Two Nationally Sponsored Initiatives for Disseminating Assurance Knowledge

Dan Shoemaker and

Nancy Mead

February 2011

ABSTRACT: Education in software assurance is an essential element in the effort to produce secure code. This paper describes two efforts that support national cybersecurity education goals: development of software assurance learning artifacts that can be integrated into conventional learning environments and development of a master of software assurance reference curriculum.

SOFTWARE DEFENSE APPLICATION

Cybersecurity has been an area of national interest for almost a decade. Yet, the Department of Homeland Security's Common Weakness Enumeration [20] documents 797 common defects, starting from errors related to environmental location (CWE 1) to Filtering Special Elements at an Absolute Position (CWE 797) and the list is still growing. That is due to current software engineering practice, which has generated software defects at a relatively constant rate for the past 40 years [10]. Nonetheless, those defects now cost the average U.S. Corporation $22 million dollars annually [9]. Worse, they leave Department of Defense (DoD) systems, as well those of all government and industry, susceptible to attack.

There is general recognition that software engineering practice can best be improved through education. In fact, the establishment of a National Cyberspace Awareness and Training Program was among the three highest priorities in "The National Strategy to Secure Cyberspace," which was published in 2003 [4]. This priority recognizes two of the barriers to the improvement of cybersecurity as "a lack of familiarity, knowledge, and understanding of the issues" and "an inability to find sufficient numbers of adequately trained...personnel to create and manage secure systems" [4]. One of the priority's major initiatives is to "foster adequate training and education programs to support the Nation's cybersecurity needs" [4].

The aim of these initiatives was to guarantee that software assurance practices would be embedded in the day-to-day actions of the overall workforce [5]. The

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

problem with software assurance is that there was no single point of reference to "guide the development and integration of education and training content relevant to software assurance" [15]. That led the Department of Homeland Security to publish a 387-page Common Body of Knowledge to Develop, Sustain and Acquire Secure Software [15]. That document, which is commonly called the CBK, specifies a comprehensive set of recommended practices for secure software assurance. These range from "heavyweight design methods" to "model contract language for vendors." The problem is that none of these recommendations have made their way into common use.

The traditional means of disseminating any kind of new knowledge into Society at-large is through formally constituted education, training, and awareness programs [5]. Back in 2002, the National Strategy recognized that fact in Action Recommendation 3/6 [4], which states that "research and development efforts [should be conducted] in the general area of secure software assurance in order to - coordinate the development and dissemination of best practices for cybersecurity" [4].

The obvious question eight years later is, "How close are we to achieving that goal?" The two projects discussed in the rest of this article are designed to encourage higher education to promote more secure software teaching. Together these represent the first attempts to ensure that the principles and practices of secure software assurance knowledge are embedded in mainstream higher education processes.

ENSURING LEARNING CONTENT FOR THE FIELD OF SOFTWARE ASSURANCE

The problem with software assurance knowledge is that it is crosscutting rather than disciplinary. In essence, the knowledge base for software assurance is located in a range of traditional studies [11]. That includes such dissimilar areas as "software engineering, systems engineering, information systems security engineering, safety, security, testing, information assurance, law and project management" [11]. As a result, secure software assurance content might appear in many different places and be taught in many different ways in conventional education settings.

It is clearly unacceptable to approach the teaching and learning process in such a disjointed way. Therefore, the way educators promulgate secure software assurance knowledge has to be coordinated. In order to coordinate the teaching and learning process, a formal effort has to be made to integrate "software assurance

content . . . into the body of knowledge of each contributing discipline" [15, 17]. There are two practical barriers to achieving that outcome. First, it is not clear what specific knowledge and skills should be taught in each area. Second, there are no validated methods for delivering that knowledge once it *has* been identified.

Logically, the first step in integrating new knowledge into conventional learning environments is to identify, relate and catalogue what is presently out there. That is the goal of a project funded by the DoD and conducted at the University of Detroit Mercy. This project has attempted to identify and document any knowledge, from any source, that could be related to the assurance of software.

That knowledge came from all traditional computing disciplines, such as computer science, software engineering and information systems. Nevertheless, besides the strictly technical areas the project also incorporated the conventional areas of information security, as well as relevant knowledge from the behavioral and social sciences. The knowledge was obtained from all accessible public and private sector sources.

The resulting knowledge base is contained in DoD's National Software Assurance Repository (NSAR). The NSAR encompasses and relates all commonly accepted practices, principles, methodologies and tools for software assurance. It is managed by an automated online knowledge management system. The mind map that underlies the knowledge management system is roughly based on the Common Body of Knowledge [15]. However, to ensure the validity of the CBK framework, the mind map was fine tuned and validated by means of a classic Delphi study. That study was conducted using a panel of 11 nationally known experts in secure software assurance.

The knowledge base contains as many life-cycle methodologies and tools for assuring software as could be identified. It also itemizes all related supporting principles and concepts that are aimed at increasing the assurance and security of internally developed and sustained software. That also includes products and services purchased from outside vendors. The knowledge base is evolutionary and inclusive. Thus, as the literature of the field expands, or new sources are identified, that material will be catalogued and added to the current knowledge base.

DEVELOPING PEDAGOGY

Nevertheless, the actual purpose of the University of Detroit Mercy project was not simply to gather knowledge. The goal was to ensure the teaching of secure software topics in all appropriate education, training and awareness settings. In support of that goal, the project has packaged the contents of its knowledge base into discrete learning modules. These modules are designed to facilitate the efficient transfer of software assurance knowledge into all relevant teaching and learning settings. As a result, the modules can be incorporated into a wide range of teaching and learning environments. They are appropriate for traditional graduate, undergraduate, community college, and even high school education, as well as training and awareness applications.

The modules are intended to be discrete, standalone learning artifacts capable of conveying all of the requisite know-how for a given topic. At a minimum, each module can be delivered in a conventional classroom. However, the modules embody supporting material that also allows them to be delivered in a range of asynchronous and web-enabled learning environments. That flexibility facilitates the efficient transfer of new workforce skills and practices to all types of education, training and awareness settings.

Each module conveys a logical element of software assurance practice. The entire collection of these modules is mapped to the body of knowledge contained in the knowledge base . Since that knowledge base is structured on the most commonly accepted model for secure software assurance practice, the DHS Common Body of Knowledge [15], this mapping provides precise guidance about the places where the newly developed instructional content fits within the commonly accepted understanding of the correct elements of practical software assurance work.

Each of the actual teaching modules incorporates a set of conventional learning artifacts, which are easily recognizable to traditional educators. Every module includes (1) a table of learning specifications, (2) presentation slides for each concept contained in the module, (3) an evaluation process, (4) any relevant web-enabled supporting material and (5) a model delivery system. There is also an accompanying pedagogical methodology for each individual learning module. That is, every module incorporates a validated set of teaching tools. These tools are optimized to ensure the maximum knowledge transfer for all potential teaching settings.

MAPPING FOR BROAD SCALE INTEGRATION

In order to ensure integration into conventional higher education curricula, the University of Detroit Mercy project has formally mapped all of its secure software assurance courseware modules to the standard set of computing topics specified for three of the five computer disciplines in the Computing Curricula 2005 standard (CC2005) [19]. This standard is a joint authorization of the Association for Computer Machinery (ACM), Institute for Electrical and Electronic Engineers (IEEE) and the Association for Information Systems (AIS). Since these are the three associations that are responsible for developing and overseeing computing curricula worldwide, CC2005 can be considered to be exhaustively authoritative.

The elements of secure software assurance practice were mapped from the CBK to the generally accepted curricular recommendations as itemized in Computing Curricula 2005 [19]. The aim of the mapping process was to identify where specifications for secure practice contained in the CBK fit within the recommendations for curricular content for each of the disciplines of computer science, software engineering and information systems. The mapping itself was a keyword-based process, utilizing the terms from the curricular requirements contained in tables 3.1 and 3.2 of the CC2005 as the search criterion. Where instances of that term were found in the CBK, anecdotal analysis was employed to determine the intent of the term with respect to the discussion of secure software assurance. Those intents were noted, aggregated and then categorized into highly specific concepts for secure software assurance that had to be communicated along with the teaching of each of the conventional CC2005 curricula elements. The detailed mapping of concepts to recommendations was used to tailor the integration of the associated secure software assurance teaching module to support or facilitate the specific intent of that term.

The project provides a detailed specification of where each learning module best fits within CC2005's curriculum. Along with that specification it also provides a justification for why the module was placed where it was in that particular curriculum. The justification is based on the mapping between the module and the recommended topics for a standard computer science, software engineering and information systems curriculum. For instance, the project provides specific recommendations for the precise place in an information systems curriculum where new secure software assurance content could be added to current testing topics. The justification is necessary to help individual curriculum designers understand where the learning module should be placed in their curricula. The justification also facilitates the integration and acceptance of that module within the traditional higher education and training communities.

MASTER OF SOFTWARE ASSURANCE REFERENCE CURRICULUM

The second education initiative to support the National Strategy focused primarily on development of a master of software assurance reference curriculum. The Software Engineering Institute at Carnegie Mellon University is leading this ongoing education effort, in support of the DHS's National Cyber Security Division. This is a particularly important focus because much of the body of knowledge in secure software assurance is based on a foundation of software engineering principles and practices. This project specifies a set of topics and all of the attendant prerequisite knowledge and requirements needed to ensure a properly educated software assurance professional. If differs from the prior project in that it identifies just the key knowledge elements required to produce a well educated practitioner and structures those elements into a comprehensive curriculum.

The curriculum development team includes technical staff from the SEI and faculty from a number of universities, including international representation. The reference curriculum includes guidelines that were used to develop the curriculum, prerequisites and proposed outcomes, curriculum architecture, a curriculum body of knowledge, implementation guidelines and a glossary of terms. A number of existing artifacts, including the software assurance guide to the body of knowledge [15], the recent graduate software engineering curriculum guidelines [18] and the older SEI reports on graduate software engineering education [7, 2], are inputs to the project. The team also referenced the SWEBOK [8] as needed, as software engineering knowledge is fundamental to software assurance. The project was inspired in part by the DHS Build Security In (BSI) website https://buildsecurityin.us-cert.gov, which contains articles providing practical advice on software assurance to practitioners. It is this practitioner focus that is central to the curriculum development effort. Another important resource for the team, also inspired by the BSI website, is the Software Security Engineering book [1], which was used along with the resources noted above, to identify software assurance practices to include in the curriculum.

In order to stay grounded, an invited review team for the curriculum was also involved in the process. In addition, some key industry managers and practitioners generously agreed to be surveyed in order to help validate our understanding of the desired outcomes. The curriculum also includes a detailed list of knowledge units and corresponding Bloom's taxonomy levels [3].

Establishment of a new degree program is a very ambitious undertaking. As a consequence, the team expects that some universities will elect to establish tracks or specializations in software assurance within existing master's degree

programs, such as in master of software engineering programs, rather than establishing a whole new degree program. Therefore guidance is provided on how to implement a track or specialization, and sample course syllabi are also provided. Team members at Monmouth University and Embry-Riddle Aeronautical University developed candidate implementation strategies for incorporating curriculum elements at their universities.

In addition to the master of software assurance reference curriculum, this project also produced a set of sample course outlines for software assurance courses that could be offered at the undergraduate level [12]. These courses might form an area of concentration within a computer science or software engineering undergraduate degree program for any prospective adopter.

CURRICULUM TRANSITION PLANS

There are a number of transition activities that accompany this curriculum work, as a curriculum is only the first step in effecting change in education. The team has started to work with the IEEE Computer Society towards professional recognition. Team members held a seminar to raise awareness among faculty at the Conference on Software Engineering Education and Training (CSEET) on March 10, 2010.The seminar will be distributed at a later time in Virtual Training Environment (VTE) format. The curriculum has been presented at a workshop in May on information assurance, at the DHS Software Assurance Working Group meeting in June, and also in the Information Assurance Capacity Building Program (IACBP), a faculty development program held at Carnegie Mellon University in summer 2010. Finally, the team will also form a group to work with and provide assistance to universities who wish to offer software assurance graduate courses. The team has also begun to tailor the curriculum into course offerings that would fit at the community college level.

Looking beyond these near-term activities, the team plans to develop more extensive faculty development workshops, course materials and course offerings in this area. They also hope to work towards certification in software assurance along the same lines as IEEE's Computer Software Development Professional (CSDP). There is an opportunity for distance education in this area, and eventually they may look at high school educational opportunities. The team feels that software assurance education is essential at all levels in order to ensure that software and software-intensive systems are developed with assurance in mind.

CONCLUSIONS

Our understanding of the knowledge that is needed to ensure capable software assurance is beginning to be shaped by these two projects. In that respect and particularly given the critical importance of secure software to the national interest, these two projects are working together to advance that process. Both projects are beginning to establish the foundation for moving what has historically been a field that is neither well understood nor well recognized into the mainstream of education, training and awareness.

The maturity of software assurance education will have advanced when MSwA programs and software assurance specializations within master of software engineering programs are widely available, when the database of software assurance materials is commonly used in software assurance course development, when software assurance offerings are standard elements of undergraduate computer science and software engineering degree programs, and when the software assurance body of knowledge has been codified.

In the case of the Carnegie Mellon project, software assurance master's programs and tracks provide an explicit curriculum of knowledge and skills necessary to produce a well-educated software assurance professional. Ultimately the curriculum will be supported by the needed course materials and course offerings. In the case of the Detroit Mercy project, every instructor in a computer-related discipline now has access to validated content and instructional materials that can be easily incorporated into existing courses.

In both projects, the boundaries and elements of the teaching and learning process for software assurance education are clarified. These two projects are initial steps in the long road to being able to assure the correctness and integrity of the nation's software with total confidence. Together they create a direction and foundation on which the future of the profession can be built.

REFERENCES

1. Allen, Julia; Barnum, Sean; Ellison, Robert; McGraw, Gary; Mead, Nancy. Software Security Engineering: A Guide for Project Managers. Addison-Wesley, 2008.

2. Ardis, M., and G. Ford. *1989* SEI Report on Graduate Software Engineering Education. SEI, Carnegie Mellon University. Technical Report CMU/SEI-89-TR-21. June 1989 http://www.sei.cmu.edu/reports/89tr021.pdf.

3. Bloom, B. S., ed. *Taxonomy of Educational Objectives: The Classification of Educational Goals: Handbook I: Cognitive Domain*. Longmans, 1956.

4. U.S. Department of Homeland Security, *The National Strategy to Secure Cyberspace*. Washington, DC: U.S. Department of Homeland Security, 2003. http://www.dhs.gov/xlibrary/assets/National_Cyberspace_Strategy.pdf.

5. Cogburn, Derrick. Globalization, Knowledge, Education and Training in the Information Age, United Nations Educational, Scientific and Cultural Organization. Director, Centre for Information Society Development in Africa and Africa Regional Director, Global Information Infrastructure Commission. Web. 1 Dec. 2009.

6. Conklin A., and G. Dietrich. "Secure software engineering: A new paradigm," in Proc. 40th Annual. Hawaii Int. Conf. System Sciences (HICSS'07). 272.

7. Ford, G. 1991 SEI Report on Graduate Software Engineering Education. SEI, Carnegie Mellon University. Technical Report CMU/SEI-91-TR-002. Apr. 1991 <http://www.sei.cmu.edu/reports/91tr002.pdf>.

8. IEEE Computer Society. Guide to the Software Engineering Body of Knowledge (SWEBOK). Web. <http://www.computer.org/portal/web/swebok>.

9. International Data Corporation IDC. Quality Problems Cost Software Companies Up to $22 Million Annually According to New Report. Coverity. Web. August 2008. Accessed January 2010. <http://www.coverity.com/html/press_story65_08_04_08.html>.

10. Jones, Capers, "A CAI State of the Practice Interview with Capers Jones", The IT Metrics and Productivity Institute, 2010, Accessed March 2010, http://www.itmpi.org/default.aspx?pageid=220

11. Mead, Nancy R., Dan Shoemaker, and Jeffrey Ingalsbe. "Integrating Software Assurance Knowledge into Conventional Curricula"." STSC Crosstalk 21.1 (2008).

12. Newman, Michael. Software Errors Cost U.S. Economy $59.5 Billion Annually. Gaithersburg: National Institute of Standards and Technology (NIST), 2002.

13. President's Information Technology Advisory Committee. Cybersecurity: A Crisis of Prioritization. Arlington: Executive Office of the President, National Coordination Office for Information Technology Research and Development, Feb. 2005.

14. Redwine, Samuel T., Ed. Software Assurance: A Guide to the Common Body of Knowledge to Produce, Acquire and Sustain Secure Software, Version 1.1. Washington: U.S. DHS, 2006.

15. Saltzer, Jerome, and Michael D. Schroeder. "The Protection of Information in Computer Systems." Communications of the ACM 17.7 (1974).

16. Shoemaker, D., A. Drommi, J. Ingalsbe, and N.R. Mead. "A Comparison of the Software Assurance Common Body of Knowledge to Common Curricular Standards." Dublin: 20th Conference on Software Engineering Education and Training, 2007.

17. Stevens Institute of Technology. Graduate Software Engineering 2009 (GSWe2009) Curriculum Guidelines for Graduate Degree Programs in Software Engineering. Web. Oct. 2, 2009. <http://www.gswe2009.org/>.

18. The Association for Computing Machinery (ACM), The Association for Information Systems (AIS), and The Computer Society (IEEE-CS). Computing Curricula 2005: The Overview Report, Computing Curricula Series. The Joint Task Force for Computing Curricula 2005, September 30, 2005.

19. U.S. Department of Homeland Security. Common Weakness Enumeration. Web. <http://cwe.mitre.org/>.

20. U.S. Information Assurance Technology Analysis Center (IATAC) Data and Analysis Center for Software (DACS). Department of Defense. Software Security Assurance State-of-the-Art Report (SOAR). By Karen Mercedes Goertzel, Theodore Winograd, Holly L. McKinley, Lyndon Oh, Michael Colon, Thomas McGibbon, Elaine Fedchak, and Robert Vienneau. IATAC with DACS, 31 July 2007. Web. <http://iac.dtic.mil/iatac/download/security.pdf>.

This material is based upon work funded and supported by Department of Homeland Security under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center sponsored by the United States Department of Defense.