Problem Statement

CS 461 - CS Senior Capstone

XZES40 | Group #40

Zixun Lu (luzi), Shuai Peng (pengs), and Elijah Voigt (voigte)

November 14, 2016

Abstract

1 INTRODUCTION

Section	Author
Research and Benchmarking	author
XML/XSLT Document Transformation	author
XML/XSLT Document Parsing	author
XML/XSLT Document Caching	author
Web API	author
Web Interface	author
Debian Package	author
Command Line Interface	author
CentOS Package	author
Windows Packae	author
BSD Package	author

CONTENTS

REFERENCES

[1] *bootstrap 3 Introduction*. http://www.w3schools.com/bootstrap/default.asp.

[2] *CSS Introduction*. http://www.w3schools.com/css/css_intro.asp.

[3] *EMCO MSI Package Builder - Overview*. http://emcosoftware.com/msi-package-builder.

[4] *Events vs. Trees*. http://sax.sourceforge.net/event.html.

[5] *Features Advanced Installer, publisher=caphyon*. http://www.advancedinstaller.com/features.html.

[6] *Getting Started With Foundation*. http://foundation.zurb.com/develop/getting-started.html.

[7] *Kore.io*. https://kore.io/.

[8] *WiX Toolset Tutorial*. https://www.firegiant.com/wix/tutorial/.

2 Research and Benchmarking

The first application necessary for XML data processing are XML parsers and XML validators. The key aim is to check correctness of the input data, i.e. their conformance to either W3C recommendations or respective XML schemes. Hence, the benchmarks usually involve sets of correct and incorrect data and the goal is to test whether the application under test recognizes them correctly.

2.1 XML Conformance Test Suites

Binary tests contain a set of documents of one of the following categories: valid documents, invalid documents, non-well-formed documents, well-formed errors tied to external entity and documents with optional errors. Depending on the category, the tested parser must either accept or reject the document correctly (therefore, the tests are called binary). The expected behavior naturally differs if the tested parser is validating or non-validating. On the other hand, the output tests enable to test whether the respective applications report information as required by the recommendation. Again, validating processors are required to report more information than non-validating ones.

2.2

2.3

3 XML/XSLT Document Transformation

3.1 Option

Xalan-C++ is an XSLT processor for transforming XML documents into HTML, text, or other XML document types. Xalan-C++ implementation is based on two part – XSLT and XPah. XSLT (version 1.0) is the initial element of the XSL template language for XML. It contain the XSL Transformation vocabulary and XPath, a language for addressing to parts of XML archives.

Saxon XSLT

3.2 Goals for use in design

The major function of XZES40-Transformer is XML/XSLT document transforming. Apech foundation provide many ways to achieve this function.

3.3 Criteria being evaluated

3.4 Table compare

3.5 Discussion

3.6 The Best option

We will choose Xalan-C++ as our solution technology.

4 XML/XSLT Document Parsing

XML document compilation is the process by which an XML formatted document is taken and parsed into an in-memory object. The library we will be using, as requested by or client, will be the Xerces-C/C++ XML parser library; this review will evaluate that library. The other point of wiggle-room we have is in what way we parse and store the document in memory.

4.1 Options

4.2 Goals for use in design

4.3 Criteria being evaluated

4.4 Comparison breakdown

Technology	Description
xxx [?]	• AAA. • BBB. • CCC.
xxx [?]	• AAA. • BBB. • CCC.
xxx [?]	• AAA. • BBB. • CCC.

4.5 Discussion

4.6 Selection

4.7 Xerces-C/C++

The Apache Xerces-C/C++ library is a feature rich XML parsing library. It implements the XML 1.0 standard, it is well documented, and implements DOM as well as SAX methods of document parsing. Most importantly it integrates well with Xalan-C/C++, which is our top choice for performing document parsing.

This library was also explicitly requested by our client, a member of the Apache Software Foundation, so we should have some amount of internal organization support if we need to use it.

4.8 DOM Parsing

The first method of XML document parsing is DOM parsing. This method stores the entire parsed XML object in memory to be traversed later on. This operation is memory intensive but is useful for carrying out many operations on a document. [?]

4.9 SAX parsing

In contrast, SAX document parsing is performed on a data stream. SAX uses callbacks to trigger transformations on an XML document. This method is less memory intensive and can result in a performance boost, but requires a document stream for transformations to happen. [?]

4.10 Conclusion

We will be using the Xerces-C/C++ library to accomplish the task of XML document parsing. It is feature rich enough to give us leeway in our development where we need it The library is also lean enough that it should not affect performance negatively.

As for choosing between SAX parsing vs DOM parsing, we will most likely choose DOM parsing since it fits our application best. DOM parsing produces an easily cached tree which we can store and retrieve for later operations. It may be worth-while to investigate using SAX parsing early on in development because we find a notable performance boost, so some amount of SAX proof of concept work will be done for the sake of being thorough, but DOM will be the targeted document parsing method.

5 XML/XSLT Document Caching

5.1 Option

5.2 Goals for use in design

5.3 Criteria being evaluated

5.4 Table compare

5.5 Discussion

5.6 The Best option

6 Parallel Document Transformation

The parallel processing of the containment queries against an XML document utilizes parallel variants of the serial algorithm. First, the entries of the fully-inverted index are distributed among the cluster nodes for processing. Second, the containment query is processed by the cluster nodes to generate the corresponding lists of index entries. Third, the elements of the generated lists are checked against one another to produce the result set. The proposed algorithms can be differentiated based on the technique used for distributing the index entries between the slave nodes for processing.

6.1 Round-robin distribution of index entries

Using a round-robin scheme, the master node distributes the index entries uniformly over the slave nodes. At the end of this step, the number of index entries allocated to each node will be (1/n) * size of index-table, where n is the number of slave nodes in the Beowulf cluster. It is important to note here that the index entries with the same term values might land into the same or different slave nodes.

6.2 Hash-based distribution of index entries

This algorithm uses a hash function to partitions the index entries into a number of disjoint sets. This number is equal to that of the slave nodes. Each one of the disjoint sets is then assign to a different slave node for further processing. As a result, the number of index entries allocated to each node will roughly be (1/n) * size of index-table, where n is the number of slave nodes in the Beowulf cluster. Several variants do exist for this algorithm based on the entity that computes the hash function and the field in the index entry for which the hash function is computed. The entity that computes the hash function can be the master node or the slave ones. In the latter case, we assume that the complete inverted index is replicated across all of the slave nodes, and therefore, each one of these nodes can apply the hash

function to its local copy and select those entries that maps to the slave node itself. This process eliminates the initial need to transfer the index terms across the network. This step is skipped if the index entries are already distributed among the slave nodes as result of processing an earlier query.

6.3

7 WEB API

7.1 Options

Our web API may be implemented via a native-code web application, a Python or Ruby web application, or an Apache webserver CGI script. Each of these has pros and cons, but they all get the job done at some cost and with some benefits.

7.2 Goals for use in design

XZES40-Transformer will be accessible via a web API. This can be implemented a few different ways, but all of them must accomplish the same goal to allow people across the world to use the service over internet protocols. Three options being evaluated here vary widely in the way they achieve this goal, and so may represent more their technology and less the specific implementation.

7.3 Criteria being evaluated

The core of this application is related to document transformation. The more time that is spent on non-document transformation tasks should be kept to a minimum. Any technology we use to implement our web API should be simple, easy to develop, and easy to maintain. In short, *keep it simple stupid*.

7.4 Comparison breakdown

Our first option is to use an Apache CGI script, which would be a simple Python, Perl, or Ruby file which calls our C program and delivers returns the results to the user, all via Apache. The second options is to write a native web application using a web-app framework like Kore to handle HTTP requests. The third option is to use a python framework like Flask to handle HTTP requests. Each of these would be something that calls our document transformation functionality and exposes it to the outside world, how we handle that is important to consider.

Technology	Description
Apache CGI Script [?]	• HTTP requests are handled by the Apache web-server. • XZES functionality is called by "shelling out" to the program and returning the results. • Requires a running Apache Server on the host.
Kore web-app framework [?] [?]	• HTTP Requests are handled by the Kore framework. • XZES functionality is called natively with C code. • Acts as an independent daemon.
Flask web-app framework [?]	• HTTP Requests are handled by the Flask framework. • XZES functionality is called either natively or by using `exec` to "shell out". • Can act as an independent daemon or be managed by Apache.

7.5 Discussion

The above three technologies are all entirely valid choices for our application, and each approach the problem from different angles.

The Apache CGI solution is the Occam's Razor solution relative to the others. Using the Apache web server we can register a script written in Python, Perl, Ruby, etc to accept requests and return a response. This is simple, maintainable, and easy to create, especially if we are only concerned with implementing the API and not fancy features like accessing a database or storing user sessions. This option also allows us to leverage existing Apache Web-server power like load balancing and simple authentication without needing to write those features ourselves, they're just a configuration option away from being a reality.

Kore is a web-app framework which would allow us to develop our application in C/C++, which has it's pros and cons. C/C++ is notoriously difficult to write, and harder to write *well*, so it may be a time-sink. That said, it is nice to have a project which is written entirely in one language as contributors (ourselves and others) do not need to learn multiple languages to contribute.

Flask is another web-app framework, but one which is substantially easier to read and write. This has the benefit of being easier to maintain than a Kore framework, and we can write exactly the level of complexity we want from our API. On the other hand we would need to maintain a knowledge base of python, python frameworks, and python dependencies. So this is versatile but ultimately not necessarily easier to maintain than a kore framework.

7.6 Selection

Since we are working with Apache on this project, we want to develop a simple solution to our API problem, and the Apache Web server is a powerful tool we will choose to use this in our design. We will write a simple CGI script (which calls our C binary) and hook this into an Apache Web server. We will need to depend on the Apache Web server for our

project's package, but this should not be as hard as writing a webapp ourselves. We can also use the server to easily host our Web UI, which is a nice bonus.

8 WEBSITE UI

8.1 Option

There is three option for website user interface.

First is plain-text CSS file. CSS stands for Cascading Style Sheets. CSS describes how HTML elements looks like on screen, paper, or in other media. It can define styles for website, such as the design, layout and variations in display. CSS save a lot of work. It is not just control the layout of multiple web pages at once, but also fit the display for different devices and screen sizes.[**?**]

Second is bootstrap. Bootstrap is a front-end framework for faster and easier web development. Bootstrap provide templates for forms, buttons, table, navigation, modal, image carousels and may other. It is optional for JavaScript plugins. [**?**]

Third is foundation. Foundation is a font-end framework. It provide a responsive grid, templates, and CSS UI components, etc. It similar to bootstrap. [**?**]

8.2 Goals for use in design

The first idea about the user interface is user-friendly interface. When user first time come to using our program, we want it just simple without any guides or instructions. We also want our website looks beautiful and modern.

8.3 Criteria being evaluated

Cost: All of option is open source and totally free. However, CSS and bootstrap provides tutorials free. If we want learn foundation, we need pay to learn it.

Efficiency: CSS give simple plain-text interface. We don't actually know what the size is, and what the pixel will be on the screen. We have to try it, then we know what it looks like on our screen. However, bootstrap and foundation is responsive design. Both of them can change the size automatically in different size of screen. Both of them provide templates for creating web pages, but CSS do not provide.

Learning speed: CSS is the basis of HTML style sheet. It is easy to understand and learning, however it hard to make good design. Bootstrap and foundation spends time to learn, but after we learn the basically knowledge, bootstrap and foundation will be faster than CSS.

8.4 Table compare

Option	Open source	Efficiency	Templates	Learning speed
CSS	YES	Slow	No	Easy to learn
Bootstrap	YES	Fast	YES	Easy to learn
Foundation	YES	Fast	YES	Maybe easy, but cost money

8.5 Discussion

The table show us that all of them did similar work. However, they have different advantages and drawback. CSS is basis of HTML style sheet, and it works great, but it hard to create web pages beautiful. CSS is not responsive design. This make us hard to move web page into different size of screen, so we don't want to take CSS as our solution technology. Bootstrap and foundation does the same work. Both of them is open source and free to use. However bootstrap is much more stable and more templates, and there is free instructions in the W3C school. Foundation provides the tutorials, but it need to pay.

8.6 The Best option

The best option is the bootstrap for our project.

9 Debian Package

XZES40-Transformer target the Debian operating system. Our team will upload XZES-40.deb through libraries(Xerces, Xalan) to the Apache website. The users can directly download from the website and run in their Debian operation system.

9.1 Binary packages

Binary packages, which contain executables, configuration files, man/info pages, copyright information, and other documentation. These packages are distributed in a Debian-specific archive format. They are usually characterized by having a '.deb' file extension. Binary packages can be unpacked using the Debian utilityÂădpkgÂă(possibly via a frontend likeÂăaptitude); details are given in its manual page.

9.2 Source packages

Source packages, which consist of aÂă.dscÂăfile describing the source package (including the names of the following files), aÂă.orig.tar.gzÂăfile that contains the original unmodified source in gzip-compressed tar format and usually aÂă.diff.gzÂăfile that contains the Debian-specific changes to the original source. The utilityÂădpkg-sourceÂăpacks and unpacks Debian source archives; details are provided in its manual page. (The programÂăapt-getÂăcan be used as a frontend forÂădpkg-source.)

9.3

10 Command Line Interface

10.1 Options

10.2 Goals for use in design

10.3 Criteria being evaluated

10.4 Comparison breakdown

Technology	Description
xxx [?]	• AAA. • BBB. • CCC.
xxx [?]	• AAA. • BBB. • CCC.
xxx [?]	• AAA. • BBB. • CCC.

10.5 Discussion

10.6 Selection

11 CentOS Package

11.1

11.2

11.3

12 Windows Package

12.1 Option

WiX Toolset is a tools that build Windows installation packages from XML source code. Traditional setup tools used a programmatic, script-based approach to be installed on the target machine. However, WiX uses a different way. It is like a programming language, and it used a text file, which is based on the XML format, to describe all the steps of the installation process. Microsoft also uses WiX with all its major software packages. For example, the setup of Microsoft Office was developed entirely with WiX. [?]

EMCO MSI Package Builder is an installation tools designed for help developer create, maintain and distribute Windows Installer packages. [**?**] It helps developer create MSI package automatically by using changes tracking technology, which is used to generate installation project data, or manually by using the visual editor.

Advanced Installer is GUI tool that can simply create Microsoft Install packages. Advanced Installer create and maintain installation package, such as EXE, MSI, ETC. It based on the Windows Installer technology. [**?**]

12.2 Goals for use in design

XZES40-Transformer can be executed in multiple platform. We want to create install package, so user do not need setup every thing manually.

12.3 Criteria being evaluated

Cost: WiX Toolset is open source and free software. However, Advanced installer is company tools that we need to pay. EMCO MSI Package Builder provide free version for individual developer, but free version of package builder limit the functionality for create MSI package.

Security: WiX Toolset is open source project since 2004. Even the Microsoft also uses WiX to create MSI package, so WiX tools should be the most safe tools than other tools. The security of Advanced installer is acceptable. The company is created since 2002, and other big company used this tools, such as Sony, Dell, etc. EMCO MSI Package Builder is same as the advance installer.

Stable: WiX Toolset is open source project, and there is thousand of developer contribute on this project, so WiX Toolset is more stale than other tools.

Learning Speed: WiX Toolset have steep learning curve. It is better to understand the fact of Microsoft install package before using this tools. Advanced installer and EMCO MSI Package Builder provides GUI for user, it is easy to using and learning.

12.4 Table compare

Option	Open source	Cost	Security	Stable	Learning speed
WiX	YES	Free	Strong	Strong	Steep learning cure
EMCO MSI Package Builder	NO	Free version	Acceptable	Natural	Easy to learn
Advanced Installer	NO	Expensive	Acceptable	Natural	Easy to learn

12.5 Discussion

After I compare all of above method. I decide to choose WiX as my solution technology. The first reason why WiX is my solution technology is that it is open source software. It's powerful set of tools available to create our Windows installation experience. The second reason is that it represents by source code rather than GUI. GUI maybe easy for developer, but it also hided every thing behind the GUI. If we get some wired problem, we can't solve it via GUI software. Third reason is that it complete integration into application build processes. When install progress start, other setup modification are made in parallel, so no vital information will be lost. The setup program will complete together with the application itself. This is not required by client, but if client want to different way to create windows install package, we can choose advanced installer as our alternate technology.

12.6 The Best option

The best option is the WiX for our project.

13 BSD PACKAGE

13.1 Options

13.2 Goals for use in design

13.3 Criteria being evaluated

13.4 Comparison breakdown

Technology	Description
xxx [?]	• AAA. • BBB. • CCC.
xxx [?]	• AAA. • BBB. • CCC.
xxx [?]	• AAA. • BBB. • CCC.

13.5 Discussion

13.6 Selection

14 CONCLUSION

Elijah C. Voigt

Signature

Date

Zixun Lu

Signature

Date

Shuai Peng

Signature

Date

Steven Hathaway

Signature

Date