DSpace XML UI Project Technical Overview

Scott Phillips, Alexey Maslov, John Leggett
Digital Initiatives, Research and Technology
Texas A&M Libraries
{sphillip, amaslov, jleggett}@lib-gw.tamu.edu

Adam Mikeal
Computer Science Department
Texas A&M University
adam@cs.tamu.edu

ABSTRACT

This paper describes the modifications to DSpace by Texas A&M Libraries to support an XML-based user interface. DSpace supports digital repositories composed of communities and collections. Each community within DSpace typically represents an organizational unit within an institution. To increase the appeal of DSpace as a digital repository to these communities, this project enables the establishment of a unique look and feel that might extend outside of DSpace into an existing web presence. We believe this may increase the adoption of DSpace by these communities.

INTRODUCTION

DSpace is an open source digital repository system used by several institutions. The DSpace Information Model [1] is similar to the organizational structure of a university composed of colleges, departments, schools, labs, centers, etc. These organizational units may be mapped to communities and collections within DSpace. When DSpace is used as the centralized Institutional Repository, it is beneficial for individual communities to present their own distinct look and feel, which might integrate with an existing web presence in the organization.

There are four design goals for this XML UI project. First, allowing each community and collection represented in DSpace to maintain a distinct look and feel. Second, to increase support for internationalization in DSpace. Third, to separate the business logic from stylistic controls, increasing ease of adaptability. Finally, to provide an alternative interface to the current JSP—based implementation, requiring no changes to the core of DSpace (including the database), while specifically enabling both user interfaces to operate simultaneously.

ARCHITECTURE

The XML user interface relies on Java Servlets, an XML Manager, many XML Objects and a Theme Manager which operate together to deliver a web page to the end user in a variety of unique styles.

In this project, there are six stages to process an HTTP request sent by a user's browser:

1. In response to the client's request, an HTTP Request object is generated by the servlet container.
2. This Request object is passed to the appropriate servlet, which determines flow control and handles posted data. That servlet identifies and creates a specific XML Object representing the page to be rendered.
3. The identified XML Object is given to the XML Manager. The manager constructs the DOM and inserts metadata and site-wide navigational links.
4. The DOM is given to the identified XML Object where the main content of the page is inserted into the DOM.
5. After the XML Object is finished, the Theme Manager determines the correct theme to apply based upon thematic configuration. The Theme Manager consults a configuration file, typically themes.xml, which describes the themes that are to be applied to a specific community or collection. If no specific theme exists for the community or collection the parent community is checked for a theme and so on until a theme is found. If no theme is found then the site wide default theme is applied.
6. Finally the XML Manager serializes the DOM to a stream (usually XHTML) for delivery by the HTTP Response Object.

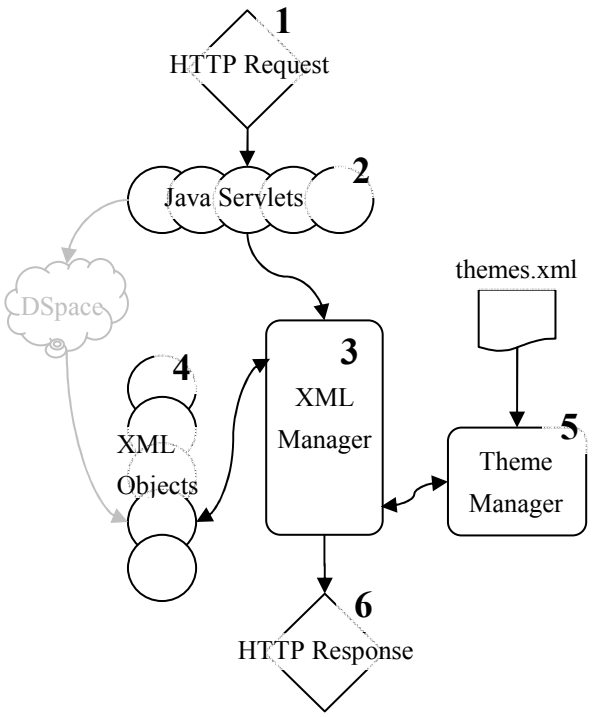

Figure 1: Six stages of an HTTP request

Architectural Comparison

The architecture used in this project is distinct from the current DSpace implementation and a possible Cocoon-based architecture. The flow of data formats for these three architectures can be expressed as:

Current DSpace architecture:
JSP → HTML

Cocoon-based architecture:
JSP → XML → XSL Transformations → XHTML

Our architecture:
Java → DOM → XSL Transformations → XHTML

Both our architecture and a Cocoon-based architecture offer advantages over the current JSP-based implementation by separating style controls from the business logic of DSpace. In our architecture the use of a DOM allows XML Objects to change any part of the DOM until it is finally serialized for delivery to the user's browser. This feature is used to modify or expand previously generated portions of the page at any time during page generation. With a Cocoon-based architecture this flexibility is not present because the XML is created in stream form and SAX events are generated immediately which can not be revoked or modified.

XML SCHEMA

The XML schema used by this project is a specialized schema for DSpace based upon the experience of past projects for the Computer Science Department at Texas A&M. Several other schemas were evaluated for decoupling style from semantic information in web-oriented applications. Existing schemas such as DocBook [2] provide methods of encoding most aspects of a web page, but are lacking elements necessary for HTML form representation. Therefore, our solution follows DocBook design patterns when applicable, but deviates when necessary for web-based semantics.

The schema used in this project involves three major elements: `meta`, `body`, and `options`. The `meta` element contains meta information about the request and page. Examples include the user agent (browser), the user's preferred language, the page title, and the page's location within the DSpace structure. The `body` element contains the meat of the information to be rendered in the browser, consisting of `tables`, `sections`, and `forms`. The `options` element exposes the standard navigational links from the current page, as well as available actions. Examples include the standard "Browse by" links, login actions, access to the user's profile, searching, and context-sensitive links that are dependent upon the user's state and the current page.

Schema Internationalization

The schema supports internationalization through the use of the `lang` attribute. This attribute is attached to *all* elements which encode textual content, excluding user-supplied data such as usernames, community and collection names, and collection meta-data. The DSpace API doesn't currently support storing this information in multiple languages, so for our project, this data was considered to be language-independent. The current theme uses the `lang` attribute along with the specified language of the user contained in the `meta` element to separate out the specific languages for rendering.

There are two main reasons for handling internationalization at the theme stage. First, themes are expected to generate content for site and community-specific purposes (such as site-specific instructions for authentication, etc.). This content will also require language translation, and the theme must be aware of the language being rendered in order to change behavior accordingly. Second, this design allows for multiple language interfaces. Multiple language interfaces may be useful in cases where the language of the collection's content differs from the user's primary language. For example, a digital library of 16th century Spanish manuscripts could create a theme that provided both Spanish and alternate languages in the same interface.

THEMES

Themes are XSL transformations that are applied to render the XML schema into an HTML page for display on the user's browser. They may consist of the following: XSL files, images, JavaScript, and CSS rules.

Many of the stylistic differences between styles may be achieved solely through the use of CSS rules. However, the use of XSL enables for more complex styles to be used. Such possibilities include a text-only version for accessibility concerns, a HTML-3.2 compliant version for pre-CSS browsers, WML for WAP-enabled devices, multilingual versions, and dynamically-generated JavaScript and PDFs.

FUTURE WORK

While this project is currently under development many opportunities exist for future development. We hope to allow user selected themes for specialized user tasks. It may be beneficial for the interface to be able to highlight important information while hiding irrelevant information tailored to the specific users needs. Another area for future work is a library of default themes for distribution with the DSpace XML UI. This library could be used by DSpace out of the box and also provide a starting point for institution's local theme development. Further work could also be done on extending internationalization into the DSpace API, supporting language-based metadata on collections and communities.

CONCLUSION

The XML UI project is currently under development at Texas A&M University Library's Digital Initiatives, Research and Technology (DIRT) group. The project promises to expand the current functionality of DSpace to allow for unique interfaces based upon the style choices of individual communities and collections. This functionality will benefit these communities, allowing them to integrate their collections with a pre-existing web presence outside of DSpace. By increasing the appeal of DSpace to those communities, we may increase the adoption of DSpace as a digital repository solution.

REFERENCES

1. MacKenzie Smith, "DSpace: An Institutional Repository from the MIT Libraries and Hewlett Packard Laboratories"', *Lecture Notes in Computer Science*, 2458, September 2002.
2. OASIS DocBook Technical Committee, DocBook v4.1 [OASIS 200101], http://www.oasis-open.org/specs/, January 2001

Visit our demonstration site at:

http://di.tamu.edu/dspace-xmlui/