Project no. 034595

SELF
Science, Education and Learning in Freedom

Instrument: SSA - Specific Support Action
Thematic Priority: IST-2005-2.5.5 – Software and Services

Deliverable D4
SELF Platform definition:
Navigational Structure and functional requirements

Period covered:
from July 1st, 2006 to 30th December 2006

Date of preparation:
13th February 2007

Start date of project:
July 1st, 2006

Duration:
2 years

Organisation name of lead contractor for this IR
Internet Society Nederland
Wouter Tebbens

Revision v0.6
Revision date: 31 July 2008

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	PU
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Index

Acknowledgements

This document has been collaboratively written by the SELF Platform Definition team. The Platform Definition team has held many discussions on various alternatives of shaping the Platform. In this team take place: Federico Heinz, Meena Kharatmal, Dragoslava Pefeva, David Jacovkis and Nagarjuna G. and Wouter Tebbens. All participants are gratefully thanked for their valuable inputs. And we should not forget to thank the development team for their inputs and plans to realise the plan that you are reading at this moment. And thanks to so many existing Free Software projects that will be reused and adapted where we see fit and what makes the realisation of this plan a challenging but manageable objective!

Document History

This document has been presented to the European Commission on February 14th 2007 and has been reviewed for the annual review later that year. As the platform definitions in this document has been a starting point for other SELF teams. As the development of the platform has been progressing, discussions in the development team, quality assessment, learning materials and communications teams have several suggested several improvements to the initial plans. The main changes and improvements to the platform definition are represented in the updated version of this document.

Table of change history

Date	Version	Comments
14 February 2007	0.5	First version sent to the EC for review
31July 2008	0.6	Updated version

Introduction

This document describes the navigational structure and high level specification of the SELF Platform to provide the design and technical development teams with the requirements (structure and functions) of the Platform.

With this aim we will first introduce the SELF Platform by explaining its objectives, main functioning and target audience. Secondly, we will define the main terms used in this document. Thirdly, we will address the main navigation structure and specific requirements for the user experience. Fourthly, we will describe the main functional requirements for the SELF Platform to be built.

In order to accommodate for a user-driven development cycle, we have decided to keep this document relatively brief to avoid fully detailed specifications. This approach has a twofold objective: to establish a first set of definitions and functions to be commented by the community of interested users, and to start with the development of the Platform without defining many details. During the development phase, the definition and development team will keep close contact in order to decide on specific details and adapt specifications when appropriate. Such agile development method should help us to speed up the realisation of the Platform and avoid limiting ourselves with initial specifications.

A Roadmap for development is maintained as can be found on the SELF project website[1]. The roadmap is divided into three main phases. The first phase is from the beginning of the development SELF team (in WP6) until the delivery of Deliverable D6 (WP6) and represents the formal development phase of the EC-funded period. The second phase constitutes the additional development work until the end of the EC-funded period (until July 2008). The Third phase is post EC-funded period and will take place by a community of voluntarily cooperating people and parties interested in bringing the SELF platform further and doesn't have formal obligations to the European Commission. Some elements described in this document are planned to be developed in the third phase.

SELF Platform overview

The SELF Platform is a web-based platform where users can find educational materials on Free Software and Open Standards. At the same time it is a collaborative production environment for the creation of new materials and the improvement and translation of existing ones. In summary it serves three main functions:

1. a central point of access to free educational materials about Free Software and Open Standards, including both general concepts and concrete applications in various forms, formats and levels.
2. a collaborative production facility for the community of users to work with the learning materials in a multi-language environment, with multiple standard formats and assuring quality levels in innovative ways.
3. a community of interested individuals and organisations that communicate with each other through the platform.

These educational materials will be offered through a central point of access in the SELF Platform, where they will also be created, adapted and translated. Users of the Platform can harvest or collect

1 http://selfproject.eu/en/TEG

existing educational materials from external sources and link them into the SELF Platform. They can create new materials and combine them with existing ones, and they can translate materials. Several quality assessment mechanisms will provide the users with different quality indicators, such as the popularity of the material, review and quality feedback loops. The resulting materials can be accessed in different ways: directly through the web interface of the SELF Platform, through downloaded materials reproduced in Computer Assisted Instruction environments (CAI, VLE or LMS) from educational institutes, books printed by interested publishers, or on CD or DVD as stand-alone distribution.

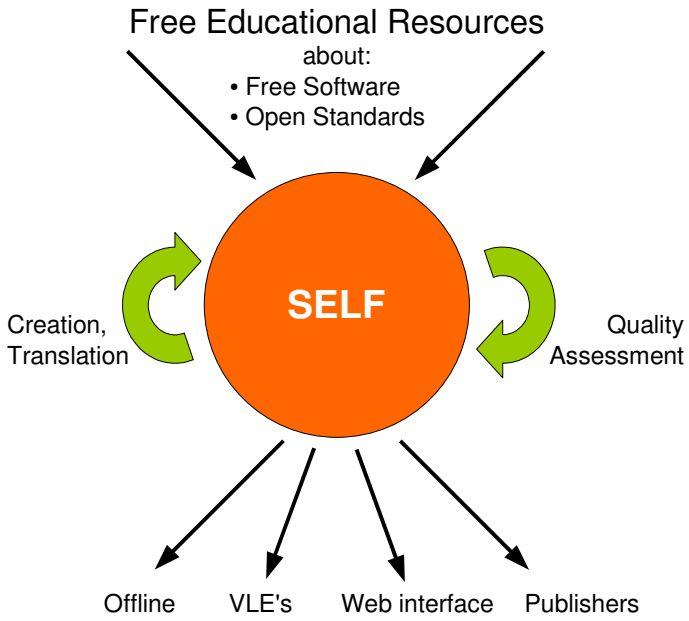

Illustration 1: SELF Platform - general working

The Platform serves the interested users and shall facilitate the establishment of a community around SELF. The users that make up the SELF Community can be found in several target audiences, such as:

- **Universities and schools** – teachers, coordinators and administrators can teach students and train internal staff, and can publish their internally developed material on SELF
- **Training organisations –** trainers can train client organisations, and can publish their internally developed material on SELF
- **Publishers –** can publish SELF educational and training materials, and publish their free educational materials on SELF
- **Industrial manufacturers** – can provide their own training materials as free educational materials
- **Free Software communities –** can promote their project and launch training materials about their software on SELF
- **Government bodies** – can have a general, more strategical interest in promoting and/or supporting a free information society infrastructure
- **Public and Non-profit organisations**: can facilitate their users with free materials on free information society tools, organise courses and workshops based on these materials and collaboratively develop new materials through SELF. These organisations include libraries, ethnic minority and cultural centres, institutions that work with marginalised groups, etc.
- **Private organisations** – can benefit directly from the available training materials on the

desired free software.

In fact, we can distinguish three different roles within these groups. First of all, *authors* that collaborate in the production of learning materials. Secondly, *instructors* that combine and adapt learning materials into new courses. Thirdly, *students* that read and discuss the materials.
In this case, the 'students' are the trainers or the educators, since SELF applies a train-the-trainer or educate-the-educator approach. Through the trainers and educators, the knowledge can be passed on to their students and trainees and a multiplier effect can be achieved. Although students and trainees are not a direct target audience, they are expected and welcome to participate in the SELF Platform as well, as they can provide relevant feedback, further development, modification according to cultural differences (i.e. localization, translation), peer based support, etc.

Definition of Terms

In this section we define the main terminology used in this document.

Learning Objects
The term Learning Object (LO) will be used to identify any single piece of instructional material stored in the SELF Platform. LO's are the basic building blocks of content in the SELF system, and are combined into courses or Learning Trajectories. An LO can be a single page, or a chapter of a course. Each LO maps to a single web page, but may also be viewed in other formats such as PDF and ODF. LO's have metadata like keywords and an abstract, may have multiple authors and a set of supplementary links. LO's can include content from a variety of sources in various formats, such as images, audio and video objects.

Learning Trajectories / Courses
The term Learning Trajectories, sometimes also referred to as **Courses**, is used to identify complete courses or sequences of Learning Objects. Learning Trajectories (LT) allow instructors to combine LO's in a specific order and provide custom titles, annotations and supplementary links. The same LO may be customized in completely different ways to produce different courses.

Repository
A Repository is a central place where data and files are stored; in the case of SELF, it refers to the collection of Learning Objects, courses and other data that is accessible through the Platform. SELF uses a distributed repository through which access to all materials is centralized, but the actual files and data can reside in many different locations. All materials are stored under version control in the SELF repository. This allows earlier versions to be viewed and compared at any time. The repository can be browsed through the web by title, author, keyword, or popularity.

Learning Materials

Learning materials, also referred to as **educational materials**, refer to all Learning Objects and courses on the SELF Platform. Sometimes we also call them **Free Educational Resources**.

IMS
IMS Global Learning Consortium, Inc. (IMS) is developing and promoting openspecifications to facilitate online distributed learning activities, such as locating and using educational content, tracking student progress, reporting student performance, and exchanging student records between administrative systems.

IR
Interim Report, refers to intermediate reports that are in a later stage published as Deliverables. This terminology is used in the context of the EC's Framework Programme.

Deliverable
A Deliverable is a report that covers a concrete result of the project in the context of the EC's Framework Programme. All public SELF reports are or will be published on http://selfproject.eu/nl/project/results

WP / Work Package
A Work Package (WP) is a fundamental part of the work in the context of the EC's Framework Programme. In the SELF Project, all Work Packages are organised around teams.

SCORM
The Shareable Content Object Reference Model (SCORM) defines a Web-based "Content Aggregation Model" and "Run-Time Environment" for learning objects. SCORM is a collection of specifications, adapted from multiple sources to provide a comprehensive suite of e-learning capabilities that enable interoperability, accessibility and reusability of Web-based learning content.

Accessibility
Accessibility is a general term used to describe the degree to which a system is usable by as many people as possible. In other words, it is the degree of ease with which it is possible to reach a certain location from other locations. It is not to be confused with **usability**, which is used to describe how easily an entity (e.g., device, service, environment) can be used by any type of user. In the context of the SELF Platform, accessibility refers to web accessibility. **Web accessibility** refers to the practice of making Web pages accessible to people using a wide range of user agent software and devices, not just standard Web browsers. This is especially important for people with disabilities such as visual impairment. In order to access the Web, some users require special software or devices in addition to a standard web browser, or specially designed web browsers. Design for accessibility is a sub-category of good design for usability.
The needs that Web accessibility aims to address include:

- Visual: Visual impairments including blindness, various common types of low vision and poor eyesight, various types of colour blindness;
- Motor/Mobility: e.g. difficulty or inability to use the hands, including tremors, muscle slowness, loss of fine muscle control, etc., due to conditions such as Parkinson's Disease, muscular dystrophy, cerebral palsy, stroke;
- Cognitive/Intellectual: Developmental disabilities, learning disabilities (dyslexia, dyscalculia, etc.), and cognitive disabilities of various origins, affecting memory, attention, developmental "maturity," problem-solving and logic skills, etc.;
- Auditory: Deafness or hearing impairments, including individuals who are hard of hearing;
- Seizures: photoepileptic seizures caused by visual strobe or flashing effects.

Usability
Usability is a term used to denote the ease with which people can employ a particular tool or other human-made object in order to achieve a particular goal. Usability can also refer to the methods of measuring usability and the study of the principles behind an object's perceived efficiency or elegance.

In human-computer interaction and computer science, usability usually refers to the elegance and clarity with which the interaction with a computer program or a web site is designed. The term is also used often in the context of products like consumer electronics, or in the areas of communication, and knowledge transfer objects (such as a cookbook or a help document). It can also refer to the efficient design of mechanical objects such as a door handle or a hammer.
See for more information: http://en.wikipedia.org/wiki/Usability.

Navigation structure

The SELF Platform will be mainly accessed by its web interface, of which we can define the main objectives as follows:

- to give access to the SELF platform and all of its resources
- to enable the usage of all functions of the SELF Platform
- to communicate the platform and the available learning objects, resources and services
- to involve interested users and organisations in the project and platform
- to provide a home for the SELF community

The platform consists of three different levels, being 1) a general entrance with general information on the concepts of Free Software and Open Standards as well as its practical applications and concrete standards; 2) a platform for the collaborative production of educational materials on these general concepts and applications; and 3) a repository with the end results, or released versions of the produced educational materials under point 2. All of these three levels shall be accessible in an easy way through the main domain name, http://selfplatform.eu while one or more other domain names could be used as alias or redirect, such as http://selfplatform.org. Access and navigation should be done according to accessibility guidelines as proposed by W3C and enacted by the EU.

In order to achieve a successful and positive user experience, the navigational structure should be presented in a simple way to the users. In this section, we define the navigational structure, which is subdivided in these four elements: a main navigational menu, which presents the menu sections that are always accessible and can have several submenu items; a set of direct access items that provide direct access to useful functions; the Learning Objects and Courses which present the main focus of the Platform; and several contextual menus that present a block with additional information and links related to the position and mode the user finds her/himself in. The next illustration presents the main navigation structure.

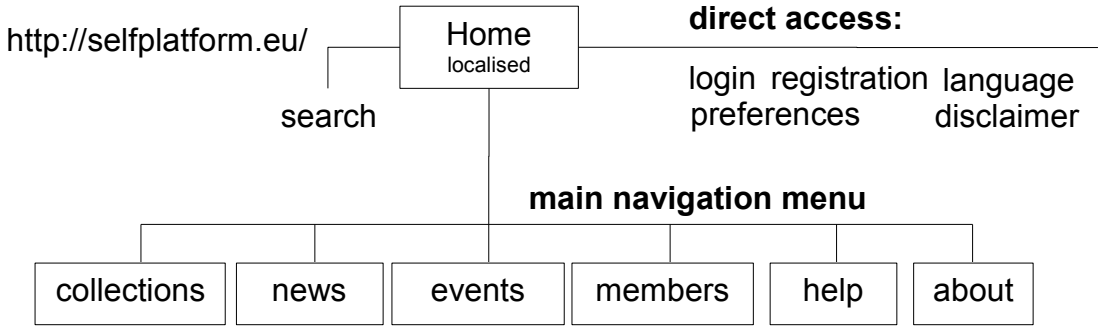

Illustration 2: Main Navigation Structure

First we discuss the main menu sections, then the direct access options, the Learning Objects and courses and at last the module overview.

Main Navigation Menu

Home

Apart from the navigation menu and functional direct access options, the home will present a series of front-page items, like featured courses, most active courses, most active users, ToDo items, calls for voluntary support, recent news and events, etc.

About

Information about the platform, its organisation, the SELF project, partners, associates, etc.

News

Press releases and news items related with platform-hosted projects will be presented here. RSS feeds will facilitate distribution.

Events

The events calendar presents a means to announce upcoming events and archive them automatically once they are passed.

Collections

Collections present a direct access to the catalog of categories, or collections, which can be browsed to find the desired materials. In each entry a description of the collection and a list of existing courses can be browsed.

Help

The Help section is aimed at quick-starting the users on how to use the Platform, how to search for Courses, start new ones, etc. It contains a glossary of terminology, FAQ's, SELF Policies, Guidelines etc. It will also contain links to courses about the use of SELF.

Contributions

This option gives access to the list of user contributions. The default listing presents the user with highest user rating first, descending. The list of users can be ordered and filtered in different ways. For example for the highest contributor in a certain field (by using tags or inference from the semantic database), recent active users, per country or language etc. By clicking on the user-id you will be presented the user's personal page with statistics on activity, produced materials and personal data (in line with the users privacy settings: only those users that have opted for public presence in their privacy preferences).

Support

In this section we will present the SELF Support Programme, that explains how individuals and organizations can contribute financially or otherwise to the SELF Platform. It will show as well the contributions from the most recent contributors divided in several categories. (note: that's all in “about”)

Direct Access options

Language Selection

The language selection option in the "direct access" options is only available for anonymous users. For logged in users the system presents the user interface in the language specified as the preferred language in the user preferences. For anonymous users, the browser language (from the HTTP-Header) is used as preferred language. In case an anonymous user wants to switch to another language, she can do so by clicking on the language access button and selecting from the available languages.

An ordered list of preferred languages can be configured in the preferred language settings. This will enable the system to present links to alternative resources in the other preferred languages. There will be a manual for translation and contact in case someone wants to create a new translation or update the current translation of the Platform's interface.

Preferences

Preferences may include:

- language1 (used in the user interface), language2, language3, language4
- privacy preferences
- preferences about notifications
- specific accessibility settings
- personal page or blog at SELF

The access "Preferences" is only visible when the user is logged in.

Registration

The registration process to become a new SELFer and join the SELF community should be simple, presenting a low threshold to the user. By providing the minimum details, a user can obtain a user-id. Required personal details are a username, password and an email address. More personal details can be added to it, during registration or later. Personal data:

- User name*
- Password*
- First name, Last name
- Attribution: options to choose the attribution of the user's authorship: a) hidden, b) User name or c) First name + Last name
- Address details (Street, Postcode, Country)
- Contact details: email address*, Jabber ID (or other IM ID), SIP account
- Avatar image upload
- Signature that is added underneath posts
- Telephone / Mobile / Fax numbers
- Birth date (dd/mm/yyyy)
- Links to website or blog: two fields
- Organisation, Function, Activity
- Preferred accessibility options

* Required fields for registration

Privacy settings allow the user to specify which personal data that will be visible in certain cases.

The options include: a) completely hidden, b) only visible to logged in users or c) always visible. Options b) and c) can be applied to all personal data fields individually.

The registration process includes an automated Turing test, using distorted letters the user must identify, to keep automatic spamming systems from creating users.

The registration access automatically disappears when a user is logged in. Instead an option to logout appears.

Login

A username and password box is provided for login. When the login is unsuccessful, a resend password option is offered. After logging in, the Preferences access and a Logout link appear.

Search

The Search engine is a crucial feature, as it provides easy and direct access to the resources on the platform. As the number of resources grows the search function is expected to become more and more important. There will be two main search functions: a basic one and an advanced one. The basic search function queries the title and content of the Courses and Learning Objects.

The advanced search engine uses ontology-based searching, in other words, it uses the ontological concepts and semantic relations that are maintained by the system to present a list of search results that are accurate and make sense to the user. Contextualized search would be possible as well, which takes into account the actual user profile of the user to present to her or him the related resources.

Search terms are matched against:

- object id
- title
- keywords
- author names
- full body text

The user can choose if she wants to search the LO's, courses or both, choose the languages to search in, give a date range between which the objects should have been published. The user can specify the range of the search: the system can search within the current search results (thus refining a search), within the user's Bookshelf, within the semantic vicinity of an LO, a topic, an author, etc. Search should be done in a way that meets accessibility requirements.

The search results will be presented in a list with the following details:

- title
- subjects/keywords
- author names
- language
- quality indicators

Learning Objects and courses

Learning Objects and courses can be accessed in various ways, for example by browsing the topics and related courses, or by using the SELF search engine to get directly to the course one is looking

for. The structure and functionality of the Learning Objects and Courses is described in the Functional Requirements section. In summary, the user interface should present the Course title and an index of the course contents. See for example: http://cnx.org/content/col10151/latest/. When the user wants to browse through the course material, the interface changes, see for example: http://cnx.org/content/m10884/latest/

Main modules

The platform has different modules that can be used through the web interface; they will be introduced below.

Collection Manager

The collection manager allows users to view the various collections and subcollections of learning materials on the platform. There is a description of each collection. The initial set of collections comes from the categories as defined by the Learning Materials team. SELF users can add new collections (only when they are logged in) and users with a manager role can edit, reorder and remove collections. All logged in users can add also new (empty) courses within any collection.

Course Manager

The Course Manager is the main authoring environment for courses and learning objects. First of all, it allows users to browse existing courses and their LO's, and once they are logged in, users can use it to add, edit and remove objects. Each edit saved to the system is considered as a new version. The removal of an object also constitutes a new version. In the Course Manager, users can add lessons, FAQ's, tests, links and glossary terms. The organizer function allows users to reorder or reorganize the structure of the lessons within the course.

Shelf Manager

The Shelf, which is short for Bookshelf, is a user's place for storing favourite courses and objects. Objects can be added, removed and upgraded to newer versions, if these come available.

File Manager

The File Manager allows users to upload files to the SELF Platform which then can be used by all users. Only files in open standard formats shall be allowed, as defined by the Learning Standards Expert Group. Special attention shall be given to the import of existing courses: such courses can be uploaded to the FileManager and afterwards they can be imported as course object. The system shall make sure no double imports will take place. The FileManager also allows for searching all files available at the platform's file system(s), within the files of the user, or within all files. Other file types:

- Image files: a preview option shall be made available for the web image formats
- Audio files: reproduction handles shall be made available for the audio files (with start, stop, pause, forward and rewind buttons)

Version and change manager

From the course manager, users can access the version history and change manager. The version

manager presents a graphical interface to see the version tree of the course as it has been developed by the contributing users. Users can select two different versions and compare the changes, and if desired merge one version into the other.

Functional Requirements

This section will describe some of the main concepts that characterize the SELF Platform: first the authoring model, which defines the way authors can collaboratively create educational materials, secondly the main life cycle of these materials, thirdly the user reputation system which recognises the contributions of the users and forms a crucial part in the way the Platform works. After describing these main concepts we will go into detail with the functional requirements of the SELF Platform. These requirements relate to the Use Case Scenarios that we have started to describe and should be extended over time. We conclude the section by defining the main architectural concepts behind the Platform.

Authoring Model

Some main requirements for authoring and contributing in the SELF platform:

- low-effort for contributors: we should have the minimum requirements for registration. Once the registration data are submitted, volunteers should be able to contribute, no need to wait for confirmation E-mail, moderators, etc.
- keep track of up-to-date data on the the document, activity status, user and material rating (see WP8: Quality Assessment mechanisms)
- documents can be commented, and any logged in user can edit courses and LO's. Malevolent users that slip in commercial stuff, unskilled users that can easily add errors are the downside of direct editing. We expect the quality assessment mechanisms built into the platform to eventually deal with malicious and bad quality contributions by itself.
- Around any LO a small team of mentors, specialists, illustrators, etc., who are subject matter experts, might evolve. The system should allow for that and stimulate such self-organising teams. They can periodically implement relevant comments and clear the comment stack (ToDo list).

The way users can create new learning materials is one of the fundamental aspects of the SELF Platform. Various models are possible, and they determine the way people can participate in the Platform. Conventional scientific magazines have authors submit their work, after which a board of editors selects which works they want to be published. Another model, as used by Connexions, lets authors create new materials, on their own or in work groups. It is only after publishing that other users can access the materials. Adaptation of the material is only allowed by the authoring group, and users can make suggestions to them but not edit directly. In Wikipedia, this is opened up so much that any user, registered or anonymous, can edit directly.

For the SELF Platform we view attribution as an essential feature, which leads to recognition of the contributors and helps build the user's reputation (more on this in the User Reputation System section). We view the open and direct editing model as the way to go, as it permits maximum collaboration and access to available learning materials. Users can thus participate in several ways in the authoring of Learning Objects. They can add to existing LO's and get recognized as authors of that LO. They can also review existing LO's and thus provide feedback to both the authors and the students who want to consider the use of the LO. The users that are viewing a Learning Object

and detect minor errors (like typos), can correct these on the fly; in case they detect more serious failures they can add an item to the discuss space of the LO. Authors that have subscribed to the LO get notified of the observation and can take part in the discussion and/or take appropriate action.

Direct editing

All logged in users can directly engage in the editing process. As each edit action saved to the system constitutes a new version, a version tree will be automatically built during the editing. Note that in most wikis, the change history is linear: contributions will always be built upon the last edits. SELF however proposes a free versioning system, as reflected with the tree metaphor. If a user doesn't like the contribution of the last editor, she can go back to an earlier version and continue from there. A new branch is then created. This versioning mechanism aims to take away the dissipated energy that can be observed during edit wars in systems like Wikipedia. Especially in a platform for the collaborative creation of educational materials, it is seen as crucial to allow for different pedagogies and different ways of explaining a similar concept, which can be reflected in different parallel branches.

Attribution: the 10% Contribution rule

When authoring is attributed in a manual way, human beings can decide on which contributors have made serious enough contributions to be attributed as author and which ones not. In the SELF Platform however, attribution is automatically generated by the system. While many users might intervene in the editing process, especially as this is made very low threshold, not any user should be attributed as an author. For example someone who has corrected a typo should not be recognised as “author” of the LO, otherwise abuse would be stimulated. In this process of authoring, the users that seriously contribute to the LO will be attributed as such. We propose to let the system compute the contributions made by all users to a certain LO and define automatically which contributors will be added to the authors list of the LO. Now, only those users that have contributed at least 10% of the highest contributor of this LO are attributed as such. In the case of creating a new LO, the first author contributes his or her initial part, which is at that moment 100% and he or she becomes attributed as an author. Subsequent users that add more than 10% of the first user's contribution (or of any subsequent user that becomes the main contributor to the LO) will be recognised as authors in the authors list. In this way the number of authors will be limited but can expand over time, as more and more authors add to it and get attributed as authors. The specific license under which the LO is published defines how such material should be treated, for example when deriving a new object based on an existing LO. The order of presentation of the authors (and possibly the size of their names) reflects the importance of the author's contributions: the highest contributors should be placed first (in descending order).

The copyright of the LO's will be with the authors, which according the SELF Legal Policy and Terms of use allow SELF to distribute them under one of the accepted licenses, as chosen by the author that has initiated the LO.

Licencing & Fiduciary Licencing Agreement (FLA) Information

Each LO should have associated with it a list of authors that have contributed copyrighted material and its legal status. Each list entry should allow to include the name, email address, real address, a list of changes with date, the licences this person agreed to licence the material under and a field giving information whether the person has signed a Fiduciary Licence Agreement (FLA) with the

FSFE. This field should allow for a status of “requested”, “confirmed” and “denied” (with explanation).

Workflow

The workflow for the licensing information would be as follows: the person submitting material would select a license from a list of acceptable licences compatible with the existing licences on the LO, potentially finding all licences used in the LO to date preselected.

The workflow for the FLA information would be the following: the person submitting their contribution can select an optional box to request an FLA transfer to FSFE, which would be recommended for LO's where other authors already signed such agreements. The FLA field would then be set to “requested” and the user would be presented with a PDF to print out and sign. Upon receiving the FLA signed, the Freedom Task Force (FTF) of the FSFE will set this FLA field to “confirmed”, or, in case of problems, “denied” with a short reasoning and possibly a link.

The platform should allow for easy association between the specific contribution and the FLA sent in to the FTF, which would imply some kind of unique ID and adequately protected means of updating the FLA status by the FTF. The FLA itself can currently be auto-generated/customised .

Material Life Cycle

The process of selecting materials, creating, improving, releasing new versions, communicating them to target groups etc. can be seen as a cyclic activity. Figure 2 to illustrates the basic material life cycle.

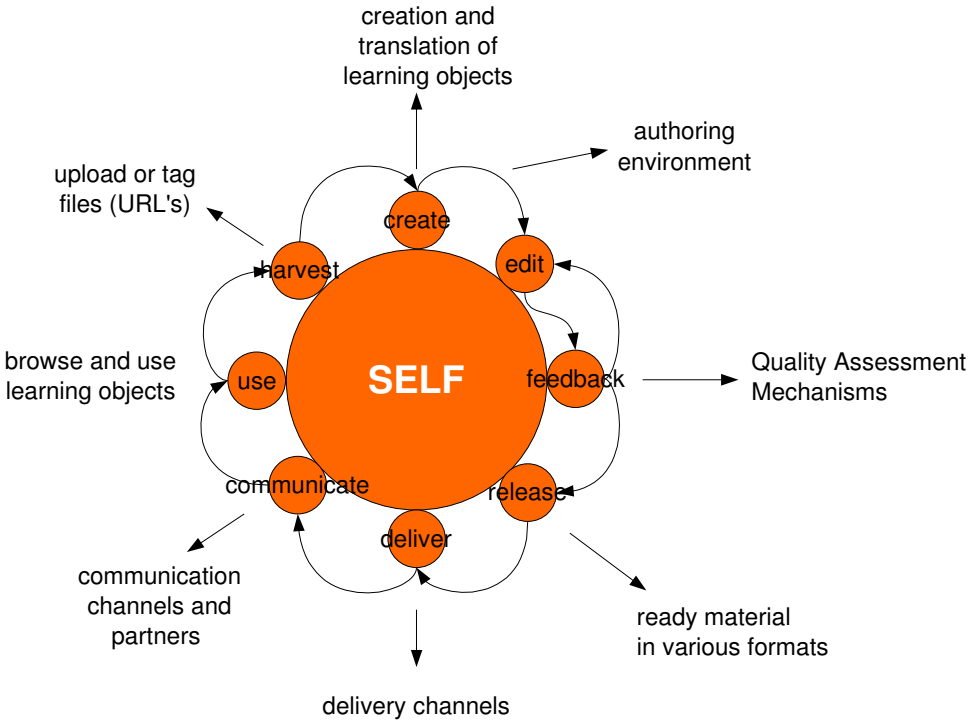

Illustration 2: Material Life Cycle

Brief description of the material life cycle:

1. harvesting or tagging of external materials that later can be reused in one or more LO's

2. creation of new LO's based on existing materials (harvested or derived from existing materials on the Platform) or from scratch
3. after initial creation of a new LO, a process of editing and feedback loops can start to improve the material
4. A specific version of a Course can pass the popularity threshold and qualify for being "released" with a editable URL. Such URL makes the course better accessible and can help boost its popularity.
5. all materials are accessible through the web platform and through other delivery channels: all releases can be exported in the SELF accepted formats to be used in other platforms, such as Computer Assisted Instruction systems, etc.; the courses can be presented with an standalone player on CD and DVD; publishers can print it in books or they can simply be downloaded in HTML format.
6. the newly created materials can be communicated to interested users in the community (or outside) to draw attention for them and get more users
7. the more users review and use a certain material, the higher the material will rise in the popularity ranking, the more people become aware of it. Besides, a high number of users is expected to generate more quality feedback and stimulates the authors to improve the material
8. the popularity of materials will recognise the contributions made by its authors, encouraging them to take part in the creation process as well as drawing in more users to create new materials.

User Reputation System

SELF aims at being a self-sustainable, community-driven platform for the production and distribution of educational materials. Being self-sustainable and community-driven go hand-in-hand, and are mutual enablers: a vibrant community is the best way of ensuring sustainability, as its members will have a vested interest in keeping it alive, and a sustainable platform makes participation more enticing, because it makes it less likely that it will go away.

Getting this positive feedback loop going will be a crucial part, and one of the largest challenges of establishing the SELF platform as a reference for the Free Software and Open Standards world. It is unlikely that SELF will start off as a large community, and while EU financing helps ensure initial sustainability, it is long-term sustainability that matters for users, so we will have to devise and implement mechanisms to provide potential users with incentives to become members and contribute to the platform.

What we can learn from others

Fortunately, the Internet has many prominent examples of successful communities from which we can draw wisdom on what works and what doesn't. Looking at community-driven projects ranging from Slashdot to Wikipedia and the Debian project, we can identify a minimal set of common features that enable them to engage members in the kind long-term, significant commitment these project needs.

When participating in a community, users have a number of expectations. Most critically for the community-building effort, users need to be able to feel a sense of belonging to the project, a personal bond not only to the project's goals, but also to the rest of the members; they need to know that there are ways in which they can leave their personal marks on the project. This need can be addressed by providing four basic features:

1. The ability to acquire an identity within the system. It is through this identity that members interact both with the system and with each other, it is the technical manifestation of the personal link to the project.
2. The ability to do real, useful work for the community, and the expectation that members will do it and take responsibility for it.
3. The ability to be recognized by their peers for their work, through acknowledgement of each user's contribution to the community and independently of any other criteria.
4. The ability to have real impact in decisions that affect the community. Members should not be just a resource, but active participants in shaping the community and its work.

These are features are matters of policy, not software, they are more the stuff of the design of the community itself than of the platform that hosts it.

Self-enforced policies

As a matter of fact, only the ability to reserve an identity within the community requires any specific support from the platform at all, and even it can be rather informal: mailing lists, IRC channels and Wikipedia are good examples of communities in which identity is very loosely coupled to the actual person: there is no need to actually identify oneself other than by a nickname, and it is rather common for people to have several identities, or for well-known members to participate anonymously.

Even when the platform does not enforce strict rules on identity, there are strong taboos related to them. Virtually no-one will object to people using an identity that suggests that they are of the opposite sex, or even lying about sex, age or other personal characteristics in the user's profiles, yet even on IRC channels, where it is rather easy to trick the system into thinking that you are someone else, impersonating other members of the community is strongly frowned upon. And since it is so easy to obtain an e-mail address for free and even forge e-mails, one would expect mailing lists to be rife with forgery, impersonation and anonymity, yet they are probably the electronic medium in which the identity is most strongly bound to the actual person.

So, communities don't actually need the platform to support any of the above policy features, but that doesn't mean that it can't be useful for them to do so. Particularly in the beginning of the community, when people still don't know each other, software mechanisms to enable people to gauge their and other's impact on the community can be useful. This usefulness remains later in the community's life-cycle, as a means for newcomers to orient themselves in the new environment. Interestingly, these mechanisms can be useful even if implemented in a tentative, imperfect way: perfecting them can be an item of discussion and constant review by the very community they attempt to reflect, thus contributing to the sense of belonging, and of being able to help shape the project.

Policy Support in the SELF Platform

The SELF platform will provide software support for the first three of the above policies, and may offer some support for the fourth, such as software to tally votes cast on communal decisions and certainly a record of the decisions made. Furthermore, as we will see later in this same document, the User Rating system will provide useful input for the material quality assessment mechanism.

Acquiring an identity

The platform will make it very easy to obtain a password-protected online identity: getting involved in SELF is a lot of work, and enough of a barrier to entry in itself. New members will just have to provide the desired nickname, password and an existing e-mail address, as well as pass an automated Turing test in which they are asked to recognize distorted letters in an effort to avoid mass registration by bots (CAPTCHA). Members can volunteer more information on themselves, but they are not required to do so.

This low entrance threshold means, of course, that we cannot trust any of the user-supplied data to be true, which in many other communities (such as Slashdot, for example) is considered a feature. For SELF, however, being able to establish a verified link between the online identity and a real person may be useful, for it would allow members to assess other people's expertise in different areas also through their off-system work.

The SELF platform may in future, should the community decide that this is an useful feature, offer alternative cryptographic sign-up and login protocols which will provide extra security to each user's connection, as well as mechanisms for real-identity verification. Protocols and infrastructure to these ends are already partially in place, such as public key signing protocols and verification procedures to assess the authenticity of a key to any desired degree of confidence (more confidence comes at the price of more work and inconvenience, so a balance must be struck).

Even if the SELF platform never attempts to give members credit for off-system activity, enabling mechanisms to verify identity may still be useful. People trust a signed work more than they do an anonymous one, and thus an LO by a verified author may have some perceived bonus over one that was authored without verification. The degree to which a user is willing to expose his or her real identity online can also be used as a measure of commitment, and thus by itself affect that user's reputation.

Contributing to the community

The bulk of the SELF platform will be concerned with enabling and facilitating member contribution. It is members who will contribute materials, produce Learning Objects and combine them into Courses. SELF will merely provide an environment in which this activity can be approached in a cooperative way. In a very real sense, SELF's livelihood depends on member contribution: unless member contribute, SELF will have very little to show for itself, and so the environment must be as encouraging and low-threshold as possible.

SELF's participation mechanisms will be geared towards advancing the project's specific goal of cooperative production of educational material. This is different from the way the Debian project works, for instance, in as much as every Debian developer is individually responsible for a number of packages, whereas in SELF responsibility is carried collectively by the people who choose to participate. It is also different from the approach used in many blog communities, in which each person contributes short independent articles, each one of them attributable to a single author. Instead, the SELF platform will have to keep track of each Learning Object's history, who contributed which parts of it, and how the changes affected the LO's acceptance by the community.

Acknowledging contributions

All on-line communities have ways of keeping track of its member's reputations, in varying degrees of formality. Communities that are embodied in formal institutions, such as Debian, even have formal authorities, officials and procedures that are decided through member's votes. Slashdot, a

popular technology blog in which users discuss current issues, computes a “popularity index” for each user, according to how other members rate the user's posts, allowing for both positive (“Insightful”, “Funny”) and negative (“Flamebait”, “Troll”) feedback. Other communities, such as mailing lists, rely entirely on social dynamics to identify those members who contribute most.

The ability of users to subjectively rate each others work, which is useful for sites that have mostly entertainment value like Slashdot, is not adequate for SELF's goals. Still, a similar idea can be used to rate each member's contribution to the community. Since the idea of SELF is to have members use each other's contributions, we can glean a better idea of what members think of each other's work by looking at how they use it, than by looking at what they say about it. This approach has two significant advantages over having users directly rate each other's work. First, it is less work for the user: instead of asking them to rate materials as a separate task, we just look at which materials they find useful enough to actually use. Second, it is more likely to be accurate: idly rating someone else's work may or may not yield dependable results, but actually using other people's work in your own is a true sign of appreciation.

Since SELF will have to keep track of each member's contribution to Learning Objects (of which courses are the general case), the platform can compute how many people are using any particular material, and establish the link back to the people who have been involved in its production. A deeper analysis can correlate the evolution of an LO's popularity with the degree of involvement of the user in it.

For the SELF project, a carefully designed user rating system can be extremely useful when combined with a system for rating materials, since both ratings can feed each other back. For instance, contributions can get a basic rating boost proportional to the participating contributors' rating, since it can be expected that they will do a reasonable job, and feedback on materials by reputed users can have more weight than feedback from anonymous, or even ill-reputed members.

Taking into account that the SELF platform will be powered by the semantics-aware gnowsys project, authors' contributions can be analysed to identify their areas of expertise, so that the impact of their contributions on the material's ratings can be modulated accordingly. Similarly, material that is widely used by other members can be assumed to be of good quality, and this can give its main contributors a rating boost. The user rating system can also be used to find ways to reward contributors who participate in particularly desirable ways.

Having a user rating system makes it possible for the platform to have a very shallow hierarchy. While it will probably be inevitable to have a split between users and administrators of the system, a good user rating system will enable the SELF platform to give differentiated treatment according to their involvement and merits, and not according to privileges allocated to them by “authority figures.” This helps users feel that they are part of the community, and that they have some say on what goes on with it.

How to rate users

For social networking sites, the number of connections of each member is probably a good indication of reputation. Similarly, for “journalist/discussion” sites such as Slashdot, just aggregating people's reactions to each of the author's posts is enough to compute a rough popularity index. It is likely that, for a repository as varied in content as the SELF platform aims to be, and which aims at being more than a popularity contest, a single, scalar “user rating” value will not do the job. Instead, a structured set of criteria may be more adequate, particularly if enough objective criteria can be found which can be evaluated automatically rather than through user feedback.

Time will be a factor in rating user actions: the system will analize the fate of contributions over time, rating those that survive for longer time a higher rating than recent contributions, for survival in this context means that all people who revised the document felt that the text was hood enough to keep.

The set of criteria to allocate credit, and their respective weights, will probably be perpetual work in progress in order to adapt to changing social dynamics and requirements of the platform. A preliminary set of criteria may look as follows:

Contribution Ratio For each material the user has contributed to, the system will keep track of the ratio of the user's accepted contribution to the total size of the LO (A_{user}/T_{LO}). This ratio will be multiplied by the Accepted Contribution Ratio (see below) to apportion the credit from the material's popularity to each member.
Credit also get apportioned only after assessing the popularity of the material, because that's when we can see whether the material has been accepted or not.
A question that is still open is how to measure the "size" of a contribution. In text, there are workable approaches, but in other media, such as graphics or video, where the amount of work does not at all relate to file size, we will need to find an appropriate metric.

Harvester's credit Harvesters of materials get a "finder's fee" in terms of credit, proportional to the popularity of the resulting LO, and inversely proportional to the LO's age (i.e. the credit fades with time). Since LO's in areas where there are few materials are more likely to become popular, this acts as an incentive to seek out materials for subjects that are lacking, making "credit bounties" for materials less likely to be needed.

Contribution to Quality The platform will keep track of material quality, in part through computable objective criteria, such as standards conformance, granularity of tagging, etc., in part by keeping track of quality issues identified in the LO's in a manner similar to a bug tracking system. When the user's contributions results in measurable improvement in material quality according to those criteria, the user gets an additional portion of the LO's popularity accredited. This bonus is proportional to the percentual increase in quality.

Density of Involvement The ratio of each user's recent activity in the system to the average of user activity since they started membership gives a weighed boost or penalty to the overall user rating. Sustained, and particularly recently sustained activity gets a higher boost than spiky involvement.

Expertise in Specific Area More than a single criterion, this rating is a function of the area in question. When evaluating a user's contribution to a material, or feedback on it, the system can compute an expertise index by evaluating how the user compares to the average when his or her rating is computed in a specific area, leaving out others.
Assuming that SELF moves at some time towards verified identities, admins will be able to credit rating points based on the member's reputation on the field outside of the SELF platform, according to rules

that will have to be decided inside the community, thus acknowledging renowned academics or authorities in the field.

In order for the system to be able to evaluate these criteria, it is crucial that it keeps track of each user's contribution to each version of each LO, and that it includes a function to periodically review the state of the knowledge base to recompute each user's rating according to flexible formulae. The exact way how these indexes will be computed will be a subject of experimentation. Our aim is that the methods for computing them will eventually be decided by the SELF community itself, either by means of collective construction (through user-supplied formulas that can be shared among users). Although we have presented the criteria as being computed by means of formulae involving measurable quantities, it is well possible that methods for detecting statistic correlations between some LO's histories and specific users' involvement in them can yield good, maybe even better results.

For more details about the Quality Assessment mechanisms see Deliverable D8, the results of Workpackage 8.

Main functions

In this section we will define the main functions of the Platform, which in the following sections will be defined in more detail. The SELF platform will provide the following functions:

- harvesting of knowledge objects: importing mechanisms
- authoring environment
- version control and package release
- multi-language aspects
- translation tools
- delivery option through online web interface and (in the future) through offline clients
- user management
- user pages, listed according to different rankings from the user rating system.
- quality assessment mechanisms
- CMS functionality
- promotion of projects and community activities through news items, a Project-of-the-Month (in the highlights section), etc.

The Platform will be accessible through a standards-compliant web interface (tested against the W3C requirements) and will work in the top 5 browsers (including Internet Explorer, even if it doesn't conform to the W3C standards). Besides, a portable platform will be made available, which will work on any operating system supporting Python. This will be distributed on CD or DVD including a selection of the Learning Objects. Future versions of this portable platform will include “package managing” capabilities to update the available LO's, and to import new objects into their knowledge base.

The various elements will be discussed in the next paragraphs.

Harvesting

One of the basic constituents of SELF platform are reusable learning objects. If the content of each learning object is sufficiently independent of the lesson, the reusability of it enhances. Though it is a big practical question to obtain lesson-independent and course independent learning objects, it is

desirable to inform the users of the SELF platform, that reusability enhances the value of each contribution. This helps people to combine each LO in different ways to instruct on different, although related, topics.

This requirement introduces a constraint on the kind of authoring environment the platform should offer. Most resources that already exist on the Internet, and also suggested by WP2, are often very large documents, which are authored for a specific purpose. Such large documents are difficult for reuse, though the content in them can be modified for reuse.

Keeping in mind this requirement, it is suggested that the SELF platform will have an "Atomizer". An atomizer is a GUI that helps authors to identify and extract each independent component and its content, and add to the knowledge base a learning object. The idea is to facilitate the division of a large document into several small LO's, only to recompose them back into a structured document.

Fortunately, substantial amount of technical documentation of free software does already exist in structured form. The Unix man pages, and the texinfo pages are well structured, so that their content can be automatically harvested into the knowledge base by writing scripts. Apart from the man and info pages, several documents are available in Docbook and LaTeX format, particularly from the GNU project and The Linux Documentation Project (TLDP). Again, these formats are also amenable for automatic harvesting. It is therefore proposed that the SELF Platform will have a form where structured files can be submitted into the knowledge base.

Such documents will then be automatically atomized into several small sections, as suggested by the markup which already exists in the document. This will provide a large amount of LO's in the knowledge base. However, these automatically harvested LO's need to be edited for ensuring that they are usable and reusable. Therefore, a GUI to search and edit the automatically harvested LO's is required. This functionality won't be built during the EC-funded period (Phase I/II), but will be in the Roadmap for phase III.

Managing Collections

Through the Collection Manager, a user can access existing learning materials by browsing through categories or collections of learning materials. Each collection has a title and description and can be a subcollection of another collection. Learning materials can be in one or more collections. Logged in users can add new categories, which will be added after the last (sub)collection in the hierarchy. Only users with specific manager privileges can edit and organise collections.

Initialisation of a Learning Object

Any user may start a new LO in various ways: 1) copy an existing Learning Object and start a derived project. 2) translate an existing LO into another language; 3) start a new LO in the harvesting process and 4) start an empty LO.

- LO's have metadata: some minimal classification is required, including the license and authors, the level, the category/ies of topics it relates to. Metadata are specified by the LOM and SCORM data models, see IR3.1

Authoring Learning Objects

A Learning Object can be edited by the authors in the following ways:

- Adding and Editing operations for the metadata, and designing the ontology of the knowledge base (handles metadata, should be automated as far as possible)
- An easy-to-use editing interface for editing parts of the learning object individually, instead

of opening a 30-page chapter in one screen. Follow the example of Connexions[2] where authors can "Edit-in-Place" only small paragraphs that can be activated by the author and then edited.

- We must cater for the possibility that several users work on the same objects at the same time. Revision control techniques can take care of concurrent editing, and we must implement it anyway because most advanced users will want to edit offline.
- Besides the "Edit-In-Place" option of editing one paragraph, it should be possible to edit the whole LO (see Connections: "Editing Full Source")
- Visual editing options to enhance the usability. Existing editors that can be used are: Kupu[3], TinyMCE[4], NicEdit[5] or the Wikipedia wiki-format editor.
- Handle images, audio and video objects easily. This does not include the editing of the objects, but refers to details such as position and size of the objects within the LO. The authorship and license of the objects should be visible.
- Metadata: as per the LOM standard, an interface be made for specifying the metadata for each LO and course. Some of the metadata are collected by virtue of who is changing, and at what time etc.
- Each course will have a set of learning objectives, which will define the scope and objectives of the course.
- Links: a provision to add internal links to other LO's or to external objects as references, or suggested for further reading. This needs an interface to establish relations between LO's and name the relation accordingly.
- Tests: just as some of the LO's are meant for referencing, some of the LO's play the role of assessment or test. In the authoring environment, users should be able to add test topics and test questions and answers under various test models (multiple choice, single correct answer, multiple correct answers).
- Glossary: terminology and concepts used in the course can be defined and explained in the glossary of terms. Glossary terms can be directly added from the authoring environment.
- Frequently Asked Questions: question and answer pairs can be accessed by FAQ topic. Users can add topics and FAQ questions and answers within each topic.

Organising courses

Once we have the learning objects in the knowledge base, the next authoring process is to organize them into a lesson, and lessons in turn into courses. This requires the following:

1. Lessons or chapters:

The various LO's can be grouped into several lessons and sublessons (or chapters and subchapters). The organiser allows for the rearranging of the hierarchy and order of lessons (or chapters) within a course (or learning material).

2. Sequencing:

The order in which the teacher wants the learner to take the lessons, would require the LO's in each section to be ordered in a sequence. An interface to move objects up-down-top-bottom will be required.

2 After having registered at CNX.org edit a module in your workspace, for example: http://cnx.org/Members/wtebbens/m12403/module_text

3 http://kupu.oscom.org/

4 http://tinymce.moxiecode.com/

5 http://nicedit.com/

The sequence is not always linear. For example, the teacher may want to give the option to the student to go for either 5th, 6th or 7th after going through 1-4 in a linear order. This needs a feature to group a few LO's suggesting any of them can be taken up (this can be realised in phase III).

Multi-language aspects

The Multilingual aspects of SELF come in three ways: firstly, the SELF system and system messages should be accessible in the various languages; secondly, content elements from the CMS-driven parts of the web platform should be accessible in the various languages and thirdly, the LO's and courses, or in other words: the complete SELF repository should be accessible in the various languages. In all of these localization requirements, the system should be capable of handling different types of character encoding, both left-to-right and right-to-left.

Translation

The basic principle of the translation of a Learning Object is that it creates a new LO, in the selected language. After that step it is an independent LO, but has a relation with the original LO, stating that one is the translation of the other. At the moment of starting a translation, when the new object is created, it inherits the metadata properties of the original object.

The translations of LO's should be clearly related in such way that the system can notify interested users of the creation of a new translation of some LO. Either the translations of LO's are different LO's or they are the same LO in different languages, in any way the system should keep track of the fact that they belong to the same LO, or at least once started as the same LO.

We need to provide for an interface that facilitates translators in their work. Both the original and translated part should be visible (in two columns). However translation assisting tools in SELF will be always very limited: professional translators use specialized tools to aid them in their work, which we cannot duplicate here. Instead of developing complex translation tools, we will concentrate in providing good support for off-line editing of LO's, including version control, so everybody can do the job the way they best see fit by exporting and re-importing the translated files again.

Version Control

Learning Objects in SELF are perpetual works in progress: people may make changes to them at any time. The system, however, keeps track of all versions, and most references to a LO (for instance in a course, on the user's Bookshelf, quality ratings, etc.) are to a specific version of it, not to “the latest version”. Thus, an LO may have varying degrees of quality or completeness over its life cycle, and users have complete access to all of the document's history. The system can also automatically gather information about versions. For instance, if a given version of the LO is more present than others in courses or Bookshelves, the system can highlight these versions as the most likely to be stable. In this respect, courses act in a similar way as configurations do in software development: an course is a combination of a given set of LO's, each at a specific version.

To assist users in exploring various versions of courses, a specific user interface of comparing of versions shall be made. This UI will show the version tree and by selecting two versions, the differences between the two can be highlighted.

Support for Quality Assessment

One key contribution of the SELF platform is its potential to deliver high-quality materials. At first sight, this aim is not easy to reconcile with SELF's ambition of being a platform for collective construction of educational material: if only high-quality material is to be delivered by SELF, then by definition the material cannot be hosted on the platform until it's "ready", and so it cannot be produced collectively.

Another apparent show-stopper to the concept that SELF can provide quality-approved materials is the fact that the platform aims to cover a very broad field of knowledge. If we were to do quality assurance the traditional way, we would need an army of specialist on each and every subject.

Quality as emergent behaviour

These two obstacles indeed make it impossible for the SELF platform to deliver *quality-assured* materials, so we will not even attempt to do it. Rather, we will provide the infrastructure for the platform to publish materials that are *quality-assessed*. Users will then be exposed to materials which are not necessarily top-quality, but they will have mechanisms to find out which ones are considered to be better than others.

Instead of relying on a large number of experts to review the materials and rate them accordingly, we will keep track of each material's popularity, i.e. how much attention each LO draws to itself, as a means to infer the LO's quality. To this end, the platform will provide users with specific facilities which will provide information on each LO's popularity without demanding any extra work from them.

Besides working around the obstacles mentioned above, this approach has the added benefit of avoiding a common pitfall of scientific publishing: the silencing of novel ideas by established authorities. SELF will offer a place even for far-out ideas who may not be popular at the time they are first put forward, but which can be recovered when the state of the art has advanced or shown some new insight.

Guiding principles

Assessing the quality of the materials will be difficult to achieve with a single approach, so we will use a combination of three mechanisms, two of them automatic, to figure out each LO's quality. Another implicit design criterion behind this approach is to minimize the amount of extra work that users must do in order to achieve the goal of assessing quality.

Automatically measurable objective criteria

Some quality-related aspects of the LO can be measured automatically by the system, such as standards conformance, level of disaggregation, presence of exercises and test questions, amount of detail in metadata, modification-friendliness of the format, and even legal aspects such as being under FLA or having certified authorship.

In order to support this feature, the system will include a facility to define a number of such criteria, and to routinely measure each LO against them, computing an index for each one of them.

Manually measurable objective criteria

There are objective criteria regarding LO's which are hard to be evaluated without the intervention

of a human brain. These include issues like clarity of exposition, orthographical and grammatical correctness, technical accuracy, suitability for the stated goals, and many others. In order to get some feedback on these criteria, the system will keep answers to a structured questionnaire associated with each version of each LO.

Users will be able to answer this questionnaire in full or in part, and to review their ratings at any time. Later versions of the same LO will "inherit" their ancestor's ratings, unless the same user re-rates them, so the system can actually track the evolution of the LO's perceived quality over time. Anonymous users will not be allowed to rate LO's, and feedback from authors above a certain (adjustable) threshold of authorship in the LO will be disregarded. The impact of the rating by each user will be proportional to that user's rating.

In order to support this feature, the system will include a facility, available only to the administrators of the SELF platform, to manage the questionnaire. It will enable users to enter questions that can be answered by the user in a structured fashion by cho sing from multiple choices, associate a specific score to each answer, and define ways to combine the scores into one or more aggregate score.

This will be complemented with a facility to display the questions to users, and gather their responses.

Popularity of the authors

The idea between the quality inference mechanism is based on two heuristics: one, that the quality of materials by the same author usually is more or less uniform, and converges over time to be proportional to the author's reputation, and two, that the fact that people are more likely to focus their attention on high-quality than on low-quality materials.

We have already outlined the way how the quality of an author's output will affect his or her user rating on the system. Closing the feedback loop, we will use the author's reputation in the system to influence the quality rating of learning objects in which he or she has participated. The impact of the authors' user ratings on LO's will be, of course, proportional to their involvement in them, and the feedback formula will be computed in such a way as to avoid up- or downward spiralling.

Popularity of the LO

Gauging which LO's catch the users' attention is a bit trickier. In order to do it, we will take advantage of the fact that users will have to gather LO's in some sort of "working area" in order to keep them handy, and to rearrange them into courses. We call this working area "the bookshelf", and each registered user has one. By looking at which LO's are present in users' bookshelves, we can infer each LO's popularity, and we can even modulate the impact of the presence of an LO being on a bookshelf according to the bookshelf owner's user rating. An LO which is present in the bookshelf of a user who has contributed significantly to it does not get any rating boost, and we will artificially keep the capacity of the bookshelf limited to a large enough number of entries, in order to make bookshelf capacity (which may be a function of the user's reputation, since we expect reputable users to use the system responsibly) a valuable resource.

Using an LO in a course is an even stronger endorsement, and thus the presence of an LO in many LTs is also a good indicator of quality, with the added advantage that LTs are Learning Objects themselves, which means that they can also be put in bookshelves and are also subject to quality rating that can reflect on the quality of its sub-LO's.

Version tracking in bookshelves and courses

The SELF platform keeps track of the different versions of each LO, enabling users to follow their history, to improve on any version they wish to, creating a branch each time they improve on a non-head version, and to merge changes from different branches. The branching capability is key to accommodate exploration of different approaches, and even for dissent.

An important feature of bookshelves and courses is that they lock onto a specific version of each Learning Object, i.e. they don't automatically upgrade their LO's, they leave this decision to the user, who also has to choose along which branch, and up to which version, he or she wishes to upgrade. To aid in this decision, the system will inform the user of new versions and branches of each LO, together with indication of their respective quality ratings so far to aid in the decision.

When a user upgrades one or more LO in a course, a new version of the course is created, and the original revision is kept on the system.

This has several important implications:

1. users can rest assured that their courses will not automatically change into something else while they aren't looking;
2. tracking the shift in preference between versions is a strong indication of quality improvement;
3. the SELF platform will be able to do without any Wikipedia-style changefests: unpopular or incorrect contributions don't need to be rolled back, they can stay on the system while users branch and upgrade around them.

Content Management System

apart from the LO's, the system presents specific pages about the Organisation of SELF, its Partners, news items, in other words: the main navigation menu sections. A CMS to maintain these sections is needed. We plan to use the Plone CMS on top of the semantic engine.

News Editing

This is a part of the CMS functionality to be provided. The SELF project will provide users with a community-driven facility to publish relevant news, through the use of a suitable content management system. Publication in the news facility will be subject to moderation by an editorial team. The facility will provide an RSS feed which will be displayed in the SELF's platform front page, so featured articles can get the attention of users.

User Management

User Management involves the creation, changing, duplicating and deleting of the following aspects:

- User Roles
- User Reputation Management: adapting of criteria, thresholds and other parameters.
- User Preferences
- User personal data
- User blocking

User Management can only be performed by users with administrative privileges.

Administration of the Platform

- edit and delete user accounts

- blacklist IP addresses and perform all other user management activities. Most probably a user will not change often computers from which he/she would register and disturb the system i.e. spam etc.
- deleting operations for users, learning objects (LO), courses and news items.
- add, edit, delete user interface templates
- view statistics of usage, contribution and feedback mechanisms,
- adjust parameters of the User Rating System, thresholds for attribution and quality mechanisms
- ...

URL structure

The domain name to be used is http://selfplatform.eu. Other domain names can be added to it as aliases. For reasons of usability for humans as well for search engines we need “clean URL's”. The exact setup of the URL should be defined, and should include some reference to the title of the project. These criteria apply to the URL setup:

- includes a unique title of the object
- includes a version or release number, which have a different setup: version numbers tend to be longer than release numbers, as only some versions are published as a release.
- avoid redundant variables that could confuse the human user or search robot
- includes format information if the object is packaged in some of the selected formats

Some concrete suggestions for URL structure:

- Homepage: should equal the domain name, i.e. <domain>
- Materials (=Collections): <domain>/collections
- A specific collection: <domain>/collections/<collection-name>{view,edit,delete,addcourse} where <domain>/collections/<collection-name>/ calls the view mode of the collection with collection-name.
- A course: <domain>/lo/<ssid>/{view,edit,changes,history,discussion,organise,translations} where <domain>/lo/<ssid>/ calls the view mode of the LO, as well as <domain>/lo/<ssid>/view. "Edit" calls the editor, "changes" the view changes, "history" the change history, "discuss" the discussion space and "organise" the organiser of the course. Adding the "lo" part is suggested to reflect that it refers to a Learning Object and not a collection.
- The fileManager: <domain>/filemanager
- The userpage of a user "username": <domain>/user/[username]/
- The Shelf manager of a user "username": <domain>/user/[username]/shelf
- The user's preferences: <domain>/user/preferences
- Future usergroups (teams) around a certain common topic of interest: <domain>/[groupname]
- and likewise: <domain>/[groupname]/members, <domain>/[groupname]/[materials] and <domain>/[groupname]/files, etc.
- For searching a intuitive query should be built up starting with <domain>/search/q= or similar.

Standard formats to be used

Standard formats have been analysed and selected in Work Package 3. For details, see Interim Report 3.1[6].

According to the analysis provided as reported in the IR 3.1., the first list of supported formats is provided below. The preferred formats (open standards *de iure*) are written with boldface fonts and underlined:

Category	*Supported formats*	*Category*	*Supported formats*
Unformatted text	**ASCII**, **ISO 8859**, **Unicode**	**Formatted text**	**ODT**, **DocBook**
Scientific text	**ODF**, **MathML**, Tex/LaTeX	**Raster images**	**JPEG**, **PNG**, PNM, GIF, BMP
Vector images	**SVG**, **ODG**	**Video**	OpenEXR, Theora, RIFF AVI
Printed-oriented	**PDF**, PS	**Hypertext**	**HTML**, XHTML
Presentation	**ODP**	**Audio**	Vorbis, FLAC, RIFF WAV
Others	**tar**, **gzip**, **XML**, 7z		

Peer-to-Peer Distributed Network

An optional feature of SELF is its distributed nature. A single SELF Platform will be maintained at a central location which will aggregate all contributions. As the knowledge base increases in size, replicating the server platform in multiple locations will be difficult. Different locations may not need the entire knowledge base. For example, not all the installations of SELF platform may need courses in Spanish, or a school in Spain may need only Spanish language learning objects. Remote locations in India or may be Africa, may not have sufficient connectivity or other resources to maintain a very large mirror. Mirroring servers need to be intelligent enough to copy only the differences or the updated learning objects. Keeping in mind the above scenarios, the following peer-to-peer network between the different SELF Platforms is proposed.

A SELF Platform instance will be configured as primary or secondary server, just as there are primary and secondary Domain Name Service (DNS) for resolving the domain names into IP addresses and vice versa. In the case of SELF platform, however, each learning object is analogous to a host computer. Since the knowledge base will be populated in the GNOWSYS server, every LO will have a unique URI, and metadata such as time-date-stamp, last modified etc. Servers will keep information about the peer servers, and the type of server, such as primary server or secondary server. Each server will also keep a list of categories suggesting the part of knowledge base. For example, a server's category list may contain, Spanish, Mathematics, Computer Science and Operating Systems. This server will receive updates from the primary server of only those LO that are members of the specified categories.

When a server P is configured as primary with respect to the server Q, and the server Q is

6 See IR 3.1. report: http://selfproject.eu/en/system/files/ir3.1_0.pdf (248 kB)

configured as primary with respect to the server P, any change in any LO in any of the servers P and Q will be copied to the other. This setup will make them mirrors of each other. When a server P is configured as primary with respect to the server Q, changes made in P will be copied to server Q. However, the changes made in server Q will not propagate back to P.

Since each LO is a very small container of information, the updates happen in a highly granular manner. This kind of distributed peer-2-peer network of servers will reduce the burden on the centralized server, since updates can be propagated from the nearest server nodes. Since contributions to local machines is easier and faster, people in the remote areas with limited connectivity could also contribute to the global network, and at the same time they could also take the benefit of the global contributions.