Software Engineering Institute | Carnegie Mellon University

The Development of a Graduate Curriculum for Software Assurance

Mark Ardis
Nancy Mead

August 2011

ABSTRACT: One of our challenges as educators is timely incorporation of research into curricula that can be adopted by universities to ultimately improve software engineering practice. In this paper, we describe the work of the Master of Software Assurance curriculum project. This includes our sources, process, products, adoption strategies, and early adoption experiences. The project used research results, prior curricula, and documented bodies of knowledge to develop a new curriculum. We are now working with early adopters and employing a number of transition mechanisms as part of our strategy to further adoption in this critical area.

INTRODUCTION

Although software is ubiquitous in modern systems, the complexity of software and software-intensive systems poses inherent risk. This complexity, along with our reliance on these systems, suggests that attackers need only take down the most vulnerable component to have far-reaching and damaging effects on the larger system. In this environment, attackers no longer need to possess technical sophistication. Due to the growing supply of shared attack strategies, an unsophisticated attacker can easily acquire and launch a sophisticated attack.

On the bright side, in recent years considerable research has been done on ways of developing assured software that is resistant to attack and capable of recovering from one. However, much of that research has not made its way into software engineering practice, nor is it routinely taught at our universities.

In order to address this disconnect between research, education, and practical development of assured software, the U.S. Department of Homeland Security (DHS) National Cyber Security Division (NCSD) enlisted the Carnegie Mellon Software Engineering Institute (SEI) to develop a curriculum for a Master of Software Assurance degree program and define transition strategies for future implementation. The curriculum development team that was assembled included a mix of SEI staff members and university faculty, with editorial and administrative support provided by the SEI. The development team members, collectively,

Software Engineering Institute
Carnegie Mellon University
4500 Fifth Avenue
Pittsburgh, PA 15213-2612

Phone: 412-268-5800
Toll-free: 1-888-201-4479

www.sei.cmu.edu

had considerable background in software assurance research, software engineering research and practice, and software engineering education.

As noted in our curriculum report, the need for a master's level program in this discipline has been growing for years [Mead 2010a]:

- A study by the nonpartisan Partnership for Public Service points out that, "The pipeline of new talent [with the skills to ensure the security of software systems] is inadequate. . . . only 40 percent of CIOs [chief information officers], CISOs [chief information security officers] and IT [information technology] hiring managers are satisfied or very satisfied with the quality of applicants applying for federal cybersecurity jobs, and only 30 percent are satisfied or very satisfied with the number of qualified candidates who are applying" [PPP 2009].
- The need for cybersecurity education was emphasized in the New York Times when Dr. Nasir Memon, a professor at the Polytechnic Institute of New York University, was quoted as saying, "There is a huge demand, and a lot more schools have created programs, but to be honest, we're still not producing enough students" [Drew 2009].
- In discussions with industry and government representatives, we have found that the need for more capacity in cybersecurity continues to grow. Anecdotal feedback from the development team members' own students indicates that even a single course with a cybersecurity focus enhances their positioning in the job market. They felt that they were given job offers they would not have received otherwise.

Another aspect of the need for cybersecurity education occurs in educational institutions. Based on our collective experience in software engineering education, we know that it can be very difficult to start a new program or track from scratch, and we want to assist those organizations and faculty members that wish to undertake such an endeavor. Our objective is to support their needs, while recognizing that there are a variety of implementation strategies.

Recognizing that software assurance is not exactly the same as software engineering or information security, one of our first tasks was to review existing definitions of software assurance. We evolved from a definition that is currently in wide use [CNSS 2009] to one that we thought was a better fit for the curriculum work: "Software assurance (SwA) is the application of technologies and processes to achieve a required level of confidence that software systems and services function in the intended manner, are free from accidental or intentional vulnerabilities, provide security capabilities appropriate to the threat environment, and recover from intrusions and failures" [Mead 2010a].

This definition emphasizes the importance of both technologies and processes in software assurance, notes that computing capabilities may be acquired through services as well as new development, acknowledges the need for correct functionality, recognizes that security capabilities must be appropriate to the threat environment, and identifies recovery from intrusions and failures as an important capability for organizational continuity and survival.

While information security is important, academic programs in information security typically focus on system administrator activities for operational systems, whereas our focus was on systems under development. Software engineering provides much excellent foundational material, and all the curriculum development team members have a software engineering background. However, we recognized that development of assured software needs to go beyond good software engineering practice, and indeed the resulting curriculum reflects this.

In the remainder of this paper, we discuss our sources, the curriculum development process, our SwA education products, transition/adoption plans, and early adoption.

SOURCES

Although we knew of no existing graduate degree programs in software assurance, we had several good sources of curriculum material from related programs. A project sponsored by the U.S. Department of Defense (DoD) had recently developed a new set of guidelines for graduate software engineering (the GSwE2009) [Pyster 2009], and some members of our team had worked on that project. We also had intimate knowledge of earlier curriculum work in graduate software engineering [Ardis 1989, Ford 1991], undergraduate software engineering [IEEE 2004a], computer engineering [IEEE 2004b], and computer science [IEEE 2010].

The Software Engineering Body of Knowledge (SWEBOK) [IEEE 2004c] is the foundation for most curriculum work in software engineering. It provides a solid taxonomy of terms and practices that has been accepted by academia, industry, and government. We used SWEBOK in our work, but we also needed a similar body of knowledge for software assurance. An earlier attempt to create such a foundation [Redwine 2007] was helpful, but it has not achieved the same level of acceptance as SWEBOK. For that reason, we needed to validate our work with current practitioners and managers in software security.

There are many good textbooks on computer security and information assurance. In fact, two members of our project team were among the co-authors of a book on software security [Allen 2008]. We used these sources and course descriptions in developing our lists of topics and security practices that students needed to learn. In addition, the SEI's CERT Program maintains the Build Security In website [DHS 2011] that contains lots of useful material for curriculum and course development.

PROCESS

We followed this eight-step process to develop the curriculum recommendations:

1. **Develop Project Guidelines:** We adapted a similar set of guidelines from the GSwE2009 project to fit our needs. These adapted guidelines helped direct our work, especially when we were developing Outcomes and the Body of Knowledge (see Step 6).
2. **Identify and Review Sources:** We reviewed about 30 respected sources of security practices, including well-known textbooks and courses. These sources were particularly helpful in expanding details of Topics (see Step 3) and Outcomes (see Step 6).
3. **Define Topics:** We expanded on the main topics from [Allen 2008] to identify important topics and practices throughout the software development lifecycle (SDLC). These topics served as a first step toward organizing all the material needed in the curriculum.
4. **Define SDLC Practices and Categories:** We expanded each Topic (from the previous step) to the level of specific security practices used in industry, government, and academia. The Sources identified in Step 2 were used to make sure that we included as many different practices as possible. Then we grouped related practices into higher-level categories.
5. **Solicit External Feedback:** At this point, we asked practitioners, managers, and educators for feedback on our content so far. We were particularly interested in knowing whether graduates who acquired the knowledge and skills we had described would be valuable in their assigned positions. Results from a three-page questionnaire were used to revise our Practices and Categories.
6. **Develop Outcomes, Body of Knowledge, Curriculum Architecture, Course Descriptions, and Implementation Guidance:** We developed Expected Outcomes for graduates of a software assurance program starting with the Categories we identified in Step 4. We also elaborated the Categories and Practices into a Body of Knowledge to be mastered by students. We developed a Curriculum Architecture and a set of example course outlines to be used in creating an academic program, and we produced some Implementation Guidance for faculty who might take on such a task.

7. **Compare Knowledge Units from Body of Knowledge to SDLC Practices:** We checked to see that all the Practices identified in Step 4 were adequately covered by the knowledge units of our Body of Knowledge. This led to some minor revisions in both the Body of Knowledge and the Outcomes.
8. **Conduct External Review and Make Revisions:** Finally, we solicited feedback from external reviewers in academia, industry, and government and made appropriate revisions.

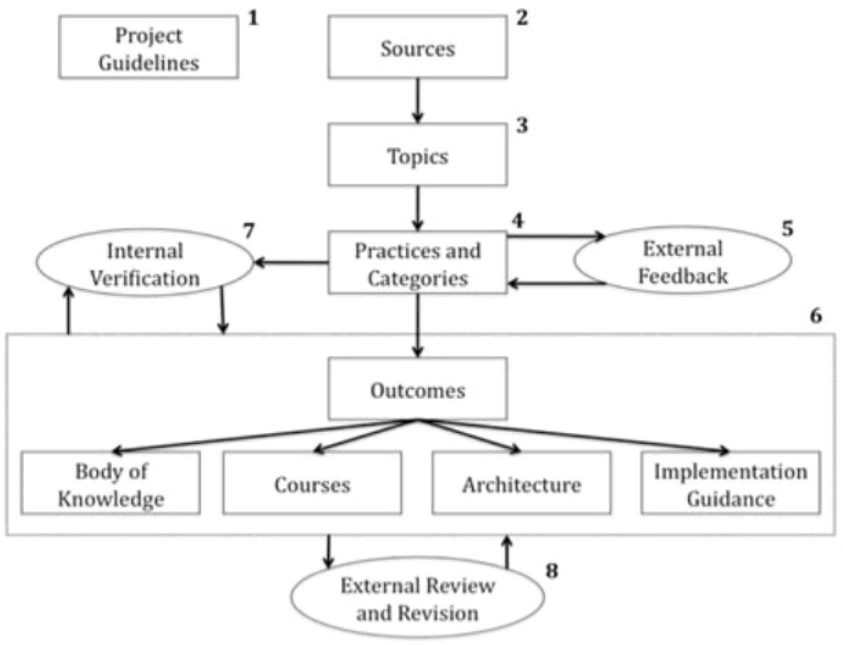

Figure 1. Relationship of project artifacts to our curriculum development process

As an example, an important artifact of the process is the body of knowledge. The Master of Software Assurance (MSwA) Reference Curriculum report recommends the core body of knowledge shown in (Table 1).

Table 1: MSwA Core Body of Knowledge [Mead 2010a]

Knowledge Area	
1. Assurance Across Life Cycles	1.1. Software Life-Cycle Processes
	1.1.1. New development
	1.1.2. Integration, assembly, and deployment
	1.1.3. Operation and evolution
	1.1.4. Acquisition, supply, and service
	1.2. Software Assurance Processes and Practices

	1.2.1. Process and practice assessment
	1.2.2. Software assurance integration into SDLC phases
2. Risk Management	2.1. Risk Management Concepts
	2.1.1. Types and classification
	2.1.2. Probability, impact, severity
	2.1.3. Models, processes, metrics
	2.2. Risk Management Process
	2.2.1. Identification
	2.2.2. Analysis
	2.2.3. Planning
	2.2.4. Monitoring and management
	2.3. Software Assurance Risk Management
	2.3.1. Vulnerability and threat identification
	2.3.2. Analysis of software assurance risks
	2.3.3. Software assurance risk mitigation
	2.3.4. Assessment of Software Assurance Processes and Practices
3. Assurance Assessment	3.1. Assurance Assessment Concepts
	3.1.1. Baseline level of assurance; allowable tolerances, if quantitative
	3.1.2. Assessment methods
	3.2. Measurement for Assessing Assurance
	3.2.1. Product and process measures by life-cycle phase
	3.2.2. Other performance indicators that test for the baseline, by life-cycle phase
	3.2.3. Measurement processes and frameworks
	3.2.4. Business survivability and operational continuity
	3.3. Assurance Assessment Process (collect and report measures that demonstrate the baseline)
	3.3.1. Comparison of selected measurements to the established baseline
	3.3.2. Identification of out-of-tolerance variances
4. Assurance Management	4.1. Making the Business Case for Assurance
	4.1.1. Valuation and cost/benefit models, cost and loss avoidance, return on investment
	4.1.2. Risk analysis
	4.1.3. Compliance justification

	4.1.4. Business impact/needs analysis
	4.2. Managing Assurance
	4.2.1. Project management across the life cycle
	4.2.2. Integration of other knowledge units
	4.3. Compliance Considerations for Assurance
	4.3.1. Laws and regulations
	4.3.2. Standards
	4.3.3. Policies
5. System Security Assurance	5.1. For Newly Developed and Acquired Software for Diverse Applications
	5.1.1. Security and safety aspect of computer-intensive critical infrastructure
	5.1.2. Potential attack methods
	5.1.3. Analysis of threats to software
	5.1.4. Methods of defense
	5.2. For Diverse Operational (Existing) Systems
	5.2.1. Historic and potential operational attack methods
	5.2.2. Analysis of threats to operational environments
	5.2.3. Designing of and plan for access control, privileges, and authentication
	5.2.4. Security methods for physical and personnel environments
	5.3. Ethics and Integrity in Creation, Acquisition, and Operation of Software Systems
	5.3.1. Overview of ethics, code of ethics, and legal constraints
	5.3.2. Computer attack case studies
6. System Functionality Assurance	6.1. Assurance Technology
	6.1.1. Technology evaluation
	6.1.2. Technology improvement
	6.2. Assured Software Development
	6.2.1. Development methods
	6.2.2. Quality attributes
	6.2.3. Maintenance methods
	6.3. Assured Software Analytics
	6.3.1. Systems analysis
	6.3.2. Structural analysis

	6.3.3. Functional analysis
	6.3.4. Analysis of methods and tools
	6.3.5. Testing for assurance
	6.3.6. Assurance evidence
	6.4. Assurance in Acquisition
	6.4.1. Assurance of acquired software
	6.4.2. Assurance of software services
7. System Operational Assurance	7.1. Operational Procedures
	7.1.1. Business objectives
	7.1.2. Assurance procedures
	7.1.3. Assurance training
	7.2. Operational Monitoring
	7.2.1. Monitoring technology
	7.2.2. Operational evaluation
	7.2.3. Operational maintenance
	7.2.4. Malware analysis
	7.3. System Control
	7.3.1. Responses to adverse events
	7.3.2. Business survivability

The leftmost column itemizes the 7 outcome areas of the curriculum. Brief descriptions for each outcome follow. More detailed descriptions can be found in the curriculum report.

Outcome 1. Assurance Across Life Cycles: Graduates will have the ability to incorporate assurance technologies and methods into life-cycle processes and development models for new or evolutionary system development, and for system or service acquisition.

Outcome 2. Risk Management: Graduates will have the ability to perform risk analysis, tradeoff assessment, and prioritization of security measures.

Outcome 3. Assurance Assessment: Graduates will have the ability to analyze and validate the effectiveness of assurance operations and create auditable evidence of security measures.

Outcome 4. Assurance Management: Graduates will have the ability to make a business case for software assurance, lead assurance efforts, understand standards, comply with regulations, plan for business continuity, and keep current in security technologies.

Outcome 5. System Security Assurance: Graduates will have the ability to incorporate effective security technologies and methods into new and existing systems.

Outcome 6. System Functionality Assurance : Graduates will have the ability to verify new and existing software system functionality for conformance to requirements and absence of malicious content.

Outcome 7. System Operational Assurance: Graduates will have the ability to monitor and assess system operational security and respond to new threats.

SOFTWARE ASSURANCE EDUCATION PRODUCTS

The list of software assurance education products developed by the SwA Curriculum Development team is extensive. All are available for download from the CERT website at http://www.cert.org/mswa/. The team developed the following materials:

1. The Master of Software Assurance Curriculum report [Mead 2010a] – This report is our principal product and as noted above includes•prerequisites expected of students entering an MSwA program
   - course architecture for both a standalone degree program and a track within a related degree program
   - a core body of knowledge that includes the fundamental topics addressed in the curriculum
   - educational outcomes that graduates from a program based on the curriculum should have achieved
   - guidelines for educational institutions interested in implementing the curriculum
2. A companion report that provides course outlines that could be used in an undergraduate degree program [Mead 2010b] – In this report we provide course outlines as well as suggestions for an undergraduate specialization in software assurance that could be incorporated into a number of degree programs.
3. Faculty Seminar Materials – a set of PowerPoint slides with notes that can be used in a faculty seminar or workshop to raise awareness.
4. Course materials•Course outlines and syllabi for the MSwA core courses
   - A master bibliography

  - Selected lecture materials made available to us, which will hopefully be expanded over time

PLANS TO PROMOTE ADOPTION

It was clear to us from the outset that we would need a comprehensive plan for promoting transition and adoption of the curriculum. Many of us had been involved in curriculum work previously and knew that transition was a lengthy process and that there were many barriers to adoption. Introducing a single new elective course is a relatively easy undertaking. However, introducing a track is ambitious, and contemplating a whole new degree program can be a daunting task. The barriers can range from lack of interested students in a particular geographic area, to lack of qualified faculty, to lack of administrative support. We therefore put a transition plan in place before the curriculum was published and are executing it this year. The planned activities include

- **Publicity** – We prepared an announcement that was broadcast via email, sent to SEI subscribers, and posted on the DHS and SEI websites. We also developed a press release that went out to a number of educational publications, professional societies such as ACM and IEEE, and ACM and IEEE publications. We developed a flyer that is being distributed by team members and their colleagues when they attend conferences.
- **Discussion group** – We established a LinkedIn discussion group so that faculty interested in implementing all or parts of the curriculum could talk to the curriculum development team and to colleagues who are also using the curriculum. We invited a number of colleagues and regularly post discussion topics on the board.
- **Awareness** – We also conducted an awareness-raising workshop at CSEET 2010 and videotaped it for viewing through the SEI's virtual training environment (VTE). Finally, we recorded a podcast to provide an overview of the work.
- **Mentoring** – The curriculum development team is providing free mentoring to universities or faculty members who wish to offer a course, track, or MSwA degree program. The mentoring support includes review of implementation plans, review of course outlines, and advice on references and other course materials.
- **Publication** – We are writing papers and giving talks on the curriculum work.
- **Professional society recognition** – We have received official recognition of the curriculum from the ACM and the IEEE Computer Society.

EARLY ADOPTION

Stevens Institute of Technology, the first school to adopt MSwA2010, has developed two tracks in software assurance within its Master of Science in Software Engineering program [SIT 2010]. One track is for students who anticipate a career in secure software development, and the other track is for students interested in acquisition and management of trusted software systems. In addition, graduate certificates are available for students who already have an advanced degree or are not yet ready to commit to a full graduate program.

Stevens was able to create its software assurance program so quickly for several reasons. First, it was able to use two courses from another program in systems security to cover some of the needed topics. Second, it was able to modify its core software engineering courses to include software assurance material. The goals of the software engineering program at Stevens are consistent with the need to develop trustworthy systems for telecommunications, defense, and financial infrastructures. Finally, one of the members of our project team teaches at Stevens.

Other schools, such as Capitol College, Hampton University, and the University of Nebraska Omaha, are considering adopting MSwA2010 for a graduate program. They are just starting to formulate plans for their programs. Southeast Missouri State University and Capitol College are considering using MSwA2010 as a foundation for an undergraduate program.

ADAPTATIONS FOR INFORMATION SYSTEMS PROGRAMS

We have also begun to study adaptations of the MSwA2010 curriculum for information systems curricula. Just as software assurance supports and complements the educational objectives of a software engineering program, it also supports and complements the educational objectives of a graduate information systems program. For example, graduates of MSIS programs need to be aware of potential system vulnerabilities, and they need to practice development methods that minimize or prevent those insecurities.

Many of the topics identified in MSwA2010 fit well within the core courses of MSIS2006 [Gorgone 2006]. For example, MSwA2010 includes material on software lifecycle and software assurance processes that could be included in the Project and Change Management course of MSIS206. Other areas of overlap are risk management, assurance assessment and assurance management. Those topics could be included in the Analysis, Modeling, and Design course and the Enterprise Models course. Some assurance topics, such as system operational assurance and system functionality assurance may not fit within the core courses of

MSIS2006, but might be covered in electives. A more complete analysis of possible overlap between MSwA2010 and MSIS2006 is reported in [Shoemaker 2011].

We feel that there is a lot of opportunity for inclusion of software assurance material in information systems programs. As part of our adoption and support activities we welcome opportunities to work with schools that wish to do this. Please contact either of the authors for more information about working with us. At the very least, please consider joining our LinkedIn group Software Assurance Education. The group provides a forum for faculty to share problems and experiences in teaching software assurance courses. We welcome participation by faculty from information systems and related areas.

CONCLUSION AND FUTURE PLANS

In this paper, we have sketched out the work of the MSwA curriculum development team in developing, publishing, and promoting adoption of software assurance degree programs and specializations within software engineering programs. Such degree programs are needed to prepare practitioners to address the vulnerability of today’s complex software systems with better development practices. In doing this work, we had to understand currently available research and bodies of knowledge, existing curricula, current organizational practices, and desired future practices.

Our current work is heavily involved in promoting adoption of all or parts of the curriculum at colleges and universities, including the service academies and schools. This will require continued team effort, exemplars, and a hands-on approach to adoption and transition.

We have recently started work on an effort to promote software assurance education at the community college level. In order to support this, we will initially develop a report that identifies existing assurance and security curricula that are relevant to community colleges, includes course outlines, and identifies suitable existing educational materials. The ACM Two-Year College Education Committee (TYCEC) will be partnering with the SwA education team in this effort. TYCEC has significant experience in this space http://www.acmtyc.org/ and much to contribute.

In the future, we hope to identify existing course materials that fit the MSwA curriculum, develop complete courses to support it, and continue to support universities that wish to implement all or part of it. We also hope to influence existing undergraduate computing curricula to include more software assurance material, while also reaching out to community colleges and high schools. Our objective is to educate future software assurance practitioners and those who wish to acquire assured software.

REFERENCES

[Allen 2008]	Allen, Julia H.; Barnum, Sean; Ellison, Robert J.; McGraw, Gary; & Mead, Nancy R. Software Security Engineering: A Guide for Project Managers. Addison-Wesley Professional, 2008.
[Ardis 1989]	Ardis, Mark A. & Ford, Gary. SEI Report on Graduate Software Engineering Education (CMU/SEI-89-TR-021, ESD-TR-89-29). Software Engineering Institute, Carnegie Mellon University, 1989. http://www.sei.cmu.edu/library/abstracts/reports/89tr021.cfm
[CNSS 2009]	Committee on National Security Systems. "Instruction No. 4009," National Information Assurance Glossary. Revised June 2009.
[DHS 2011]	Department of Homeland Security. Build Security In. https://buildsecurityin.us-cert.gov/bsi/home.html (2011).
[Drew 2009]	Drew, C. "Wanted: 'Cyber Ninjas.'" New York Times, 2009. Retrieved December 29, 2009 from http://www.nytimes.com/2010/01/03/education/edlife/03cybersecurity.html?emc=eta1
[Ford 1991]	Ford, Gary. 1991 SEI Report on Graduate Software Engineering Education (CMU/SEI-91-TR-002, ESD-TR-91-002). Software Engineering Institute, Carnegie Mellon University, 1991. http://www.sei.cmu.edu/library/abstracts/reports/91tr002.cfm
[Gorgone 2006]	Gorgone, John T.; Gray, P.; Stohr, E. A; Valacich, J. S.; & Wigand, R. T. "MSIS 2006: Model Curriculum and Guidelines for Graduate Degree Programs in Information Systems." Communications of the Association for Information Systems 17, 1, Article 1 (Jan. 2006): 2-75.
[IEEE 2004a]	IEEE Computer Society (IEEE-CS) & Association for Computing Machinery (ACM). Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. Computing Curriculum Series. http://sites.computer.org/ccse/SE2004Volume.pdf (2004).
[IEEE 2004b]	IEEE Computer Society (IEEE-CS) & Association for Computing Machinery (ACM). Computer Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering. Computing Curriculum Series. http://www.acm.org/education/education/curric_vols/CE-Final-Report.pdf (2004).
[IEEE 2004c]	IEEE Computer Society (IEEE-CS). Software Engineering Body of Knowledge (SWEBOK). http://www.computer.org/portal/web/swebok (2004).

[IEEE 2010]	IEEE Computer Society (IEEE-CS) & Association for Computing Machinery (ACM). Computer Science Curriculum 2008: An Interim Revision of CS 2001, http://www.acm.org//education/curricula/ComputerScience2008.pdf (2010).
[Mead 2010a]	Mead, N. R., et al. (2010a). Software Assurance Curriculum Project Volume I: Master of Software Assurance Reference Curriculum (CMU/SEI-2010-TR-005, ESC-TR-2010-005). Software Engineering Institute, Carnegie Mellon University, 2010. http://www.sei.cmu.edu/library/abstracts/reports/10tr005.cfm
[Mead 2010b]	Mead, N. R., et al. (2010b). Software Assurance Curriculum Project Volume II: Undergraduate Course Outlines (CMU/SEI-2010-TR-019, ESC-TR-2010-019). Software Engineering Institute, Carnegie Mellon University, 2010. http://www.sei.cmu.edu/library/abstracts/reports/10tr019.cfm
[PPP 2009]	Partnership for Public Service & Booz Allen Hamilton. Cyber IN-Security: Strengthening the Federal Cybersecurity Workforce. Partnership for Public Service, 2009. Retrieved July, 2009, from http://ourpublicservice.org/OPS/publications/viewcontentdetails.php?id=135
[Pyster 2009]	Pyster, A. (ed.). Graduate Software Engineering 2009 (GSwE2009) Curriculum Guidelines for Graduate Degree Programs in Software Engineering, Integrated Software & Systems Engineering Curriculum Project, Stevens Institute, September 30, 2009. http://www.gswe2009.org/curriculum/recommendations/document/
[Redwine 2007]	Redwine, Jr., Samuel T. (ed.). Department of Homeland Security (DHS) Software Assurance (SwA) Workforce Education and Training Working Group. Software Assurance: A Curriculum Guide to the Common Body of Knowledge to Produce, Acquire and Sustain Secure Software (2007).
[Shoemaker 2011]	Shoemaker, Dan; Mead, Nancy R.; & Ingalsbe, Jeff. Integrating the Master of Software Assurance Reference Curriculum into the Model Curriculum and Guidelines for Graduate Degree Programs in Information Systems (CMU/SEI-2011-TN-005). Software Engineering Institute, Carnegie Mellon University, 2011 (forthcoming). http://www.sei.cmu.edu/library/abstracts/reports/11tn005.cfm
[SIT 2010]	Stevens Institute of Technology. Software Assurance at Stevens Institute of Technology. http://stevens.edu/softwareassurance (2010).