Ecdosis: historical texts on the Web

Desmond Schmidt\textsuperscript{1}, Paul Eggert\textsuperscript{2}

\textsuperscript{1}University of Queensland, \textsuperscript{2}Loyola University, Chicago

June 22, 2016
Why we need an editing system for historical texts

- Our libraries are full of original historical documents
- The ‘digital first’ policy means virtually no new printed material is being stored, but they still have ‘kilometres’ of rare books and manuscripts
- The vast majority of this material is modern. It consists of scrappy collections of letters, receipts, agreements, diaries, drafts, literary works etc.
- We have few usable tools to edit and publish this material.
Why we need for a practical system for editing

- Business document systems are not designed for historical materials
- They focus on formatting, embedded images, squeaky clean texts
- Our requirements instead focus on:

Desmond Schmidt\textsuperscript{1}, Paul Eggert\textsuperscript{2} \textsuperscript{(1)University of Queensland, 2}Loyola University, Chicago)

Ecdosis: historical texts on the Web

June 22, 2016 3 / 19
Why we need for a practical system for editing

- Business document systems are not designed for historical materials
- They focus on formatting, embedded images, squeaky clean texts
- Our requirements instead focus on:
  1. Incomplete texts, many copies, ‘dirty’ drafts with heavy revisions
  2. A general system for a wide range of material types
  3. Durable tools that last beyond a single research project
  4. High level of usability for editors who don’t want to become programmers
The boom-bust cycle of DH software projects

1. Research idea: build a tool
2. Apply for funding
   - yes: Obtain funding
     - Hire developers
       - Build tool
         - Money runs out
           - Open source code
             - Community steps in
               - Tool survives
           - Developers leave
             - Tool breaks
               - Tool is forgotten
Is open source really the saviour?

“...If you start off with some ‘kumba-ya feeling’ where you think people from all the world are going to come together to make a better world by working together on your project, you probably won’t be going very far. ”

Linus Torvalds, 2011
There are a lot of silo projects that have been developed for specific sets of documents ... but often do not have sufficient infrastructure to ensure longevity.

Hajo 2010

That all sounds very good, but it doesn’t change one bit the fact that in my opinion TextGrid, probably in the next few years, will disappear. And it is good if it does.

Fotis Jannidis, 2015
Is software really that fragile?

There are programs from the 1970s that still work perfectly

In order to last software needs to be:

1. Written with minimal dependencies on other software
2. Based on older technologies that are still popular
3. Written in smaller units, not monolithically
4. Well designed to match user needs
User-centric design

- The foundation of modern interaction design
- Software needs to be designed:
  - Backwards from the user’s true needs
  - *Not* ‘build it and they will come’
  - *Not* starting from the features we want to encode in the text
Waterfall vs agile development

**Waterfall**
- Survey of user needs
- Wire-frame design
- Build/refine prototype
- User testing
- Final product

**Agile**
- Minimal prototype
- User-testing
- ‘We need feature X’
- Add new feature
- Final product
Technical foundations of *Ecdosis*

- *Not* based on XML (eXtensible Markup Language)
- Avoids XML’s problems with interoperability, reusability, overlapping hierarchies
- Document data was therefore split into two parts:
  1. Markup (properties applied to the text)
  2. Variation (the structure of the text itself)
## Stand-off properties in a nutshell

**Embedded XML:**

```xml
<Kendall had written me: Alas! I fear
Our dear good Friend is breaking—meaning <emph>you</emph>.
```

**Standoff XML:**

```
<Kendall had written me: Alas! I fear

Our dear good Friend is breaking—meaning you;
```

**Standoff properties:**

- **Kendall had written me:** Alas! I fear
- **Our dear good Friend is breaking:** meaning you;
- **line:**
- **quotation:**
- **line-indent-1:**
- **emphasis:**
Advantages

- Textual properties can freely overlap
- Different sets of textual properties can be combined: text-to-image links, formatting, textual differences
- Sources stored as durable, reusable, interoperable plain text
- Can be easily converted into editable forms (e.g. Markdown) or as viewable/archivable HTML
Representing variation

MS: while its walls of lofty mountains remain hidden under the sombre festoonings
TS2: with its walls of lofty mountains hidden under the sombre festoonings
S: with its walls of lofty mountains hung with the mourning draperies
Advantages

- Dramatic simplification of markup
- Can be searched more accurately than XML
- Facilitates comparison of versions/layers
- Enables other visualisations (tree, table view, more readable texts)
User interface

There are essentially two kinds of user:

1. Editors, those updating/creating the website
2. Users, essentially viewers, commentators on the website

This is reflected in the division in Ecdosis between ‘back-end’ and ‘front-end’ tools
Module breakdown

[Diagram showing Drupal's back-end and front-end modules]

- **Back-end**
  - Documents
  - Project edit
  - Uploader
  - Gallery
  - Tilt edit
  - Tilt preflight

- **Front-end**
  - Login/users
  - Menus
  - Browse
  - Msviewer
  - Find
  - Annotator
  - Compare
  - Biography
  - Mvdsingle
  - Works
  - Timeline
  - Tabs
  - Tree
  - Table
  - Para (misc)
  - Gridhome
  - Quoteticker

Desmond Schmidt¹, Paul Eggert² (¹University, ²Loyola University, Chicago)
Standardisation

![Diagram showing standardisation in the context of cultural heritage websites and technologies like XML, RDF, MODS, METS, and CIDOC/CRM.]
Demo

1. Browse
2. Single view
3. Compare
4. Tree
5. Table
6. Biography
7. Timeline
8. TILT preflight