

Work in Progress – The Software Enterprise

Kevin Gary, Barbara Gannod, Gerald Gannod, Harry Koehnemann, Tim Lindquist, Richard Whitehouse
 Division of Computing Studies
 Arizona State University at the Polytechnic Campus
 Mesa, AZ 85212 {kgary,bgannod,gannod,hek,tim,row}@asu.edu

Abstract - Computer science and software engineering programs are in a constant struggle to update curriculum content and instructional methodologies so graduates are better prepared to enter today's fast-paced technology sector. Despite these efforts, the general perception of industry remains that new graduates are ill equipped to perform their duties when entering the marketplace. We find our students are often caught between the academic mission to learn the science and the need to find gainful employment when they complete their degree requirements. Our response to this dilemma is the introduction of a four-semester capstone project sequence dubbed "The Software Enterprise". The instructional methodology involves a combination of lecture, problem-centered learning, and project work to expose, practice, and apply concepts over short intervals. In this way, students understand not only the skills but also how they are applied. The length of the sequence ensures prolonged and repeated exposure to applied concepts. This Work-In-Progress paper presents our project sequence features.

Index Terms – Applied teaching and learning, Capstone sequence, Collaborative learning, Software process

INTRODUCTION

Students must emerge from a "write-a-program-get-a-grade" to a "follow-a-process-produce-a-deliverable" mentality and eventually to "use-and-improve-processes-to-solve-customer-problems". This evolution from learner to practitioner is a cultural mindset even at the personal level, and should begin to form while the student is still in the degree program. In our efforts to address these challenges, we asked ourselves how graduating students entering the marketplace gain the skills needed to become competent professionals. We identified some key characteristics then went about designing ways in which these experiences could be incorporated into our Capstone course. The result is a four-semester Capstone project sequence, currently in its second semester, which focuses on developing applied knowledge in a broad range of software engineering practices.

The newly formed Division of Computing Studies (DCST) on the Arizona State University East campus is tasked with developing programs in the polytechnic model. Graduating students are expected to be "industry-ready". In the model of a polytechnic, an increased emphasis is placed on hands-on practice over pure scientific study. DCST has responded by offering a new Bachelor of Applied Computer Science program that embodies the polytechnic spirit. Within DCST programs a single semester capstone project course was

offered. We have reported valuable lessons learned through several iterations of this course [1]. We noted that continuity of software projects across semesters was very difficult, yet single semester projects were limited in size and complexity. Students were usually confined to a single role in a project team, if project roles were adhered to at all. It was also difficult to teach process material, such as requirements gathering and management techniques, while facilitating a single semester project. Despite these challenges, the project course was generally rated favorably on student evaluations.

Given the emphasis of DCST and its programs, and our recent lessons learned, our capstone project course is undergoing a significant evolution in response to these needs. This paper presents the Software Enterprise, our plan for evolving our single semester capstone project course into a multi-year, multi-semester, and multi-project sequence.

SOFTWARE ENTERPRISE CURRICULAR PLAN

The curriculum plan calls for two one-year projects that a student participates in serially, as shown in Table 1.

TERM	FALL	SPRING
Year 1	Tools & Process	Development, Quality & Deployment
Year 2	Requirements & Analysis	Project & Process Management

TABLE 1. STUDENT PARTICIPATION TRAJECTORY

A student entering the Enterprise sequence, typically as a junior, begins by taking a Software Tools and Process course. In this course a student gains exposure to a set of tools that support the software process, including IDEs, data gathering and analysis tools (metrics), unit testing, configuration management, build, and deployment tools. This exposure is presented in the context of the Personal Software Process (PSP [2]). We are adapting the PSP in order to ensure we have sufficient time to cover both tools and process in one semester.

The student's second semester is spent in Construction and Transition. Students spend significant time developing software according to specific project requirements. Students are also responsible for verification and validation activities against the requirements, and for transitioning activities such as packaging, deployment scripts, performance and scalability testing, and product documentation. Students are managed and mentored by students completing the fourth semester of the sequence. The completion of this semester also marks the completion of the student's first project.

In the third semester, a student begins a new project by working on Requirements and Analysis. Under constraints structured by the course facilitator, students write product inception plans, develop requirements, and perform

requirements analysis resulting in an analysis model of the system. The analysis model serves as the input product for the next semester's development phase undertaken by the first-year participants.

In the fourth and final semester of the Enterprise sequence, a student serves as a process manager, test planner, and mentor. Fourth semester students manage the Construction and Transition activities of second semester participants. As process managers, these students are responsible for process planning, process monitoring, and change management. Process planning is the selection and instantiation of specific process models within the framework of the Rational Unified Process [3]. Process monitoring employs structured activity reporting, using reporting templates provided by the facilitator. The course facilitator may introduce pseudo-random events such as requirements changes, schedule changes, and resource changes that students must handle according to the instantiated process model.

As noted above, students in the second and fourth semesters of the sequence combine to form project teams. The meeting schedule in Spring 2005 allowed for separate one-hour lectures per section per week and a shared three and one-half hour lab. In the presence of the facilitator, lab sessions were used to conduct project status updates and also to work on "problems in the small", emphasizing lecture topics.

PRELIMINARY RESULTS

Three projects were completed in the 04-05 academic year.

- *Time Management for Software Engineers*. This project augments an open source Personal Information Management (PIM) tool with functionality to support Time Management and the Personal Software Process.
- *Digital Logic Design Tool*: This project extends the SIM environment for constructing digital logic design and simulation tools for teaching and learning.
- *Program Visualization Tool*: This project extends an animation tool for program visualization. The extensions allow for importing and animating design patterns.

Anecdotally, students communicated they felt the course was worthwhile, though the projects themselves were not as successful as was hoped. We attribute this to the emphasis on process over product, a concept that was difficult for students to accept. We did collect assessment data from affinity process studies, anonymous surveys, peer reviews, project postmortems, and the standard University course feedback process. We intend to publish our findings as soon as we are able to analyze all the data.

DISCUSSION

Our first-year experience gives us confidence the Enterprise approach is a viable method for capstone projects in software technology programs. The Enterprise has several defining features distinguishing it from typical project-based courses:

- Continuous – Projects are ongoing; students who enter the Enterprise will (most likely) work with a software product

line that already exists. They are asked to extend, port, modify, and/or maintain this software product line.

- Multi-semester – The Enterprise is designed as a four-semester sequence. This exposes students, in a specified order, to all phases and roles of the software lifecycle.
- Multi-project – Students are expected to work through two projects instead of one during the Enterprise sequence. Students to get exposed to process phases in the proper order, and do not get too "honed in" on a particular project, shortchanging process-related activities.
- "Real-world" – Students are exposed to the full spectrum of forces affecting software development projects. Teams are asked to cope not only with technical issues but also with social or soft-skill issues. For example, changing requirements, changing business models, changes in team membership, changes in project direction, and so on.
- Collaborative – Students work in teams, and also work across course and academic year boundaries. Students role-play, with participants responsible for different process-oriented roles. Teams also deal with outside roles, such as customers and CXX-level management.

It is worthwhile noting that our phases (and courses) are named according to the RUP process meta-model, though best practices from currently accepted methodologies are utilized throughout the sequence. Table 3 lists some of the process practices incorporated throughout the sequence.

SEM	PROCESS	EXAMPLE PRACTICES
1	PSP	Time Management, Defect Tracking
2	Agile	Test-driven dev., Refactoring
3	RUP, SADM, XP	Use case, DFD, STD, User Story
4	RUP, Spiral	Iterative/Incremental, Risk Mgmt., Win-Win

TABLE 3. PROCESS FAMILY UTILIZATION IN THE SOFTWARE ENTERPRISE

Space limitations preclude a full discussion and comparison to other capstone approaches in similar programs. The informed reader will recognize some unique characteristics with our approach, yet also recognize many characteristics adopted from other programs. As such, the Enterprise represents our contribution to the community's evolving understanding and practices for producing industry-ready software professionals.

ACKNOWLEDGMENT

This work is supported in part by a grant from the Arizona Board of Regents.

REFERENCES

- [1] Koehnemann, H, Gannod, B, "Experiences Using Student Project to Create University Business Applications", Proc. of the American Society for Engineering Education Conf., Salt Lake City, June 2004.
- [2] Humphrey, W.S., *Introduction to the Personal Software Process*, Addison-Wesley, Boston, 1997.
- [3] Krutchen, P., *The Rational Unified Process – An Introduction (2nd ed.)* Addison-Wesley, Boston, 2000.