Software Engineering Institute | Carnegie Mellon University

It's a Nice Idea but How Do We Get Anyone to Practice It? A Staged Model for Increasing Organizational Capability in Software Assurance

Dan Shoemaker

January 2009

ABSTRACT: This article presents a standard approach to increasing the security capability of a typical IT function. This five level model involves the development of a common set of security best practices, which are then deployed in a staged fashion to leverage an optimal security capability across the organization. At the lowest level the organization will have minimal assurance of security capability. At the highest level the organization can be trusted to produce products and provide services that are both dependable and secure. The article presents the practices and the maturity framework. It also discusses the practical mechanisms for implementing this model in a real world setting.

INTRODUCTION: ADDING A NEW CHALLENGE TO AN EXISTING PROBLEM

Software projects have always been a crapshoot, with the odds seriously stacked against the player. For instance, a recent Borland study found that approximately 33% of all projects are canceled prior to deployment, 75% of all projects are completed late and nearly 50% lack originally scheduled features and functions [Borland 2005, pg. 4]. In addition, it has been well documented that depending upon project size between 25% and 60% of all projects will fail; where "failure" means that the project is canceled or grossly exceeds its schedule [Jones 2005].

Worse, this is not exactly a new phenomenon. Throughout the 1990s, industry studies reported almost exactly the same outcomes. During that period, the average project exceeded its budget by 90% and its schedule by 120%, and fewer than half of the projects initiated during that time finished on time and on budget [Construx 1998]. Likewise, a similar study done by KPMG Pete Marwick found that 87% of failed projects exceeded their initial schedule estimates by 30% or more. At the same time, 56% exceeded their budget estimates by 30% or more and 45% failed to produce expected benefits [KPMG 1996].

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

The root cause of this less than sterling track record lies in the nature of software itself. Try building something that is invisible or accurately documenting something whose form only exists in the mind's-eye of a customer and you will understand the problem. Software development involves translating a customer's abstract ideas about functionality into tangible program behaviors. That makes it hard to ensure anything consistent and repeatable about the process or its outcomes. Given those conditions, it might seem miraculous that anything useful has ever been produced by the industry, but the problem is just getting started. Now the product also has to be secure.

When defects were just quality issues, the problem of buggy code had marketing and customer relations ramifications. Today, the right kind of defect, exploited by the wrong kind of adversary, can lead to a 9/11 style outcome. That is the reason why no matter what the current list of excuses for defects the "buck" has to "stop" when it comes to producing secure software.

MAINTAINING THE MINIMUM ORGANIZATIONAL CAPABILITY TO ENSURE SECURE SOFTWARE

In practical application, it is hard to make the business case for secure software. That is because organizations are composed of people and those people have varying degrees of capability. Variation isn't a problem if a particular level of performance isn't required because the company can always just keep patching their mistakes. However, where a specific level of proficiency is necessary to ensure a given level of performance, staff capability is a serious issue.

Staff capability is a major concern for business, since it is almost impossible to maintain a specific level of proficiency where constant turnover is a given. In that case, it becomes very important to adopt a well-defined process for developing and then assuring the organization's overall capability. Best practice is essential for secure software work, since it defines the proper way to perform a given task. However, all sorts of factors can influence how closely and consistently any particular worker will follow any given practice. As a result, a standard organizational process has to be instituted to ensure that all required best practices are executed as specified in the software assurance plan.

Creating that process is an organizational development issue. It is also a precondition for making the business case for software assurance. It is a given that the organization is only going to be as secure as the capabilities of its people. Therefore, any discussion about the costs and benefits of secure processes is pure

speculation until the people who will carry them out can be assured to be capable and willing to follow proper practice.

The term discipline simply denotes that a practice is reliably performed. Software assurance requires disciplined practice because in order to ensure a consistently secure product, all of the right practices have to be executed, by all participants, at all times, in a coordinated fashion. Accordingly, disciplined practice is essential to ensure that all of the products that are produced are secure all the time.

LEARNING TO DISCIPLINE CATS

In many cases, consistent performance of disciplined practices will ensure the general security of code. But those practices will also impose additional work requirements. Because it is more work, it cannot just be assumed that the people who do that work will naturally accept and follow those new additional requirements. Instead, it should be assumed that people within the software organization have to be consciously motivated to carry out additional security tasks.

Motivation is an important factor in the software assurance process. Motivation initiates, directs, and sustains all forms of human behavior. Motivation is the factor that ensures a person's willingness to consistently execute a given task or achieve a specific goal, even if the performance of the task itself is personally inconvenient. It also dictates the level and persistence of a person's commitment to the overall concept of secure software. Consequently, motivation is the factor that underwrites disciplined performance.

Motivation is typically geared to accountability. This accountability comes from the enforcement of appropriate-practice (not best-practice) policies. Appropriate-practice policies are developed and documented by the organization to guide the entire process by which the software is created. These policies are then monitored for compliance as part of the overall organizational accountability system. The accountability system then rewards appropriate actions and discourages the inappropriate ones. However, it is impossible to enforce accountability if all of the appropriate-practice policies are not known or understood. Therefore, the organization also has to ensure that all of its employees know what they are expected to do, as well as the consequences of non-compliance.

Being able to ensure that everybody in the organization understands his or her exact role, responsibility, and function is the single most critical requirement in ensuring that software is developed correctly. That is because, no matter how

potentially correct the security practices might be, if the people responsible for following those practices do not understand what they are supposed to do, there is almost no chance that the resulting work products will be secure.

Therefore, every organization has to undertake a deliberate effort to maintain every worker's up-to-date knowledge of his or her individual security duties and accountabilities. The need to have an organized function in place to ensure a continuous level of security knowledge is particularly essential in light of the fact that the workforce in most businesses is constantly changing. As trained workers leave, or change jobs, and untrained people are added, there has to be a consistent effort to maintain a requisite level of knowledge and understanding.

Consequently, besides perfecting the technical end of the software assurance process, another aim of the software assurance function has to be to make certain that the function that ensures that understanding operates as intended. The means that most organizations employ to meet that obligation are awareness, training, and education.

ENSURING THAT EVERYBODY IN THE OPERATION IS KNOWLEDGEABLE

In ordinary use, the combination of awareness, training, and education is often called an AT&E program. But each of these delivery models represents a different approach to learning. Each has a distinct application and each is characterized by progressively more rigorous and extensive learning requirements. Because of that progression, these approaches are normally rolled out in practical application as a hierarchy.

At the basic level, which is awareness, the purpose of the learning is very broad, but the learning requirements themselves are limited. The next level up, which is training, builds on the awareness function. However, the application of training is restricted to fewer people and the learning is more in depth. Finally, at the top of the hierarchy, which is education, the application might be limited to a few key people, but the learning requirements are very broad and in depth.

Awareness Programs

Awareness is the lowest rung in the ladder. Effective awareness programs ensure that all employees at every level in the organization appreciate the need for, and are capable of executing, disciplined secure software practice, in a coordinated manner. This meets basic software assurance aims. However, the requirement for

awareness varies across the organization. Awareness at the highest levels of the corporation sets the "tone at the top." So, awareness programs at the executive level are focused on ensuring the strategic policy awareness issues facing the organization, as well as the costs, benefits, and overall implications of security.

At all of the other levels, it is necessary to maintain a relatively high degree of awareness of relevant software assurance practices. Therefore, everybody in the organization must be made aware of the specific security requirements that apply to their position. In addition, they have to be motivated to practice security in a disciplined fashion. Thus, a good awareness program will

- strengthen motivation—the program must motivate all users to practice security
- ensure effective focus—the program must concentrate on relevant and appropriate topics
- maintain participant interest—the program must ensure that individual participants will continue to be interested in security
- underwrite capable performance—the program must ensure effective security
- integrate the content—the program must ensure the full integration of the proper set of practices

However, awareness alone does not ensure reliable software assurance practice. It is also necessary to ensure that individuals responsible for executing specific assurance functions, such as static tests and inspections, are knowledgeable in the precise requirements of their role. That implies the need for a greater degree of knowledge and capability than is typically provided by an awareness function. This is typically underwritten by formal training.

Training Programs

Training is organized instruction that is intended to produce an explicit outcome. Consequently, it emphasizes job-specific skills. The purpose of training is to make sure that organizational functions, which are required to ensure safe and secure software, are performed correctly. Training ensures that all participants in the process have the specific skills necessary to carry out their assignments and that the level of organizational capability is continuously maintained. Training can be expensive, but it is an effective way to guarantee capable long-term execution of software assurance processes.

Nonetheless, because it is based on skills rather than concepts, training is too narrow to ensure that the software assurance process itself is executed correctly

across the entire organization. Instead, training prepares individual workers to execute a series of steps without concern for the context or the reasons why the steps might be necessary. Training provides a quick and satisfactory outcome if the known threats to software never change or if adaptation to new threats is not required. However, most assurance situations are dynamic and complex. Therefore, training does not provide the overall strategic understanding that is necessary to establish a lasting security solution. A program of formal education is required to ensure that the organization's code is maintained continuously secure.

Education Programs

Education is oriented toward knowledge acquisition, rather than the development of short-term skills. It ensures an intelligent, rather than rote, response. It establishes understanding of the principles of secure software development as well as the critical thinking abilities that will be needed to evolve the software development process through a continually changing and uncertain threatscape. For that reason, the few individuals in the organization who are responsible for the long-term guidance of the security function must undergo formal and in-depth education in software assurance principles and practices.

Education can be distinguished from training by its scope, as well as the intent of the learning process. In a training environment, the employee acquires skills as part of a defined set of job criteria. In an educational context, the employee is taught to think more about the implications of what he or she is learning. The learner must be able to analyze, evaluate, and then select the optimum security response from all alternatives. Thus learners are encouraged to critically examine and evaluate the problem and to respond appropriately by tailoring fundamental principles into a solution that precisely fits the situation.

The practical aim of education is to develop the ability to integrate new knowledge and skills into day-to-day security practice. The specific outcome of an institutionalized education process is the ability of executives, managers, and workers to adapt to new situations as they arise. Given what has been said about the constantly changing nature of threats and vulnerabilities, this is an essential survival skill for the leadership of any organization.

INCREASING ORGANIZATIONAL CAPABILITY THROUGH AT&E

The outcome of a properly administered AT&E program is an increased level of organizational capability. This is a strategic concept. It is based on the achievement of five progressively more capable states of security:

- Recognition—the organization recognizes the need for security.
- Informal Realization—the organization understands informal security practices.
- Security Understanding—the security practices are planned and monitored.
- Deliberate Control—decisions about security practices are based on data.
- Continuous Adaptation—practices adapt to changes and are continuously improving.

The levels of capability are progressively achieved through targeted awareness, training, and education processes.

Security Recognition

The most fundamental level is simple Recognition. Here, the majority of the participants are able to recognize that secure software is a valid and necessary concern. Until that fundamental state of recognition is achieved, the organization is essentially operating without any concept of secure practice. Once adequate recognition is established, however, individual members begin to understand that exploitation of coding flaws is a concern. This may not necessarily be in any deliberate or actively organized fashion, but it does involve a persistent underlying appreciation that security practice is necessary.

Informal Realization

At the next level, Informal Realization, members of the organization become more conscious of the need to ensure against software defects. Every worker is aware that those concerns exist. Workers might also follow rudimentary assurance procedures in response to that understanding. Thus, this level is supported by a more involved awareness program.

The awareness program that underlies informal realization presents security issues that have been expressly identified as concerns, such as buffer overflows. It might also present general practices to address these concerns, such as parameter checking. This is done on an ad hoc or informational basis. The best practices that are designed to avoid common coding errors are not sufficiently specific and their performance is not organized well enough to ensure that security is embedded in the standard operation. That happens in the next step.

Security Understanding

The third stage, Security Understanding, is the first level where a consciously planned and formal security effort takes place. At this stage, the organization understands and acts on a commonly accepted understanding of the need for some form of formal security practice. The response might not be extensive and it is often dependent on individual willingness, but it is recognizable in that standard software assurance procedures are planned and documented in a systematic fashion.

The fact that security procedures have been formally documented allows the organization to implement a training program. Training is typically done to enforce understanding of the requisite security practices that are associated with each generic role. For instance, there might be targeted programs for executives regarding the business consequences of exploitation, a different one for managers aimed at implementing monitoring and control functions, and another for workers aimed at ensuring that best practices are followed.

The worker training programs might be subdivided by operation, such as development, versus acquisition, versus sustainment. The aim of each program though is to foster understanding of the security procedures that are appropriate to that role or function. These programs are generally not oriented toward ensuring specific skills beyond the understanding of the security practices that are required to carry out basic work. That is done in the next stage.

Deliberate Control

The fourth stage, Deliberate Control, is typical of a well-organized software assurance operation. Deliberate control is characterized by an institutionalized software assurance response that is built around providing a tailored set of skills for each relevant position. These skills are defined and managed based on a precise knowledge of the requirements of each individual's role in the organization.

The execution of these security tasks is monitored using quantitative measures of performance, such as defect density. Deliberate control is enforced by defined accountability. Because it is objectively monitored, the security operation is fully managed by the organization's top-level executive team. At this level of functioning, the organization can be considered both safe from common threats and actively practicing the steps that are necessary to maintain that requisite level of security.

This state comprises a targeted mix of training and education. Coordination and administration of the program is designed to achieve specific assurance out-

comes. The training and education program communicates the precise knowledge and skills that are needed to correctly perform specific security practices that are required by each function. This is reinforced through periodic retraining.

Training at this level is a carefully planned activity that requires many of the activities performed by the personnel security function, such as job definition, job classification, and privilege setting, to make it successful. The outcome provides a very high level of carefully controlled assurance. However, this is not yet the highest level of education possible.

Continuous Adaptation

At this final and fifth level, the software assurance function is fully optimizing. It not only carries out all of the practices necessary to ensure secure code within the dictates of the situation, but it continues to evolve those practices as conditions change. Organizations at this level are capable of adapting to new threats as they arise. That allows them to maintain consistently effective software assurance countermeasures as well as an active response to any new threat. They are safe from harm because they are protected from all but the most unforeseen events, and they are capable of a rapid and meaningful reaction to any threat that might occur.

This stage is achieved by ensuring that workers master the critical thinking skills necessary to identify and solve problems. That requires a high level of knowledge of the elements and requirements of the field, as well as the thought processes to allow people to adapt these principles to new situations as they arise.

The classic mechanism for reaching this level of competence is a well-designed educational program. Skill training might also be among the factors needed to achieve this level. Nevertheless, the integration of that knowledge into the capability to respond correctly to new or unanticipated events falls within the realm of education.

SOME GENERAL CONCLUSIONS

A formal and well-run AT&E program is a critically important advantage for a software organization because, in the end, no matter how well intentioned your staff might be, without sufficient knowledge in secure coding practice, your assurance capability will be limited.

The type of dynamic approach outlined here is an ongoing commitment. Therefore, it cannot be stressed enough that the organizational entity that is given the responsibility for training must constantly monitor and control the development of the program and the personnel resource through formal assessment and review.

The maturation of an AT&E program is a continuous activity that flows from the refinement of security knowledge as well as new knowledge gained through performance of security activities. The training operation requires a total commitment by the organization, particularly the top-level people, to maintaining a dynamic and complete understanding of all necessary requirements and capabilities. This is essential in order to develop the programmatic responses required to meet the demands of an evolving threatscape. Nonetheless, if this dictate is adhered to, AT&E can provide the operational backbone necessary to ensure that the organization will stay secure.

REFERENCES

[Borland 05] Borland Software Corporation. "Software Delivery Optimization Maximizing the Business Value of Software," Borland Vision and Strategy Solution Whitepaper, 2005.

[Construx 98] Construx Software Builders website, www.construx.com, 1998, cited in Shoemaker, D. and Vladan Jovanovic, "Engineering a Better Software Organization," Quest Publishing House, Ann Arbor, 1998.

[Jones 94] Jones, Capers. *Assessment and Control of Software Risks*. Prentice-Hall: Englewood Cliffs, NJ, 1994.

[Jones 05] Jones, Capers. "Software Quality in 2005, a Survey of the State of the Art." Software Productivity Research, Marlborough, Massachusetts, 2005.

[KPMG 98] KPMG Technology and Services Group website, www.kpmg.ca 1996, cited in Shoemaker, D. and Vladan Jovanovic, "Engineering a Better Software Organization," Quest Publishing House, Ann Arbor, 1998.

[McConnell 07] McConnell, Steve. "The Business Case for Software Development." Construx Software Builders Inc., 2007.

[McGibbon 99] McGibbon, Thomas. "A Business Case for Software Process Improvement Revised." DoD Data Analysis Center for Software (DACS), 1999.

[Paulk 93] Paulk M., B. Curtis, M. Chrissis, & C. Weber. "*Capability Maturity Model, Version 1.1*," Technical Report, Software Engineering Institute, Carnegie-Mellon University, 1993.

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.

DM-0001120