Introducing veraPDF

Carl Wilson

Open Preservation Foundation

Preserving Documents Forever

Oxford 15th July 2015

Open Preservation Foundation

veraPDF

veraPDF is three things:

- a project
  http://verapdf.org/
- a consortium
  http://verapdf.org/project/#team
- a software product
  http://github.com/veraPDF/

Open Preservation Foundation

Overview

- Project Background
  A short history lesson & the veraPDF Consortium
- PDF/A Validation
  Some technical background & our approach to validation
- Building Software & Community
  Projects and tools used in making veraPDF
  How you can get involved in our future plans

PREFORMA & Project Beginings

The project started when these organisations:

- Open Preservation Foundation
- PDF Association
- Digital Preservation Coalition

joined forces to answer the Challenge Brief written by the EU funded PREFORMA project.

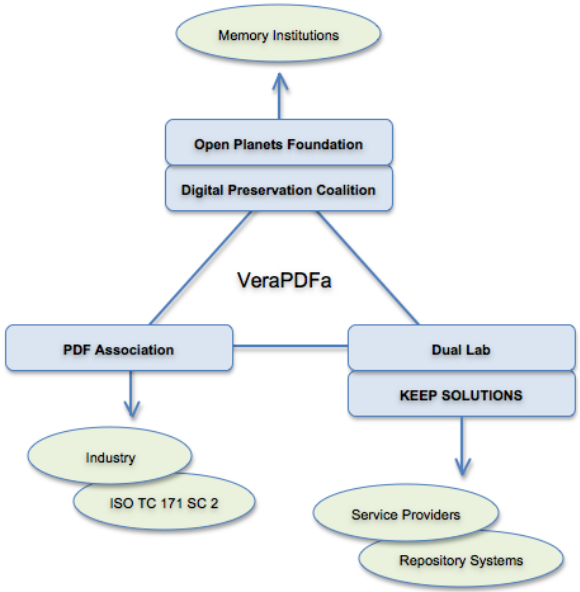
veraPDF Consortium

Bringing together:

- Memory institutions
- PDF industry
- ISO domain experts

Plus technical partners:

- Dual Labs
- KEEP Solutions



PREFORMA

PREservation FORMAts for culture information/e-archives

- EC FP7-ICT pre-commercial procurement (PCP)

http://www.preforma-project.eu/

The PREFORMA Challenge

For one of three formats, PDF/A, TIFF, or “moving image” produce a Conformance Checker that:

- validates a file against according to the specification of a standard file format;
- validates whether a file matches institutional acceptance criteria for long term preservation;
- reports the results of validation; and
- performs automated fixes for simple metadata problems.

Conformance Checker

Or put another way the sub-components are:

- Implementation Checker (format validator)
- Policy Checker (long term archiving policy enforcement)
- Reporter (human and machine readable)
- Metadata Fixer (does what it says on the tin)
- User Interfaces (library/API, CLI, Desktop GUI, & web)

Specifications Galore

Our response:

- Functional Specification:
  Scope of PDF/A validation, policy requirements, used cases and integrations with repository systems.

- Technical Specification:
  Implementation technologies, validation and policy profiles, report formats, test corpora….

Plus Other Documentation

- Community and comms plan
- Review of existing test corpora
- Evaluation of PDFBox
- License compatibility assessment
- Project plans

and on, and on == 300+ pages

The Result

In May the veraPDF tender was successful, here's the PREFORMA timetable:

- First prototyping: May 2015 - Oct 2015
- Redesign: Nov 2015 - Feb 2016
- Second prototyping: Mar 2016 - Dec 2016

The PDF/A Validation Problem

- Relies on two different versions of PDF, with specifications 1000+ pages long
- PDF/A comes in three versions, three levels and two technical corrigenda for PDF/A-1
- The PDF/A standard aims for long-term (100+ years!) preservation of electronic documents
- Requires very formal analysis of the requirements

PDF/A ISO Standards

All PDF/A standards parts of ISO 19005

Document management -- Electronic document file format for long-term preservation

- ISO 19005-1:2005 Part 1: Use of PDF 1.4 (PDF/A-1)
- ISO 19005-2:2011 Part 2: Use of ISO 32000-1 (PDF/A-2)
- ISO 19005-3:2012 Part 3: Use of ISO 32000-1 with support for embedded files (PDF/A-3)

Note that ISO 32000-1 is the ISO standard for PDF 1.7 and that PDF 1.4 (the basis for PDF/A-1) is NOT an ISO standard.

PDF/A Levels

veraPDF

Subdivided into conformance levels:

Parts 1 - 3 define:

- Level b - ensures the reliable reproduction of visual appearance.
- Level a - Level b plus requirements for document structure, tagging, and language.

In addition Parts 2 & 3 define:

- Level u - Level b plus unicode character mappings.

Open Preservation Foundation

8 PDF/A Flavours

- PDF/A-1b - PDF 1.4, visual reproduction
- PDF/A-1a - PDF 1.4, level b + structure
- PDF/A-2b - PDF 1.7, visual reproduction
- PDF/A-2a - PDF 1.7, level b + structure
- PDF/A-2u - PDF 1.7, level b + unicode map
- PDF/A-3b - PDF 1.7, visual reproduction + attachments
- PDF/A-3a - PDF 1.7, level b + structure + attachments
- PDF/A-3u - PDF 1.7, level b + unicode + attachments

Establishing Ground Truth

- Isartor, Bavaria, BFO: 323 test atomic self-documented test files.
- Initial test case analysis shows about 500 extra files needed for proper test coverage of all PDF/A versions and levels.
- Subject to adjustment based on real-world cases and practical considerations.
- To compare: the W3C corpus for XML 1.1 (specified on 38 pages) contains over 600 test files.

Creating Test Files

- Sample clause of PDF/A-1 specification:
  - ISO 19005-1:2005, 6.1.6 String objects: Hexadecimal strings shall contain an even number of non white-space characters, each in the range 0 to 9, A to F or a to f.
  - ISO 19005-1:2005, 3.17 white-space character: NULL (00h), HORIZONTAL TABULATION (09h), LINE FEED (0Ah), FORM FEED (0Ch), CARRIAGE RETURN (0Dh) or SPACE (20h) character
- Test cases identified:
  - Odd number of characters 0-9,A-F,a-f (fail)
  - Non white-space characters outside the range 0-9,A-F,a-f (fail)
  - Even number of 0-9,A-F,a-f; all possible white-space characters (pass)

Creating Test Files

- Test files created (QA engineer):
  - veraPDF test suite 6-1-6-t01-fail-a.pdf
  - veraPDF test suite 6-1-6-t01-fail-b.pdf
  - veraPDF test suite 6-1-6-t01-pass-a.pdf
- Github pull request is submitted (QA engineer)
- The file is merged to the master branch after the internal review (Architect):

https://github.com/veraPDF/veraPDF-corpus

Acceptance Process

- In case the specification allows several interpretations, or when a representative set of existing validators generates mixed results:
  - Discuss with industry experts on the PDF Validation TWG mailing list
  - Review and resolve discussions during TWG meetings
- The corpus is announced as a release candidate
- Formal acceptance by PDF Validation TWG
- The next milestone: July 15, 2015
  (release candidate of the enhanced PDF/A-1b corpus)

TWG Discussions

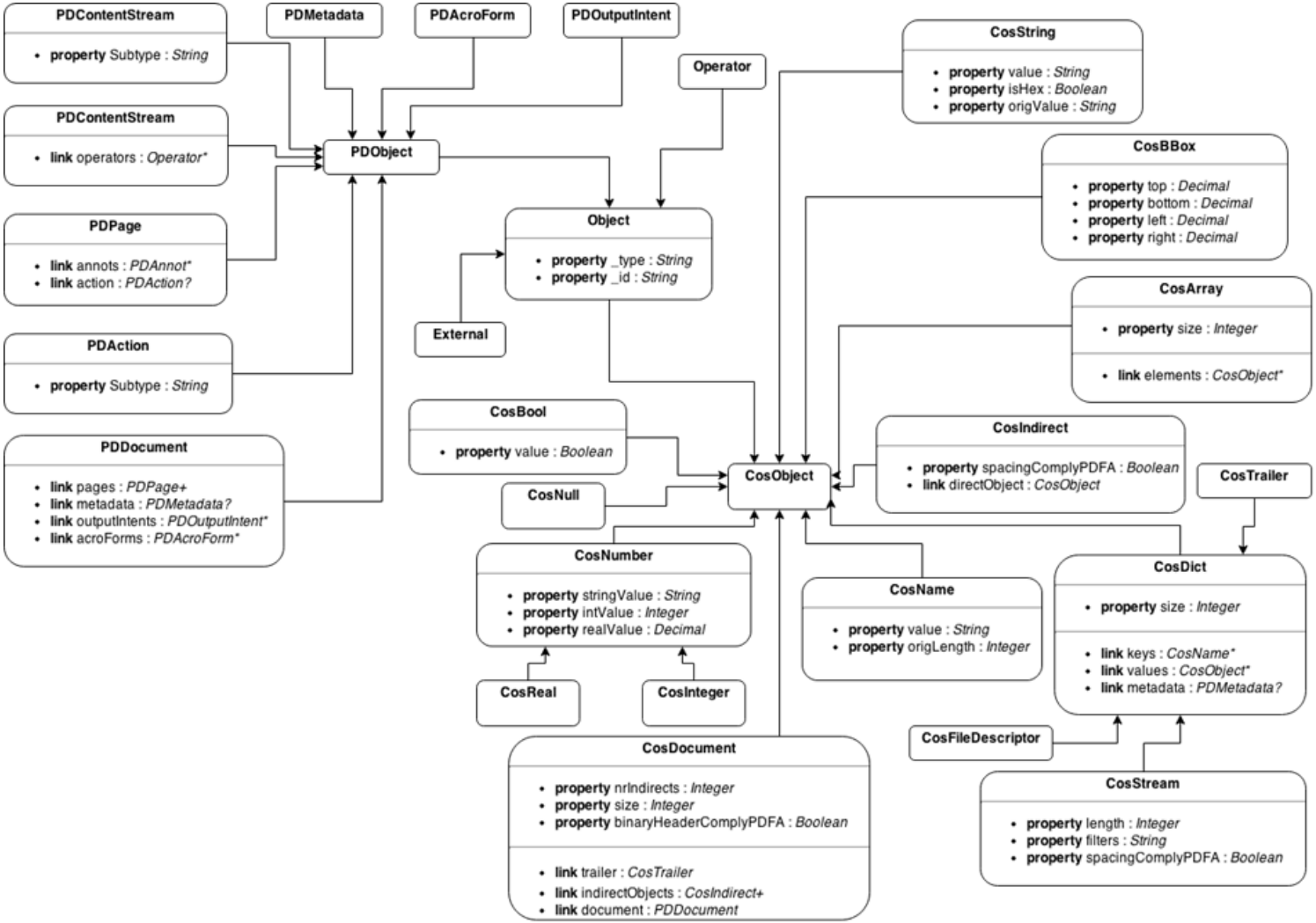
- Private data: does it need to be validated for File Structure requirements?
- Which glyphs shall have widths synchronized with embedded fonts data?
- What is the length of a Name, which is a subject to implementation limits?
- Is it a violation of PDF/A-2,3 requirements, if a Page object does not contain a Resources dictionary, but inherits resources from its parent Pages object?

Validation Extents

- PDF/A specification refers to PDF specifications, which rely on a number of external standards. How far PDF/A validation shall go to guarantee the reliable long-term preservation of PDF documents?
- Validity of embedded fonts, ICC profiles, font CMaps, XMP Metadata, Image compression, digital certificates is crucial!
- Collaboration with experts from relevant technologies is necessary for complete coverage

Validation Model

- Hierarchy of object types
- Inheritable properties and named links to other object types
- Validation rules defined per object type (inheritable) by boolean expressions in JavaScript
- Each rule has metadata including the ID, description, normative references, error message
- Rules are combined into validation profiles and signed



Open Preservation Foundation

PDF Model as Xtext Syntax

```
# a single root type
type Object {};
# parent object for all basic object types in PDF
type CosObject extends Object {};
# the object representing PDF Array type
type CosArray extends CosObject {
     property size: Integer;
     link values: CosObject*;
};
# parent object for all content stream operators
type Operator extends Object {};
# Operator q (save graphics state)
type qOperator extends Operator {
     property nestingLevel: Integer;
};
```

```
<rule id="1a-6-1-12-r05" object="CosArray">
<description>Maximum capacity of array in elements less than
8192</description>
    <test>size &lt; 8192</test>
<error>
    <message>Capacity of array greater than 8191</message>
</error>
<reference>
    <specification>ISO 19005-1:2005</specification>
    <clause>6.1.12</clause>
</reference>
<reference>
    <specification>PDF Reference 1.4</specification>
    <clause>Table C.1</clause>
</reference>
</rule>
```

PDF Parser

- PDF Parser implements the interfaces automatically generated from the PDF Model syntax
- Proof of concept is implemented using PDFBox, an open-source Java library
- Due to licensing requirements of PREFORMA all code of the PDF/A Conformance Checker shall be delivered under the dual license: GPLv3 + MPLv2
- As PDFBox is Apache-licensed, PREFORMA requested a greenfield implementation of a PDF parser

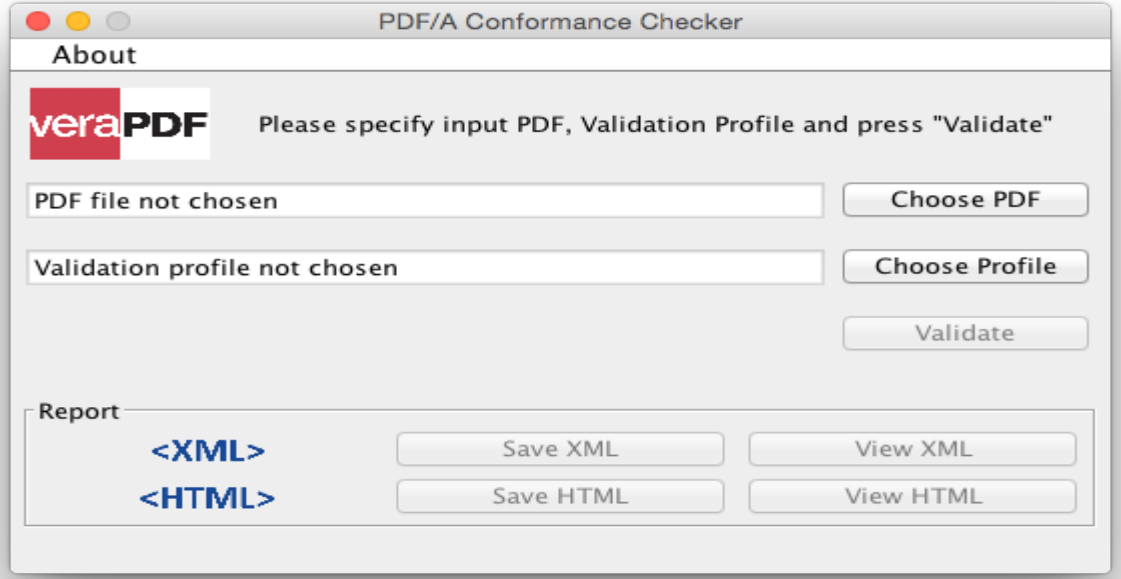
Keep up with Developments

- Github is the repository for both code and test corpora
  https://github.com/veraPDF
- Travis manages the continuous builds
  https://travis-ci.org/veraPDF
- Jenkins manages automated tests and deployment
  http://jenkins.opf-labs.org/job/veraPDF-library/
- Sonar monitors code quality
  http://sonar.opf-labs.org/dashboard/index/8021

Current veraPDF Library

- Clone Java repository https://github.com/veraPDF/veraPDF-library
- Build the veraPDF library and the veraPDF GUI client
- Check the location of test files and validation profiles
- Run the veraPDF client selecting any PDF file and any validation profile
- Generate the machine-readable report (XML)
- Convert it to HTML and view in a system browser

Screenshot


PDF/A Conformance Checker
About
veraPDF
Please specify input PDF, Validation Profile and press "Validate"
PDF file not chosen
Choose PDF
Validation profile not chosen
Choose Profile
Validate
Report
<XML>
Save XML
View XML
<HTML>
Save HTML
View HTML

Reporting

- Machine-readable reports generated in XML format
- Error messages defined in validation profiles
- Conversion to HTML by XSLT technology
- Conversion to PDF by XSLT to XSL-FO and further to PDF by Apache FOP processor
- Localization of the final report messages via the standard TMX syntax of translations

Policy Checking

Schematron selected because:

- it provides a mechanism for defining and enforcing assertions for XML documents;
- used successfully for JP2K policy enforcement; and
- used successfully to test PDF policy checking.

Acknowledgment to the work done by Johan van der Knijff of the KB and the BL's FLINT project in this area.

What does this Look Like?

Disallow TrueType fonts:

```
<rule context="/verapdf:report/verapdf:pdfDetails/verapdf:
documentResources/verapdf:fonts/verapdf:font">
    <report test="count(current()[@name='TrueType']) > 0">Policy check
error: usage of the filter TrueType fonts in the document is not allowed!
</report>
</rule>
```

Testable Policy Requirements

To be accepted for testing in the prototyping phase policy requirements must consist of:

- a textual statement of the policy rule, supplied by the institution;
- an owner, usually an individual from the institution;
- a schematron rule, or rules created with the help of veraPDF consortium; and
- test files that express pass and fail cases for inclusion in the corpora, again created with the help of the consortium.

Get Involved Early

Early participation mean first hand assistance with:

- expressing policy requirements as Schematron rules; and
- creating test files that represent your institutional requirements.

In turn this guarantees that your policy requirements are:

- represented in the veraPDF policy checking corpus;
- included in early testing; and
- considered during the Phase 2 re-design if they’re not satisfied by the first design.

Getting Involved

- Submitting bug reports and enhancement requests via GitHub issues.
- Helping to create user documentation.
- Translation of software messages and documentation.
- Developers can submit fixes and enhancements.