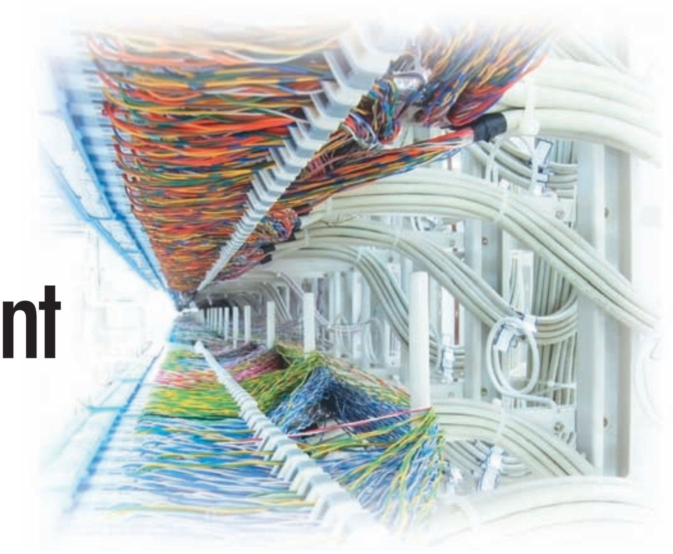
Are Companies Actually Using Secure Development Life Cycles?



David Geer

As threats to applications have increased, developers have begun including security in their software design. Secure development life cycles are methodologies for accomplishing this, but are companies actually using SDLs?

Traditionally, developers design software to accomplish a set of functions and then later add—or don't add—security measures, according to Robert Thibadeau, chief scientist for security-software vendor Wave Systems.

After all, said Marisa Fagan, project manager with consultancy Errata Security, "It's no secret that software companies value features over security."

Over time, though, Thibadeau noted, as security threats have increased, some developers have begun including security in their application design.

In 2004, organizations began releasing processes for building security into the software-development life cycle. According to Fagan, these processes have been called secure coding programs, software assurance, app sec, and secure development life cycles (SDLs).

The first process was Microsoft's Security Development Lifecycle, initially implemented internally but openly available since 2004.

Since then, organizations have released processes such as Microsoft SDL-Agile; the Software Assurance Maturity Model (SAMM); the Building Security in Maturity Model (BSIMM); the Secure Software Development Lifecycle (SSDL); and the Comprehensive, Lightweight Application Security Process (CLASP).

All provide a road map for software vendors and individual developers to include security as they design applications, not as an afterthought.

Errata recently released a survey of information-security and software-development consultants, managers, and developers designed to determine how widely SDLs are used.

The survey determined that companies are starting to utilize these security approaches more. In fact, 81 percent of respondents said they were aware of formal methodologies.

This was encouraging, Fagan said, but adoption rates are still low, with only 30.4 percent of respondents using a formal methodology.

And the technology still faces challenges such as cost, performance overhead, and lack of management support.

THE ERRATA SURVEY

Errata conducted the survey on its website. The company advertised it in the Errata Security blog and in tweets; at the BSidesSanFrancisco security conference; and at the RSA Conference, a leading security event.

The company conducted the survey to find out which organizations aren't implementing security in software development and why, and to help Errata make a better business case for software assurance, Fagan explained.

"Security saves companies money because they don't have to do incident response or deal with damage to their reputations," she said.

The survey received 46 responses. While the survey has a margin of error of about 14.5 percent, Fagan explained, "if you are the type of person who would attend an RSA Conference, here is what your peers are thinking."

Published by the IEEE Computer Society

SDL adoption levels

Half of survey respondents said security is always a concern in software development; and only five participants, representing 10.9 percent of the total, said an SDL is unnecessary. However, only 14 respondents, 30.4 percent of the total, said they use a formal SDL, according to Fagan.

"Almost all organizations have their own custom implementation or interpretation [of an SDL]," said Danny Allan, IBM Rational Software's director of security research.

However, said Fagan, "the market is new, and companies are still waiting to find a methodology that fits their program."

Comments that respondents added to their Errata survey responses indicated that companies with fewer than 10 software developers have implemented formal SDL methodologies at a higher rate than those with 100 or more developers.

"Management is the main driver for adoption of secure development methodologies," Fagan explained. "It is easier for a manager to lead a smaller team."

"A larger company will have policies, procedures, infrastructure, a large code base, and other things to transition. A smaller organization could move faster on this," said Steve Lipner, senior director for security engineering strategy with Microsoft's Trustworthy Computing Group.

To adopt or not to adopt

As Figure 1 shows, of all Errata survey respondents, 11, representing 23.9 percent, said formal SDLs are too time consuming; four, representing 8.7 percent, said they're unnecessary; two, representing 4.3 percent, said they're too expensive; and seven, representing 15.2 percent, said they require too many resources. Nine, representing 19.6 percent, said they weren't aware of SDLs.

"Part of the [SDL adoption] problem is the sheer number of common

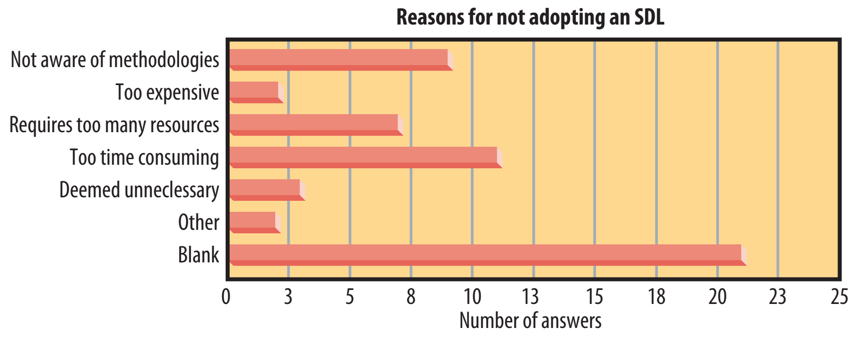


Figure 1. Errata Security survey respondents gave reasons for not adopting an SDL.

software weaknesses, which [include] about 7,000 items, How do smaller vendors without deep pockets afford to fix these?" asked Robert S. Seacord, secure coding team lead for CERT, a security-research organization based at Carnegie Mellon University's Software Engineering Institute.

"Our survey showed that the responsibility for adding security falls on upper management," said Fagan.

Only eight of 46 survey respondents said their company sends management to security training. Therefore, management frequently doesn't understand the problem, explained Eugene Schultz, consultancy Emagined Security's chief technology officer. If securing code causes cost overruns, management wants to write it off as acceptable risk, he said.

"We still see members of management citing resource requirements as a main reason they're choosing not to use secure coding," said Fagan. "Until more third-party analysis is done on the actual costs of integrating security activities, we will see resistance."

"The cost of implementing secure development methodologies isn't nearly as expensive as developers assume," noted Chris Wysopal, software-security vendor Veracode's chief technology officer.

But, added Seacord, "customers aren't really clamoring for more security. If they have a choice between getting the software product today with more functionality and lower cost or getting it delivered later at a higher cost with less functionality and more security, consumers will go for the former."

MULTIPLE APPROACHES

"The root cause of software vulnerabilities is found in the early stages of the software development life cycle. The majority of vulnerabilities could easily be taken out at this stage," Fagan stated.

There are multiple SDL approaches to accomplishing this.

Microsoft SDL and SDL-Agile

Microsoft SDL—available for free at www.microsoft.com/security/sdl/default.aspx—adds specific security-related steps to the process of developing, testing, and releasing software, the company's Lipner explained.

He said the most important step is using threat modeling to identify application vulnerabilities and then determining the best way to address them. The approach also uses a compiler to analyze code and find potential problems.

Static analysis examines code without executing the application.

Fuzz testing finds software problems by adding invalid, unexpected, or random data to an input to see if the program fails.

Microsoft's SDL-Agile (http://go.microsoft.com/?linkid=9708426) is tailored to the agile-development framework. The company pared back its standard SDL requirements to a core subset that could be completed within the several-week subcycles within which agile development typically works.

According to the Errata survey, Microsoft's SDL and SDL-Agile are the best-known secure-development-life-cycle methodologies, as Figure 2 shows.

SAMM

Pravir Chandra, director of strategic services for security-assurance vendor Fortify Software, began the OpenSAMM Project, which released SAMM'S beta version in 2008 and version 1.0 (www.opensamm.org/2009/03/samm-10-released) last year.

Chandra has since given SAMM to the Open Web Application Security Project (OWASP; www.owasp.org) to manage.

"SAMM is designed to help organizations formulate and implement a strategy for software security that is tailored to the specific risks facing that organization," said Chandra.

SAMM, available for free, is divided into four basic software-development functions, each with three security practices that are divided into three sophistication levels.

Companies can determine which of the practices and sophistication levels are most appropriate for them.

SAMM calls for activities such as threat assessments, secure architecture practices, software checking and testing, and vulnerability management.

BSIMM

Fortify and software-security consultancy Cigital released BSIMM (www.bsimm2.com) in 2009, noted Fortify chief scientist Brian Chess.

"[We] did a series of in-person interviews with people in charge of software-security initiatives at well-known places such as Microsoft, Wells Fargo, and 28 other companies," explained Chess. "We wrote what they told us, then organized those practices and published them as the BSIMM."

BSIMM, which is available for free, is organized much like SAMM, said Chess. For example, BSIMM's Software Security Framework has four software-development areas, each with three security practices.

The methodology calls for steps such as security training; threat modeling; security design, analysis, and assurance; architecture analysis; security testing; integration with existing measures; and configuration and vulnerability management.

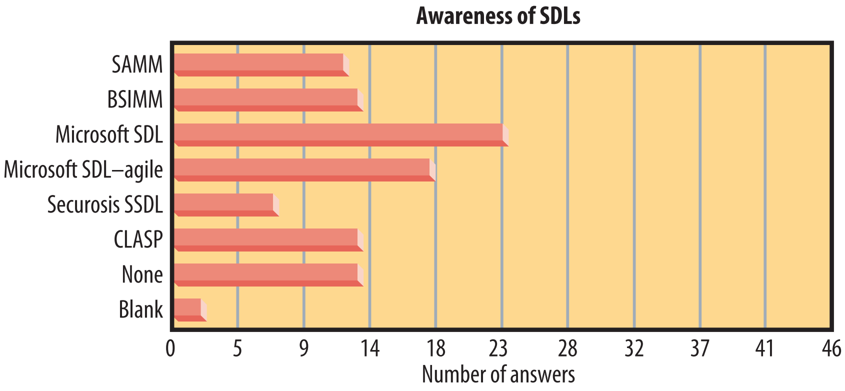


Figure 2. Microsoft SDL and SDL-Agile were the secure-development-life-cycle methodologies Errata Security survey respondents were most aware of.

Securosis' SSDL

SSDL, created by security consultancy Securosis, consists of blog posts about agile software development posted by Adrian Lane, the company's security analyst and chief technology officer.

Agile development doesn't address security, and it moves too quickly through short subcycles to make the process easy, he explained.

SSDL—available at http://securosis.com/blog/comments/agile-development-and-security, http://securosis.

com/blog/comments/comments-on-microsoft-simplified-sdl, and http://securosis.com/blog/comments/structured-security-prgram-meet-agile-process—deals with issues such as training and testing.

SSDL emphasizes prioritizing security issues and adjusting agile processes modestly. For example, security testing could take place over more than just a two- or four-week development subcycle.

CLASP

OWASP released CLASP in 2006. This approach addresses secure-software issues in the critical first stages of the software-development life cycle. It uses seven best practices including instituting security-awareness programs for developers, architects, project managers, people who specify requirements, and even executives, says Fortify's Chandra, who directed work on CLASP.

In addition, the approach entails capturing a project's security requirements and assessing applications for security weaknesses via measures such as threat modeling.

CLASP, available for free at www.owasp.org/index.php/CLASP, also includes an online community that lets users add to the project.

Others

In 2006, the nonprofit BITS industry consortium, part of the Financial Services Roundtable, released the Shared Assessments Program Agreed upon Procedures. The program (www.sharedassessments.org) is now managed by the Santa Fe Group consultancy. Some of the procedures relate to secure development, noted Santa Fe Group senior vice president Michele Edson.

TopCoder—a global organization that conducts contests among developers throughout the world to deliver software, which it then licenses for profit—released the TopCoder Methodology in 2002, according to Mike Lydon, the group's chief technology officer. The approach includes guidelines for building security into applications.

The US National Security Agency released its Information Security Evaluation Methodology, which includes information on secure software development, for the NSA, other government agencies, and contractors.

The US Department of Defense Information Assurance Certification and Accreditation Process, released in 2006, applies risk management to military-related information systems, particularly software development, said Morely Haber, vice president of product management for security vendor eEye Digital Security.

THE LIFE CYCLE'S LIFE CYCLE

Major challenges facing software assurance include cost and, more importantly, complexity, according to Errata's Fagan.

She cited the lack of mature software-development life cycles into which organizations can build software assurance, the paucity of security-related education and training, and a failure to give security experts the authority to delay projects if there are problems.

Only the 10 percent of companies that are the most technically sophisticated are adopting SDL, said Veracode's Wysopal. Even these companies are adopting SDL for only the most critical 10 percent of their applications, he added.

"But I see this as the start of a 10-year process to embed security into the standard software-development process," he said.

"We must show the business logic in secure coding, [via] saving money in incident response and the marketing opportunities for robust code," said Fagan.

"Part of the solution is to make software-security technologies and processes require less time and less security-specific expertise. Less disruption to the development schedule will allow quicker adoption," noted Wysopal.

Over the next few years, CERT's Seacord predicted, existing developer tools may be strengthened to better include security considerations.

Also, there should be even more secure-development methodologies, said Fagan. "We're already seeing this trend with the release of Cisco's SDL," she noted.

However, predicted Ryan English, practice principal with Hewlett-Packard Professional Services, there will be some methodology consolidation.

Ultimately, though, said IBM Rational's Allan, SDL adoption and use will depend on regulatory requirements and executive mandates.

As education and awareness increase and secure components are inherently built into the more common frameworks, he added, many of the troubling SDL implementation issues will be reduced.

"Most executives want to see a convincing cost/benefit argument [for secure software]," said Seacord, "and there is no empirical evidence that any one secure-development approach is better than any other."

Said Fagan, "When customers demand more security, management can make a business case for it and will begin to improve their development process. We've seen several breaches in the news lately, and we can expect customer expectations will be shifting in response. These types of vulnerabilities could be prevented with a secure development life cycle." C

David Geer *is a freelance technology writer based in Ashtabula, Ohio. Contact him at david@geercom.com.*

Editor: Lee Garber, *Computer*, l.garber@computer.org