Problem Statement

CS 461 - CS Senior Capstone

XZES40 | Group #40

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

November 13, 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

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 Xalan-C++

Xalan-C++ is an XSLT processor for transforming XML documents into HTML, text, or other XML document types. Xalan-C++ variant 1.10 is a hearty execution of the W3C Recommendations for XSL Transformations (XSLT) and the XML Path Language (XPath). It works with a perfect arrival of the Xerces-C++ XML parser: Xerces-C++ rendition 2.7.0. The concentration for this discharge is on bug fixes, pluggable memory administration, and upgraded security changes in template execution.

3.2 Xerces-C++

Xerces-C++ is a validating XML parser written in a portable subset of C++. Xerces-C++ makes it simple to give your application the capacity to peruse and compose XML information. A mutual library is provided for parsing, producing, manipulating, and validating XML archives utilizing the DOM, SAX, and SAX2 APIs.

3.3

other

3.4 Conclusion

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 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.

4.2 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. [**?**]

4.3 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. [**?**]

4.4 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

5.2

5.3

6 Parallel Document Transformation

he 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

7.2 Goals for use in design

7.3 Criteria being evaluated

7.4 Comparison breakdown

Technology	Description
Apache CGI Script	Description
Kore Web-application framework	Description

7.5 Discussion

7.6 Selection

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: people across the world to use the service over the web.

7.7 Web application framework: kore.io

One option for every web-accessible application is to write a web application which exposes your project to the world. To this end a fairly simple library in development since 2013 called *kore* seems like a fair candidate. This would be used to handle incoming requests, call the XZES40-Transformer application, and return the transformed document.

This has some upsides and down-sides. The application will no-doubt be fast and allow our application to be entirely self contained in the C code. The application daemon which preserves our cache in memory could double as our web-application daemon. The downside to this is that it will increase the size of our code-base, which increases the amount of code we have to maintain, and can easily be a resource sink-hole.

7.8 Apache CGI script

A solution proposed by our client was to use the Apache web-server's CGI interface to expose our application. This would require writing a short script, probably in python, to handle the incoming and outgoing request. The script would call our application and take the output file and send it back to the requested user. [**?**]

This has many benefits, chief among them being that it will cut down on development time. Once we have a working command-line interface for local document transformation we can write a simple script to shell out to this, making it accessible to the world.

7.9 ???

7.10 Conclusion

We will chose to use an Apache CGI script for our web API.

8 WEBSITE INTERFACE

There is 3 different way to create a website pages. First is HTML, which is the core of the website language. Second is JavaScript, which is dynamic programming language used for creating website. Third is PHP, which is server language, but it make website too.

8.1 HTML

HTML stands for Hyper Text Markup Language. HTML was designed to display data - with focus on how data looks. HTML elements are resented by tag. HTML elements are the building blocks of HTML pages. HTML is a triad of cornerstone technology for website. There is no doubt that world wide web is created by HTML.

8.2 PHP

PHP (Hypertext Preprocessor) is a server scripting language, and it also is a powerful tool for making dynamic and interactive web Pages. PHP can be embedded into HTML code. PHP can be used in combination with various web template system, web content manamgement systems and web frameworks. Here is the PHP usage state,[**?**] and 224M sites use this language to make a website. The online cloud storage system[**?**] is PHP project that I find online. This project is designed for uploading and downloading file from the server. Becasue of PHP is server scripting language, it easy to transform data between clinet and server.

8.3 JavaScript

JavaScript is the programming language of HTML and the web. However, HTML do display data, it shows static information. JavaScript can improve that function, so we will have dynamic information on the website. JavaScript is prototype-based with first-class functions, making it a multiparadigm language, supporting object-oriented, imperative, and functional programming styles. There is 103,717,163 live website using JavaScript in World Wide Web [**?**]

8.4 Conclusion

In my opinion, I prefer to take HTML as my solution technology for this project. The first reason is that HTML is easy website language for everyone, and it support other script language, such as CSS and JavaScript, so I can create better and user-friendly UI. PHP can create dynamic pages, but it works better on server, not for client. The second reason is that we don't need many dynamic information in our website interface. HTML can feedback a link to user via ajax, so user can download new xml file. If clients want alternatives way to make website, I can take PHP as my solution technology, but the website will be looked like simple without designed.

9 DEBIAN PACKAGE

XZES40-Transformer target the Debian operating 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

10.2

10.3

11 Centos Package

11.1

11.2

11.3

12 Windows Package

XZES40-Transformer can be executed in windows platform. So we decide have windows install packages. MSI can be produced by many ways.

12.1 WiX Toolset

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. [**?**]

12.2 EMCO MSI Package Builder

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.

12.3 Advanced Installer

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. [**?**]

12.4 Conclusion

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.

13 BSD Package

13.1

13.2

13.3

14 Conclusion

15 References

[?] [?] [?]

References

[1] *Apache Tutorial: Dynamic Content with CGI*. http://httpd.apache.org/docs/current/howto/cgi.html.

[2] *Apache Xalan*. "http://xalan.apache.org.

[3] *Apache Xerces*. http://xerces.apache.org.

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

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

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

[7] *International Components of Unicode*. http://icu-project.org.

[8] *Javascript Usage Statistics*. https://trends.builtwith.com/docinfo/Javascript.

[9] *Online Cloud Strorage System PHP Project*. http://1000projects.org/online-cloud-storage-system-php-project.html.

[10] *The PHP usage stats for January 2013*. http://php.net/usage.php.

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

Elijah C. Voigt

Signature

Date

Zixun Lu

Signature

Date

Shuai Peng

Signature

Date

Steven Hathaway

Signature

Date