SCORE Project Proposal:
Global Studio Project (GSP)
Industrial Sponsored Project
December 12, 2007

Background

Siemens Corporate Research (SCR) has been performing research to decompose large-scale software system requirements into a well-structured set of software components that can be developed in parallel among globally distributed development teams. The processes resulting from this research are being experimentally applied within the Global Studio Project (GSP). This project develops software components for a Siemens product line while applying an experimental global development process using globally distributed student teams located in universities in Europe, Asia, and the Americas.

What Will the Student Team Develop?

The anticipated product that will be developed by the student team is called TDE/UML-PM. TDE/UML is a software tool that SCR uses for developing model-driven tests. For the GSP, the tool is being extended for the purpose of generating system performance models (PM). A number of performance modeling tools will be enhanced and integrated into a common service oriented architecture to create a powerful platform for the computer system performance analyst. Student teams will have the opportunity to develop and test models, software components, and interfaces for products for model-driven testing and analysis.

During this fourth year of the GSP, the focus is on improving the user experience and extending the model transformation capabilities of TDE/UML-PM to specify and solve problems that are not solvable using product form queuing networks. These networks can be solved using analytical formulas that have a well-defined structure. The requirements engineering of general-purpose software applications using UML may specify conditions that violate these product form requirements. A methodology is used to specify general node type behavior at the UML model layer by using a library of node behaviors as implemented in the performance modeling layer. TDE/UML-PM provides performance engineering support for the evaluation of system architecture alternatives. These evaluations are usually executed early in the software development process, when a UML Deployment Diagram Model and a Sequence Diagram Model are specified. The models are annotated with operation arrival rates and resource execution times.

What are the Roles of the Students and SCR?

The students will function as members of a small software component development team. Thus, they will define a component development process and fill the roles associated within that process (e.g., architect, subject matter expert, requirements engineer, tester, developer, quality assurance, project lead), and they will develop and incrementally deliver the component. SCR staff and student interns in Princeton, New Jersey will function as the central product management team who will define as high level stakeholder feature requests the software product to be developed by the student team. Thus, SCR will be the sponsor, stakeholder, or customer for the student development team. SCR will assign points-of-contact to the student component team, such that requirements can be elicited, analyzed, and clarified. Project artifacts will be delivered to the central product management team as well as the SCORE Program Committee.

What has to be Done?

To initiate the development of a new software component, SCR will provide the student component development team a documentation package consisting of:

- Business intent: What the product will generally do, into what domain it would be sold, and some background information about the market.
- Stakeholder requests: A description of some of the key features that the software will implement.
- Acceptance tests: The acceptance test plan that the component must satisfy.
- Incremental development plan and integration dates: A proposed skeleton schedule is provided specifying the durations for the component development increments and the fixed dates for iterations to be released to the central integration and test team at SCR.

Upon receipt of the documentation package, each student team will review it and begin the work necessary to deliver to SCR the first set of features for the initial increment per the desired release date. The students will interact with their Siemens point-of-contact to answer questions about the documentation package and domain.

For scoping purposes, it is assumed that each software component to be developed by the student teams will consist of approximately 200 function points or 10 KLOCs. Furthermore, components will be assigned to the student teams to be of a size and complexity such that it can be fully implemented within the academic year school term (8-12 months), with an estimated team size of 4-6 students.

Desired Student Skills

Students making up the component teams should be formed based on the following desired skills and experience profile.

1. Undergraduate degree in software engineering, computer science or related discipline.
2. Some experience with industrial software development.
3. Experience with project software development tools and languages (e.g., C#, Visual Basic, C++, Java, UML).
4. Understanding of the Microsoft .NET or other middleware frameworks.

5. English reading, speaking, and writing skills, since product documentation and communication will be in English.
6. Familiarity with software development processes, including agile processes such as Scrum.
7. Willingness to sign a Siemens non-disclosure agreement (NDA) concerning any proprietary product documentation that may be provided (e.g., system test procedures).
8. Ability to communicate with the Siemens point-of-contact person, the central team staff in Princeton, and any case study investigators or evaluators who may conduct periodic interviews with the students.
9. Some understanding of software metrics and data collection.
10. Some interest in system performance analysis, modeling, or simulation.

Software Tools

Component development teams will utilize a common set of software tools for developing their product. Some tools may be *required* and student teams will not be able to deliver development artifacts without them. Some tools may be *desirable*, but project work may be accomplished without them. There may also be some Siemens supplied tools that will be *experimental*. To minimize costs, the students are encouraged to use open source tools whenever possible. A web site project portal will be set up for the student teams containing non-proprietary project information and the set of common tools to be used for the GSP. An example list of the functions provided by software tools is given below.

1. Requirements management – It is recommended that Teamcenter Systems Engineering be used for requirements management. University teams can find an application on the SCORE web site to acquire Teamcenter as an in-kind grant.
2. Design
3. Configuration management
4. Defect management
5. Test environment management

Approach – Student Teams

Upon registration and selection, the student teams will be given the documentation package described above. Per the academic requirements, the experiments to be performed, and staff availability, each team will likely have a project kick-off teleconference meeting with their Siemens point-of-contact. Documentation, training, guidance, and overall project goals will be discussed with the student teams and the Siemens point-of-contact person. Additional Siemens staff may participate in the kick-off meeting depending on availability and the experiments to be performed.

The student teams will begin the project by reviewing the documentation package, self-organizing into roles, defining their development process, analyzing requirements, detailed design, and test planning. The teams will begin by defining how they will accomplish the work required to deliver iterations of the component per the specified intermediate integration dates.

After an initial planning time period (assumed to be approximately one month for most schools), the student team will be required to submit a proposed project plan or statement of work to SCR. This provides an opportunity for discussion concerning the realism of the project scope and schedule dates.

Each student team will be required to provide external communications and periodic information of two types, depending on the experiments. One type of information will be project status related. This would likely be done via telephone conferences and email to the central team at SCR in Princeton, typically weekly. In addition, case study information will be given to the experiment data collectors. This will most likely be in the form of telephone interviews and web-based surveys. Students are encouraged to maintain a log of how much time they spent on different tasks within the project.

Students may also interact with the SCR central team and/or their point-of-contact person for various quality assurance type reviews such as requirements or design reviews. These reviews may take the form of meetings at the university or using collaborative communications tools (e.g., videoconference, teleconference). Information to the student teams (e.g., for answering requirements questions) will come primarily from the point-of-contact person. The point-of-contact person will be the first "line of defense" for answering students' questions as they come up during the course of development. We anticipate that the industrial point-of-contact person will be available to the student teams on the average of 1-2 hours per week including the weekly status meetings.

Students' code will be validation tested at the system testing location after each engineering release starting on the integration dates specified. Defect reports will be generated by the system test team, and communicated to the student teams via email or defect tracking tools. Corrections should be made and integrated prior to the next release date.

Deliverables

The following deliverables are required from the student component teams to the central team at SCR.

1. Project Plan: Delivered a few weeks after the start of the project as described above as a Word document.
2. Requirements Model: Use cases described in UML for detailed functionality to be implemented.
3. Design Specification: Detailed design description of the component architecture to be implemented.
4. Test Plan: Description of the tests to be performed on the component to be developed.
5. Code: Operational code will be developed and checked into a CM tool. Test harness code will also be delivered.

6. Release Notes: For each engineering release corresponding to the integration dates, a set of release notes will be generated. The release notes will briefly describe the features in the release, defects corrected, known outstanding defects, and any special instructions concerning build, installation, or test environment. An email will be sent to the central team buildmeister announcing the release with the release notes attached as a Word document during the local work day (or prior to midnight) of the specified integration date.

Note that the deliverables listed above correspond to product development artifacts as defined by most development processes. Additional information may be requested (e.g., personal time logs) from the students to support the case study/experiments data collection.

If appropriate, the students may be graded on the quality of the delivered artifacts as per university course requirements.

In addition to project deliverables, the student team will assemble the SCORE contest deliverables; i.e., the Summary Report and Final Deliverables in accordance with the contest rules. The SCORE deliverables will be evaluated based on the quality of the artifacts, not on the amount of effort that was expended to create them.

Lessons to be Learned

The students will have the opportunity to develop a software component across the full development lifecycle within an environment that simulates an industrial development team. The students will learn how to define an agile component development process, organize a small project, and perform the roles within such a development team. They will also have the opportunity to experiment with and be constrained by development processes within which they must fit with respect to schedule, quality, interfaces, and deliverables. The students will apply modern software engineering technologies and have the possibility to contribute to developing a new industrial product line.

Financial Support

Students will be prohibited from being paid directly by the sponsoring company for their work on a SCORE project. However, the sponsoring company may cover the travel costs of the students to visit the company or the ICSE Conference. This will be requested and approved in advance by the sponsoring company. Reimbursements may be done by grants to the university, the ICSE Conference, or to the traveling students. For student teams located a large distance from Princeton, New Jersey, project meetings will be done with teleconferences.