JHOVE2 Next-Generation Characterization
A Project Update

JHOVE2 Project Team
California Digital Library, Portico, Stanford University
Agenda

Introduction and concepts
Demonstration
Architecture and APIs
Assessment
Sustaining the JHOVE2 open source community
Discussion
Agenda

Introduction and concepts
Demonstration
Architecture and APIs
Assessment
Sustaining the JHOVE2 open source community
Discussion
“Tell me about yourself…”
“What? So what?”

Characterization is the automated determination of the intrinsic and extrinsic properties of a formatted object

- Identification
- Feature extraction
- Validation
- Assessment
“We report, you decide...”

© Fox News Network LLC
JHOVE2 feature set

Multi-stage processing

– Signature-based identification

✓ DROID
http://droid.sourceforge.net/

– Feature extraction

– Validation

– Message digesting

✓ Adler-32, CRC-32, MD2, MD5, SHA-1, SHA-256, SHA-384, SHA-512

– Rules-based assessment
JHOVE2 feature set

Processing of objects spanning files and objects that are subsets of files

Recursive processing of objects arbitrarily-nested within containers

Granular modularization with generic plug-ins

Clean APIs and common module design patterns

Buffered I/O

Internationalized output
Supported formats

JHOVE2 can identify (by DROID) many more formats than it can validate (by modules)

– PRONOM registry documents over 550 formats; approx. 220 with signatures  http://www.nationalarchives.gov.uk/PRONOM
Supported formats

ICC color profile (ICC.1:2004-10)
PDF PDF 1.0 – 1.7, ISO 3200-1, PDF/A-1 (ISO 19005-1), PDF/X-1 (ISO 15920-1), -1a (ISO 15930-4), -2 (ISO 15930-5) -3 (ISO 15930-6)
SGML
Shapefile Main, Index, dBASE, ...
UTF-8 ASCII (ANSI X3.4)
WAVE BWF (EBU N22-1997)
XML
Zip
Supported formats

netCDF
http://www.unidata.ucar.edu/software/netcdf

Grib

– Developed by the Wegener Institute (Germany)
http://www.awi-potsdam.de

– Widely used for meteorological data
(Un)Supported formats

- AIFF
- GIF
- HTML
- JPEG

- HTML can be expressed in terms of SGML or XML
- We’re investigating funding options for subsequent development of GIF and JPEG modules
Source units

A formatted object about which characterization information can be meaningfully reported

- **Unitary**
  - ✔ File
  - ✔ File inside of a container
  - ✔ Byte stream inside a file
    - e.g. TIFF
    - e.g. TIFF inside a Zip
    - e.g. ICC inside a TIFF

- **Aggregate**
  - ✔ Directory
  - ✔ Directory inside of a container
  - ✔ Clump
    - e.g. Shapefile
  - ✔ File set
    - e.g. command line arguments

For purposes of characterization, directories, file sets, and clumps are considered formats
Properties and reportables

A property is a named, typed value

- Name (based on the terminology of the underlying format)
- Unique formal identifier
- Data type
  - Scalar or collection
  - Java types, JHOVE2 primitive types, or JHOVE2 reportables
- Typed value
- Description of correct semantic interpretation

A reportable is a named set of properties

- Reportables correspond to Java classes
- Properties correspond to fields
Characterization strategy
Characterization strategy
Characterization strategy

- Shapefile
- clump
- abc.tif
- xyz.pdf
- abc.shp
- abc.shx
- abc.dbf
- Main
- Index
- dBASE
- GeoTIFF
- PDF
Characterization strategy

```
directory/
  clump
     abc.shp
     abc.shx
     abc.dbf
  abc.tif
  xyz.pdf
```

- Shapefile
- "GIS object"
- Main
- Index
- dBASE
- GeoTIFF
- PDF
Agenda

Introduction and concepts

Demonstration

Architecture and APIs

Assessment

Sustaining the JHOVE2 open source community

Discussion
Agenda

Introduction and concepts

Demonstration

Architecture and APIs

Assessment

Sustaining the JHOVE2 open source community

Discussion
API design idioms

Separation of concerns

– Annotation and reflection
  confluence.ucop.edu/display/JHOVE2Info/Background+Papers

Inversion of control (IOC) / dependency injection

– Martin Fowler
  martinfowler.com/articles/injection.html

– Spring framework
  www.springsource.org/
Separation of concerns

“Let POJOs be POJOs”

– Focus on modeling the format itself

“Let the code write itself”

– Reportables “know” how to expose their properties for display

– Reference documentation generated from the code

✔ JHOVE2Doc application

Reportable: Name: UTF8Module
Identifier: [JHOVE2]
http://jhove2.org/terms/reportable/org/jhove2/module/format/utf8=UTF8Module
Package: org.jhove2.module.format.utf8

From: Class UTF8Module

Property: Name: NumCharacters
Identifier: [JHOVE2]
http://jhove2.org/terms/property/org/jhove2/module/format/utf8=UTF8Module/NumCharacters
Type: long
Description: Number of UTF-8 characters
Annotation and Reflection:
Reportable properties

Each reportable property is represented by a field and accessor and mutator methods.

The accessor method must be marked with the @ReportableProperty annotation.

```java
public class MyReportable
    implements Reportable
{
    protected String myProperty;

    @ReportableProperty(order=1, desc="description", ref="reference")
    public String getMyProperty() {
        return this.myProperty;
    }

    public void setMyProperty(String property) {
        this.myProperty = property;
    }
}
```
Displayer directives

jhove2/src/main/resources/properties/displayers.properties

```
<property-identifier> http://jhove2.org/terms/property/org/jhove2/module/Agent
<directive> http://jhove2/property/.../DirectorySource/isExtant
... Never IfFalse

- Always (default)
- IfTrue
- IfNegative
- IfPositive
- IfZero
- Never
- IfFalse
- IfNonNegative
- IfNonPositive
- IfNonZero
```
Results

JSON

"Path": "C:\shapefiles"

Text

Path: C:\shapefiles

XML

<j2:feature name="Path"
    fid="http://jhove2.org/terms/property/org/jhove2/core/source/DirectorySource/Path" fidns="JHOVE2">
    <j2:value>C:\shapefiles</j2:value>
</j2:feature>

- Intended as an intermediate form suitable for stylesheet transform to any desired final form (Transform to Mets provided)
Format Modules: Reflection as Facade

- Format module “from scratch” (TIFF, UTF-8, WAV)
- Format module as façade over Java tool (XML, Shapefile)
- Format module as façade over non-Java tool (SGML)
Dependency injection

All JHOVE2 function is embodied in pluggable modules

– Flexible customization
  ✓ Re-sequencing of pre-existing modules

– Easy extensibility
  ✓ Additional format modules and profiles
  ✓ Additional aggregate identifiers
  ✓ Additional displayers
  ✓ New behaviors

RenderabilityModule
JHOVE2 framework

Embodiment of a characterization strategy as a configurable sequence of command-invoked modules

```java
public void characterize(Source source, Input input)
    throws IOException, JHOVE2Exception {
    source.getTimerInfo().setStartTime();
    /* Update summary counts of source units, by type. */
    this.sourceCounter.incrementSourceCounter(source);
    for (Command command : this.commands){
        TimerInfo time2 = command.getTimerInfo();
        time2.resetStartTime();
        try {
            command.execute(this, source, input);
        } finally {
            time2.setEndTime();
        }
    }
    source.getTimerInfo().setEndTime();
}
```
Characterization
Key Interfaces

• Reportable

• Command

• Module
  – Identifier
  – FormatModule
  – Aggrepier
  – Digester
  – Assessor
  – Displayer
Spring configuration: Identification

```xml
<!-- Identifier module bean -->
<bean id="Identifier" class="org.jhove2.module.identify.IdentifierModule"
    scope="prototype">
  <property name="developers">
    <list value-type="org.jhove2.core.Agent">
      <ref bean="CDLAgent"/>
      <ref bean="PorticoAgent"/>
      <ref bean="StanfordAgent"/>
    </list>
  </property>
  <property name="fileSourceIdentifier" ref="droidIdentifier"/>
</bean>

<!-- DROID identifier bean -->
<bean id="droidIdentifier" class="org.jhove2.module.identify.DroidIdentifier"
    scope="prototype">
  <property name="developers">
    <list value-type="org.jhove2.core.Agent">
      <ref bean="CDLAgent"/>
      <ref bean="PorticoAgent"/>
      <ref bean="StanfordAgent"/>
    </list>
  </property>
  <property name="configFilePath" ref="droidConfigFilePath"/>
  <property name="sigFilePath" ref="droidSigFilePath"/>
</bean>
```
Spring configuration: Identification

```xml
<bean id="Identifier" class="org.jhove2.module.identify.IdentifierModule"
    scope="prototype">
    <property name="developers">
        <list value-type="org.jhove2.core.Agent">
            <ref bean="CDLAgent"/>
            <ref bean="PorticoAgent"/>
            <ref bean="StanfordAgent"/>
        </list>
    </property>
    <property name="fileSourceIdentifier" ref="bsdIdentifier"/>
</bean>

<bean id="bsdIdentifier" class="org.myinstitution.identify.BsdFileIdentifier"
    scope="prototype">
    <property name="developers">
        <list value-type="org.jhove2.core.Agent">
            <ref bean="MYINSTITUTIONAGENT"/>
        </list>
    </property>
    <property name="runtimepath" ref="bsdFileRuntimePath"/>
</bean>
```
Documentation

http://www.jhove2.org/

Installation and Configuration

– JHOVE2 User’s Guide

Technical information

– Architecture Document
– Format Module Specifications
– How to Write a Format Module
Agenda

Introduction and concepts
Demonstration
Architecture and APIs
Assessment
Sustaining the JHOVE2 open source community
Discussion
Assessment

Evaluation of prior characterization information relative to local policy

Assessment results can inform preservation decision making

– Determine level of risk
– Assign level of service
– Take action now or later
Assessment rules

Assertions whose terms are logical expressions based on prior characterization properties

– Presence/absence of a property
– Constraints on property values
– Combinations of properties/values

The evaluation of the assertion results in new characterization properties

– Custom metadata that has significance in a local context
Assessment implementation

Each format module has a default rule set

Rules are configured using ARules

– Utility developed by CDL to create rule set in XML
– Future plans: a GUI

Predicates (conditions) are evaluated using MVEL

– http://mvel.codehaus.org/
Assessment rules

Logical expressions of the form:

If *condition* then *consequent* else *alternative*

- A condition is defined by either a universal or existential qualifier
  
  $\forall$ “for all”
  
  $\exists$ “for any”
  
  ¬ “not any”

  and an arbitrary set of predicates (logical assertions) of the form

  *property* *relation* *value*

- Supported relational operators

  $==$  $!=$  $<$  $>$  $<=$  $=>$  contains  exists
Assessment rule

JPEG 2000 example *(pseudo-code)*

If ALL_OF
  validity == true;
  exists(colourBox);
  exists(resolutionBox.capture)
Then
  Acceptable
Else
  Not acceptable
End If
Assessment rule

TIFF example

If ANY_OF
    validity == true ;
    ((ifh.messages contains ‘offsetNotByteAligned’) or
     (ifd.messages contains ‘offsetNotByteAligned’) or
     (ifd.messages contains ‘dateNotWellFormed’))
Then
    Acceptable
Else
    Not acceptable
End If
Assessment rule

WAVE example

If ALL_OF validity == true ;
exists(broadcastWaveExtensionChunk) ;
waveFormatChunk.nSamplesPerSec == 96000 ;
waveFormatChunk.nBitsPerSample == 24
Then
Acceptable
Else
Not acceptable
End If
Assessment rule

XML example

If ANY_OF
  validity == true ;
  (validity == undetermined) and
  (wellFormed == true)
Then
  Acceptable
Else
  Not acceptable
End If
ARules configuration

ruleset XmlRuleSet enabled org.jhove2.module.format.xml.XmlModule
desc Ruleset for XML module

rule XmlStandaloneRule enabled
desc Does XML Declaration specify standalone status?
cons Is Standalone
alt Is Not Standalone
quant all
pred xmlDeclaration.standalone == "yes"

rule XmlAcceptableRule enabled
desc Is the XML status acceptable?
cons Acceptable
alt Not Acceptable
quant any
pred valid.name() == "True"
pred (valid.name() == "Undetermined")
   && (wellFormed.name() == "True")
ARules utility output

<bean id="XmlRuleSet" class="org.jhove2.module.assess.RuleSet" scope="singleton">
    <property name="name" value="XmlRuleSet" />
    <property name="description" value="RuleSet for Xml Module" />
    <property name="objectFilter"
        value="org.jhove2.module.format.xml.XmlModule" />
    <property name="rules" value-type="org.jhove2.module.assess.Rule">
        <ref local="XmlStandaloneRule" />
        <ref local="XmlValidityRule" />
    </list>
    <property name="enabled" value="true" />
</bean>
ARules utility output

<!-- Rule bean for evaluating validity value -->
<bean id="XmlValidityRule"
    class="org.jhove2.module.assess.Rule" scope="singleton">
    <property name="name" value="XmlValidityRule"/>
    <property name="description" value="Is the XML validity status acceptable?"/>
    <property name="consequent" value="Acceptable"/>
    <property name="alternative" value="Not Acceptable"/>
    <property name="quantifier" value="ANY_OF"/>
    <property name="predicates">
        <list value-type="java.lang.String">
            <value><![CDATA[(valid.toString() == 'true')]></value>
            <value><![CDATA[(valid.toString() == 'undetermined') && (wellFormed.toString() == 'true')]]></value>
        </list>
    </property>
    <property name="enabled" value="true"/>
</bean>
Module {AssessmentModule}:
AssessmentResultSets:
AssessmentResultSet:
  RuleSetName: XmlRuleSet
  RuleSetDescription: Ruleset for XML module
  ObjectFilter: org.jhove2.module.format.xml.XmlModule
  BooleanResult: false
AssessmentResults:
AssessmentResult:
  RuleName: XmlStandaloneRule
  RuleDescription: Does XML Declaration specify standalone status?
  BooleanResult: false
  NarrativeResult: Is Not Standalone
  AssessmentDetails: ALL_OF { xmlDeclaration.standalone == "yes" => false; }
AssessmentResult:
  RuleName: XmlAcceptableRule
  RuleDescription: Is the XML status acceptable?
  BooleanResult: true
  NarrativeResult: Acceptable
  AssessmentDetails: ANY_OF { valid.name() == "True" => true;(valid.name() == "Undetermined") && (wellFormed.name() == "True") => false; }
Practical applications

Assessment has practical applications in

– Ingest workflows
– Migration workflows
– Digitization workflows
– Publishing workflows

It can be extended to build tools capable of more complex analyses

– Weighted scoring system
– “Institutional technology profiles”
Other Assessment Activities

- Archive Ingest and Handling Test
  Stanford University Libraries

- AONS II (Automated Obsolescence Notification System)
  National Library of Australia and APSR

- CIV (Configurable Image Validator)
  Library of Congress

- Institutional Technology Profiles
  National Library of New Zealand
Agenda

Introduction and concepts
Demonstration
Architecture and APIs
Assessment
Sustaining the JHOVE2 open source community
Discussion
User survey

145 respondents, 88 institutions, 23 countries

3) How quickly do you plan to begin using JHOVE2 after its release?

- 0-6 months: 64%
- 7-12 months: 18%
- 13-18 months: 2%
- Not sure: 16%

5) Please characterize how you will use JHOVE2

- Integral part of validation process for all digital objects we ingest into our repository: 56%
- Integral part of validation process for all digital objects we create: 20%
- Ad hoc use: 14%
- Integral part of quality assurance process for all digital objects we receive from outside: 24%
- Other: 3%
- Not sure: 3%
Sustainability

Project partners will provide 3 years of self-funded maintenance (*but not development*)

- Support and maintain the core JHOVE2 code
- Provide training on integration and use
- Solicit and support 3rd party module development
- Solicit and support integration with other systems
- Grow a lightweight community structure to guide and foster JHOVE2 technical development

Define a long-term sustaining strategy
Community roles

Users (read-only)
Contributors / Documenters (read/submit)
Committers (read/write/commit/release)
Sponsors (fund/resource)
Steering group (strategize/prioritize/incubate/outreach)
Educators (support/train)
Workshops and training

Workshop possibilities

- Code4lib  (Bloomington, Feb. 7-10, 2011)
- IS&T Archiving  (Salt Lake City, May 16-19, 2011)
- Open Repositories  (Austin, June 8-11, 2011)

Anticipate more trainings, more vehicles

- Train the trainer (Planets? Washington DC?)
- Webinars and videos

Suggestions welcome, volunteers encouraged
Future developments

3rd party development activities

– Integration with DuraCloud (DuraSpace)
– ARC module (Bibliothèque nationale de France)
– GIF, HTML, JPEG, PNG, virus, WARC modules (CDL / Deutsche Nationalbibliothek)

Possible development efforts

– Additional format modules
– Configuration GUIs
– JHOVE2-as-a-service
– Integration with DAITTS, DSpace, Fedora, FITS, etc.

Suggestions, volunteers and funders welcome
Questions / Discussion

http://jhove2.org

JHOVE2-Announce-L@listserv.ucop.edu
JHOVE2-Techtalk-L@listserv.ucop.edu

CDL
Stephen Abrams
Patricia Cruse
John Kunze
Isaac Rabinovitch
Marisa Strong
Perry Willett

Portico
John Meyer
Sheila Morrissey

Library of Congress
Martha Anderson
Justin Littman

With help from
Walter Henry
Nancy Hoebelheinrich
Keith Johnson
Evan Owens

Stanford University
Richard Anderson
Tom Cramer
Hannah Frost

Advisory Board
Bibliothèque national de France
Deutsche Nationalbibliothek
Dspace / MIT
Ex Libris
Fedora Commons / Rutgers
Florida Center for Library Automation
Harvard University
Koninklijke Bibliotheek
National Archives (UK)
National Archives (US)
National Library of Australia
National Library of New Zealand
Planets / Universität zu Köln
Tessella