A Contextual Inquiry-Based Critique of the Strudel Web Site Maintenance System

Tessa Lau Jason Staczek
TR 99-01-01
Department of Computer Science and Engineering
University of Washington
Box 352350
Seattle, WA 98195-2350

January 21, 1999

1 Introduction

Over the last several years the World Wide Web (WWW) has become a preferred mechanism for organizations of all kinds to publish many different types of information. Organizational attempts to satisfy information consumers and compete for attention in a noisy information environment have caused web sites to steadily increase in size, complexity of structure, and sophistication of graphical presentation. The creation and maintenance of such large, information-rich sites is difficult, particularly given the expectation that published Web information be up-to-date at all times.

Researchers at AT&T have suggested that the process of web maintenance can be decomposed into three independent tasks: the selection of data from many (possibly dynamic) sources, the design of the structure of the site, and the design of the graphical layout of each page in the site [4]. At the same time, they note that existing web tools create interdependencies between these tasks. For example, manual WYSIWYG site design tools combine the selection, structuring and presentation tasks by focusing the user on the creation of individual HTML pages containing data, links to other HTML pages, and graphical layout. Database-driven tools such as Vignette's StoryServer and Lotus Domino separate the task of data selection from layout, but still combine site structure with presentation.

The AT&T group has implemented a web-site maintenance system called Strudel that attempts to fully decouple all three tasks. While Strudel does mitigate maintenance interdependence problems, it suffers from a database bias, and in particular sports a conceptually difficult text-based SQL user interface. Recognizing its potential, we proposed to investigate the issues involved in bringing Strudel from a research prototype to a commercially viable system. We chose to use contextual-inquiry based methods to gain insights into the current state of the art in web site maintenance, and use these insights to drive a critique of Strudel.

In the process of conducting the interviews, it became clear that the class of potential Strudel users was technically proficient and most would likely be capable of using Strudel's query language as is, with sufficient instruction. However, we were surprised to find that the task decomposition concepts underlying Strudel's design were not evident in current web maintenance practice; although users realized that data was independent of its presentation, they conflated structure with graphical layout. We were also surprised to find that many of the issues raised were in fact not technical.

This paper presents the results of our interviews, condenses the results into a number of issues involved in web site maintenance, and analyzes Strudel in terms of how well it addresses each of these issues. We begin with a description of our experimental methodology.

2 Methodology

We decided on an approach based on contextual inquiry to investigate current solutions to the problem of web site maintenance. Contextual inquiry [2] is a method for achieving user-centered design. We began by conducting interviews with people involved in the web site maintenance process. We attempted to focus on existing work practices by watching users in normal work environments. We asked users to describe their work, asked them to demonstrate some of the software tools they used, and observed the work environment. We were concerned with their opinions and conceptions about the web site maintenance process, and solicited opinions about problems with their current system.

We interviewed six people involved in web site maintenance: Betsy Raleigh, Melody Winkle, Ginger Brower, and Alexis Raphael from the University of Washington's C&C Web-Guides team; Jim Kerr from Active Voice; and David Joslin from Adobe Systems Incorporated. The sample includes representatives from both educational and commercial institutions, and various types of web sites. The UW web site disseminates information of interest to the 18,000 employees of the University, such as benefits information and payroll policies; most of this information is in the form of policies and processes mandated by the state of Washington. The Active Voice web site publishes information about the company, its products, job openings, and information of interest to potential investors. Adobe maintains a large web site; we focused on the subsection devoted to product support. We picked this cross-section of web sites to better see the web maintenance problem from a variety of perspectives.

After conducting several interviews, our CI team regrouped and synthesized a model of the work practice based on our experiences. We went over our interview notes and generated a plethora of sticky notes, each describing one aspect of web site maintenance. We then went through the stickies one by one, discussing and clarifying what each one meant, and grouped them together loosely based on similarity. We named each of these groups to reflect its topic. When finished grouping the stickies, we had collected fourteen topics.

To organize these topics, we grouped the topic-level stickies further into meta-groups. These meta-groups coincided well with Strudel's division of the site maintenance process: content, site structure, graphical presentation, maintenance, and general comments on the process as a whole. We use these meta-groups to structure our summary of the interview results in the next section.

3 Contextual Interviews

Our interviews covered all parts of the web site maintenance process, from the technical side of taking data and publishing it on the web, to the social and cultural issues involved in teaching people how to take advantage of the web. To a large degree, we were able to divide up the comments among the three tasks in web site maintenance advocated by Strudel. We discuss each of these in turn, then address the comments that span more than one aspect of the maintenance process. Along the way, we highlight the major issues we've identified in the process.

3.1 Web site content

The people we talked with identified several types of content. Most of these were originally on paper, and are being converted to the web. One member of the WebGuides team spoke of the foundational data that forms the bulk of the content of several administrative web sites. This foundational data includes text documents, such as policies and benefits information, that are legislated by the state capital. Other sources of content include Word documents, operations manuals, and information exported from Lotus Notes. In the latter case, the data is cleaned up and edited by Perl scripts before it is placed on the web.

A second type of content is links to existing services, often telnet-based services. This is related to the third type of content, administrative processes: for example, the travel web site is organized as a linear sequence of web pages that steps one through the process of arranging a trip.

The main problems identified in this topic fall into the category of converting paper documentation to the web. No one wants to do up-front work on document design; they'd rather migrate existing paper designs directly to the web. However, the web presents conceptual challenges to some users.

Issue 1 *Pointers to shared documents are a difficult concept for users to grasp.*

In particular, users seemed to have difficulty understanding that a link points to a file only, the contents of which could change without affecting the link. Not using shared documents can cause maintenance headaches: for example, a set of documents which all include the same list of phone numbers would each have to be updated when a phone number changes.

Interviewees agreed that the web is better than paper (since paper is expensive and ineffective). However, they pointed out that there is no good tie-in between paper materials and web materials—getting paper documents online is a problem.

Issue 2 *People have difficulties converting existing information to a format suitable for the web.*

Another issue in designing for the web is that good web design requires awareness of file and directory names, since on the web these are visible to the user.

Issue 3 *Directory names and structure should be chosen carefully since they will be visible in the browser's URL window.*

The idea of "owning content" was also prevalent.

Issue 4 *The people who own the content should be responsible for publishing it on the web.*

Not only are content owners then motivated to learn about the web and HTML, but they are also more invested in keeping the information on their web site up to date.

3.2 Site structure

The next phase of the process is to organize this content into a structured web site. Structuring happens on several scales. At the lowest level, related information is grouped into a single file; for example, employees are usually eligible for medical, dental, and life benefits, so the three sources are grouped together onto a single web page.

Issue 5 *The level of granularity is typically at the file/page level.*

The web sites we surveyed ranged from hundreds to hundreds of thousands of files. At one site, 100-300 documents are modified or added per week; at another, foundational data changes once a year.

Directory structure is often used to group related information together to facilitate maintenance. Methods of deciding on a directory structure include alphabetical, logical (grouping related topics together), and political (grouping based on organizational boundaries). Directory structures often go no more than two levels deep; in one case, all files are kept in a single directory and prefixed to indicate section membership. Another site keeps all graphics in the root directory to facilitate location and maintenance.

Directory structure usually corresponds to site structure, which is yet another level of organization.

Issue 6 *Site structure is designed using an incremental, interactive, collaborative process.*

The process was different at each site. The UW web site was designed with the aid of physical artifacts such as a tree structure drawn on whiteboards. In one office, two walls are covered in post-it notes arranged into columns. Yellow post-its represent subtopics, and the yellow columns are topped by purple notes designating topic headings.

Another group began by arranging existing paper documents on a large table. From these documents, topics were proposed and written on index cards that were stuck to the wall. The cards were grouped into clumps, using string to simulate links between groups. Each clump was then named. As a rule of thumb, one person stated that web sites typically converged to 7 or 8 clumps—any more than that suggested a need for reorganization.

Issue 7 *Changes to the directory structure can result in broken links.*

One site plans to never change the directory structure (or, if it must change, they plan on being able to redirect linkers to the new page).

One person noted that page layout often matched the directory structure, which brings us to the next section on graphical presentation.

3.3 Graphical presentation

The next step in the web site maintenance process is to realize the site structure with an HTML presentation. None of the people we talked with used Microsoft's FrontPage or other GUI-based HTML-editing tools; one person stated deprecatingly that such tools might be useful for novices constructing their own personal home page, but not for large sites such as theirs. In practice, Pico, vi, and BBEdit (for the color syntax highlighting) are the text editors of choice for editing HTML. Accessibility is the primary reason why text editors are preferred to WYSIWYG tools; WYSIWYG tools can't be trusted to display what will be seen on individual browsers. Others pointed out that WYSIWYG tools don't help with Javascript or getting user input from forms.

Issue 8 *Tools are needed to help design interactive elements such as forms and Javascript.*

Standard WYSIWYG tools have another flaw: they don't typically enforce any design guidelines on the generated web pages. For example, a member of the WebGuides team pointed out that they try to make their sites obey certain guidelines, such as ensuring that information can be located within three clicks from the starting point, putting important information at the top of the page, and making sure that pages load quickly.

Issue 9 *Page layout tools should enforce design guidelines on web page layout.*

Multiple views can be used to present specialized pages for different browser types. The UW site is unique in using different style sheets for different browsers; neither of the other sites did any special casing on browser type, although they do test their site on a small list of supported browsers (typically the latest versions of Internet Explorer and Netscape).

Issue 10 *Multiple views of a web site can be a problem.*

The UW also uses a meta-HTML language called CHTML [3]. Among other features, CHTML has a conditional construct that allows one to define views (which are called scopes) to serve to people connecting from different domains. The three scopes used by the UW web site are UW, NWNet, and World. These scopes exist mainly to provide access control for databases which the University subscribes to; people inside the University can use them free of charge, while the NWNet scope exists because of an agreement to provide access to selected medical databases. Use of this conditional construct causes three versions of the same web page to be generated from a single master copy. These three versions are replicated on the production web servers.

3.4 Maintenance

Once a web site has been created, it must be kept up to date as information changes. As mentioned earlier, this is generally done by editing the HTML directly. However, all recognized that this is a difficult problem in general. The single maintenance person at one site admitted that the site was always out of date because he was the bottleneck.

Issue 11 *Maintenance of content is facilitated by being decentralized and having the content owners be responsible for updating the information on the web.*

The WebGuides team requires that their clients produce maintenance plans—decide how the web maintainer should receive notice when information must be changed, such as through meetings or mailing lists.

One interviewee mentioned a content provider who had developed an update model that required no interaction with the webmaster: individual article authors simply updated their content in a database (without any knowledge of HTML) and it was automatically formatted and placed on the web. Another webmaster proposed allowing clients to email content to a robot that would update the site.

Issue 12 *Maintenance of global or duplicated data is difficult.*

In general, the maintenance problem covers any type of update to a web site, including structure reorganization and changed data. A more specific problem has to do with updating replicated information. One person stated that a full-featured editor with multiple-file grep was essential for maintaining the site. Another mentioned implementing global search and replace using custom shell scripts. In some cases, the UW's CHTML language is used to alleviate this problem, for example, to keep page headers and footers consistent. However, a member of the WebGuides team admitted that CHTML wasn't used everywhere it could be.

Most people mentioned using HTML templates. Page creation typically starts by editing an existing template, which can include global shared items.

Issue 13 *Web site expansion is done by reusing existing pages and components, not by creating new elements from scratch.*

For example, the UW team creates new pages by importing standard header and footer directives into a blank text file. These directives place site-standard navigation toolbars at the top and bottom of the page, and establish the area in which new content should be inserted. Pages are maintained and restructured by hand using a text editor as well.

Issue 14 *File locking, access control, and revision control are important.*

One site used no formal revision control (the latest version was always on the web site). Another recognized that if a second web maintainer were hired, locking and access control would become an issue.

3.5 General comments on all parts of the process

On the whole, the people we interviewed were technically savvy, were proficient in HTML, and had a good understanding of how the web works (e.g., as file transfer from a remote machine to the browser). However, as sites grow, content will become decentralized and spread over many parts of the organization. Hence, training will become more of an issue.

Issue 15 *Introduction of a new tool will require training.*

The WebGuides team manages a large site whose content is distributed among many client organizations. As a result, one of their functions is to train representatives from other departments in how to structure and lay out their own web sites. While this training process is partly technical (teaching HTML, UNIX commands, etc.), a big part of it is teaching people how to communicate more effectively and how to take advantage of the web (e.g., linking to shared documents rather than replicating data). Users are naive; while some are very interested in learning about the web publishing process, others are very afraid.

Training is a problem in the other sites as well. For instance, one person noted that it might be difficult to get budget for training when introducing a new tool across department lines.

The next section introduces the Strudel system and analyzes how it addresses each of the issues brought up in this section.

4 Strudel

In this section we examine Strudel to determine how it addresses the main issues uncovered in the contextual inquiry. We begin with a brief overview of Strudel, then discuss the issues raised in the previous section.

4.1 Strudel architecture

Strudel is a web site development system created by AT&T Research. In contrast to other web site management systems, Strudel takes a database-oriented view of web site creation and maintenance activities. Its conceptual model is quite simple—a web site is built in three phases:

- Information to be displayed at a site is aggregated from various data sources.
- The site structure is defined in terms of the available data.
- The structured data is formatted into HTML for final presentation to the user.

This model provides several advantages. It allows the site structure to be described in a compact form which actively supports the creation of multiple web site views from the same source data. It also completely decouples a site's visual presentation from its content and structure, allowing visual design to proceed separately from data collection and structure.

The Strudel model acknowledges that information on web sites may be drawn from many sources, such as relational databases, Excel spreadsheets, Word documents, text documents or even other HTML files. To insulate the site designer from the details of each data type, Strudel incorporates a uniform data model. Using simple conversion programs known as wrappers, source data is converted into Strudel's own format, a labeled directed graph representation of a semi-structured data model [1] known as a data graph. This data model (Data Description Language, or DDL) can flexibly accommodate arbitrary data types and is quite robust in the face of missing or incomplete data. In general, each data source type requires its own wrapper, although wrappers for a data source type need only be produced once. All subsequent Strudel site operations are performed on the DDL representation.

Having aggregated the raw data sources into Strudel's DDL model, web site design proceeds by specifying the site's structure by contructing a query in the Strudel query language (StruQL). An example is shown in Figure 1. Queries consist of two major portions. The first is a selection clause (known as the WHERE clause) that uses SQL-like syntax to specify which portion or portions of the available data are to be included in the site. The results of applying this query to the database are a set of data objects that may be operated on in the second clause, the graph building, or LINK clause.

```
input phone
link RootPage() -> "People" -> PeoplePage()
         RootPage() -> "Phone Numbers" -> PhonePage()
{
         where Employee(x), x -> "Phone" -> v
         link PhonePage() -> PhoneNode(v)
}
output phonesite
end
```

Figure 1: An example Strudel query.

Formally, the LINK clause allows the user to create an arbitrary new data graph based on the input data graph. Intuitively, the LINK clause allows the user to specify exactly how objects in the site are to be related. For instance, it may specify that the site is to contain a page called `Phone Numbers` that contains names and phone numbers for all objects of type `Employee` in the database. Note that at this step, there is still no mention of how the data will be formatted into HTML. The output of this step is known as the site graph, itself a DDL object.

Having defined the structure of the site through relations between its data objects, the remaining step is conversion of these objects to HTML format. Strudel introduces an HTML formatting language that allows users to build HTML templates that include code to direct the embedding of objects resident in the site graph. In Strudel's parlance, this step is known as materialization.

The formatting step is quite flexible, as each object in the site graph may be embedded in a template according to its type or its membership in DDL collections. This flexibility, coupled with conditional constructs in the HTML template language, allows for very complex data-driven layout.

4.2 Strudel and Issues Found in Current Practice

In this section, we categorize the main issues resulting from the contextual inquiry interviews. We evaluate Strudel's response to the issues within each category.

4.2.1 Concept/Model Issues

Two specific issues arose with regard to user models of the web-site creation and maintenance tasks.

(**Issue 1**) *Pointers to shared documents are a difficult concept for users to grasp.* While the Web is built on this very structure, we found evidence that designers found this to be quite a tricky concept. Extrapolating, this suggests that new users will face difficulty with the Strudel model at several levels.

At the data level, Strudel's use of a DDL data model forces users to think of data as a set of linked objects. Further, DDL allows links to existing objects as well as atomic values, inviting the possibility that the Strudel representation of the data underlying a site might itself be more complicated than than the non-Strudel HTML representation of the same site.

At the structure level, StruQL requires mental manipulation of link structures in both the selection and graph creation steps. In the WHERE clause, data is selected using path descriptors that specify link traversals in the data graph. In the LINK clause, site structure is built by explicitly adding new objects with links to existing objects.

Finally, at the presentation level, the user is faced with the additional problem of distinguishing links in the site graph from links that will exist in the materialized HTML form of the web site. This is complicated by the fact that objects may contain multiple links of the same name and type to different instances of an object or atomic value.

The Strudel model builds on and amplifies a key user conceptual barrier. While this model provides much of Strudel's advantage, we believe that a successful implementation of Strudel must be able to hide the details of this data model from the user, and perhaps reveal it gradually as the user becomes accustomed to working with Strudel operation.

This may be an opportunity to exploit Strudel's leniency in allowing data to exist in HTML templates. For example, an existing site could be migrated trivially to Strudel by converting all HTML pages to Strudel HTML templates. As the user becomes familiar with Strudel, it may be possible to gradually and automatically move data from the templates to a DDL representation while allowing the user to oversee the process.

(**Issue 13**) *Web site expansion is done by reusing existing pages and components, not by creating new elements from scratch.* At all interview sites, we found evidence that new HTML pages are created by copying and modifying existing pages. This process appears to arise out of convenience, but it also seems to provide a level of comfort for the designer who knows that the new page will begin its life syntactically correct and conformant to the graphical requirements of the site. Also, the copy step produces an immediate artifact, in essence a placeholder page that may be edited to its final form at the designer's convenience.

Strudel appears to lack the ability to directly support this mode of operation. While new pages can be added to a site by copying and modifying an existing data object, this operation might also require a modification to one or more queries and HTML templates that define the structure and presentation of the data represented by that object. Essentially, Strudel's model expressly disallows manipulation of the final artifact and forces editing its abstract components.

This suggests that a Strudel user interface incorporate a mechanism to allow information to flow back from a page instance to the DDL, query and HTML templates that could be used to create it. For instance, one can imagine a "copy page" operation that simultaneously

creates a new page artifact and updates the underlying query and DDLs as necessary.

4.2.2 Data/Wrapper Issues

Wrapper issues plague all systems that attempt to integrate data from many sources. In current practice, we observed that site builders are actively involved in converting source formats to HTML. The Strudel model, however, introduces yet another conversion layer—source data to DDL.

(**Issue 2**) *People have difficulties converting existing information to a format suitable for the web.* All interview sites experienced difficulty converting existing data into publishable HTML. In many cases, there was a desire to take existing paper designs directly to the web, specifically to avoid the effort required to rework documents that had already been satisfactorily formatted, and that may have been expensive or difficult to produce. In other cases, raw data was already available online, and was converted to HTML through processes involving database exports and custom-built scripts.

In the case of unformatted, electronic data, Strudel has the potential to solve many of the observed problems. The flexibility of DDL and the power of its HTML template language would in many cases be sufficient to replace custom scripting. In essence, current script behavior could be converted to Strudel, with some script behavior implemented at the wrapper level and other behavior implemented in the HTML template language. We believe that users will embrace this flexibility, something not offered in the less customizable "export to HTML" features of current database systems.

The situation is less favorable with respect to formatted documents. Formatted paper documents tend to be created in applications such as Adobe PageMaker or Microsoft Word. Increasingly, these applications offer "export to HTML" features that allow a single formatted source to drive both paper and web output. We saw some evidence of this in current practice and would expect that it would increase as HTML exporting features become more prevalent and the HTML language adapts to allow more sophisticated layout control.

Strudel's separation of data, structure and presentation make it difficult to integrate highly designed documents into the system at the DDL and query levels. Given highly formatted source, it's not particularly desirable to attempt a decomposition into Strudel abstractions. Further, Strudel lacks the tools to do interactive graphic design, and probably would not displace established layout tools even if it did. While it may lack the ability to produce highly formatted source, Strudel makes no restrictions on including pre-formatted HTML as a portion of the web site. Strudel remains useful in these cases, perhaps through the use of dummy data in the DDL model to trigger the inclusion of various preformatted HTML documents.

(**Issue 5**) *The level of granularity is typically at the file/page level.* In a topic related to the practice of "copying and editing" and pointer manipulation, we found that most users tended to deal with information in chunks at the level of single HTML files. In fact, the custom scripts tended to operate on HTML files or groups of HTML files.

The Strudel model suggests that much smaller units of data (DDL objects) be used to build a web site, but it is a suggestion only. For instance, the HTML language enforces no rules about embedding data directly in HTML templates. In practice, this ambiguity allows a particular instance of a web site to be realized through many different combinations of

data graph, site graph and HTML templates. The system provides no clear indication of which method might be superior for any application.

4.2.3 Process Issues

Many of the issues uncovered in the contextual inquiry interviews regarded process and cultural practices rather than technical problems. In particular, we noted that site design, creation and maintenance are collaborative and distributed processes, and that acceptance of tools will be driven in part by their ability to support these methods.

(**Issue 6**) *Site structure is designed using an incremental, interactive, collaborative process.* At all interview sites, we found that initial web-site design is done by committee, typically using paper, index cards or a whiteboard to sketch a site structure proposal. This structure is iteratively refined over some period of time. The key features of these processes seem to be

- a graphical, easily editable representation of the site structure under design, and
- the ability for all interested parties to view and comment on the evolving design

Strudel makes no particular commitment about how site structure should be designed, and its use should not interfere with existing collaborative processes. However, current models tend to illustrate the final physical form of the site as represented by links between HTML documents. This top-down design method is somewhat at odds with Strudel's notion of data-driven bottom-up site design. It suggests that a Strudel implementation should include tools to assist in developing the bottom-up schema from a top-down specification.

(**Issue 4**) *The people who own the content should be responsible for publishing it on the web.* (**Issue 11**) *Maintenance of content is facilitated by being decentralized and having the content owners be responsible for updating the information on the web.* At all interview sites we noted the strong desire on the part of webmasters that content owners should be responsible for publishing their own information. In fact, sites tended to be quite out of date due to the bottleneck established by the webmaster. The Adobe site was a notable exception. Their use of an automated database process allowed multiple authors to contribute new data to the web site on a continuous basis. However, the Adobe webmaster noted that when design (as opposed to content) changes were required, the automated process was forced to halt, often for several days, in order to test the effects of the changes—mainly due to a two hour site export/create cycle time.

This is a tricky area for Strudel. The current Strudel model makes absolutely no commitment about how data, queries and templates come together to build a web site. The system can easily support a process that allows individual contributors to supply various pieces of the data, queries or templates that drive the system. The issues, however, are similar to those faced by software development groups sharing access to a large code base. Strudel's model forces strict dependencies between wrapper output format, query syntax and references and variable reference in HTML templates. Any implementation of Strudel designed to support group access must provide some level of safety features to prevent users from inadvertently breaking connections between wrapper, data, queries and templates. In light of Adobe's experience, it is probably not sufficient to require regenerating and retesting the

entire site to verify incremental changes. Exactly how Strudel might support such features remains a significant open question.

4.2.4 Technical Maintenance Issues

All interviewees noted technical problems with respect to routine maintenance of existing sites. We discuss two widespread problems here.

(**Issue 7**) *Changes to the directory structure can result in broken links.* Strict adherence to Strudel's bottom-up design paradigm will guarantee that links within a site always remain intact. In practice, we found that site maintainers also had to deal with links into their site from external locations. In general, we found that site maintainers preferred to treat external linkers kindly, and redirect out-of-date links correctly as internal site structure changed.

As with existing tools, the base Strudel system makes no explicit provision for maintaining external links, and in fact might complicate the situation because its architecture makes it easier to change a web site's structure at will. In contrast to other systems, Strudel may be uniquely poised to to encode the existence of external links, perhaps through shadow data objects that aren't realized at materialize time. Documentation of these access points could facilitate automatic maintenance of these external links.

(**Issue 12**) *Maintenance of global or duplicated data is difficult.* Most interview sites reported some need to employ scripts to periodically make small changes to many HTML files simultaneously. Typically, this was done to revise addresses or phone numbers that should have been shared, but were embedded in HTML for some reason. Even sites that reported using CHTML to include common header and footer information noted some problem with this.

Once again, strict adherence to Strudel's design methodology will render this problem solved. However, Strudel's HTML template language is quite lenient, and allows users to place data in templates that should perhaps reside in the data repository. Problems of this type can be avoided through good documentation and user education.

4.2.5 Software Feature Issues

The interviews revealed that site designers often go to great lengths to build custom tools to allow them to circumvent feature limitations of currently available authoring applications. In some cases, designers rely on manual methods where custom tool design is not feasible. This section discusses several examples in this area.

(**Issue 8**) *Tools are needed to help design interactive elements such as forms and Javascript.* Our interviews found participants unwilling to use existing WYSIWYG tools due to poor support for active content of any kind. Strudel makes no particular concessions in this area. One can imagine that insertion of existing active content could be supported at the HTML template level, or by representing active content in the semi-structured data model, perhaps by attaching active elements as file attributes of existing data objects. This improves control and the ability to share code, but leaves open the problem of design assistance for these elements.

(**Issue 9**) *Page layout tools should enforce design guidelines on web page layout.* As with interactive elements, Strudel provides no mechanism for enforcing design guidelines on

the materialized version of the web site. The proposed research direction of incorporating constraints into Strudel may someday address this problem.

(**Issue 3**) *Directory names and structure should be chosen carefully since they will be visible in the browser's URL window.* Strudel may be at a disadvantage with respect to this issue, since the filenames in the materialized web site are typically automatically generated.

(**Issue 10**) *Multiple views of a web site can be a problem.* Several interviewees expressed a desire to generate multiple web site views to accomodate users from different domains or users with certain disabilities. This is perhaps Strudel's strongest selling point. Having converted an existing site to the Strudel model, producing multiple views is as simple as creating and modifying the base site query. Testing issues remain open, although integrity constraint checking on the finished web may be made easier through techniques under development by the AT&T group [5].

(**Issue 14**) *File locking, access control, and revision control are important.* Although none of the interviewees used a version or access control system in their development, they acknowledged that such control would be required as their sites grew and more people contributed to them.

Strudel makes no particular commitment in this area, and in fact presents additional difficulties. In general, a site's declarative representation is very compact compared to its physical materialization. Small changes in the site query can have a large impact on the physical site. So, while revision control at the query level remains important, a review of query changes alone does not permit a rapid survey of changes to the materialized site. Further, Strudel requires revision and access control on wrappers, data and HTML templates in addition to the query, and all must remain synchronized.

Given Strudel's dynamic nature, these issues are closely related to the problem of efficient site materialization discussed in [6]. Their solution will depend on the approach taken to incremental site evaluation.

4.2.6 Training and Support

(**Issue 15**) *Introduction of a new tool will require training.* Our interviews found that attempts to cross organizational lines with software tools can be difficult and costly. For instance, a marketing department may be accustomed to supplying the webmaster with paper documents for conversion to the web site. Supplying the marketing department with tools to streamline this process might require significant disruptions to the cultural fabric, including budget allocations, training time, application support time and a general change in work processes. In one extreme case, we noted that the webmaster was required to obtain source applications for documents of all types received from other portions of the company. The support costs and training burdens were contained at the expense of overall process efficiency.

Strudel's complexity places a unique burden on organizations that may wish to adopt it. Given its novelty, the system does not yet have well established work practices, so there remains an opportunity to partition it to allow various site maintenance participants to interact only with the portion relevant for their work. For instance, content providers could use a simple data submission application that interacts with the data repository, but leaves query construction to the webmaster. Reducing Strudel's profile in some fashion may be a

key factor to its adoption in the workplace.

5 Conclusion

We have conducted a contextual inquiry into the current state of the art in web site maintenance. Our insights have allowed us to determine a number of important issues in the process. We have critiqued the Strudel web site management system using these issues, determined at what points in the process Strudel would be most helpful, and suggested extensions to the system that would make it more useful in current practice.

References

[1] Serge Abiteboul. Querying Semi-Structured Data. In *Proceedings of the International Conference on Database Theory, ICDT '97*, pages 1–18, Jan 1997.

[2] Hugh Beyer and Karen Holtzblatt. *Contextual Design: Defining Customer-Centered Systems*. Morgan Kaufmann, 1998.

[3] *Conditional HTML*. `http://www.washington.edu/webinfo/chtml.html`.

[4] Mary Fernandez, Daniela Florescu, Jaewoo Kang, Alon Levy, and Dan Suciu. Catching the Boat with Strudel: Experiences with a Web-Site Management System. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, SIGMOD '98*, pages 414–425, June 1998.

[5] Mary Fernandez, Daniela Florescu, Alon Levy, and Dan Suciu. Reasoning About Web-Site Structure. Draft, 1998.

[6] Mary Fernandez, Daniela Florescu, Alon Levy, and Dan Suciu. Warehousing and Incremental Evaluation for Web Site Management. Draft, 1998.