

Chapter 3.9

Assimilating and Optimizing Software Assurance in the SDLC: A Framework and Step–Wise Approach

Aderemi O. Adeniji

University of North Carolina at Charlotte, USA

Seok-Won Lee

University of North Carolina at Charlotte, USA

ABSTRACT

Software Assurance is the planned and systematic set of activities that ensures software processes and products conform to requirements while standards and procedures in a manner that builds trusted systems and secure software. While absolute security may not yet be possible, procedures and practices exist to promote assurance in the software lifecycle. In this paper, the authors present a framework and step-wise approach towards achieving and optimizing assurance by infusing security knowledge, techniques, and methodologies into each phase of the Software Development Lifecycle (SDLC).

INTRODUCTION

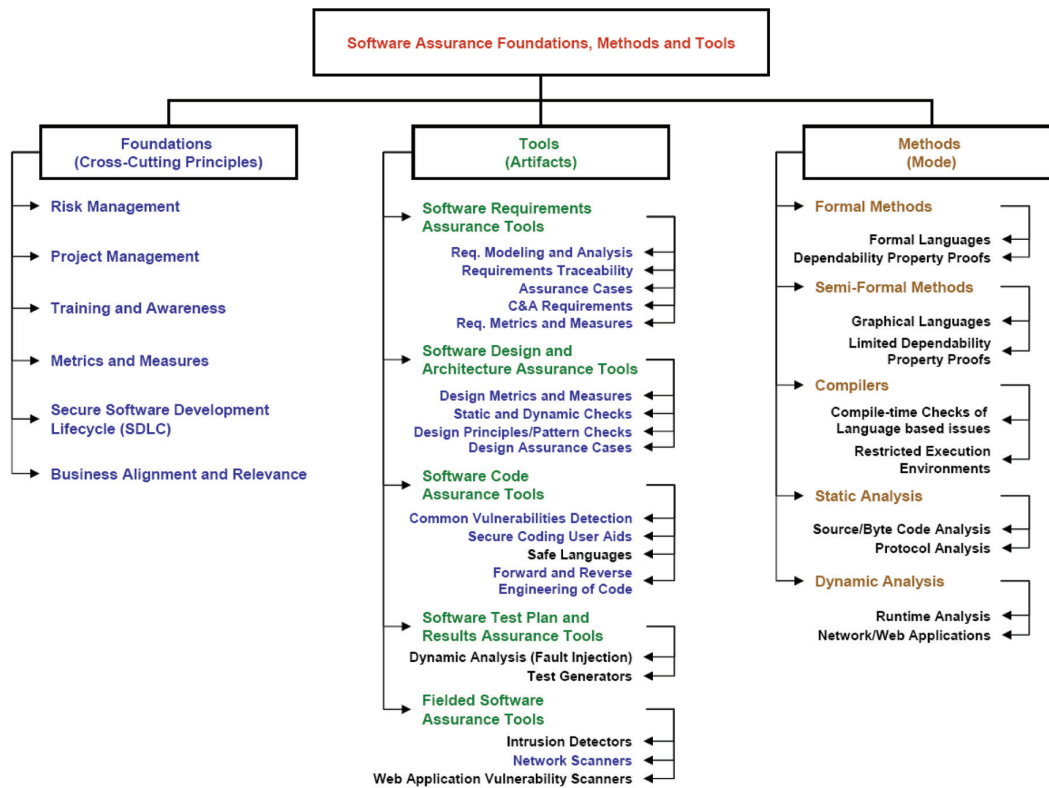
Software Assurance is steadily gaining ground in the Information Technology industry. The notion of proving secure software while supporting organization and system priorities is appealing to developers and customers alike. Software as-

surance aims to provide *justifiable confidence* that software is trusted to behave as intended even amidst intentional and unintentional attacks (Goertzel et al., 2007; Sinclair, 2005).

Based on experiences and lessons learned from designing a graduate level software assurance curriculum, assurance optimization is aided by implementing techniques in each phase of the

DOI: 10.4018/978-1-61350-456-7.ch3.9

Figure 1. Software assurance foundations, methods and tools



SDLC. The intent of this paper is to share a strategy for integrating software assurance throughout the lifecycle in a methodical manner, proving a secure and trusted system. Several of the foundations, tools and methods used for optimization, shown on Figure 1, will be highlighted throughout the context.

BACKGROUND

Software is the core component of modern products and services, supporting business operations for all sectors of life. With each software use, there are factors which contribute to increased mission risk including: project size and complexity, attack sophistication, and use of third-party vendors (Ellison, 2006; McGraw, 2005). Dependence on this software makes security a primary concern (Allen et al., 2010). Software Assurance is achieved by

understanding the mechanics of software built and/or acquired and incorporating validation tools and strategies into each phase of its lifecycle to build a trusted and secure product. Figure 2 diagrams this process, showing a step-wise approach for infusing assurance techniques into the SDLC by outlining approaches and artifacts produced. Knowledge gained from performing each step in a methodical and well-defined manner is carried forward, resulting in progressive learning. This is an iterative process, as education acquired from one phase will allow for more intelligent review in another. Assurance optimization can be achieved by mitigating common weaknesses in software throughout the aforementioned process. Peter G. Neumann identified nine sources of problems in computer systems (1994). A framework for assurance in the SDLC has been developed and these vulnerability sources will be addressed in appropriate phases, shown in Table 1.

17 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage:

www.igi-global.com/chapter/assimilating-optimizing-software-assurance-sdlc/62469

Related Content

Learning With Software-Defined Area

Anurag Tiwari and Suneet Gupta (2018). *Innovations in Software-Defined Networking and Network Functions Virtualization* (pp. 291-305).

www.irma-international.org/chapter/learning-with-software-defined-area/198204

Leveraging Big Data and Advanced Technologies for Enhanced Sustainability in Healthcare: An IPO Model Approach

Mary Metilda Jayaraj and Subbulakshmi Somu (2025). *Navigating Cyber-Physical Systems With Cutting-Edge Technologies* (pp. 287-308).

www.irma-international.org/chapter/leveraging-big-data-and-advanced-technologies-for-enhanced-sustainability-in-healthcare/363635

Requirements Refinement and Component Reuse: The FoReVer Contract-Based Approach

Laura Baracchi, Alessandro Cimatti, Gerald Garcia, Silvia Mazzini, Stefano Puri and Stefano Tonetta (2018). *Computer Systems and Software Engineering: Concepts, Methodologies, Tools, and Applications* (pp. 1397-1432).

www.irma-international.org/chapter/requirements-refinement-and-component-reuse/192929

Object Oriented Software Testing with Genetic Programming and Program Analysis

Arjan Seesing and Hans-Gerhard Gross (2012). *Computer Engineering: Concepts, Methodologies, Tools and Applications* (pp. 992-1006).

www.irma-international.org/chapter/object-oriented-software-testing-genetic/62493

Theory Driven Modeling as the Core of Software Development

Janis Osis and Erika Nazaruka (Asnina) (2021). *Research Anthology on Recent Trends, Tools, and Implications of Computer Programming* (pp. 88-107).

www.irma-international.org/chapter/theory-driven-modeling-as-the-core-of-software-development/261023