A Graduate Education in Software Management and the Software Business for Mid-Career Professionals

Ray Bareiss and Gladys Mercier
Carnegie Mellon Silicon Valley
{ray.bareiss,gladys.mercier}@sv.cmu.edu

Abstract

Given the unique nature of the software business, the faculty of Carnegie Mellon University's Silicon Valley campus concluded that mid-career software professionals would be better served by a tailored master's degree focusing on software management and more broadly on the business of software than by a typical MBA. Our software management master's program integrates business, technical, and soft skills to prepare our students for technical leadership in their current companies or in entrepreneurial ventures. Our initial program built on the strengths of Carnegie Mellon's world-class software engineering education. We targeted students working in large companies, engaged in large-scale enterprise software projects, employing "high ceremony" software development processes. However, the majority of our students came from Silicon Valley companies which shared a product development focus, engaged in smaller projects, favored agile development processes, and measured development cycles in weeks rather than years. Our program has evolved to align with these interests. It employs a unique team-based and project-based pedagogy which emphasizes practical skills over theory, depth over breadth, and coaching over lecturing. High student satisfaction and growing enrollment have validated our curriculum decisions and have led us to make this program the educational centerpiece of Carnegie Mellon's Silicon Valley campus.

1. Introduction

U.S. business schools are experiencing a significant surge in MBA applications [1]. At the same time, business educators are questioning the MBA curriculum and its delivery, perhaps as never before [2]. Criticisms leveled at the typical MBA curriculum include the lack of practical relevance of much of what is taught, the lack of education in soft skills, and the lack of opportunities to practice leadership. There is a realization that more pedagogical innovation is needed, in particular, more opportunities for experiential learning should be provided, perhaps supervised by a "clinical faculty" of successful business practitioners.

Beginning in 2003, faculty of Carnegie Mellon's Silicon Valley campus -- then engaged primarily in graduate education in software engineering -- undertook a review of the appropriateness of MBA education for mid-career software professionals, and arrived at perhaps a more extreme conclusion than the Harvard Business School Report [2]: *Because of the unique challenges of managing software development, and indeed of the software business per se, a new master's program tailored to educating software managers would better suit their unique career advancement needs, providing a more specialized and technical alternative to an MBA.*

In reaching this conclusion, we considered these aspects of the software business:

- Software products and systems are complex and intangible

- Poor requirements elicitation and analysis has, historically, resulted in low quality products that don't satisfy customer needs
- Project estimation is not widely understood nor well implemented
- Product development tends to focus narrowly on software features rather than on complete solutions to customers' problems
- There is no consensus on the "best" development process for a given project
- Project progress and team productivity are difficult to measure and track
- Underlying technologies change rapidly, and the implications of new trends, such as open source software, are poorly understood
- Managers need a significant level of technical competence to make informed decisions and to command the respect of their employees
- Globally distributed teams are becoming the norm
- Many software professionals have entrepreneurial ambitions

In particular, we saw the need for a shared body of knowledge between software engineers and software managers, enabling the latter to understand and thus to effectively plan and manage the work of the former. We believed this common core to be technical: requirements engineering, software architecture, and project/program metrics. Software managers should also receive specialized education in strategic, program, process, and people management, while software engineers educated in our program should receive specialized education in the construction, testing, and deployment of software systems. Thus, we saw a software management program as a natural outgrowth of our highly regarded software engineering program.

Our initial strategy was to play to the strengths of Carnegie Mellon's software engineering education, focusing on "the development of business-critical, network-centric, software-intensive systems of systems (employing commercial off-the-shelf components, outsourcing, and internal development) requiring predictability and quality in their development and operation." Such systems are typically developed by companies employing relatively "high ceremony" development processes and measuring their organizational capabilities via the Capability-Maturity Models of Carnegie Mellon's Software Engineering Institute [3].

Our initial partners in launching this degree program were large companies, especially aerospace and defense companies. Their technical managers, who have typically risen through the ranks of the engineering organization, valued an approach to management education grounded in software engineering principles that would provide an evolutionary educational experience for their mid-career technical employees. There was also a bit of an anti-MBA bias, and at least one large company would only provide tuition assistance for employees continuing their technical educations. Our software management program, providing business and management courses specific to the software industry, was sufficiently technical to merit their support.

Surprisingly, in the beginning, only 2/5 of our students came from these large companies. The remainder came primarily from Silicon Valley companies possessing quite a different world view. These companies, and thus their employees, are product focused, generally undertake smaller projects, favor agile development methodologies over high-ceremony processes, and measure development cycles in weeks instead of

years. During the first few years of the program, we experienced a significant tension between the needs of these two constituencies. Enrollment trends resolved this tension.

The number of Silicon Valley professionals enrolling in our program increased, while the number of aerospace and defense professionals enrolling declined. To meet the needs of our changing student population, our program has evolved from teaching software management to teaching both software management and, more broadly, the business of software. During this evolution, we have retained the appropriate emphasis on technical skills. Our program now features a strong product development focus, incorporating agile development and entrepreneurship. Over time the tension has diminished. Our students from aerospace and other large companies increasingly share these interests and now have a history of taking agile development ideas back to their large projects, as well as engaging in "intrapreneurship" within their companies.

Although our program has evolved, the overarching educational mission has remained invariant: *to provide a transformative educational experience for our students*:

- Students gain facility at applying business and technical skills to authentic problems
- Students learn principled decision-making frameworks and how to instantiate such frameworks logically
- Students practice expressing their ideas clearly and persuasively in written communications and presentations to executives
- Students learn how to negotiate effectively and with authority
- Students become adept at working effectively in local and virtual teams -- the key soft skill in modern software development of any scale
- Students become reflective, self-aware practitioners and effective self-directed learners

Students participate in regular soft skills practice via subject area "threads" that are woven into the curriculum, assisting our students to become technical leaders, meriting the respect of their peers, subordinates and superiors.

2. The Software Management Curriculum

The Software Management master's program is a two-year, part-time program featuring flexible delivery to accommodate the working professional's busy schedule. On average thirty percent of the students participate remotely; many of these are employees of aerospace and defense companies. Local students are drawn in small numbers from a large range of Silicon Valley companies, including IBM, Hewlett-Packard, Oracle, Google, Microsoft, Sun Microsystems, Cisco, and others including entrepreneurial ventures. To encourage a strong sense of community, all students, whether remote or local, are required to come to campus for a three-day orientation, one mid-program weekend event, and graduation.

Initially, our course sequence matched the temporal business progression of software product development: product visioning and business strategy, requirements engineering, software architecture, and finally management of in-house or outsourced development. But the reality is that a typical software professional's career progresses in the opposite direction. In 2006, we resequenced the curriculum so that coursework

begins nearer to where an incoming student's knowledge and experience typically ends. Thus, student skills are built incrementally, starting with an introduction to thinking about software as a business, proceeding through basic people and process management (including metrics), and concluding with advanced product and business strategy courses. We also changed the type of projects our courses emphasized from large enterprise projects to smaller, more agile projects typical of Silicon Valley companies.

The current curriculum comprises six semesters of project-based courses, each providing key practical knowledge and skills, as well as new opportunities for experiential learning. All courses, with the possible exception of summer electives, are taken one at a time, thus dividing each semester into two intensive "mini semesters" of six or seven weeks each. Students have a one week break between "minis" and a longer break between semesters.

Table 1 - Software Management Curriculum

	First Year	**Second Year**
Fall	Elements of Software Management Metrics for Software Managers	Software Product Definition Requirements Analysis
Spring	Project and Process Management Managing Software Professionals	Software Product Strategy The Business of Software
Summer	Electives (1 or 2) including: Managing Outsourced Development Avoiding Software Project Failures Open Source Software Human-Computer Interaction Software Architecture	Electives (1 or 2) including: Innovation and Entrepreneurship Entrepreneurial Finance Product Management Corporate Strategy Practicum Project

The first year of our curriculum can be broadly characterized as focusing on *the management of software development*, while the second year can be characterized as focusing on *the business of software*.

We believe that each course in our curriculum offers a balance of theory coupled with ample opportunities to apply the practical knowledge and skills acquired. In designing the curriculum, we have made a conscious decision to emphasize depth and application rather than breadth, cf. [4]. We are also aware that, given different individual learning objectives and different project experiences, different students will cover various topics in more or less depth, focusing on those areas in which they most need or want to gain mastery, which is appropriate for a professional master's program.

2.1 First year courses

The first course, *Elements of Software Management*, is designed to take our typically very technical students out of their comfort zone to begin the transition to thinking about software as a business. Each student is assigned a public company to study for the duration of this course. The student learns to apply standard frameworks for strategic market analysis, for analyzing business strategies and execution processes, and for analyzing financial statements, all in the context of his or her assigned company. The course concludes with a presentation of the student's analysis, accompanied by a two-

year prognosis for the company (which the student is encouraged to revisit at the end of the program).

After the first course, which calls primarily for individual work (and allows for in-depth assessment of individual students) learning takes place in the context of Story-Centered Curricula [5]. Working primarily in teams, students act as employees of a fictional company engaged in a range of projects. Within this framework, and with coaching support from faculty members, students confront realistic technical and business problems, including conflicting requirements, unclear product and business strategies, limited resources, and challenges of team leadership.

The second course, *Metrics for Software Managers*, equips students with a range of tools to answer the question "how are we doing?" at both project and program levels. In realistic scenarios, students learn a range of traditional and agile metrics for tracking progress, productivity, quality, and cost; how to analyze metrics data; and how to report them up and down the organization. They also learn how such data provides a sound basis for improving estimation. Finally, they learn how to roll out a metrics initiative and use change management strategies to maximize its chances of success.

The third course, *Project and Process Management*, provides in-depth exposure to software development methodologies and their application to projects with particular characteristics. First, student teams research a range of component software development and supporting processes (e.g. project management, requirements analysis, architecture, construction, testing, quality assurance) to learn how each is typically implemented in the contexts of traditional and agile development methodologies. The result is a student-authored "encyclopedia" of process definitions which the students then use to tailor a development methodology for a project which is not optimally served by either a traditional or agile approach. Finally, students must respond to a number of classic management challenges as the project "runs" in simulation.

The fourth course, *Managing Software Professionals*, equips students with current theories and best practices to hire appropriately, to provide a work environment which attracts and retains top performers, to evaluate performance, and to deal with situations in which workers must be laid off or fired. During this course, they practice a number of key skills via simulation (e.g. hiring interviews and performance evaluations.)

The four courses above also serve as the second year curriculum for nearly half of our software engineering students who are enrolled in a development management (SE-DM) track. The SE-DM track is designed for software engineers interested in taking a first step towards management, becoming technical project leaders and software development managers. (SE-DM students take core technical software engineering courses during their first year.)

The first summer's electives provide a range of business, management and technical alternatives. Two merit brief discussion: *Avoiding Software Project Failures* is a case study course in which students analyze the causes of significant real-world project failures and explore how such failures might be avoided. *Managing Outsourced Development* revisits the focal project from *Project and Process Management*; students have the comparative experience of outsourcing the project (including solicitation and vendor selection) and then managing the outsourcing relationship via simulation.

2.2 Second year courses

The first four courses of the second year collectively provide perhaps the most uniquely valuable educational experience of the program, especially for our many students interested in product management and entrepreneurial ventures. Students begin with the germ of a faculty-supplied idea for a software product and develop it into a well-researched plan for building a software product business through four courses.

The first course, *Software Product Definition*, guides students to apply modern methods of requirements elicitation to deeply understand the problems faced by potential users and business stakeholders of a new product. This deep understanding feeds an analysis and high-level design process in which students develop a compelling, realistic vision of how a proposed software system and accompanying "whole product components" (in the form of complementary systems, services, and technology) will address the identified problems, resulting in a complete solution.

The second course, *Requirements Analysis*, guides students to further elicit, analyze, and document functional, technical, and business requirements for the new product. as part of this process, they perform an analysis of competing products already in the market in light of those requirements. Finally, they define a product roadmap, embodying a strategy for developing and releasing the functionality of the new product in coherent increments.

The pair of courses, *Software Product Definition* and *Requirements Analysis*, provides what we believe are the key technical skills needed by software managers. This educational emphasis on product requirements is intended to reduce, if not eliminate, the high incidence of requirements-based project failures which plague the industry. [6].

The third course, *Software Product Strategy*, guides students to move the product idea forward by analyzing the macro environment, the company's capabilities, market opportunities, competition, and other related factors, culminating in recommendations regarding whether and how to proceed with product development. These are summarized in a "high stakes" presentation to faculty posing as management of the simulated company considering the project which culminates in a go/no-go recommendation and asks for management buy-in.

The fourth course, *The Business of Software*, concludes the product development sequence by guiding students in formulating a business model for the software product, revenue and cost estimates, and a development plan. After assessing the viability of the project, students make a final presentation in which they recommend if and how the company should proceed.

After completing this sequence, the majority of students complete their studies by taking the *Innovation and Entrepreneurship* elective, in which they self-organize into teams to develop, plan, and "pitch" their own ideas for technology start-ups to the faculty and often to Silicon Valley venture capitalists. In parallel, most take *Entrepreneurial Finance*, which deals with financing options for start-ups. Alternatively, some students opt for a *Practicum Project* in which a team applies their knowledge and skills to an actual industrial project, with a real client, coached by a faculty member.

2.3 Future Curricular Directions

As enrollment in the software management program grows, it becomes both feasible and desirable to offer multiple tracks. Two are currently planned: The first is planned to serve entrepreneurial students starting their first (or a subsequent) software company and those who are hopeful of doing so but haven't yet made the commitment. Candidates will submit original product or service ideas for evaluation by a panel of faculty and industry professionals, and students admitted to this software entrepreneurship track will form teams around the most promising proposals. During the second year, they will complete course assignments in the context of a real-world project in place of the simulated project provided for the second year course sequence. The second track is planned to serve mid- to senior-level managers working in large enterprises who wish to accelerate their already successful corporate careers. These self-identified students will have the option to take an alternate sequence of specialized courses and a research project focusing on the challenges of software innovation and management in a large company context.

3. Principles of Instruction at Carnegie Mellon Silicon Valley

Our pedagogical approach is informed by cognitive science research on how people think and learn (e.g., [7]). As noted earlier, nearly all of our courses are Story-Centered Curricula [5] in which the bulk of student learning occurs in the context of team-based work on realistic but simulated projects [8]. When a project is assigned, student teams are expected to plan their own work *and the learning of the new knowledge and skills which are required to succeed*. Faculty primarily teach by coaching at the team level, sometimes providing direct guidance, but more often modeling problem solving techniques and asking open-ended questions to lead the students to discover relevant knowledge and to solve problems themselves [9]. Due to student demand, all courses also have a weekly "seminar session," involving the entire class, in which faculty facilitate discussions of readings and issues of general interest.

We have implemented the "clinical faculty" advocated by the Harvard report [2]. All of our faculty have significant management experience within large companies and/or entrepreneurial ventures as well as traditional academic credentials and, thus, have the real-world credibility resulting from having "been there and done it" that our mid-career professional student body demands. We have also engaged local talent from the Silicon Valley -- in the form of serial entrepreneurs, corporate executives, venture capitalists, HR directors, and attorneys -- to be guest speakers in many of our courses.

4. Current Status and Outcomes

Carnegie Mellon Silicon Valley has graduated 125 students with an MS in software management; 79 students are currently enrolled. We have also graduated 236 students with an MS in software engineering. For the past three years, we have surveyed all Carnegie Mellon Silicon Valley alumni to ascertain the career value they attribute to their graduate education. (No comparison data for other programs are available.) In September 2009, 47 of 125 Software Management alumni completed the survey.

Table 2 - Results (as a percentage of survey respondents)

Believe the program gave them a competitive advantage relative to peers	93%
Promoted - during the program	45%

- after graduation	74%
Changed jobs either within their company or at a new company	69%
Salary - increased more than 40% - increased 21-40% - increased 11-20% - increased less than 10%	 10% 24% 38% 28%
Included one or more soft skills among the top three things they learned	74%
Would recommend Carnegie Mellon Silicon Valley to a friend or colleague	97%

Rather than ending with dry statistics, let's allow an alumna to speak for herself:

Carnegie Mellon's software management program has greatly helped transform my career and boost my personal confidence. Today, I'm a product manager at a leading enterprise software company in the Bay Area. Thanks to the hands-on program curriculum, I was able to transition from a technical lead role into a more outward facing business role. I find myself well prepared to execute on the job with the required skills and knowledge. I feel that this program was the best investment I've made -- one that has helped me reap multiple dividends in a very short period of time.

— Silicon Valley MS-SM graduate

6. Acknowledgments

We would like to thank the past and present faculty of the software management program, especially Lynn Carter, Patricia Collins, Stuart Evans, Martin Griss, Martin Radley, and Tony Wasserman. In particular, Tony Wasserman made a large contribution to the evolution of our curriculum from a large-company, program management focus to a small-company, entrepreneurial focus.

7. References

[1] A. Damast, "MBA Applications Surge Again," Business Week, August 27, 2008. (http://www.businessweek.com/bschools/content/aug2008/bs20080826_158181.htm?campaign_id=rss_null)

[2] Staff, HBS Alumni Bulletin, "Harvard Business School Discusses Future of the MBA," November 24, 2008. (http://hbswk.hbs.edu/item/6053.html).

[3] B. Boehm and R. Turner, Appendix C: Capability Maturity Models, Balancing Agility and Discipline: A Guide for the Perplexed, Addison Wesley, Boston, MA, 2005.

[4] M. Shaw(editor), Software Engineering for the 21st Century: A basis for rethinking the curriculum, Technical Report CMU-ISRI-05-108, Carnegie Mellon University, Institute for Software Research International, Pittsburgh, PA, 2005.

[5] R.C. Schank, Making Minds Less Well Educated than Our Own, Lawrence Erlbaum Associates, Mahwah, NJ, 2004.

[6] Anonymous, *The Chaos Report*. The Standish Group, 1995, 2005. (The 1995 version is available at http://www3.uta.edu/faculty/reyes/teaching/general_presentations/chaos1994.pdf .)

[7] Bransford, J. D., Brown, A. L., and Cocking, R. R. (Eds.). *How People Learn: Brain, Mind, Experience, and School*. National Academy Press, Washington, DC, 2000.

[8] R. Bareiss and M. Griss, "A Story-Centered, Learn-by-Doing Approach to Software Engineering Education," Proceedings of SIGCSE 2008, Portland, OR, March 2008.

[9] R. Bareiss and M. Radley, "Coaching Via Cognitive Apprenticeship," submitted to SIGCSE 2010, Milwaukee, WI, March 2010.