Client Documentation
Student Writing Support XML Project
Liv Riggins
10/25/19

Background

In the previous project, I redesigned the website for Student Writing Support (SWS) according to their needs and preferences for a simpler, more student-oriented site. This website has several pages of additional online writing resources for students, all of which are very similar in form and content. The list of resources is continually updated and expanded, and has lots of data in it, so it is a good candidate for organization via XML.

Client Use for XML

I saw a potential need for structured authoring both in SWS's consultant biography grid and in SWS's list of online help resources. I initially sketched out DTD hierarchies for both, and decided to go with the list of online help resources because the content and data that SWS needs there is more consistent than the content and data needed for the consultant biography grid. Additionally, SWS prefers to have consultants write their own bios, and so structured authoring would likely not be that helpful in the actual authoring of the bios. In contrast, the list of online help resources is more consistent and authored exclusively by the SWS webmaster, Jasmine. Finally, the online resource lists could be reformatted for distribution via PDF, physical handout, or in other online contexts.

The XML I developed is oriented towards capturing key data for SWS's internal operations, creating the potential for a search/sort function by several different characteristics, and allowing for easy republication in different forms. One of the current weakness of SWS's online library is lack of a search function (beyond command+F), high volume of data, and limited information on form and subject of each resource. As such, I developed this XML hierarchy to track both the current data and data that would be needed to have a basic search/sort function, envisioning potential for content customization and personalization (Anderson, 2014).

I had to use a couple of workarounds to get the elements and features needed for this type of document. For example, I wanted to limit the subject and format to a set of defined options that could be linked to categories/keywords for a search/sort function, and so created an empty element that could contain multiple attributes with the defined set of options correlating to potential keywords. I also entered URLs as CDATA, because some had characters that were not allowed as #PCDATA but the information itself was necessary for the online resource library and any print handouts.

Potential Future Uses

XML and structured authoring provide several new options for SWS help resource publishing and reuse across multiple platforms. SWS currently has the online resource library and a few print handouts available in the SWS offices based on the online resource library listings. With structured authoring for the online help resource library, SWS could publish to a variety of online platforms, easily create groupings different than the current categories and based on type of document, like a list of databases for writing help or a list of helpful templates, or on subject, like a list of resources for formatting or a list of resources for professional writing. These could then be published in print or online.

Additionally, these tags and categorization create a potential structure for a search/sort function that would allow students to generate their own help lists based on their needs. With a search/sort function, students would be able to dynamically pull information for their search purposes based on the same categories that would allow SWS to output different lists of help resources.

Finally, creating an XML document for this content makes it so that the content could be easily replicated in a writing help app, were SWS to develop an app for their services. As students especially move towards a mobile-focused experience of the internet, SWS could consider creating a mobile app for appointment scheduling and online help. Structured XML content could then be published to this app with minimal changes to structure/format.

Process Documentation
Student Writing Support XML Project
Liv Riggins
10/25/19

Initial Experience with XML

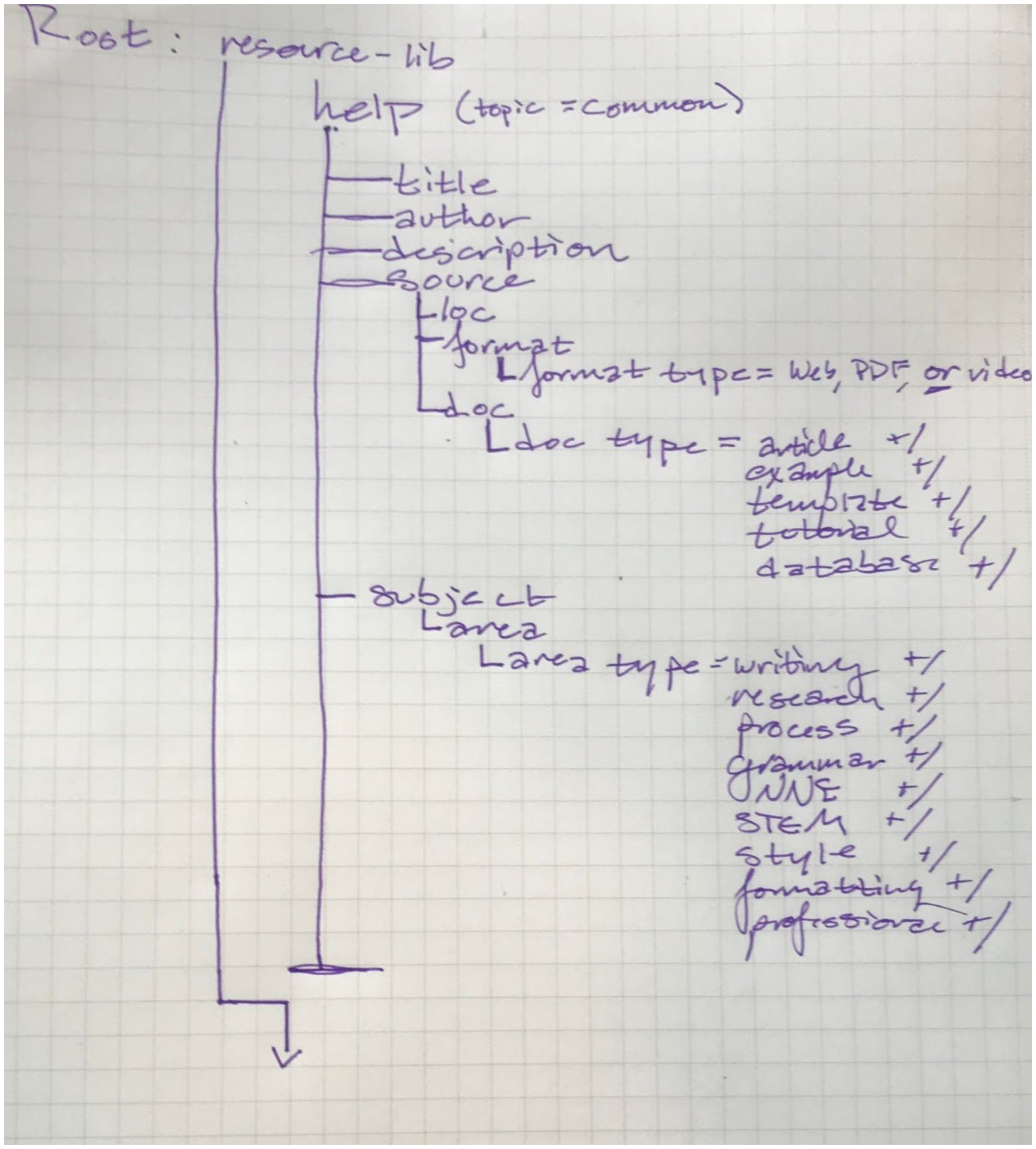
I had no experience with structured authoring before besides using Arbortext to author documents, and so struggled with the concept and the language. I think learning XML was further complicated for me by my experience with both HTML and DITA, as I made assumptions about how XML worked and what options were available in XML based on my experiences with HTML and DITA.

The purpose of each of the XML tag types was hard for me to get my head around, as well as the interplay between the DTD, CSS, and XML. The LinkedIn Learning videos for this one confused me more than helped; I didn't know if we had to learn to do an XML schema or not, and all the terms got jumbled in my head. It was also hard to see how the business card example could translate to other document types.

Document Analysis and Design

I started to be able to conceptualize what the roots, elements, and attributes were and did in XML from Tina's example and description on the WRIT 4662 resource site. I sketched out the hierarchies on paper for both a possible XML for consultant biographies and one for the online help documents, and chose to go with the online help documents based on my analysis of the client's needs and document functions, as Anderson recommends technical communication professionals do to create effective CMS systems that add value to the client's content (Anderson, 2014). As described more fully in my client documentation, I created a CMS system that provides semantic categorization for existing content, and adds new categorizations with an eye towards dynamic document generation by both the student end users and SWS.

I chose to create a hierarchy based on the root element of <resource-lib>, as this is an accurate title for all the documents in the Student Writing Services (SWS) website's online help area. As can be seen in the image on the next page, I defined <help> as the next element, which must be modified by topic of (common|STEM|grad|multi|grammar|research|genre|process), which correlates to the current groupings for help resources on the SWS website. The <help> element then contains the child elements of <title>, <author>, <description>, <source>, and <subject>. The <source> element contains child elements on the document URL, the general form of the source (web|PDF|video), and the specific format of the document content (article|example|template|tutorial|database). The <subject> element contains the element <area>, which is used to define several subject area types from which document writers should choose.



Challenges

However, once I had this general concept down, it was hard to jump to putting it into practice and understand how the XML writing correlated with the functions and concepts I had in mind.

For example, I wanted to provide a set of limited inputs that the document writer could, but would not have to, choose multiple options of to tag content and provide potential for dynamic document outputs based on characteristics including different subject area, source format, and source document type. I wanted to have a limited list of keywords so that the inputs would be limited and therefore useful in future document uses like dynamic publishing and user-generated lists, yet have enough flexibility to allow for accurate and useful categorizations. I was envisioning something like a picklist, but couldn't find a way to make this work in XML.

I emailed Tina with my questions and she confirmed that it was not possible to do exactly what I hoped to do with one element and attribute list. However, she gave me the solution of creating an element with a + designation for one or more occurences, an empty child element below that, and then the choices I wanted to provide as options included in the attribute list attached to the empty element. I could then add one or more of these attribute options to the document itself in the form of:

```
<subject>
        <area type= "style"/>
        <area type= "writing" />
  </subject>
```

Once I had this down in the DTD, it did take me a couple of tries to find the right way to express it in the XML, as I did above. It was frustrating to continue being told there were errors when I tried to validate it, but I looked carefully at my DTD and XML and found where I had tags in the wrong order or attributes incorrectly written and fixed these errors. I used W3 Schools to help me find my errors.

W3 Schools also helped me realize that I had to include the URLs for documents as CDATA because of the characters in them. URLs are important information for these documents, so I changed the element type declaration to ANY for the element <loc> in the DTD, and then entered URLs as CDATA in the XML document.

I added basic CSS styling to the XML document on Friday, and learned how to make an attribute show as text in the document with CSS styling. I didn't know this was possible until I looked it up on W3 Schools, and found that I could use the content property to make the attribute appear as text. This way, the keywords I created to define documents were available to users as well.

Future Development

Through this project, I learned about what was possible in XML and, just as importantly, what was not possible in XML. By the end of the project, I was able to understand the relationship between the DTD and the XML, and how crucial being able to define content categorization is for deciding what the output will be. Structured authoring like XML gives the potential for exciting reuse and output options, including user-generated help guides, several different publishing formats for SWS itself, and the potential to easily transfer this content into new formats ranging from apps to paper handouts (Anderson, 2014). I still feel that I do not fully understand the interplay between attributes and elements in XML as I feel I do for HTML, but now have a basic idea of how to use both in a DTD to fulfill client content management needs.