High Performance XML/XSLT Transformation Server

Spring 2017 Midterm Progress Report

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

CS 462 | CS Senior Capstone | Winter 2017

June 12, 2017

Abstract

An update on the development of the High Performance XML/XSLT Transformation Server named *XZES40 Transformer*.

Figure 1: Source: Wikimedia Commons [2]

Figure 2: Source: Apache Software Foundation [1]

1 Project Purpose

The purpose of the XZES40 document transformer is to accept input XML and XML stylesheet documents, perform an XML transformation with those files, and rerun the resulting transformation.

The application also adds the following useful features:

- Provides a user with a web interface to request XML transformations via file upload.
- Caches processed documents, reducing overhead on subsiquent transformations.
- Performs document transformations in parallel to service multiple users and requests simultaneously.

2 Project Status

2.1 Zixun Lu

At the middle of term, our team turned in whole the project and poster. Our application is done. Our application can transfer the XML files by using stylesheet. This application has three features. First, the application used the cache method storing the binary files in order to saving time and increase the application's usability for users. Second, you can add parameters when transferring the files. Third, the users can use website interface to transfer the files. Our application are completed by client's requirement.

2.2 Shuai Peng

For now, our team have finished our major requirement of our program. Our program has basic XML and XSLT style sheet transformation, caching old file, parallel computation, web interface, and user custom parameter passing. The command line interface and install package is option for our program, we may add it in the future. The multiply platform is hard, this may not support for our program. We have did a test to make sure that our program runs correctly, and the result of test is good. We did nice job for our program. Although we still missing some error catch, those error catch dose not impact our program, and the program still runs correct, so we may fix it in the future. To improve our program, we are revising the document. The documents will be easy to read and understand by user.

2.3 Elijah C. Voigt

At this point in development we are (mostly) feature complete. This was not without a bit of drama and stress, it was a bumpy road.

We have completed our requirements for the project, and even completed an additional requirement on top of that. These requirements include:

- Basic XML + Stylesheet document transformation.
- Document caching.
- Parallel document transformation.
- Web interface for document processing.

The additional requirement was one we found while talking with our sponsor. We found that our application did not meet the requirements necessary to perform *real world* document transformations, due primarily to our inability to handle extra resource files and custom build parameters. We added the build parameters (custom variables for the current transformation), but did not add the extra resource files.

3 Remaining Tasks

3.1 Zixun Lu

For this term, there are three things to do by end of the term. EXPO, three short writings and final report are remaining tasks. The most important thing is attending and presenting in EXPO. I think we should make two plans for the EXPO. One is what should we do during the EXPO, the other is if I have problems in the EXPO, what should I do? When we are preparing for the EXPO, we must to know all of knowledge of application and how to present to audience. The audience may not understand what we are talking and explaining for this application and we have to use simple words and simple logic method to make them to understand easily. We should practice more and introduce application to our friends and ask them the feedback. During the EXPO, if I can't answer or explain clearly, I will look for helps from our team members.

3.2 Shuai Peng

Here is what I should do in the future.

- we need polish our document to make it easy to read and understand.
- I need go throught what we have, and perpare for the EXPO day presentation.
- I may need do more test to make sure that it is not broken during we EXPO demo.

3.3 Elijah C. Voigt

I have no remaining development tasks. The only things left we have are:

- This report.
- The demo accompanying this report.
- Engineering expo.
- Our final project report.

This is all completely managable and I believe we will complete all of it on time and with relatively few hiccups along the way. I am not concerned about our performance at Expo, or at least not concerned about my own performance at expo. I have practiced our pitch many times and am able to get a live demo of our product quickly and dependably.

As a group, we did not hit all of our project requirements, most of which were the job of Zixun Lu; the largest of these beign a software installation package.

4 Problems Encountered

4.1 Zixun Lu

On the last term, I didn't mention the XML which are provided by our client are different from simple XML files to our group members. Those high level XML files are transformed by adding more extra files, like ent and dtd files. So I delay the time of developing the application. They have to add those features to this application. If we don't add those features to this application, our application can not transform high level XML files, just only transform simple XML files. At the beginning of this term, our team finished the whole project. I did the bench-marking of our application. We our two competitors which are Xalan and Altova. One is open source and other one is paying to use. Our goal is faster than Xalan. But the result of our application transformed time is much faster than Xalan and Altova. Our application

transformed time is 0.110 seconds. The Xalan and Altova application are both over 0.300 seconds. We think this is different computer configuration effect the final result. I did the bench-marking of our application in Elijah's laptop and did the Altova application and Xalan in my laptop. His laptop is faster than me. We want to make sure the benchmark data is correct and re-do the benchmark for Xalan and Altova application at the same computer configuration laptop. At the end, our application is still faster than Xalan and Altova application. Our application transformed time is 0.110 seconds and Xalan transformed time is 0.196 seconds. The Altova application is 0.161 seconds. Our teams are happy we did the awesome works.

4.2 Shuai Peng

The major problem since last term is that we get new requirement from the client. Cline give us a new requirement that we need pass the parameter from the website to our transformer. Thus we need add a new feature for that. However, during we were developing this function, we faced a problem is that we need additional file supporting for the XML and XSLT file transformation, and those files can not import to our transformer, because those file looks like plugin for the XML and XSLT file. We talked to our client, but he did not response for us, so we finally decided to ignore that additional file, and we just simply install into our system. Therefore, our program is support only simple transformation, and simple parameter passing.

After we fix the additional file, we get some wired bug for the parameter passing. Elijah and me dose not a good front end developer. When we passing the parameter from command line, it works perfect, but if we change it to the website, we get bugs for the parameter passing. We did lots of test on that, and finally we fix it by check the Ajax problem. It looks like the Ajax parse the string into HTML, so our transformer can not get correct data from the website. We fix it via forcing string pass in the Ajax.

4.3 Elijah C. Voigt

Our group has had many problems. During this term most problems fell into one of two buckets:

- Technical stumblings.
- Unfair work distribution.

The first are obvious. They were problems encountered in development like an edge case or a bug or something *'just not working'*. These were solved swiftly and we were able to squash all of the bugs well before our code-complete deadline.

The second has been a problem in our group the entire year. These were things like Zixun not being technically able to perform a task and offloading one of his responsibities onto another team-member. For example I ended up being responsbile for generating benchmarking data for our application **and** creating the entire website. He was also responsible for some component of the C/C++ code, but was not able to write any C/C++ code so this responsibility was offloaded to Shuai.

Of course these responsibilities could have been 'dropped', but for the sake of the project I decided that it was in our best interetst to ensure that the aplication *worked* instead of selfishly allowing major, show-stopping components of the project to go undone.

5 Interesting Code

5.1 Zixun Lu

There are some interesting commands when doing the benchmark. sends test results to an XML-based results file

```
logFile resultsFileName.xml
```

For StylesheetTestletDriver, use a different class for the testing

```
testlet TestletClassname
```

Most tests can either accept options from a properties file, via:

```
TestName -load file.properties
```

5.2 Shuai Peng

Here is our parameter passing code for the transformation. This part of code is parsing the request from the website.

```
xzes::job_t* xzes::parse_request( char* input )
{
    xzes::job_t *out = new xzes::job_t;

    std::string tmp (input);

    std::vector<std::string> tmpv = xzes::split(input,',');

    out->jid     = tmpv[0];
    out->xml.uri = tmpv[1];
    out->xsl.uri = tmpv[2];
    out->out.uri = tmpv[3];

    // Stream ends in an empty "," for buffer cruft.
    for (int i = 4 ; i+2 < tmpv.size(); i += 2)
    {
        param_t x = {tmpv[i], tmpv[i+1]};

        printf("%s:%s\n", x.key.c_str(), x.val.c_str());

        out->param.push_back(x);
    }


    return out;
}
```

Parse_request function receive input from the CGI script, and parse it as a string. We create a vector to handle the data, and split by the ",", so our program understand what value need to be set in the transformer.

5.3 Elijah C. Voigt

I learned JQuery for this project! JQuery is a very small Javascript framework focused on manipulating the HTML dom and doing tasks that are *very* common in Javascript, like AJAX requests.

Here's some of the code I wrote for the frontned interface.:

```
// When happens when you submit the form
$("form#transform").submit(function(event) {
    event.preventDefault();

    var file = "";
    var formData = new FormData($(this)[0]);
    formData.append("parameters", JSON.stringify(processParameters()));

    // Send the modified form to the /cgi-bin/xzes.py endpoint.
    // If you messed with the apache service this will need to change.
    $.ajax({
        url: "/cgi-bin/xzes.py",
        type: "POST",
        data: formData,
        dataType: "text",
        async: true,
        statusCode: {
            404: function(data) {
                notSuccessful("&#x1F6AB; &#x2601; Could not reach the transformer server...");
            },
            400: function(data) {
                notSuccessful(data.responseText);
            },
            // On successful transformation
            200: function(data) {
                // If the repsonse is non-empty
                if (data.responseText != ""
                 && data.responseText != undefined) {
                    // Print the success, the filename, and give buttons to view/downlaod the file.
                    successful(data.responseText);
                } else if (typeof(data) == "string") {
                    // The response may just be a string...
                    // We're not sure why this happens sometimes...
                    successful(data);
                } else {
                    // Otherwise give a generic failure message
                    // Weird edge case where output xml is parsed by jquery as html
                    notSuccessful("The transformation was not successful.");
                }
            },
        },
        cache: false,
        contentType: false,
        processData: false,
    });
```

```
});
```

This describes what happens when you click the "transform" button. In broad terms, it constructs an HTTP POST request form, sends that form, and reacts based on the response. When it gets either a positive or negative status code from the server it will either display an error to the user or give the user buttons to download and display the transformation results.

This results in the frontend dynamically loading the result of a document transformation in the background. The user is not redirected to a new webpage when they submit a document transformation, instead they click the button, the transformation is carried out in the background, and they are shown the new files when the transformation is complete.

6 Relevant Media

References

[1] *ASF Press Kit: Apache Software Foundation Logo*. URL: https://www.apache.org/foundation/press/kit/.

[2] *Wikimedia Commons: Oregon State University Logo*. URL: https://commons.wikimedia.org/wiki/File:Oregon_State_University_log