Supporting Oral Histories in Islandora

Name of Authors (alphabetically) - Marcus Emmanuel Barnes, Natkeeran Ledchumykanthan, Kim Pham, Kirsta Stapelfeldt

Abstract

Since 2014, the University of Toronto Scarborough Library’s Digital Scholarship Unit (DSU) has been working on an Islandora-based solution for creating and stewarding oral histories (the Oral Histories solution pack). Although regular updates regarding the status of this work have been presented at Open Repositories conferences, this is the first article to describe the goals and features associated with this codebase, as well as the roadmap for development. An Islandora-based approach is appropriate for addressing the challenges of Oral History, an interdisciplinary methodology with complex notions of authorship and audience that both brings a corresponding complexity of use cases and roots Oral Histories projects in the ever-emerging technical and preservation challenges associated with multimedia and born digital assets. By leveraging Islandora, those embarking on Oral Histories projects benefit from existing community-supported code. By writing and maintaining the Oral Histories solution pack, the library seeks to build on common ground for those supporting Oral Histories projects and encourage a sustainable solution and feature set.

Introduction: Institutional Context

The Digital Scholarship Unit (DSU) at the University of Toronto Scarborough (UTSC) Library works with campus IT to design stable best-practices infrastructures that can support the research conducted by and with faculty on campus as well as digitized and born-digital special collections. The DSU also works with liaison librarians to provide consultation and development services for experimentation and co-curricular pedagogy. Activities undertaken by the unit include helping stakeholders preserve and manage data, teaching digital scholarship tools, providing training on issues relating to scholarly publishing and open access, coordinating access to repository and data services developed by the central University of Toronto libraries system, and giving advice on metadata development and management.

UTSC is 50 years old, and prides itself on a relationship with multimedia innovation, having originally been built with a television studio to develop and transmit closed-circuit lectures (UTSC Timeline). Today, several courses at UTSC incorporate the creation of audiovisual materials into the assignments completed by undergraduate students and researchers seek the means to develop and analyze unique collections of video and audio materials under the banner of Oral History. In 2014, the library embarked on an expansion of the services of the
Digital Scholarship Unit, and several faculty approached the library seeking support for oral history projects.

**Scoping Oral History Support - Requirements**

Oral History is defined more as process than as product, “a field of study and a method of gathering, preserving and interpreting the voices and memories of people, communities, and participants in past events” [1] The breadth of this definition can challenge those tasked with developing infrastructure, software, and workflows to make possible this “gathering, preserving and interpreting” work. Teams working on oral histories commonly request abilities to do the following:

- relate born-digital and surrogate materials drawn from an archive together in a non-exclusive fashion (graph-type relationships)
- manage materials and administrative processes as required by research ethics boards and granting agencies. Materially this means granular permissions, long-term stewardship, and reliable access and discovery of materials
- describe, transcribe, and translate audio and video files

It is also important to note that these features need to be flexible enough to address the unique research and/or collection building goals of each project. Relevant materials can be in multiple formats and contain any number of speakers, and research questions can require, for example, that annotation rather than transcription information be appended to objects and indexed for discovery. As projects evolve, it’s also possible that new data sources will be included, enriched and otherwise integrated into pre-existing data sources. In the end, the processes of oral history appear to defy any consistent structure or approach.

As a result, a software platform that must address the needs of multiple groups needed to be highly flexible and configurable. The UTSC library found that Islandora provided ways of addressing most of these requirements.

**Relating born-digital and surrogate materials together**

Oral history project teams often ask for the ability to link audio and video files with other born-digital and surrogate materials for both administrative and historical context. For example, a recording of an interview might be associated with consent forms agreeing to preservation and publication of all or parts of a recording, or an interview may be associated with other files contextualizing the content of the interview for an audience. Recordings might also be linked together to form an exhibition or collection of interviews on a particular topic. These connections may be non-exclusive - a single article or image may, for example, be relevant to two recorded interviews.

Using Islandora Solution Packs, it is possible to support ingest and display of a number of content types. Islandora Solution Packs are Drupal modules that offer content models,

---

workflows, forms, and integration with other solution packs or third party software projects to serve the needs of particular types of content or a certain user base. Solution packs vary quite widely in their scope and application [2] and often leverage other solution packs or contributed Drupal modules as dependencies. Moreover, the underlying Fedora 3 architecture supports complex and graph type relationships between objects (these functions are further enhanced and extended in Fedora 4) and there are interface mechanisms for linking objects together in compounds and collections, and sharing objects in multiple relationships. The ecosystem of solution packs allowed us to utilize a great deal of existing functionality and extend it for our purposes in storing, displaying, and interrelating multimedia content.

Managing materials and administrative processes

A common request from UTSC researchers is that oral histories be available to the source community in which they were collected, an audience that may be located internationally and may have permissions to some materials and not to others. An academic audience may seek and use a collection of oral histories differently from members of the general public, and these different modes of access require granular permissions for access. In Islandora, Fedora’s XACML-based security is highly configurable, as is the role-based Drupal permissions system. Managing not just access, but also authorship and administration, Islandora allows for fairly complex use cases to address the (often fraught) notions of authorship inherent in Oral History work.

The oral history materials themselves, often containing video and audio files, can be difficult to preserve and serve reliably. Codecs change and these files tend to be large in size. It is widely recognized that the problem of preserving multimedia materials will continue to grow, as the availability of recording equipment makes generating these materials ever easier. The Islandora Video and Audio Solution Packs produce several formats, an access copy, and can be configured alongside utility modules to create and store technical, administrative, and preservation metadata.

Describing, transcribing, and translating audio and video files

We recognized that our need to flexibly describe and index oral histories could be addressed using Islandora’s form building modules (XML Forms) and search (Solr) functions. Islandora’s form building modules (XML Forms) [3] allow for the creation of form templates that can be used to author XML datastreams in a Fedora object. A user can create a new form, associate it with a Content Model in Fedora, and it will become available to users creating a new instance of an object using the Content Model. Once the XML datastream has been created, the GSearch application leverages an XSL stylesheet to transform the new XML into Solr documents, and make the information contained available to Islandora.

---


search functions [4]. Indexing new XML elements often requires an extension to the XSL stylesheet to address any new elements not covered in the default XSL packaged with solution packs.

While the forms and search infrastructure provided a lot of functionality, there was no Islandora mechanism available for researchers who wish to describe, transcribe, or translate interview content and associate it with time codes. Beyond the research applications, UTSC is also bound by the Accessibility for Ontarians with Disabilities Act. This act requires compliance with Web Content Accessibility Guidelines 2.0, and includes a requirement for text representations of non-text content.

Given Islandora’s coverage of many of the other requirements identified as essential in an Oral Histories infrastructure, the group determined that it would be best to develop an additional solution pack that provided these additional transcription, translation, and description features. They were bundled (with some hubris) by the unit as the “Oral History Solution Pack.”

**The Islandora Oral History Solution Pack**

The Oral History Solution Pack (OHSP) has existed in beta form since 2015, and was recently accepted into the Islandora Foundation Labs Repository. The intention is to continue the work to bring the module into the Foundation and release it as part of a regular Islandora release. Those familiar with Islandora will be familiar with the workflow by which one or more source files are transcoded and ingested as Islandora datastreams in an Islandora object: For interface ingestion, users are presented with the option of creating an “Oral History” object, and asked to provide a file and fill out descriptive metadata. The object is uploaded and triggers a series of processes that transcode the source file to one or more derivatives, creates relevant datastreams, and provide feedback to the user once the object has been created. The OHSP follows the same pattern, making available an oral history specific Content Model to collection policies, and leveraging Islandora’s Video and Audio Solution Packs as dependencies to generate derivatives. The module is designed to operate with the rest of Islandora’s released Solution Packs and Utility Modules, and the intention is to provide substantial flexibility in implementation. A major contributor to the concept for implementation is Edward Garrett of Pinedrop, whose transcript Drupal modules were discovered early in the research phase for this project. Transcript UI leverages the concept of “tiers” (also used in transcription tools such as ELAN) to encode various layers of information about an audio or video file (including transcription, translation, and description).

**The Viewer**

Edward Garrett was contracted by the University to abstract the viewer functions from his module into the Transcript UI module, which is a dependency and sub-module for the Oral History Solution Pack. Garrett’s viewer design makes possible the display of information tiers. In Figure 1, it is possible to see how an annotation and transcription tier are both

---

available from the drop-down menu. The names and number of tiers are highly customizable. In Figure 2, it is possible to see how individual tier “cues” are highlighted as a video plays. In Figure 3, it is possible to see user options for the closed captioning functions of the viewer, out-of-the-box elements of the video.js player used at the heart of the viewer and fed using Islandora data. Note that users not using a bootstrap theme need to include a custom bootstrap build targeting the oral history elements in order for the viewer to display properly (an example is packaged with the module).

Fig 1 - Viewer configuration by user
Fig 2 - highlighting and CC on video

Fig 3 - available CC options in player

The viewer displays captions in the video.js player using a simple WebVTT file Timed Track Element; transcript cues are parsed and synchronised via HTML5 Video Events and API. A WebVTT file is either provided at ingest, or generated via a source XML file.

When installing and configuring the OHSP, administrators have the option to turn off the creation of WebVTT files, to enable CC for display, to enable the transcript display, and to display the media and transcript side-by-side (rather than the default, which is to display the transcript underneath). Administrators can also configure the speakers and tiers in source
files for display as you can see in Figure 4. At the top, you can see that two custom tiers were used in the source XML for the transcript. Here in the interface, an administrator can define the human-readable name that will be associated in the viewer drop-down menu with the tier provided in the source XML. Similarly, speaker name elements can be glossed with more human readable names using a typical Drupal convention of machine name|human-readable name. This configuration panel is designed to enable users to flexibly define the nature of their source transcript files.

Fig 4 - Configuring Transcript tiers and speaker names for display.

Ingest Formats and Workflow

To serve the functions of the viewer, the OHSP supports the addition of enriched source material and produces additional derivatives on ingest. In addition to a source video or audio file, users ingest time-coded content files in either a) a custom XML format or b) a WEBVTT format.

The XML format is required to support tiers. An XSD schema is provided with the module so that people creating XML files can ensure they are producing a valid format.[5] The custom

---

XML format has a `<cues>` root element containing a number of nested `<cue>` elements. Each `<cue>` consists of a `<start>` and `<end>` element indicating the timecode for each cue, and an optional `<speaker>` element. Each cue can contain multiple customizable tier elements (such as transcription, translation, and annotation). If tiers are used, they must be used in every cue. For instance, if a `<translation>` element exists in one cue, it needs to exist in every other cue even if it is empty. In addition, the cue tiers need to appear in the same order in each cue. The XSD schema does not account for these tier customizations. Tiers are identified via the use of tags when authoring an input XML file. See an example of the required XML format in Figure 5. Further discussion of methods for creating source XML files is provided in the implementation section.

WebVTT is a W3C standard that provides a very simple time-encoded text format to display captions or subtitles. It was developed to support the HTML5 `<track>` element for timed text tracks for closed captions and subtitles in the `<video>` element. Currently, modern browsers and popular HTML5 video players such as video.js only parse simple WebVTT format, which is suitable for simple use cases like closed captioning/subtitles to meet web accessibility requirements. This means that although the WebVTT specification technically allows for the inclusion of structured JSON or XML data snippets, a feature which could have been exploited to capture multiple data tiers within a single WebVTT file, the decision was made to continue with a custom XML format and limit use of WebVTT to closed captions and subtitles in the viewer.

```xml
<cue>
  <speaker>Dorjee Lhatoo</speaker>
  <start>36.0</start>
  <end>45.0</end>
  <transcript> Poor company with me. She would pour me a glass, one sip and it goes right through, you see –</transcript>
</cue>

<cue>
  <speaker>Interviewer2</speaker>
  <start>46.0</start>
  <end>49.0</end>
  <transcript> [indiscernible] on fire. She sounds like great fun.</transcript>
</cue>

<cue>
  <speaker>Dorjee Lhatoo</speaker>
  <start>49.0</start>
  <end>52.0</end>
  <transcript> And then she used to bidi</transcript>
</cue>
```

```xml```
<speaker>Interviewer2</speaker>
<start>52.0</start>
<end>53.0</end>
<transcript> Oh!</transcript>
</cue>

Fig 5- sample snippet of XML

When a source XML file is ingested, the OHSP assigns the TRANSCRIPT datastream ID, and crosswalks the content to an additional MEDIATRACK datastream in WebVTT format. See a sample of WebVTT content in Figure 6.

00:13.480 --> 00:17.017
Anna: It can also be used to carry supplies such as cargo

00:17.017 -->00:25.926
Anna: food, materials or anything that can be carried to the International Space Station and to work up there.

Fig 6 - sample WebVTT format

A flowchart depicting the ingest process is provided in Figure 7.
Support for Ingesting WebVTT

Support for ingesting WebVTT files natively is a relatively new feature. If a single WebVTT is ingested, it will be used to generate a transcript display. Multiple WebVTT files can be ingested and will appear in the Closed Captioning portion of the viewer, but only a single WebVTT file can appear in the transcript viewer itself. There are no current plans to expand this feature based on local use cases, though this may be a useful feature. All WebVTT files ingested with the source media file are stored with the MEDIATRACK datastream ID (DSID). In the case of multiple WebVTT files, a language code (such as _fr) is appended to the end. Supporting WebVTT ingest was a community requirement, but it substantially expands the use of the module for those who may be coming from other systems and who already have a variety of source files containing time-encoded transcripts. For example, WebVTT files can be crosswalked from other popular formats for captioning such as SRT (SubRip).

One shortcoming of the native WebVTT ingest has to do with the inability of Solr GSearch (used by Islandora) to index this file format. Left with the option to cast WebVTT as plain text or XML, the decision was made to create an additional XML derivative datastream for indexing. This datastream has a DSID of INDEXMEDIATRACK_languagecode, with the language code being drawn from the source WebVTT file. Additional development may be
required on this front in order to more fully flesh out this feature and ensure consistency
between oral histories using source WebVTT or XML files.

Our unit is committed to continuing to refine these functions and making every attempt to
ensure backwards compatibility for both UTSC’s local installation and the installations of
others. Existing implementations reveal the flexibility of the software as it is currently being
deployed by the library and in the community.

Implementation Details

Despite the fact that the module has yet to be included in a formal Islandora release, a
recent Islandora listserv discussion revealed active installations at Michigan State University
(US); Hamilton College (US) (The Digital Humanities initiative (DHii)); Italy’s National
Research Council (Consiglio Nazionale delle Ricerche) and the Hagley Museum and Library
(US) \[6\]. This suggests that the initial goal, to develop a tool that was generalized and
flexible in implementation, is at least partially successful. Feedback has included a desire to
streamline the creation and ingestion of transcript files in XML and WebVTT format.
Currently, many desktop tools make it possible to create these files. At UTSC, Inqscribe and
Audacity are common tools, though there has been some exploration of how to use
YouTube to create these files.

The format required for transcripts is simple enough to use any common transcript editor. A
good amount of user documentation has been created to guide people through the process,
and is available in Github.\[7\]\[8\] The UTSC community is currently using Inqscribe\[9\] or
Audacity\[10\] to manually create tracks for ingest. Michigan State reports using
Audiogrep\[11\] to automatically create transcripts, and the Hagley Museum is creating their
timed transcripts in OHMS\[12\]. At UTSC the workflow is to take the outputs of Inqscribe
and Audacity as .csv or text files (depending on the source application) and use
OpenRefine’s\[13\] template export feature to create the appropriate XML file. More detailed
documentation about using these options to create timed transcripts is available in the
module documentation.

The size of source media files has also posed an issue for UTSC’s local installation, and a
PHP script was developed to provide an alternative model for the batch ingest of large media

\[6\] Oct 21, 2016; Subject: Oral Histories Solution Pack as an Islandora component [Internet]
https://groups.google.com/forum/#!topic/islandora/erzU08nDjA0

\[7\] Generating Transcripts [Internet]. Toronto (ON): Digital Scholarship Unit; [cited 2017,
January 2]
https://github.com/digitalutsc/islandora_solution_pack_oralhistories/blob/master/docs/administrator/01
-creatingTranscripts-transcriptionsoftware.md

\[8\] Introduction for Administrators [Internet]. Toronto (ON): Digital Scholarship Unit; [cited
2017, January 2]
https://github.com/digitalutsc/islandora_solution_pack_oralhistories/tree/master/docs/administrator

\[9\] Inqscribe [Internet] https://www.inqscribe.com/

\[10\] Audacity [Internet] http://www.audacityteam.org/

\[11\] Audiogrep [Internet] https://github.com/antiboredom/audiogrep


\[13\] Openrefine [Internet] http://openrefine.org/download.html
files. This script is also available in Github[14]. When using the script, Oral histories are placed in directories and ingested directly into Fedora using a FOXML file template and cURL. Arguments are passed using the command line (example Figure 8)

```bash
php fedora_ingest.php user=fedoraAdmin pass=fedoraPassword
url=http://localhost:8080/fedoraURL ns=demo
cmodel=islandora:sp_videoCModel collection=demo:collection
target=/absolute/path/to/ingest/directory
log=/absolute/path/to/ingest/directory email=admin@example.com
```

Figure 8 - sample command line call to PHP script for alternative ingest path.

At UTSC, two digital scholarship projects and one course are leveraging the solution pack, serving scholars and students in History, Women’s Studies, and Sociology. For classroom use, students are submitting packages that are further prepared and ingested by the course instructor or TA. Typically, the creation of the source XML or WebVTT files is the largest barrier to entry for users of the software. For some time, project members have been requesting the ability to author and publish media transcription, description, and annotation directly within the web interface. Initial work sought to develop these features directly in the Oral History Solution Pack, but this work has recently been abstracted to form the base of a new utility module that aims to facilitate annotation across Islandora repositories.

**Future Directions - Web-based Annotation**

Supporting web-based annotation is a priority feature identified by scholars at UTSC, including scholars that are not actively engaged in Oral History projects. After discussion and research, it was determined that a common approach to annotation across Islandora-hosted content types would present the best option for adhering to the W3C Web Annotation Data Model. The Web Annotation Data Model[15] is the recommendation produced by the W3C Web Annotation Working Group that provides a standard way of dealing with annotations online. In this model, “an Annotation is a rooted, directed graph that represents a relationship between resources” [Web Annotation Data Model]. This model leverages a concept of bodies, targets, and motivations. Below is an example of an Annotation resource, serialized in the preferred JSON-LD format:

```json
"@context": "http://www.w3.org/ns/anno.jsonld",
"id": "http://example.org/anno5",
"type":"Annotation",
"body": {
  "type": "TextualBody",
  "value": "<p>j'adore !</p>",
  "format": "text/html",
  "language": "fr"
}
```

---

[15] https://www.w3.org/TR/annotation-model/
The specification builds upon REST principles and the Linked-Data Platform recommendation, meaning it will natively work with Fedora 4, which is being utilized in the next generation Islandora (Islandora CLAW). In the current Fedora 3 framework, Islandora objects will be created with relationships to the source object. However, the Web Annotation Solution Pack is conceptualized as a CLAW-facing project, in that development assumes a migration of content and features to Drupal 8 and Fedora 4 as the CLAW project becomes production ready. The unit is open to collaboration with others interested in a utility module to serve Islandora viewers and adhere to the emerging Web Annotation Data Model.

**Conclusion**

Open scholarship is best served by generalized software tools that can be configured to serve diverse needs while being collaboratively maintained. The OHSP is an attempt to further and facilitate a conversation about how institutions will support Oral Histories projects. The module furthers UTSC Library’s goal of aiding in the creation of oral histories, and the stewardship of newer forms of scholarship, as well as addressing the realities of the ways that events are witnessed and recorded and how these affect the professional commitments of libraries to preserve a shared cultural heritage.

**Acknowledgements**

Lingling Jiang and Edward Garrett authored much of the original code for the Oral Histories module and Jiang was an author of the OR2015 presentation from which this paper drew some of its material.

**References**


About the Authors

Marcus Emmanuel Barnes is a software developer in the University of Toronto Scarborough Library’s Digital Scholarship Unit. With over eight years of web programming and application design experience, Marcus specializes in creating tools that enhance academic libraries for faculty, staff and students.

Natkeeran Ledchumykanthan is a software developer in the University of Toronto Scarborough Library’s Digital Scholarship Unit. Nat has more than eight years of full stack development experience (LAMP, PHP, Java, Javascript, XML technologies) and is an active volunteer for community digital preservation/library projects.

Kim Pham is the Digital Projects and Technologies librarian in the University of Toronto Scarborough Library’s Digital Scholarship Unit and a liaison librarian for Physics. She works on a bunch of great projects, rock climbs, and loves meticulous metadata review as well as code testing. She also has an awesome dog named Merle, and she is definitely going to see you at the conference.

Kirsta Stapelfeldt is the Coordinator of the University of Toronto Scarborough (UTSC) Library’s Digital Scholarship Unit and liaison librarian for Computer Science. She has been working with Islandora and managing Digital Scholarship/Collections projects since 2010. Previously at the University of Prince Edward Island, Kirsta moved to UTSC in 2014 and maintains active connections with Islandora through participation on the Roadmap Committees and Islandora Board.