A New Application of CONOPS in Security Requirements Engineering

by Darwin Ammala
MPI Software Technology Inc.

The Concept of Operations (CONOPS) document, IEEE Standard 1362-1998, is a powerful tool for communicating a customer's vision for a new system. Normally used for describing full systems, CONOPS can also be used to address one single aspect—such as security in a large-scale project. This paper reports on a recent Navy contract effort, which demonstrated this use of the CONOPS. This paper will describe and analyze the results of this effort.

The IEEE CONOPS document [1] will gain in popularity, once the software engineering community discovers its flexibility, extensibility, and versatility. This paper summarizes a recent case of employing the CONOPS document to describe the desired multilevel security, and high-performance computing infrastructure characteristics to be included in the Navy's next generation combat vessels. The DD21 (21st Century) land attack destroyer is one such ship undergoing design competition. The Navy's Small Business Innovations Research solicitation topic [2] identified this vessel as a candidate for the most advanced multilevel security in a computer-intensive environment.

User Requirements

Traditionally, the Software Requirements Specification (SRS) has been developed as the primary document for stating user requirements. The SRS usually is produced by the developing organization following discussions with the potential user of the system and analysis of the requirements gleaned from these discussions. This document is well suited for use by its authors, but may be of less value to others, such as when users are presented the document for their comment. Users do not express their requirements at the level of specificity found in completed SRS documents.

An ideal SRS for a software-intensive system can be characterized as having requirements that completely capture a problem, is devoid of design evidence, and has succinctly stated requirements. This type of document would generally overwhelm the user in the sheer density of detail, repetitiveness of the language, and the total number of requirements stated. Most users would not be readily able to determine if the system described in this manner would truly address their original needs. The user's confidence in the development process can be fortified if he can see steady transitions of requirements from his expressed concepts, to requirements to specification, to code, to product.

> "With the new Unified Modeling Language now standardized, convergence of development methods is more likely. From a security, software, development perspective, this could improve the state of software security engineering."

People are most comfortable with describing things in the language of their problem domain. The primary objective of software engineering is to build the product that the user needs (validation); after this is to build that product correctly (verification). A high priority should be placed on allowing the user to directly express desires and ideas for needed capabilities of the expected system. The user may also provide some insight into specific testing criteria to be considered. Early attention to testing is also relevant during concept definition, as the system testing is a peer *off-core* development activity in the software development life cycle [3].

Security Requirements

A compounding requirement problem arises when security requirements are considered along with functional requirements. The area of Systems and Software Systems Security underwent growth during the 1990s. The Internet's emergence spurred numerous e-companies to rapid growth and success, while the field of electronic commerce remains in its infancy. Like all software, security-relevant software often is designed and released with latent defects. With the user community's heightened awareness of security, the developer community will observe a raised concern for sound, functional security within the software products that they are hired to produce.

Developers of these products employ disparate development life-cycle approaches, many of which are described by Hassan Gomaa[4] Barry Boehm[5], Alan M. Davis [6], et al. These authors do not single out security development practices per se, but one can infer that a definable development process is followed. With the new Unified Modeling Language now standardized, convergence of development methods is more likely. From a security, software, development perspective, this could improve the state of software security engineering.

Users, in most cases, have only rudimentary knowledge of security, and thus, are less likely to be able to help articulate their concerns in a language other than that of their native domain. The user is less likely to understand documents, such as an SRS, written by a developer community having security domain knowledge due to the specialized jargon. Terms such as *mandatory access control*, *nonrepudiation*, and *ubiquitous login capability* are far from everyday usage.

This leads us to conclude that perhaps security-relevant software engineering would benefit from a more thorough and well-understood collection of requirements. This would ensure that the right product is being built correctly. The CONOPS document written in user language is an ideal vehicle for this purpose.

CONOPS Overview

A CONOPS is a user-oriented document that describes system characteristics for a proposed system from the users' viewpoint [7]. The CONOPS document is written in order to communicate overall quantitative and qualitative system characteristics to the user, buyer, developer, and other organizational elements. It describes the existing system, operational policies, classes of users, interactions among users, and organizational objectives from an integrated systems point of view [1].

The CONOPS is intended to aid in requirement capture and communication

of need to the developing organization. Posing the problems to be solved in the user's language ensures that the user can more accurately express the problem. The developers then have a good basis to begin the requirements refinements, and initial design of the system.

Bringing different communities of people together (users in their domain, and software developers in theirs) implies that communication will be a critical issue. The IEEE CONOPS standard does not stipulate whether the user or the developer must write the document [7]. There are tradeoffs in every scenario—whether the user or developer writes the document. Users will better articulate the desired capability in terms of their domain situation, while developers are more likely to be familiar with current computing technology, and would express the required capability in terms of technology. [7].

Once a draft CONOPS is written, it is presented to the user and developer organizations for review and comment. Ideally, the user should write the CONOPS; however, the user must first be taught the intentions, and guidelines for doing so. Thus, collaboration between the user and developer is paramount until the user organization has mastered the techniques. The IEEE standard for the CONOPS document [1] does a good job of explaining the purpose for each section. Additionally, not all sections of the document are required. An agreement between the user and developer can establish sections that are needed, and which ones could be omitted. In practicing due diligence, each of the sections addresses unique aspects of the system life cycle and should be addressed.

A Security CONOPS

This paper now turns to the case study of the use of the CONOPS to articulate the security relevant portion of a proposed new ship-based computing facility.

Need and Solicitation

The overarching requirement came from an SBIR Phase 1 solicitation [2]:

Problem: The Navy's new ships require the most advanced technology and security services, adaptable to quickly changing conditions, and functional at new levels of efficiency while carrying fewer sailors to operate them.

Objective: To develop techniques to address multilevel security in a complex, software-intensive system. Of particular concern is maintaining multilevel security while supporting a robust ability to migrate and reallocate tasks through a complex computer network architecture [2].

Successful Proposal

The approach to this problem was to propose a business process review dialogue with the sponsor that would elicit the requirements—both computing system, and computing system security—to produce a security-specific CONOPS document. The anticipated interplay was for the developer to produce the initial draft of the CONOPS, and allow the user representative an opportunity to comment upon and edit the draft.

This approach rationalized that in light of the great complexity of a modern-day destroyer; the security should receive early, concentrated focus.

Delivery

The contract team met with the sponsors, and learned that this system effort was the first to employ a revised procurement procedure within the Navy. In prior procurements, the developer was given statements of work and statements of requirements that were refined and worked with the contractor to produce the requirements baseline documents, including the SRS.

The new procurement model employed by the Navy places more degrees of freedom on the developing organization to develop the product in the way it does best, while working with the sponsor. We learned that little information was available on specific DD21 requirements, primarily because the competition phase between the two project teams was under way and 18 months from completion. These factors gave us incentive to be creative in general computing capabilities, and concentrate on developing an adaptable computer-intensive environment that also employs sufficient security to meet mission requirements. This gave the CONOPS a technology-oriented composition.

The work from this contract served a bilateral purpose. The prime focus was to define security and system functionality requirements for the customer's new class of ships. The secondary focus was to initiate the users to the use of a specialized CONOPS document. Since the developer derived the requirements, the user's true requirements can be obtained only from the user community's review and modification of the document

Analysis

Computer-intensive environments, particularly those that must also provide multilevel security protection and services, are software-intensive systems. Building such software systems requires the proper employment of requirements engineering practices and tools. Among these practices are requirements elicitation and refinement. In cases where the developer has limited access to the users, the developer can offer a preliminary CONOPS document. This preliminary CONOPS will encourage the users to comment, clarify, or produce a response document, which is determined to reflect their views of what is needed. Once each side has had a chance to contribute to the requirements concept, a round of more probing analysis and questioning should follow. Employing an interviewing technique along the format described by Joseph Groguen and Charlotte Linde [8] to isolate areas of requirements that need more clarification would be an efficient use of time. This type of *zooming in* isolates a specific topic deemed critical in the new system, and allows detailed exploration of the problem and requirements related to it.

The security requirements were selected as the focus of initial operational concept. This was based on existing best practice in computing and information systems security that the system's security is designed prior to the production [9, 10]. Not doing so would entail retrofitting security into a complex software system, which if inadequately designed would be a precursor to failure. In the military domain, it is critical that security is implemented correctly and completely. This gets us back to understanding and capturing security requirements. Studies have proven that the developing organization's lack of understanding of the requirements is the

leading cause of project failure [3].

Our CONOPS document presents the general concepts for the type of computer-intensive control system that would be needed to meet the Navy's stated requirements for system resilience under dynamically changing conditions. The security CONOPS is the first document in the system's development life cycle. It is conceivable that many parts or subparts would have similar CONOPS documents written to describe them. The full system can be decomposed into mission or functional entities, each of which would have a CONOPS written to describe it, e.g., command and control, radar/sonar, fire control, general information technology support, etc. This full collection of CONOPS documents would represent the user requirements for the entire system. With this full set of documents, refining requirements and creating the System Requirements Specification, or Software Requirements Specification, could begin.

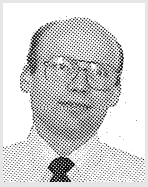
Conclusion

Complex software-intensive systems must have a thoroughly understood and soundly engineered set of requirements, which can be used along with best analysis practice to contribute to an effective design and ultimate implementation. The requirements are the foundation for the entire project, and must be understood precisely and managed diligently because changes are inevitable. It is advantageous to ensure that the real users of the systems are offered an opportunity to share their views and visions based on their working experience with the strengths, and weaknesses of their existing systems. Traditional software projects have been undertaken with less than complete understanding of requirements. The CONOPS document is a stride toward allowing the user's views to be heard, and allowing the developer to demonstrate to the users that their needs are understood and acknowledged.◆

References

1. Institute for Electrical and Electronics Engineers, *IEEE Guide for Information Technology-System Definition-Concept of Operations (CONOPS) Document. IEEE Std 1362-1998*, IEEE Computer Society Press, 1998.
2. Department of Defense, Topic N991-079 Multi-Level Security for Computer-Intensive Environments, *1999 DoD SBIR/STTR Program Solicitation*, U.S. Government Printing Office, 1999.
3. Forsberg, K. and H. Mooz. *System Engineering Overview. Software Requirements Engineering.* 1996. IEEE Computer Society Press.
4. Gomaa, H. The Impact of Prototyping on Software System Engineering. In *System and Software Requirements Engineering.* R. Thayer and M. Dofrman, eds. IEEE Computer Society Press. 1990.
5. Boehm, B. A Spiral Model of Software Development and Enhancement, in Thayer ed. *Tutorial: Software Engineering Project Management* IEEE Computer Society Press, 1988.
6. Davis, A. E. Bersoff, and E. Comer. A Strategy for Comparing Alternative Software Development Life Cycle Models." *IEEE Transactions on Software Engineering*. IEEE Computer Society Press. 1988.
7. Fairley, R. and R. Thayer, The Concept of Operations: The Bridge from Operational Requirements to Technical Specification, *Software Engineering*, M. Dorfman, and R. Thayer, eds. IEEE Computer Society Press. 1996.
8. Groguen, J. and C. Linde. Techniques for Requirements Elicitation. *Proceedings of the International Symposium on Requirements Engineering*. IEEE Press, 1993.
9. Ford, W. *Computer Communications Security Principles, Standards, Protocols, and Techniques*, PTR Prentice Hall, 1994
10. Pfleeger, C. *Security in Computing*, PTR Prentice Hall, 1996.

About the Author



Darwin E. Ammala is a senior-level software engineer with MPI Software Technology Inc., which has performed contract work for the National Science Foundation, Navy, Department of Energy, and NASA JPL. Previously, he was a senior computer scientist with 14 years of experience at Fort Mead, Md. He is pursing a master's degree in science from Mississippi State University, with an emphasis in security in high-performance and distributed-cluster computing.

MPI Software Technology Inc.
101 South Lafayette ST.
Starkville, Miss. 39759
Voice: 662-320-4300, ext. 11
Fax: 662-320-4301
E-mail: dammala@mpi-softtech.com