Journal of the Text Encoding Initiative

Issue 7 | November 2014
Reaching out, Opting in

A Design Methodology for Exploring and Communicating System Values and Assumptions

Daniel Carter

Electronic version
URL: http://journals.openedition.org/jtei/974
DOI: 10.4000/jtei.974
ISSN: 2162-5603

Publisher
TEI Consortium

Electronic reference
Daniel Carter, « A Design Methodology for Exploring and Communicating System Values and Assumptions », *Journal of the Text Encoding Initiative* [Online], Issue 7 | November 2014, Online since 01 January 2014, connection on 22 April 2019. URL : http://journals.openedition.org/jtei/974 ; DOI : 10.4000/jtei.974

Journal of the Text Encoding Initiative

Issue 7 | 2014
Reaching out, Opting in

A Design Methodology for Exploring and Communicating System Values and Assumptions

Daniel Carter

revues.org

Publisher
TEI Consortium

Electronic version
URL: http://jtei.revues.org/974
ISSN: 2162-5603

Electronic reference

A Design Methodology for Exploring and Communicating System Values and Assumptions

Daniel Carter

1. Introduction

1 This project adopts the process of building a prototype for displaying TEI documents that encode collations of different text versions as a method for exploring and intervening in a sociotechnical system comprised of technical standards, theoretical goals, available tools, and social configurations. In addition to asking the methodological question of what is gained from the process of designing a prototype, I also explore conceptual issues related to the TEI standard, asking how technical choices and theoretical goals influence conceptions of textual editing.

2 This methodology is influenced by work in the fields of design and human-computer interaction (HCI) that asks what can be learned and achieved through the design of prototypes. Specifically, I take three points to be especially relevant to this project:

1. Design can build conceptual knowledge. Daniel Fallman (2007) argues that in addition to solving problems and producing products, design can lead to the creation of new knowledge. Building prototypes that explore possibilities outside current paradigms and placing them in use, Fallman claims, allows researchers to arrive at an understanding or description of the world that might be otherwise unobtainable. While this knowledge may come from observing a prototype in use, Fallman (2003, 231) also notes that the design process itself may lead researchers to new understanding through reflection. Seen in this way, prototypes do not solve technical problems but create situations for creating knowledge. Thus, these prototypes do not attempt the technical completeness of a final product, instead occupying a space Fallman describes as "a kind of middle ground between a thought experiment and a real thing" (2007, 197).
2. Design can allow for critical reflection. Matt Ratto (2011, 253) extends Fallman's interest in design as a research method by introducing a critical dimension to the process, suggesting that working with materials and standards is a way to understand the values present in existing (and potential) systems. He describes critical making as a method for negotiating

the perceived divide between critical thinking and pragmatic making. Prototypes are, for Ratto, not ends in themselves but instead means to better understand a system—combined with critical reflection on materials, processes, and outcomes, the act of making allows for a consideration of both practical and theoretical issues.

3. Design can communicate alternative visions. Where Ratto's critical making is primarily a process in which an individual or group explores a concept, Dunne (2005) introduces the term *value fictions* to refer to objects that are primarily rhetorical and that use the material reality and technical feasibility of a prototype to intervene in an existing conversation. In addition to considering prototypes as built and used, Dunne also considers them as rhetorical objects that are seen and imagined. While Dunne's work tends to be in the aestheticized realms of personal electronics and biotechnology, I'm interested in transporting the idea of value fictions to academic informatics and asking how a functioning prototype can be created to explore and communicate alternative values and methods of working.

3 Taken together, these theories suggest a process of design that moves from the exploration of concepts and values to an intervention in an existing system. While researchers working with TEI often build prototypes and systems as part of their work—and often see such building as producing conceptual knowledge—theories like those discussed above suggest a potentially productive framework for viewing and articulating that building and knowledge production.

2. Methodology

4 In the following, I begin with a discussion of how technical systems and standards interact with theoretical positions and social configurations and analyze how an existing system for displaying TEI documents, the Versioning Machine,[1] influences conceptions of textual editing. I then present a prototype that uses this analysis to reimagine the Versioning Machine. While I begin with analysis and proceed to design, one of my claims is that these two processes are intertwined—understanding a sociotechnical system and designing a prototype to intervene in that system do not, in practice, happen sequentially but instead reinforce and feed back into each other. Designers draw on sociotechnical analysis as a way to identify issues and focus attention on areas of interest. Similarly, designers interact with technical systems in a way that provides a unique perspective from which to analyze those same systems.

5 Approaches to communicating these intertwined processes vary. Some researchers in HCI (see Aoki 2007, Höök 2010 and Rosner 2012) have argued for various autoethnographic approaches, relying on their personal experiences and self-documentation to reflect on knowledge obtained through action. These researchers respond to the same conditions that prompt Donald Schön (1983) to claim that technical rationality and the scientific method are insufficient to describe the work of professionals such as designers. Claiming that professional knowledge is bound up in action, Schön (1983) argues that when confronted with a problematic situation, a professional "reflects on the understandings which have been implicit in action, understandings which he surfaces, criticizes, restructures, and embodies in further action" (1983, 50). Autoethnographic narratives seem designed to present this reflection in a straightforward and transparent manner. Indeed, for knowledge that pertains to processes that are closely tied to the body and that do not produce an artifact, autoethnography seems a particularly apt method.

6 However, for communicating design research that does produce artifacts, John Bowers (2012, 68) suggests the value of creating annotated portfolios that "captur[e] the family resemblances that exist in a collection of artefacts, simultaneously respecting the

particularity of specific designs and engaging with broader concerns." While annotated portfolios move away from the direct recording of reflection, they offer the researcher a way to structure a research process around issues that extend beyond an individual's experience. For the kind of investigation into sociotechnical systems suggested here, annotated portfolios seem to offer a way to focus design research, picking out features of design that can be used to guide further work. As Bowers (2012, 70) points out, designed artifacts engage with a heterogeneous set of concerns (practical, aesthetic, social, psychological), and written accounts can only remark on some of these facets. William Gaver (2012) similarly describes this process of narrowing, claiming that annotated portfolios position an individual artifact within a range of possibilities and communicate a designer's opinions on the central concerns in a given domain. In this way, a written account of design research will respond to only some of the issues raised by an object, serving not as a replacement for the process of design but as a strategic product made possible by that process. By annotating a set of objects, researchers are able to use the reflective process of design to focus and communicate a set of theoretical goals in a way that bridges local, tacit knowledge and broader issues of relevance to a sociotechnical system. And while Bowers and Gaver both describe annotated portfolios as drawing together artifacts created by an individual designer (or a group of related designers), the method of focused annotation also seems a productive method for identifying the central concerns of a system prior to (and during) the design of a prototype meant to intervene in that system.

7 In communicating my own design research, I adopt here a method similar to the annotated portfolio, first giving an analysis of the Versioning Machine that draws out issues central to the tool's conception of the work of textual editing and then using those issues to focus my discussion of the prototype presented. This prototype is not intended to be an optimized solution to the issues discussed—indeed, several of the technical choices made in its design may be most productively viewed as pointing to difficult questions that remain unresolved. As one reviewer of this essay noted, the particularly thorny issue of whether a system for working with encoded texts can avoid imposing an interpretive schema is addressed in the prototype only by introducing other, equally serious problems. This is certainly the case. The prototype requires a somewhat idiosyncratic modification of the parallel segmentation method of encoding variance, and it stores annotations to texts externally and in the JavaScript Object Notation (JSON) format. For the purposes of the prototype, this file was manually created, but it could of course be automatically generated from `<note>` elements encoded in a TEI document. However, these debatable choices can be seen as ways of testing limits, sketching boundaries, and communicating questions related to a system. As thought experiments that draw on the experiences of design and use, prototypes offer perspectives that can further current conceptual understandings and build toward better future systems.

3. Current States

8 During the creation of software tools, the assumptions and intentions of creators combine with pragmatic choices and accidental variations to encourage or discourage certain user behaviors. While aspects of these tools may reflect technological limitations or unintended consequences entailed by the complicated process of writing software, intentional authorial choices are also incorporated as rules into code, suggested in

documentation, or encouraged through interface design. Lawrence Lessig (2006, 6), for example, examines the ways code has been used to design an Internet with various values built into its structure and suggests that we should ask who does this building and with what values in mind. Similarly, Florence Millerand and Geoffrey C. Bowker (2009, 152) point to the way technical standards are never merely technical, acting as social forces that arrive bearing assumptions about the configuration of people and tools with which they will interact. While the TEI Guidelines are adopted in various ways and for various purposes, their values and assumptions—as well as those of tools built to interact with these standards—similarly structure the social and theoretical work of textual editing. As tools and interfaces are built, as guidelines are refined, and as institutions and scholars make decisions about whether and how to adopt TEI, these values and assumptions should be made transparent.

9 When preparing a digital scholarly edition, a textual editor must negotiate between technical constraints, theoretical goals, available tools, and social configurations. Standards and guidelines influence technical and theoretical possibilities. Available tools and interfaces influence the practice of encoding texts. And encoding practices influence how users access and scholars work with texts. The Modern Language Association's (MLA) *Guidelines for Editors of Scholarly Editions* (2011) notes the diversity of possible approaches to textual editing, suggesting that editors focus on different aspects of a text based on their theoretical perspective and that these perspectives "range broadly across a spectrum ... and editors may select a given methodology for a variety of reasons." In this way, interfaces and theories may be contingent on each other, with some theoretical approaches to textual editing producing documents that are incompatible with the assumptions of some tools (and vice versa). Tanya Clement (2011, 8) suggests that the process of choosing what to encode may even begin with a consideration of an eventual interface and claims that "the standard or model for encoding a text depends on how the scholar defines the digital textual event in which it will be enacted." This interaction between theoretical goals and imagined uses, with the process of encoding caught up in the middle, draws attention to the way a text encoded in TEI might, to some extent, be linked to an eventual instantiation.

10 For example, as part of his Visualizing Variation project, Alan Galey presents a prototype that shifts between witnesses, visually fading between textual variants without warning. Based on a theory of textual editing that claims digital editions can "represent variants dynamically, presenting their ambiguity to readers not as a problem to solve, but as a field of interpretive possibility," Galey's prototype works within the TEI standard to create an interface that meets these preexisting theoretical goals (Galey 2013). It also makes choices about how texts should be encoded to meet his theoretical and technical needs. For example, of the three methods described by the TEI Guidelines for encoding textual variants, Galey's prototype is designed to work with texts encoded using the *parallel segmentation* method. Parallel segmentation encodes textual variants by creating `<app>` elements inside which the aligned textual variants from all witnesses are presented and compared. For example, a simplified version of the encoded text used by Galey records three variants from *Hamlet*:[2]

Example 1: Variants encoded using the parallel segmentation method.

```
O that this too too
<app>
  <rdg>solid</rdg>
  <rdg>sallied</rdg>
  <rdg>sullied</rdg>
</app>
flesh would melt,
```

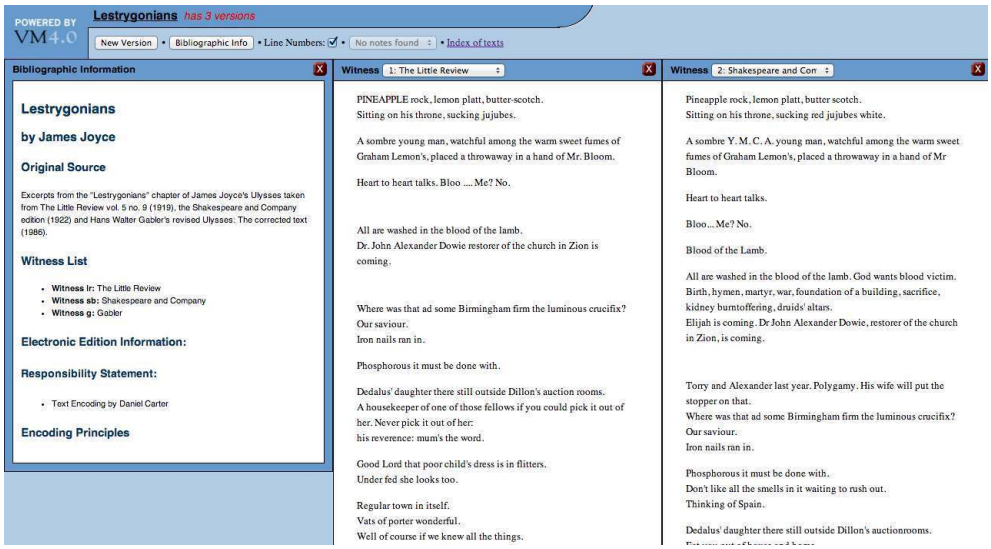
11 From a theoretical perspective, parallel segmentation works well with Galey's stated goals, as it does not require that the editor establish a hierarchy by specifying a base text, a witness that will be taken as the original or the reading against which all other witnesses will be compared. From a technical perspective, parallel segmentation makes the design of Galey's prototype simpler by bringing all variants together in one element where they can easily be manipulated, whereas the *double end-point attachment* method allows variants to be distributed either within a single file or across several. And from a perspective that considers how editors currently work with TEI, parallel segmentation has the advantage of being relatively easy to write and interpret in a basic text editor.

12 The choice to work with the parallel segmentation method means that texts encoded using other methods cannot be used with Galey's prototype. The *location-referenced* method requires editors to identify only the beginning of the text to which a variant refers, although they may also use the `<lem>` element to specify the full text referred to. Using the `<lem>` element in this way, however, would create additional work in programming the interface (which would need to perform text-matching to identify the location of the variants) as well as in specifying the lemmas in the encoded text. Indeed the TEI Guidelines notes that editors often choose not to perform the extra work of including lemmas and that, if a full reconstruction of all witnesses is desired (as it is with Galey's prototype), the location-referenced method is less appropriate than other methods (TEI Consortium 2012). Galey's choice, then, of parallel segmentation can be seen as an example of how theoretical goals, technical constraints, and social configurations can determine how texts are encoded.

13 The Versioning Machine provides an opportunity for a more nuanced exploration of these relationships, as well as a discussion of how they influence conceptions of textual editing. Conceived by Susan Schreibman, the Versioning Machine is a set of XSLT, HTML, CSS, and JavaScript files that allows textual editors to create web interfaces to display TEI documents that encode collations of different text versions (Schreibman, Kumar, and McDonald 2003, 102). As I worked to redesign and reimagine the Versioning Machine, I asked how certain technical features determine how texts are encoded and how these choices might be seen as encouraging—but not formally enforcing—a conception of textual editing as the creation, by a single editor, of a document that is intended for one instantiation. I also asked how a prototype could be designed that would experiment with alternate conceptions of textual editing, exploring ways that an encoded document might be shared by researchers and instantiated in a variety of ways. The observations below result from repeated cycles of interaction with the Versioning Machine and development of the prototype presented here, which I am arguing can be used not to fix, improve, or

innovate on a given system but to come to understand the values and assumptions of that system and to communicate alternatives. In this way, the following annotations are not a complete description of the Versioning Machine but instead a focused interpretation that delimits a field of interest discussed in the next section.

Figure 1: Excerpts from the "Lestrygonians" chapter of James Joyce's *Ulysses* displayed in the Versioning Machine interface.



3.1 Creating Annotations

14 The conception of textual editing implied by the Versioning Machine is perhaps best seen in its treatment of editorial annotations. Using the `<note>` element, editors place annotations directly within the text, adjacent to the element on which they want to comment. The TEI Guidelines refer to this method as *implicit linking*, as the relationship between annotation and text must be understood through positioning rather than a more explicit declaration (TEI Consortium 2012). While the method is less precise than using `@target` and `@targetEnd` attributes to specify the exact scope of the reference, it does allow for notes to be placed within larger units (for example, adjacent to a single letter within a line) and is easy for editors to implement in a basic text editor.

15 However, the Versioning Machine's implementation of implicit linking can also encourage a conception of the encoded text as entangled with content specific to an editor (or consistent editorial team). While the `<note>` element allows for a `@resp` attribute to differentiate the contributions of multiple editors, this information is not reflected by the Versioning Machine interface. In the Versioning Machine's provided sample texts, editorial responsibility is claimed in the `<teiHeader>` for the entire document, and in the instances when the `@resp` attribute is attached to elements, the value of `"editor"` is used, implying that responsibility is consistent across the document. Together with the interface's lack of support for the `@resp` attribute, these examples suggest how systems can encourage certain assumptions without formally requiring them. While a text could be encoded with annotations from multiple editors with differing interpretations, the Versioning Machine is not designed to make these choices visible. Further, its sample texts model encoding as a process that enmeshes a set of interpretations with a text, encouraging the document to be seen as the work of a

specific editor or editorial team following a specific interpretational framework. This conception of textual editing is not unusual—indeed, John Bryant (2002) notes that an editor's critical goals will influence choices made in compiling an edition, and the MLA's *Guidelines for Editors of Scholarly Editions* (2011) makes similar claims. Still, the relationship between individual interpretation and systems for working with encoded text presents an interesting problem, and the Versioning Machine represents a useful marker for thinking within that space.

16 Stand-off markup, described by Piotr Bański (2010) as "creating/organizing a structure in resource A out of elements of resource B by pointing to them" offers an alternative approach to this issue. Essentially, stand-off markup represents a broad approach to encoding in which annotations are separated from the elements to which they refer. In the context of editorial annotations, for example, the concept of stand-off markup suggests that notes and other features specific to an editor might be stored in a separate file.

17 While the Versioning Machine assumes that `<note>` elements will be placed directly in the text, the TEI Guidelines describe alternative methods that would allow these notes to exist in other locations. For example, `<note>` elements can be linked to sections of text either by inserting `<ptr>` elements into the text or by using the XPointer Framework, which relieves the need to clutter a text with a potentially large number of `<ptr>`s. Both methods suggests different conceptions of the work of textual editing. For example, moving an individual editor's annotations into an external file suggests that encoded texts might be used by multiple editors to express various critical goals and interpretations. Similarly, storing annotations externally suggests that an interface to display encoded texts might bring together multiple interpretations in flexible ways. Rather than seeing an encoded text as the work of one editor, this conception of textual editing imagines that interpretations might be mixed and matched and that encoded texts might have a variety of instantiations. And while a system that relies on implicit linking might not technically foreclose these possibilities, the relationship between standards, practice, and interfaces is more than technical—editors work based on models of what is possible and, to some extent, imagine those outcomes that available tools encourage.

3.2 Changing Styles

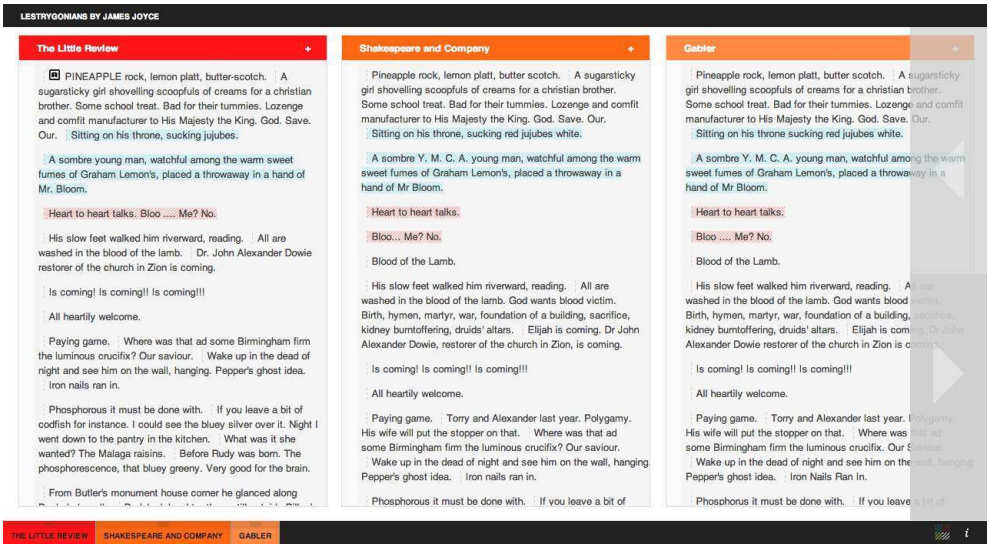
18 The Versioning Machine also presents an interesting context to think about the relationship between a text's encoding and subsequent instantiation in an interface. Specifically, in the Versioning Machine, environment changes to the interface are seen as connected with changes to the text encoding, suggesting that editors might be encouraged to see the work of encoding as leading to a single instantiation. On a surface level, this can be seen in the way editors choose between the two stylesheets provided for use with the Versioning Machine. For example, to select the alternate style of interface created by Tanya Clement for her Elsa von Freytag-Loringhoven project, an editor is instructed to change the XML stylesheet declaration, found at the top of the TEI document, to reference the `vmachine_evfl.xsl` file that is packaged with the tool. This minor change alters the appearance of the Versioning Machine interface. More importantly, though, it also has several implications for how encoded texts are processed —and, as a consequence, for how editors encode texts. `<note>` elements with a `@type`

attribute of "image," for example, are displayed normally with the default styling but are not displayed at all with the alternate styling. Similarly, if <note> elements with a @type attribute of "critIntro" are included in a specific location in the TEI document, they will be displayed, with the alternate styling, in a special popup panel. Finally, selecting the alternate stylesheet allows editors to use a modified version of the location-referenced method of encoding textual variants in addition to the parallel segmentation method. The Versioning Machine documentation notes that using this method allows editors to connect sections of text that may appear in different locations, and Tanya Clement and Gaby Divay (2012) connect the need for this method of encoding variants with the goal of allowing readers a high level of engagement with a complex text. However, this high level of engagement, performed as described in the documentation, limits the ways the text can be displayed. In the Versioning Machine, a change to the style of the interface is often accompanied by a change to the way the text is encoded.

19 Thus, an editor who decides to use the alternate stylesheet makes at least one change to an encoded document (by changing the stylesheet declaration) and will likely make other changes to either accommodate the alternate behavior (by changing the @type attribute of <note> elements that might have appeared using the original stylesheet or by including location-referenced encoding). While these modifications may be seen as minor, they reinforce a conception of textual editing that sees encoded files as leading to a particular instantiation. As with the issue of stand-off markup noted above, this is in many ways consistent with current understandings of textual editing. Alongside the problem area of how individual interpretations relate to the practice of encoding is the issue of how encoding relates to an eventual instantiation in an interface.

4. Alternate States

Figure 2: Excerpts from the "Lestrygonians" chapter of James Joyce's *Ulysses* displayed in the prototype interface.



20 While the prototype created for this project[3] is based on the Versioning Machine, it diverges from that original in several ways designed to experiment with alternative conceptions of textual editing.

4.1 Changing Styles

21 Like the Versioning Machine, the prototype discussed here provides users with options to change the appearance and behavior of the created interface. However, the options offered by the prototype differ from those of the Versioning Machine in several ways. First, the options do not alter the processing of the encoded text—where I argue above that changing styles in the Versioning Machine assumes that an editor will return to the text and modify the encoding, the prototype's options change only the appearance of the interface, meaning there is no reason for an editor to modify an encoded text in response to a change in interface style. Second, where users of the Versioning Machine select the alternate style of interface by modifying the encoded text, the prototype inserts an intermediary step in the process. Instead of directly transforming an XML file into an interface using XSLT, the prototype includes an additional step, using JavaScript, that specifies options for the appearance and functioning of the created interface.

Example 2: JavaScript code providing options for changing the appearance of the interface created by the prototype.

```
$('#teiHolder').data('modVers', {
xmlFile: 'data/teiFile.xml'
annotations: 'data/annotations.json',
fixFirst: false,
fullscreen: false,
height: 300,
ids: 'a,b,c',
witnesses: 'v1,v2'
});
```

22 Using these options, users can control the appearance and behavior of the interface, specifying its height, whether the first witness should be scrollable or fixed, and whether the interface should fill the entire browser window or should instead be constrained within another element such as a column of text. Users can also choose to have the interface display only certain segments of text, specified by `@xml:id` attributes of `<app>` elements, or only certain witnesses. These options allow users to create a variety of instantiations from one encoded text. For example, a user might want to compare only some witnesses from the many encoded in one TEI document or focus on only one section from a long text. These options suggest alternative ways of working with encoded text, giving editors the ability to use (and reuse) documents, experimenting quickly with different configurations of text and interface to, ultimately, make a broad range of arguments. Editors can try out new ways of presenting texts and can present the same text in multiple ways by creating a series of editions based on one TEI document, offering readers of digital editions new ways to experience a text.

4.2 Creating Annotations

23 As a way of experimenting with how encoded files might be interpreted and presented in multiple ways, the prototype also avoids the implicit linking of annotations that was seen

in the Versioning Machine. Instead, it implements a version of stand-off markup that places annotations in a separate file, encoded in JavaScript Object Notation (JSON). When writing the JavaScript code that instantiates and gives options for the interface, users can specify a file containing annotations, and when the interface is created the annotations in the external file are matched to the `@xml:id` attributes of the document's `<app>` elements. In this way, the same TEI document could be used in multiple instantiations, each bringing together the text with a different set of annotations.

Example 3: A sample JSON file of annotations.

```
"annotations": {
    "items": [
        {
          "id": "1",
          "wit": "lr",
          "text": "This is an annotation applied to one witness.",
          "resp": "Editor 1"
        },
        {
            "id": "38",
            "text": "This is an annotation applied to all witnesses.",
            "resp": "Editor 2"
        }
    ]
},
```

24 In example 3, the `id` option refers to `<app>` elements in the encoded text. The `wit` option declares with which witnesses the annotation should be associated; if it is not included, the annotation will be applied to all witnesses. The `text` option provides the content of the annotation, and the `resp` option specifies the editor responsible for the annotation.

Example 4: An excerpt of the XML to which the annotations in example 3 refer.

```
<app xml:id="1">
  <rdg wit="#sb #g">
    Pineapple rock, lemon platt, butter scotch.
    </rdg>
  <rdg wit="#lr">
    PINEAPPLE rock, lemon platt, butter-scotch.
    </rdg>
</app>
<!-- ... -->
<app xml:id="38">
  <rdg wit="#sb #g">
    Regular world in itself.
    </rdg>
  <rdg wit="#lr">
    Regular town in itself.
    </rdg>
</app>
```

25 As noted, this adoption of stand-off markup poses technical and theoretical problems. While it removes the more overt marks of an individual editor from the encoded text, there are still significant aspects of the encoding that will stem from specific theoretical goals and choices—for example, the segmentation of the textual variants into `<app>` elements. Further, only by requiring the use of the `@xml:id` attribute on `<app>` elements does the prototype overcome the implicit nature of parallel segmentation, but this is a somewhat idiosyncratic solution adopted to deal with the current technical difficulty of working extensively with the double endpoint attachment method. This concession highlights the negotiated nature of any kind of segmentation of the text. While multiple editors may annotate a text and multiple annotations may be brought together in various configurations, the segments that can be commented on are, as described here, a limit to the separation of interpretation and textual markup. Similarly, the tension between standards adoption and idiosyncrasy raises the question of how far this separation can extend, as well as the circumstances in which it would be desirable. The negotiation of segments, after all, could prove to be a productive conversation at the local level while simultaneously representing a failure from a system perspective.

26 While one of the benefits of building prototypes is the opportunity to foster these conversations about the boundaries of practice, I would also argue that there is a benefit to allowing users to see the effects of an imperfect implementation. Moving annotations into an external file, regardless of technical specificities, allows for certain practices of textual editing to be enacted and also for the implications of stand-off markup beyond this prototype to be imagined. By allowing multiple sets of annotations to be created based on a single TEI document and by presenting users with the option to create instantiations that bring encoded texts and annotations together in different ways, the prototype presents and opens for debate an alternative conception of textual editing. James Cummings (2009, 307) refers to editions that use a similar kind of stand-off markup as "agile editions," claiming that moving markup that is "not directly related to the

transcription of the text" into external files allows the encoded documents to be used (and reused) easily and flexibly (307). This issue of reuse is fundamentally connected with conceptions of the work of textual editing—both Cummings' work and the prototype presented here point to ways that a single encoded text could serve as a common hub for multiple editors to come together to annotate a text based on each individual's theoretical goals, and to ways that an exhibit of these annotations could be created to present, in series, a range of interpretations. At the same time, prototypes, as opposed to written articles, can also make visible the technical and theoretical problems inherent in such a system. Indeed, one reviewer of this article points out that the system described would require multiple editors to comment on a text that has already been segmented: "Each individual's theoretical goals will (by design) still be secondary to those of the first editor ... Additional annotations will always have to function as commentary to some basic editorial decisions." This issue points to areas for further work, suggesting that future systems might require either an extremely fine-grained segmentation (at the word level, for example) of the text or the use of the XPointer framework to specify areas of text without prior segmentation.

5. Methodological Reflections

27 There are certainly ways of looking at these modifications to the Versioning Machine as irrelevant or even wrong. For example, the options that are specified by JavaScript are in many ways similar to those that might be set in an XSLT stylesheet. Similarly, it might be claimed that it makes little difference where annotations are stored; even those stored internally in `<note>` tags and implicitly linked can easily be processed and moved to an external file. The systems imagined above—in which an encoded text is brought together in various ways with a range of interpretations—are certainly possible without implementing the specific changes described here. And the choice to store annotations in the JSON format instead of XML would be unlikely to hold in a system designed to be technically optimal.

28 Further, the prototype, in responding to the Versioning Machine, creates its own assumptions about textual editing. It assumes competency in the JSON format as well as the current limitations associated with text editors and the infeasibility of working with extremely complex XML documents. It also opts, in some instances, to limit certain possibilities (the visual differentiation of certain kinds of notes, for example) while integrating others, such as the `@resp` attribute, that speak to the concerns addressed here. And, in requiring the use of the parallel segmentation method, the prototype struggles with questions of the extent to which an encoded text can be displayed in multiple systems. While it was noted that the Versioning Machine requires a change in encoding to effect a change in appearance, the degree to which the prototype addresses this issue may be seen as slight. The value here is perhaps not in solving a problem but in triangulating its existence: the prototype presents another view of the current situation of textual editing. Placed alongside the Versioning Machine, I hope the prototype will allow for a richer space of discussion.

29 As Millerand and Bowker (2009, 152) note of another specialized XML schema, the `Ecological Metadata Language`, a markup language is "defined a priori as a solution to a set of technical problems—a solution from which will issue the one good tool that can be used by all." However, Millerand and Bowker also note that these technical

solutions have social consequences and that the "one good tool" must be considered in relation to local enactment, the process by which it is modified for actual use. There may be optimal solutions, but there are also workarounds and experiments that do not necessarily seek optimal solutions to technical problems. Reimagining a system like the Versioning Machine allows designers to confront difficulties and interact with workarounds and experiments. This experience can be used to reflect on conceptual issues, and these issues can be communicated and debated through the prototype designed. Indeed, for design research, the futures imagined need not necessarily be desirable; instead, provocative or even undesirable designs might be used to investigate conceptual issues (Odom et al. 2012, 339).

30 And because prototypes like the one described here are based on current technologies and present themselves as available for use, this interaction with conceptual issues is also available to users through experiment and discussion. Stephanie Schlitz and Garrick Bodine (2009, 340) note this value in describing the Versioning Machine as primarily serving a communicative function, claiming that, while the tool had been used in only one project at the time of their writing, its value in contributing to discussions of tool development "cannot be overstated." Indeed, the greatest value of some tools may be their rhetorical qualities in relation to a field of discourse. As with Dunne's value fictions, both the Versioning Machine and the prototype presented here gain rhetorical force because they are based on existing standards; they adhere to current states of knowledge and are placed in a context of use that provokes a certain kind of discussion. While objects displayed in an art gallery, Dunne (2005, 86) notes, draw attention to the skill of their creators, they "overlook the challenge to the status quo insertion into everyday life might bring about." Prototypes of academic tools seem especially suited to this kind of intervention. Necessarily grounded in an existing discourse and often remarking on their own theoretical goals—but also put before scholars as potential tools to be picked up and used—these projects can challenge assumptions not just through conventional argument but also through presentation and use. The Software Studies Initiative's *ImagePlot* software, for example, allows users to download a software tool and quickly begin experimenting with a new method. Instead of reading an article as a way of evaluating the group's method of visualizing large numbers of images based on features such as brightness and hue, users can, in a sense, give it a test run. Users can accept the method as useful or find it inappropriate for their area of interest—but in either case, they have interacted with a visualization method put forward as of use in humanities contexts and have considered its merits in a way perhaps not afforded by a written article.

31 Among digital humanities scholars, there has been considerable interest in the epistemological properties of such built objects, from Willard McCarty's (2004, under "Conclusion") notion of modeling as "a continual process of coming to know by manipulating representations" to discussions of the rhetorical function of prototypes. Alan Galey and Stan Ruecker (2010) describe prototypes as making arguments, claiming the need for appropriate interpretation and evaluation. Stephen Ramsay and Geoffrey Rockwell (2012, 78) respond that once an object functions discursively, "the artifact has ceased to be a tool and has become something else." One value of the theories I have referenced from outside the humanities is their negotiation of this divide between rhetoric and utility; because a prototype functions, these theories suggest, it can make especially persuasive arguments. If the values and assumptions of systems are of interest, it may be the case that descriptions of proposed standards or technical articles are ill-

suited for discussing them. Working prototypes—written about in journals, presented at conferences, or made available, with documentation, for download—ask users to evaluate new methods in relation to existing projects and give the opportunity to enact various logics that may be otherwise inaccessible. Prototypes give users access to potential futures and provide a solid ground on which debate can happen; as Carl Disalvo (2009) notes, issues alone may not bring together an informed public for debate, but designed objects can communicate issues in a way that invites productive response. The prototype described here can be accepted, rejected, or modified in ways that theoretical arguments cannot—its propositions can, in effect, be downloaded, tested, forked, and modified. Design research is valuable for opening these spaces of critical analysis and discussion, and for imagining and presenting for evaluation alternative futures to the sociotechnical systems that underlie the current academic work of textual editing.

BIBLIOGRAPHY

Aoki, Paul M. 2007. "Back Stage on the Front Lines: Perspectives and Performance in the Combat Information Center." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 717–26. CHI '07. New York: ACM. http://doi.acm.org.ezproxy.lib.utexas.edu/10.1145/1240624.1240735. doi:10.1145/1240624.1240735.

Bański, Piotr. 2010. "Why TEI Stand-off Annotation Doesn't Quite Work and Why You Might Want to Use It Nevertheless." In *Proceedings of Balisage: The Markup Conference 2010*. Balisage Series on Markup Technologies, vol. 5. doi:10.4242/BalisageVol5.Banski01.

Bowers, John. 2012. "The Logic of Annotated Portfolios: Communicating the Value of 'Research through Design.'" In *Proceedings of the Designing Interactive Systems Conference*, 68–77. DIS '12. New York: ACM. doi:10.1145/2317956.2317968.

Bryant, John. 2002. *The Fluid Text: A Theory of Revision and Editing for Book and Screen*. Ann Arbor: University of Michigan Press.

Clement, Tanya. 2011. "Knowledge Representation and Digital Scholarly Editions in Theory and Practice." *Journal of the Text Encoding Initiative* Issue 1. http://jtei.revues.org/203. doi:10.4000/jtei.203.

Clement, Tanya, and Gaby Divay. 2012. "The Firstling/Erstling/He Complex by Baroness Elsa von Freytag-Loringhoven." *Scholarly Editing* 33. http://www.scholarlyediting.org/2012/editions/baroness/main.baroness.html.

Cummings, James. 2009. "Converting Saint Paul: A New TEI P5 Edition of The Conversion of Saint Paul Using Stand-off Methodology." *Literary and Linguistic Computing* 24 (3): 307–17. doi:10.1093/llc/fqp019.

DiSalvo, Carl. 2009. "Design and the Construction of Publics." *Design Issues* 25 (1): 48–63. doi:10.1162/desi.2009.25.1.48.

Dunne, Anthony. 2005. *Hertzian Tales: Electronic Products, Aesthetic Experience, and Critical Design*. Cambridge: MIT Press.

Fallman, Daniel. 2003. “Design-Oriented Human-Computer Interaction.” In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 225–32. CHI ’03. New York: ACM. doi:10.1145/642611.642652.

———. 2007. “Why Research-Oriented Design Isn’t Design-Oriented Research: On the Tensions Between Design and Research in an Implicit Design Discipline.” *Knowledge, Technology & Policy* 20 (3): 193–200. doi:10.1007/s12130-007-9022-8.

Galey, Alan. 2013. “Animated Variants.” *Visualizing Variation* (code library). Accessed May 6 2013. http://individual.utoronto.ca/alangaley/visualizingvariation/animated.html.

Galey, Alan, and Stan Ruecker. 2010. “How a Prototype Argues.” *Literary and Linguistic Computing* 25 (4): 405–24. doi:10.1093/llc/fqq021.

Gaver, William. 2012. “What Should We Expect from Research through Design?” In *Proceedings of the 2012 ACM Annual Conference on Human Factors in Computing Systems*, 937–46. CHI ’12. New York: ACM. doi:10.1145/2207676.2208538.

Höök, Kristina. 2010. “Transferring Qualities from Horseback Riding to Design.” In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries*, 226–35. NordiCHI ’10. New York: ACM. doi:10.1145/1868914.1868943.

Lessig, Lawrence. 2006. *Code*. New York: Basic Books.

McCarty, Willard. 2004. “Modeling: A Study in Words and Meanings.” In *Companion to Digital Humanities*, edited by Susan Schreibman, Ray Siemens, and John Unsworth. Oxford: Blackwell Publishing Professional. http://www.digitalhumanities.org/companion/.

Millerand, Florence, and Geoffrey C. Bowker. 2009. “Metadata Standards.” In *Standards and Their Stories: How Quantifying, Classifying, and Formalizing Practices Shape Everyday Life*, edited by Martha Lampland and Susan Leigh Star, 149–165. Ithaca: Cornell University Press.

Modern Language Association. 2011. *Guidelines for Editors of Scholarly Editions*. http://www.mla.org/cse_guidelines. Last revised 29 June 2011.

Odom, William, John Zimmerman, Scott Davidoff, Jodi Forlizzi, Anind K. Dey, and Min Kyung Lee. 2012. “A Fieldwork of the Future with User Enactments.” In *Proceedings of the Designing Interactive Systems Conference*, 338–47. DIS ’12. New York: ACM. doi:10.1145/2317956.2318008.

Ramsay, Stephen, Geoffrey Rockwell, and Matthew K. Gold. 2012. “Developing Things: Notes toward an Epistemology of Building in the Digital Humanities.” In *Debates in the Digital Humanities*, edited by Matthew K. Gold, 75–84. Minneapolis: University of Minnesota Press.

Ratto, Matt. 2011. “Critical Making: Conceptual and Material Studies in Technology and Social Life.” *The Information Society* 27 (4): 252–60. doi:10.1080/01972243.2011.583819.

Rosner, Daniela K. 2012. “The Material Practices of Collaboration.” In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work*, 1155–64. CSCW ’12. New York: ACM. doi:10.1145/2145204.2145375.

Schlitz, Stephanie A., and Garrick S. Bodine. 2009. “The TEIViewer: Facilitating the Transition from XML to Web Display.” *Literary and Linguistic Computing* 24 (3): 339–46. doi:10.1093/llc/fqp022.

Schön, Donald A. 1983. *The Reflective Practitioner: How Professionals Think in Action*. New York: Basic Books.

Schreibman, Susan, Amit Kumar, and Jarom McDonald. 2003. “The Versioning Machine.” *Literary and Linguistic Computing* 18 (1): 101–7. doi:10.1093/llc/18.1.101.

TEI Consortium. 2012. *TEI P5: Guidelines for Electronic Text Encoding and Interchange*. Version 2.2.0. Last updated October 25. N.p.: TEI Consortium. http://www.tei-c.org/Vault/P5/2.2.0/doc/tei-p5-doc/en/html/index.html.

NOTES

1. More information about the Versioning Machine is available at http://www.v-machine.org/.

2. In order to present a basic example of parallel segmentation, I've removed several attributes from this example and replaced one `<lem>`element with a `<rdg>` element.

3. The prototype, with documentation, is available at http://github.com/danielcarter/jquery.modVers.

ABSTRACT

This paper attempts to make two contributions to discussions related to TEI: (1) an analysis of how tools used for working with TEI documents encourage certain values and make certain assumptions about the work of textual editing and (2) a report on a methodological framework from outside the humanities that suggests a unique way to study such systems. Borrowing models of design research from the fields of design and human-computer interaction, I argue that prototypes can be used to create new conceptual knowledge, to investigate the values and assumptions of sociotechnical systems, and to communicate alternative visions of those systems. I first analyze an existing tool, the Versioning Machine, as a way of focusing the design of a prototype that reimagines several aspects of that original—specifically, I argue that the Versioning Machine creates an environment that to some extent assumes that TEI documents are created by one editor and intended for one instantiation. The prototype presented experiments with an alternative vision of textual editing as bringing encoded texts and interpretations together in multiple and flexible instantiations. Rather than a technical problem with an optimal solution, I approach this design process as an opportunity to ask how prototypes can give designers access to conceptual issues and allow users to enact alternative values and imagine alternative futures. This research was supported by the Modernist Versions Project, which is funded by a Social Sciences and Humanities Research Council of Canada Partnership Development Grant.

INDEX

Keywords: design research, prototypes, interface development, tool development, versioning, scholarly editing

AUTHOR

DANIEL CARTER

Daniel Carter has an MA in English Literature from The Ohio State University and is currently a PhD student in Information Studies at the University of Texas at Austin. He's online at http://www.daniel.inletters.com.