltd (member of the DS Group).
Copyright ©2003, All Rights Reserved, LEAF Consortium.
More Information at www.leaf-eu.org.

If you wish to unsubscribe, please send a message with the text "unsubscribe" in the body of the message to: co-ordinator@sbb.spk-berlin.de.