

# A Format-Registry-Based Automated Workflow for the Ingest and Preservation of Electronic Journals

Evan Owens, Chief Technology Officer

Suku Sukumar, Architect and Technical Lead, Systems & Applications

John Meyer, Technical Lead, Data Programming

David Copperman, Systems Analyst

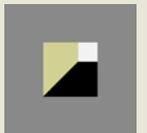
Roland Mesde, Senior Programmer

Digital Library Federation Fall Forum 2005

Charlottesville, Virginia

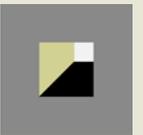
[evan.owens@portico.org](mailto:evan.owens@portico.org)

[www.portico.org](http://www.portico.org)



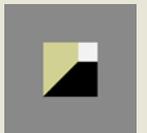
## This Presentation

- Our current project
- E-journal ingest workflow
- Format and tools registry implementation
- Some interesting issues concerning formats and format registries



## Portico: Business Summary

- A long-term preservation archive
  - [www.portico.org](http://www.portico.org)
- Initial funding by Andrew W. Mellon Foundation, JSTOR, Ithaka, and Library of Congress NDIIPP (starting in 2006)
- Goal is to be a trusted third party archive for electronic journals
  - Operational in 2006; publishers committed
- Source file archiving
  - Not web renditions per se
  - SGML/XML, graphics, page renditions, etc.
  - Normalize to standard XML DTD for long-term maintenance
  - HTML as last resort
- Get content into system
  - As cost-effectively as possible
  - Minimal intervention
  - “Archive” not “aggregate” or “re-publish”



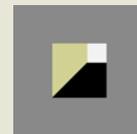
## Portico: Technology Summary

- Planning began in early 2003
- Key technical influences:
  - GDFR, PreMIS, METS, MPEG-21, ARK, OAIS
- Key technologies:
  - Service-oriented architecture
  - XML, XML schema, Schematron, JHOVE, NOID
  - Documentum, Oracle, Java, JMS, LDAP
- Design goals:
  - Pluggable tools to facilitate new providers and replacement tools
  - Clean separation of process view and structural view of content model
  - Configurable workflows for different content types
- Building a system that can manage non-trivial intervention in the content prior to archiving and preserve the record of the source data, the normalized data, and everything that happened during the normalization is a big step toward managing future migrations!

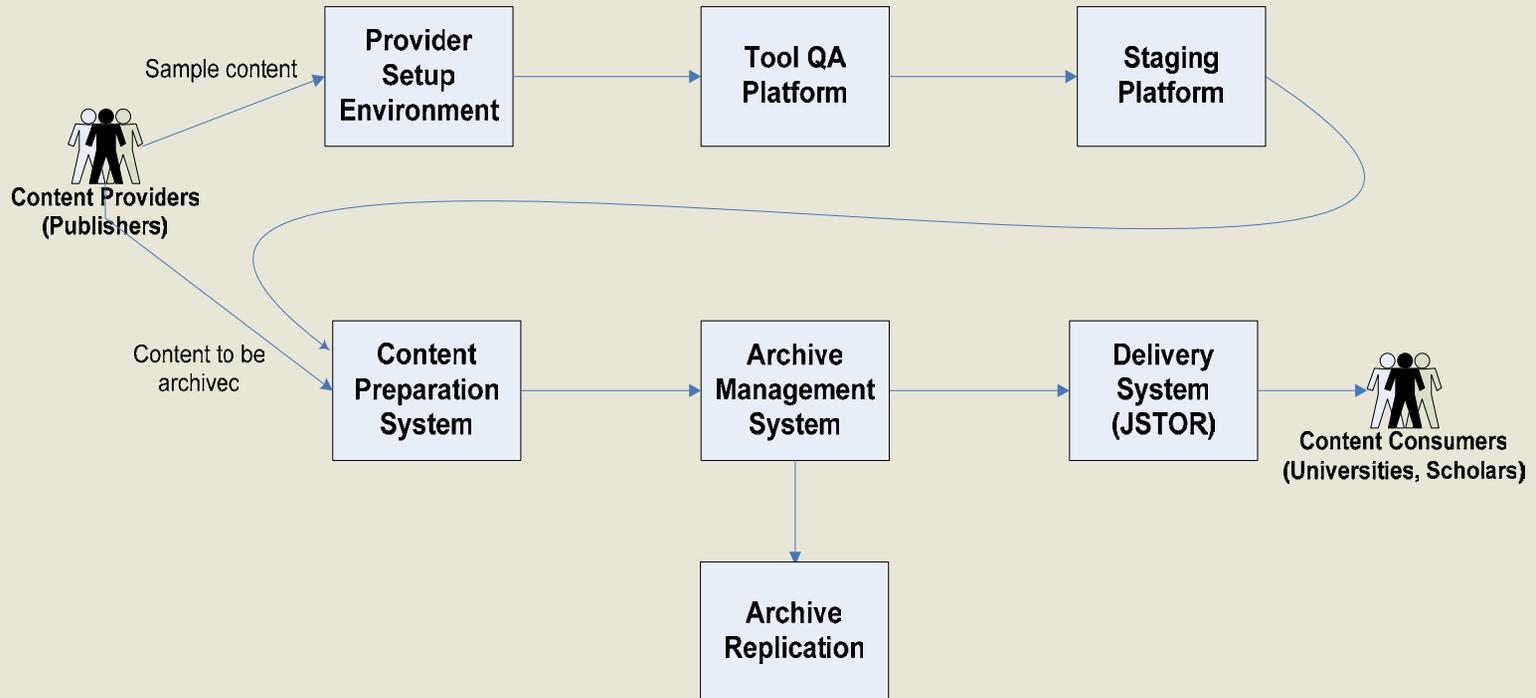


## Electronic Journal Data Issues

- **Inputs**
  - Per article: one text or metadata file, zero or more other files
  - Arbitrary (publisher-specific) collections of data
    - Proprietary file & directory naming conventions
    - Proprietary formats
  - Undocumented business rules hidden in the data
- **Outputs**
  - Normalized content
  - Metadata: technical, descriptive, events
  - Packaged in Portico METS
- **Workflow goals**
  - Taking apart and reassembling the submission package
  - Managing the normalization of proprietary formats
  - Validating formats
  - Extracting and collecting metadata
  - Assigning preservation levels based on policies
  - Match content with contracts (agreements)



## Process Overview

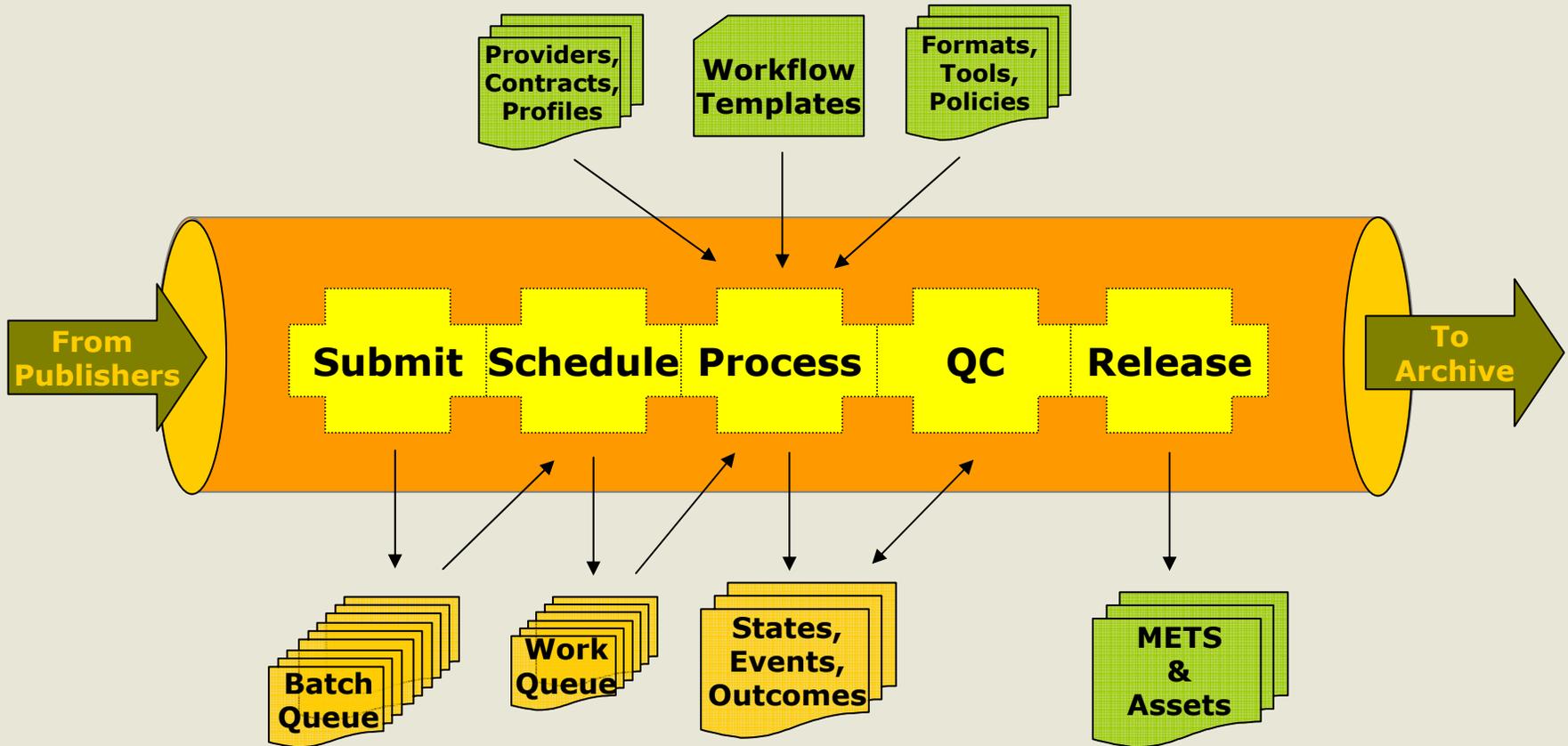


## System Components

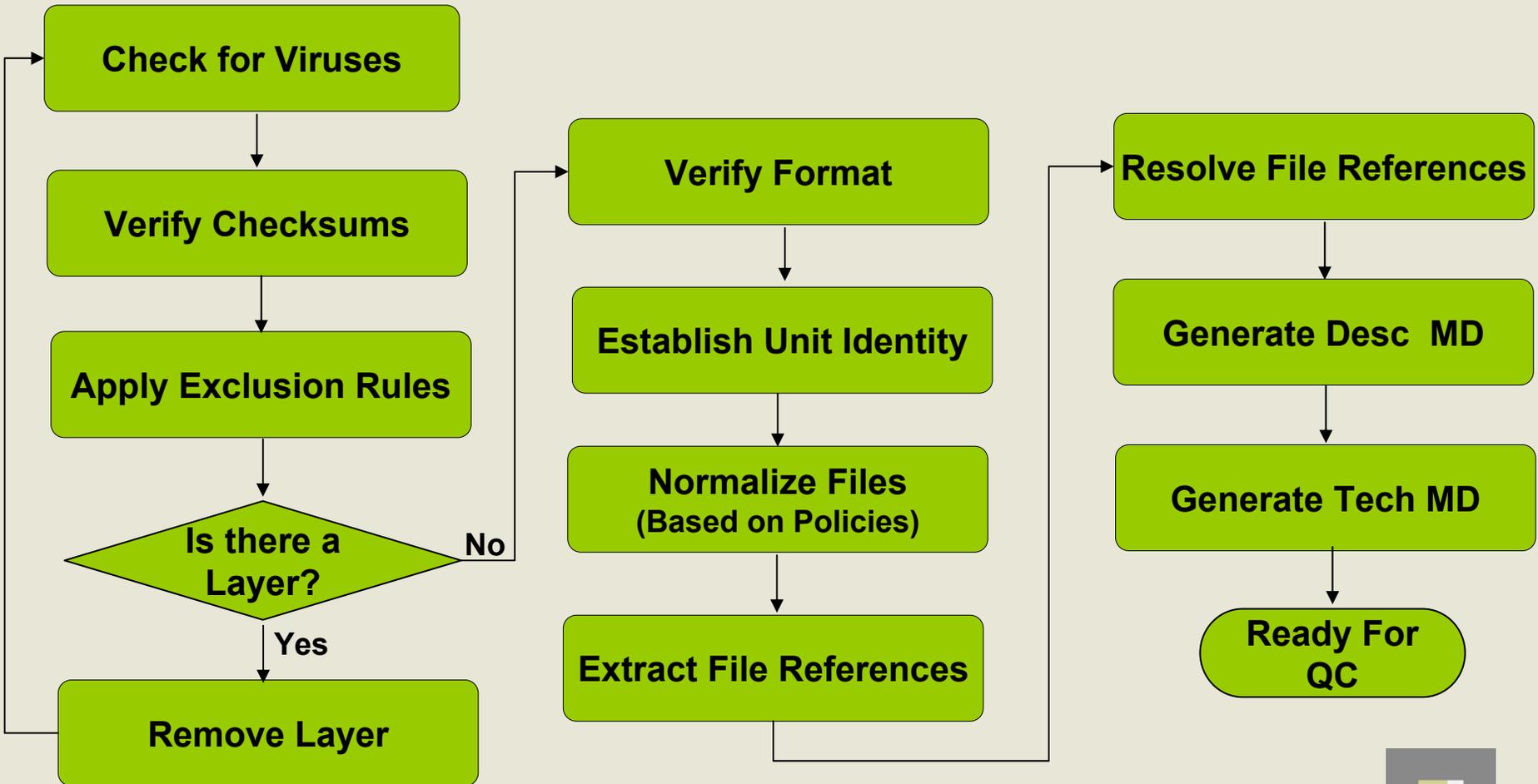
- Workflow
  - Per content type (E-Journals, Business artifacts, Technical artifacts)
  - New and updated content
- Profiles (per provider)
  - Provider-specific rules and policies
  - Packaging rules
  - File name extract rules
- Format registry
  - List of formats known to the archive
  - Links to policy documents, technical documentation, and “required files”
- Preservation policy registry
  - What promises can the archive make for a given format?
- Tools registry & Tools service
  - What tools for which formats?
  - Where are they located?
  - How are they invoked?



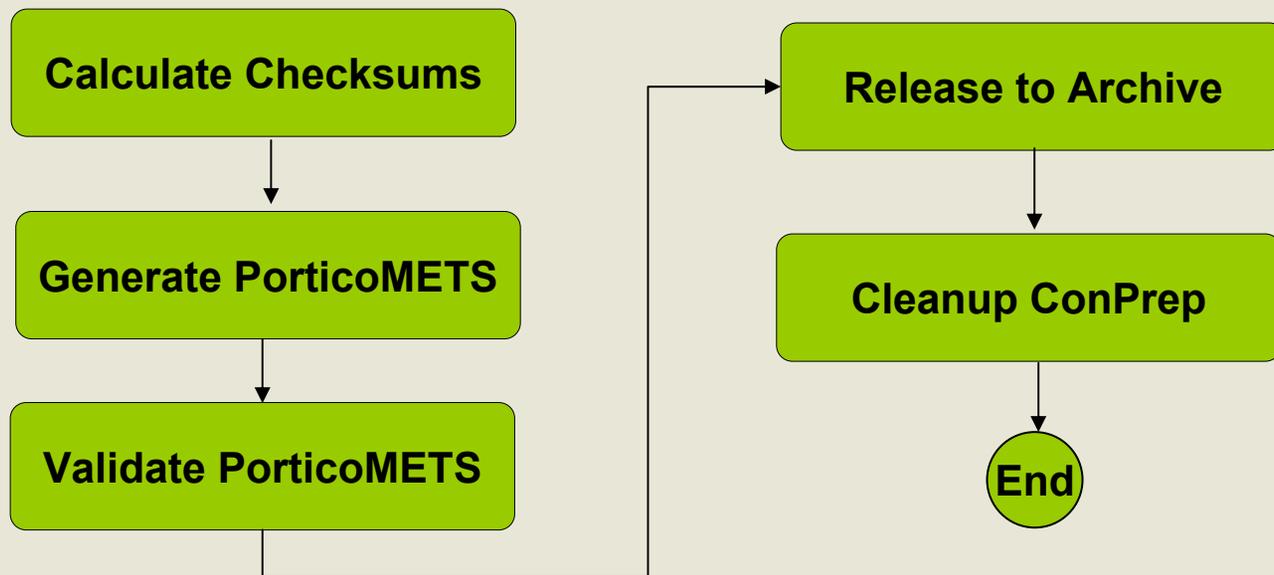
## Process View



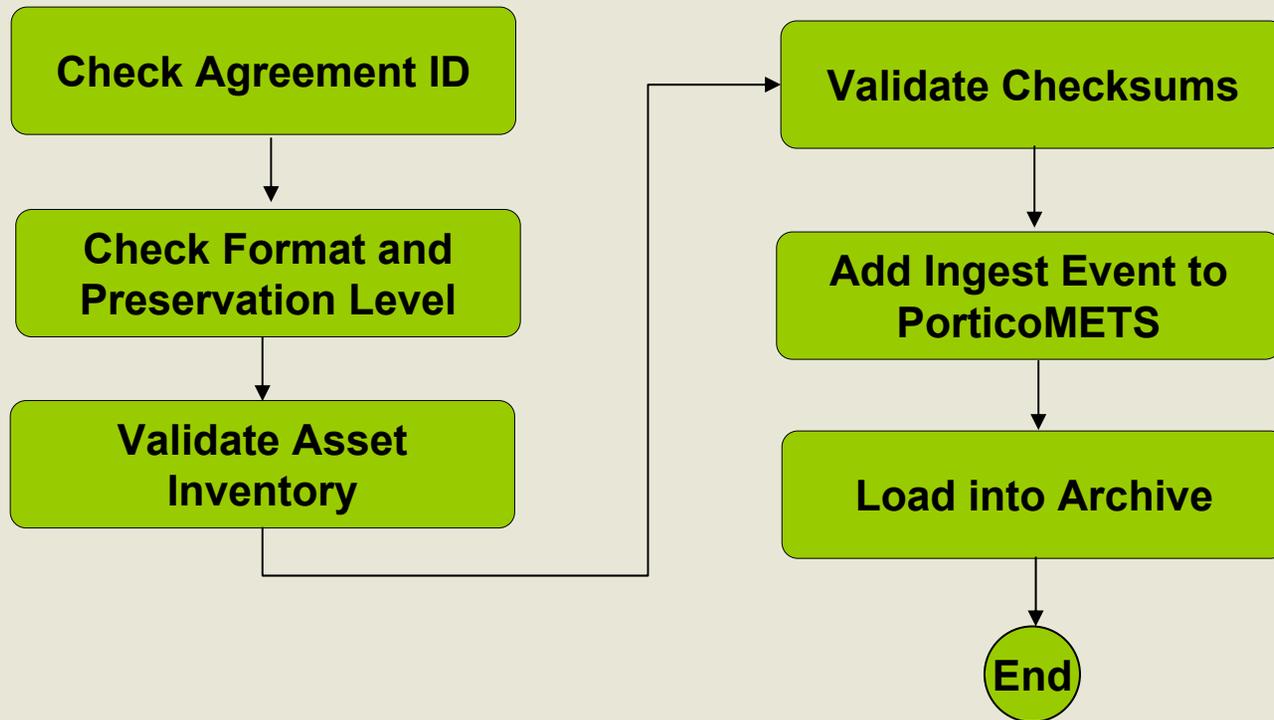
## Automated Processing for E-Journal Content (high-level summary)



## Automated Processing after QC (for all content types)



## Archive Ingest Processing



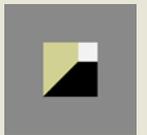
## The GDFR Context

- Global Digital Format Registry meetings in 2002, 2003
  - [hul.harvard.edu/gdfr/](http://hul.harvard.edu/gdfr/)
- Use cases from Stephen Abrams:
  - Identification
    - “I have an object; what format is it?”
  - Validation
    - “I have an object purportedly of format  $F$ ; is it?”
  - Characterization
    - “I have an object of format  $F$ ; what are its salient properties?”
  - Assessment
    - “I have an object of format  $F$ ; is it at risk of obsolescence?”
  - Processing
    - “I have an object of format  $F$ ; how can I perform operation  $X$  on it?”
  - (*The Role of Format Registries in Digital Preservation*, 2004)
- GDFR still in the future
  - We built assuming that it would be there someday soon



## Portico Format Registry Implementation

- Light-weight; we expect to redesign after GDFR becomes a reality
- Information per format:
  - Portico unique name
  - Description
  - Owner
  - Maintainer
  - Default Mime Type
  - Default File Extension
  - Category (for our own reporting)
  - Preservation strategy set
    - List of preservation planning documents
  - Required File set
    - Lists of required files stored in archive
  - Registered name set
    - Lists of external identifiers
- A flat list, not hierarchic; a simplifying assumption for v1.0



## Portico Tools Services

- Format-neutral services:
  - Virus check (ClamAV)
  - Checksum (various)
  - Identification (JHOVE, BSD file; returns a format ID and/or MIME type)
- Format- or MIME type-specific services:
  - Validation (JHOVE)
  - Characterization (JHOVE)
  - Layer removal (e.g., unzip)
  - Transformation (XSLT; per source format and destination format)
- DTD-Specific XML services:
  - Descriptive metadata extraction (XSLT)
  - HTML rendition (XSLT)
  - Descriptive metadata curation (java & XSLT)
  - File reference extraction (XSLT)
  - File reference replacement (XSLT)
  - QC errors & warnings (Schematron)
- And more to come

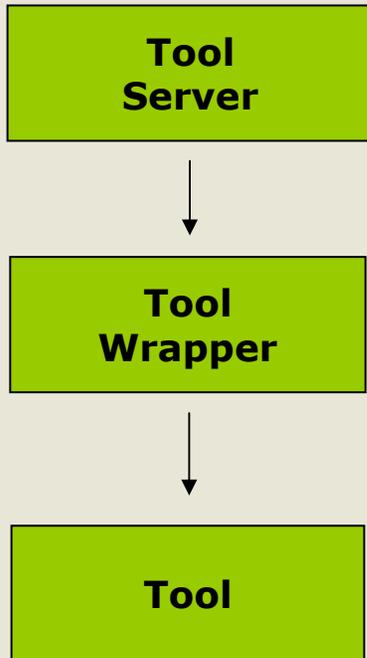


## Tool Registry & Services Implementation

- Registry provides information about tools utilized to process content
- Registry does not know whereabouts of tools or itself offer services
- Supports invocation strategies – collective, conditional, and selective
- Loose coupling of tool and format registries to facilitate independent evolution



## Tool Services



- Dispatcher that listens for requests; upon arrival, spawns a worker thread to process

- Adapter that hides tool-specific behavior and converts tool-specific interface to tool-neutral interface

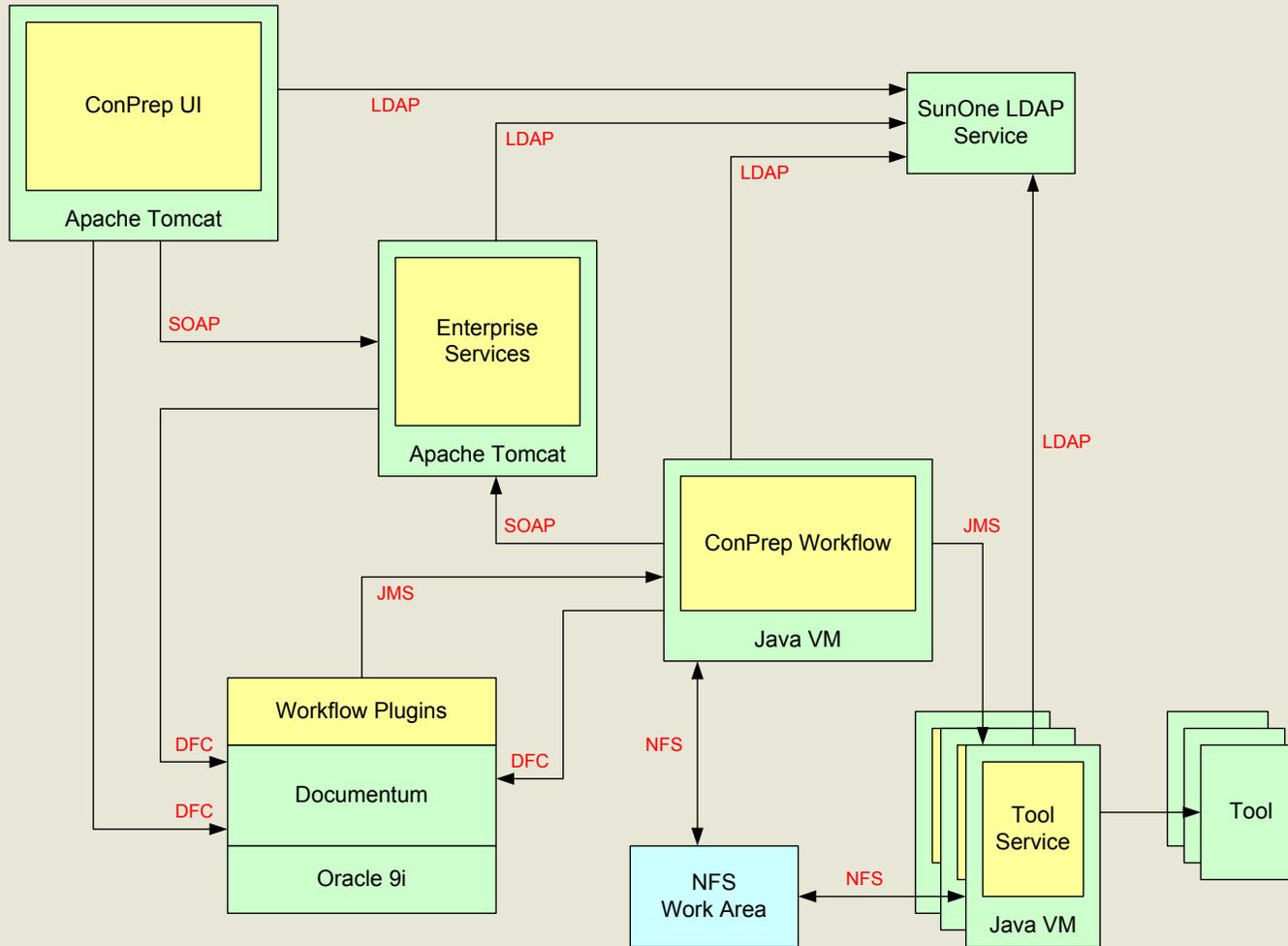
  - e.g., maps specific return values to standard values

- A COTS product, open-source, or custom software that provides a specific service

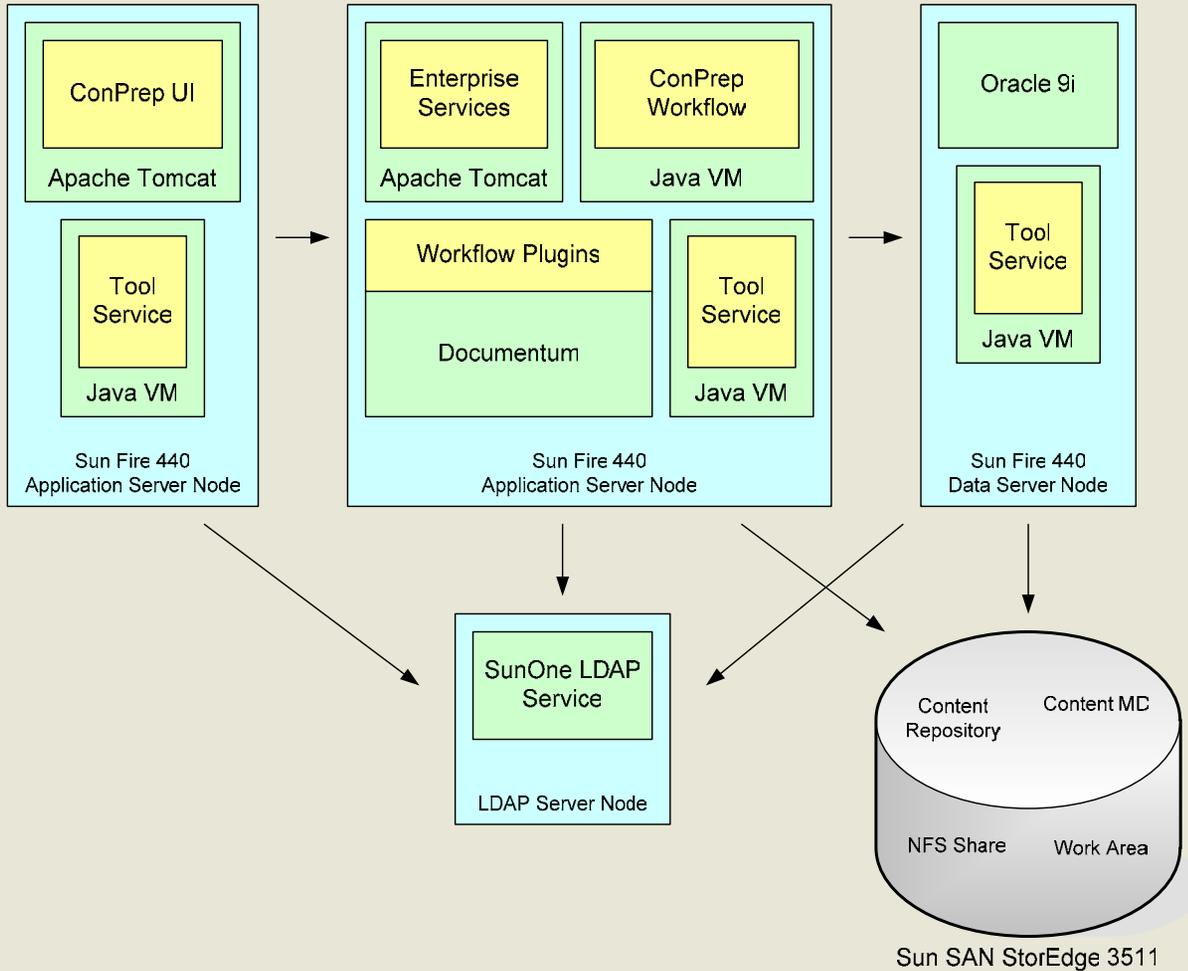
  - e.g., JHOVE, ClamAV, gzip



## Component View



## Deployment View



## Some Interesting Implementation Issues

- What granularity?
  - Every DTD version a separate format
    - Helps with version control
    - Helps make transforms into format-based services
- What about system formats?
  - Did not include system schemas unless used in archival content
  - XML schemas used in system not included
- What about format hierarchy and relationships?
  - Not in version 1.0
  - DTD XYZ => XML => ASCII not helpful
  - PDF 1.0 <=> 1.2 <=> 1.3 maybe in the future
- Do we need all that technical metadata?
  - We trim the output of JHOVE
  - Sometimes a synoptic statement is more valuable than the details:
    - Are all fonts embedded (yes/no) rather than a list of embedded fonts
  - We ignore embedded XMP metadata...at least for now



## A Major Issue: Varying Degrees of Badness

- “Repositories need to ensure that...digital object content streams are valid with respect to their formats” (Abrams, 2004)
- What format is a defective file?
  - The purported format? The actual format?
  - Format “Re-identified” (a business concern as well as technical)
- Can a file be damaged but still usable?
  - XML: No, we have to have valid XML file to extract metadata!
  - PDF: Yes, Acrobat reader can read some WFNV or NWF PDF?
- On what do you base the preservation policy for a bad file?
  - The actual format?
  - Best-effort on purported format?
  - What about well-formed but not valid?
- Some use cases:
  - Defective file (varying degrees)
  - Purported format is in error (e.g. wrong extension)
  - Both of the above



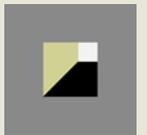
## Bad File and Mislabeled File Use Cases

Expected MIME type or Format	Verified Format	Verified Format Status	Identified Format	Identified Format Status	Format in METS	Format Status in METS	Re-Identified Flag	Preservation Level
PDF	PDF 1.4	WFV			PDF 1.4	WFV		FULL
PDF	PDF 1.4	WFNV			PDF 1.4	WFNV		BYTE-PRESERVE
PDF		NWF	PDF 1.4		BYTESTREAM	WFV	Yes	BYTE-PRESERVE
PDF		NWF	TIFF 6.0	WFV	TIFF 6.0	WFV	Yes	FULL
PDF		NWF	TIFF 6.0	NWF	BYTESTREAM	WFV	Yes	BYTE-PRESERVE
PDF		NWF	BYTESTREAM	WFV	BYTESTREAM	WFV	Yes	BYTE-PRESERVE
TIFF	TIFF 6.0	WFV			TIFF 6.0	WFV		FULL
TIFF		NWF	TIFF 6.0		BYTESTREAM	WFV	Yes	BYTE-PRESERVE
TIFF		NWF	PDF 1.4	WFV	PDF 1.4	WFV		FULL
TIFF		NWF	PDF 1.4	WFNV	PDF 1.4	WFNV	Yes	BYTE-PRESERVE
TIFF		NWF	GIF 87	WFV	GIF 87	WFV	Yes	FULL
TIFF		NWF	GIF 87	NWF	BYTESTREAM	WFV	Yes	BYTE-PRESERVE
TIFF		NWF	BYTESTREAM	WFV	BYTESTREAM	WFV	Yes	BYTE-PRESERVE
XML w/DTD	XML 1.0	WFV			XML 1.0 w/DTD	WFV		FULL
XML no DTD	XML 1.0	WF			XML 1.0 no DTD	WF		FULL
XML w/DTD	XML 1.0	WFNV			XML 1.0 w/DTD	WFNV		BYTE-PRESERVE
XML (any)		NWF	XML 1.0	NWF	BYTESTREAM	WFV	Yes	BYTE-PRESERVE
XML (any)		NWF	UTF-8	WFV	UTF-8	WFV	Yes	BYTE-PRESERVE
XML (any)		NWF	BYTESTREAM	WFV	BYTESTREAM	WFV	Yes	BYTE-PRESERVE



## Verification / Identification Sequence

- To distinguish between bad files and mislabeled files:
- Verify purported format (MIME type)
- If verification succeeds
  - Record format
  - Capture technical metadata
- If verification fails, do identification
- If identified format is same as purported format
  - File is bad
- If identified format is not same as purported format
  - Might be mislabeled
- Verify identified format
  - If fails again, file is bad



## More Implementation Issues of interest

- MIME Type is still useful
  - Even when you have a format registry
  - To interact with the outside world
  - When you have incomplete information
- “Purported format” can be
  - Purported MIME type
    - e.g., PDF but unknown which version
  - Purported Format
    - e.g., Profile expects a specific DTD (format)
- Is a format registry
  - A database or a document?
  - How volatile? How granular?
- Problems we haven't dealt with yet
  - Embedded formats
    - E.g., LaTeX as an XML/SGML notation
  - XML instances that conform to more than one schema



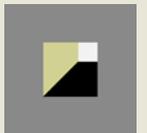
## Another Interesting Issue: Not Yet Supported Formats

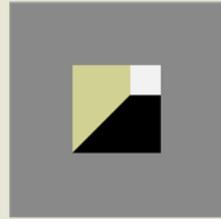
- What do we do when we don't have tools yet?
  - What preservation commitment?
  - What values for format and validity?
- Some use cases:
  - Purported MIME Type
  - Purported Format
  - Completely unknown
- Some possibilities:
  - Record MIME type in lieu of a format?
  - Create generic formats in the format registry?
    - e.g., "PDF of unknown version"
  - Allow format validity of "unknown"?
  - Preservation level of "Byte Preserve Pending"
  - Don't allow the content into the archive
    - Ideal solution!



## Some Lessons Learned

- Format registry is a powerful concept
  - We are eager for the GDFR work to take off
- MIME type is still useful
  - Somewhat to our surprise
  - A surrogate for relationships between formats?
- XML / SGML DTDs (structured markup) feel very different from graphics formats
  - Does one size fit all types of formats, as it were?
  - Well-formed
- JHOVE and the JHOVE framework work really well
  - Please contribute modules!
  - We are working on one for SGML





PORTICO