

# **DSpace 2.0 and 1.5.2 XMLUI Enhancements to Modularity**

## ***Abstract***

DSpace 2.0 will support modular capabilities at multiple levels of its architecture. While the DSpace Kernel will provide a means to share Services and Configuration detail across multiple deployed web applications within a Servlet Container, the DSpace XMLUI has been ported to Cocoon 2.2, which includes support for the dynamic deployment of “Service Blocks”. Blocks enhance the DSpace XMLUI web-application to support not just the Manakin Aspects deployed at assembly time, but also additional services (or servlets) that may be accessed and replaced polymorphically.

The lessons learned from the migration of DSpace 2.0 to use Cocoon 2.2 have been back-ported to DSpace 1.5.2. By migrating DSpace 1.5.2 to Cocoon 2.2 we are now able to utilize the same Block support to allow DSpace Manakin XMLUI users to more easily add their modules into the Core. We will discuss the Cocoon 2.2 Block capability and review an example Block that will introduce new Services, Manakin Aspects and Theme additions into the DSpace XMLUI web application, showing how it assists in keeping customizations cleanly separated and manageable by the developer.

## ***Introduction:***

Cocoon has evolved since the Digital Initiatives group at Texas A&M University created DSpace Manakin. The latest stable version is Cocoon 2.2, which replaces much of its underlying configuration and implementation originally done in Apache Avalon with the Spring Framework. While not intrinsically critical for DSpace 2.0 development, it became evident that not only DSpace 2.0 and the Manakin XMLUI would benefit from migration to this new version of Cocoon, but that the same changes can be back-ported to DSpace 1.5.2 to provide solutions to the problem of modularity for the next production release of DSpace.

## ***Modular Services***

Once it became used and customized within the community, one of the most significant points of difficulty with Manakin was the lack of modularity and separability in overriding any non-DRI cocoon pipelines occurring within Manakin’s default Cocoon sitemap. Once a developer needed to insert their own content pipelines that might transform anything other than the DRI and METS provided by Manakin for rendering, they were required to retrieve the default sitemap from the dspace-xmlui-webapp and begin overriding its contents. While a developer may easily solve the dilemma by performing this surgery and placing it into the DSpace XMLUI modules overlay, the process still repeats the continued mistake of taking original DSpace source and superimposing ones changes upon it as a means of customization. Cocoon 2.2 Blocks provide a means to inject configuration of new pipelines into Manakin without the direct alteration of either the Manakin dspace-xmlui-webapp or any of its supporting libraries. We will review the details of what Cocoon Blocks are and how you can use them in DSpace 1.5.2 and 2.0

There is an added benefit with Cocoon 2.2 that the Apache Cocoon community provides Maven “Archetypes” used to rapidly generate Cocoon 2.2 Blocks for your DSpace application, we will walk the user through the creation of a Cocoon 2.2 Block within the Eclipse IDE and Maven as an exercise in inform the developer on how to complete this process and create their own DSpace Manakin Addons.

### ***Conclusions:***

Lessons learned from the migration of DSpace 2.0 to use Cocoon 2.2 have been backported to DSpace 1.5.2. By migrating DSpace 1.5.2 to Cocoon 2.2 we are now able to utilize the same Block support to allow DSpace users to more easily add their modules into the Core. As Cocoon 2.2 is backward compatible with Cocoon 2.1, DSpace will continue to support the same Aspect and Theme capabilities found in 1.5.1 and there should be little work in migrating ones own themes and aspects to the new platform.