Introducing Piper

A repository-agnostic batch deposit tool

Micah Cooper, James Creel, Doug Hahn, Jeremy Huff, Bruce Herbert, Lilly Li, Alexey Maslov, and Sarah Potvin
Presentation Outline

- Introduction and Motivations
- Developing Piper
- Feature Demonstration
- Retrospective and Ambitions
- Discussion
Collection owner creates structured metadata and content

Programmer converts structured metadata and content into DSpace Simple Archive Format (SAF)

Import is conducted on a test server for collection owner’s review

Import is conducted to live repository
Problems with Classic DSpace Workflow

- The “structure” of the metadata and content provided by the collection owner is ill defined.
- Thus, each batch deposit is a unique data processing job.
 How do metadata records correspond to content?
 How are content files structured? Are they explicit or implicitly specified in the metadata or elsewhere?
 What conditions are required at the batch level (as opposed to item level)
  ▪ Licenses
  ▪ Permissions/Embargos
 How do metadata records map to Dublin Core fields?
Problems Arising from Variations in Structure

- Missing files
- Character encoding
- Improper Licensure/Access
- Metadata problems
  - Ill formatted field values
  - Ill segregated fields
Example:
Ill-segregated field values
Conceiving the Application

- Developer time was costly to spend on metadata and content curation
- Librarians needed more control over ingestion of content into the IR
- We wanted a tool to automate or facilitate the error checking and structure review.
- Better yet, the tool could deposit with SWORD instead of at the DSpace command line!
Developing the Application

- A Java-based Webapp suited our environment
- We practiced the Agile methodology
  - Short (i.e. 2 week) iterations of feature development
  - Customer feedback and critique after each iteration
  - Amendment of product backlog at each iteration
- Imagined workflow: Import metadata/content, validate it with a chosen (perhaps destination appropriate) validation modules, and deposit it (getting back the new entities’ URIs)
Technical Demo

- Technologies Overview
- Live Demo
- Future Development
Technology Overview

Java

Play!

Websocket

jQuery

<Sword />
Live Demo

- Ingest
- Validation
- Batch Documents
- Editing Metadata
- Deposit
- Administration/Settings
Technical Plans

• Websockets
• Additional Custom Validators
• OAI-PMH and ORE Ingest
• SWORD2
• Supporting Use Cases
• Limitations
• Supporting Campus Initiatives around Open Access
Stage One: Supporting New Workflows and Complex, but Typical, Use Cases
Stage One: Supporting New Workflows and Complex, but Typical, Use Cases
Stage One: Supporting New Workflows and Complex, but Typical, Use Cases

Identifying errors earlier in the process
Limitations:
Continued Need for Expert Intervention

OAK Trust image gallery theme
Stage Two: Serving Broader Campus Initiatives around Open Access
Digital Initiatives, Texas A&M University Libraries
Micah Cooper
James Creel
Doug Hahn
Jeremy Huff
Alexey Maslov

Office of Scholarly Communication, Texas A&M University Libraries
Bruce Herbert
Yu “Lilly” Li
Sarah Potvin

Many thanks to those in DI and OSC who contributed feedback that helped initiate and guide Piper’s development. Special thanks to Scott Phillips.