DSpace Release 1.5.0 Terminology

<?xml version="1.0" encoding="utf-8"?>
<html>

Potential 1.5 Documentation terminology:

There has been some disagreement between the "placeholder" terminology we should use for the 1.5 release. In particular, there's disagreement over what the placeholders *dspace* and *dspace-source* should refer to.

Placeholders in 1.4.x

The documentation for 1.4.x included the following defined "placeholders":

- *dspace* - This was defined as the installation location of DSpace
- *dspace-source* - This was defined as the initial DSpace Source directory, or the location of the unzipped Tarball downloaded from SourceForge.

Placeholders in 1.5

There's two basic proposals (so far) for how we refer to DSpace locations via placeholders:

Proposal 1: Terminology based on 1.4.x (and previous)

In this proposal, we should attempt to keep our terms as similar as possible to the terminology used for 1.4.2 (and previous)

- *dspace* - Would still refer to the installation location of DSpace
- *dspace-source* - Would still refer to the location of the unzipped Tarball downloaded from SourceForge. For those folks downloading via SVN, this would correspond to the location of the *dspace* module (which builds DSpace from all other source modules).
  ***dspace-parent* (or similar) - This would refer to the "parent" directory, which is only available if you checkout directly from SVN. This "parent" directory contains*all** DSpace modules (i.e. *dspace*, *dspace-api*, *dspace-jspui*, *dspace-xmlui*).

Pros:

- Keeps the terminology the same as in 1.4.2 (and previous)
  Cons:
- Because of the architecture changes, these terms may not be as appropriate as in 1.4.2. For example, the *dspace-source* placeholder would refer to the *dspace* module, which actually doesn't contain any Java source code.

Proposal 2: New Terminology for 1.5

In this proposal, we should change our terminology in 1.5 to be more appropriate for the 1.5 architecture changes

- *dspace* or *dspace-bootstrap* (or similar) - Would refer to the location of the *dspace* module (which builds DSpace). This also corresponds to the location of the unzipped Tarball downloaded from SourceForge.
  ***dspace-source* - Would refer to the "parent" directory, which is only available if you checkout directly from SVN. This directory contains the source of*all** DSpace modules (i.e. *dspace*, *dspace-api*, *dspace-jspui*, *dspace-xmlui*).
- *dspace-install* - Would refer to the DSpace installation location (i.e. the place where DSpace is installed to after running *ant fresh_install*)

Pros:

- Terminology may be more appropriate for new architecture. The placeholders are more descriptive of their meanings.
  Cons:
- This is a complete change from the terminology/placeholders used in 1.4.2 documentation. We'd need to document that *dspace-source* refers to something different in 1.5, than it did in 1.4.2. The same with *dspace* placeholder (unless we do away with this placeholder altogether).

Potential 1.5 Build/Distribution approaches:

(1) Easy, pre-built overlay modules: (APPROVED)

Mark Diggory did a prototype of this build architecture: http://dspace-sandbox.googlecode.com/svn/prototypes/binary-build-prototype-2/

Note: This overlay architecture has also been committed to 1.5.x branch

Folder Structure

- dspace/
  - bin/
  - config/
  - etc/
  - modules/
    - jspui/ (*easy overlay, pom.xml already exists*)

        - src/
            - main/
                - webapp/
        - pom.xml
    - xmlui/ (*easy overlay, pom.xml already exists*)
        - pom.xml

Steps to Build DSpace:

- Checkout the primary 'dspace' module (this can also be distributed as a Tarball via SourceForge): http://dspace-sandbox.googlecode.com/svn/prototypes/binary-build-prototype-2/dspace/
- Create WARs

```
mvn package
```

  . This creates the WARs in *dspace/target/dspace-1.5-SNAPSHOT-build.dir/webapps*
- Install via Ant:

```
cd dspace/target/dspace-1.5-SNAPSHOT-build.dir;  ant update
```

Customizing JSPs for JSPUI: (Verified this works - TD)

- Place any custom JSPs in *dspace/modules/jspui/src/main/webapp*
- Rebuild+DSpace (see above)

Customizing Messages.properties for JSPUI: (Verified this works - TD)

- Place a custom Messages.properties file in *dspace/modules/jspui/src/main/resources*
- Rebuild+DSpace (see above)

Advantages:

- no knowledge of pom.xml necessary to overlay interface (but need good documentation obviously)
- everything is self-contained under 'dspace' folder (could be distributed as a tarball in this fashion)
- Replacement for *jsp/local*. Now non-technical folks can put their custom JSPs in *modules/dspace-jspui/src/main/webapp/* to ensure they are "picked up" by DSpace.

Disadvantages:

- Not as easy for developers to develop separate custom DSpace modules??

Questions:

1. Could we make two versions of a tarball distribution here? One that would require internet access to build (a 'skinny version'), and potentially one that comes with all of DSpace already (a 'full version') and doesn't require you to be on the internet to build.

(2) Advanced, Maven-based build & install: (DECIDED AGAINST)

Folder Structure

- dspace/
    - bin/
    - config/
    - etc/
    - modules/ <-- *Not used, or potentially only used for 3rd party modules?*
    - pom.xml <-- *modified to load in below custom modules*
- myapp-jspui-overlay/
    - src/
    - pom.xml <--*requires creating your own pom.xml*
- myapp-xmlui-overlay/
    - src/
    - pom.xml
- myapp-api/
    - src/
    - pom.xml

Advantages:

- easier to separately manage your various overlays (or custom modules/APIs)

Disadvantages:

- requires building your own pom.xml and more understanding of Maven
- likely no tarball distribution here...this is more for developers, so it'd all be pulled down via SVN.

</html>