Deliver your EAD – maybe without XSLT, or XML, or ...?

A work in progress
Terry Catapano & Joanna DiPasquale
The problems

• Archivists want EAD converted into web-ready finding aids very quickly
• EAD a permissive schema
• Some finding aids are extremely long
• Different repositories have different descriptive practices and display requirements
• We wanted a dynamic system for a variety of reasons
• We wanted to work with the tools we had
The solutions

• eXist used as document repository
• Seeded daily with collection level EAD derived from OPAC records
• Data source in oXygen editor
• Documents valid against local customizations of EAD (using NVDL and Schematron)
XQuery

• Triggers chmod to protect working drafts from overwriting, move final drafts to public eXist collection
• XQuery to serve up fragments of finding aids at once
• “REST-ful” API so that we can get XQuery results from anywhere on the web
Example

• Get “summary” (collection-level DID)
  – URI = http://ldpd.lampdev.columbia.edu/fa/ead/nnc-rb/ldpd_6259383/summary

Section is predefined snippet of the EAD document (doesn’t have to be EAD, or XML at all – just getting back data needed / requested):
Looking at producing XML, JSON, RDF, ATOM, RSS, etc...
Parsing the data

- We have a LAMP environment, so we use XMLReader library from PHP: http://us3.php.net/manual/en/book.xmlreader.php
- Reads the document similar to a "streaming" file
  - Very fast
  - Can work with extremely large documents
$reader = new XMLReader();
$reader->open($file);
while ($reader->read()) {
    if ($reader->name && $reader->name == "#comment") {
        // skip the comments
    }
    if ($reader->nodeType == XMLReader::ELEMENT) {
        // start a tag
    }
    if ($reader->nodeType == XMLReader::END_ELEMENT) {
        // end a tag
    }
    if ($reader->hasValue() && $reader->value != "") {
        // write out value
    }
}
Result

• We can use EAD how we need to (as XML, or JSON, RSS, or whatever else); we’re just parsing a structured document.
• We can parse large documents very quickly
• Archivists can change finding aids in real time
• We have URIs and an API that is useful to us (and maybe to you...?)