BitstreamFormat Renovation: DSpace Gets Real Technical Metadata

Larry Stone – MIT Libraries, Digital Library Research Group

Abstract

This paper introduces a new way of modeling Bitstreams' technical metadata, available now as prototype code, that lets DSpace interoperate with external data format registries and preservation tools. It also delivers much more powerful and accurate file format identification, and gives DSpace access to high-quality format technical metadata. This constitutes an essential first step toward improving the digital preservation capabilities of DSpace. There is also a "backward-compatible" mode to accommodate sites that have no interest in preservation and prefer the old simplistic view of file formats.

1 Motivation for Technical Metadata

The new BitstreamFormat design and prototype code wrap a plugin framework around the tasks of naming data formats, and identifying the format of a Bitstream. This gives your DSpace repository the option of getting its format names and technical metadata from an external format registry, which in turn gives it a language in common with other applications that subscribe to the same format registry. The new automatic format identification generates more accurate technical metadata and helps the archive administrator assess its success, which also improves interoperability because more resources will indicate a correct and meaningful data format. Finally, since accurate and fine-grained format technical metadata is the foundation of digital preservation activities such as format migration and obsolescence notification, this code is essential for DSpace to fulfill its potential as a preservation platform.

We also reiterate that the plugin architecture allows complete backward-compatibility as well as great flexibility and extensibility, all through add-ons and configuration changes. Sites that prefer the current DSpace behavior and its internal file format “registry” can continue to use it, and still enjoy the improvements in format identification.

2 About Data Formats

"Data format" is defined as technical metadata that describes how *abstract information* is encoded and structured in a digital document. This abstract information is the *intellectual content* represented by the digital object. For example, take a digital image of Van Gogh's *Starry Night*: the *intellectual content* is the picture that appears when the image is displayed or printed, i.e. the painting. The *data format* is technical metadata that describes how to interpret the bytes of that digital object to render the image correctly on an output device. It shows how the colors and pixels of the image are encoded.

What does this format technical metadata do for DSpace, and what *might* it do?

- When disseminating content, it identifies the format to the recipient. For example, when serving a Bitstream to an HTTP (Web) client, it provides a `Content-Type` header naming the Internet Media Type (MIME type) of the Bitstream.
- Internally, the `MediaFilter` matches Bitstreams to filter plugins by their file formats. The formats must be identified accurately for this to work.
- Formats affect the UI, too: For example, an Item is displayed differently if its principle Bitstream has an HTML format (implying an archived website)

- When exchanging digital objects with other systems, DSpace should be able to recognize format metadata when ingesting a SIP, and supply recognizable format names in DIPs.
- The DSpace software detects when a Bitstream's format becomes obsolete or otherwise presents a preservation problem, and notifies the archive administrator appropriately.

3 Problems with Current Format Technical Metadata Model

The data model for technical metadata (the BitstreamFormat object) has been essentially unchanged from the inception of DSpace through release 1.5. Although its design anticipated the use of external format registries, it was never completed. There are some other problems with the current data model and implementation:

1. Formats are identified by arbitrarily-assigned descriptive names such as "Adobe PDF" which have no meaning outside of DSpace. This impedes any attempt to share format descriptions with any other application.
2. There is no provision for collecting more extensive format technical metadata, such as standards documents, that would help future preservationists interpret obsolete formats.
3. A Bitstream's format is only identified by comparing its filename extension to entries in the format registry. This method is prone to errors, ambiguous results, and outright failures.
4. Since the format "registry" is stored in the RDBMS, and populated at installation time, it is difficult to update with new versions (and in fact has not yet been modified in an upgrade). The way additions and alterations to format entries are simply folded into the same table further complicates automated upgrades.

The new model for format metadata offers solutions for all of these problems, as well as new capabilities that support digital preservation activities.

4 Introducing the BitstreamFormat Renovation Prototype

The new BitstreamFormat implementation has two major components: a data model for format technical metadata that relies on *external format registries*, and a new framework for identifying the format of Bitstreams. The BitstreamFormat object now functions mainly as a pointer to an entry in an external registry. It caches some format metadata, both for efficiency and to allow local modifications, but the external registry is the authoritative source. The external format registry is accessed through a plugin interface, so new registries can be integrated easily as add-ons.

The internal cache of BitstreamFormats is automatically populated: When an external format identifier is assigned to a Bitstream, a new cache entry is created for it if necessary. The cache can also be updated in bulk against an external registry, to keep up with changes and corrections.

A new format-identification framework improves the accuracy of automatic format identification and adds some metrics. Formats are identified by a stack of *format identifier method* plugins, each of which may examine anything about the Bitstream (including its content) to deduce the format. Each Bitstream now has a record of the *confidence* (expectation that it is accurate) of the identified format, and its *source* (software module or plugin that produced the answer). These metrics help archive administrators evaluate the accuracy of automatic format identification and easily fix problems and take advantage of improvements.

For complete details please see the *BitstreamFormat Renovations* page on the DSpace Wiki: http://wiki.dspace.org/index.php/BitstreamFormat_Renovation.

4.1 Backward Compatibility

For sites that are satisfied with the current small set of file formats, the prototype includes a backward-compatible "external registry" named `DSpace` that maintains the current behavior. They can still benefit from the improvements in automatic format identification, and easier management of locally-added format metadata. Also, it is always possible to change configurations later, to a more sophisticated external format registry.

4.2 Architecture

Figure 1 shows an example of the new object-model architecture connected to the PRONOM[1] registry. As before, each Bitstream links to a BitstreamFormat object, which may be shared. The solid arrow from the BitstreamFormat shows how its *external identifier* connects it to a PRONOM format description. The dashed arrows back from PRONOM represent metadata fields that were initially imported from the PRONOM entry when the BitstreamFormat was created, but they are not tightly coupled and the archive administrator can modify them locally. Below, note how the DROID[2] format identifier service also refers to PRONOM entries. DSpace includes a DROID format identifier method plugin that, through the heavy dashed line, relies on DROID.

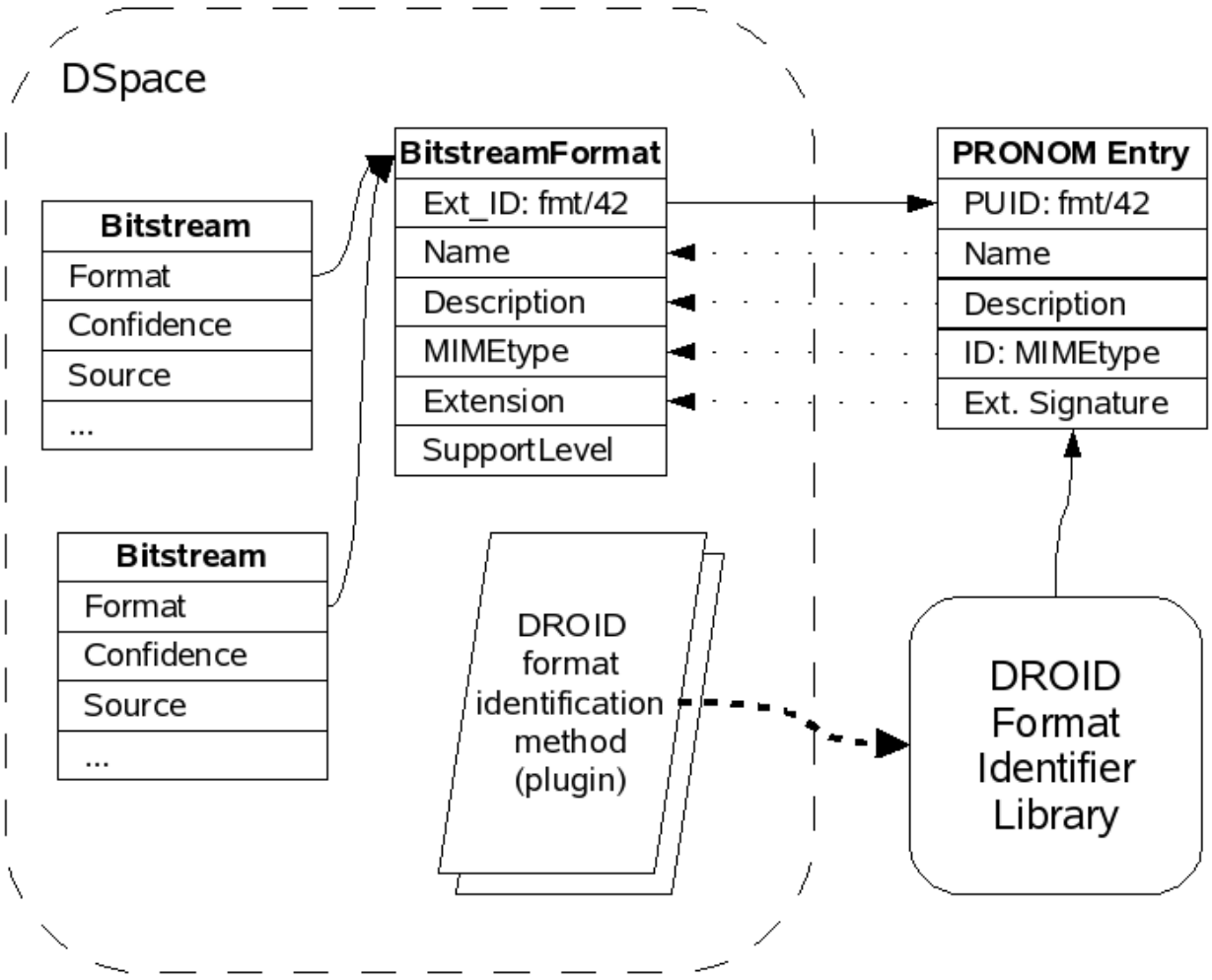

Figure 1: Architecture with PRONOM external registry

1 For more on PRONOM, see http://www.nationalarchives.gov.uk/PRONOM/ .

2 DROID software home page: http://droid.sourceforge.net/ .

The Bitstreams have two new fields: *Confidence* and *Source*. *Confidence* is a metric of how much to believe the accuracy of the identification, perhaps indicating the method. Its values are symbolic constants such as POSITIVE_INTERNAL (identified positively by contents of Bitstream), HEURISTIC (a guess based on the contents), CIRCUMSTANTIAL (using external cues such as the filename), etc. *Source* is a string naming the software element responsible for identifying the format. It is helpful to know this when diagnosing identification problems, searching out Bitstreams to re-identify, and generating reports for analysis.

4.3 Interface to External Registry

The interface to the external format registry includes the following operations:

- **Get Synonyms** for identifier: Returns a list of all identifiers which are bound to the same format record. Some registries, e.g. the GDFR[3], can be expected to contain synonyms since they are compilations of other format registries.
- **Import**: Turns an external format description into a new BitstreamFormat entry, initializing its metadata fields from the external registry.
- **Update**: Refresh the metadata fields of a BitstreamFormat from the indicated external registry entry, to keep up with changes.
- **ConformsTo**: Test whether the format described by one identifier in the external registry "conforms to", or is a subtype of, another format. This is used e.g. by the MediaFilters to detect whether a Bitstream matches a filter's input type.

The prototype includes registry plugins for PRONOM, the backward-compatible "DSpace" registry, and a "Provisional" registry for the site administrator to add local extensions to the registry. When the GDFR is released as a production service (planned for later in 2008), it can easily be integrated with DSpace by adding a plugin for it.

Note that the external registry is expected to have much more extensive technical metadata than we actually make use of. The BitstreamFormat's external identifier leads directly to the registry entry, so it is quite straightforward to get more information. For example, a Web-based user interface could construct a link to that entry on the format registry's Web UI.

4.4 Automatic Format Identification

We believe the improved automatic format identification mechanism is one of the most important benefits of this project. Very few (if any) submitters truly understand the significance of data formats so they simply accept the default format when submitting interactively. Batch and package submissions don't even have a way to confirm formats so they depend completely on the automatic format identification. (Aside: The prototype includes a PREMIS[4] crosswalk that lets the package ingester take a Bitstream's format from technical metadata in a METS package, instead of guessing it. It only works for globally-recognized persistent format identifiers such as PRONOM PUIDs.)

The automatic format identifier calls each of the configured *identifier methods*, in order, and returns a ordered list of results. The first element is usually the answer but the list is available in case there is an interactive user interested in viewing it. Each method is given the Bitstream and the current results list. It may do any or all of:

---

3 GDFR website: https://collaborate.oclc.org/wiki/gdfr/about.html .

4 PREMIS home page: http://www.loc.gov/standards/premis/ .

- Attempt to identify the format based on the Bitstream's contents and/or metadata. Each "result" includes the external format identifier, confidence value, and name of the method as "source".
- Review other results and attempt to refine the result, e.g. if the Bitstream is a supertype of its format. For example, if the results include a hit for "XML", check if the root element matches one for a the schema of a known format and add the more precise result if found.
- Analyze the results list, rearranging or deleting some of the elements.

As you can see, this is a very flexible framework that encourages constant experimentation refinement. We expect to develop a thriving commerce in new identifier methods and configuration hints among DSpace site administrators.

4.5 Initial Upgrade Procedure

To deploy this prototype, the administrator has to run a Java-based upgrade procedure to transform the data model to the new schema. This, in turn, requires making some policy decisions: most importantly, whether to use the backward-compatible "DSpace" registry or an actual external format registry. Then, the format-identifier stack has to be configured, and other configuration entries set.

In the current prototype, the administrator performs the upgrade in several steps, and is required to check results and/or make decisions after each one. This could all be substantially automated, especially in the simplest case of the backward-compatible format registry. The upgrade process will have to be simple, automatic, and trouble-free, especially to be acceptable at sites that do not need its benefits in preservation and interoperability and so will resent any extra trouble.

The upgrade is done in 4 phases, each one initiated by a single command:

i. Add new tables and columns to RDBMS, moving old Bitstream format representation to a temporary column.

ii. Populate (with human assistance) internal "Provisional" format registry. Transform existing format entries to their equivalents in the "DSpace" and "Provisional" registries, if possible.

iii. Invoke the automatic format identifier mechanism to assign formats to all remaining Bitstreams. When the configured format registry is PRONOM, this re-identifies most of them, but it is minimal when using the backward-compatible "DSpace" registry.

iv. Check for anomalies and mistakes in the format conversion, and finish changes to the RDBMS. The old format columns and tables are deleted.

5 Results and Conclusions

The new BitstreamFormat architecture addresses all of the problems mentioned earlier:

1. By naming formats with identifiers from an external registry, it allows interoperability with any other application using that registry. For example, the AONS II[5] notification service also names formats with PRONOM PUIDs, so it will understand when DSpace names formats with PUIDs in Bitstream technical metadata.
   Note that using the "backward compatible registry" *prevents* interoperability, since it retains the old DSpace-specific format names.
2. Connecting DSpace to a sophisticated external format registry like PRONOM gives it access to much more extensive technical metadata. The GDFR has a similar data model and will build on

5 For more about AONS II, see http://pilot.apsr.edu.au/wiki/index.php/AONS_II .

PRONOM's contents, so these are the two best options for the forseeable future.

3. We address the problem of format identification by leveraging DSpace's open-source model. By establishing an open framework we invite the community to add to and improve the format-identification tools. The prototype includes a set of methods that demonstrate enough of an advance to validate this approach.
4. DSpace version upgrades are addressed in two ways: the backward-compatible "DSpace" registry is driven by an easily-upgraded data file, which is never modified locally because any site-specific changes go in the parallel local registry named "Provisional". The "DSpace" registry is treated as an immutable part of the software distribution.

   If the archive configures an external format registry instead, then that registry gets upgraded independently of DSpace. The GDFR will be automatically updated in a distributed manner, for example.

5.1 Quantitative Results

Here are the results of an experiment using the contents of one MIT production DSpace. We deployed the prototype code, configured with PRONOM as its external format registry and DROID (along with some heuristic and helper methods) for automatic format identification.

The archive contains about 155,000 Bitstreams. Before converting to PRONOM formats, there were 1,020 unidentified Bitstreams (or 0.65%). Afterward, there were only 162 (or 0.104%). Under the old DSpace format model, the archive had 21 distinct formats. Using the PRONOM registry and DROID identifier it classified the same Bitstreams into 52 distinct file formats, almost two-and-one-half times as many. (Some of this growth is due to the way PRONOM discriminates between separate versions of a format, however.) The new format identification also revealed several Bitstreams which had been identified incorrectly before, and a few dozen cases of corrupt formats.

5.2 Digital Preservation Strategies

To address digital preservation problems and take advantage of the tools currently being developed, DSpace needs more fine-grained format classification, as well as globally-recognized identifiers for formats that will be understood by other applications. Both of these are provided by a preservation-minded external format registry such as PRONOM or GDFR.

A recent paper by the PRESERV project[6] demonstrates a practical Web-based service using PRONOM identifiers. Their application scans an archive for digital objects with formats in danger of becoming obsolete, and reports them to the archive administrator. This service requires fine-grained format identifiers, i.e. discriminating between versions of a format: the oldest versions of a given format (e.g. Microsoft Word 2003) are likely to become endangered years before the newer ones. So, coarse-grained format metadata such as "MS Word" is useless to it. To make up for this lack in DSpace, the PRONOM-ROAR project expends a lot of effort fetching all of a DSpace's assets and running the DROID identifier on them to get fine-grained PUIDs. If DSpace had PUIDs as format identifiers and could export them -- e.g. through the OAI-PMH protocol that ROAR already uses and a PREMIS crosswalk plugin) then ROAR could save all that overhead. It would also get the format technical metadata produced by DSpace's FormatIdentifier stack, which is likely to be more accurate (and better tailored to the repository's contents) than DROID alone.

6 On http://preserv.eprints.org/papers.shtml see PRONOM-ROAR: Adding Format Profiles to a Repository Registry to Inform Preservation Services

Validation is another valuable preservation tool. When promising to keep the information in a Bitstream available, it is essential to know that the contents are a correct and valid example of the format – otherwise it may prove impossible to convert it to a newer format, or even access it with current interpreters. Projects like JHOVE[7] supply validation tools that depend on knowing the identified format, so again it is crucial to have the format expressed as a globally-recognized identifier to ensure interoperation. The data model is ready to record validation; successfully validated Bitstreams are marked with a Confidence value of "VALIDATED".

5.3 Next Steps

The BitstreamFormat prototype is a necessary step in the evolution of the DSpace platform. The original designers intended the built-in BitstreamFormat to be replaced by an external registry; now that a reliable and widely-adopted registry is available in the form of PRONOM (and the GDFR holds even greater promise), the environment is ready. Interoperating with distributed preservation tools and communicating recognizable technical metadata in SIP and DIP packages are compelling reasons to adopt external format registries. The backward-compatible option makes this change painless for sites that do not want more sophisticated format technical metadata. The improvements in automatic format identification are valuable to both kinds of sites. We hope this work can be included in the next major release of DSpace.

7 JHOVE home page: http://hul.harvard.edu/jhove/ .