CMPSC 441
Distributed Systems
Spring 2016

Laboratory Assignment Five: Using XML-RPC for Client-Server Communication

Introduction

The ability to implement and evaluate software that supports communication between a client and a server is a crucial laboratory skill that will support further development and experimentation in the field of distributed systems. In this laboratory assignment, you will download and run a program performing client-server communication through the use of a remote procedure call (RPC) that encodes parameters using the eXtensible Markup Language (XML). Additionally, you will implement five programs that will support the experimental evaluation of the performance of using XML-RPC for client-server communication on a single host. Finally, using either Markdown or the LaTeX text formatting language, you will write a detailed scientific report explaining the performance results that you identified while completing this assignment. You will complete this project by working with a partner; you and your partner will collaborate through a Slack channel and a Git version control repository throughout the week during which you finish this assignment.

Review Your Textbook

Before starting this laboratory assignment, you and your partner should review and discuss the content in Chapters 1 through 3 of your textbook. If one member of your partnership has questions about this content, then please resolve them before starting this laboratory assignment. Next, you should collaboratively examine the material in Section 4.1, paying particularly close attention to the technical digram in Figure 4-6 and the ten steps associated with performing a remote procedure call, as given on page 129 of your textbook. Please see the instructor if you have any questions.

Access the Course Repository

Now that you and your partner have gained access to the `cs441S2016-share` repository, both of you should run "`git pull`" to receive the source code and two Java archive files needed to complete this assignment. Then, one of you should create a new repository called `cs441S2016-lab05-<first user name>-<second use name>`. Finally, make sure that both individuals in your partnership, and the course instructor, have access to the repository that you will use for this assignment. Students who have questions about the use of the Git version control system should work with the instructor to ensure that they have resolved them before completing this laboratory assignment.

Exploring Client-Server Communication

Please find the `labs/lab5/` directory in our course's repository and work with your partner to study and understand the source code for the files called `FactorizationXMLClient` and `FactorizationXMLServer`. What are the key aspects of how these two Java classes support client-server communication? For instance, you will notice that these Java classes specify items like a "host", a "port", and a "Web server" — what are the meanings of these terms? Can you draw a

diagram that illustrates the general interaction between a client and a server, customized for these two classes that use XML-RPC? Finally, make sure that you understand how these two classes perform communication in a way that is similar to and different from those classes that performed socket-based messaging. Please see the course instructor if you have questions about these issues.

You and your partner should work together to fully comment all of the source code in these two Java classes, making sure that you correctly explain how all of the methods and lines of code work. If you consult online references to support the development of your comments, then please include a reference to those sources in the comments. Once you have finished compiling these two classes, please run the `FactorizationXMLServer` in one terminal window and then attempt to have the `FactorizationXMLClient` connect to it. For now, you should run both of these programs on the same computational node. Once you have gotten the client-server communication to work correctly, record the output of the client and the server to demonstrate that they are working. If you are interested in exploring this further, please try to run the client on the computer of one partner and the server on the computer of another. Students should be aware that, depending on their setup, they may need to enhance these two Java classes to fully support remote communication.

Finally, you will notice that the `FactorizationXMLClient` and `FactorizationXMLServer` have some "hard-coded" values in them. For instance, the port on which XML-RPC communication takes place can only be defined if a programmer changes a value in the source code and recompiles the Java class. Since this is not ideal from either a usability or an experimentation perspective, you and your partner should revise the source code so that it accepts command-line arguments for any values that you deem to be configurable by the user. While you may use the `args[]` array to accept these parameters, students are encouraged to consider the use of the JCommander library available at `http://jcommander.org/`. Please see the instructor with questions about this task.

Implementing and Using a Benchmarking Framework

Following the paradigm from a previous laboratory assignment, you and your partner should implement the code for an `LSXMLClient` and the `LSXMLServer` — these Java classes will serve as the start of a simple benchmarking framework that you and your partner will implement. The "LS" abbreviated in the names of these classes is for the words "Large" and "Small" — what about the client-server interaction in this system should be classified in this fashion? After discussing this question with your partner, you should also create six additional Java classes respectively called, `SLXMLClient`, `SLXMLServer`, `SSXMLClient`, `SSXMLServer`, `LLXMLClient`, and `LLXMLServer`. As you are implementing all of these Java classes make sure that the ways in which these clients and servers interact adhere to the names that you have assigned to the corresponding Java classes.

Overall, these programs will enable you to establish some "baseline" measurements for the cost of performing client-server communication on a single computer. Next, you and your partner should implement one final program that is more like the `FactorizationXMLClient` and `FactorizationXMLServer` — that is, it should perform some type of "useful" computation on the server whenever the client makes the appropriate request. As long as full acknowledgement is given, teams are encouraged to reuse code from a previous assignment (e.g., the socket-based benchmarks) as they are implementing the final program. If you do adopt code from a previous assignment, then please remember that past benchmarks used sockets and this assignment requires the use of XML-RPC. Your team can brainstorm ideas for new benchmarks with the instructor.

As you are creating your programs, make sure that you give them command-line arguments that will effectively support running experiments and collecting data about the response time of the client-server interaction. Once you have implemented all of the Java classes that you will use in your benchmarking framework, you and your partner should divide up the work associated with running your experiments. For each of the four "baseline" client-server configurations, you should run the client multiple times to collect many timings and then calculate, at minimum, arithmetic means and standard deviations. Then, you should run both of the "useful" computations (i.e., the one for factorization and the one that you have implemented with your partner), again collecting multiple timings and calculating arithmetic means and standard deviations. For this laboratory assignment, you and your partner should focus on benchmarking local client-server communication on the same computational node. However, teams that wish to potentially earn extra credit should consider also running their experiments to measure the costs of remote communication. Finally, please make sure that your report compares and contrasts the cost of communication when using either sockets or XML-RPC. Which approach is faster? Can you explain why this is the case?

You and your partner should organize all of your empirical results into tables of data. While not absolutely required, you may consider preparing graphs of your results using the R language for statistical computation. Next, you should analyze the results in attempt to find and explain patterns in the data. Overall, what do your results show you about the cost of performing client-server communication? Once you are finished running these experiments, can you identify any ways in which you could have improved the performance of the client or the server? Finally, you and your partner should write a detailed report, using either the Markdown or the LaTeX text formatting language, that introduces the design of your experiment and your research questions, explains how you conducted the experiments, and then thoroughly presents and analyzes the results.

Summary of the Required Deliverables

This assignment invites you to submit printed and signed versions of the following deliverables. Additionally, all of these deliverables must be in the repository that you created for this assignment.

1. The well-commented source code of the Java classes that form the two "useful" benchmarks.

2. The well-commented source code of the Java classes for the four "baseline" benchmarks.

3. Using both text and diagrams, a description of client-server communication with XML-RPC.

4. A detailed paper that reports on the empirical results arising from the use of the benchmarks.

5. A description of the challenges that you encountered when completing this assignment.

6. A detailed listing of the tasks completed by each member of your partnership.

In adherence to the Honor Code, students should complete this assignment on an group basis. While it is appropriate for students in this class to have high-level conversations about the assignment, it is necessary to distinguish carefully between the student who discusses with others the principles underlying a problem and the student who produces assignments that are identical to, or merely variations on, someone else's work. This means that, for instance, all of the other comments, source code, data, and written reports — with the exception of the code that the instructor provided through the "share" repository — should be the original work of the members of the partnership. Students who have questions about the Honor Code and how it applies to this assignment should schedule a meeting with the course instructor before this assignment's due date.