Integrated Software and Systems Engineering Curriculum

Second Workshop of the Early Start Team

December 11th & 12th, 2007

Arlington, Virginia, USA

WORKSHOP REPORT
Contents

1. iSSEc Project ........................................................................................................... 1
2. Second Workshop .................................................................................................... 1
3. Workshop Proceedings ............................................................................................ 3
   3.1 Welcome & Introduction ..................................................................................... 3
   3.2 As-Is Analysis ..................................................................................................... 3
   3.3 Task Group presentations .................................................................................. 4
      3.3.1 Guidance and Outcomes Group ................................................................. 4
   3.4 Architecture ........................................................................................................ 4
   3.5 Body of Knowledge ............................................................................................ 5
   3.6 Strawman paper .................................................................................................. Error! Bookmark not defined.
   3.7 General Points ................................................................................................... 6
   3.8 Sponsors’ Views: ................................................................................................. 6
4. Way Ahead .............................................................................................................. 7

Appendix A: Workshop Objectives .............................................................................. 8
Appendix B: Workshop Agenda ..................................................................................... 9
Appendix C: “iSSEc Project”, Art Pyster ....................................................................... 10
Appendix D: “As-Is Analysis”, Art Pyster .................................................................... 15
Appendix E: “Guidance and Outcomes”, Art Pyster ...................................................... 29
Appendix F: “Architecture Document” ........................................................................ 35
Appendix G: “Architecture”, Mary Shaw ........................................................................ 38
Appendix H: “Body of Knowledge”, Tom Hilburn ......................................................... 42
1. iSSEc Project

Software engineering (SwE) is the acknowledged discipline by which large-scale and complex software is developed. Many universities teach software engineering at the undergraduate level. More than 30 colleges and universities helped create the 2004 model curriculum for undergraduate SwE education that was published by the IEEE and ACM.

Many universities offer a Masters degree in SwE. Yet the only existent model curriculum for graduate SwE was created in 1991 by the Software Engineering Institute. Since then the World Wide Web has vastly changed how the world communicates, and software being developed today has changed considerably as a result. Considering the reliance of the world economy on the quality of senior SwE professionals, the lack of a current model graduate curriculum is dismaying.

The iSSEc (integrated Software and Systems Engineering curriculum) Project is creating a new model graduate SwE curriculum that reflects new understandings in how to build software, how software engineering depends on systems engineering, and how software engineering education is influenced by individual domains, such as telecommunications and defense systems. The resulting curriculum will be suitable for a university education leading to a Masters Degree in SwE.

As per the initial iSSEc project plan, an Early Start Team (EST) was formed and an organizing workshop was conducted in August 2007 to bring the members together and to kick start the project activities. Following the workshop, members of the EST began working on an initial “Strawman” version of the reference curriculum. A second workshop was conducted in December 2007 to review the current status and to plan future activities. This report presents the proceedings of the second workshop.

2. Second Workshop

The second workshop of the EST was held on Dec 11-12, 2007 at the ITS corporation in Arlington, Virginia courtesy the National Coordinating Office for NITRD. The primary workshop objectives were to review the current status of activities leading to release of the Strawman Curriculum, make mid course corrections where required and to plan further activities and plan for future activities. Workshop objectives are included in Appendix A.

More members were inducted into the EST since the August 2007 workshop. The EST now includes members from industry, academia, and government and professional
societies, as well as volunteers from the IEEE, ACM and SWEBOK. International participation however, is still limited.

As on December 11th, 2007 the Early Start Team consisted of:

1. Edward Alef, *General Motors*
2. Bruce Amato, *Department of Defense*
3. Mark Ardis, *Rochester Institute of Technology*
4. Barry Boehm, *University of Southern California*
5. Pierre Bourque, *Quebec University and SWEBOK volunteer*
6. John Bracket, *Boston University*
7. Murray Cantor, *IBM*
8. Lillian Cassel, *Villanova and ACM volunteer*
9. Robert Edson, *ANSER*
10. Dennis Frailey, *Raytheon & Southern Methodist University*
11. Gary Hafen, *Lockheed Martin*
12. Thomas Hilburn, *Embry-Riddle Aeronautical University*
13. Greg Hislop, *Drexel University and IEEE volunteer*
14. Philippe Kruchten, *University of British Columbia*
15. James McDonald, *Monmouth University*
16. Ernest McDuffie, *National Coordination Office for NITRD*
17. Bret Michael, *Naval Postgraduate School*
18. William Milam, *Ford*
19. Fernando Naveda, *RIT and IEEE volunteer*
20. Ken Nidiffer, *SEI*
21. Paul Robitaille, *INCOSE, Lockheed Martin*
22. Doug Schmidt, *Vanderbilt*
23. Mary Shaw, *Carnegie Mellon University*
25. Robert Suritis, *IBM*
26. Richard Thayer, *California State University at Sacramento*
27. Joseph Urban, *National Science Foundation*
28. Osmo Vikman, Nokia, Finland
29. David Weiss, Avaya *
30. Art Pyster, Stevens Institute of Technology
31. Larry Bernstein, Stevens Institute of Technology *
32. Richard Turner, Stevens Institute of Technology
33. Sarah Sheard, Stevens Institute of Technology
34. Devanandham Henry, Stevens Institute of Technology
35. Kahina Lasfer, Stevens Institute of Technology

* Members unable to attend the workshop

Of the above, Mary Shaw joined just for the first day while James McDonald and Lillian Cassel joined the workshop on its second day.

3. Workshop Proceedings

3.1 Welcome & Introduction

The two day workshop (agenda attached as Appendix B) began with a welcome by Dr. Art Pyster, principal investigator of the iSSEc project. He presented the workshop objectives, roadmap and roles/timelines for release of the Strawman Curriculum, as decided during the organizing workshop in August 2007 (attached as Appendix C).

3.2 As-Is Analysis

Preliminary analysis of 11 graduate software engineering programs was presented during the August 2007 workshop. Since then, more programs and further analysis has been conducted. Art Pyster repeated a presentation originally made to the project sponsors on Nov 30, 2007 (attached as Appendix D).

Analysis of graduate software engineering programs from 28 schools including 3 non-US schools was presented. Observations from the analysis brought out the current state of graduate software engineering education, in addition to the great diversity in program focus and courses taught among many others. The analysis clearly articulated the need for a model reference curriculum that would help better define the knowledge and skills that a typical graduate of a software engineering program would possess.

Possible further analysis of already collected data and collection of additional data was also presented. While there now exists a reasonably good coverage of US schools,
more non-US schools need to be studied to make the analysis and observations valid internationally.

Some comments and additional analysis suggested by the EST members are summarized below:

- The analysis shows Software Engineering as a discipline in the process of maturation, similar to the state of the Computer Science discipline in the 1960s.

- The relative size and role of adjunct faculty, and their influence on the program, courses offered, etc. needs to be better understood.

- Analyzing the required courses separate from the semi-required courses would be useful.

- Correlation between SWEBOK coverage and the size of the programs could be studied.

- The relative size and influence of international students in the software engineering program could be studied.

3.3 Task Group presentations

Three of the four task groups made presentations on the work done since the August 2007 workshop, current state and work to be done before finalizing their contribution for the Strawman Curriculum. The “Course Packaging” group was the only group that did not make a formal presentation. This group’s work depended on outputs from other working groups, and had had their first meeting only a week before the second workshop. Hence there was not much to report by that group.

3.3.1 Guidance and Outcomes Group

Art Pyster presented version 0.5 of the Guidance and Outcomes document that was a result of discussions within the Group (attached as Appendix E). The EST was generally in agreement with the contents of the document.

The task of the group was also to define the entrance requirements for the curriculum. This was decided based on discussions and voting by all participants of the workshop (reported in section 3.4).

3.3.2 Architecture Group

In the absence of the team leader Jim McDonald, Mary Shaw presented version 1.5 of the Architecture document and a graphical representation of the proposed architecture (attached as Appendices F & G). Examples of how the Body of Knowledge could be presented and courses could be packaged was also shown graphically.
The members in general were not clear on the definition of two types of “Core” material and the “university specific” material in addition to electives. The architecture group would work further based on feedback from the workshop to finalize the proposed architecture.

3.3.3 Body of Knowledge

Thomas Hilburn presented the work done by the Body of Knowledge group beginning with the SWEBOK (attached as Appendix H).

SWEBOK is a good starting point but it does not reflect recent trends such as open source, agile development, and geographically distributed development. Thomas Hilburn discussed the spreadsheet where members of the BOK group individually indicated the coverage in terms of contact hours.

The BOK group would go through one or two more rounds of refining the list of competencies and their coverage before handing it over to the course packaging team.

3.4 Entrance Requirements / Curriculum Baseline

After much discussion and individual voting, the workshop participants decided on the following criteria as entrance requirements or baseline for the reference curriculum:

- **Background:** Equivalent of BS in CS +1 to 2 years of relevant experience
- **Duration:** 10 courses + Capstone
- **Capstone experience:** Required
- **Core courses:** 4 out of 10 is desirable, but it is understood that 5 courses may be needed

The guidance and outcomes group would elaborate the specific competencies (knowledge & skills) that would characterize the BS in CS and 2 years of relevant experience. This would help universities to design ramp courses for students who may fully satisfy this requirement.

The course packaging group will package the core material, as defined by the Body of Knowledge group, into 4 or 5 courses as an exemplar. If time permits, the course packaging group will create a second exemplar for Version 0.25 of the curriculum.

3.5 First version of the Reference Curriculum

Pierre Bourque expressed concern over use of the term “Strawman Curriculum” and explained how this would not be understood internationally. He strongly
recommended use of version numbers as against typically American terms like “strawman”. The “strawman curriculum” was therefore renamed as “initial version”.

Richard Turner went through various sections of the initial version and the people who would contribute. The four main sections will be provided by each of the task groups. The Stevens team will take the lead in writing other portions of the document as well as integrating the whole document.

### 3.6 General Points

Some general points raised and discussed by the participants of the workshop not included in the preceding sections, are mentioned below:

- The essential “Systems Engineering” knowledge that every software engineer must know, should be explicitly detailed in the Body of Knowledge. This should be expanded further as the integrated software and systems engineering curriculum is developed.

- Definitions of “Software Engineering” and “Systems Engineering” should be included in the curriculum.

- While the curriculum being developed is more of a “professional” one preparing a student for the industry, it must allow for students to work on a PhD if they so desire.

- Examples of actual courses and programs could be included as possible course packaging options.

- The Body of Knowledge should be traceable back to the guidance and outcomes document and vice-versa.

- While it would depend on the individual schools, the curriculum efforts could consider the economic implications of adopting the reference curriculum.

### 3.7 Sponsors’ Views:

The sponsors of the project actively participated in the workshop. Kristen Baldwin, Deputy Director, Software Engineering and System Assurance was present during the first and last sessions of the workshop; Bruce Amato, who is Kristen Baldwin’s deputy, attended both days of the workshop; and Mark Schaeffer, Director of Systems and Software Engineering joined for the concluding session of the workshop. Between them, the motivation and commitment of the DoD to the iSSEc project was clearly articulated. It was clarified to all EST members that DoD views the graduate reference curriculum as a contribution to the worldwide software engineering community. It was made clear that specific needs or interests of the DoD would not be imposed on the curriculum.
4. Way Ahead

The following milestones were discussed and decided for release of the initial version of the reference curriculum:

- **end Jan 08** - Receive sections from each of the task groups
- **mid Feb 08** - Mini-workshop with leaders of the task groups, Stevens team and other EST members who wish to join to review the first draft of the curriculum document
- **end Feb 08** - Release of curriculum version 0.25 to a limited group of reviewers (US and international) outside the EST
- **end Mar 08** - Receive feedback from all reviewers
- **Apr 08** - 3rd workshop of EST to review feedback and plan action
- **July 08** - Release curriculum version 0.5 to a large international audience across academia and industry sectors for further review

Journal and conference papers are being prepared based on the As-Is analysis. Presentations on the As-Is analysis as well as the curriculum efforts are planned during various software engineering conferences. EST members were requested to suggest conferences where the curriculum efforts could be presented and discussed. They were also encouraged to make presentations on the iSSEc project in conferences they would be attending.

Efforts will continue to obtain endorsement from the IEEE and ACM concurrent with publication of Version 0.5 of the curriculum.
Appendix A: Workshop Objectives

1. To review the work-in-progress on the Strawman Curriculum.

2. To make any mid-course corrections in our plan to release the Strawman curriculum by February 28, 2008

3. To agree on steps to be taken after February 28, 2008 as the larger Core Team is formed
Appendix B: Workshop Agenda

Day 1: December 11th, 2007

The first day focuses on Objectives 1-6, and there may be some holes in satisfying those Objectives that will be closed on the second day.

<table>
<thead>
<tr>
<th>Time</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:00a</td>
<td>Continental Breakfast</td>
</tr>
<tr>
<td>8:30a</td>
<td>Introductions - Art Pyster, Kristin Baldwin</td>
</tr>
<tr>
<td>8:45a</td>
<td>Overview of project since last workshop - Art Pyster</td>
</tr>
<tr>
<td>9:30a</td>
<td>Recap last workshop and refresh people on project purpose and assumptions – Art Pyster</td>
</tr>
<tr>
<td>10:15a</td>
<td>Break</td>
</tr>
<tr>
<td>10:30a</td>
<td>Summary of four group results - Art Pyster, Mary Shaw, Tom Hilburn, Bret Michael</td>
</tr>
<tr>
<td>11:15a</td>
<td>Report and discussion on guidance and outcomes - Art Pyster</td>
</tr>
<tr>
<td>12:15p</td>
<td>Lunch</td>
</tr>
<tr>
<td>1:00p</td>
<td>Report and discussion on Body of Knowledge - Tom Hilburn</td>
</tr>
<tr>
<td>3:00p</td>
<td>Break</td>
</tr>
<tr>
<td>3:15p</td>
<td>Report and discussion on architecture - Mary Shaw</td>
</tr>
<tr>
<td>5:00p</td>
<td>Adjourn for the day</td>
</tr>
<tr>
<td>5:30p</td>
<td>Group supper for those interested</td>
</tr>
</tbody>
</table>

Day 2: December 12th, 2007

The second day is largely focused around satisfying Objectives 5-9, although the conversation may revisit some of Objectives 1-6 that were the primary focus of the previous day.

<table>
<thead>
<tr>
<th>Time</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:00a</td>
<td>Continental breakfast</td>
</tr>
<tr>
<td>8:30a</td>
<td>Recap from first day - Art Pyster</td>
</tr>
<tr>
<td>8:45a</td>
<td>Report and discussion on group’s assumptions – Art Pyster</td>
</tr>
<tr>
<td>9:45a</td>
<td>Plan the remaining material in strawman curriculum - Rich Turner</td>
</tr>
<tr>
<td>10:15a</td>
<td>Break</td>
</tr>
<tr>
<td>10:30a</td>
<td>Continue planning remaining material - Rich Turner</td>
</tr>
<tr>
<td>11:00a</td>
<td>Plan the Core Team workshop - Art Pyster</td>
</tr>
<tr>
<td>11:30a</td>
<td>Decide and mid-course corrections in plan to produce and release strawman curriculum and others we need to embrace in the near term - Art Pyster</td>
</tr>
<tr>
<td>12:15p</td>
<td>Lunch</td>
</tr>
<tr>
<td>1:00p</td>
<td>Recap, action items and next steps - Art Pyster</td>
</tr>
<tr>
<td>2:00p</td>
<td>Adjourn workshop</td>
</tr>
</tbody>
</table>
# Appendix C: “iSSEc Project”, Art Pyster

## iSSEc

**Integrated Software and Systems Engineering Curriculum**

## Early Start Team Workshop #2

Dec 11th & 12th, 2007, Arlington – Virginia, USA

### EST Members

<table>
<thead>
<tr>
<th>No.</th>
<th>Member Name</th>
<th>Affiliation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Edward Alef, GM</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>Bruce Amato, Department of Defense</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Mark Ardis, Rochester Institute of Technology</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Barry Boehm, University of Southern California</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>Pierre Bourque, Quebec University; IEEE</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>John Brackett, Boston University</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>Murray Cantor, IBM</td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>Boots Cassel, ACM; Villanova University</td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>Robert Edson, ANSER</td>
<td></td>
</tr>
<tr>
<td>10.</td>
<td>Dennis Fraley, Raytheon; Southern Methodist University</td>
<td></td>
</tr>
<tr>
<td>11.</td>
<td>Gary Hafen, Lockheed Martin</td>
<td></td>
</tr>
<tr>
<td>12.</td>
<td>Thomas Hillburn, Embry-Riddle Aeronautical University</td>
<td></td>
</tr>
<tr>
<td>13.</td>
<td>Greg Hislop, Drexel University; IEEE</td>
<td></td>
</tr>
<tr>
<td>14.</td>
<td>Philippe Kruchten, University of British Columbia</td>
<td></td>
</tr>
<tr>
<td>15.</td>
<td>James McDonald, Monmouth University</td>
<td></td>
</tr>
<tr>
<td>16.</td>
<td>Ernest McDuffie, National Coordination Office for NITRD</td>
<td></td>
</tr>
<tr>
<td>17.</td>
<td>Bret Michael, Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td>18.</td>
<td>William Milam, Ford</td>
<td></td>
</tr>
<tr>
<td>19.</td>
<td>Fernando Naveda, RIT, IEEE</td>
<td></td>
</tr>
<tr>
<td>20.</td>
<td>Ken Nidiffer, SEI</td>
<td></td>
</tr>
<tr>
<td>21.</td>
<td>Paul Robitaille, INCOSE: Lockheed Martin</td>
<td></td>
</tr>
<tr>
<td>22.</td>
<td>Douglas Schmidt, Vanderbilt</td>
<td></td>
</tr>
<tr>
<td>23.</td>
<td>Mary Shaw, Carnegie Mellon University</td>
<td></td>
</tr>
<tr>
<td>24.</td>
<td>Ann E Sobel, Miami University; IEEE</td>
<td></td>
</tr>
<tr>
<td>25.</td>
<td>Robert Suritis, IBM</td>
<td></td>
</tr>
<tr>
<td>26.</td>
<td>Richard Thayer, California State University at Sacramento</td>
<td></td>
</tr>
<tr>
<td>27.</td>
<td>Joseph Urban (observer), NSF</td>
<td></td>
</tr>
<tr>
<td>28.</td>
<td>Osno Vikman, Nokia, Finland</td>
<td></td>
</tr>
<tr>
<td>29.</td>
<td>David Weiss, Avaya</td>
<td></td>
</tr>
<tr>
<td>30.</td>
<td>Art Pyster, Stevens Institute of Technology</td>
<td></td>
</tr>
<tr>
<td>31.</td>
<td>Larry Bernstein, Stevens Institute of Technology</td>
<td></td>
</tr>
<tr>
<td>32.</td>
<td>Richard Turner, Stevens Institute of Technology</td>
<td></td>
</tr>
<tr>
<td>33.</td>
<td>Sarah Sheard, Stevens Institute of Technology</td>
<td></td>
</tr>
<tr>
<td>34.</td>
<td>Deva Henry, Stevens Institute of Technology</td>
<td></td>
</tr>
<tr>
<td>35.</td>
<td>Kahina Lasfer, Stevens Institute of Technology</td>
<td></td>
</tr>
</tbody>
</table>
Workshop Objectives

- Review the work-in-progress on the Strawman Curriculum
- Make any mid-course corrections in our plan to release the Strawman curriculum by February 29, 2008
- Agree on steps to be taken after February 29, 2008 as the larger Core Team is formed

Agenda (Dec 11th)

8:30a Introductions - Art Pyster
8:45a Overview of progress since last workshop - Art Pyster
9:30a Recap last workshop and refresh people on project purpose and assumptions - Rich Turner
10:15a Break
10:30a Summary of four group results - Art Pyster, Mary Shaw, Tom Hilburn, Bret Michael
11:15a Report and discussion on guidance and outcomes - Art Pyster
12:15p Lunch
1:00p Report and discussion on BOK - Tom Hilburn
3:00p Break
3:15p Report and discussion on architecture - Mary Shaw
5:00p Adjourn for the day
5:30p Group supper for those interested
**Agenda (Dec 12th)**

8:30a  Recap from first day - *Art Pyster*
8:45a  Report and discussion on *course packaging* - *Bret Michael*
9:45a  Plan the remaining material in strawman - *Rich Turner*
10:15a  *Break*
10:30a  Continue planning remaining material - *Rich Turner*
11:00a  Plan the Core Team workshop - *Art Pyster*
11:30a  Decide any mid-course corrections in plan to produce Strawman Curriculum and plan release and others we need to embrace in the near-term - *Art Pyster*
12:15p  *Lunch*
1:00p  Recap, Action Items, and Next Steps - *Art Pyster*
2:00p  Adjourn

---

**Creating the Strawman Curriculum**

1. Held workshop on August 15-16 at Applied Systems Thinking Institute
2. Reviewed foundational documents: SEI graduate curriculum reports from 1991, SWEBOK, SE2004, INCOSE SE Model Graduate Curriculum
3. Agreed to create strawman curriculum and agreed on outline of document
4. Divided into 4 primary teams with leads from 4 different universities
   - Guidance and Outcomes - Art Pyster, Stevens Institute
   - Curriculum Architecture - Jim MacDonald, Monmouth
   - Body of Knowledge - Tom Hilburn, Embry-Riddle
   - Course Packaging - Brett Michael, Naval Postgraduate School
5. Agreed to work in parallel where possible to speed delivery
Creating the Strawman Curriculum

6. Build Guidance and Outcomes as deltas from SE2004 Principles (Draft 1 done)
7. Build Architecture starting with 1991 SEI curriculum architecture (Draft 1 under review)
8. Build Body of Knowledge as deltas from SWEBOK using INCOSE Handbook, PMI BOK, and current state of SWE graduate programs as primary sources for additions (Draft 1 begun)
9. Build Course Packaging after first three teams have solid drafts
10. Hold second workshop in December to review progress
11. Refine drafts and publish at end of February

Working Groups

Outcomes
Art Pyster
Murray Cantor
Gary Hafen
Thomas Hilburn
William Milam
Bret Michael
David Weiss
Mary Shaw

Architecture
Jim McDonald
Gary Hafen
Ken Nidiffer
Sarah Sheard
Bret Michael
Mary Shaw

BOK
Thomas Hilburn
Mark Ardis
Bret Michael
William Milam
Richard Turner
Richard Thayer
Boots Cassel
John Brackett

Packaging
Bret Michael
Larry Bernstein
Ken Nidiffer
Doug Schmidt
Dennis Frailey
Mantak Shing
Outline of Strawman Curriculum

Size estimate 25-50 pp

<table>
<thead>
<tr>
<th>Section</th>
<th>Page count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Executive Summary</td>
<td>1</td>
</tr>
<tr>
<td>Introduction (Addressable</td>
<td>3</td>
</tr>
<tr>
<td>markets, spectrum of</td>
<td></td>
</tr>
<tr>
<td>degrees, project rationale)</td>
<td></td>
</tr>
<tr>
<td>Student capabilities</td>
<td></td>
</tr>
<tr>
<td>- Masters program</td>
<td>2</td>
</tr>
<tr>
<td>entrance requirements</td>
<td></td>
</tr>
<tr>
<td>(expected knowledge and</td>
<td></td>
</tr>
<tr>
<td>skills)</td>
<td></td>
</tr>
<tr>
<td>- Masters graduate</td>
<td>5</td>
</tr>
<tr>
<td>Capabilities (expected</td>
<td></td>
</tr>
<tr>
<td>knowledge and skills)</td>
<td></td>
</tr>
<tr>
<td>Curriculum</td>
<td></td>
</tr>
<tr>
<td>- Requirements (for</td>
<td>1</td>
</tr>
<tr>
<td>curriculum “ilities”)</td>
<td></td>
</tr>
<tr>
<td>- Body of Knowledge</td>
<td>10</td>
</tr>
<tr>
<td>(Deltas + and - to</td>
<td></td>
</tr>
<tr>
<td>SWEBOK)</td>
<td></td>
</tr>
<tr>
<td>- Curriculum Architecture</td>
<td>3</td>
</tr>
<tr>
<td>(to meet requirements)</td>
<td></td>
</tr>
<tr>
<td>- Course Packagings</td>
<td>10</td>
</tr>
<tr>
<td>(order, content, texts,</td>
<td></td>
</tr>
<tr>
<td>readings, radical</td>
<td></td>
</tr>
<tr>
<td>packaging)</td>
<td></td>
</tr>
<tr>
<td>(alternative packagings,</td>
<td></td>
</tr>
<tr>
<td>such as integrative vs.</td>
<td></td>
</tr>
<tr>
<td>discrete)</td>
<td></td>
</tr>
<tr>
<td>Discussion of Teaching</td>
<td>2</td>
</tr>
<tr>
<td>Methods (philosophical?)</td>
<td></td>
</tr>
<tr>
<td>Curriculum implementation</td>
<td></td>
</tr>
<tr>
<td>(guidance)</td>
<td></td>
</tr>
<tr>
<td>Appendices (as necessary)</td>
<td></td>
</tr>
<tr>
<td>- Rationale (on detailed</td>
<td></td>
</tr>
<tr>
<td>decisions)</td>
<td></td>
</tr>
<tr>
<td>- summary of survey</td>
<td></td>
</tr>
<tr>
<td>results</td>
<td></td>
</tr>
<tr>
<td>- program support (faculty, infrastructure, scope – evolution &amp; growth)</td>
<td></td>
</tr>
<tr>
<td>- mapping</td>
<td></td>
</tr>
</tbody>
</table>

References/Bibliography

Strawman Curriculum Original Schedule

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Requirements</td>
<td></td>
<td>DRAFT 1</td>
<td></td>
<td>REVIEW</td>
<td>DRAFT 2</td>
<td></td>
</tr>
<tr>
<td>2. Architecture</td>
<td></td>
<td>DRAFT 1</td>
<td></td>
<td>REVIEW</td>
<td>DRAFT 2</td>
<td></td>
</tr>
<tr>
<td>3. BOK</td>
<td></td>
<td>DRAFT 1</td>
<td>REVIEW</td>
<td></td>
<td>DRAFT 2</td>
<td></td>
</tr>
<tr>
<td>4. Packaging</td>
<td>DRAFT 1</td>
<td>REV</td>
<td>DRAFT 1</td>
<td>REVIEW</td>
<td>DRAFT 2</td>
<td></td>
</tr>
<tr>
<td>5. Remaining</td>
<td>DRAFT 1</td>
<td>REV</td>
<td>DRAFT 1</td>
<td>REVIEW</td>
<td>DRAFT 2</td>
<td></td>
</tr>
<tr>
<td>6. Assembly</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix D: “As-Is Analysis”, Art Pyster

An Analysis of Current Masters Programs in Software Engineering

Art Pyster
November 30, 2007

Background

- Software drives the performance of virtually all major systems.
- Being able to produce software that can be trusted as reliable, secure, safe, correct, and available while being delivered on-time and within budget is a major challenge for both the government and industry.
- Many steps must be taken to meet that challenge - including ensuring our workforce is well educated in software engineering (SWE) principles and practices.
- Yet, the last effort to create a reference curriculum for graduate software engineering education was by the SEI in the early 1990s.
iSSEc - The Way Forward

The Integrated Software and Systems Engineering Curriculum Project (iSSEc) is creating a reference curriculum leading to a Masters degree in software engineering.

iSSEc - The Way Forward

- iSSEc is sponsored by the Deputy Director, Software Engineering and System Assurance, OSD and led by Stevens Institute, involving 4 sets of stakeholders:
  - The industrial and government workforce who are the customers of SWE graduate education
  - Academics who provide SWE and systems engineering (SE) graduate education
  - Professional societies with a vested interest in SWE and SE graduate education
  - Government organizations who fund improvements in SWE graduate education
- iSSEc recognizes that the divide between systems and software engineers in industry, government, and academia works against successfully delivering modern systems in which software is almost always central.
- iSSEc will integrate SE principles and practices into the SWE curriculum.
The Approach

1. **Understand the current state of SWE graduate education**  
   (November 30, 2007)
2. Create a strawman model curriculum, suitable for broad use, with a small representative team (February 2008)
3. Publicize effort through conferences, papers, website, etc. (continuous)
4. Obtain endorsement from ACM, IEEE, INCOSE, NDIA, and other professional organizations (continuous)
5. Create full model curriculum, suitable for global use, with a large representative team (deliver approximately September 2008 and September 2009)
6. Seek early adopters (continuous)

Methodology to Understand Current State

- Selected diverse set of universities with Masters programs in SWE
  - Vary in size, geography, maturity, resources, target market, …
  - Focused on programs with degree in SWE or Computer Science with a SWE specialization - not degrees in information technology and related areas
- Used Software Engineering Body of Knowledge (SWEBOK) as the primary framework for SWE competencies
- Collected data from school websites
  - Degree, faculty size, student population, target market, …
  - Degree structure, individual course descriptions
  - Map between courses and SWEBOK
- Validated data with one or more professors from each school
- Analyzed for commonalities and uniqueness
Schools Studied

1. Air Force Institute of Technology
2. Brandeis University
3. California State University – Fullerton
4. California State University – Sacramento
5. Carnegie Mellon University
6. Carnegie Mellon University West
7. DePaul University
8. Drexel University
9. Dublin City University (Ireland) *
10. Embry-Riddle Aeronautical University
11. George Mason University
12. James Madison University
13. Mercer University
14. Monmouth University
15. Naval Postgraduate School
16. Penn State University – Great Valley
17. Quebec University (Canada) *
18. Rochester Institute of Technology
19. Seattle University
20. Southern Methodist University
21. Stevens Institute of Technology
22. Texas Tech University
23. University of Alabama – Huntsville
24. University of Maryland University College
25. University of Michigan – Dearborn
26. University of Southern California
27. University of York (UK) *
28. Villanova University

* Non-US Schools

Observations from 28 Schools

1. SWE is largely viewed as a specialization of Computer Science - much as systems engineering was often viewed as specialization of industrial engineering or operations research years ago
2. Faculty size is small - few dedicated SWE professors, making programs relatively brittle *
3. Student enrollments are generally small compared to CS and to other engineering disciplines
4. Many programs specialize to specific markets such as defense systems or safety critical systems
5. The target student population varies widely - anyone with Bachelors and B average to someone with CS degree and 2+ years of experience
6. Online course delivery is popular

* A program is brittle if a small change in faculty composition can have a large impact on program content or capacity.
More Observations

7. Objective for graduates vary widely - software developer to researcher to software manager
8. Wide variation in depth and breadth of SWEBOK coverage in required and semi-required courses
9. Many programs have required or semi-required courses that cover material that is either not in the SWEBOK at all or is not emphasized in the SWEBOK
10. Some significant topics are rarely mentioned - agility, software engineering economics, systems engineering
11. Some topics are ubiquitous - formal methods and architecture
12. “Object-oriented” is the standard development paradigm - creating a “clash” with many systems engineering programs that emphasize structured methods

* A student has a 50% or greater probability of taking a semi-required course.

Programs Have Diverse Focuses

1. Development of defense systems
2. Acquisition of defense systems
3. Embedded real-time systems
4. Entrepreneurial technology companies
5. Quantitative software engineering
6. Software economics
7. Safety critical systems
8. Secure software engineering
9. Highly dependable software systems

No focus dominated
Programs Prepare Graduates for Diverse Roles

Most programs prepare students to be practicing software engineers. Two prepare students to acquire software. Several prepare students to manage teams of software engineers or to be researchers.

1. Develop and modify large, complex software systems
2. Be an effective member of software development team
3. Procure highly dependable, trustworthy software-intensive systems
4. Understand and apply advanced SWE principles vital to industry
5. Realize software products on time, within budget and with known quality
6. Provide industrial leadership as a software engineer
7. Prepare students for research
8. Prepare students to be software engineers and software process managers in industry and government
9. Develop future chief engineers, head designers, principal technical officers
10. Provide software development professional with a sound educational basis for their work, and an opportunity to broaden skills

Differing Entrance Requirements

Most programs offer leveling courses for students lacking entrance requirements
Many programs routinely waive academic requirements for students with industrial experience

% programs

Know how to Program
Degree in Eng/Sci / Math
Degree in Computing
Industry Experience
Number of Full-time and Adjunct Faculty

- Many software engineering faculty also teach computer science
- Heavy dependence on adjunct faculty
- Most faculty have industry experience

Number of Students Currently Enrolled

- >100: 29%
- ≤25: 33%
- 26-50: 17%
- 51-75: 0%
Year that Program Started

<table>
<thead>
<tr>
<th>Year</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1989 &amp; before</td>
<td>2</td>
</tr>
<tr>
<td>1990 - 1995</td>
<td>8</td>
</tr>
<tr>
<td>1996 - 2000</td>
<td>7</td>
</tr>
<tr>
<td>2001-2005</td>
<td>4</td>
</tr>
<tr>
<td>2006 &amp; after</td>
<td>1</td>
</tr>
</tbody>
</table>

Department or School in which Degree is Offered

- School of Systems and Enterprises (Stevens)
- Dept. of Electrical and Computer Engineering (AFIT, Drexel, Mercer)
- Dept. of Engineering and Information Science (Penn State – Great Valley)
- Dept. of Information and Technology Systems (University of Maryland – University College)
- Division of Professional Studies (Brandeis)

“Department or school names may vary slightly, but are close to "software-engineering" or "computer science".”
Degree Offered

- MS in CS (SWE spec.) 11%
- MS in IT (SWE spec.) 4%
- MSE 14%
- MS in SWE 71%

- MSE – Master of Software Engineering
- MS in SWE – Master of Science in Software Engineering
- MS in CS (SWE spec.) – Master of Science in Computer Science with specialization in Software Engineering
- MS in IT (SWE spec.) – Master of Science in Information Technology with specialization in Software Engineering

What Percentage of Courses are Required?

- ≤25% 0%
- ≥75% 37%
- 26% - 50% 19%
- 51% - 75% 44%

- Required: Student must take the course
- Semi-Required: 50% or more probability that course will be taken

- On average, students take 11.6 courses for the degree of which 8.3 are required.
- For 43% of the programs, this includes a **required** thesis, project or practicum, which is normally the equivalent of 2 or 3 courses.
SWEBOK Coverage in Introductory Course(s)

- 0: No coverage of topic
- 1: Some coverage of topic
- 2: Significant coverage of topic

*Only 39% of the programs offer an introductory course
Most common reason for not offering course: expect students to have industrial experience before entering program

Introductory/Foundation level course in Software Engineering, as indicated in Course Title or Description

Individual SWEBOK Coverage in Introductory Course(s)
Composite SWEBOK Coverage in Introductory Course(s)

- Reasonable consistency in coverage across programs that offer an introductory course
- Requirements and design are best covered
- Tools and quality are least covered

% of Programs with coverage

SWEBOK Coverage in Required and Semi-Required Courses

- 0: No coverage of topic
- 1: Some coverage but no dedicated course
- 2: One dedicated course
- 3: Two or more dedicated courses

- Required: Student must take the course
- Semi-Required: 50% or more probability that course will be taken
Individual SWEBOK Coverage in Required and Semi-Required Courses

Composite SWEBOK Coverage in Required and Semi-Required Courses

- No consistency in coverage across programs
- Requirements, design, and management are best covered
- Maintenance, configuration management, quality, and tools are least covered

% of Programs with one or more dedicated courses
Some Novel Required & Semi-Required Courses

1. Reverse Engineering (Drexel)
2. Software Evolution and Re-engineering (Rochester)
3. Software Documentation (Penn State Great Valley)
4. Software Risk Assessment in DoD (NPS)
5. Refactoring (Mercer)
6. Structured Document Interchange and Processing (DePaul)
7. Avoiding Software Project Failures (Carnegie Mellon – West)
8. Mathematical Foundations of Software Engineering (Monmouth)
11. Professional, Ethical and Legal Issues for Software Engineers (Cal. State Univ. – Fullerton)
13.Lean & Agile Software Processes (Mercer)
14. Artificial Intelligence (Michigan - Dearborn)
15. Software Engineering Economics (GMU, USC)
16. Computer Game Design & Implementation (Michigan - Dearborn)
17. Service Oriented Architecture (Dublin City)

Special Emphasis Required and Semi-Required Courses

<table>
<thead>
<tr>
<th>Category</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>HCI: Human Computer Interfaces, Graphics &amp; Visualization</td>
<td>100</td>
</tr>
<tr>
<td>ALG: Algorithms</td>
<td>90</td>
</tr>
<tr>
<td>DBMS: Databases &amp; Information Management</td>
<td>80</td>
</tr>
<tr>
<td>SS: Safety &amp; Security</td>
<td>70</td>
</tr>
<tr>
<td>NET: Networks, Internet &amp; Distributed Systems</td>
<td>60</td>
</tr>
<tr>
<td>OS: Operating Systems &amp; Middleware</td>
<td>50</td>
</tr>
<tr>
<td>HW: Hardware &amp; Computer Engineering</td>
<td>40</td>
</tr>
<tr>
<td>RES: Research &amp; Experimental Methods</td>
<td>30</td>
</tr>
<tr>
<td>RTES: Real-time &amp; Embedded Systems</td>
<td>20</td>
</tr>
</tbody>
</table>

Shown are 9 categories with required or semi-required courses in at least 10% of the programs
Possible Additional Studies

- Analysis Based on Already Collected Data
- Analysis of Electives
- Differentiation between different types of schools (e.g. Private vs. Public; Large vs. Small; Specialized vs. General)

- Analysis Requiring Additional Data Collection
- Collect data from additional Non-US schools
- Program design questions (e.g. “Why do you have MSE vs. MS in CS (with SWE spec.)?”; “How did you decide the number of required courses?”; “How did you design your curriculum?”)
- Program evolution questions (e.g. “How has your program content changed since inception?”; “How have student demographics changed over time?”)
Appendix E: “Guidance and Outcomes”, Art Pyster

Guidance and Outcomes for a Model Curriculum for a Masters Degree in Software Engineering (MSwE2008)

Version 0.5

October 12, 2007

This section describes the foundational guidance for developing the MSwE2008 materials: the guiding principles, assumptions, and context for the entire MSwE2008 effort, and the desired student outcomes for a Masters degree in software engineering.

1. MSwE2008 Guidance

The following guidance was strongly influenced by the principles stated in SE2004; in some cases, they repeat verbatim the wording of those principles. Differences arise primarily by distinguishing the higher expectations of graduate education from undergraduate education and by more explicitly recognizing how the software engineering discipline ties to other disciplines such as systems engineering and project management. Moreover, we recognize that this guidance is, in many cases, not unique to software engineering curricula. It is valid for virtually all engineering disciplines. For example, guidance statement [4] is “All software engineering students must learn to integrate theory and practice.” Substituting “mechanical”, “electrical” or any other engineering discipline for “software” would not change its validity. Nevertheless, these statements are helpful to those developing MSwE2008 and are therefore included here.

1) The principle purpose of MSwE2008 is to provide a framework for development and improvement of curricula that provide advanced software engineering education at the master’s degree level.

MSwE2008 is intended to help address the needs of industry and government for software engineers, by equipping them with the most current theory and practice, and developing their ability to lead in addressing the future challenges to software development.

2) Computing is a broad field that extends well beyond the boundaries of any one computing discipline.

MSwE2008 concentrates on the knowledge and pedagogy associated with a graduate software engineering curriculum. Where appropriate, it will share or overlap with material contained in other Computing Curriculum reports and it will offer guidance on its incorporation into other disciplines.

3) Software Engineering draws its foundations from a wide variety of disciplines.

Graduate study of software engineering relies on many areas in computer science for its theoretical and conceptual foundations, but it also draws from other fields, including statistics, logic, calculus, discrete mathematics, formal languages, and other mathematical specialties, from systems and domain engineering, from project and portfolio management,
and from one or more application domains.

4) *All software engineering students must learn to integrate theory and practice.*

Students must be able recognize the importance of abstraction and modeling for software architecture, design and specification, to be able to acquire special domain knowledge beyond the computing discipline for the purposes of supporting software development in specific domains of application, and to appreciate the value and attributes of good design.

5) *The rapid evolution and the professional nature of software engineering require an ongoing review of the corresponding curriculum.*

Universities, industry, and government, in cooperation with professional associations in this discipline must establish an ongoing review process that allows individual components of the curriculum recommendations to be updated on a recurring basis. Also, because of the special professional responsibilities of software engineers to the public, it is important that the curriculum guidance support and promote effective external assessment and accreditation of graduate software engineering programs.

6) *Development of a graduate software engineering curriculum must be sensitive to changes in technologies, practices, and applications, new developments in pedagogy, and the importance of lifelong learning.*

In a field that evolves as rapidly as software engineering, educational institutions must adopt explicit strategies for responding to change. Institutions, for example, must recognize the importance of remaining abreast of well-established progress in both technology and pedagogy, subject to the constraints of available resources. Software engineering education, moreover, must seek to prepare students for lifelong learning that will enable them to move beyond today's technology to meet the challenges of the future.

7) *MSwE2008 must go beyond knowledge elements to offer significant guidance in terms of individual curriculum components.*

The MSwE2008 curriculum models should assemble the knowledge elements into reasonable, easily implemented learning units. Articulating a set of well-defined curriculum models will make it easier for institutions to share pedagogical strategies and tools. It will also provide a framework for publishers who provide the textbooks and other materials.

8) *MSwE2008 must support the identification of the fundamental skills and knowledge that all graduates of a Masters program in software engineering must possess.*

Where appropriate, MSwE2008 must help define the common themes of the software engineering discipline, including its dependencies on other related disciplines such as systems engineering and project management, and ensure that all graduate program recommendations include this material.

9) *Guidance on software engineering curricula must be based on an appropriate definition of software engineering knowledge and a flexible architecture.*

The description of this knowledge should be concise, appropriate for graduate education, and should use the work of previous studies on relevant bodies of knowledge and curricula, especially:

Bodies of knowledge from related disciplines should be included as appropriate, especially INCOSE’s Systems Engineering Body of Knowledge and the Project Management Institute’s Body of Knowledge. A core set of required topics, from this description, must be specified for all graduate software engineering degrees. The core should have broad acceptance by the software engineering education community and related communities. Coverage of the core will start with the introductory graduate courses, extend throughout the curriculum, and be supplemented by additional courses that may vary by institution, degree program, or individual student.

10) **MSwE2008 must strive to be international in scope.**
Despite the fact that curricular requirements differ from country to country, MSwE2008 must be useful to software engineering educators throughout the world. Where appropriate, every effort should be made to ensure that the curriculum recommendations are sensitive to national and cultural differences so that they will be widely applicable throughout the world. The involvement by national computing societies and volunteers from all countries should be actively sought and welcomed. For the Strawman version of MSwE2008, most of the authors will be from the United States, but for subsequent versions, significant international involvement is seen as critical.

11) **The development of MSwE2008 must be broadly based.**
To be successful, the process of creating software engineering education recommendations must include participation from the many perspectives represented by software engineering educators and by educators from related disciplines and by industry, commerce, and government professionals.

12) **MSwE2008 must include exposure to aspects of professional practice as an integral component of the graduate curriculum.**
The professional practice of software engineering encompasses a wide range of issues and activities, including problem solving, management, ethical and legal concerns, written and oral communication, working as part of a team, and remaining current in a rapidly changing discipline.

13) **MSwE2008 must include discussions of strategies and tactics for implementation, along with high-level recommendations.**
Although it is important for MSwE2008 to articulate a broad vision of software engineering education, the success of any curriculum depends heavily on implementation details. MSwE2008 must provide institutions with advice on the practical concerns of setting up a
curriculum. That advice must recognize that academic institution and department visions and missions vary widely and may require different approaches to develop and maintain a graduate software engineering curriculum. For example, they may differ in student demographics, teaching/research/professional focus, delivery mechanisms, external constituents, infrastructure, regulations and accreditation issues, and other areas. The core curriculum and implementation guidance should try to accommodate such differences, including guidance on how programs might extend the core to incorporate institution-specific needs (e.g., focus on a particular applications domain).

14) The distinction between SE2004 and MSwE2008 must be clear and apparent.

Compared to SE2004, MSwE2008 content is more narrowly focused on software engineering and related disciplines. The MSwE2008 curriculum expects much greater sophistication in student reasoning about software engineering principles, and expects students to demonstrate their accumulated skills and knowledge in a significant project, practicum, or thesis.

15) MSwE2008 must identify prerequisite requirements for students to enter an MSE program.

Undergraduate computing programs and industry experience in software engineering vary greatly. To help institutions build programs that address the needs of the broad software engineering community, MSwE2008 recommends minimum prerequisite knowledge necessary to successfully engage in a program based on the MSwE2008 curriculum. Generally, that knowledge comes from a technical, scientific, or engineering undergraduate degree including coursework in computer science. However, relevant work experience can substitute for formal education. Schools that wish to admit students lacking that minimum prerequisite knowledge should provide preparatory courses that those students should take before entering the Masters program.

2. Student Outcomes

Graduates of a Masters program in software engineering shall be able to:

[1] Show mastery of the software engineering knowledge and skills, and professional issues necessary to practice as a software engineer in a variety of application domains with demonstrated performance in at least one application domain.

Students, through regular reinforcement and practice, need to gain confidence in their abilities as they progress through a software engineering program of study. At graduation, a student should understand what distinguishes practice in different application domains such as finance, medical, transportation, and telecommunications, should understand how to learn a new domain as needed, and should demonstrate skill as a software engineer in at least one application domain. Such demonstration will include (as defined in Bloom’s Taxonomy)

- At least comprehension level competency across all MSwE2008 BOK knowledge areas, not including the KA on “Knowledge Areas of the Related Disciplines”.
- Application level competency, or above, in 75% of the MSwE2008 BOK knowledge areas.

Hence, a graduate should be able to analyze, design, verify, validate, implement, apply, and
maintain a modest-sized software system and understand the challenges of scaling to larger software systems. In addition, graduates need to have gained an understanding and appreciation of professional issues related to ethics and professional conduct, economics, and the societal needs.

[2] Show mastery of software engineering in at least one technical specialty such as embedded devices, safety critical systems, or highly distributed systems.

Students must go beyond core knowledge and skills and develop depth in at least one technical specialty.

[3] Work effectively as part of a team, including teams that may be international and geographically distributed, to develop quality software artifacts, and to lead in one area of project development, such as project management, requirements analysis, architecture, construction, or quality assurance.

Students need to complete tasks that involve work as an individual, but also many other tasks that entail working with a group of individuals. For group work, students ought to be informed of the nature of groups and of group activities/roles as explicitly as possible. This must include an emphasis on the importance of such matters as a disciplined approach, the need to adhere to deadlines, communication, and individual as well as team performance evaluations. Students should have an appreciation of team dynamics and leadership techniques and be able to lead at least one of the areas. Increasingly, teams are assembled from many geographical sites, often across national boundaries. This presents additional challenges of time, language, and culture that students must know how to address.

[4] Reconcile conflicting project objectives, finding acceptable compromises within limitations of cost, time, knowledge, existing systems, and organizations.

Students should engage in realistic exercises that expose them to conflicting and changing requirements. Curriculum units should address these issues, with the aim of ensuring high quality requirements and a feasible software design. Master students should be able to reason about overall project planning and updating to reflect evolving understanding of stakeholder needs.

[5] Design appropriate software engineering solutions that address ethical, social, legal, and economic concerns.

Throughout their study, students need to be exposed to a variety of appropriate approaches to engineering design in the general sense, and to specific problem solving in various kinds of applications domains for software. Their proposed design solutions must be made within the context of ethical, social, legal, security, and economic concerns.

[6] Understand and appreciate the importance of negotiation, effective work habits, leadership, and good communication with stakeholders in a typical software development environment.

The presence of a Capstone project or a thesis, an important final activity at the end of a software engineering program of study, is of considerable importance in this regard. It offers students the opportunity to tackle a major project and demonstrate their ability to bring together topics from a variety of courses and apply them effectively. This mechanism allows students to demonstrate their appreciation of the broad range of software engineering topics and their ability to apply their skills to genuine effect. This should also include the ability to offer reflections on their achievements.
[7] **Learn new models, techniques, and technologies as they emerge, and appreciate the necessity of such continuing professional development.**

In a field as dynamic as software engineering, life-long learning is essential to continued success.

[8] **Analyze a current significant software technology, be able to articulate its strengths and weaknesses, and be able to specify and evangelize improvements or extensions to that technology.**

Technologies change frequently. A software engineer must be able to select new technologies, understanding their limitations and appropriate uses.
Appendix F: “Architecture Document”

Architecture for a Model Curriculum
For a Masters Degree in Software Engineering (MSwE2008)

Version 1.5

December 4, 2007

This section describes a structure into which courses and the curriculum of the MSwE2008 model curriculum can fit. It is similar to the architecture proposed in the 1991 SEI report [SEI 1991] and is compatible with the existing MSSwE programs for which course and curriculum data have been collected. It is intended to provide a structural basis for programs which deliver the outcomes described in the Guidance and Outcomes and Body of Knowledge sections of this recommendation.

The architecture includes preparatory material, high priority core materials, lower priority core materials, university-specific materials and elective materials and a capstone experience. Figure 1 (in Appendix G) provides an overview of the architecture.

Students entering the program could have undergraduate degrees in computer science, software, computer, electrical or other fields of engineering, business or other degrees, with or without significant industrial experience. MSwE2008 recommends a minimum level of prerequisite knowledge and ability that is approximately at the level of an undergraduate who has completed a Bachelor of Science degree in computer science or software engineering. That expected level would be necessary to successfully engage in a program based on the MSwE2008 curriculum. Colleges and universities that wish to admit students lacking that minimum prerequisite background will provide preparatory courses containing materials that those students should take before entering the master’s program. Those are the preparatory materials shown above the dark horizontal line in Figure 1. It is anticipated that a few students, with undergraduate degrees in a variety of fields plus extensive experience, might enter directly into courses which cover only a subset of the core materials and, perhaps occasionally, directly into courses which include the university specific and elective materials.

MSwE2008 identifies the fundamental skills and knowledge that all graduates of a master’s program in software engineering must possess. Where appropriate, it defines the common themes of the software engineering discipline, including its dependencies on other related disciplines such as systems engineering and project management, and ensures that all graduate program recommendations include this material. Courses that include this material are called mandatory, or core, courses. They fit into the mandatory courses block of Figure 1 (in Appendix G).
Virtually all students will be required to take courses that cover the high priority materials indicated by the smallest of the half-circles in Figure 1. The lower priority core materials include materials that it would be desirable for all students to master. They have been included in the core Body of Knowledge (BOK) developed by the BOK work group. However, in order for an institution to tailor its program to achieve specific outcomes some of these materials might not be included in a specific institution’s program.

The next half circle in Figure 1 (in Appendix G), labeled University specific materials, represents materials that an institution might include in order to tailor its program to meet its specific objectives. These may vary by institution or degree program. They may differ because of student demographics, teaching/research/professional focus, delivery mechanisms, external constituents, infrastructure or accreditation issues. The optional or elective materials within the architecture are intended to accommodate differences among programs and among individual students, including differences in how programs handle institution-specific needs (e.g., the focus on particular application domains).

The MSwE2008 curriculum expects students to demonstrate their accumulated skills and knowledge in a capstone experience which might be a project, a practicum, or a thesis. In this context a practicum would be a software development project done for a real external customer by a group of students. A project would be a practically oriented undertaking done by a single student or a group for or with someone within the offering institution. And, a thesis would be a piece of software engineering research work completed by an individual student under the guidance of a research oriented member of the faculty. Students completing the curriculum must be able to understand and appreciate the importance of negotiation, effective work habits, leadership, and good communication with stakeholders in a typical software development environment. The presence of a capstone project, a practicum or a thesis at the end of the curriculum is of considerable importance in this regard. It offers students the opportunity to tackle a major undertaking and demonstrate their ability to bring together topics from a variety of courses and apply them effectively as shown by the broken lines connecting the capstone experience back to the materials contained in the various layers of the curriculum.

There is no intent in this architectural specification to require either the content of preparatory courses or the content in core courses to be self-contained in courses with names corresponding to the topics. Figure 2 (in Appendix G) provides an example showing how this might happen. The yellow wedges in this figure correspond to standard courses corresponding directly to courses that might be developed by the Packaging and Delivery Group. Others, such as the green wedges in Figure 2 might be university specific courses that concentrate primarily on university specific materials and/or elective materials. Some might be within one level of the architecture and some might cross the circular boundaries of the architecture. Topics can be self-contained or they can be spread throughout several courses in each ring, or even across multiple rings. Courses might have names that are quite different from the names of the materials specified in the curriculum.

There is also no intent that all of the courses containing preparatory or core materials must be
completed before coursework in the next ring can begin. It is anticipated that the sequencing of courses will be controlled primarily by the prerequisite specifications of each course in a specific institution’s curriculum.

Figure 3 (in Appendix G) provides an example of how a track could be constructed within this architectural framework. In this example the track would include all of the high priority core materials, some of the lower priority core, some university specific materials and a capstone experience concentrating on a topic associated with the specific track.

Appendix G: “Architecture”, Mary Shaw

MSWe2008 Architecture

Gary Hafen
Anthony Lattanze
Jim McDonald
Bret Michael
Ken Nidiffer
Mel Rosso-Llopart
Mary Shaw
Sarah Sheard

OBJECTIVES OF ARCHITECTURE GROUP

• Respect the legacy
  – Recognize the contribution of the SEI 1991 curriculum architecture
  – Be generally consistent with existing MSSE programs

• Provide a model for guidance with minimal constraint on individual university decisions
  – Avoid the implication that a specific set of courses must be taken in a specific order
  – Allow enough flexibility so that institutions can add materials to help achieve its own unique objectives
  – Provide for the possibility that materials might be organized into pre-packaged standard courses or into courses that cut across the curriculum to include a variety of materials
Figure 1 (for Appendix F)
Figure 2 (for Appendix F)

Figure 3 (for Appendix F)
OPEN QUESTIONS AND FOLLOWUP

• Can the size of the core be reduced?
  – Perhaps breaking the core into a high priority core and a lower priority core would resolve this issue
  – The architecture team has assumed this resolution

• What are the appropriate percentages?
  – Core as % of total? High priority core? Low priority core?
  – University-specific materials? Electives? Capstone?
  – What % of low priority core should all schools cover?

• We expect that these issues will be discussed by the BOK group during this workshop
• We welcome additional input that will improve the content of the architecture document
Appendix H: “Body of Knowledge”, Tom Hilburn

MSwE 2008
Body of Knowledge Draft 1.0

integrated Software and Systems Engineering
curriculum (iSSEc) Project
Early Start Team (EST) Workshop
December 11-12, 2007

BOK Team

- Mark Ardis
- John Brackett
- Tom Hilburn
- Bill Milam
- Rich Turner
- Support
  - Deva Henry
  - Kahina Lasfer
BOK Purpose

- The purpose of the BOK is to identify those knowledge areas, units and topics, which are recommended to be part of the core of all masters programs in software engineering. Draft 1.0 also provides guidance as to the amount and degree of consideration that should be devoted to various areas and units.

BOK Process

- The BOK Team developed the draft through three iterations. It started with a review of the 2004 SWEBOK [SWEBOK 2004], and through study and discussions identified elements that were missing, were not needed, or would benefit from reorganization. The two other primary sources used for added elements were the SEEK in [SE 2004] and the INCOSE BOK [INCOSE 2007]. The spreadsheet includes comments about elements added and changed. Although the Team considered the use of the PMI BOK [PMI 2006], it was decided that the SWEBOK provide sufficient coverage of software management topics.

- The Team considered several metrics for designating the significance of an element to an MSweE program (Bloom’s taxonomy, contact hours in the core, and percentage of the core). In the initial effort at categorization, the Team used a 300 contact hour allocation to judge the amount of consideration that should be given to BOK elements. Through a Delphi process the 300 hour allocation was refined to reach a “Team” view about the amount of consideration that should be given to the BOK elements. To better accommodate the work of the Architecture Team and other parties, the 300 hour budget was converted to a “% of the Core” recommendation.
BOK Knowledge Areas

Questions and Comments