From student hard drive to web corpus (part 2): the annotation and online distribution of the Michigan Corpus of Upper-level Student Papers (MICUSP)

Matthew Brook O'Donnell[1] and Ute Römer[2]

Abstract

This paper continues the detailed account of the central steps involved in compiling and distributing the Michigan Corpus of Upper-level Student Papers (MICUSP). In this paper, we discuss the annotation process used to encode MICUSP files in TEI-compliant XML, and the development of MICUSP Simple, the online application through which the corpus is now freely available online. We also describe how MICUSP Simple can be used to carry out simple word/phrase searches and to browse papers within different categories.

1. Introduction

The Michigan Corpus of Upper-level Student Papers (MICUSP) is a new corpus of proficient student academic writing samples developed at the English Language Institute of the University of Michigan, Ann Arbor. This paper is the second of two that describes the process of developing MICUSP from conception to the initial release through an online interface (see Römer and O'Donnell, 2011).

Each of the 829 papers in MICUSP has been marked up in TEI-compliant XML and maintains the structural divisions (sections, headings, paragraphs) of the original paper as well as a typology for quotation marks and emphatic features (e.g., italics and underline). A file header that has been added to each MICUSP file includes, among other things, information about these text types, the discipline and the student's level, native-speaker status, and sex, which makes it possible to carry out customised searches

[1] RCGD, Institute for Social Research, University of Michigan, 426 Thompson Street, Ann Arbor, MI 48106, USA.
Correspondence to: Matthew Brook O'Donnell, *e-mail*: mbod@umich.edu
[2] Department of Applied Linguistics and ESL, Georgia State University, 34 Peachtree Street, Suite 1200, Atlanta, GA 30303, USA.

Corpora 2012 Vol. 7 (1): 1–18
DOI: 10.3366/corp.2012.0015

www.eupjournals.com/cor

in subsections of the corpus, (e.g., only in Psychology reports written by native-speaker first-year graduate students). In December, 2009, the corpus was made freely available to the global research and teaching community through a user-friendly online search and browse interface (MICUSP Simple) that allows for customised searches, either in the entire corpus or in subsets of it.[3] This paper discusses the steps that were central to the annotation and online distribution of MICUSP.

2. Corpus conversion, markup and annotation

For MICUSP, we accepted papers in Microsoft Word, PDF, and text-based formats. Following common corpus encoding and markup practice we chose to encode the corpus files using the Unicode UTF-8 character encoding (see McEnery and Xiao, 2005) and we used eXtensible Markup Language (XML) for the markup and annotation. In line with many corpus projects, we adopted the Text Encoding Initiative (TEI) guidelines in the design of the XML schema for the MICUSP files to capture both structural elements of the student papers and the associated metadata (Burnard, 2005). We plan to make MICUSP available in both full XML and plain text (i.e., with no annotation or metadata). While for each of the original file formats (e.g., Word, PDF) it might have been possible to make use of the formatting codes and styles in the original document (as demonstrated in Ebeling and Heuboeck, 2007), we decided to transform all the original files into plain text Unicode at the start of the conversion process in order to achieve consistency in the subsequent steps.

2.1 Conversion process

The diagram in Figure 1 illustrates the conversion process that takes each uploaded paper file, and the associated student and paper metadata, submitted through the online submission system through to the final TEI-compliant XML form.[4]

Step 1a. The information about the student (e.g., demographics, disciplinary programme and language background) and their paper(s) which was entered through the web interface and stored in the MySQL database is exported into two tab-delimited text files. Each of the papers has an integer value acting as its unique identifier. Identifiers are generated for each paper with

[3] The MICUSP Simple interface can be found at: http://search-micusp.elicorpora.info/

[4] Various tools were used in this process, including a series of Perl scripts written by Gregory Garretson. He also developed the first version of the MICUSP DTD (Document Type Definition) using the TEI Pizza Chef tool (see: http://www.tei-c.org/pizza.html). We are grateful to Gregory for his skilful work and clear documentation that enabled consistency in the conversion process, even during a number of transitions in project personnel.

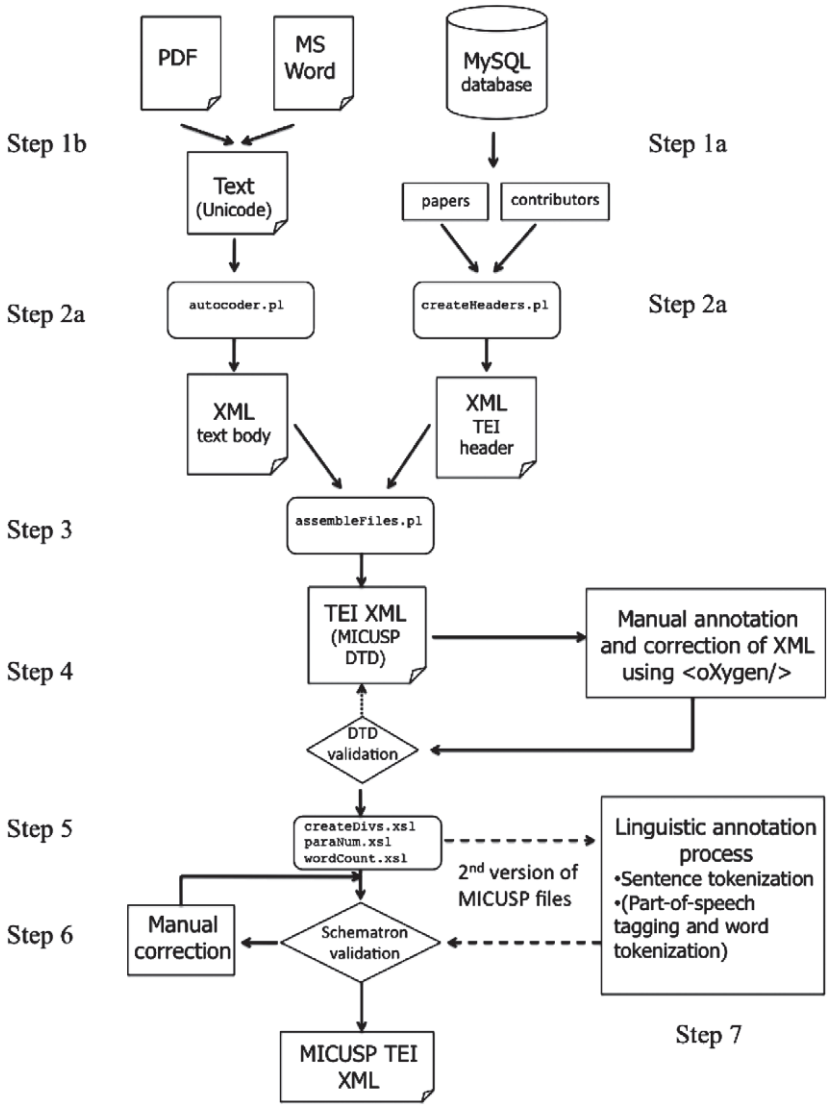

Figure 1: Schematic representation of MICUSP file conversion and markup process

the format: [three letter discipline code].[student level].[student identifier within discipline].[paper number for that student]. For example, BIO.G0.05.1 indicates that this is a Biology paper, written when the student was a final-year undergraduate (G0), that they are the fifth Biology student to be processed and that this is their first paper.

Step 1b. When the student uploaded the original file of their paper, it was assigned a filename beginning with the integer identifier in the matching

```
<profileDesc>
       <creation>Paper submitted to instructor in Apr
2005</creation>
       <particDesc>
              <person id="P75" sex="f" age="TwentyToTwentyThree">
                     <affiliation>Biochemistry</affiliation>
                     <firstLang>English</firstLang>
                     <dominantLang>Mandarin</dominantLang>
                     <englishInPrimarySchool value="YES"/>
                     <englishInSecondarySchool value="YES"/>
                     <englishInUndergraduate value="YES"/>
              </person>
       </particDesc>
       …
</profileDesc>
```

Figure 2: Part of a MICUSP file header

row of the papers table in the database. At this stage, the file is renamed with the full identifier code assigned in the previous step. The text from original student papers is extracted and saved in a Unicode text file: for Word files this is done in a copy of the file, and for PDF documents we used either the 'Save as text' function, cut and paste, or an online PDF to Word convertor.[5] Non-textual and data features including tables, figures, graphs and complex appendices are deleted and replaced with gap tags (e.g., < gap desc = "table"/ >). Each file is then checked using a Perl script (checkCharacters.pl) that identifies and reports on any unusual characters such as a smiley face and other icons or formatting codes from the original format or character set. These characters are usually noted and removed during the manual annotation stage (see Step 4) using the XML editor.

Step 2a. The next step combines the two metadata text files exported in Step 1a and generates an XML file for each paper which consists of a TEI header with the different values assigned to the appropriate elements. For instance, the XML fragment in Figure 2 shows part of the profileDesc element for the paper BIO.G0.05.1, encoding the metadata about the student. A Perl script called createHeaders.pl is used to create the TEI Header.

Step 2b. Each of the Unicode text files from Step 1b are then processed by the Perl script autocoder.pl, which analyses the paper for structural elements and transforms the text into the appropriate TEI XML for the text element (see Figure 3). It applies a three-part structure to the paper consisting of: (*i*) an opener (title, subtitle, abstract and table of contents); (*ii*) the main body of the paper; and (*iii*) a closer (references/bibliography, notes and appendices). The script assumes that each paper contains these three components, and

[5] We found the tool PDF to Word™ (see: http://www.pdftoword.com/) was able to produce very good results extracting the text of PDF files where their format was particularly complex (i.e., double column format papers or richly illustrated engineering reports) or where the PDF had been generated as an image so that the text could not easily be extracted in the PDF reader.

```
<text>
        <body>
            <div type="opener">
                <head>Mn (III) TPPS4: A metallophorphryin used
                       for tumor identification in MRI</head>
                <p/>
            </div>
            <div type="main">
                      <p>From as early as 1948, scientists have
                      studied… </p>
                      <p rend="head_1">Synthesis</p>
                    <p>MnTPPS4 can be synthesized by … </p>
                      <gap desc="formula"/>
                      …
                       <p>Tetraphenylporphine sulfonate, …
                          in propionic acid.<ptr type="footnote"
                       target="fn6"/>
                    </p>
            </div>
             <div type="closer">
                <div type="footnotes">
                      <note type="footnote" id="fn1">1 Figge, F.;
                      Weiland,
                      G.; Manangiello, L. Proc. Soc. Exptl. Biol.
                      Med. 1948, 68, 640-641.</note>
                    …
                </div>
            </div>
        </body>
</text>
```

Figure 3: Main sections of the XML markup for the body of a MICUSP paper

it uses a series of keywords and regular expressions to match commonly used headings for these components (e.g., Abstract, Introduction, References, Works Cited and Appendix). The script attempts to make use of as many cues of as varied a kind as it can to parse the structure of the paper. It will, for example, try to sense the presence of sections and subsections, and the numbering scheme (1, 1a, 2, 2a; I, II, III; 1., 1.1., 1.2.3; *etc*.) used. We found that it performed remarkably well for the vast majority of MICUSP papers. However, where there were systematic errors caused by missing a cue or by the use of an unusual section heading (e.g., 'THIS TEXT REFERS TO:' used as the heading for the bibliography), tags (such as References-MICUSP, Appendix-MICUSP, not-a-footnote-MICUSP) can be added at the appropriate location in the text file to guide the autocoder script.

The autocoder script also recognises straight and curly single and double quotation marks, and it marks them with an XML element. We wanted to be able to distinguish between text written by the student and cited text in quotations. Also this markup facilitates the study of how quotes are used in student writing (Ädel and Römer, 2012) – for emphasis or to highlight or 'problematise' a term using so-called 'scare-quotes'. This is discussed in more detail in Step 4 and under Section 2.2.

Step 3. The outcome of Steps 2a and 2b is two XML files for each MICUSP paper, one containing the header and the other the body text of the paper. A further Perl script, assembleFiles.pl, combines these files into a single TEI XML file.

Step 4. The most involved and time-consuming step in the conversion process is manual correction and annotation. We found that the < oXygen/ > XML editor[6] worked particularly well with our conversion workflow where a number of research assistants were working concurrently on MICUSP files stored on a shared network drive. The annotator opens an unchecked MICUSP XML file in the XML editor and also the original paper file (Word or PDF) for comparison. < oXygen/ > will quickly give the annotator feedback as to where problems exist in the file as the file is automatically validated using the MICUSP DTD. The kinds of errors discovered relate primarily to missing elements (e.g., if one of the three main sections of the body is not found, or if required values are missing in any of the header elements). As each error is fixed, the XML is revalidated and the remaining errors list is updated. This helps the annotator to gauge how much work remains to be done. Alongside correcting XML element placement, ordering and absence to ensure validity against the MICUSP DTD, annotators also work with a checklist of items. This includes checking against the original document (in PDF form) to ensure the section divisions have been recognised, and that lists, examples and block quotations have been converted to the appropriate TEI XML elements. Where there are various levels of nesting in the sections, (e.g., 2., 2.1, 2.1.3), a rend attribute is added to the < p > element around the heading – for example, < p rend = 'head_3' > 2.1.3. Experiment 3 < /p > signifies a level 3 heading. The most 'involved' component of the manual annotation stage is the functional classification of quotation marks, and restoring other typological elements, namely the use of bold, italic and underline, where the student uses them to introduce terms, mark titles and so on, that were lost in the conversion to text. This process is discussed in more detail under Section 2.2.

The amount of time required to carry out this step varied considerably depending on the complexity of the paper – for instance, the use of diagrams, tables, formulas, along with the number of quotation and emphatic devices used. The discipline and the paper type are both factors in this complexity.

Step 5. After completion of the previous four steps, each paper was a well-formed and valid XML document. We then applied a series of XSLT stylesheets to automatically enhance each of the files. When used properly, XSLT minimises the risk of losing structure or, worse, parts of a document because you have not anticipated them. Among the transformations used for the MICUSP XML are the creation of hierarchical

[6] See: http://www.oxygenxml.com/

Flat XML representation after manual annotation (Step #4)

```
<div type="main">
     <p rend="head_1">Part 1a: Lesson
             Plan Analysis</p>
     <p rend="head_2">Criterion 1:
             Classroom Management</p>
     <p rend="head_3">How will
             students get materials?
     </p>
     <p>The lesson plan lists ...</p>
     <p rend="head_3">How will the
         teacher call the ...</p>
     <p>The lesson plan omits ...</p>
     <p rend="head_3">How will
           students be monitored?</p>
     <p>The lesson plan does ...</p>
     <p rend="head_2">Criterion 2:
          Establishing a Sense ...</p>
     <p>Students must unders...</p>
     <p rend="head_3">Does the lesson
              help teachers...</p>
     <p>The lesson does a nice ...</p>
     ...
</div>
```

Hierarchical XML representation after transformation using createDivs.xsl (Step #5)

```
<div type="head_1">
        <head>Part 1a: Lesson
              Plan Analysis</head>
        <div type="head_2">
            <head>Criterion 1: Classroom
                  Management</head>
            <div type="head_3">
                <head>How will students
                    get materials? </head>
                <p>The lesson plan lists
                    ...</p>
            </div>
            <div type="head_3">
                <head>How will the teacher
                      call the ...</head>
                <p>The lesson plan omits
                    ...</p>
            </div>
            <div type="head_3">
                <head>How will students be
                      monitored?</head>
                <p>The lesson plan does
                    ...</p>
            </div>
        </div>
        <div type="head_2">
            <head>Criterion 2:
                 Establishing a ...</head>
            <p>Students must unders...</p>
            <div type="head_3">
                <head>Does the lesson help
                   teachers ...</head>
                <p>The lesson does a nice
                    ...</p>
            </div>
        ...
        </div>
...
</div>
```

Figure 4: Part of a MICUSP XML paper before and after the application of script to add hierarchical structure to sections and subsections

divisions (< div > elements) for the sections and subsections (see Figure 4) section, adding word counts and the numbering of paragraphs.

Step 6. The MICUSP DTD works well to ensure that only the appropriate MICUSP XML elements are used and that they are used in appropriate contexts. However, an XML DTD is very similar to a context-free grammar that might be used in formal syntactic analysis or by a parser, and has some constraints on what kinds of patterns and structures can be mandated. Further, it is not possible to enforce requirements for the content of elements easily (e.g., a < head > element should begin with an uppercase letter). We have experimented with additional schema technologies and other techniques to carry out the further checks that were needed, and we found Schematron to be a particularly useful tool.[7]

[7] Schematron uses an assertion-based test–fail approach based upon XPath. See van der Vlist (2007) for more detail, and see also the Schematron website, at: http://www.schematron.com/

> Grice divides implicatures into the following general categories: '**conventional**'[A] and '**non-conventional**'[B], A conventional implicature can be derived from the literal meaning of the speaker's words. For example, consider the following sentence: '**Bobby quit smoking**'[C]. The meaning of the word '**quit**'[D] results in the implicature that Bobby *had* a smoking habit (at some time prior to the utterance of the aforementioned sentence). Grice is not concerned with this type of implicature, but rather, focuses on a specific kind of non-conventional implicature, which he names '**conversational implicature**'[E] (p. 167). Grice's definition of '**conversational implicature**'[F] is expressed in relation to his '**Cooperative Principle**'[G] and to a series of maxims that characterize typical conversation.

Figure 5: Paragraph from a MICUSP paper (PHI.G0.03.1), exemplifying different functions of quotes (text version)

Step 7. The MICUSP files used in the MICUSP Simple web interface (see Section 3) have been processed up to Step 6. We have experimented with further kinds of processing to add linguistic annotation to the corpus, including sentence and word tokenisation using the Natural Language Toolkit (NLTK; Bird *et al*., 2009) and part-of-speech tagging and lemmatisation using TreeTagger (Schmid, 1994) and the Genia tools (Tateisi and Tsujii, 2004). O'Donnell and Römer (in preparation) make use of the sentence tokenised version of MICUSP to explore the distribution of phraseological items across various textual positions. Future releases of MICUSP may include these more richly annotated versions.

2.2 Annotating quotes and other emphatic typography

In the design of the annotation scheme we identified a potential case in which it would be desirable to be able to distinguish between the student's writing and quoted text. Such distinctions are particularly important when extended quotations have been used. To achieve this, it was necessary to examine text marked with quotes and attach a type attribute. As we examined more instances of quotation, we discovered more varied usage than we originally anticipated. Consider, for example, the paragraph from a Philosophy report 'Grice's Analysis of Metaphor and Irony' (PHI.G0.03.1) under Figure 5; in particular, note the use of quotation marks (emphasis added).

There are seven sets of quotations marks in this one paragraph. Of particular interest is the way in which the same typographical device is used for different functions in close proximity. The first two uses (A and B) are

```
<p>
        Grice divides implicatures into the following general
        categories:
        <q type="term">conventional</q>
        and
        <q type="term">non-conventional</q>
        .  A conventional implicature can be derived from the literal
        meaning of the speaker's words.  For example, consider the
        following sentence:
        <q type="example">Bobby quit smoking.</q>
        The meaning of the word
        <q type="example">quit</q>
        results in the implicature that Bobby
        <i>had</i>
        a smoking habit (at some time prior to the utterance of the
        aforementioned sentence).  Grice is not concerned with this
        type of implicature, but rather, focuses on a specific kind of
        non-conventional implicature, which he names
        <q type="quote">conversational implicature</q>
        (p. 167).  Grice's definition of
        <q type="term">conversational implicature</q>
        is expressed in relation to his
        <q type="term">Cooperative Principle</q>
        and to a series of maxims that characterize typical
        conversation.
</p>
```

Figure 6: Paragraph from a MICUSP paper (PHI.G0.03.1), exemplifying different functions of quotes (XML version)

technical terms. In the third instance (C), the quotes surround an example sentence, part of which is restated in D. The fifth (E) and sixth instances (F) surround the same term 'conversational implicature', but are arguably used in slightly different ways. Instance E is a quote on account of the page reference and the use of the verb, *name*. In contrast, the next two instances (F and G) are used to mark out the technical terms, 'conversational implicature' and 'Cooperative Principle'. Figure 6 shows how these analyses of the function of quotes are encoded in the XML using type attributes on the $<q>$ elements.

We expanded the list of types for quotation marks to include the following:

- title (book title, title of organisation, title of place, person, *etc*.);
- term (scientific term, discipline-specific terminology, *etc*.);
- example (linguistic examples);
- quote (quoted material or anything not originating from the student author AND has a citation or reference to source or origin of the quoted material);
- soCalled (scare-quotes); and,
- studentVoice (questions posed, and thoughts or comments made by the student author).

Table 1 contains examples of each of these types from a selection of MICUSP papers. They demonstrate the wide range of usage for the

same typographic device and illustrate the involved nature of the manual annotation (Step 4) described under the previous section. Future analyses will explore the quantitative distribution of the different quotation mark types.

Table 2 shows the distribution of quotation mark types for the whole corpus and in four selected disciplines. The dominant use of quotation marks by MICUSP writers is for quotation from other sources, although students of Mechanical Engineering appear to use them less in this way and more for indicating terms. The soCalled category is second, overall, but there are disciplinary differences. Psychologists are particularly fond of such 'problematisation', while in English the reference to literary works accounts for the high proportion of the use of titles.

3. MICUSP search interface: design, usability testing and functionality

We determined that there are two key target groups for MICUSP: (*i*) EAP/ESL teachers and learners and (*ii*) those who are conducting corpus linguistic research. The second group are likely to be somewhat sophisticated corpus users who are either proficient in the use of existing corpus analysis tools or develop their own software for analysis. Such researchers are less likely to want to be constrained to a web interface and their needs are best met through the release of the corpus files in XML and text formats (see Römer and O'Donnell, forthcoming). Many EAP teachers and learners, on the other hand, are unfamiliar with, or at least inexperienced in, corpus analysis and the use of corpus tools. So it was decided that the first release of MICUSP would take place through an online interface designed specifically to be useful to EAP teachers and learners. We designed an interface, MICUSP Simple, with a focus on simple browsing and search functionality that supports the tasks we anticipated would be most commonly needed by users of MICUSP in an EAP context. In teaching and studying academic writing, the focus is less on the extract of large numbers of short examples, as might be presented using a standard KWIC display, and more on being able to locate good examples of particular types of writing in specific disciplines (e.g., Biology reports or project proposals in Mechanical Engineering).

3.1 MICUSP Simple development and architecture

Before beginning development of MICUSP Simple, we considered some of the web-based corpus frameworks that are available and might be suitable for hosting MICUSP, such as SketchEngine,[8] Xaira[9] and the various options

[8] See http://the.sketchengine.co.uk/ and Kilgarriff *et al*., (2004). The British Academic Written English (BAWE) corpus is available as one of the free access corpora in the SketchEngine.

[9] See: http://xaira.sourceforge.net/

Paper ID	Example	Quote type value
POLG0.23.1	The '**Creationism Act**' forbids the teaching of the theory of evolution in public elementary and secondary schools unless accompanied by instruction in the theory of '**creation science**.'	title, title
NRE.G1.06.1	There are two key components to the Conservation Framework. The first is the '**balance easement**' of 90,000 acres required by the concept planning process to be contributed by Plum Creek to balance out the residential development included in the plan.	term
BIO.G3.02.1	I would add that these details do not have to be small, since we are not always studying '**global**' quantities.	soCalled
BIO.G1.07.1	After that, we can trace the movement of PGCs (from mid-gastrula stages, after the expression of GFP), and we can address the question about '**who doesn't move?**'	studentVoice
POL.G0.43.1	As a nation, the Jewish people told themselves '**never again**.'	studentVoice
PHI.G0.03.1	For example, if I say '**Juliet is the 10th planet**', it is unclear whether I mean something like '**Juliet is distant**', or '**Juliet is elusive**', or '**Juliet is sought by many**', *etc*.	example, studentVoice (x3)
POL.G0.43.1	Anatol Lieven, author of *America Right or Wrong*, points to the Cold War as a turning point in which an ideological threat motivated '**permanent mobilization**.'	quote
PSY.G0.42.2	To illustrate their mindset, the Lafayette would say '**if I grow up**' to talk about future career goals, instead of the tradition children's saying '**when I grow up**' (1991, preface).	quote, quote
POL.G0.47.1	'**I am George W. Bush, and I approve this message**.' Unlike most other political ads put out by candidates running for presidential elections, this is how the '**Safer, Stronger**' ad began its 30 seconds spot, which first aired on March 3rd, 2004.	quote, title

Table 1: Examples of different quote types from MICUSP papers

Type	MICUSP	BIO	MEC	PSY	ENG
quote	8,694	71	13	644	2,689
soCalled	1,531	39	19	233	125
term	1,088	15	30	121	186
title	700	13	4	46	241
example	503	9	0	20	24
studentVoice	189	0	1	41	5
unclear	35	0	0	0	11
	Σ 12,740	Σ 147	Σ 67	Σ 1,105	Σ 3,281

Table 2: Distribution of types for quote elements

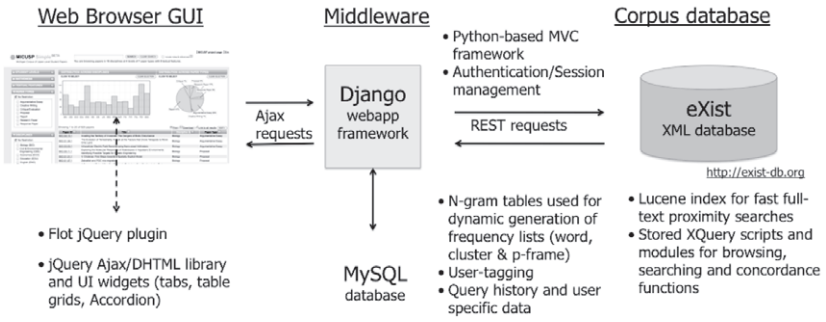

Figure 7: Architecture of MICUSP Simple web application

based upon the IMS Corpus Workbench.[10] The focus of these, however, is towards providing functionality for a corpus researcher, and the interfaces are not always intuitive to a 'non corpus user'. We chose, therefore, to design our own interface and to make use of recent advances in web technologies, particularly in terms of interactive features that attempt to mimic desktop applications in a browser and move away from the page-to-page structure of a traditional web application.

The technical details of the implementation are beyond the scope of this paper and likely to be of only limited interest to most readers. Figure 7 illustrates the three-tiered architecture we utilised. To get the benefit of the XML structure of the corpus, we made use of the eXist XML database[11] to store and index the MICUSP files. eXist supports the XQuery language that allows both for complex queries based on specific XML elements and pattern structures to be carried out, and the construction of entire web-applications.

[10] See: http://cwb.sourceforge.net/. A number of web corpus tools have been developed using this platform, including BNCWeb (see Hoffmann *et al*., 2008) and CQPWeb (see Hardie, submitted).
[11] See: http://exist-db.org

It is well suited to two-tier web applications where browser-based JavaScript (using AJAX) would interface directly with the database. However, in order to ensure robustness and to allow for future expansion and the integration of other datasources (e.g., a relational database for certain types of query and data analysis), we chose to use Django, a Python web framework, to marshal requests from the browser and queries in eXist.

On the browser side we made use of the jQuery library and a number of plugins to provide rich graphical interfaces (e.g., histograms and piecharts) and to reduce the complexities involved in supporting a range of browsers, including Internet Explorer, Firefox and Safari.[12]

3.2 MICUSP Simple usability study

Once we had a prototype implementation of MICUSP Simple in place, we carried out a usability study with a group of potential MICUSP users, including EAP teachers, students and researchers. Users were asked to carry out a number of tasks such as:

- Find all reports in History, Biology, Civil and Environmental Engineering, and Philosophy.
- Find all papers that use the word *therefore*.
- Find all Psychology papers that are reports that use the phrase *in terms of*, and find a paper that uses this term more than once.
- Find the second instance of this term. Sort the results alphabetically according to title.

The study revealed that most users had little trouble completing basic tasks that they would normally perform using other online corpus tools. The task that gave users the most trouble was finding a paper by students of a specific level (G0, G1, *etc*.). Similarly, users had some difficulty sorting results alphabetically, and finding papers with more than one instance of a search term. Many of the testers thought the presentation of the application could be improved and simplified. In response to the findings of the usability study, we made a number of changes to the MICUSP Simple interface; for example, we rearranged the layout of the user interface by moving the feature selection checkboxes to the left hand side of the screen and added a show/hide accordion widget so that not all options were visible by default. We also added mouseover events that produced popup information messages to explain the meaning of various components.

---

[12] For information on the jQuery JavaScript library see http://jquery.com/. The interactive graphics used in the MICUSP Simple interface rely on a modified version of the Flot jQuery plugin (see: http://code.google.com/p/flot/).

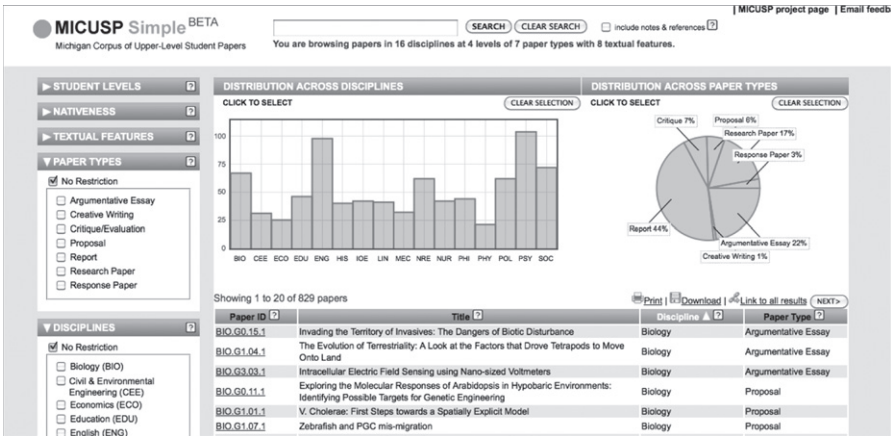

Figure 8: Browsing MICUSP papers using MICUSP Simple

3.3 Using MICUSP Simple

The online interface to MICUSP is available for open access.[13] The two basic functions are: (*i*) browsing papers, and (*ii*) simple word and phrase searches with results displayed with a paragraph of context.

3.3.1 Browsing papers using MICUSP Simple

MICUSP Simple allows users to browse the papers in the corpus according to a series of categories, by interacting with the graphs and by using the checkbox selections on the left hand side (see Figure 8). Each of these feature sets can be hidden or expanded by clicking on the header bar for the feature. For example, under Figure 8, the checkboxes for student levels, nativeness and textual features are hidden, while those for disciplines and paper types are visible. Selecting one or more of the departments from the histogram graph or the left-hand select list will allow users to view just papers belonging to those disciplines. The histogram and the discipline checkboxes are linked so making (de-)selections on one will be reflected immediately in the other. The paper type checkboxes and the clickable pie chart are also linked.

The screenshot under Figure 9 shows two disciplines, English (ENG) and History and Classical Studies (HIS), selected (see #1). Two messages show the current selections and the number of papers in the corpus that match these selections (see #2). The list of papers that match the user's selection are shown in the table in the bottom half of the screen. The paper identification code, title, discipline and paper type are given for each paper. If the mouse

[13] See: http://search-micusp.elicorpora.info/

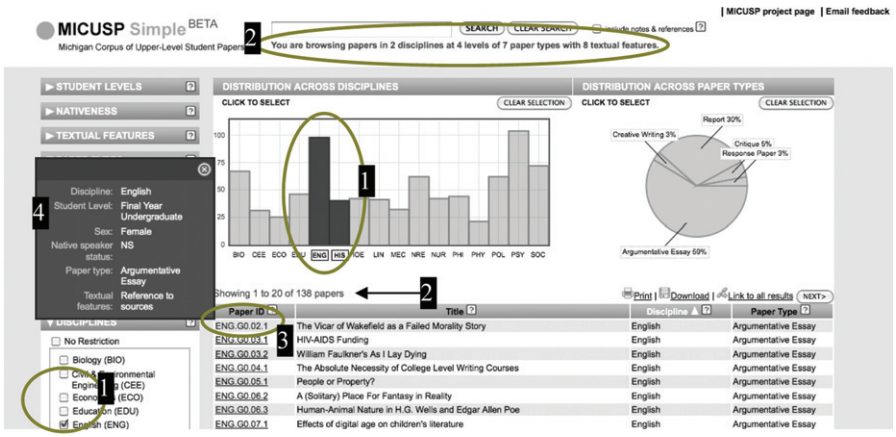

Figure 9: Browsing papers from selected disciplines using MICUSP Simple

pointer is made to hover over the paper identification code (see #3), a popup box will appear that provides further information about the paper and its author (see #4).

Users can export their browse results in different formats by clicking one of the three links, 'Print', 'Download' and 'Link to all results', found above the results table. Clicking on the paper identification code will open up another tab or window in the browser and display the complete paper and related information. Users can also look at the original version of the paper in PDF format using the 'View original paper' link on this screen. This view will also present users with a word/phrase cloud that consists of key words and phrases in the text, providing a quick sense of what the paper is about (see Figure 10).

3.3.2 Searching for words and phrases using MICUSP Simple

MICUSP Simple allows users to search for words and phrases in the whole corpus or just in papers that match the user's selections of the discipline, paper type, student level, nativeness status and textual features. A user enters a word or phrase in the text search box (see #1 in Figure 11) and clicks Search.

The number of times that the search term occurs and the number of papers in which it is found are shown in the text right under the search box (see #2). The histogram displays the results in terms of actual occurrences of the search term or instances per 10,000 words, and hovering over a bar reveals this number (see #3). The search results are presented in the table in the bottom half of the screen (see #4). As with the paper browsing function, it is possible to limit the search results by interacting with the graphs and

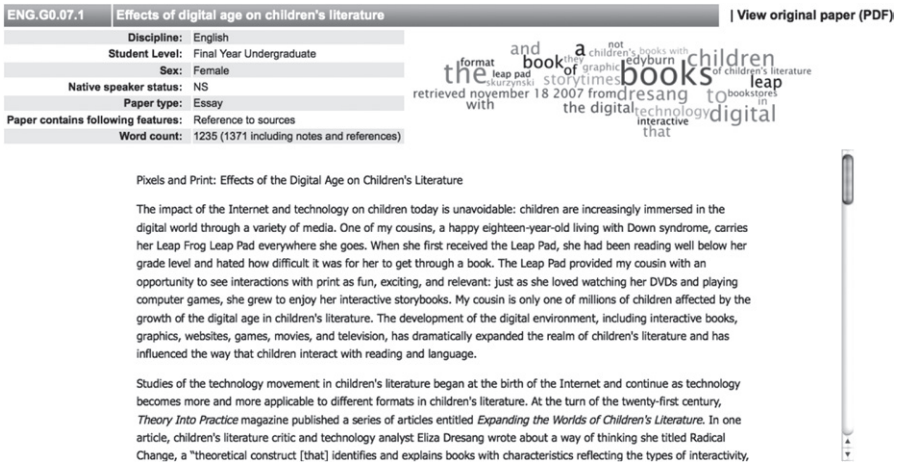

Figure 10: Viewing the full text of one paper in MICUSP Simple

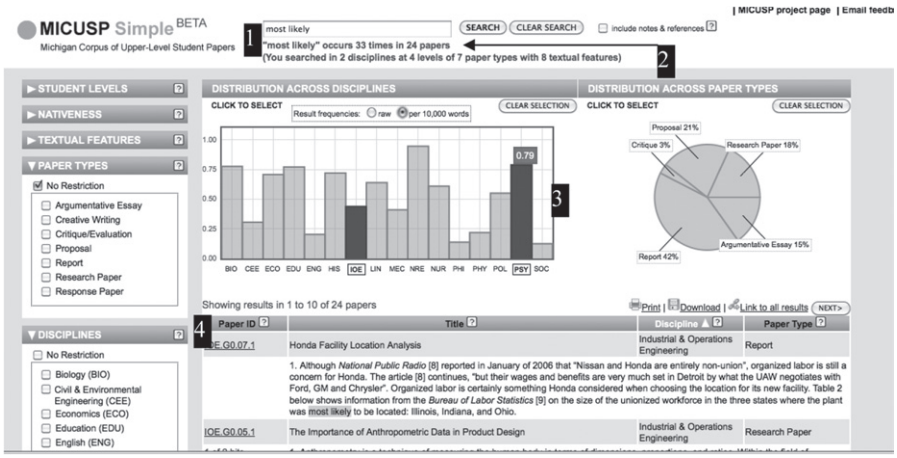

Figure 11: Using MICUSP Simple to search for words and phrases

selecting checkboxes. For MICUSP Simple users we decided to provide results with an entire paragraph of context as opposed to a KWIC display. Where there is more than one instance of the search term in the paragraph, the paragraph is only displayed once with all instances highlighted. Clicking on the paper ID for a specific result will open another browser tab, and show the whole paper with all paragraphs and instances of the search term highlighted.

4. Conclusion

In this two-part series (for part 1, see Römer and O'Donnell, 2011) we have provided a full account of the compilation and online distribution of

MICUSP, a corpus of upper-level student papers from different academic disciplines. We have discussed important issues related to MICUSP text solicitation and collection, the composition of the corpus, the text conversion and markup process, the annotation of textual and layout-specific characteristics, the classification of the papers according to text types, and the design and implementation of a user-friendly search-and-browse interface for the corpus. We have presented each of these issues in sufficient detail for future MICUSP users to learn enough about what is in the corpus and what information can be retrieved from it. We have also provided a considerable amount of technical detail which may be useful for corpus developers and compilers. In the final part of this paper, we have offered a brief tutorial on how the corpus can be easily accessed through the MICUSP Simple interface.

While MICUSP Simple enables easy access to all corpus files, and facilitates simple word and phrase searches, we are aware that some users, especially corpus linguists, may wish to go beyond the functionality of the online tool and work with the corpus (and their favourite concordance tools or computer scripts) offline. In order to address this issue, the next stages of the project will include the release of annotated (XML) and unannotated (plain text) versions of MICUSP, accompanied by a book that is meant to function as a resource guide to MICUSP users (Römer and O'Donnell, forthcoming). We invite students, EAP and ESL teachers, and researchers to join us in exploring MICUSP to find out more about advanced student writing across disciplines.

Acknowledgements

MICUSP and MICUSP Simple have been very much a team effort. The authors would like to acknowledge the support of a number of people who were involved in the project at various stages (in alphabetical order): Annelie Ädel, Derek Blancey, Kate Boyd, Yung-Hui Chien, Gregory Garretson, Geoffrey Ho, Lucas Jarmin, Miranda Kozman, Emily Lin, Kelly Lockman, Nasy Pfanner, Jesse Sielaff, Rita Simpson-Vlach, John Swales, Madison Stuart, Edwin Teng and Beilei Zhang. We are also grateful for the support of instructors and testing specialists at the University of Michigan English Language Institute and members of the University of Michigan Corpus Analysis Group.

References

Ädel, A. and U. Römer. 2012. 'Research on advanced student writing across disciplines and levels: introducing the Michigan Corpus of Upper-level Student Papers', International Journal of Corpus Linguistics 17 (1), pp. 1–32.

Bird, S., E. Klein and E. Loper. 2009. Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit. Cambridge, Massachusetts: O'Reilly Media.

Burnard, L. 2005. 'Metadata for corpus work' in M. Wynne (ed.) Developing Linguistic Corpora: A Guide to Good Practice, pp. 30–46. Oxford: Oxbow Books. Accessed 30 July 2010, at: http://ahds.ac.uk/linguistic-corpora/

Ebeling, S.E. and A. Heuboeck. 2007. 'Encoding document information in a corpus of student writing: the British Academic Written English corpus', Corpora 2 (2), pp. 241–56.

Hardie, A. Submitted. 'CQPweb – combining power, flexibility and usability in a corpus analysis tool'.

Hoffmann, S., S. Evert, N. Smith, D. Lee and Y. Berglund-Prytz. 2008. Corpus Linguistics with BNCweb – A Practical Guide. New York: Peter Lang.

Kilgarriff, A., P. Rychly, P. Smrz and D. Tugwell. 2004. 'The Sketch Engine' in Proceedings of EURALEX 2004. Lorient, France. Accessed 30 July 2010, at: http://trac.sketchengine.co.uk/attachment/wiki/SkE/DocsIndex/sketch-engine-elx04.pdf

McEnery, A. and R. Xiao. 2005. 'Character encoding in corpus construction' in M. Wynne (ed.) Developing Linguistic Corpora: A Guide to Good Practice, pp. 47–58. Oxford: Oxbow Books. Accessed 30 July 2010, at: http://ahds.ac.uk/linguistic-corpora/

O'Donnell, M.B. and U. Römer. In preparation. 'Investigating the interaction between phraseological items and textual position'.

Römer, U. and M.B. O'Donnell. 2011. 'From student hard drive to web corpus (part 1): the design, compilation and genre classification of the Michigan Corpus of Upper-level Student Papers (MICUSP)', Corpora 6 (2), pp. 159–77.

Römer, U. and M.B. O'Donnell. Forthcoming. MICUSP: A Corpus Resource for Exploring Proficient Student Writing across Disciplines. (Book and CD-ROM; provisional title). Amsterdam: John Benjamins.

Schmid, H. 1994. 'Probabilistic part-of-speech tagging using decision trees'. Accessed 30 July 2010, at: http://www.ims.uni-stuttgart.de/ftp/pub/corpora/tree-tagger1.pdf

Tateisi, Y. and J. Tsujii. 2004. 'Part-of-speech annotation of biology research abstracts' in the Proceedings of 4th International Conference on Language Resource and Evaluation (LREC2004), IV, pp. 1267–70. Lisbon, Portugal.

van der Vlist, E. 2007. Schematron. Cambridge, Massachusetts: O'Reilly Media.