The Alveo Virtual Laboratory: A Web Based Repository API

Steve Cassidy[1], Dominique Estival[2], Tim Jones[3], Peter Sefton[2], Denis Burnham[2], Jared Burghold[4]
Macquarie University[1], University of Western Sydney[2], RMIT[3], Intersect NSW[4]

Abstract

The Alveo Virtual Laboratory is an eResearch project funded under the Australian Government NeCTAR program to build a platform for collaborative eResearch around data representing human communication and the tools that researchers use in their analysis. The human communication science field is broadly defined to encompass the study of language from various perspectives but also includes research on music and various other forms of human expression. This paper outlines the core architecture of the Alveo and in particular, highlights the web based API that provides access to data and tools to authenticated users.

Keywords: repository, corpora, tools

1. Introduction

The Alveo Virtual Laboratory is an eResearch project funded under the Australian Government NeCTAR[1] program to build a platform for collaborative eResearch around data representing human communication and the tools that researchers use in their analysis of this data.

This project is motivated by our experience in the earlier Human Communication Science Network (HCSNet), a government funded initiative that brought together researchers from the diverse fields related to the study of human communication. This was a very successful vehicle to promote collaboration across traditional disciplines. One of the issues that emerged from this experience was the difficulty for a researcher from one discipline to apply the tools and techniques of another discipline, or to explore data collected under one paradigm via a completely different analytical perspective. Moreover, research conducted in isolation entails inefficient repetition of analysis of local data sets. Therefore the HCS Virtual Laboratory is designed to provide a platform for easy access to language, speech and other communication-relevant databases and for the integrated use of a range of analysis tools. The hope is that such an environment will not only eliminate the waste of unshared analyses being repeated, but that it will afford the serendipity of new combinations of tools and datasets, facilitating research that will provide new insights into old problems, or the combinations of old ideas to approach new problems. As HCS is such a multi- and cross-disciplinary field, it relies upon various types of data and various tools by which these data can be analysed. The Alveo will enable easy access to shared tools and data, and overcome the resource and access limitations of individual desktop systems. It will allow a diverse range of researchers to access an amalgamation of existing data collections and corpora and to use analytical tools created by other researchers. Providing access to corpora, tools and the analyses conducted with these, into an easily accessed, shared, and replicable environment will not only promote collaboration between institutions and disciplines, but also dramatically improve scientific replicability. It makes it possible to standardise, define, and capture procedures and data output so that research publications can be supported by re-runnable re-usable data and coded procedure.

[1] http://nectar.org.au/

At the time of writing the project is coming to the end of the initial 15 month development phase and will deliver a first production system to the community in the next few months. A further phase of the project will follow on from that where we will learn from the experiences of researchers using the platform to improve the integration with tools and develop a workflow for the ingestion of new data sets.

This paper outlines the core architecture of the Alveo and in particular, highlights the web based API that provides access to data and tools to authenticated users.

2. Sources of Data and Tools

A major goal of the project is to connect tools and data in a way that helps researchers to discover new ways to work and new data to work with. To this end we surveyed the research community to find data sets that could be contributed from different disciplines and tools that could be made available for integration into the platform.

The goals for tools and data were somewhat distinct. Since we have already had some experience with curating and publishing data sets in the earlier Australian National Corpus project (Cassidy et al., 2012), this project was able to build on that foundation, develop a broader set of data collections and solidify the infrastructure for data management. However, with respect to tool integration, the starting point was the diverse experience of the community in building stand-alone tools for particular tasks within a discipline. Therefore, the goal with tool integration was to begin the task of providing a platform and a set of interfaces that could be used in future to provide these tools with access to the data collections. We also wanted to find a way to make complex tools available to a non-technical audience where possible.

2.1. Data Collections

The data being stored on the platform represents a wide range of human communication and has been collected to support research in different disciplines. At the heart of the collection is the Australian National Corpus (AusNC, (Cassidy et al., 2012)). AusNC contains data ranging from texts representing early communications from the Australian colony to modern video recordings of oral history interviews with many samples of text and transcribed conversation also included. To this we have added collections

that represent a number of new disciplines. For example, the PARADISEC collection (Thieberger et al., 2011) which contains audio, video, text and image resources relating to Australian and Pacific Island languages; the ClueWeb12 dataset used in web information retrieval research; and the Macquarie Battery of Emotional Prosody, used in Psychological testing of emotional responses. We have also included a number of new spoken language collections including the recently completed Austalk corpus containing audio recordings of around 1000 Australian speakers (Burnham et al., 2011). The full list of collections is shown here.

1. Australian Corpus of English
2. Australian Radio Talkback Corpus
3. Sample of AustLit Australian Literature
4. The AusTalk Audio Visual Speech Corpus
5. The AVOZES Audio Visual Speech Corpus
6. Braided Channels Video interviews
7. The ClueWeb12 Dataset
8. Corpus of Oz Early English (COOEE)
9. Colloquial Jakartan Indonesian corpus
10. Griffith Corpus of Spoken English
11. International Corpus of English (Australia)
12. Macquarie Battery of Emotional Prosody
13. Mitchell & Delbridge Corpus
14. Monash Corpus of Spoken English
15. The PARADISEC collections
16. Pixar Emotional Music Excerpts
17. Sydney Room Impulse Response collection

The data model that we have developed for the storage of language resources is built around the concept of an *item* which corresponds (loosely) to a record of a single communication event. An item is often associated with a single text, audio or video resource but could include a number of resources, for example the different channels of audio recording or an audio recording and associated textual transcript. Items are grouped into *collections* which might correspond to curated corpora such as ACE or informal collections such as a sample of documents from the AustLit archive (http://www.austlit.edu.au/).

Each collection is described by metadata using a standard Dublin Core description, however the main metadata stored is at the level of *items* within a collection. These are described using a mixed metadata vocabulary based on the descriptions that are provided to us by the original data owners. While the range of metadata fields available varies significantly across the collections, we have mapped the fields to common names where possible using a mixture of Dublin Core, OLAC and custom namespaces. The result is that there are only a small number of fields that will be available for all items in the system but that common names are used where possible to facilitate searching across collections.

As part of the ingest process, each collection is transformed into a standardised format to separately identify item metadata, text (if present) and annotations and encode them in a standard format. For example, some data is supplied as PDF transcripts of audio recordings; these are processed to identify metadata (often supplied in a table at the start of the PDF file) and extract the raw text of the transcript and parse any annotations (such as speaker turns) that might be encoded in the transcript. These 'cleaned' versions of the documents can then be indexed for full text search and stored in the system.

2.2. Tools

The collection of tools we are working with represent a similarly broad range of disciplines and are largely authored by Australian researchers. There are two distinct kinds of tool - those that process data, generating some output for further analysis and those that provide a data manipulation and analysis environment. The goal of the project is to build a platform with interfaces rich enough to support this range of tools so that in future, new tools can be built to extend the range of services that are available to users. In this sense, the tools we are working with now can be seen as a representative sample of the tools used by the research community and certainly not as a final set of tools that will work with the platform.

Examples of data processing tools are the Johnson-Charniak dependancy parser (Charniak and Johnson, 2005), the Python NLTK library (Bird et al., 2009), the DeMoLib video processing library[2] and the signal processing tools included with the Emu system (Cassidy and Harrington, 2000). These tools need to be able to read source data (text, audio, video) from the data store and possibly store analysis results in the form of annotations back in the store. Most of these tools require significant expertise to set up for a particular kind of analysis and one of our goals is to make this easier for non-technical researchers. For example, allowing a linguist to run the Johnson-Charniak parser over a text or using the NLTK part-of-speech tagger.

Interactive analysis tools provide an environment for analysis of audio or text data. The main examples in our collection of tools is the Emu/R package which extends the R statistics environment with tools to handle annotated speech data, and the NLTK toolkit which adds sophisticated text processing capabilities to the Python environment. In both cases our goal is to first provide a useful link between the data sources on our platform into the analysis environment.

3. System Organisation

At the core of the platform is a repository that stores the different parts of the items within a collection (Figure 1). The repository provides a document store for the files associated with an item (audio, video, text and any ancilliary files) and a separate annotation store that manages the annotations on items. Metadata for items is also stored and is indexed along with the full text of textual documents to allow users to select items based on either metadata descriptions or textual content.

Layered on top of the core repository is an access control layer that manages user access to the different collections. As described later, access is granted on a per-collection bases after a user agrees to the licence terms or is granted access by the data owner. This layer mediates all access

[2] http://staff.estem-uc.edu.au/roland/research/demolib-home/

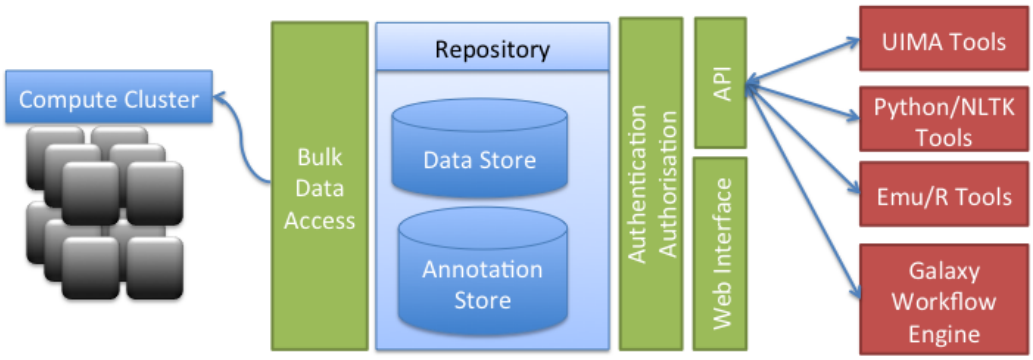

Figure 1: The overall architecture of the Virtual Laboratory

to data in the system through the web front end or the programmatic API.

The user facing layer of the platform is built as a web front end that supports faceted browsing and generalised search to allow a user to find items of interest within the data holdings. The same web application also provides an HTTP based API that supports programmatic access to all of the functionality of the platform.

While access to documents via the HTTP based API is sufficient for many purpose, in some cases the latency of this path is significant when processing large amounts of data. One example is the use of the ClueWeb collection is 389GB of compressed text and 1.95T uncompressed; the way that this is used by researchers is to send the compressed text bundled as WARC archives through an analysis and indexing pipeline. Similarly, the Austalk corpus consists of a large amount of speech data that might be used to train a speech recognition system; in this case the ability to push large quantities of data through an analysis pipeline is most important. To support high bandwidth access to individual collections we are currently exploring a number of methods including raw SSHFS access to the underlying file system and file transfers based on the Aspera[3] file transfer system. An important consideration in implementing this level of access is maintaining the authorisation controls for each collection. We are currently working with our partners to ensure that we can achieve this.

As well as storing all documents associated with an item, textual data is processed to generate a 'plain text' version and associated standoff annotation following the DADA RDF model (Cassidy, 2010). Annotations on audio and video data are also converted to RDF and all RDF annotations are stored in an RDF triple store (Sesame). This approach preserves whatever format of annotations are supplied with the data but puts all primary data and annotations into a single framework to support query and annotation management.

4. The Website

The core data repository is presented via a web based front end that supports a particular workflow for corpus based research. This begins with the researcher using the faceted browse *discovery* interface to identify items suitable for a particular research question. These might be speech recordings of interviews or texts from a particular date range. These items are then saved into an *item list* that is given a name and is stored under the user's account; item lists can be shared and published and have a persistent URL that could in future be used to reference the list in a publication. The contents of an item list can then be used in one of a number of analysis tools. These range from a simple concordance search to more complex analysis using parsers, part of speech taggers, acoustic analysis etc.

An example of this workflow might be a study looking at the use of a particular word over different time-spans. The researcher could make item lists containing texts from each time period and save them both to their account. They can then conduct a concordance search on each item list and compare the results. The texts from each item list could be passed through a word frequency analysis tool to give comparative frequency counts and through another tool to look at collocations of the target words in each period.

Another example that we have implemented as part of the testing of the platform passes an item list containing spoken digit recordings to the HTK toolkit to train an acoustic model for a speech recogniser. A second item list of digits is then used to test the trained recogniser.

4.1. Licensing

By nature, many of the collections that we are dealing with are not able to be made open or released under a liberal licence. In some cases this is due to the original terms under which the data were collected, in others because of cultural or privacy issues relating to the people who were recorded. To deal with this, all access to data on the platform is mediated by an authorisation layer which checks which licence agreements apply to a collection and whether a user has agreed to those terms or been granted access to that collection by the data owner. On registering with the Alveo, a user is able to review the collections that are held and the licence terms under which they are available; if the user agrees to the licence terms they can be granted access to that collection. In some cases, access is mediated through the data owner or a delegate; this allows us to handle more complex licence arrangements such as when a fee is paid to a third party for access or where membership of a particular

[3] http://asperasoft.com/

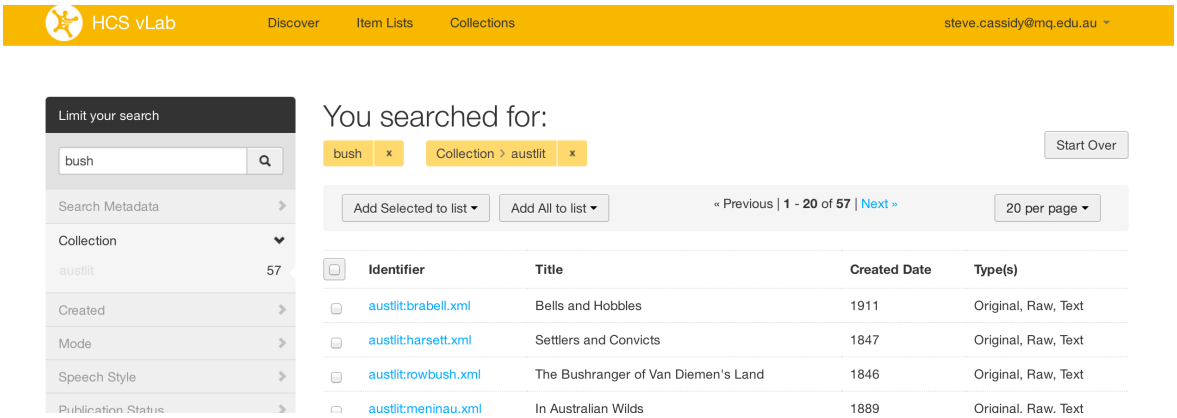

Figure 2: A screenshot of the discover interface showing the results of a search

institution is required.

The licences used with each collection are those provided by the collection owner. Many of the older collections we hold have existing licences. Any new collections are encouraged to adopt an open access licence such as one of the Creative Commons suite.

All access to data is mediated via this authorisation layer including search via the website and all access via the web API. To use the API, a user downloads an authentication token which allows a script to act as a proxy for the user, gaining access to data that they are permitted to access.

5. The Web API

All data and services of the Alveo are made available via a RESTful web API. All entities in the system (collections, items, documents, annotations, etc.) are identified via a URI and, following the principles of Linked Data, that URI resolves to a representation of that entity. HTTP content negotiation is used to determine whether to return an HTML or JSON representation for any URL.

Use of the API requires authentication. This is achieved by a token that is unique to each user and can be downloaded from the website. The token is sent along with every HTTP request using the X-API-KEY header. A user can invalidate their API token via the website if it is accidentally published and a new key can be generated. This mechanism allows a client application to operate on behalf of a user via a relatively simple and reasonably secure mechanism.

5.1. Items and Item Lists

The website allows access to any item lists created by a user via the URL `/item_lists`[4]. By default this returns an HTML page providing access to the item lists but if we add `.json` to the URL or include an Accept header that includes only `"application/json"` then a JSON representation of the item lists is returned. This includes one entry per item list including the URL of the item list itself. The item list has a URL of, for example `/item_lists/91`, and can be retrieved as JSON in a similar manner. An example of the JSON representation of an item list is shown in Figure 3.

[4]Note that the full URL is `http://app.alveo.edu.au/item_lists`.

```
{
  "name": "cooeesample",
  "num_items": 3,
  "items": [
    "http://xx.au/catalog/cooee/2-334",
    "http://xx.au/catalog/cooee/2-339",
    "http://xx.au/catalog/cooee/2-343"
  ]
}
```

Figure 3: An example JSON item list. URLs have been shortened for brevity.

As shown in the figure, the item list JSON representation contains links to the individual items. These can again be retrieved as a JSON document that contains all of the metadata associated with that item along with links to the individual documents and any annotations available for this item. Since the metadata description contains namespaced (RDF) property names we have made use of the JSON-LD format[5] to deliver the description of items. JSON-LD is a recently ratified W3C standard that can is designed to support publication of Linked Data in JSON. Using this standard allows us to define property names like 'dc:title' as fully qualified URIs while maintaining the readability of the shortened version. The metadata descriptions returned from the API are both easy to use as JSON documents and complete as formal descriptions of the metadata properties we are using.

In addition to the metadata properties for an item, the returned JSON-LD description also contains the plain text version of the item if there is an obvious single text associated with it. This is the case for many text collections where one item corresponds to one text document. Including the text with the item description means that applications that just want to process the text need go no further with requests to the API. However, full URLs for each document associated with an item are also included. This might include associated audio and video files as well as original versions of documents as PDF files or even images. Metadata can also be associated with documents in the JSON-

[5]`http://json-ld.org`

LD description of the item, for example the type of document or its extent. Each document URL can be retrieved independently through separate HTTP requests using the API key.

The API also allows the creation of item lists via the API. Creating an item list is done in two stages: first a metadata query is sent to the API via the `/catalog/search` URL, the result of this is a JSON search result containing a list of item URLs. This can then be stored as an item list via a POST request to `/item_lists` URL. Splitting the process in this way allows off-line editing of the list of items before it is stored as an item list for the user.

5.2. Annotation Interchange

Annotations associated with each item are stored in a separate annotation store following the DADA RDF based annotation model (Cassidy, 2010). As part of the ingest process any annotations included with the data are parsed and converted to the RDF format. These are then stored in a Sesame RDF database (triple-store) that forms part of the core repository. In addition to the annotation data, all item and collection metadata is included in the database so that it can be referenced in queries over the annotations.

The DADA annotation model is an RDF model based on the ISO LAF model for annotation interchange. Annotations are represented as a graph based data structure with attributes and values attached to nodes. RDF provides a graph based data model that is a good fit to the annotation graph model. We take advantage of the infrastructure available for storage and query of RDF to build the large scale annotation store required for this project.

Annotations can be accessed via the API using a URL included in the JSON-LD description of an item. This URL by default returns all of the annotations held for an item but can be restricted by query parameters appended to the URL. The JSON-LD format is used to encode the annotations which allows us to use fully qualified names for each property included in the annotation description. These namespace prefixes are defined in a *context file* which is referenced by URL in the JSON-LD document. The example annotation result in Figure 4 shows an annotation on a text document that encodes an annotation of type *icea:text* which is a text unit in the ICE-AU tag set (Wong et al., 2011). The *@type* attribute denotes this as a TextAnnotation and informs the interpretation of the start and end properties as character offsets in the text. Figure 5 shows an alternative example of a SecondAnnotation where the offsets would be interpreted as seconds.

The same JSON-LD format can be used to upload new annotations for an item to the repository. In this case the annotation identifier (the *@id* attribute) is omitted as this will be generated by the system. New annotations are stored in a way that associates metadata about the creator and creation date with the annotations such that these facets can be queried in future and they can be identified as separate from the original annotations on the data. All annotations uploaded become available to all users of the system and will show up in future queries on the annotations.

```
{
  "@context": "http://xx.au/schema/json-ld",
  "commonProperties": {
    "alveo:annotates": "http://...S1A-001"
  },
  "alveo:annotations": [
    {
     "@id": "http://...annotation/3000",
     "@type": "dada:TextAnnotation",
     "dada:start": "0",
     "dada:end": "57",
     "dada:type": "icea:text"
    }
  ]
}
```

Figure 4: An example JSON-LD format for annotations. URLs have been shortened for brevity.

```
{
  "@id": "http://.../annotation/233356",
  "@type": "dada:SecondAnnotation",
  "start": "0.76",
  "end": "1.08",
  "label": "}:",
  "type": "maus:phonetic"
},
```

Figure 5: An fragment of JSON-LD format for a phonetic annotation on an audio file. URLs have been shortened for brevity.

5.3. SPARQL Endpoints

The API provides a high level interface to query and create new annotations but since annotations and metadata are stored as RDF we also provide a low-level SPARQL endpoint to allow developers to query the raw form of the annotation data store. To allow appropriate access control, the annotations and metadata for each collection is stored in a separate Sesame store. The URL of the SPARQL endpoint for each store is included in the metadata for the collection. SPARQL queries can be sent using the standard SPARQL protocol[6] with the addition of the X-API-KEY header required in all API calls.

5.4. Client Libraries

The design goal of the API is to enable new services to be built using the facilities of the core Alveo platform. As part of this project we are implementing interfaces with tools that make use of Python (NLTK), R (Emu), Matlab, Java (UIMA) and Go environments; part of this implementation will be a set of interface libraries for these languages that provide a programmatic API for Alveo.

5.5. Performance

This section presents some basic benchmark timing information for parts of the API to illustrate the performance that we are able to achieve from the system. There is an

[6] `http://www.w3.org/TR/rdf-sparql-protocol/`

obvious cost in the layers of processing that we place between the data and the user to implement the services the platform provides. If this cost is too great the user will be inclined to just download the data directly and work on it locally; however, if the overhead is not significant relative to the task being performed, then the idea of a centralised repository may gain some traction.

The first set of timings relate to the creation of item lists from a list of item URLs retrieved from an earlier query. The overhead in this operation involves creating the indexes to allow subsequent queries over these item lists.

Items Added	Time in seconds
300	4.4
600	8.9
1000	15.0

The second benchmark looks at retrieval of an item list; this operation retrieves only the JSON representation of the item list, that is just the list of item URLs for each item.

Items Retrieved	Time in seconds
300	0.47
600	0.69
1000	0.90

The final benchmark measures the time to retrieve the actual documents over the web API. Here the overhead of HTTP and the authentication layer in the system are included. These are obviously significant with smaller files but the larger files show a better throughput of around 5Mb/s.

Size	Time in seconds
375k	0.8
5M	4.8
19M	4.2
340M	58.8
700M	119.4

The overhead of downloading individual files led to our implementation of an API call to download all documents associated with an item list as a ZIP or WARC archive. These allow larger scale compression of the returned data stream and we are able to achieve significant improvements in download speeds compared with individual downloads.

6. The Galaxy Workflow Engine

Galaxy (Goecks et al., 2010) is a web based workflow system originally designed for use in the life sciences to support researchers in running pipelines of tools to manipulate data. We have adapted Galaxy to the Human Communication Science domain to allow it to run tools based on some of the contributed tools in our project. Galaxy is able to run any Python script as a tool and we have packaged a number of example NLTK based scripts that carry out tasks such as part-of-speech tagging, stemming and parsing. We have also used Galaxy to implement tools in R and in Matlab. A data acquisition tool can use the web API to retrieve data for an item list and pass it to subsequent tools for analysis.

Using Galaxy a researcher is able to design a workflow that combines data acquisition with the application of one or more analysis tools and visualisation of the output. While the number of tools we have implemented so far is small, we are encouraging researchers to develop and contribute new tools to extend this platform.

Workflows in Galaxy can be stored, shared and published. We are hopeful that this may become a way to codify and exchange workflows for common analyses of human communication data.

7. Summary

This paper has described the development of the HCS Virtual Laboratory project and in particular the API that allows access to the data and services provided by the platform. The virtual laboratory consists of a repository that stores text, audio, video and associated documents and provides a separate annotation store making use of an RDF based model of annotations. The API allows programmatic access to the data on the platform so that interfaces can be built for common research tools used to process this data. Annotations for individual items can be queried and are delivered as JSON-LD documents; the same format can be used to upload new annotations generated by analysis of the data.

Alveo (`http://alveo.edu.au`) will enter production in June 2014 at which time it will be open to any user around the world to use for research on human communication data. The software implementing the platform is available under an open source licence including a fully tested procedure for setting up a new instance of the virtual laboratory. We also invite authors of tools to explore the API for interfacing their tools to the repository.

8. References

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python - Analyzing Text with the Natural Language Toolkit*. O'Reilly Media.

Burnham, D., Estival, D., Fazio, S., Cox, F., Dale, R., Viethen, J., Cassidy, S., Epps, J., Togneri, R., Kinoshita, Y., Gcke, R., Arciuli, J., Onslow, M., Lewis, T., Butcher, A., Hajek, J., and Wagner, M. (2011). Building an audio-visual corpus of australian english: large corpus collection with an economical portable and replicable black box. In *Proceedings of Interspeech*, Florence, Italy.

Cassidy, S. and Harrington, J. (2000). Multi-level Annotation in the Emu Speech Database Management System. *Speech Communication*, 33:61–77.

Cassidy, S., Haugh, M., Peters, P., and Fallu, M. (2012). The australian national corpus : national infrastructure for language resources. In *Proceedings of LREC*.

Cassidy, S. (2010). An RDF Realisation of LAF in the DADA Annotation Server. In *Proceedings of ISA-5*, Hong Kong, January.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics.

Goecks, J., Nekrutenko, A., Taylor, J., and Team, T. G. (2010). Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computa-

tional research in the life sciences. *Genome Biology*, 8(11).

Thieberger, N., Barwick, L., Billington, R., and Vaughan, J., editors. (2011). *Sustainable data from digital research: Humanities perspectives on digital scholarship. A PARDISEC Conference*. Custom Book Centre.

Wong, D., Cassidy, S., and Peters, P. (2011). Updating the ice annotation system: tagging, parsing and validation. *Corpora*, 6(2):115–144.