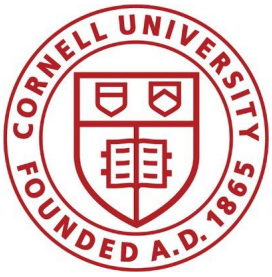
Save The Constitution

A Project For the Legal Information Institute



CS 5150 : Progress Report III

April 13, 2018

Table Of Contents

1. Introduction

The client for this project is Thomas Bruce, Research Associate and Director at Legal Information Institute at Cornell Law School (LII), (tom@liicornell.org). Advisors are Craig Newton, Associate Director for Content Development at LII (craig@liicornell.org) and Sara Frug, Associate Director at LII (sara.frug@cornell.edu). Technical advisors are Sylvia Kwakye (sylvia@liicornell.org) and Nicholas Ceynowa (nic.ceynowa@liicornell.org) from LII.

The task is to extract data from the 2017 PDF version of the CONAN (The Constitution of the United States, Analysis and Interpretation) to make legal information more accessible for LII and the public.

There are two major steps in this project :
1. PDF to XML conversion of CONAN, which involves developing an XML schema and XML File, and extracting footnotes from the PDF
2. Resource description framework (RDF) implementation, which involves developing RDF models and building RDF repositories

2. The Team and Roles

Given the time constraint, we are now completing both steps of the project in parallel instead of chronologically. Three members are working on the XML conversion step and two members are working on the RDF repositories implementation step.

XML :
- ❏ Brendan Rappazzo (bhr54@cornell.edu)
- ❏ Maxwell Anderson(mga58@cornell.edu)
- ❏ Taira Davey (td284@cornell.edu)

RDF :
- ❏ Anusha Chowdhury (ac2633@cornell.edu)
- ❏ Garima Kapila (gk347@cornell.edu)

Please note that this division was done to keep each of us responsible for managing a section. We were always in close collaboration with one another.

3. XML Schema

3.1 Fourth iteration

Since the last iteration, we have come to better understand the CONAN, so our main focus has been to design a schema that better describes the structure of the document. Our previous iteration was able to match the text of the conan with elements in the schema, but the names incorrectly described the document. For example, we described the source text as a summary, and the elements containing the commentary about articles were different from those about amendments because of an incorrect perceived difference. Because the names were a bit arbitrary, we realized that people using the XML in the future would have trouble understanding it. We decided to research the structure of the CONAN and better figure out the structure of the document and determine more meaningful names for elements in the schema.

The following is a snippet of the CONAN for amendment 1, and it shows the basic structure of all amendments and articles.

Source Text ---------->

Congress shall make no law respecting an establishment of religion, or prohibiting the free exercise thereof; or abridging the freedom of speech, or of the press; or the right of the people peaceably to assemble, and to petition the Government for a redress of grievances.

Commentary ---------->

RELIGION

An Overview

Madison's original proposal for a bill of rights provision concerning religion read: "The civil rights of none shall be abridged on account of religious belief or worship, nor shall any national religion be established,

Scholarly Commentary.—The explication of the religion clauses by the scholars has followed a

Court Tests Applied to Legislation Affecting Religion.—Before considering the development of the

FREEDOM OF EXPRESSION—SPEECH AND PRESS

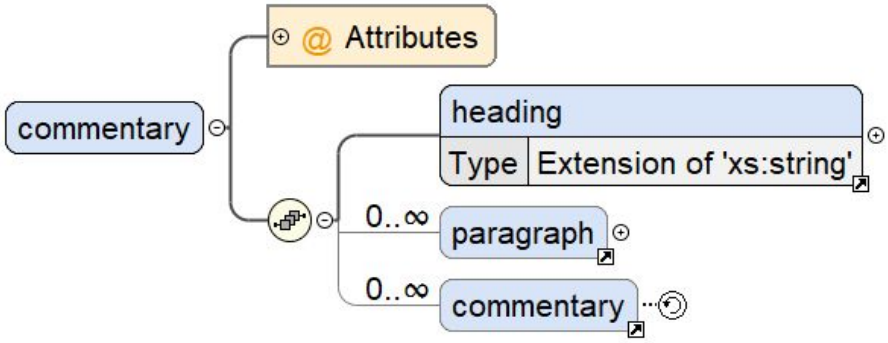
Adoption and the Common Law Background

Madison's version of the speech and press clauses, introduced in the House of Representatives on June 8, 1789, provided: "The people shall not be deprived or abridged of their right to speak, to write, or to

The structure of the CONAN is source text, the text of articles and amendments taken directly from the constitution, followed by commentary about that text.

When we compare the structure of the commentary for amendment 1 with its respective table of contents (toc), we noticed that the nested structure in the toc matches with the division of sections in the text.

As a result, we decided to model the schema based on the toc and make a recursive commentary element where each element has its own heading, an unbounded amount of paragraph elements, and nested commentary elements.



Our commentary element is structured with a heading with a level attribute that describes the location of the heading (centered, left-aligned, or inline), paragraph elements that describe the text and nested commentary elements for sub-sections of the commentary.

Amendments

The following is amendment 1, and like all amendments, it is a single blob of text :

> Congress shall make no law respecting an establishment of religion, or prohibiting the free exercise thereof; or abridging the freedom of speech, or of the press; or the right of the people peaceably to assemble, and to petition the government for a redress of grievances.

For amendments in the CONAN, the entire source-text is displayed and there is commentary about all of it.

> Congress shall make no law respecting an establishment of religion, or prohibiting the free exercise thereof; or abridging the freedom of speech, or of the press; or the right of the people peaceably to assemble, and to petition the Government for a redress of grievances.

RELIGION

An Overview

Madison's original proposal for a bill of rights provision concerning religion read: "The civil rights of none shall be abridged on account of religious belief or worship, nor shall any national religion be established,

Scholarly Commentary.—The explication of the religion clauses by the scholars has followed a

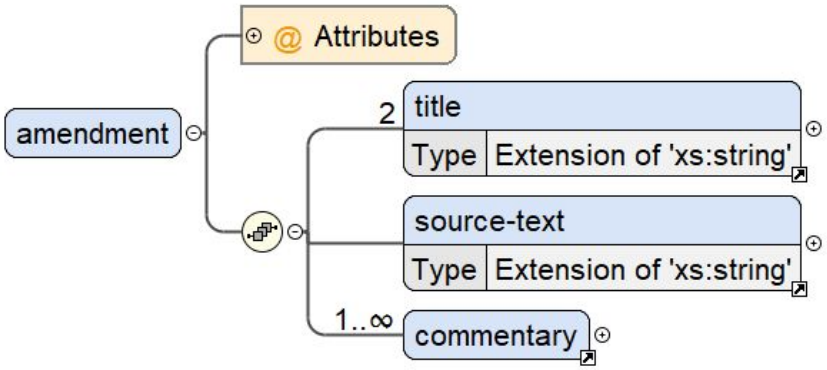
Court Tests Applied to Legislation Affecting Religion.—Before considering the development of the

FREEDOM OF EXPRESSION—SPEECH AND PRESS

Adoption and the Common Law Background

Madison's version of the speech and press clauses, introduced in the House of Representatives on June 8, 1789, provided: "The people shall not be deprived or abridged of their right to speak, to write, or to

Therefore the amendment element can be simple with just source text followed by multiple commentary elements.



Articles

The following is a snippet of the source text from article 1 :

> Section 1.
>
> All legislative powers herein granted shall be vested in a Congress of the United States, which shall consist of a Senate and House of Representatives.
>
> Section 2.
>
> The House of Representatives shall be composed of members chosen every second year by the people of the several states, and the electors in each state shall have the qualifications requisite for electors of the most numerous branch of the state legislature.
>
> No person shall be a Representative who shall not have attained to the age of twenty five years, and been seven years a citizen of the United States, and who shall not, when elected, be an inhabitant of that state in which he shall be chosen.

From a snippet of article 1, we can immediately see two differences between amendments and articles. First, there are multiple sections and second, each section can contain multiple paragraphs of text.

> Section 2. Clause 1. The House of Representatives shall be composed of Members chosen every second Year by the People of the several States, and the Electors in each State shall have the Qualifications requisite for Electors of the most numerous Branch of the State Legislature.[p.106]

CONGRESSIONAL DISTRICTING

A major innovation in constitutional law in recent years has been the development of a requirement that election districts in each State be so structured that each elected representative should represent substantially equal

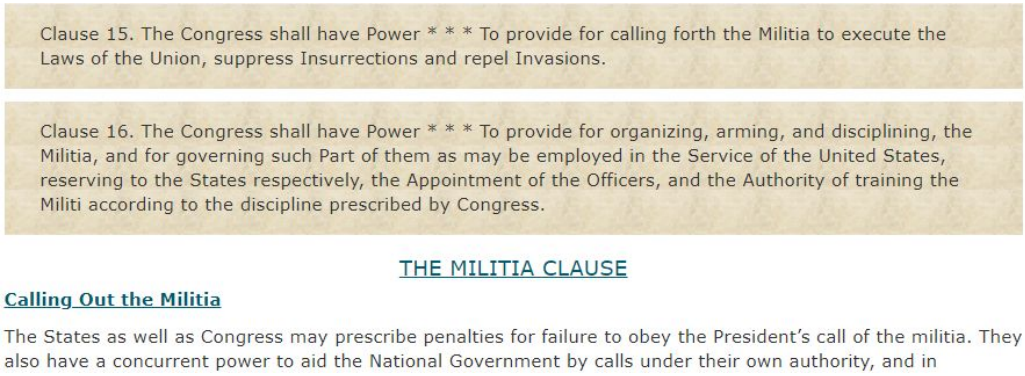
> Clause 2. No person shall be a Representative who shall not have attained to the Age of twenty-five Years, and been seven Years a Citizen of the United States, and who shall not, when elected, be an inhabitant of the State in which he shall be chosen.

QUALIFICATIONS OF MEMBERS OF CONGRESS

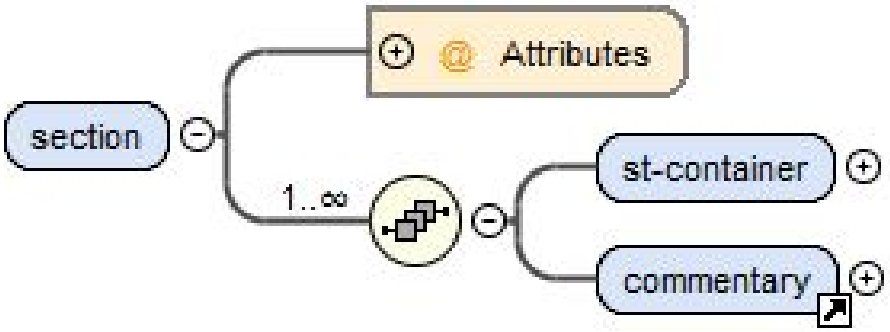
When the Qualifications Must Be Possessed

A question much disputed but now seemingly settled is whether a condition of eligibility must exist at the time of the election or whether it is sufficient that eligibility exist when the Member-elect presents himself to take the

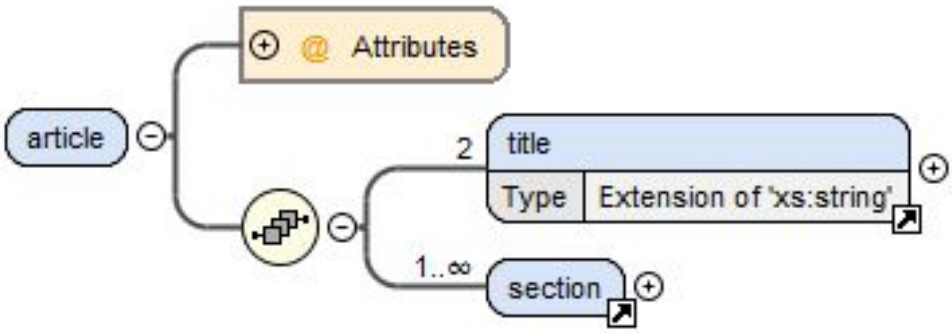
By looking at each section in the CONAN, we can see that each separate paragraph in the sections of the articles of the constitution can have its own commentary.



The last difference unique detail for the article is that multiple clauses related to a single section of commentary. Therefore, we need a was to group the related source texts together.



Because of these differences, we decided to make a section element that contains any amount of sequences of an st-container, which groups related source text, followed by commentary.

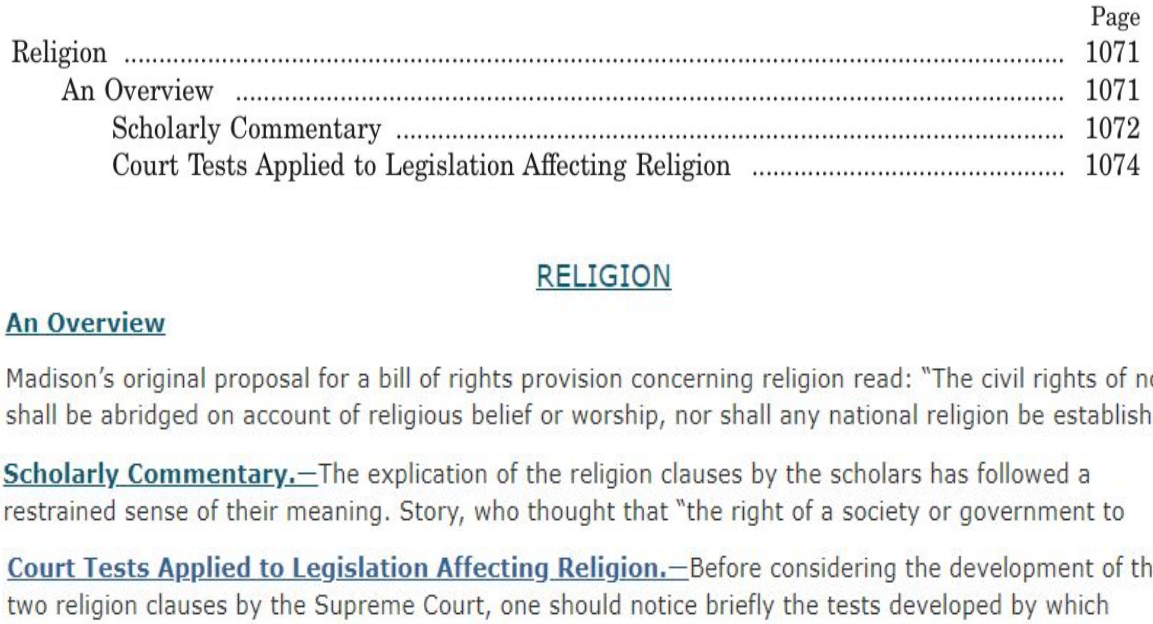


The article element itself is just a title followed by multiple section elements.

3.2 Fifth iteration

We wanted to make the document more cohesive by relating the entries in the table of contents to the relevant sections of commentary they are referencing. This would allow future users to make use to the relationship like linking to each other in html.

The following shows the relationship between the table of content entries with their corresponding headings in the document :



Page

Religion 1071

An Overview 1071

Scholarly Commentary 1072

Court Tests Applied to Legislation Affecting Religion 1074

RELIGION

An Overview

Madison's original proposal for a bill of rights provision concerning religion read: "The civil rights of none shall be abridged on account of religious belief or worship, nor shall any national religion be established,

Scholarly Commentary.—The explication of the religion clauses by the scholars has followed a restrained sense of their meaning. Story, who thought that "the right of a society or government to

Court Tests Applied to Legislation Affecting Religion.—Before considering the development of the two religion clauses by the Supreme Court, one should notice briefly the tests developed by which

We decided to do this by giving table of contents entries an id attribute and the relevant sections an id-ref attribute. For example, we would link the 'religion' toc-entry with the commentary element containing the 'religion' section, link the 'An Overview' toc-entry with the commentary element containing the 'An Overview' section, and so forth.

3.3 Next steps

We will continue to refine the elements for the minor details of the CONAN like the footnotes by researching the CONAN at the same level as we did for the amendments and articles. We will also continue thinking about different structures based on difficulties using that schema elements for XML generation. If difficulties arise, we will try to reduce some of the XML features while trying to maintain the structure of the document.

4. XML File

4.1 Change of Methodology

In this phase of the project we made a significant change to our approach for generating XML that matches our schema. Initially our process was to go through the text of the raw XML file and try to use font ID patterns to match text to the tags in our schema. We found that this process was very difficult and we were spending a lot of time to 'shoehorn' the text into our schema. The method we have adapted to which has been extremely beneficial, is to instead first match the raw XML to the structure give in the table of contents. This has been a much easier process as the way the text is displayed, and thus the resulting format in the raw XML, is exactly indicative of the structure given in the table of contents. This means that it is extremely easy and intuitive to first markup the text in the format given by the table of contents.

Besides being an easier, this approach has several benefits. First of all the result is something that is already useable. What is meant by this is that given we are matching text to table of contents entries we have an easy way to provide links from the table of contents to the text. So we can already have the data in a format that would enable the proper links when displayed in a web format. Secondly, we have

found that once we have the structure given in the table of contents it is very intuitive to further refine the text to separate what is source text of the constitution and what is commentary.

Lastly, and perhaps most importantly, it provides a way for us to guarantee a deliverable in the end, while iteratively working to make it better. The previous method, to the group, felt like an 'all or nothing' approach, in that if and only if all of the corner cases and bugs were worked out would we have an XML that matched our schema. This, while possibly successful, produces a lot of uncertainty in the final project being completed. With the new approach we essentially relax the constraints of the schema and try to work to iteratively improve our XML file to match our idealized schema. In this way we guarantee at each step that we have a deliverable XML, and further, that at each step we are improving this XML. This is a critical point, to ensure our project achieves some degree of success.

4.2 Improvements

Since the last script besides changing methodologies we have also made several improvements to our XML file. The first is that we are now able to process Articles whereas before we only processed Amendments. Secondly we now have a clear distinction between what is source text of the constitution and what is commentary from CONAN. We have also continued to work on fixing corner cases, such as dealing with hyphenation and whitespace issues. We acknowledge, however, that by nature it is hard to write general rules to fix these corner cases, and this is a continual process.

4.3 Updated Requirements

Given the change in our methodology we also have some update XML requirements. The first requirement is that the XML be marked up in a way that reflects the structure of the table of contents. The second is that we are able to mark up what is source text in the document and what is commentary. The third is that we iteratively work to make our XML match our idealized schema. The fourth requirement is that we properly mark up citations, and court cases. Lastly, we want to ensure that the final document is returned in a format the client is satisfied with, that is whether it should be a single document or multiple documents.

4.4 Raw XML Generation

Similarly to last report, the first step in the process is to generate XML from the PDF document. We need to get the PDF into a format that we can massage and manipulate into the format we desire. In order to do this we use a UNIX script called *pdfToHtml* which takes as input a PDF and outputs what we refer to as raw XML. The raw XML has two main sections, the first is a font identification section, which lists all the fonts appearing in the document, classified based off of type, size, color, font family ect., and it assigns a font id to each font. A sample section can be seen below :

```
<page number="1" position="absolute" top="0" left="0" height="12
  <fontspec id="0" size="17" family="Times" color="#231f20"/>
  <fontspec id="1" size="14" family="Times" color="#231f20"/>
```

The second section is the actual content of the PDF, marked up per line by font id as well as position on the page. A sample section can be seen below :

```
<text top="359" left="184" width="11" height="18" font="5">S</text>
<text top="363" left="195" width="47" height="12" font="3">ECTION</text>
<text top="359" left="247" width="10" height="18" font="5">1</text>
<text top="361" left="257" width="4" height="15" font="4">.</text>
-<text top="359" left="266" width="421" height="18" font="5">
  All legislative Powers herein granted shall be vested
</text>
-<text top="385" left="148" width="539" height="18" font="5">
  in a Congress of the United States, which shall consist of a Sen-
</text>
<text top="412" left="148" width="289" height="18" font="5">ate and House of
```

4.5 Raw XML to Meaningful XML

Once we have the raw XML we then want to use the font identification tags and indentation of the lines to infer how the structure fits into the hierarchy given in the table of contents. This is a relatively intuitive task because the labels in the table of contents are almost always set in the text by indentation and/or by a bold or bigger typeface. Using those features we can then extract the structure of the document in terms of the table of contents. Once we have that done we have a semantic understanding of what all of the text means, and it becomes easy to then separate the text into actual source text and commentary, and then we can iteratively work to make that fit the tags of our schema. We additionally at this step aim to mark up all of the citations in the text. For example we can change the raw XML picture above into the following :

```
-<Section1>
 -<SourceTextContainer1>
  -<ClauseContainer1>
     All legislative Powers herein granted shall be vestedin a Congress of
   </ClauseContainer1>
  </SourceTextContainer1>
```

As seen we successfully markup the text into its correct tag. Additionally here is one more example :

```
-<text top="697" left="214" width="326" height="14" font="6">
   <b>The Theory Elaborated and Implemented</b>
 </text>
-<text top="718" left="241" width="446" height="15" font="4">
   When the colonies separated from Great Britain following the
 </text>
-<text top="736" left="214" width="473" height="15" font="4">
   Revolution, the framers of their constitutions were imbued with the
 </text>
-<text top="754" left="214" width="473" height="15" font="4">
   profound tradition of separation of powers, and they freely and ex-
 </text>
 <text top="772" left="214" width="324" height="15" font="4">pressly embodied the
 <text top="773" left="538" width="6" height="9" font="7">2</text>
```

After converting to a more meaningful XML, this becomes :

```
−<Comments3 title="The Theory Elaborated and Implemented">
  −<text>
      When the colonies separated from Great Britain following theRevolution, the framers of
      powers, and they freely and expressly embodied the principle in their charters.
      <citation citationNum="2"/>
    </text>
```

We can now successfully mark up the commentary of the document as well as mark in-line citations. For converting amendments and articles, we currently have an error rate of about 2 errors for every 1-2 pages. These errors involve inaccurate formatting/whitespace.

4.6 Post Processing

The final step, similar to last report, is the post-processing of the document. In this step we integrate the client's script that correctly marks up the citations of the text. We additionally do some processing to try and capture the corner cases of the document. As stated previously, however, this is an ongoing problem and is presenting difficulties for the team.

4.7 Completed Work

At this step we have a script that is successfully and fully automatically going from the PDF versions of articles to marked up text that preserves the structure of the table of contents, separates source text and commentary, is matching the majority of our schema and handles in-line citations. We are additionally in the process and nearing completion of a script that will split CONAN into its constituent parts and feed it into the scripts we already have completed.

4.8 Next Steps

We have several issues and features we still need to address. The first issue to deal with is that sometimes there will be an element indicated in the table of contents that is actually not present in the text. In this case we need to add an empty element in our schema to represent this error. Additionally we need to actually integrate the client's citation script into our own, and not run it as a post process. And lastly we need to continue to work to iteratively improve our XML to match our schema.

5. RDF Models

RDF (Resource Description Framework) is a standard model for data interchange on the Web. RDF has features that facilitate data merging even if the underlying schemas differ.

5.1 Learning Curve

We spent the first week (till middle of spring break) learning about basics of RDF and RDF models. We read the book titled 'Semantic Web For Dummies' which was given to us by our client. We also read the existing LII documents on RDF models, so that our designs are compatible with existing products. We also spent a lot of time learning how to use Protege (which is a tool from Stanford and was suggested by our client for working with RDF). We watched videos about how to use Ontograf, Graphviz, compatibility of versions, etc. Finally we explored the Python package rdflib, which we would be using extensively while implementing our rdf models. We also referred to the Wikipedia article on RDF.

5.2 Purpose

RDF repositories will model the important legal information contained in the CONAN. The problem that we are trying to solve by building RDF models is as follows. The RDF will help the client to query important information contained in the CONAN such as :

- **Co-occurrence :** RDF will allow to query with which other citations does a particular citation occur in a single footnote. This can also be used to find out which legal cases are related, since they have been referred to within the same footnote.
- **Association** : which citations and what commentary are associated with a given clause
- **Relation** : what commentary is associated with a given case, and which parts of the Constitution are related with this case

Our goal is to be able to answer the above and at the same time keep the models simple.

5.3 RDF Model Requirements

The model requirements became clear through client meetings, where we discussed in detail the expectations of the client from these models and the necessities.

The RDF model should demonstrate the following :

- Hierarchical structure of the commentary
- Relationship of the headings to the text of the CONAN
- Location of each footnote within the hierarchical structure of the commentary
- Relationship between each citation and the footnote in which it is contained

5.4 Progress in RDF

We decided to change to Iterative Refinement from Agile Development because the design for the RDF models do not require any testing. The goal is to get closer to satisfying client requirements in each iteration. We are currently on the third iteration for designing RDF models. The versions we made and the feedback from the client for each of them are given in figure below. The details of each of the versions and how we incorporated changes are outlined in the sections below.

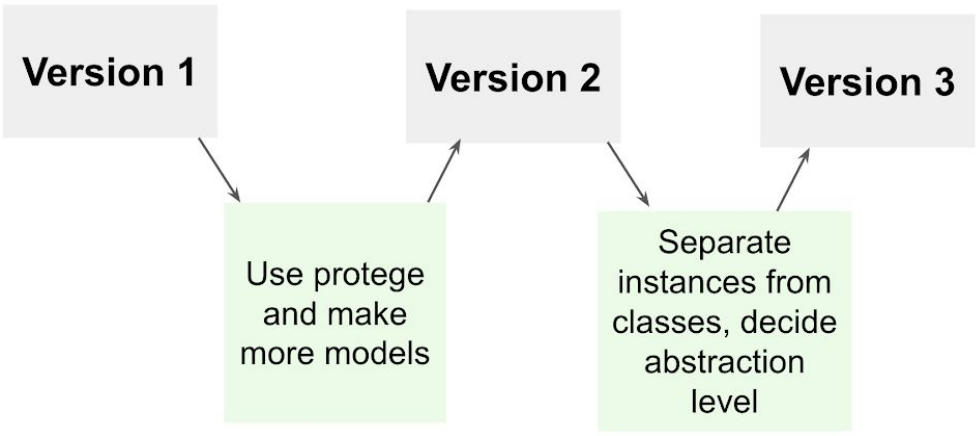


Fig : Versions of RDF model and Client feedback

5.5 First Iteration

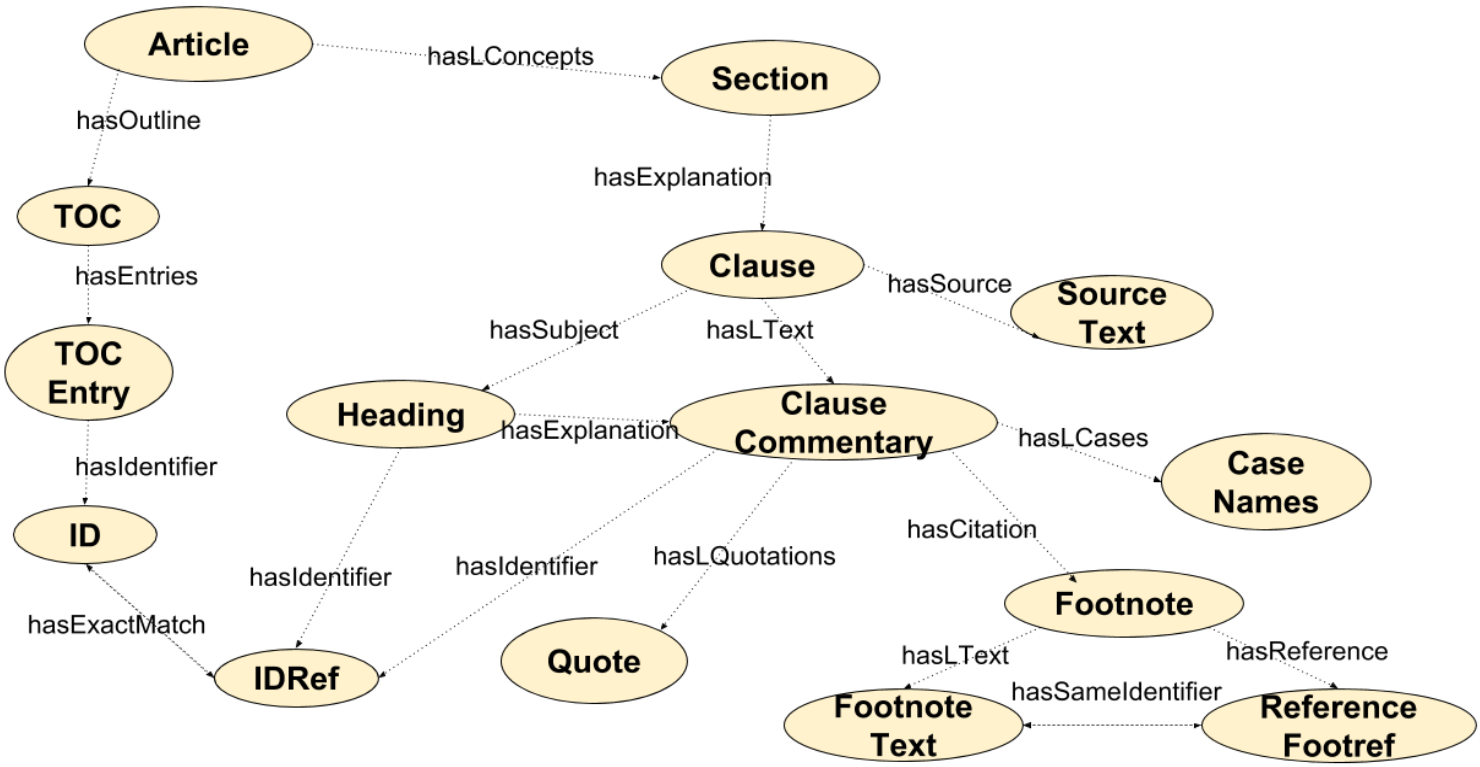


Fig : RDF Model (first iteration)

Client Feedback for First Version

The first version was an initial prototype we had made to know whether we were on the right track or not. The client suggested the following changes :

- Make more than one model
  - Models that describe the structure of the commentary in the CONAN, including footnotes
  - Models that describe all the interesting metadata found in the commentary
- These models should be related by properties which describe how classes and individuals in one model are related to another
- A set of metadata objects which take properties that can be used to find where the given thing is found in the structures described by the models
- Use Protege tool to make RDF models
  - Protege is a free and open source ontology editor developed by Stanford Center for Biomedical Informatics Research
  - We used version 4.1 of Protege, which is a version that is compatible with graphics. We used Turtle as the ontology format and Ontograf for viewing classes and object properties. The other option was Graphviz but we stuck with Ontograf.

5.6 Second Iteration

We made three models : one for commentary, one for footnote and one for first amendment.

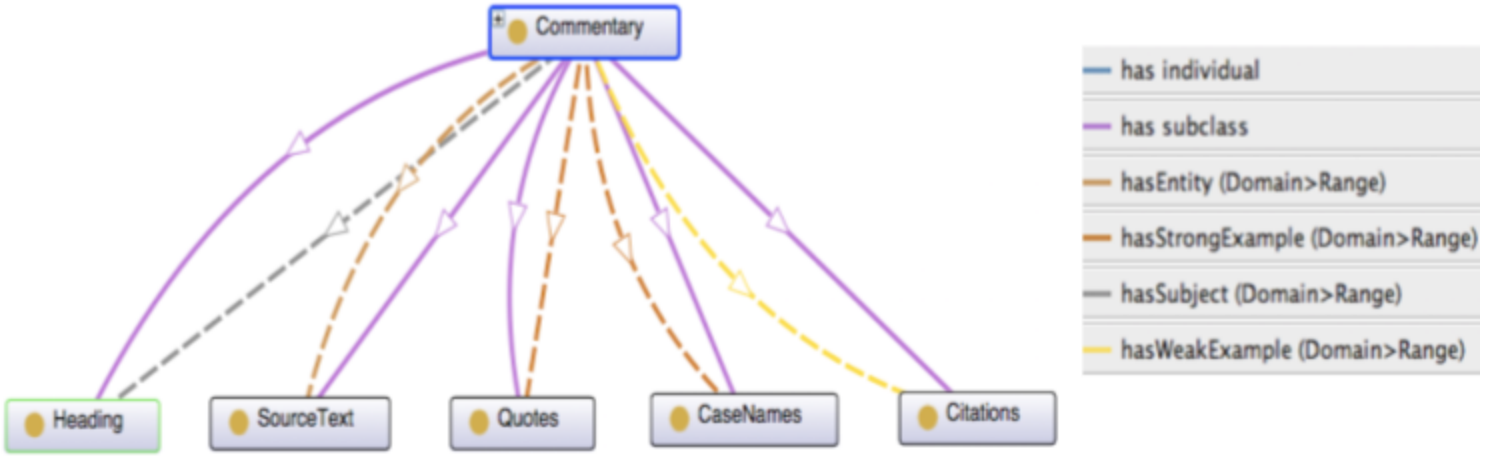


Fig : RDF Model for Commentary

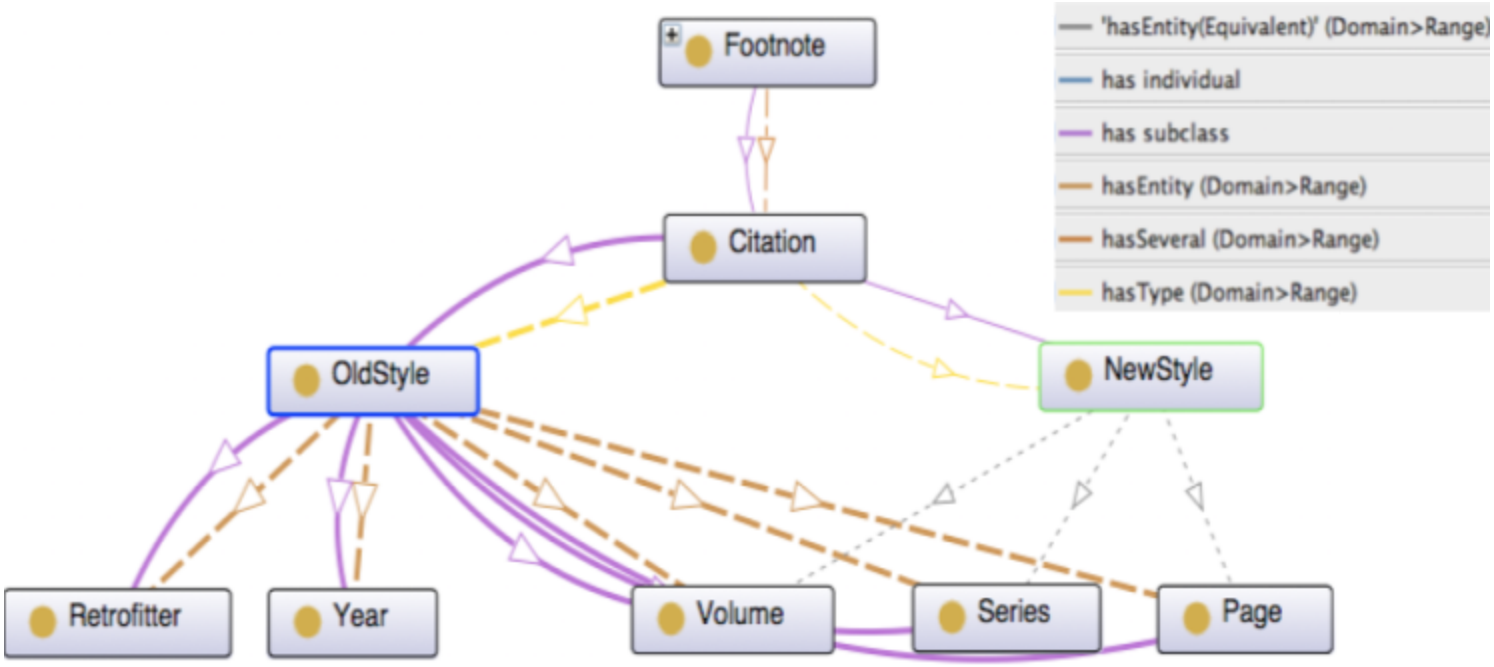


Fig : RDF Model for Footnote

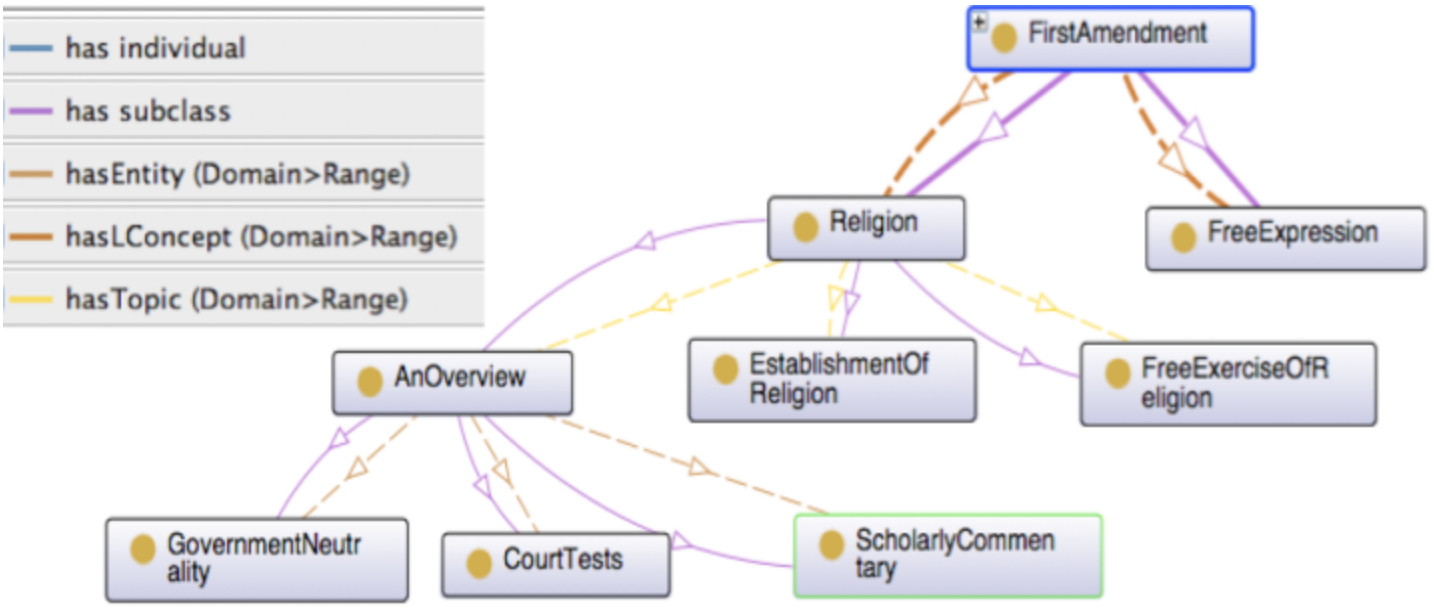
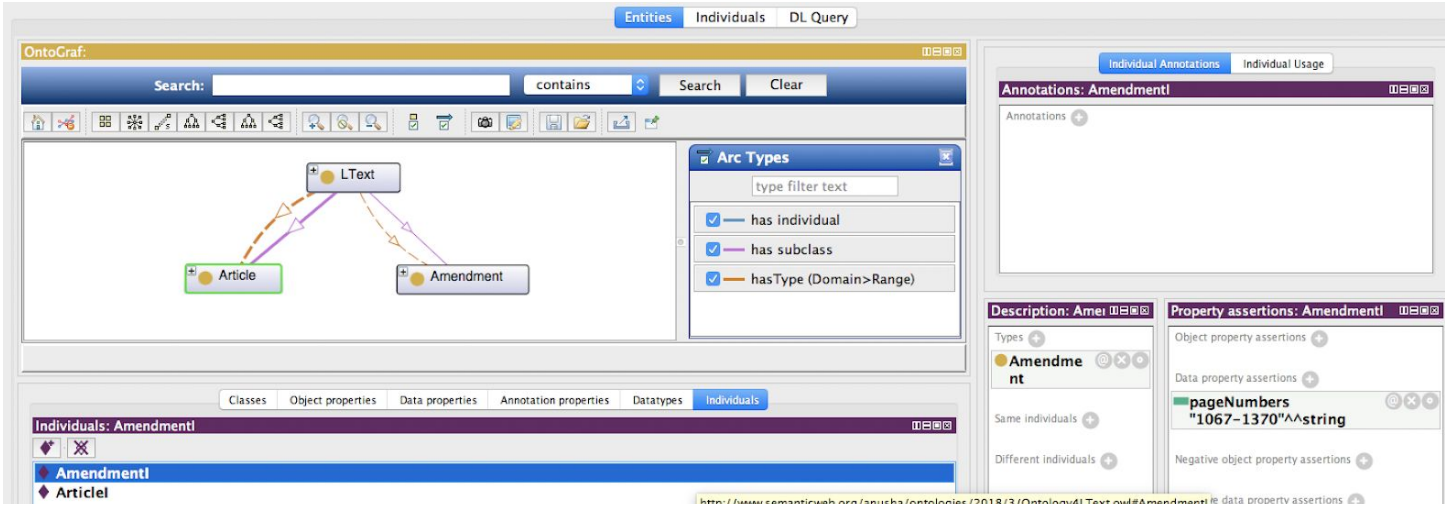


Fig : RDF Model for First Amendment

Client Feedback for Second Version

- Separate classes from instances, ex. LText can be a class while first amendment is an instance
- Keep it simple
- Show metadata as data properties of these instances to convey the structure of the document
- Relate subclasses of different models through properties

5.7 Third Iteration



We have LText as a class since articles and amendments are all legal texts. Article I is an instance, and the page number range for it is one of the metadata that has been shown as data property of individual. We can have more properties of the article like heading, number of sections and clauses as metadata.

5.8 Next Steps

The next steps are to finalize the design for RDF model and then implement it.

RDF Model Design

Iteration 4 : This requires the team to refine the design for the data model and finalize it.

Timeline :

1. April 14 - Incorporate changes suggested by the client in iteration 3 in existing models

2. April 16 - Make new models in Protege as suggested by the client
3. April 17 - Present the model to client

We have kept a slack of 2 days to avoid too ambitious planning. Our deadline for designing the RDF model is April 19. We know it might be difficult to achieve the final model in just one more iteration but we have set this target because this is the only way we can finish both design and implementation for RDF on time, and we will try our best to complete both.

RDF Implementation

Please note that this phase will be done only after the design for RDF model has been finalized. We currently have the following plan for the implementation phase but depending on the timeline for the design phase, this may need manipulation.

Iteration 1 : We will begin this phase on April 19 and show the first version to the client on April 24. We know this is a hard deadline, but since we started exploring rdflib, it should not be too difficult to experiment with existing packages and implement parts of the model.

Iteration 2 : Depending on the changes suggested by the client, we plan to show the second version on April 27, with a possible slack of 1 day. Here we plan to have a complete implementation for the RDF model.

Iteration 3 : The deadline we target for this iteration is April 30. Again there can be a slack of 2 days. This plan leaves around a week for the acceptance testing.

6. Development Timeline

6.1 Development Process

We are following the iterative software development process for both steps of this project. Previously, we planned to follow the agile development process for implementing RDF repositories. However, we realized that similar to the XML conversion step, there is nothing to test until our RDF implementation is complete. Hence, the RDF implementation step is following the iterative software development process as well.

6.2 Current Progress

The following tables show our updated schedule. Since the previous report, we have completed 2 iterations for developing the XML schema, XML file, and RDF models. The table cell colors mean :

- green - iterations completed before the first progress report
- yellow - iterations completed between the first and second progress reports
- blank - iterations we expect to complete before the final progress report

Iteration #	XML Schema	XML File	Extract Footnotes
1 - 4	March 14	March 16	March 16
5	March 27	March 29	
Slack	Spring Break	Break	
6	April 11	April 11	
7	April 17	April 20	

8	---	April 27	

Iteration #	RDF Models	RDF Repositories
1	April 5	April 24
2	April 10	April 27
3	April 12	April 30
4	April 17	---
Slack	2 days	2 days

6.3 Milestone Deadlines for Remaining Iterations

XML Schema Development

Iteration 7

- Milestone 1 (April 12) : Incorporate changes suggested by the client on version 6
- Milestone 2 (April 14) : Perform XSD validation
- Milestone 3 (April 15) : Polish schema and fix any errors found during testing
- Milestone 4 (April 16) : Finish writing documentation for schema generation
- Milestone 5 (April 17) : Submit to the client for acceptance testing along with documentation

This completes seventh iteration. We have kept a slack of 2 days for this iteration.

XML File Development

Iteration 7

- Milestone 1 (April 13) : Incorporate elements from newly designed schema
- Milestone 2 (April 14) : Test and analyze error rates for separate amendments and articles
- Milestone 3 (April 16) : Incorporate changes to reduce error rate
- Milestone 4 (April 17) : Show to the client for feedback
- Milestone 5 (April 19) : Adjust file to changes suggested by the client on version 7
- Milestone 6 (April 20) : Test file and fix remaining errors

This completes seventh iteration.

Iteration 8

- Milestone 1 (April 23) : Show to the client in case any further changes are needed
- Milestone 2 (April 24) : Finish testing, validate and polish the XML file
- Milestone 3 (April 25) : Finish documentation for XML file generation
- Milestone 4 (April 27) : Submit to the client for acceptance testing along with documentation

This completes eighth iteration.

RDF Models Development

Iteration 4

- Milestone 1 (April 14, 2018) : Incorporate changes and complete prototype of data models
- Milestone 2 (April 15, 2018) : Test and polish the models

- Milestone 3 (April 16, 2018) : Write documentation for generating the models
- Milestone 4 (April 17, 2018) : Submit models to client along with documentation

This completes fourth iteration. We have kept a slack of 2 days for this iteration.

RDF Repositories Implementation

Iteration 1

- Milestone 1 (April 19, 2018) : Research RDF repositories and read articles suggested by client
- Milestone 2 (April 20, 2018) : Learn about the existing repository implementation systems in LII
- Milestone 3 (April 21, 2018) : Change existing systems to fit our RDF models
- Milestone 4 (April 23, 2018) : Store data for one article and one amendment in a repository
- Milestone 5 (April 24, 2018) : Show to the client for feedback

This completes first iteration.

Iteration 2

- Milestone 1 (April 24, 2018) : Incorporate changes suggested by the client on version 1
- Milestone 2 (April 26, 2018) : Test repositories and determine error rate, incorporate changes
- Milestone 3 (April 27, 2018) : Show to the client for another round of feedback

This completes second iteration.

Iteration 3

- Milestone 1 (April 28) : Incorporate changes suggested by the client for version 2
- Milestone 2 (April 29) : Finish testing, polish repositories, and write documentation
- Milestone 3 (April 30) : Submit to the client for acceptance testing along with documentation

This completes third iteration.

We plan to complete our PDF to XML conversion step by April 27th and our RDF implementation step by April 30th while giving ourselves 2 slack days. We expect to have approximately 2 weeks for acceptance testing before the project deadline on May 17th.

6.4 Test Plan

We plan to test our system as follows.

XML Schema

- Validate XSD format
- Check if all tags in the schema are used in the XML

XML File

- Validation of XML format : test against schema
- Preservation of formatting : create a stylesheet to use visual indicators
- Preservation of content in PDF : check the presence of headings in proper levels, check for proper concatenation of hyphenated words against English dictionary

RDF Models

- Ensure all classes, properties and instances required by the client are present
- Ability to represent structure of commentary of the CONAN

RDF Repositories

- Validation : ensure that elements mentioned in the models are present
- Preservation of content in PDF : consistency of information stored in repositories with the PDF

7. Risks and Challenges

7.1 Technical Challenges We Handled

Since its last report, the team has tackled several challenges. First, work has begun on handling corner cases with improperly parsed text from the PDF (eg. white space preservation, spelling, words missing spaces in between). Next, it changed its approach to determining the structure of the CONAN by using tables of contents and linking headings within the document's body to link to elements in the table's tree structure using ID/IDRef relationships. Finally, this change in approach saw a refactor of the schema and a shift in development approach in which schema design would work more closely with development and the schema would be based off of both requirements and what is the most robust/simple to implement for developers.

7.2 Remaining Challenges

First and foremost, important issues introduced in the last report still remain as core challenges of the project - PDF data is dirty and hard to parse, it is hard to build structured data out of PDF that is robust enough to meet requirements, and many special cases must be handled. Additionally, the change of approach to more iterative schema/XML development led to a rework of the schema which consumed valuable time, and required more dev energy. Schema development took a sharp turn and still needs to work with development to decide on a final structure that is both sufficiently robust, but able to handle the unavoidable inconsistencies in the XML and the CONAN - the CONAN is human-generated, and it has become clear that it is impossible to handle all special cases; the question becomes one of how we can accept these issues and oddities gracefully without the software failing or the schema failing to validate. Though there are many development benefits to the new approach, work needs to be done to consider all of its pitfalls : because names are inconsistent, a consistent method of generating ID/IDrefs needs to be decided on, more code is required for table of contents parsing, and, if the error rate of the current strategy of linking sections by name to the table of contents proves too high, a new method of linking will be required.

7.3 Error Rate

The error rate for converting the entire document is of great importance and needs to be better quantified. In general, with the XML software moving towards completion as the project nears its end, a new challenge has emerged : testing. It is important for us to not forget to rigorously test our code, and to deal with the bugs and issues that will inevitably come up - what makes this such a challenge and a risk is that we do not yet know the results. Moreover, it is important to determine error rate, and make a difficult decision about accepting the approach we have currently, or to modify it yet again. With a 2000+ page document, proper testing will be a challenge : discovering and fixing corner cases will be difficult, writing automated testing to cover all cases will not be easy, and the team and client need to work together to discuss what defines success for the software and the project, and what needs to be improved on before the final delivery.

7.4 Risk Analysis

Naturally, these challenges lend themselves to risks. First and foremost are considerations about the new XML development approach. With the change in strategy to using the table of contents, large amounts of work was done to remake the schema, and more work remains for both developers and schema developers to learn what will and won't work with this approach to parsing the CONAN's structure. Though this will result in a more robust final product, it without a doubt leaves the team with extra work (be it code or design), and research that needs to be done. This is fine, but it is indeed a risk

when the project is roughly a month away from its deadline. Time constraint as a whole is a concern, but it affects XML/XML Schema development the most.

The RDF portion of the project also comes with a lot of risk - RDF has a steep learning curve, and even though the team is spending time to go through iterations of an RDF model design, it is still learning the basics RDF. With little time left and, more importantly, incompleteness of the XML portion of the project, it is difficult for those working on RDF to move forward easily with a design that it can deliver, nevermind implement. RDF is indeed a source of risk, though not as large as XML as the XML is the most important component of the project.

Finally, the status of team members has become a risk. Sadly, multiple members have had a series of health issues as of late which delayed work and may continue to delay work. With a 5-member team, having two members being away for significant time could immensely slow down development and make communication among various sections of the team more difficult.

7.5 Next Steps

In reviewing these risks the team has decided to take a pragmatic approach and to prioritize its tasks for the remaining time that it has. First, lots of work and focus is being spent on completing the XML as it is a core requirement. For example, the schema has been simplified (though work has been done to ensure that it is still robust), work is being done to determine error rates for various PDF parsing strategies (relying on table of contents, or typographic information), and efforts are being made to determine which corner cases are most important to handle. Second, the team has and advises that the client temper their expectations for a completed RDF implementation. In discussing with the client it was determined that the XML was more important, and that the project would still be of much value if it delivered a working XML markup and perhaps an unimplemented but robust RDF model. Ultimately, the team will likely need to cut back on requirements due to time and resource constraints, but is still willing and able to deliver a working and useful final product, even if it may not be absolutely perfect. It does believe that the product will meet requirements, pass tests to an acceptable degree, and provide a useful base for future work to use, expand, complete, and perfect.