Lenya 3.0

This document is a first draft for design the future Lenya major version.

Useful reading and information have to be picked up from this documents :

- RoadmapLenya3.0
- WishlistLenya2.0

Here comes some headings that have to be filled down. Headings and content are proposals, so don't hesitate to modify, suppress, etc...

General Requirements

- Automatic content migration from old version to the new one
  - LenyaDocTransformer can be a begin of a solution with his capability to create documents from xml with api usage.
  - Another option: Export to standard format + import (similar to JCR)

Use Cases

- ```
  write here you how-to use idea, like if you are a simple user
  ```

- Use Case 1 : Chain modules
  - be able to easily (via user interface) to chain modules in order to create a new process in my publication.
  - For example, I have in my module stack :
    - A - upload module (build interface for the end user)
    - B - Write to file system module
    - C - To html parser (ie Tika)
    - D - Lenya document creator module
  - so I can define via administrative interface :
    - process I) A --> B
    - process II) A --> C --> D
- Use Case 2 : Forms by clic ²
  - Be able to define forms with clic and decide what I want to do with this after validation (write in a resource, write on FS, put in database, send by mail,...)

Features

- ```
  wish list of all features you would like to see
  will be organize / topicazed soon.
  ```

Needed

- all is module, core is restrict to his more simple expression
- easy way to integrate webservices
- GUI module configuration
- more "2.0":
  - user can create his account
  - possible to comment each document
- hook feature : each module can define pre or post hook for predefined function, with this, module don't be intrusive into existing process, easily add features on it
- user interface
  - workflow more closed to the article
  - suppress "site" tab and put all functionalities into "authoring" mode
- molecular sharing system :
  - each user can define one or more "friends group" and share with it read, write, comment access. 4 bases levels : individual, personnal groups, site groups, world
- easy way to create forms
- forums
  - a definition for forums functionalities : https://issues.apache.org/bugzilla/show_bug.cgi?id=36192
- blog
- synchronisation between different instances (via svn for example)
- possibility to edit site when not connected
- links with mail-box
- import Ooo 3.0

    - be able to edit imported Ooo 3.0
- presentation / slide module
- MetaDatas :
    - Propose a more flexible MetaDatas solution for be able to deal with any xml format (like RDF, OWL,...)
    - metaData Navigator : be able to navigate trow documents by date, autor, key-words, ...
- fine grained workflow
    - be able to define particular workflow for a path in the site structure (so a part of the site can become wiki, and the other verified area)
- fine grained users/groups rights :
    - navigation by profile : https://issues.apache.org/bugzilla/show_bug.cgi?id=36193
        - target group oriented navigation (prefered interest e.g. industry specific of a certain group). Each Content page can be adressed to a specific User group (target group)
    - admin access by profile : https://issues.apache.org/bugzilla/show_bug.cgi?id=37403
        - create a user-group which has restricted access to the admin-area
- Authentification and authorisation :
    - use an existing lib like this :
        - http://download.oracle.com/javase/6/docs/technotes/guides/security/jaas/JAASRefGuide.html#Subject
        - or this : http://shiro.apache.org/
    - search for the powerfull and simplier solution.
- Unit-testing on sitemap's results.
- New UI template processing : have a really content <-> presentation independance : 3 majors ideas
    - 1) All the template in one folder
    - 2) UI architecture/structure is just another content
        - The build of this structure can be done via the "forrest template mechanism" for lenya templating : refer torsten's answer here http://www.mail-archive.com/user@lenya.apache.org/msg06961.html
    - 3) Content can be place anywhere in the structure
        - For realise this point, we have to define a "block-template" interface for each module.
        - This interface is just a pipeline match into the module that output an xml with sections head/body/footer
        - Just have to write the block-template interface in the structure for having an "in place I want" result
    - This photo represent a first draft done during the MeetingBordeaux2010 : template.jpg
- Email Notification/reminder to new users
    - see tickets :
        - https://issues.apache.org/bugzilla/show_bug.cgi?id=29273
        - https://issues.apache.org/bugzilla/show_bug.cgi?id=29279
- search improvement :
    - 1) use solr
    - 2) an idea from ticket : https://issues.apache.org/bugzilla/show_bug.cgi?id=33702
        - The lucene integration should expose more details about the search, and it should be easier to add new fields to the index via a GUI.
        - http://www.getopt.org/luke/ might be helpful
- use HTML5 and CSS3
- Easy to use, an excellent site administration.
- Good performance in big sites (better Lenya 2 than Lenya 1.2)
- Standars compliance.
- Content Comments
    - be abble to add comments/ reviews / evaluation function to all Lenya contents (document, photos, ...)
- Be able to login with shared identification services (facebook, google, etc...)
- Have routines and easy integration with integration with famous social media... twitter, facebook, youtube, issuu (http://issuu.com/)
    - have a built-in easy to use there api
        - note : there is a component near from that in development in the apache camel environment
- GUI composition module :
    - Be able with clic and mouse to configure position of contents, blocks, menu elements in a Lenya graphical template.
        - See the Drupal way of do, it's really cool and easy.

whised

- implementation of webcalendar calendar format propose calendar service

user interface

- ```
  Here comes idea about the UI design (nice pictures or hand-made scan are welcome)
  ```

- be more user friendly, interactive, web 2.0

Architecture

- ```
  main core design lines, libs and external systems that are used
  ```

- Andreas' Potential Architecture Diagram : architectureimage.jpg

Principles

- clean all Lenya specific implementations with standards ones when possible :
  - Content repository ==> jcr
  - workflow ==> ??
  - ??
- use maven for building

which cocoon version ?

```
Lenya is actually in 2.1, 2.2 was "stable" and out from a moment and C3 is actually under development in alpha
stage. Which version will be use ? Let's discuss about possible scenarios and pro and cons for each.
```

- Cocoon 2.1 :
  - pro, with cleaning and optimisation : http://mail-archives.apache.org/mod_mbox/lenya-dev/201006.mbox/%3Chu8074$pj1$1@dough.gmane.org%3E
- Cocoon 2.2 :
  - con : http://mail-archives.apache.org/mod_mbox/lenya-dev/201006.mbox/%3Chu8074$pj1$1@dough.gmane.org%3E
- Cocoon 3 :
  - pro : http://mail-archives.apache.org/mod_mbox/lenya-dev/201006.mbox/%3C1275640168.2879.29.camel@mcKenny%3E
  - hesitate : http://mail-archives.apache.org/mod_mbox/lenya-dev/201006.mbox/%3Cdbd818479e6a63d14fefa2c3db7a8114@4sengines.eu%3E
- Cocoon 2.1 with module by module migration to 3
  - see :

process

```
the way works the app
```

- GUI template
  - separate into the actual (2.1) template idea :
    - template configuration : inheritance of users, workflow configuration,...
    - template GUI : independent, more flexible mechanism that allow to easily share template into community

api

```
api have to be simple, understandable, etc... but it's not easy ! Let's discuss about it here.
```

Ideas to reuses

- this ideas are from an old 3.0 roadmap. * some of this ideas have to be reuse into this document :

Options

A: Evolution

- Preserve backwards compatibility

B: Clean Cut, Reuse Architecture

1. Migrate to Cocoon 2.2
2. Replace content repository with JCR
3. Start with minimal core
4. If possible, replace home-grown stuff with out-of-the-box components (a lot has happened since our components have been built)
   - Identity management: no internal user storage (maybe proxies), SSO support (OpenID integration etc.)
   - Workflow engine
   - GUI framework (GWT, Dojo, ...)
5. Migrate modules step by step

C: Clean Cut, Reuse Experience

1. Reconsider all requirements thoroughly
2. Derive architectural constraints
3. Choose architecture
4. Reuse code where it makes sense

Architectural Constraints

- No changes to URL space and mapping between URLs and pages required
- Workflow-driven content manipulation
- Access control on repository level - permissions assigned to content objects rather than URLs