XPath-Based Template Language for Describing the Placement of Metadata within a Document

Vijay Kumar Musham
Old Dominion University

Follow this and additional works at: https://digitalcommons.odu.edu/computerscience_etds

Part of the Cataloging and Metadata Commons, and the Programming Languages and Compilers Commons

Recommended Citation
Musham, Vijay K.. "XPath-Based Template Language for Describing the Placement of Metadata within a Document" (2010). Master of Science (MS), Thesis, Computer Science, Old Dominion University, DOI: 10.25777/vekv-w277
https://digitalcommons.odu.edu/computerscience_etds/143

This Thesis is brought to you for free and open access by the Computer Science at ODU Digital Commons. It has been accepted for inclusion in Computer Science Theses & Dissertations by an authorized administrator of ODU Digital Commons. For more information, please contact digitalcommons@odu.edu.
XPATH-BASED TEMPLATE LANGUAGE FOR DESCRIBING THE
PLACEMENT OF METADATA WITHIN A DOCUMENT

by

Vijay Kumar Musham
B.Tech. May 2006, Andhra University, India

A Thesis Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirement for the Degree of

MASTER OF SCIENCE

COMPUTER SCIENCE

OLD DOMINION UNIVERSITY
December 2010

Approved by:

Steven J. Zeil (Director)

Kurt Maly (Member)

Mohammad Zubair (Member)
In the recent years, there has been a tremendous growth in Internet and online resources that had previously been restricted to paper archives. OCR (Optical Character Recognition) tools can be used for digitalizing an existing corpus and making it available online. A number of federal agencies, universities, laboratories, and companies are placing their collections online and making them searchable via metadata fields such as author, title, and publishing organization. Manually creating metadata for a large collection is an extremely time-consuming task, and is difficult to automate, particularly for collections consisting of documents with diverse layout and structure. The Extract project at ODU has developed an automated metadata extraction system to support document collections with diverse structure and layout. A template language was developed for this purpose to describe the layout of metadata within diverse document layouts and has been used for several years. An alternative template language based on XPath was later proposed. This thesis involves the implementation of a Java based interpreter which executes the new template language to extract metadata from the documents. This thesis also involves the evaluation of the relative power and understandability of the two template languages.
©Copyright, 2010, by Vijay Kumar Musham, All Rights Reserved.
This thesis is dedicated to my mother and brother,
who have always inspired me to soar to greater heights.
ACKNOWLEDGMENTS

This thesis arose in part out of years of research that has been done since I came to Extract research group. By that time, I have worked with a great number of people whose contribution in assorted ways to the research and the making of the thesis deserved special mention. It is a pleasure to convey my gratitude to them all in my humble acknowledgment.

In the first place I would like to record my gratitude to Dr. Steven Zeil for his supervision, advice, and guidance from the very early stage of this research as well as giving me extraordinary experiences throughout the work. Above all and the most needed, he provided me unflinching encouragement and support in various ways. His truly scientist intuition has made him as a constant oasis of ideas and passions in science, which exceptionally inspire and enrich my growth as a student, a researcher and a scientist want to be. I am indebted to him more than he knows.

I gratefully acknowledge Dr. Kurt Maly and Dr. Mohammad Zubair for their advice, supervision, and crucial contribution, which made them a backbone of this research and so to this thesis. Their involvement has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come. I am grateful to them in every possible way for the financial support which made my stay happy at Norfolk.

Many thanks go in particular to, the members of my committee, for their thorough review of this dissertation and for their valuable feedback. I am thankful that in the midst of all their activity, they accepted to be members of the reading committee.

I also benefited by outstanding works from Divya Josyala. A special thanks for her helpful discussions and contributions to this work. I have also benefited by advice and guidance from Paul Flynn who also always kindly grants me his time even for answering some of my unintelligent questions. I am very grateful for their helpful discussions and contributions to this research project.

To the role models for hard workers in the lab, Pranay Pramod and Justin John, I would like to thank them for being the first persons who taught me about the research work at the lab. It is a pleasure to pay a special tribute to Pranay, for introducing me to the Extract group and helped me in getting the research assistant position.

Collective and individual acknowledgments are also owed to my colleagues at Digital Library Research Group whose present somehow perpetually refreshed, helpful, and memorable. Many thanks go in particular to Sandeep Gadi, Sateesh Kadiyal, Abhishek Biswas, Sandeep Gampa, and Vinay Paul Kanumuri, for giving me such a pleasant time.
when working. I convey special acknowledgement to my roommates and friends who made my life happy. Thanks for the parties, cricket, lunches and lots of fun.

Where would I be without my family? My parents deserve special mention for their inseparable support and prayers. My Father, Anantha Ramulu, in the first place is the person who put the fundament my learning character, showing me the joy of intellectual pursuit ever since I was a child. My Mother, Arundhati, is the one who sincerely raised me with her caring and gently love. My Brother Ajay, whose dedication, love and persistent confidence in me, has taken the load off my shoulder. I owe him for being unselfishly let his intelligence, passions, and ambitions collide with mine.

Finally, I would like to thank everybody who was important to the successful realization of thesis, as well as expressing my apology that I could not mention personally one by one.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF TABLES</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES</td>
<td>xi</td>
</tr>
</tbody>
</table>

## Chapter

### 1. INTRODUCTION

1.1 Overview ................................................................. 1
1.2 Problem statements .................................................... 3
1.3 Approach ........................................................................ 4
1.4 Objectives ...................................................................... 4
1.5 Organization of the thesis ............................................ 5

### 2. PREVIOUS WORK

2.1 Input processing .......................................................... 6

#### 2.1.1 Independent document model (IDM) ......................... 6

2.2 Form processing ........................................................... 7

2.3 Non-form processing ..................................................... 8

#### 2.3.1 Template construction ............................................ 8

2.4 Non-form classification .................................................. 10

2.5 Output processing ........................................................ 11

#### 2.5.1 Post-processing ................................................... 11

#### 2.5.2 Validation ............................................................ 12

2.6 Experimental results .................................................... 13

#### 2.6.1 Non-form extraction experiments ............................ 13

2.7 Limitations of present non-form engine ............................ 13

### 3. XPATH TEMPLATE LANGUAGE

3.1 XPATH ........................................................................... 14

#### 3.1.1 Introduction .......................................................... 14

3.2 Template language ......................................................... 17

#### 3.2.1 <TEMPLATE> ....................................................... 17

#### 3.2.2 <BIND> ............................................................... 18

#### 3.2.3 <LET> ................................................................. 19

3.3 Probable xpath extensions .............................................. 19

#### 3.3.1 Third-party xtensions ............................................. 20

3.4 Example ......................................................................... 21
4. SYSTEM IMPLEMENTATION ............................................................ 25
4.1 INTERPRETER FRAMEWORK .................................................. 25
4.2 EVALUATING THE XML TAGS ............................................. 25
4.2.1 TEMPLATE IMPLEMENTATION ................................ 26
4.2.2 LET TAG IMPLEMENTATION ................................ 26
4.2.3 BIND TAG IMPLEMENTATION ................................ 28
4.3 XPATH FUNCTIONS ......................................................... 30
4.4 EXTENSION FUNCTIONS .................................................... 33
4.5 THIRD PARTY EXTENSIONS ............................................... 36

5. COMPARING THE TWO LANGUAGES ........................................... 38
5.1 POWER OF THE LANGUAGE .................................................. 39
5.1.1 DOES THE NEW LANGUAGE COVER ALL THE
      FUNCTIONALITY OF THE OLD LANGUAGE? ............... 39
5.1.2 CAN WE WRITE EFFECTIVE TEMPLATES USING
      THE NEW LANGUAGE? .......................................... 42
5.2 EFFECTIVENESS OF THE LANGUAGE ..................................... 52
5.2.1 UNDERSTANDING THE NEW LANGUAGE ...................... 52

6. CONCLUSION AND FUTURE WORK ........................................... 54
6.1 CONCLUSIONS .................................................................. 54
6.2 FUTURE WORK .................................................................. 54

BIBLIOGRAPHY ......................................................................... 55

APPENDICES
A.1 AFIT ............................................................................. 57
A.1.1 AFIT DOCUMENT ...................................................... 57
A.1.2 AFIT TEMPLATE IN OLD LANGUAGE ......................... 57
A.1.3 AFIT TEMPLATE IN NEW LANGUAGE ......................... 57
A.2 ACGSC ................................................................. 57
A.2.1 ACGSC DOCUMENT ................................................. 57
A.2.2 ACGSC TEMPLATE IN OLD LANGUAGE ..................... 57
A.2.3 ACGSC TEMPLATE IN NEW LANGUAGE ..................... 57
A.3 AFRL4 ......................................................................... 57
A.3.1 AFRL4 DOCUMENT ................................................. 57
A.3.2 AFRL4 TEMPLATE IN OLD LANGUAGE ...................... 62
A.3.3 AFRL4 TEMPLATE IN NEW LANGUAGE ...................... 62
A.4 USACERL ...................................................................... 62
A.4.1 USACERL DOCUMENT ............................................. 62
A.4.2 USACERL TEMPLATE IN OLD LANGUAGE ................. 62
A.4.3 USACERL TEMPLATE IN NEW LANGUAGE ................. 62
A.5 HEAD-ABSTRACT ......................................................... 67
A.5.1 HEAD-ABSTRACT DOCUMENT .................................. 67
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A.5.2</td>
<td>HEAD-ABSTRACT TEMPLATE IN OLD LANGUAGE</td>
<td>67</td>
</tr>
<tr>
<td>A.5.3</td>
<td>HEAD-ABSTRACT TEMPLATE IN NEW LANGUAGE</td>
<td>67</td>
</tr>
<tr>
<td>A.6</td>
<td>HEAD-ABSTRACT-TOPMARGIN</td>
<td>67</td>
</tr>
<tr>
<td>A.6.1</td>
<td>HEAD-ABSTRACT-TOPMARGIN DOCUMENT</td>
<td>67</td>
</tr>
<tr>
<td>A.6.2</td>
<td>HEAD-ABSTRACT-TOPMARGIN TEMAPLATE IN NEW LANGUAGE</td>
<td>70</td>
</tr>
<tr>
<td>VITA</td>
<td></td>
<td>73</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table                                      Page

1. Sample fragment of validator confidence values ...................... 11
2. The results table showing the performance of the old language .................. 45
3. The results table showing the performance of the new language .................. 46
4. The table showing the sample of erroneous titles using old language .................. 47
5. The table showing the performance of the titles using old language .................. 47
6. The table showing the sample of erroneous titles using new language .................. 48
7. The table showing the performance of the titles using new language .................. 49
8. The table showing the sample of erroneous authors using old language .................. 49
9. The table showing the performance of the authors using old language .................. 49
10. The table showing the sample of erroneous authors using new language .................. 50
11. The table showing the performance of the authors using new language .................. 50
12. The table showing the performance of the abstracts using old language .................. 51
13. The table showing the performance of the abstracts using new language .................. 51
14. The table showing the results of the quiz ........................................ 53
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Documents with similar layout</td>
<td>2</td>
</tr>
<tr>
<td>2. Metadata extraction flow diagram</td>
<td>3</td>
</tr>
<tr>
<td>3. Form-based template fragment</td>
<td>7</td>
</tr>
<tr>
<td>4. Non-form template fragment</td>
<td>8</td>
</tr>
<tr>
<td>5. Validation script fragment for classifying DTIC collection</td>
<td>9</td>
</tr>
<tr>
<td>6. Sample fragment of validator confidence values</td>
<td>10</td>
</tr>
<tr>
<td>7. Final validation script fragment for DTIC collection</td>
<td>12</td>
</tr>
<tr>
<td>8. AU document</td>
<td>22</td>
</tr>
<tr>
<td>9. Original template</td>
<td>23</td>
</tr>
<tr>
<td>10. New template</td>
<td>24</td>
</tr>
<tr>
<td>11. Interpreter framework</td>
<td>26</td>
</tr>
<tr>
<td>12. Interpreter flow chart</td>
<td>27</td>
</tr>
<tr>
<td>13. Build the input document into a tree form</td>
<td>27</td>
</tr>
<tr>
<td>14. Let function flow chart</td>
<td>29</td>
</tr>
<tr>
<td>15. Example 1 Let tag</td>
<td>29</td>
</tr>
<tr>
<td>16. Code snippet to evaluate XPath expression</td>
<td>29</td>
</tr>
<tr>
<td>17. Value of the XPath expression in Example 1</td>
<td>30</td>
</tr>
<tr>
<td>18. Bind function flow chart</td>
<td>31</td>
</tr>
<tr>
<td>19. Example 2 Bind tag</td>
<td>31</td>
</tr>
<tr>
<td>20. Code snippet to evaluate XPath expression with variables</td>
<td>32</td>
</tr>
<tr>
<td>21. Value</td>
<td>32</td>
</tr>
<tr>
<td>22. Meta data extracted</td>
<td>32</td>
</tr>
<tr>
<td>23. Example to demonstrate various XPath functions</td>
<td>32</td>
</tr>
<tr>
<td>24. Example to demonstrate various XPath functions</td>
<td>33</td>
</tr>
<tr>
<td>25. Flow chart to demonstrate the WithinAndMargins extension function</td>
<td>34</td>
</tr>
<tr>
<td>26. Flow chart to demonstrate the Larger size words extension function</td>
<td>35</td>
</tr>
<tr>
<td>27. Flow chart to demonstrate the Vertical Split extension function</td>
<td>36</td>
</tr>
</tbody>
</table>
28. Example 3 Bind Tag with Third party extension function .......................... 36
29. Code snippet to evaluate XPath expression with functions ............................. 37
30. Output date ................................................................................................. 37
31. An example to select a page ........................................................................ 40
32. Sample document of the head-abstract class ................................................... 43
33. A Sample variant of the head-abstract class ..................................................... 44
34. Afit document .............................................................................................. 58
35. Afit template written in old language .............................................................. 59
36. Afit template written in new language ............................................................. 59
37. Acgsc document .......................................................................................... 60
38. Acgsc template written in old language .......................................................... 61
39. Acgsc template written in new language .......................................................... 62
40. Afrl4 document ........................................................................................... 63
41. Afrl4 template written in old language ............................................................ 64
42. Afrl4 template written in new language ........................................................... 64
43. Usacerl document ......................................................................................... 65
44. Usacerl template written in old language ........................................................ 66
45. Usacerl template written in new language ........................................................ 67
46. Head-abstract document .............................................................................. 68
47. Head-abstract template written in old language .............................................. 69
48. Head-abstract template written in new language .............................................. 70
49. Head-abstract-TopMargin document ............................................................ 71
50. Head-abstract-TopMargin template written in new Language ...................... 72
CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

In the recent years, there has been a tremendous growth in Internet and online resources that had previously been restricted to paper archives. OCR (Optical Character Recognition) tools can be used for digitalizing an existing corpus and making it available online. A number of federal agencies, universities, laboratories, and companies are placing their collections online and making them searchable via metadata fields such as author, title, and publishing organization. Manually creating metadata for a large collection is an extremely time-consuming task, and is difficult to automate, particularly for collections consisting of documents with diverse layout and structure. According to Crystal [6], to create metadata for 1 million documents would take about 60 employee-years. Because of the high costs involved with manual metadata creation, there is a need for automated metadata extraction tools. To identify the challenges in automatic metadata creation, a study named Automatic Metadata Generation Applications (AMEGA) [8], is sponsored by The Library of Congress Cataloging Directorate [7].

The visual cues in the formatting of the document along with accumulated knowledge and intelligence make it easy for a human to identify various metadata fields. Writing a computer program to automate this task is a huge challenge. Many researchers made it possible to automate the process of metadata extraction for homogenous collection (a collection consisting of documents with a common layout and structure). Unfortunately those procedures did not work well for heterogeneous collections. A number of federal organizations such as Defense Technical Information Center (DTIC), U.S. Government Printing Office (GPO), and National Aeronautics and Space Administration (NASA) manage heterogeneous collections. Even with the presence of external sources of error (for example, text obscured by smudges, signatures, or stamps), the process for metadata extraction must remain robust. It is still a major research issue to automate such “intelligent” behavior to reach the desired accuracy and robustness. The digital library group at ODU has developed an automated metadata extraction system to support document collections with diverse structure and layout.

The extraction problem is broken into two phases. Firstly, the documents are grouped together whose title or other metadata-containing pages would appear similar when viewed
FIG. 1: Documents with similar layout

(by humans) from several feet away. For example, look at Fig. 1.

Then a template is associated with each class of document layouts. A template is a scripted description of how to associate blocks of text in the layout with metadata fields. For example, a template might associate the text set in the largest type font in the top-half of the first page to document title. The template-based metadata extraction system is composed of commercial and public domain software in addition to components developed by our team. Fig. 2 shows the complete process. Documents are input into the system in the form of PDF files, which may contain either text PDF or scanned images. Some documents may contain a Report Document Page (RDP), one of several standardized forms that is inserted into the document when the document is added to the collection. For the DTIC collection, more than 50% of the documents contain RDPs offering more than 20 metadata fields.

When the documents enter the input processing system they are truncated and are converted to a standardized XML format by an Optical Character Recognition (OCR) program. The first extraction step is to search for and recognize any RDP forms present. Any
documents without recognized forms enter the non-form extraction process. The non-form extraction process generates a candidate extraction solution from the templates available. After extraction, the metadata from both form and non-form processing enter the output processor. The output processor is comprised of two components: a post-processing module and a validation module. The post-processing module handles cleanup and normalization of the metadata. The final automated step of the process is the validation module which, using an array of deterministic and statistical tests, determines the acceptability of the extracted metadata. Any document that fails to meet the validation criteria is flagged for human review and correction.

1.2 PROBLEM STATEMENTS

This thesis mainly focuses on the Non-form Processing engine. The template structure currently used for non-form processing poses some limitations namely

1. Non-concise representation.

2. Ad-hoc semantics - The template rules are designed for specific tasks and cannot be easily extended to add further functionality.
3. Limited ability to express or exploit document structure. For example the rules can express “find the line containing AIR COMMAND” but cannot express “find the paragraph containing AIR COMMAND”.

4. Cannot combine basic search/test functions. For example the rules can’t express “title came after a header AND would start with the phrase ‘Fact Sheet’”. It can express only one of them but not both.

1.3 APPROACH

The proposed language consists of high-level syntactic elements and lower-level text expressions. The syntactic elements are the <template> element that serves as the root of the entire template, <let> elements that define a variable name, and <bind> elements that establish a value for a metadata field. The text expressions consist of XPath expressions, which are evaluated in a context consisting of the document root and the variable bindings from the LET.

1.4 OBJECTIVES

The main objective of this thesis involves the implementation of a Java based interpreter which executes the new template language to extract metadata from the documents. We downloaded a collection of documents from DTIC (Defense Technical Information Center) [9] for testing. We have the below-mentioned items as the objectives.

1. Develop templates for documents using the new XPath template language [12].

2. Implement a Java based interpreter which executes the templates to extract metadata from the documents.

3. Compare the new language against the old language and answer the following questions.

4. Power of the language

   (a) Does the new language cover all the functionality of the old language?

   (b) Can we write effective templates using the new language?

5. Effectiveness of the language
(a) Can a beginner to template writing understand the new language, is it easy or difficult to understand the templates?

(b) Using the new template language, is it easy to write templates?

1.5 ORGANIZATION OF THE THESIS

The rest of the thesis is organized as follows:

Chapter 2 Previous Work: In Chapter 2, we will present the previous work to extract the metadata from a document.

Chapter 3 Xpath Template Language: In Chapter 3, we will introduce you to the new template language. We will explain the structure of the template, the rules and functions of the Xpath language and the extension functions.

Chapter 4 System Implementation: In Chapter 4, we will show the details of System implementation. In this chapter, we will present the overall architecture of the non-form engine and the implementation details of the XPath functions and the extension functions explained in Chapter 3.

Chapter 5 Comparing the two languages: In Chapter 5, we will compare the two languages by answering the questions mentioned in the objectives section.

Chapter 6 Conclusions and future work: Finally, in Chapter 6, we will summarize the contributions of our work as well as the issues we addressed. In this chapter, we will also provide directions for the future work.
CHAPTER 2
PREVIOUS WORK

The overall metadata extraction system architecture is described in Section 1.1. In this section we will look at the specific implementation details.

2.1 INPUT PROCESSING

The source documents come into the system as PDF format files. These documents range from several pages to hundreds of pages in length. The research into the collections has shown that the metadata we are interested in can typically be found in the first or last five pages of a document. Based on this observation, the program PDFBox[13] is used to split the first and last five pages out of the document into a new PDF document. This truncated PDF document might contain text PDF (born digital) pages or image PDF (scanned images of pages) pages. Text PDF pages are rendered into an IDM-like format called “raw IDM”. Any of the desired pages that appear likely to be image PDF are written into a new PDF file using PDFBox. The PDF file is sent through OCR and the OCR-engine-specific format is converted to raw IDM. Many documents will have all of their pages handled as text PDF, small number of documents will have all of their pages treated as image PDF, and some documents will have a mixture of the two. The raw IDM versions of the text PDF pages and the image PDF pages are merged and segmented, organized into words, lines, paragraphs, regions, etc.

2.1.1 INDEPENDENT DOCUMENT MODEL (IDM)

The new Independent Document Model (IDM) is based on the OmniPage 14 schema that is already supported with the project, which helped in minimizing the re-coding cost for the extraction engine. The main structural elements are pages, regions, paragraphs, lines and words. The geometric boundaries of each of the structural elements are included as attributes. At the line and word levels, style information such as font face, font size and font style is recorded. At the paragraph elements, alignment and line spacing are recorded. Tables are composed of a sequence of cells that represent a virtual row-column table with each cell encoded with the upper-left coordinate and the row and column spans of the cell. Using XSL 2.0 stylesheets, IDM documents are created from OCR outputs. For each type of source document a different style-sheet is used. To support creation of IDM documents
from either OmniPage 14 or 15, stylesheets are created. The decision to pursue the IDM model was validated when DTIC changed their preferred OCR engine to the Luratech ABBY OCR program. Creation of an XSL stylesheet to convert Luratech to IDM took less than 20 man hours.

## 2.2 FORM PROCESSING

When the IDM based documents exit the input processor and enter the form processor, the input processing finishes. Experience with the DTIC collection has shown that roughly 50% of the documents contain an RDP form. The regular layout present in an RDP form makes it an attractive target for a template-based extraction process. Taking the advantage of the geometric relationships between fields in a form, a form template language and extraction engine are created. The metadata fields are specified by a matching string and a set of rules indicating a positional relationship to one or more other fields (e.g., Fig. 3). Each form differs from the other by the number and layout of the fields. This constitutes a unique identifier for that form class. If a template describing form A is applied to a document containing form B, the resultant metadata returned will contain few if any fields.

As shown in Fig. 3, the (line) elements in the (field) elements define string matching criteria. The (rule) elements defined for each (metadata) element defines the geometric placement.
The form processor is populated with a template developed for each version of RDP form found in the collection. Eleven different RDP forms are found within 9825 documents in the DTIC collection. The form processor attempts to identify cells based on a combination of cell titles and geometric constraints (described in the form template). The closest-matching template is selected and extraction takes place using only that one template. If the form processor fails to match any template the document moves into the non-form extraction process described below. The extracted metadata is sent into the output processor.

2.3 NON-FORM PROCESSING

As shown in Fig. 2, documents without an RDP form enter the non-form processor. Both the form and non-form engines operate upon a "scroll", a line-by-line index into the IDM document tree. This simplified structure allows the extraction engine to repeatedly iterate over the content to apply the rules.

2.3.1 TEMPLATE CONSTRUCTION

A rule-based template language is used by the non-form extraction engine to locate and extract metadata. Fig. 4 shows a template example. Each template contains a set of rules designed to extract metadata from a single class of similar documents. Each desired metadata item is described by a rule consisting of a pair of selectors designating the beginning and
FIG. 5: Validation script fragment for classifying DTIC collection

the end of the metadata. The rules are limited by features detectable at the line level resolution. This deficiency is addressed in the new template language which will be described in the later sections. Depending on the structural or visual similarity, the documents are manually classified into different groups. For purposes of our discussion we define a class as a group of documents from which the metadata can be extracted using the same template. Once a class is selected, the template author determines the selectors for each metadata field by identifying the appropriate function to select the beginning and the end of the field.
2.4 NON-FORM CLASSIFICATION

In the original design of the system in order to separate the incoming documents into the appropriate class for extraction [14, 2], several different layout classification schemes are used. A validation system is created to flag suspicious data extracted by a template [3, 4]. By applying every available template to a document, the validator can be used as a post hoc classification system for selecting the proper template. This post hoc classification system is configured by creating a “validation script” (e.g., Fig. 5), which defines a set of rules to be used for calculating a confidence value for individual fields as well as an overall confidence calculation. Fig. 6 is an example of the validator output for the “alr.2” template. Table 1 shows the validation values for five of the eleven templates applied by the extraction system for the same file. (The other six templates did not produce any output for the file.) The best result, “alr.2”, differs from the next best, “alr.1”, by the extraction of an additional personal author.

In the example shown in Fig. 6, the second CorporateAuthor gives a low confidence score because of the existence of too many words not in the CorporateAuthor dictionary.
TABLE 1: Sample fragment of validator confidence values

<table>
<thead>
<tr>
<th>Template</th>
<th>Total Confidence</th>
<th>Field Confidences</th>
<th>Personal Author</th>
<th>Corporate Author</th>
<th>Report Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>alr_2</td>
<td>4.694</td>
<td>0.891</td>
<td>0.785</td>
<td>0.76</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.713</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.546</td>
<td></td>
</tr>
<tr>
<td>alr_1</td>
<td>3.436</td>
<td>0.891</td>
<td>0.785</td>
<td>0.76</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>nsrp</td>
<td>1</td>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>rand</td>
<td>0.848</td>
<td>0.848</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>nps_thesis</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

2.5 OUTPUT PROCESSING

Referring back to the architecture diagram in Fig. 2, the extracted metadata from both form and non-form processes enter output processing for post-processing cleanup and validation.

2.5.1 POST-PROCESSING

The post-processing step is designed to accommodate differences between metadata as it appears in the document and metadata as it is formatted in the database indexing a specific collection. The post-processing system is modularized to provide a variety of post-processing functions for each metadata field. For example, modules may be designed to re-format date fields into a specific standard and to parse multiple authors from a single personal or corporate author entry. As an example of a post-processing module, there is a module for metadata fields with a finite, enumerable set of possible field values. This module attempts to overcome the potential for misrecognition by the OCR software and to standardize variant wordings in the input documents. The module analyzes specific fields by comparing the extracted data to values in an authority file. These values are compared by the module via fuzzy string matching based on edit distance. Additionally, the post processor can match variable phases where the comparison is successful so long as every word in the authority file entry is contained in the extracted data. The authority file is generated by extracting field data from more than 9000 documents.
2.5.2 VALIDATION

The final step in the process is the validation step. The primary purpose of this step is to determine whether or not to flag the extracted metadata for human review. The same validation engine as mentioned above in the post hoc classification is used. This validation engine uses statistical models of previously extracted metadata in the collection along with dictionaries for names and specialized content to determine the norms for the collection.

As shown in Fig. 7, a different validation script for final validation is used. In validation an attempt is made to fix a lower bound for determining whether or not to accept the extraction, while the classification script sums all the component fields to find the template which extracts the most potentially correct data.

2.6 EXPERIMENTAL RESULTS

2.6.1 NON-FORM EXTRACTION EXPERIMENTS

To confirm the efficiency of the post hoc classification system and the ability to extract the metadata experiments are conducted. To test the ability of the system to select the
appropriate template for extraction, the DTIC non-form documents are manually classified into 37 separate classes with at least 5 members. Templates are written for the 11 largest classes and the ability of the extractor to correctly identify the proper class is tested. A classification accuracy of 87% is achieved when compared to manual classification results. The overall accuracy for the non-form extractor is 66% for DTIC. Due to the fact that a limited number of templates are written, the accuracy value is low.

2.7 LIMITATIONS OF PRESENT NON FORM ENGINE

The template structure currently used for non-form processing poses some limitations namely

1. Non-concise representation.

2. Ad-hoc semantics - The template rules are designed for specific tasks and cannot be easily extended to add further functionality.

3. Limited ability to express or exploit document structure. For example the rules can express “find the line containing AIR COMMAND” but cannot express “find the paragraph containing AIR COMMAND”.

4. Cannot combine basic search/test functions. For example the rules can’t express “title came after a header AND would start with the phrase ‘Fact Sheet’”. It can express only one of them but not both.
CHAPTER 3
XPATH TEMPLATE LANGUAGE

3.1 XPATH

3.1.1 INTRODUCTION

The data model defined in XQuery/XPath Data Model (XDM)[21] provides a tree representation of XML documents as well as atomic values such as integers, strings, and booleans, and sequences that may contain both references to nodes in an XML document and atomic values.

XPath 2.0[15] is an expression language that allows the processing of values conforming to the data model. The path expression, provides a means of hierarchic addressing of the nodes in an XML tree. Using an XPath expression, particular nodes can be selected from the input documents, or an atomic value, or more generally, any sequence allowed by the data model can be extracted.

The basics and the grammar of the XPath can be learnt from the W3C specs included in the references. I would like to jump directly into locationPaths and core function libraries that would be useful in writing the non form templates.

3.1.2 LOCATIONPATH

Evaluating a location path, which is the important kind of expression, results in a set of nodes relative to the context node. Recursive expressions in a location path can be used to filter sets of nodes. A straightforward but rather verbose syntax is used for expressing a location path.

Using abbreviated syntax, this section will explain the semantics of location paths. Some of these are quoted from XPath W3 document [15].

1. . - Context node is selected.

2. section - section element children of the context node are selected.

3. * - All element children of the context node are selected.

4. text() - All text node children of the context node are selected.

5. @name - The name attribute of the context node is selected.
6. `@*` - All attributes of the context node are selected.

7. `section[1]` - The first section child of the context node is selected.

8. `section[last()]` - The last section child of the context node is selected.

9. `*/section` - All section grandchildren of the context node are selected.

10. `/doc/chapter[5]/section[2]` - The second section of the fifth chapter of the doc is selected.

11. `chapter/section` - The section element descendants of the chapter element children of the context node are selected.

12. `//chapter` - The chapter descendants of the document root and thus select all chapter elements in the same document as the context node are selected.

13. `//chapter/title` - All the title elements in the same document as the context node that have an chapter parent are selected.

14. `//para` - The para element descendants of the context node are selected.

15. `..` - The parent of the context node is selected.

16. `../@lang` - The lang attribute of the parent of the context node is selected.

17. `para[@type="italic"]` - All para children of the context node that have a type attribute with value italic are selected.

18. `para[@type="italic"][2]` - The second para child of the context node that has a type attribute with value italic is selected.

19. `para[2][@type="italic"]` - The second para child of the context node if that child has a type attribute with value italic is selected.

20. `chapter[title="Introduction"]` - The chapter children of the context node that have one or more title children with [string-value] equal to Introduction is selected.

21. `chapter[title]` - The chapter children of the context node that have one or more title children are selected.
3.1.3 CORE FUNCTION LIBRARY

There are many functions which help in evaluating expressions. This section describes some of the functions that XPath supports. More detailed information can be obtained from the XPath-functions W3 document [20].

Each function in the function library is specified using a function prototype, which gives the return type, the name of the function, and the type of the arguments. If an argument type is followed by a question mark, then the argument is optional; otherwise, the argument is required.

Node Set Functions

This subsection gives a brief about Node Set functions

- number last() - Returns a number equal to the [context size] from the expression evaluation context.
- number position() - Returns a number equal to the [context position] from the expression evaluation context.
- number count(node-set) - Returns the number of nodes in the argument node-set.

String Functions

This subsection gives a brief about String functions.

- string string(object?) - Converts an object to a string.
- string concat(string, string, string) - Returns the concatenation of its arguments.
- boolean starts-with(string, string) - Returns true if the first argument string starts with the second argument string, and otherwise returns false.
- boolean contains(string, string) - Returns true if the first argument string contains the second argument string, and otherwise returns false.
- string substring-before(string, string) - Returns the substring of the first argument string that precedes the first occurrence of the second argument string in the first argument string, or the empty string if the first argument string does not contain the second argument string. For example, substring-before("metadata extraction project"," ") returns metadata.
• string substring-after(string, string) - Returns the substring of the first argument string that follows the first occurrence of the second argument string in the first argument string, or the empty string if the first argument string does not contain the second argument string. For example, substring-after("metadata extraction project"," ") returns extraction project, and substring-after("metadata extraction project","data ") returns extraction project.

• string substring( string , number , number? ) - Returns the substring of the first argument string with the second argument as the starting position and with the third argument as the length. For example, substring("vijay",2,3) returns “ija”. If the third argument is not specified, it returns the substring starting at the position specified in the second argument and continuing to the end of the string. For example, substring("vijay",2) returns “ijay”.

3.2 TEMPLATE LANGUAGE

This section explains the new template language [12]. With the permission from the author, the technical details are quoted from [12].

“The proposed language consists of high-level syntactic elements and lower-level text expressions. The syntactic elements are the <template> element that serves as the root of the entire template, <let> elements that define a variable names, and <bind> elements that establish a value for a metadata field. The text expressions consist of XPath expressions, which are evaluated in a context consisting of the document root and the variable bindings from the LET elements (described below).

3.2.1 <TEMPLATE>

The <template> element serves as the root of the template structure.

Attributes:

• @id: an identifier for the template and, by implication, for the class of documents processed by this template.

Children:

• a list of <let> and/or <bind> elements.

Semantics:
• Each child is evaluated in sequence.
• If any child fails, the template fails.
• If all children succeed, return the concatenation of the metadata bindings from all children.

3.2.2 <BIND>

The <bind> element establishes a binding between a metadata field name and a string denoting a value for that field. It can also supply attribute values to annotate that binding.

Attributes:

• @name: an identifier for a metadata field.
• @select: a XPath expression.
• @optional: 0 or 1 (defaults to 0).
• others as desired.

Children:

• none.

Semantics:

• The @select attribute expression is evaluated.
• If @optional != 1 and evaluation of @select fails or returns an empty list of nodes, the <bind> fails.
• If @optional = 1 and evaluation of @select fails or returns an empty list, return an empty list of bindings.
• If evaluation of this expression succeeds, return a list containing one binding associating the text value of the selected nodes with the field name indicated in the @name attribute. Any other attributes of <bind> are attached to this binding as descriptive attributes.
3.2.3  <LET>

The <let> element establishes a binding between an XPath variable name and the value of an XPath expression.

Attributes:

- @name: an identifier for an XPath variable.
- @select: a XPath expression.
- @optional: 0 or 1 (defaults to 0)

Children:

- a list of <let> and/or <bind> elements.

Semantics:

- The @select attribute expression is evaluated.
- If @optional != 1 and evaluation of @select fails, the <let> fails.
- If @optional = 1 and evaluation of @select fails, return an empty list of bindings.
- If evaluation of @select succeeds, evaluate each child in a context augmented by the binding of the @select expression result to the XPath variable indicated by the @name attribute.
  - If @optional != 1 and any child fails, the <let> fails.
  - If @optional = 1 and any child fails, return an empty list of bindings.
  - If all children succeed, return the concatenation of the lists of bindings returned by the children.

3.3  PROBABLE XPATH EXTENSIONS

XPath engines commonly provide simple mechanisms for extending the set of available functions. We would exploit this to tailor this language for the purpose of examining documents.

This section discusses some likely extension functions.
3.3.1 THIRD-PARTY EXTENSIONS

From the EXSLT project, the XPath functions in the modules dates and times, math, regular expressions, sets, and strings.

3.3.2 GEOMETRIC PROPERTIES

WithinAndMargins

WithinAndMargins(nl,l,t,r,b), for the nodes in the nodelist nl, recursively discard any leaves that have no text values or whose bounding boxes lie even partly outside the rectangle formed using l,t,r and b values (see how the rectangle is constructed below). And also the rectangle can be constructed in such a way that header or tail or left or right lines i.e., the marginal lines can be selected.

Construction of the rectangle - For example, a nodelist of all paras in a page and 0.2,0.3,0.1,0.3 are given as values for nl,l,t,r,b respectively. A bounding box is constructed using the nodelist and now using the x,y,w,h co-ordinates of this bounding box helps in constructing the user defined rectangle. The user defined rectangle will be such that it has its left side at a distance of l(0.2) towards inside the bounding box of nodeList, top side at a distance of t(0.3) towards inside, right side at a distance of r(0.1) towards inside and bottom side at a distance of b(0.3) towards inside.

If the user wants to get the top margin he would give the parameters as withinAndMargins(nl,,0.9)(others i.e., l,t,r default to zero).

If the user wants to get the bottom margin he would give the parameters as withinAndMargins(nl,,0.9,).

vsplit

vsplit(nl,k) The nodes in the list nl are treated as a long scroll and the k-1 largest internal vertical whitespace boxes selected within that scroll. Return a list of k region nodes representing the contents of the k boxes separated by those selected whitespace areas.

hsplit

hsplit(nl,k) Like vsplit, but based upon horizontal separation.
3.3.3 DOCUMENT PROPERTIES

LargerSizeWds

LargerSizeWds(nl) - Returns a list of dominant font words in the list nl.

3.4 EXAMPLE

A sample document for "au" class of documents can be found in Fig. 8.
The template written in old language can be found in Fig. 9.
A rewrite of the au template can be found in Fig. 10.
IDENTIFYING AND MITIGATING THE RISKS OF COCKPIT AUTOMATION

by

Wesley A. Olson, Major, USAF

A Research Report Submitted to the Faculty
In Partial Fulfillment of the Graduation Requirements

Adviser: Lieutenant Colonel Steven A. Kimbrell

Maxwell Air Force Base, Alabama
April 2000

FIG. 8: AU document
FIG. 9: Original template
FIG. 10: New template
CHAPTER 4
SYSTEM IMPLEMENTATION

4.1 INTERPRETER FRAMEWORK

Writing a LISP interpreter is so simple. Our interpreter framework is not a whole lot more complicated. The code snippet of the interpreter is shown in the Fig. 11.

Basically, to interpret a tree, we treat the XML tag as the name of an operator, look it up in a table (boundOps) to get the actual operator, and “eval” that operator. The most common implementation of “eval” is

1. Interpret all of this node’s children, collecting all the return values from their execution.

2. Perform an operator-specific calculation on those return values to compute the return value of this node.

4.2 EVALUATING THE XML TAGS

The interpreter can be best explained using the Fig. 12.

1. Both the template and the non-form XML document enter the JDOM interpreter. An instance of SAXBuilder is used to build the template (which is in XML) into a JDOM document tree.

2. And the DocumentBuilder provided by S9API of SAXON Processor is used to build the input pdf document (which is in XML) into a tree form.

3. The SAXON Processor implements a tiny tree model as its internal data structure which is faster to build, occupies less space but is slower to navigate when compared to the Linked Tree model. The code snippet is shown in the Fig. 13.

4. Using the code snippet, shown in Fig. 12, the interpreter starts processing the template starting with the root node <template> followed by the children <let> and <bind> nodes of the <template>. The corresponding functionality provided for each of these elements in the JDOM interpreter is as follows.
public class Interpreter {

public Value interpret(Content ASTroot) throws InterpretationError {
    if (ASTroot instanceof Element) {
        Element el = (Element)ASTroot;
        String tagName = el.getName();
        String namespace = el.getNamespaceURI();
        if (namespace == null)
            namespace = "";
        Operation<Value> op = boundOps.getOp(tagName, namespace);
        if (op == null)
            throw new InterpretationError("No operation bound to " + namespace + ":" + tagName);
        return op.eval(this, el);
    } else if (ASTroot instanceof Text) {
        TextOperation<Value> t_op = boundOps.getTextOp();
        if (t_op == null)
            throw new InterpretationError("No operation bound to text nodes");
        return t_op.eval(((Text)ASTroot).getText());
    } else {
        throw new InterpretationError("Cannot interpret " + ASTroot);
    }
}

FIG. 11: Interpreter framework

4.2.1 TEMPLATE TAG IMPLEMENTATION

Function template( ) for <template> :

1. As shown in the Fig. 12, the attribute "templateID" is used to get the name of the template, that is being used to extract the metadata. And the template name is stored in the global symbol table.

2. Each child is evaluated in sequence.

3. If any child fails, the template fails.

4. If all children succeed, return the concatenation of the metadata bindings from all children.

4.2.2 LET TAG IMPLEMENTATION

Function Let( ) for <let> :
FIG. 12: Interpreter flow chart

// Create an instance of the saxon processor.
Processor proc = new Processor(false);

DocumentBuilder builder = proc.newDocumentBuilder();

// idm is the input pdf in the form of an xml
XdmNode Doc = builder.wrap(idm);

FIG. 13: Build the input document into a tree form
1. The `@select` attribute expression is evaluated.

2. The evaluation of the expression is explained using the Fig. 14.

3. If there are any extension functions used in the expression, do the specific extension functionality. (The Xpath functions and the extension functions are explained in the further subsections of the chapter).

4. Else check if the expression uses any previously resolved variables, get them from the symbol table.

5. If evaluation of `@select` succeeds, evaluate each child in a context augmented by the binding of the `@select` expression result to the XPath variable indicated by the `@name` attribute.

   (a) If all children succeed, return the concatenation of the lists of bindings returned by the children.

6. Add the result to the symbol table for further use.

   For example, as shown in the Fig. 15, the first page of the document is bound the variable “pagel” and is stored in the symbol table. The code snippet is shown in Fig. 16 and the value of the expression is shown in Fig. 17.

### 4.2.3 BIND TAG IMPLEMENTATION

The variables defined by the `<let>` nodes, take values of type `XdmValue` which is a sequence of `XdmNodes`. Each Node is a sub-tree.

Function Bind() for `<bind>`:

1. The `@select` attribute expression is evaluated.

2. The evaluation of the expression is explained using the Fig. 18.

3. Check if the expression uses any previously resolved variables, get them from the symbol table.

4. If evaluation of this expression succeeds, return a list containing one binding associating the text value of the selected nodes with the field name indicated in the `@name` attribute. Any other attributes of `<bind>` are attached to this binding as descriptive attributes.
FIG. 14: Let function flow chart

\[
\text{<let name ="page1" select="doc/page[1]">}
\]

FIG. 15: Example 1 Let tag

```java
Processor proc = new Processor(false) ;
XPathCompiler xpath = proc.newXPathCompiler();
XPathSelector selector = xpath.compile(expression).load();
selector.setContextItem(Doc);
XdmValue val = (XdmValue)selector.evaluate();
m.put(name, val);
```

FIG. 16: Code snippet to evaluate XPath expression
For example, as shown in the Fig. 19, the line that starts with “AU/” is bound to the variable “identifier”. The code snippet is shown in fig 20 and the value of the expression is shown in Fig. 21.

Finally from the global symbol table, the metadata is collected. The metadata field identifier, is similarly found from the global table and looks as shown in the Fig. 22.

### 4.3 XPATH FUNCTIONS

As discussed in section III.1, various XPath functions. In this section I would like to explain the usage of some of these functions with examples.

The sample code in Fig. 23 is taken from the template “au” which is written to extract metadata from the document “ADA394844”. The templates and the document can be found in Section 3.4.

- //para

  line 1, in the code selects the para, which has words either “COMMAND” and “AIR” or “WAR” and “AIR”.

- “.” and `>>`

  line3, “.” in the expression denotes the context note, which is “page1” and the expression “page1//para[. >>CApar]” grabs all para nodes after the “CApar” node into afterCA.
Do Bind

Xpath uses previous resolved variables

yes

Get them from the symbol table

no

Evaluate the xpath using XPathSelector

Add to global symbol table

FIG. 18: Bind function flow chart

<bind name ="identifier" select ="$page1//[starts-with(.,'AU/')]"/>

FIG. 19: Example 2 Bind tag
<bind name="identifier"Processor proc = new Processor(false) ;
XPathCompiler xpath = proc.newXPathCompiler();
xpath.declareVariable(new QName(var[i]));
XPathSelector selector = xpath.compile(expression).loadQ;
selector.setContextItem(Doc);
selector.setVariable(new QName(var[i]), var_value[i]);
XdmValue val = (XdmValue)selector.evaluate(); select = "Spagell.line[wd[starts-with(.,'AU')]*/">

FIG. 20: Code snippet to evaluate XPath expression with variables

<line l="1440" t="1488" r="1577" b="1661">AU/ACSC/138/2000-04</line>

FIG. 21: Value

<identifier>AU/ACSC/012/1999-04</identifier>

FIG. 22: Meta data extracted

1) <let name="C.Apar" select="$Page1/para[line[wd = 'COMMAND' and wd = 'WAR' or (wd = 'WAR' and wd = 'AIR')]]" />  
2) <bind name="CorporateAuthor" select="$C.Apar" optional="true" />  
3) <let name="afterCA" select=$Page1/para[ > $C.Apar]" />  
4) <let name="authorPar" select="$Page1/para[line[wd=starts-with(., 'by')]]" />  
5) <bind name="DeclCiteTitle" select="$afterCitationTitle/AuthorPar >> J" />  
6) <let name="PlAuthor" select=$Page1/para[line[wd=starts-with(.,'by')]]" />  
7) <bind name="PersonalAuthor" select="$PAuthor/line[2]" />  
8) </let>  
9) </let>  
10) </let>  
11) </let>

FIG. 23: Example to demonstrate various XPath functions
• starts-with()

   line4, starts-with() is a string function and, the expression grabs all lines that have words that start with “by”.

   line5, the expression grabs all lines in “afterCA” node, that comes before the “authorPar” line.

• line[2]

   line7, the expression grabs the second line of the “PAuthor” node.

The sample code in Fig. 24 is taken from the template “head-abstract” which is written to extract metadata from the document “ADA394844”. The template and the document can be found in Appendix A.

matches()

The function returns true if input matches the regular expression supplied as pattern otherwise, it returns false. Here, it grabs the para, which contains a word “Abstract” or “ABSTRACT”.

### 4.4 EXTENSION FUNCTIONS

In this section, we will discuss the implementation details of the extension functions which are mentioned in section 3.3.

**WithinAndMargins extension function**

The function can be best explained using the flow chart given in the fig 25.

Segmentation - The given input is divided into logical parts, which can be used later is done as folows.

1. CollectIDM - From the nodeList which is given as the input, the leaf nodes i.e., words are collected into a linked list. The words are decoupled from each other such that the parent hierarchy of the word nodes is maintained.
2. Compute the bounding boxes - Using the union of bounding boxes of the word nodes in the linked list, a bigger bounding box is created.

WithinAndMargins functionality - For example, (nodeList,0.1,0.1,0.1,0.1) are the arguments given, it means that the user wants to get all the words that lie in the box, whose sides are at 0.1 distance inside the bounding box of the nodeList.

3. Check if the expression uses any previously resolved variables, get them from the symbol table.

4. If evaluation of this expression succeeds, return a list containing one binding associating the text value of the selected nodes with the field name indicated in the @name attribute. Any other attributes of <bind> are attached to this binding as descriptive attributes.

5. Construct the rectangle using the arguments and the bounding box of the nodeList.

6. Collect all words that lie inside the rectangle constructed.

RenderIntoIDM - This is the process in which the collected words are re-grouped such that a nodeList similar to the input is generated.

LargerSizeWd extension function

The function can be best explained using the flow chart given in the Fig. 26.
Segmentation - The given input is divided into logical parts, which can be used later is done as follows.

1. CollectIDM - From the nodeList which is given as the input, the leaf nodes i.e., words are collected into a linked list. The words are decoupled from each other such that the parent hierarchy of the word nodes are maintained.

2. Compute the bounding boxes - Using the union of bounding boxes of the word nodes in the linked list, a bigger bounding box is created.

LargerSizeWd functionality - Compute the largest font size among the words and collect the words that have largest font size.

RenderIntoIDM - This is the process in which the collected words are re-grouped such that a nodeList similar to the input is generated.

**Vertical Split extension function**

The function can be best explained using the flow chart given in the Fig. 27.

Segmentation - The given input is divided into logical parts, which can be used later is done as follows.

1. CollectIDM - From the nodeList which is given as the input, the leaf nodes i.e., words are collected into a linked list. The words are decoupled from each other such that the parent hierarchy of the word nodes are maintained.
2. Compute the bounding boxes - Using the union of bounding boxes of the word nodes in the linked list, a bigger bounding box is created.

Vertical Split functionality - Using the bounding boxes of the words, the intervals between them are identified and sorted depending on the length of the interval. Now depending on the number of splits the user wants to make, the words are iteratively split into regions.

RenderIntoIDM - This is the process in which the collected words are re-grouped such that a nodeList similar to the input is generated.

4.5 THIRD PARTY EXTENSIONS

Using the example in the Fig. 28, an extension function to get the date is shown. The output is shown in the Fig. 30.
Processor proc = new Processor(false);
XPathCompiler xpath = proc.newXPathCompiler();
xpath.declareNamespace("date", "java:java.util.Date");
XPathSelector s = xpath.compile(path).load();
s.setContextItem(Doc);
XdmAtomicValue val = (XdmAtomicValue)s.evaluate();

FIG. 29: Code snippet to evaluate XPath expression with functions

<Current Date> Thu Nov 04 19:58:30 EDT 2010 </Current Date>

FIG. 30: Output date
CHAPTER 5
COMPARING THE TWO LANGUAGES

As mentioned in section 2.7, we would like to address the limitations of the old language and how the new language helps in overcoming those.

Non-concise representation

The proposed language consists of high-level syntactic elements and lower-level text expressions. The syntactic elements are the <template> element that serves as the root of the entire template, <let> elements that define a variable name, and <bind> elements that establish a value for a metadata field. The text expressions consist of XPath expressions, which are evaluated in a context consisting of the document root and the variable bindings from the LET.

Ad-hoc semantics

The template rules are designed for specific tasks and cannot be easily extended to add further functionality. The templates rules are mentioned in Chapter III, most of the functionality comes with the XPath Core function Library and the Interpreter framework is designed such that, any other extended functionality can be easily plugged in, for example the rules based on “Geometric Properties” and “Document Properties” which are mentioned in Section 3.3.

Limited ability

Limited ability to express or exploit document structure. For example the rules can express “find the line containing AIR COMMAND” but cannot express “find the paragraph containing AIR COMMAND”.

The functions provided by the XPath itself has this ability, for example look at the Fig. 24, the rule “$page1//para[line[(wd='COMMAND' and wd='AIR') or.. ]]]” at line 1 selects the para, which has words either “COMMAND” and “AIR” or “WAR” and “AIR”.
Combining the rules

Cannot combine basic search test functions. For example the rules can not express “title came after a header AND would start with the phrase ‘Fact Sheet’”. It can express only one of them but not both.

Using the new language, we can extract all the data that comes after a header and store it into a variable using <let> element and then using that variable we can check for the line that starts with “Fact Sheet” and bind it into a metadata field using a <bind> variable.

The subsections of this chapter deals with the comparison of the two languages addressing the objectives in Section 1.4.

5.1 POWER OF THE LANGUAGE

The objective of this section is to compare the new template language against the old language, by answering the questions 5.1 and 5.2 mentioned in section 1.4.

5.1.1 DOES THE NEW LANGUAGE COVER ALL THE FUNCTIONALITY OF THE OLD LANGUAGE?

As explained in Section 2.4, The non-form extraction engine uses rule-based template extraction to locate and extract metadata. Each template contains a set of rules designed to extract metadata from a single class of similar documents. Fig. 4 shows a template example. Each desired metadata item is described by a rule set designating the beginning and the end of the metadata. The rules are limited by features detectable at the line level resolution. The first step in constructing a template is to identify a set of documents which share a structural or visual similarity. Once a class is selected, the template author determines the set of rules for each metadata tag by identifying the appropriate function to select the beginning and the end of the tag.

The template language and the ruleset is best described in the template writer’s manual [5].

<template>

As explained in section 2.3.1 of the template writer’s manual [5] In the old language, the <template> element has the metadata field descriptions, and it has the “pagenumber” and “templateID” attributes.
FIG. 31: An example to select a page

"pagenumber" attribute to tell the non form engine to examine only these pages. Whereas in the new language, as shown in the Fig. 11, line 2, we can extract the data of the page we want into a let variable and work on it. If we are not sure of the page where the metadata is present, we can write a rule as shown in Fig. 31. And later we can extract the metadata using the "page".

"templatedID" attribute, as explained in Section 3.3, in the new language also the <template> element has the "templatedID", which helps in identifying the template.

Metadata Rules

There are different optional attributes mentioned in Section 2.3.2 of the template writer’s manual [5]. The implementation of these attributes in the new language is as follows.

"min" and "max" attributes - minimum and maximum number of repetitions of this field that should be expected in the document. There are no such attributes in the new language, the field can be repeated by giving the same name in the <bind> element.

"require" - If it is "yes", failure to extract this field indicates that this template describes a different document layout than is actually present in this document. In the new language "optional" attribute as explained in Section 3.3 is introduced for the same objective.

"ignore" - this field is used merely as a convenience to identify a position within the document. No metadata value will actually be extracted into the output. The same attribute when given in the <bind> element, no metadata will be bound to the field specified.

"filter" and "replace" - these attributes help in extracting the particular portion from the metadata using a regular expression, in the new language the Xpath itself has these string functions.

While writing rules in the old language, we need to follow the order in which the metadata is found i.e., we can extract metadata only one field after the other, where in new language traversing within the document for metadata is easy. In the old language the order of rules has meaning (because the end of one rule is the default start of the search for the next rule). In the same way in the new language the same behavior can be achieved. For example, as shown in Fig. 10, the template written follows the semantics explained
Begin and End selectors

As explained in section 2.3.3 of the template writer's manual [5]. In the old language, each rule will contain one `<begin>` element and one `<end>` element. These describe the beginning and ending line of the text to be extracted for that metadata field. For example, in the Fig. 9, the metadata field “identifier” begins with a line “AU/” and ends in one section. Whereas in the new language, as shown in Fig. 10, avoids usage of the `<begin>` and `<end>` elements. The XPath expression itself provides functionality to select a line or para or a `<let>` within `<let>` can be used to find the beginning and ending of the metadata. The extraction of “UnclassifiedTitle” in Fig. 10 explains the usage of `<let>` within `<let>`.

Line selectors

The various number of rules defined for a specific task are explained in the Section 2.3.4 of the template writer’s manual [5]. This section explains how these functions are done in the new language.

Different rules which help in traversing and selecting specific lines, paras and changing pages like “beforeField”, “begin”, “end”, “pageChange”, “paraEnd” can be implemented easily and effectively by writing the Xpath as shown in the Fig. 24.

The implementation for rules like, “dateformat(formats)”, “dateformat” in the new language is shown using the example given in the Fig. 10. The metadata field “ReportDate” is extracted using the Xpath.

The implementation for the rule “verticalSplit” is mentioned in the Section 4.2.5.

Out of the Document based properties, only the “largeststrsize” rule is implemented in the new language. Instead of using the “Document based properties” using the Xpath functions, the extension functions like “WithinAndMargins” and the third party extension functions mentioned in Section 4.2, We had written a number of templates using the new language. We are able to pass all the template tests and the same metadata is being extracted. A rewrite of many such templates are included in Appendix A.

Apart from having the functionality of the old language, the advantage of the new language is, it is designed to be extensible, so new selector functions can be added as desired.
5.1.2 CAN WE WRITE EFFECTIVE TEMPLATES USING THE NEW LANGUAGE?

Out of our previous experience in writing templates for DTIC, we observed that writing templates for a set of documents called “head-abstract” is difficult. Though looking at the documents that seem to be of same layout, it has got small variations in the documents.

For Example, a general head-abstract document is as shown in the Fig. 32. The recognizing characteristics of this layout are:

1. There might be a topmargin.
2. A title with large size.
3. An abstract which follows a key word “Abstract” and it ends in a key word “Introduction”.
4. A personal author field in between the title and the abstract fields.

A small variant of the above is shown in the Fig. 33. The recognizing characteristics of this layout are:

1. There might be a topmargin.
2. A title whose size is same as the authors (i.e., differentiating title and author based on size is difficult).
3. An abstract which follows a key word “Abstract” and it ends in a key word “Introduction”.
4. A personal author field in between the title and the abstract fields.

We had downloaded around 2000 documents from the DTIC collection. Out of the 2000 documents around 300 documents are manually classified under “head-abstract” class. We ran these documents using the old template language and with the new template language. We also downloaded the OAI Metadata from the DTIC website for the 308 documents. This helps in analyzing our test results, we compared our extractions of titles, authors and abstracts with the OAI metadata and basing on the edit distances, we entered the erroneous ones into an xl sheet. We automated this process also using a java program. And the results are shown below.
On Kalman Filtering With Nonlinear Equality Constraints

Simon J. Julier, Member, IEEE, and Joseph L. LaViola, Jr., Member, IEEE

Abstract—This data spans description of some physical system process nonlinear equality constraints between state space variables. In this paper, we consider the problem of applying a Kalman filter-type estimator to the presence of such constraints. We categorize previous approaches into pseudo-observation and projection methods and identify two types of constraints—those that act on the entire distribution and those that act on the means of the distribution. We argue that the pseudo-observation approach identifies better types of constraint and that the projection method identifies the first type of constraint only. We propose a new method that utilizes the projection method twice—once to constrain the entire distribution and once to constrain the statistics of the distribution. We illustrate these algorithms in a tracking system that uses both quadrature to correlate orientation.

Index Terms—Kalman filtering, quadrature, measurement matrix, nonlinear constraints.

I. INTRODUCTION

Some physical systems have equality constraints between their state variables. These constraints arise for several reasons. Some constraints arise from the basic laws of physics. The mass of coordinates in a sealed chemical reactor, for example, never remains constant throughout the reaction process [1]. Other constraints arise from the mathematical description of a state vector. If the mass is modeled as a constant entity, then the mass of the mixture must be conserved [2]. Constraints can arise from kinematic or geometric considerations of a system. The coordinated turn model for an aircraft, for example, assumes that the acceleration vector is orthogonal to the velocity vector [3], [4].

A number of approaches have been developed to apply these equality constraints within the Kalman filter (KF) framework. These can be broadly classified into pseudo-observation, projection, and extrapolation methods. The pseudo-observation method creates a fictitious observation whose variance is zero [5]-[9]. By substitution, it can be shown that the EKF update equation projects the state estimate onto the constraint surface [7]. The projection approach constructs a projection operator that transforms the estimate so that it lies on the constraint surface [5], [10]-[12]. The extrapolation method simply extrapolates the system so that the equality constraint is not required [12]. Since the focus of this work is on examining the effects of different algorithms for constrained estimation, the extrapolation method does not fall within the scope of this paper: Thus, we focus on the first two methods.

In this paper, we argue that although the pseudo-observation and projection approaches have clear, simple, intuitive interpretations when the constraints are linear, none of these properties hold when the constraints are nonlinear. The main reason is that both approaches implicitly assume that the probability distribution obeys the nonlinear constraint, whereas the means of the distribution (which is the property maintained in the filter) obeys the constraint as well. The pseudo-observation method makes the further assumption that the Kalman filter update rule, which is a linear projection operator, is sufficient to constrain a probability distribution to a nonlinear surface.

To overcome these difficulties, we propose a two-step constrained application method. The first step applies the projection method to the unconstrained estimate. As a result, the probability distribution of the estimate is constrained to lie along the constraint surface. This causes the covariance to the estimate to decrease. In the second step, the distribution is translated so that its mean lies on the constraint surface. This, in turn, causes the covariance in the estimate to increase.

The structure of this paper is as follows. The problem statement is described in Section II. Section III introduces an illustrative example which is used throughout much of this paper: estimating an angle of rotation about a single axis using complex numbers. We analyze the pseudo-observation and projection methods and show that they fail. Section IV extends the properties of the constraints in detail and identifies two types of constraints. The first type is a constraint on each sample in the entire distribution. The second type is a constraint on just the mean of the estimate. We show that neither the pseudo-observation nor the projection approaches fully satisfy either constraint interpretation. Section V derives the new two-step approach for applying constraints. In Section VII, the new algorithm is illustrated in a head-tracking application that uses unit quaternion to represent orientation. Conclusions are drawn in Section VII.

1) The research for this project was partly supported by the Office of Naval Research, the Department of Energy, and Microsoft. The authors wish to acknowledge the advice of Dr. P. Van Valkenborgh for providing the ID-IDT code.
2) J. Julier was with the 3D Visual and Motion Estimation Laboratory, Naval Research Laboratory, Washington, DC 20375 USA. He is now with the Computer Science Department, University College London, London, U.K. (e-mail: julier@acm.org).
3) J. L. LaViola, Jr. is with the School of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816 USA (e-mail: jla@ece.ucf.edu).
4) Additional versions of this document are available online at http://www.ics.uci.edu.
Digital Object Identifier 10.1109/TSP.2007.909949

FIG. 32: Sample document of the head-abstract class
GLOBAL MECHANICAL RESPONSE AND ITS RELATION TO DEFORMATION AND FAILURE MODES AT VARIOUS LENGTH SCALES UNDER SHOCK IMPACT IN ALEUMINA AD995 ARMOR CERAMIC

D. P. Danielson*, J. W. McCauley, and W. H. Green
Army Research Laboratory, AFRL, MD 21006

N. K. Bourne
University of Manchester, UK

M. W. Chen
Johns Hopkins University, Baltimore MD and Tohoku University, Sendai, Japan

ABSTRACT

Polycrystalline aluminum oxide (Al₂O₃ - alumina), known as sapphire in single crystal form, has been used for many years in both personal and ground vehicle armor applications. Yet the ballistic performance of alumina based armor ceramic materials can vary significantly, a major challenge for systematic material optimization, validation, simulation, prediction, and design of armor systems. We seek to address this problem through identification of the fundamental mechanisms of deformation and failure of alumina in the ballistic event.

Past efforts have used instrumented laboratory high strain rate high pressure mechanical tests including Split Hopkinson Pressure Bar (SHPB: Ehlers Bar) and plate impact (shock waves) to mimic, in a controlled environment, the response of armor materials under ballistic high strain rate - high stress conditions. But there are two major problems associated with this approach: (A) recovery of samples from the test specimen is difficult to characterize, and (B) identification of the deformation and failure mechanisms that cut across length scales from centimeters to nanometers. This has been a major challenge over the years.

Using a newly developed shock recovery technique (Bourne et al. 2006), an international team drawn from the Army Research Laboratory (ARL), Royal Military College of Science (RMCS), Shrivenham, UK, the University of Manchester, Manchester, UK, Johns Hopkins University and Tohoku University, Sendai, Japan, focused on AD995 (CorExTop, Co.), a very well known commercial high purity alumina. This collaboration has, for the first time, demonstrated micro and meso deformation and damage mechanisms maps that relate the experimentally measured global mechanical response of a material through measured shock wave diagnostics to the nature of concurrent deformation and damage generated at varying length scales under shock wave loading.

1. INTRODUCTION

Current operations in Iraq and Afghanistan increasingly demonstrate the need for threat specific, reduced weight, transparent and opaque armor in many ground systems, including personnel protection. At the threats have evolved and because more varied, the challenges for rapidly developing optimized threat specific armor packages have grown complex. Certain high performance structural ceramics, Al₂O₃, B₄C, SiC, TiB₂, AlN, AlCN, graphite, glass, etc. have proven to be effective armor materials at much lower weights in many systems. A critical key to further accelerating optimization of these materials is development of validated predictive performance computer models. This approach is based on the demonstration and quantification of the various ballistic energy absorption mechanisms, including the various deformation modes, damage mechanization and accumulation processes, and the resulting eventual failure of armor ceramics at high rates under very high impact stress (shock wave), computable

FIG. 33: A Sample variant of the head-abstract class
Table 2 shows the results obtained when the documents were run against the old Language non form engine. Out of 308 documents, the templates could extract metadata for 273 documents and there are no templates written for the remaining 35 documents. Using the values in the table the performance of the template language can be measured. The columns in the table can be understood as follows:

1. Various templates that are successful in extracting metadata from the documents are listed in the “template” column.

2. “count” column gives the number of documents that are extracted using that template.

3. As explained in section 2.6, the metadata extracted has a confidence value. The “zero count” column gives the number of documents for which the template scored “0.0” confidence value.

4. The “mean” column gives the mean of all confidence values. For example, if we take the “head-abstract” class, there are 22 documents that are extracted using this
TABLE 3: The results table showing the performance of the new language template. The mean of all the 22 documents confidence values is calculated to be “0.87”.

5. Similarly, the column “stddev” gives the standard deviation of the confidence values.
6. The column “min” gives the minimum confidence value of all extractions made by that template.
7. The column “max” gives the maximum confidence value of all extractions made by that template.

Table 3 shows the results obtained when the documents were run against the old Language non form engine. Out of 308 documents, the templates could extract metadata for 271 documents and there are no templates written for the remaining 37 documents.

Table 5 shows the results of erroneous titles obtained when the documents were run against the old Language non form engine. There are 58 erroneous extractions.

Using a java program, for each document we tabulated UnclassifiedTitle extracted and the UnclassifiedTitle from the OAI Metadata of the document. In Table 6 the “OAI Metadata” column contains the title taken from the OAI Metadata and the “Extracted” column contains the “UnclassifiedTitle” extracted. The column “score” contains the edit distance between the two titles.

Table 7 shows the results of erroneous titles obtained when the documents were run against the new Language non form engine. There are 73 erroneous extractions.

Table 9 shows the results of erroneous authors obtained when the documents were run against the old Language non form engine. There are 142 erroneous extractions.

Table 11 shows the results of erroneous authors obtained when the documents were run against the new Language non form engine. There are 138 erroneous extractions.

Table 12 shows the results of erroneous abstracts obtained when the documents were run against the old Language non form engine. There are 213 erroneous extractions.
TABLE 4: The table showing the sample of erroneous titles using old language

<table>
<thead>
<tr>
<th>File Name</th>
<th>Class</th>
<th>OAI Metadata</th>
<th>Extracted</th>
<th>Score</th>
<th>Warning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADA497386</td>
<td>head-abstract</td>
<td>a technical approach</td>
<td>technical</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA494622</td>
<td>head-abstract-1b</td>
<td>iterated class-iterated class</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA498911</td>
<td>head-abstract-1b</td>
<td>multi-tof/mass multi-tof</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA494596</td>
<td>head-abstract-choose</td>
<td>a multi-resolution multi-resolution</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA497097</td>
<td>head-abstract-pttiMeeting</td>
<td>time transfer using time transfer</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA497146</td>
<td>head-abstract-typewritten</td>
<td>characterization of characterization</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA497248</td>
<td>head-abstract-typewritten</td>
<td>relationship between relationship</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA480144</td>
<td>head-abstract-1b</td>
<td>determination of determination</td>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>ADA493208</td>
<td>head-abstract</td>
<td>linear txt analysis 10.2: linear text analysis</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>ADA491307</td>
<td>head-abstract-AFRL</td>
<td>a guided-ion beam a guided-ion beam</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>ADA496155</td>
<td>head-abstract-pttiMeeting</td>
<td>new steering new steering</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>ADA497300</td>
<td>head-abstract-pttiMeeting</td>
<td>analysis noise, analysis noise</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>ADA497322</td>
<td>head-abstract-pttiMeeting</td>
<td>use of ionsphere use of ionsphere</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>ADA497947</td>
<td>head-abstract-typewritten</td>
<td>various uses of various uses</td>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>ADA495995</td>
<td>head-abstract</td>
<td>wavelet-transform wavelet transform</td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>ADA496548</td>
<td>head-Noabstract</td>
<td>doe's industrial doe's industrial</td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>ADA497737</td>
<td>head-abstract-choose</td>
<td>exact that a new exact that a new</td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>ADA496464</td>
<td>head-abstract-choose</td>
<td>monitoringResearch monitoringResearch</td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>ADA497302</td>
<td>head-abstract-pttiMeeting</td>
<td>a high performance a high performance</td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>ADA497753</td>
<td>head-abstract-typewritten</td>
<td>narrow 87b and 1:narrow 87a ar</td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>ADA497491</td>
<td>head-abstract-choose</td>
<td>hybrid verification hybrid verification</td>
<td></td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>ADA497392</td>
<td>head-abstract-pttiMeeting</td>
<td>design of a satellite design of a satellite</td>
<td></td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>ADA494478</td>
<td>head-abstract-typewritten</td>
<td>no warmup crystal no warmup crystal</td>
<td></td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>ADA502467</td>
<td>head-Noabstract</td>
<td>film behavior in gas film behavior in gas</td>
<td></td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>ADA498842</td>
<td>head-abstract-choose</td>
<td>12.75 in. synthetic 12.75&quot; synthetic</td>
<td></td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>ADA494711</td>
<td>head-abstract-1b</td>
<td>how bad receiver how bad receiver</td>
<td></td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>ADA495364</td>
<td>head-abstract-choose</td>
<td>fast scanning and fast scanning</td>
<td></td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>ADA496034</td>
<td>annual Report</td>
<td>advanced under advanced under</td>
<td></td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>ADA481139</td>
<td>head-abstract-physicsJournal</td>
<td>thermal conductivity thermal conductivity</td>
<td></td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>ADA509668</td>
<td>head-abstract</td>
<td>two-way satellite two-way satellite</td>
<td></td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>ADA498863</td>
<td>head-abstract-choose</td>
<td>the mosquito aedes blackwell pub the mosquito aedes blackwell pub</td>
<td></td>
<td>27</td>
<td></td>
</tr>
</tbody>
</table>

TABLE 5: The table showing the performance of the titles using old language

<table>
<thead>
<tr>
<th>Class</th>
<th>No of Errors</th>
<th>Total No of Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-abstract</td>
<td>1</td>
<td>58</td>
</tr>
<tr>
<td>head-abstract-1b</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>head-abstract-1b</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>head-abstract-AFRL</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>head-abstract-choose</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>head-abstract-choose</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>head-abstract-typewritten</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>head-abstract-typewritten</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>head-Noabstract</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>
TABLE 6: The table showing the sample of erroneous titles using new language

<table>
<thead>
<tr>
<th>File Name</th>
<th>Class</th>
<th>OAI Metadata</th>
<th>Extracted</th>
<th>Score</th>
<th>Warning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADA497646</td>
<td>head-abstract</td>
<td>the role of the dee</td>
<td>the role of the d</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA497648</td>
<td>head-abstract</td>
<td>composite-type</td>
<td>87</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>ADA497947</td>
<td>head-abstract</td>
<td>various uses of the</td>
<td>various uses</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>ADA489529</td>
<td>head-abstract</td>
<td>standardization of standardization</td>
<td>11</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>ADA494478</td>
<td>head-abstract</td>
<td>no warmup crystal</td>
<td>no warmup cry</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>ADA500796</td>
<td>head-abstract</td>
<td>flow simulations: t flow simulations</td>
<td>17</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>ADA495364</td>
<td>head-abstract</td>
<td>fast scanning and fast scanning a</td>
<td>19</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td>ADA500553</td>
<td>head-abstract</td>
<td>global positioning</td>
<td>global position</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td>ADA497601</td>
<td>head-abstract</td>
<td>unprecedented syr</td>
<td>n p e c e n e</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>ADA497477</td>
<td>head-abstract</td>
<td>time code dissemit</td>
<td>time code dts</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>ADA500668</td>
<td>head-abstract</td>
<td>two-way satellite t</td>
<td>90th annual pre</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>ADA495903</td>
<td>head-abstract</td>
<td>the range covered</td>
<td>90th annual pr</td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>ADA499561</td>
<td>head-abstract</td>
<td>an overview of net</td>
<td>an overview of n</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>ADA497674</td>
<td>head-abstract</td>
<td>design of the st</td>
<td>design of the st</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td>ADA496264</td>
<td>head-abstract</td>
<td>high yield magneti</td>
<td>high yield magr</td>
<td>59</td>
<td></td>
</tr>
<tr>
<td>ADA499901</td>
<td>head-abstract</td>
<td>performance of the 29th annual p</td>
<td>61</td>
<td>61</td>
<td></td>
</tr>
<tr>
<td>ADA487586</td>
<td>head-abstract</td>
<td>synthetic seismog</td>
<td>2008 monitoring</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td>ADA487636</td>
<td>head-abstract</td>
<td>explosion source</td>
<td>c2008 monitoring</td>
<td>67</td>
<td></td>
</tr>
<tr>
<td>ADA487650</td>
<td>head-abstract</td>
<td>estimating local ar</td>
<td>2008 monitoring</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>ADA488872</td>
<td>head-abstract</td>
<td>working the nights dr.</td>
<td>nita lewis m</td>
<td>79</td>
<td></td>
</tr>
<tr>
<td>ADA494815</td>
<td>head-abstract</td>
<td>nonlinear optics</td>
<td>ar final report to th</td>
<td>96</td>
<td></td>
</tr>
<tr>
<td>ADA497248</td>
<td>head-abstract</td>
<td>relationship bet</td>
<td>relationship bet</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>ADA494375</td>
<td>head-abstract</td>
<td>discussion of cloc</td>
<td>7</td>
<td>101</td>
<td></td>
</tr>
<tr>
<td>ADA497753</td>
<td>head-abstract</td>
<td>narrow 87rb and</td>
<td>1: narrow &quot; 8b and</td>
<td>105</td>
<td></td>
</tr>
<tr>
<td>ADA497146</td>
<td>head-abstract</td>
<td>characterization of c</td>
<td>characteriza</td>
<td>110</td>
<td></td>
</tr>
<tr>
<td>ADA497641</td>
<td>head-abstract</td>
<td>the fundamental st</td>
<td>the fundamenta</td>
<td>114</td>
<td></td>
</tr>
<tr>
<td>ADA494906</td>
<td>head-abstract</td>
<td>discovery of highly aflr-v-ha-tr-200f</td>
<td>168</td>
<td>168</td>
<td></td>
</tr>
<tr>
<td>ADA494956</td>
<td>head-abstract-1</td>
<td>a multi-resolution</td>
<td>a multi - resolut</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA494622</td>
<td>head-abstract-1</td>
<td>iterated class-spe</td>
<td>iterated class -</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA498842</td>
<td>head-abstract-1</td>
<td>12.75 in. synthetic 12.75&quot; synthetic</td>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA498911</td>
<td>head-abstract-1</td>
<td>maldi-tof mass</td>
<td>maldi-tof/tof ma</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>ADA481327</td>
<td>head-abstract-1</td>
<td>numeric simulation</td>
<td>numeric simul</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>ADA493208</td>
<td>head-abstract-1</td>
<td>linear twt analysis</td>
<td>10.2: linear twt</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

TABLE 7: The table showing the performance of the titles using new language

<table>
<thead>
<tr>
<th>Class</th>
<th>No of Errors</th>
<th>Total No of Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-abstract</td>
<td>28</td>
<td>73</td>
</tr>
<tr>
<td>head-abstract-1</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>head-abstract-topMargin</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>head-noAbstract-Introduction-TopMargin</td>
<td>18</td>
<td></td>
</tr>
</tbody>
</table>
TABLE 8: The table showing the sample of erroneous authors using old language

<table>
<thead>
<tr>
<th>File Name</th>
<th>Class</th>
<th>OAI Metadata</th>
<th>Extracted</th>
<th>Score</th>
<th>Warning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADA49034</td>
<td>annualReport</td>
<td>harbor branch</td>
<td></td>
<td></td>
<td>117 No found</td>
</tr>
<tr>
<td>ADA495402</td>
<td>head-abstract</td>
<td>schamiloglu, edl</td>
<td></td>
<td></td>
<td>1 Partial match</td>
</tr>
<tr>
<td>ADA498490</td>
<td>head-abstract</td>
<td>gendron, martin</td>
<td></td>
<td></td>
<td>1 Partial match</td>
</tr>
<tr>
<td>ADA500688</td>
<td>head-abstract</td>
<td>shibuya, yasu</td>
<td></td>
<td></td>
<td>1 Partial match</td>
</tr>
<tr>
<td>ADA500688</td>
<td>head-abstract</td>
<td>nakadan, kazuhiro</td>
<td></td>
<td></td>
<td>1 Partial match</td>
</tr>
<tr>
<td>ADA499490</td>
<td>head-abstract</td>
<td>martin, charles</td>
<td></td>
<td></td>
<td>2 Partial match</td>
</tr>
<tr>
<td>ADA497338</td>
<td>head-abstract</td>
<td>paulsen, r j</td>
<td></td>
<td></td>
<td>5 Partial match</td>
</tr>
<tr>
<td>ADA492178</td>
<td>head-abstract</td>
<td>power, range</td>
<td></td>
<td></td>
<td>9 No found</td>
</tr>
<tr>
<td>ADA490134</td>
<td>head-abstract</td>
<td>anastasia, j i</td>
<td></td>
<td></td>
<td>12 No found</td>
</tr>
<tr>
<td>ADA492178</td>
<td>head-abstract</td>
<td>cashin, william</td>
<td></td>
<td></td>
<td>2 Partial match</td>
</tr>
<tr>
<td>ADA499947</td>
<td>head-abstract</td>
<td>soref, richard</td>
<td></td>
<td></td>
<td>3 Partial match</td>
</tr>
<tr>
<td>ADA492201</td>
<td>head-abstract</td>
<td>renz, t</td>
<td></td>
<td></td>
<td>1 Partial match</td>
</tr>
<tr>
<td>ADA499490</td>
<td>head-abstract</td>
<td>tamanaha, cy</td>
<td></td>
<td></td>
<td>2 Partial match</td>
</tr>
<tr>
<td>ADA493848</td>
<td>head-abstract</td>
<td>weidemann, a</td>
<td></td>
<td></td>
<td>2 Partial match</td>
</tr>
<tr>
<td>ADA493848</td>
<td>head-abstract</td>
<td>amone, robert</td>
<td></td>
<td></td>
<td>2 Partial match</td>
</tr>
<tr>
<td>ADA496613</td>
<td>head-abstract</td>
<td>young, l e</td>
<td></td>
<td></td>
<td>2 Partial match</td>
</tr>
<tr>
<td>ADA497743</td>
<td>head-abstract</td>
<td>spence, bob</td>
<td></td>
<td></td>
<td>2 Partial match</td>
</tr>
<tr>
<td>ADA498118</td>
<td>head-abstract</td>
<td>goff, j a</td>
<td></td>
<td></td>
<td>3 Partial match</td>
</tr>
<tr>
<td>ADA498118</td>
<td>head-abstract</td>
<td>breakiron, lee</td>
<td></td>
<td></td>
<td>4 Partial match</td>
</tr>
<tr>
<td>ADA498118</td>
<td>head-abstract</td>
<td>choi, jee w</td>
<td></td>
<td></td>
<td>4 Partial match</td>
</tr>
<tr>
<td>ADA493849</td>
<td>head-abstract</td>
<td>gray, j</td>
<td></td>
<td></td>
<td>5 No found</td>
</tr>
<tr>
<td>ADA497827</td>
<td>head-abstract</td>
<td>willcox, by r</td>
<td></td>
<td></td>
<td>6 No found</td>
</tr>
<tr>
<td>ADA497746</td>
<td>head-abstract</td>
<td>martin, liz</td>
<td></td>
<td></td>
<td>6 No found</td>
</tr>
<tr>
<td>ADA493144</td>
<td>head-abstract</td>
<td>rongfong, li</td>
<td></td>
<td></td>
<td>8 No found</td>
</tr>
<tr>
<td>ADA495033</td>
<td>head-abstract</td>
<td>ave, grace h</td>
<td></td>
<td></td>
<td>8 No found</td>
</tr>
<tr>
<td>ADA495063</td>
<td>head-abstract</td>
<td>sp, j</td>
<td></td>
<td></td>
<td>8 No found</td>
</tr>
<tr>
<td>ADA495063</td>
<td>head-abstract</td>
<td>ocean, naval</td>
<td></td>
<td></td>
<td>9 No found</td>
</tr>
<tr>
<td>ADA495063</td>
<td>head-abstract</td>
<td>entomol, j m</td>
<td></td>
<td></td>
<td>9 No found</td>
</tr>
<tr>
<td>ADA495063</td>
<td>head-abstract</td>
<td>chenwu, lan</td>
<td></td>
<td></td>
<td>10 No found</td>
</tr>
<tr>
<td>ADA495063</td>
<td>head-abstract</td>
<td>entomol, j m</td>
<td></td>
<td></td>
<td>10 No found</td>
</tr>
<tr>
<td>ADA495063</td>
<td>head-abstract</td>
<td>daniel, i m</td>
<td></td>
<td></td>
<td>11 No found</td>
</tr>
<tr>
<td>ADA495063</td>
<td>head-abstract</td>
<td>rajapakse, y c</td>
<td></td>
<td></td>
<td>12 No found</td>
</tr>
</tbody>
</table>

TABLE 9: The table showing the performance of the authors using old language

<table>
<thead>
<tr>
<th>Class</th>
<th>No of Errors</th>
<th>Total No of Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-abstract</td>
<td>10</td>
<td>142</td>
</tr>
<tr>
<td>head-abstract-1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>head-abstract-1b</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>head-abstract-AFRL</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>head-abstract-choose</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>head-abstract-monitoringResearch</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>head-abstract-physicsJournal</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>head-abstract-ptlMeeting</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>head-abstract-typewritten</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>head-Noabstract</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>head-Noabstract-Pg1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>head-Noabstract-received</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
### TABLE 10: The table showing the sample of erroneous authors using new language

<table>
<thead>
<tr>
<th>File Name</th>
<th>Class</th>
<th>OAI Metadata</th>
<th>Extracted</th>
<th>Score</th>
<th>Warning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADA487636</td>
<td>head-abstract</td>
<td>bonner, jessie</td>
<td>bonner', jessie</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA487636</td>
<td>head-abstract</td>
<td>ledig, mark r</td>
<td>ledig', mark r</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA487636</td>
<td>head-abstract</td>
<td>doughearty, sara I</td>
<td>doughearty', sara I</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA487650</td>
<td>head-abstract</td>
<td>gerstoft, peter</td>
<td>gerstoft', peter</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA487666</td>
<td>head-abstract</td>
<td>zhang, jian</td>
<td>zhang', jian</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA487680</td>
<td>head-abstract</td>
<td>kuperman, william a</td>
<td>kuperman', william a</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA487680</td>
<td>head-abstract</td>
<td>bonner, jessie</td>
<td>bonner', jessie</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA489529</td>
<td>head-abstract</td>
<td>hargus, jr, william a</td>
<td>hargus jr, william a</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA495903</td>
<td>head-abstract</td>
<td>meo, daniela</td>
<td>meo, daniela</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA497666</td>
<td>head-abstract</td>
<td>hammersfahr, j</td>
<td>hammersfahr', j</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA500668</td>
<td>head-abstract</td>
<td>nakadan, kazuhiro</td>
<td>nakadan, kazuhiro</td>
<td>1</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA483848</td>
<td>head-abstract</td>
<td>weidemann, alan</td>
<td>weidemann, alan d</td>
<td>2</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA483848</td>
<td>head-abstract</td>
<td>amone, robert</td>
<td>amone, robert a</td>
<td>2</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA495215</td>
<td>head-abstract</td>
<td>cashin, william f</td>
<td>cashin, william f</td>
<td>2</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA497477</td>
<td>head-abstract</td>
<td>detonia, e</td>
<td>detonia, e</td>
<td>2</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA494733</td>
<td>head-abstract</td>
<td>breakiron, lee a</td>
<td>breakiron, a</td>
<td>4</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA483848</td>
<td>head-abstract</td>
<td>gray, j</td>
<td>gray</td>
<td>5</td>
<td>No found</td>
</tr>
<tr>
<td>ADA494691</td>
<td>head-abstract</td>
<td>kusters, a</td>
<td>kusters, a</td>
<td>5</td>
<td>No found</td>
</tr>
<tr>
<td>ADA494840</td>
<td>head-abstract</td>
<td>de marchi, andrea</td>
<td>de marchi, andrea d</td>
<td>5</td>
<td>Partial match</td>
</tr>
<tr>
<td>ADA497601</td>
<td>head-abstract</td>
<td>davis, d</td>
<td>davis, d</td>
<td>6</td>
<td>No found</td>
</tr>
<tr>
<td>ADA494691</td>
<td>head-abstract</td>
<td>giffard, p</td>
<td>giffard, p</td>
<td>6</td>
<td>No found</td>
</tr>
<tr>
<td>ADA497146</td>
<td>head-abstract</td>
<td>mmcarecm, j e</td>
<td>mmcarecm, j e</td>
<td>6</td>
<td>No found</td>
</tr>
<tr>
<td>ADA497648</td>
<td>head-abstract</td>
<td>na, shigeya</td>
<td>na, shigeya</td>
<td>7</td>
<td>No found</td>
</tr>
<tr>
<td>ADA487650</td>
<td>head-abstract</td>
<td>diego', aan</td>
<td>diego', aan</td>
<td>8</td>
<td>No found</td>
</tr>
<tr>
<td>ADA487788</td>
<td>head-abstract</td>
<td>naiou, r b</td>
<td>naiou, r b</td>
<td>8</td>
<td>No found</td>
</tr>
<tr>
<td>ADA487788</td>
<td>head-abstract</td>
<td>mikhail, r r</td>
<td>mikhail, r r</td>
<td>8</td>
<td>No found</td>
</tr>
<tr>
<td>ADA487786</td>
<td>head-abstract</td>
<td>barrett, wei z</td>
<td>barrett, wei z</td>
<td>9</td>
<td>No found</td>
</tr>
<tr>
<td>ADA494691</td>
<td>head-abstract</td>
<td>al, w</td>
<td>al, w</td>
<td>9</td>
<td>No found</td>
</tr>
<tr>
<td>ADA495903</td>
<td>head-abstract</td>
<td>clock, by a</td>
<td>clock, by a</td>
<td>10</td>
<td>No found</td>
</tr>
<tr>
<td>ADA494375</td>
<td>head-abstract</td>
<td>single, with a</td>
<td>single</td>
<td>11</td>
<td>No found</td>
</tr>
<tr>
<td>ADA494691</td>
<td>head-abstract</td>
<td>time, allan's</td>
<td>time, allan's</td>
<td>12</td>
<td>No found</td>
</tr>
<tr>
<td>ADA494815</td>
<td>head-abstract</td>
<td>hiller, robert j</td>
<td>hiller, robert j</td>
<td>12</td>
<td>No found</td>
</tr>
</tbody>
</table>

### TABLE 11: The table showing the performance of the authors using new language

<table>
<thead>
<tr>
<th>Class</th>
<th>No of Errors</th>
<th>Total Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-abstract</td>
<td>49</td>
<td>139</td>
</tr>
<tr>
<td>head-abstract-1</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>head-abstract-topMargin</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>head-noAbstract-Introduction-TopMargin</td>
<td>27</td>
<td></td>
</tr>
</tbody>
</table>
TABLE 12: The table showing the performance of the abstracts using old language

<table>
<thead>
<tr>
<th>Class</th>
<th>No of Errors</th>
<th>Total No of Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-abstract</td>
<td>21</td>
<td>213</td>
</tr>
<tr>
<td>head-abstract-management-Science</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>head-abstract-1b</td>
<td>47</td>
<td>47</td>
</tr>
<tr>
<td>head-abstract-AFRL</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>head-abstract-choose</td>
<td>79</td>
<td>79</td>
</tr>
<tr>
<td>head-abstract-monitoring-Research</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>head-abstract-physics-Journal</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>head-abstract-ptti-Meeting</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>head-abstract-typewritten</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>head-Noabstract</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>head-abstract-AFOSR_pg1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>head-Noabstract-received</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>head-abstract-subject</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>head-abstract-objective</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

TABLE 13: The table showing the performance of the abstracts using new language

<table>
<thead>
<tr>
<th>Class</th>
<th>No of Errors</th>
<th>Total No of Errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-abstract</td>
<td>36</td>
<td>153</td>
</tr>
<tr>
<td>head-abstract-1</td>
<td>69</td>
<td>69</td>
</tr>
<tr>
<td>head-abstract-topMargin</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>head-noAbstract-Introduction-TopMargin</td>
<td>28</td>
<td>28</td>
</tr>
</tbody>
</table>
Table 13 shows the results of erroneous abstracts obtained when the documents were run against the new Language non form engine. There are 153 erroneous extractions.

Summarizing the results from the tables and figures-

1. Using fewer templates with the new template language, we got the same percentage of extraction when compared with the old language.

2. Though percentage of erroneous titles is little more using the new template language, the percentage of erroneous authors is little less.

3. Using the old templates languages out of 308 documents, there are 213 erroneous extractions. When compared with the new language it is higher. Using the new template language out of 308 documents, there are 153 erroneous extractions.

5.2 EFFECTIVENESS OF THE LANGUAGE

5.2.1 UNDERSTANDING THE NEW LANGUAGE

This section speaks about understanding the new language i.e., for a beginner to template writing, can he understand the new language? For this purpose we took a survey. We organized a quiz to a group of people. The group consisting of students who worked for the extract project[18], who have knowledge of the old language and are able to write templates in it and a student who does not have any idea of the old language but has some knowledge of xml.

We explained them the new language’s features i.e., the xpath functions and the extension functions and using a sample template from Appendix A showed them the process of metadata extraction. Then we gave them a quiz consisting of 9 questions. The questions consisted of some code snippets taken from the templates given in Appendix A. By looking at the corresponding documents in Appendix A, the students had to tell what metadata would be extracted by the code snippets.

The results are found satisfying:

- All students were able to answer the questions correctly.
- They found the language exciting and easy to understand.
- Table 14 shows the results of the quiz.
- The average percent of accuracy is 93.3%.
TABLE 14: The table showing the results of the quiz

<table>
<thead>
<tr>
<th>Student</th>
<th>Quiz Score</th>
<th>Mean</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student 1</td>
<td>9</td>
<td>8.4</td>
<td>0.894427191</td>
</tr>
<tr>
<td>Student 2</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Student 3</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Student 4</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Student 5</td>
<td>7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- The standard deviation is 0.89.
- The quiz starts with straightforward questions and ends in trickier questions (head-abstract templates are little complex). The experienced group was able to get all the answers right except one student, while the student in inexperienced group made few mistakes.

We took the feedback from the students also, a few of them are mentioned below.

1. Student 1 has mentioned in his feedback that he liked the closed property of the language i.e., each “let” returns a block within a block like SQL returns a table from table. The margin function will come in very handy.

2. Student 2 has mentioned in his feedback that the language is easily understandable and when it comes to writing of templates, initially it would be hard to write templates. But once we write 2-3 templates it becomes easy to write.

3. Student 3 has mentioned in his feedback that the language is easy to understand on the first go itself, but to write a template we need to have a little bit of experience of understanding 4-5 different templates. The easy or the part which he liked was the “margins” function and it was the one he found out to be effective to grab the specific region which we want.

4. Student 4 has mentioned in his feedback that he was able to understand the language. He found the new language that it reduces the work of a template writer. With the previous language, when a small variation of the document occurs, we used to create a new template. But now this language increases the scope of the template writer by using the various functions effectively. Previously, it was like a top-down parsing of the document, there was no flexibility to zero-in onto a particular word of the document, but now we can achieve that.
CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 CONCLUSIONS

1. We introduced a new language for extracting metadata from non form documents.

2. We developed templates for the documents using the new language and tested them using the new non form processing engine.

3. We compared the new language against the old language, by answering the questions mentioned in section 1.4.

4. We are able to get all the functionality of the old language.

5. As mentioned in section 5.1.2, we achieved the same kind of accuracy for a hard set of documents with less number of templates.

6. We did a survey for finding out whether the new language is understandable to everyone and the results are encouraging.

6.2 FUTURE WORK

The question "Using the new template language is it easy to write templates?" mentioned in section 1.4 is left as future work. To answer this question, an automated intelligent assistant program for writing templates can be developed. Using that program, a survey can be conducted to write new templates and feedback can be taken.

Section 4.2.5 and 4.2.6 gives a description of how to use extension functions using SAXON XPath Processor. Similarly, in future user defined extension functions can be developed to extract metadata which will further simplify the template and help to examine a wider variety of document layouts.
BIBLIOGRAPHY


APPENDIX A

SAMPLE DOCUMENTS AND TEMPLATES

In this Chapter we can find some templates which are written in old language and a rewrite of it in new language.

A.1 AFIT

A.1.1 AFIT DOCUMENT

Afit Document can be found in Fig. 34.

A.1.2 AFIT TEMPLATE IN OLD LANGUAGE

Afit template in old language can be found in Fig. 35.

A.1.3 AFIT TEMPLATE IN NEW LANGUAGE

Afit template in new language can be found in Fig. 36.

A.2 ACGSC

A.2.1 ACGSC DOCUMENT

Acgsc Document can be found in Fig. 37.

A.2.2 ACGSC TEMPLATE IN OLD LANGUAGE

Acgsc template in old language can be found in Fig. 38.

A.2.3 ACGSC TEMPLATE IN NEW LANGUAGE

Acgsc template in new language can be found in Fig. 39.

A.3 AFRL4

A.3.1 AFRL4 DOCUMENT

Afrl4 Document can be found in Fig. 40.
Analyzing Carbohydrate-Based Regenerative Fuel Cells as a Power Source for Unmanned Aerial Vehicles

THESIS

Olick Wagner, Captain, USAF
AFIT/GAE/ENY/08-M31

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY
Wright-Patterson Air Force Base, Ohio

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

FIG. 34: Afit document
FIG. 35: Afit template written in old language

FIG. 36: Afit template written in new language
Decentralizing Centralized Control
Reorienting a Fundamental Tenet for Resilient Air Operations

A Monograph
by
Major Mark E. Blomme
United States Air Force

School of Advanced Military Studies
United States Army Command and General Staff College
Fort Leavenworth, Kansas

AY 2008

Approved for Public Release; Distribution is Unlimited

FIG. 37: Acgsc document
FIG. 38: Acgsc template written in old language
A.3.2 AFRL4 TEMPLATE IN OLD LANGUAGE

Afrl4 tempalte in old language can be found in Fig. 41.

A.3.3 AFRL4 TEMPLATE IN NEW LANGUAGE

Afrl4 tempalte in new language can be found in Fig. 42.

A.4 USACERL

A.4.1 USACERL DOCUMENT

Usacerl Document can be found in Fig. 43.

A.4.2 USACERL TEMPLATE IN OLD LANGUAGE

Usacerl tempalte in old language can be found in Fig. 44.

A.4.3 USACERL TEMPLATE IN NEW LANGUAGE

Usacerl tempalte in new language can be found in Fig. 45.
NEEDLE IN THE HAYSTACK SECURE COMMUNICATION

University of Central Florida

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

STUNED COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

FIG. 40: Afrl4 document
FIG. 41: AFRL template written in old language

FIG. 42: AFRL template written in new language
Severe Weather Impact Analysis for Military Construction Projects

by
Stephen L. Sheen
Diego Echeverry
Mohammed Aboushousha
Simon S. Kim

Some construction delays caused by severe weather are inevitable, and evaluating the weather severity and resultant impact to construction activities can be difficult. The U.S. Army Corps of Engineers (USACE) attempts to give contractors a reasonably accurate estimate of anticipated severe weather impact before construction begins, to avoid normal delays into construction schedules and contract durations. This report develops a concept for an improved method to estimate and evaluate the impact of severe weather on Corps military construction projects.

This improved approach will provide (1) accurate preconstruction estimates of anticipated severe weather, and (2) a fair and reasonable way to evaluate delays during construction due to severe weather.

The proposed method helps to estimate and evaluate the impact of severe weather on Corps construction projects by analyzing historical weather data along with severe weather limitations of specific construction activities as the bases for analysis. This interim report focuses on the principal components of the proposed method, including: (1) location of an appropriate source of historical weather data, (2) development of an approach for performing data analysis, and (3) definition of severe weather parameters and criteria for various construction activities.

Approved for public release; distribution is unlimited.

FIG. 43: Usacerl document
FIG. 44: Usacerl template written in old language
<template templateID = "usacerl">
<let name = "page1" select="doc:page1[]">
<let name = "onlyBy" select="Spagel//line[matches(., "by\$\])]">
<let name = "titleBy" select="Spagel//line[OnlyBy >> ]">
<bind name = "dummy" ignore="yes" select="StyBly:">
<let name = "vertSplit" select="ext:vSplitStyBly,1>
<bind name = "DescriptiveNote" select="SvertSplit/Region[1]//line[matches(., USACERL.*INTERIM.*REPORT')]">
<bind name = "ReportDate" select="SvertSplit/Region[1]//line[wd[matches(., SDate)]]"/>
<bind name = "UnclassifiedTitle" select="SvertSplit/Region[2]">
</let>
</let>
</let>
<let name = "paraBy" select="Spagel//para[line[matches(., "by\$\])]">
<let name = "afterBy" select="Spagel//para[ . >> $paraBy]">
<let name = "topMargin" select="ext:margins$afterBy,0.8")">
<bind name = "PersonalAuthor" select="StopMargin">
</let>
</let>
</let>
<bind name = "DistributionStatement" select="Spagel//line[matches(., "{(Approved: for) (Distribution)\}]">
</let>
</template>

FIG. 45: Usacerl template written in new language

A.5 HEAD-ABSTRACT

A.5.1 HEAD-ABSTRACT DOCUMENT

Head-Abstract Document can be found in Fig. 46.

A.5.2 HEAD-ABSTRACT TEMPLATE IN OLD LANGUAGE

Head-Abstract template in old language can be found in Fig. 47.

A.5.3 HEAD-ABSTRACT TEMPLATE IN NEW LANGUAGE

Head-Abstract template in new language can be found in Fig. 48.

A.6 HEAD-ABSTRACT-TOPMARGIN

A.6.1 HEAD-ABSTRACT-TOPMARGIN DOCUMENT

Head-Abstract-TopMargin Document can be found in Fig. 49.
On Kalman Filtering With Nonlinear Equality Constraints

Suman J. Julier, Member, IEEE, and Joseph J. LaViola, Jr., Member, IEEE

1. INTRODUCTION

Some physical systems have equality constraints between their state variables. These constraints can arise for several reasons. Some constraints arise from the basic laws of physics. The laws of constitutive equations for a closed chemical reactor, for example, must remain constant throughout the reaction process. Other constraints arise from the mathematical description of a state vector. If the state is modeled as a constant matrix, the rows of the matrix must be orthonormal [2]. Constraints can arise from hierarchic or geometric considerations of a system. The coordinated sets modeled for an aircraft, for example, ensure that the acceleration vector is orthogonal to the velocity vector [2], [4].

A number of approaches have been developed to apply these equality constraints within the Kalman filter [27] framework. These can be broadly classified into pseudo-observation, projection, and reparameterization methods. The pseudo-observation method creates a fictitious observation whose variance is zero [3]-[5]. By substitution, it can be shown that the EF update equation projects the state estimate onto the constraint surface [7]. The projection approach constructs a projection operator that transforms the estimate so that it lies on the constraint surface [5]-[7]-[11]. The reparameterization approach simply reparameterizes the system so that the equality constraint is not required [11]. Since the focus of this work is on examining the effects of different algorithms for constrained estimation, the reparameterization approach does not fall within the scope of this paper. Then, we focus on the first two methods.

In this paper, we argue that although the pseudo-observation and projection approaches have clear, simple, intuitive interpretations when the constraints are linear, none of these properties hold when the constraints are nonlinear. The main reason is that both approaches implicitly assume that if the probability distribution obeys the nonlinear constraint, then the means of the distribution (which is the property maximized in the filter) obeys the constraint as well. The pseudo-observation method makes the further assumption that the Kalman filter update, which is a linear projection operator, is sufficient to constrain a probability distribution to a nonlinear surface.

To overcome these difficulties, we propose a new, two-step constraint application method. The first step applies the projection method to the unconstrained estimate. As a result, the probability distribution of the estimate is constrained to lie along the constraint surface. This causes the covariance in the estimate to decrease. In the second step, the distribution is translated so that its mean lies on the constraint surface. This, in turn, causes the covariance in the estimate to increase.

The structure of this paper is as follows. The problem statement is described in Section II. Section III introduces an alternative example which is used throughout much of this paper, estimating an angle of rotation about a single axis using complex numbers. We analyze the pseudo-observation and projection methods and show that they fail. Section IV examines the properties of the constraints in detail and identifies two types of constraints. The first type is a constraint on each sample in the entire distribution. The second type is a constraint on just the means of the estimate. We show that neither the pseudo-observation nor the projection approaches fully satisfies either constraint interpretation. Section V derives the new two-step approach for applying constraints. In Section VI, the new algorithm is illustrated in a head-tracking application that uses intact generations to represent constrained estimation. Conclusions are shown in Section VII.

This research project was supported by the National Science Foundation under grant number CDR-8818656. The National Science Foundation's approval of this manuscript does not constitute endorsement of the views contained therein.
FIG. 47: Head-abstract template written in old language
A.6.2 HEAD-ABSTRACT-TOPMARGIN TEMPLATE IN NEW LANGUAGE

Head-Abstract-TopMargin tempalte in new language can be found in Fig. 50.
KALMAN PLUS WEIGHTS: 
A TIME SCALE ALGORITHM

Charles A. Greenhall
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive, MS 298-100
Pasadena, CA 91109, USA
E-mail: Charles.Greenhall@jpl.nasa.gov

Abstract

KFW is a time scale algorithm that combines Kalman filtering with the basic
time scale equation (BTSE). A single Kalman filter that estimates all clocks si-
multaneously is used to generate the BTSE frequency estimates, while the BTSE
weights are inversely proportional to the white FM variance of the clocks. Results
from simulated clock ensembles are compared to previous simulation results from
other algorithms.

INTRODUCTION

The purpose of a time scale is to create a virtual clock from an ensemble of physical clocks whose
differences from each other are measured at a sequence of dates (a date being the displayed time
of a clock). The virtual clock is defined as an offset from one of the clocks, computed from the
measurement data by some algorithm. We usually want the virtual clock to be quieter than any of
the real clocks in both the short term and the long term.

One approach, which was tried in the early 1980s [1], is to run a Kalman filter on the clock
difference measurements, the scale of each clock having previously been modeled by a stochastic
linear system. The filter produces an estimate, unbiased and with minimum variance, of the
phase and frequency of each clock; moreover, if we offset the tick of each clock by its phase estimate,
we arrive at a single point on the time axis (if the measurements are noiseless). It makes sense,
then, to regard this point as the estimated origin of the ensemble, and to use the sequence of these
values as a time scale. This time scale, which was realized as TA (NIST), was reported to follow
the clock with the best long-term stability, regardless of its short-term stability [2]. My goals have
been to reproduce this finding, understand it, and improve the method. I seem to have achieved
the first and third goals, but not the second.

It turns out that a good time scale algorithm can be constructed by injecting some of the Kalman-
filter information into the traditional "basic time scale equation" (BTSE), which requires frequency

\textit{This work was performed by the Jet Propulsion Laboratory, California Institute of Technology, under a contract
with the National Aeronautics and Space Administration.}

FIG. 49: Head-abstract-TopMargin document
FIG. 50: Head-abstract-TopMargin template written in new Language
VITA

Vijay Kumar Musham
Department of Computer Science
Old Dominion University
Norfolk, VA 23529

EDUCATION
M.S, Computer Science, Dec 2010, Old Dominion University.
B.Tech, Computer Science, May 2006, Andhra University, India.

PROFESSIONAL EXPERIENCE
Worked as a Research Assistant for 2 years from Aug 2008 - Aug 2010 at Extract Project
group, Old Dominion University
Worked as a Software Engineer for 2 years from June 2006 - June 2008 at HCL Technologies, Bangalore India

Typeset using \LaTeX.