CommitterInfo 2fMeeting05072005

DSpace Committers' Meeting, 5 July 2005, University of Cambridge

Attendees:

- Jim+Downing (JimDowning) (University of Cambridge) - JD
- Rob Tansley (RobertTansley) (HP Labs) - RT
- Richard Jones (RichardJones) (University of Bergen) - RJ
- Scott Philips (ScottPhilips) (Texas A&M University) - SP
- Scott Yeadon (ScottYeadon) (Australian National University) - SY
- MacKenzie Smith (MIT) - MS
- Gabriela Mircea (GabrielaMircea) (University of Toronto) - GM
- Richard Rodgers (RichardRodgers) (MIT) - RR

Update process and 1.3 release:

- The process of update is not too hard, but we are having problems getting code reviewed and tested by the community
- If we set a standard for patches, we can also start to raise the bar on the quality of contributions
- Would starting to branch the code be a good idea: e.g. maintain a 1.3.x branch and move on to 1.4 asap.
- Make it clear that testers can just check their own usage and record which aspects they have checked.

Patch Contributions:

- We need to make patches faster and easier to commit
  - Need better information in patches that enable committer to easily validate the changes.
  - Patches need to be smaller
- Improving overall software quality
- **We should use a code style checker (Check***Style) and other formatting tools (Jalopy, Jacobe)
  -
    - Raise the bar on contributions
    - We should have a skeleton patch file that people could work to: Config options for every feature, UI internationalisation, Online help and systems docs
  - Large patches against the HEAD can cause problems with the branch
  - Perform large patches only straight after a major release

Release Cycle/Development Processes:

- Unsure whether small quick incremental releases are better than large releases
- Small incremental changes will be quicker getting into production, and may get people used to regular updates
- Could we put the eclipse project files into the CVS
- Is there a better way we can support non-core development: have our own SVN server? Better way of building add-ons into the code using ant?
- "DSpace in a Box"
- We should, after DSpace 1.3, go to Java 1.5 as a source code standard. We will still have the option to compile for Java 1.4 as the bytecode is back compatible.
- Many small changes may cause problems for users
- Are timeboxed betas a good idea?
- Add sandbox module to CVS?

User Interface:

- General consensus that JSPs are bad
- We may be stuck with JSPs for a long time
- Can we frame requests in some kind of object which would at least make validation and error tracking easier
- Cocoon experiments, but there are modularity issues which make things complicated for implementors
- Why not try several UI approaches?
- If we remove a lot of logic from the servlets and JSPs we can make the UIs easy to implement
- Plugin inclusion into the Cocoon sitemap would have to happen at build time
- End user interface could be ready for 1.5

Modularity:

- Could we implement a middle tier API behind the servlets allow them to become much more simple, and keep functionality behind interfaces
- Could we use WebDAV

- Can we increase db abstraction by using a tool such as Hibernate
- We do not have the resources to do a rewrite from scratch for DSpace 2.x
- JD warned that incremental development is not an easy option, requiring more effort in development and user support in the long run.
  We can do this incrementally, and should include the Package*Manager code for 1.4 (PackagerPlugins)
- Value objects behind JSPs and forms

Documentation:

- We should have a particular document which explains how to check out and install DSpace quickly and easily
- It would be good to have smaller XHTML files for the docs and a framework in which to expose them.
- The current online tech docs are now out of date - can we get make [CVS backed and give the committers access to the CVS?
- Documentation should be moved to the dspace CVS module for ease of management

Asset Store:

- Recreate working data from clear set of data stores
- RT presented his initial new asset store proposal: this would be transactional, and we will need synchronisation between the db and the AIP storage in some way.
- Could automatically generating METS files with bitstreams be a good first step; this is a reliable place to put extensible metadata with checksums. Is this too small a step?
- What do the AIPs look like? Do we have implicit agreement on METS?
- We need to figure out a way of evaluating the different asset store models
- Some essential information is only in the database (e.g. authentication) which may cause problems in the assetstore layer.
- Is there a way that we can separate asset store data from other data in a way that allows us to keep the linkages to other data in a reliable way? This may still not be able to deal with the possibilities of inconsistencies in constraints of the front end data
- Could we insert a validating interceptor stack in the transaction as it inserts data into the assetstore
- METS practice is to represent the communities and collections using a bottom up approach; problem is we may not necessarily want an item's collection data in other repositories (therefore perhaps we should do this top down instead/as well)
- The assetstore should fully decouple the content from the application (i.e. no epeople and authorisations and so forth)
- What do we need to be easy, and what is allowed to be hard?
- May be necessary to assemble a group to discuss this in more detail

Asset Store options for consideration:

1. Metadata improvements without change to large scale storage
2. Mods to DB first
    a. Asset store updates in db transaction
    b. Async reliable updates to asset store
    c. Decouple asset store and resolve conflict
3. Mods to Asset store first
    a. Asset store updates in DSpace transactions
    b. Async reliable updates to asset store
    c. Decouple db and resolve conflict.

Action list:

- Check Style implementation (JD)
- Add on mechanism at build time (JD, RJ)
- DSpace in a Box test (JD - Debian, RJ - OS X, SP - Solaris)
- Get as much documentation online, via Margret Walkers (RR)
- Move docs CVS module to dspace module (SY)
- Split documentation into manageable chunks in a basic framework (RJ)
- Policy documentation for patches (RT, SY)
- Put the requirements of the assetstore out to the list for discussion (RT); aim to make a decision by 5th - 12th August.

Release Coordinators:

- 1.4: RR as RC, RT as backup
- 1.5: TBC (possibly SP as RC)

Download this information in a PDF: committer-meeting-05-07-2005.pdf]