CS 578 – Software Architectures
Spring 2009
Class Project
Due: *May 1st, 2009*
– see course websites for submission details –

Introduction

As you will have most likely experienced in your careers as software developers, not all software is well architected (all software has an architecture, but not all architectures yield insight into the software system). In many application domains, practitioners who are not software engineers are the primary developers of software; in some cases, the resulting software systems are trivial, though in many cases these efforts result in very complex code. In this project we will be studying one such domain: scientific software.

In the last 20 years, scientists in a variety of fields, including chemistry, physics, and materials science, have made new discoveries using computer simulation, but at the same time they have faced multiple hurtles in technology adoption. Scientific software is interesting to study for software architects because it is both complex in its functionality (new science is often very difficult to simulate in software), and also in its engineering (scientific validation usually requires large amounts of data to be validated).

There are a number of technological aids to help scientists create large-scale software systems, including workflow languages and, more recently, domain specific software architectures. Workflow languages allow users to orchestrate execution of multiple programs and map data dependencies/control flow between these programs. Domain-specific software architectures aid the scientist in separating the computational complexity from engineering concerns such as data discovery and marshalling, data provenance reporting, and error handling.

In your class project, you will be asked to forms teams of 3-4 students and study a piece of scientific software written in Java. You will be asked to refactor this code into scientific software modules (which we call *kernels*), create executable programs that implement these kernels including requisite data marshalling and un-marshalling, integrate these executables with a scientific workflow language, and finally, develop a version of the software using a domain-specific software architecture called SWSA and an architecturally-aware middleware framework known as Prism-MW.

Assignment Details

The first activity you will perform for this project is team formation. Please select teammates from class in order to form groups of three or four. We ask that you select teammates such that all members of the team have the same course instructor. Additionally, if you are a DEN student and need help forming a group, please email the TAs. We request that you send an email to the TAs no later than March 30th to notify us of your team formation (see Timeline/Deliverables for more information including a description of the required information).

Upon receiving your team formation email, we will assign you to one of a number of different scientific software packages. You will receive a package via email that includes the original source, test dataset, instructions for execution, and a short (1 page) description of the scientific software. You will be required to perform the following activities using this package:

1. *Kernel Extraction*

We define a kernel to be a software module that have defined input and output and implements a single high-level concept in the code. Examples of kernels are the code to solve an equation, or the code to update the state of a system represented by a matrix. This is only an intuitive understanding. In the next week, we will give a lecture that will introduce this concept more formally. You will need to decompose the scientific code you receive into kernels and justify your modularization.

2. *Create Executables*

For each kernel you extract from the original code, you will need to create an executable. Each executable will be callable from the command-line and must take only one parameter (the path to an input file). Each executable will write an output file. The format of these files (with the exception of the existing input data set) is up to your team. You will need to write date marshalling and un-marshalling code as part of this effort.

3. *Create a Workflow Version of the Science Code*

Using a workflow technology, you will recreate control flow and data dependencies for these executables that match the original scientific code. A workflow is a processing model that orchestrates tasks (the executables) and manages data dependencies between tasks (the files you are reading and writing). This can be considered the application of dataflow architecture to the original system. More information about the workflow technologies will be presented to you in lecture next week, and you will be assigned a particular workflow technology to use with your assignment package. You will be required to run the resulting system and gather performance statistics for the input dataset.

4. *Create a SWSA Version of the Science Code*

Using Prism-MW (http://sunset.usc.edu/~softarch/Prism/), an architecturally-award middleware technology, you will implement a version of the scientific software in SWSA. SWSA (Scientific Workflow Software Architecture) is a domain-specific software architecture for scientific code that implements a workflow (or dataflow) architecture in such as way as to separate engineering tasks like data marshalling and error handling from scientific concerns. A specification for SWSA and a mini-tutorial for Prism-MW will be given next week during lecture.

You will need to create Prism-MW components that execute your kernels and you will need to create special SWSA connectors that will orchestrate execution in the same manor as your workflow specification. You will be required to run the resulting system and gather performance statistics for the input dataset.

5. *Reports*

In addition to these deliverables, you will be asked to prepare two status reports (to be emailed to the TAs following the timeline specified below) and a final report. The final report will consider of documentation regarding the four previous parts of the assignment, including the following elements:

i. Justification for your selection of kernels
ii. Documentation of the design of your executables (including data format explanations)
iii. Workflow specification
iv. SWSA specification
v. Performance characteristics of each resulting system
vi. Critique of SWSA as a useful abstraction

Timeline/Deliverables

- Monday, March 30: Email woollard@usc.edu and csci578@usc.edu with the names, email addresses and student ID numbers of each team member. One email per team please.
- Thursday, April 9: Status report 1 due. One per team. Details and format will be sent to the teams by April 2.
- Thursday, April 23: Status report 2 due. One per team. Details and format will be sent to the teams by April 16.
- Friday, May 1: Project materials due (project report, workflow-based system, and SWSA system).