Annals of Software Engineering 3 (1997) 417–432

The concept of operations: The bridge from operational requirements to technical specifications

Richard E. Fairley

Colorado Technical University, 4435 N. Chestnut Street, Colorado Springs, CO 80907, USA

Richard H. Thayer

California State University, Sacramento, 6540 Chiquita Way, Carmichael, CA 95608, USA

This paper describes the role of a Concept of Operations (ConOps) document in specification and development of a software-intensive system. It also describes the process of developing a ConOps, its use and benefits, who should develop it, and when it should be developed. The ConOps described in this paper is compared to other forms of operational concept documents. A detailed outline for ConOps documents is provided in an appendix to the paper.

1. Introduction

The goal of software engineering is to develop and modify systems that satisfy user needs, on schedule and within budget. Accurate communication of operational requirements from those who need a software-intensive system to those who will build the system is thus the most important step in the system development process. Traditionally, communication of this information is accomplished in the following manner: the developer analyzes users' needs and buyer's requirements and prepares a requirements specification that defines the developers' understanding of those needs and requirements.[1] The users and buyer review the requirements specification and attempt to verify that the developer has correctly understood their needs and requirements. A draft users' manual is sometimes written by the developer to assist the users and buyer in determining whether the proposed system will operate in a manner consistent with their needs and expectations. A prototype of the user interface may be constructed to demonstrate the developers' understanding of the desired user interface.

This traditional way of specifying software requirements introduces several problems:

[1] Users are those who will interact with the new or modified system in the performance of their daily work activities; users include operators and maintainers. The buyer is a representative of the user community (or communities) who provides the interface between users and developer: the developer is the organization that will build (or modify) and deliver the system.

© J.C. Baltzer AG, Science Publishers

- First, the buyer may not adequately convey the needs of the user community to the developer, perhaps because the buyer does not understand the users' needs.
- Second, the developer may not be expert in the application domain, which inhibits communication.
- Third, the users and buyer often find it difficult to understand the requirements produced by the developer.
- Fourth, the developer's requirements specification typically specifies system attributes such as functions, performance factors, design constraints, system interfaces, and quality attributes but typically contains little or no information concerning operational characteristics of the specified system [IEEE 1984].

This leaves the users and buyer uncertain as to whether the requirements specification describes a system that will provide the needed operational capabilities.

A draft version of the users manual can provide some assurance that the developer understands user/buyer needs and expectations, but a draft version of the manual may not be written. If it is written, considerable time and effort have usually been spent by the time it is available for review. Major changes can require significant rework. Furthermore, it is difficult to demonstrate that the correspondences among technical specifications, users' manual, and (undocumented) operational requirements are complete and consistent.

A prototype of the user interface can be helpful, but there is a danger that demonstration of an acceptable user interface will be taken as assurance that the developer understands all of the users' operational needs. In summary, the traditional approach does not facilitate communication among users, buyer, and developer; nor does it emphasize the importance of specifying the operational requirements for the envisioned system.

Concept analysis helps users clarify their operational needs, thereby easing the problems of communication among users, buyer, and developer. Ideally, concept analysis and development of the ConOps document are the first steps in the development process; however, (as discussed below) developing a ConOps at later stages of the system lifecycle is also cost-effective.

A Concept of Operations document contains a description of the current system or situation, justification for and nature of proposed changes and/or new features, operational concepts for the new or modified system, operational scenarios for that system, a summary of organizational and operational impacts, and an analysis of the proposed system (see the Appendix).

Subsequent sections of this paper describe the evolution of the ConOps technique, the concept analysis process, the Concept of Operations document, roles to be played by a ConOps, some guidelines on when and how to develop a ConOps, development scenarios and a process for developing the ConOps, the recommended format for a ConOps, and some issues concerning the maintenance of a ConOps throughout the development process and the operational life of a software-intensive system.

2. History of the ConOps approach

One of the earliest reports on formalizing the description of operational concepts for a software system was written by Lano [1980]. The importance of a well-defined operational concept (e.g., definition of system goals, missions, functions, components) to the success of system development is emphasized in the report. The report presented tools, techniques, and procedures for more effectively accomplishing the system engineering tasks of concept formulation, requirements analysis and definition, architecture definition, and system design.

In 1985, the Joint Logistics Commanders' Joint Regulation "Management of Computer Resources in Defense Systems" was issued. This Joint Regulation included DoD-STD-2167, which contained a Data Item Description (DID) entitled "Operational Concept Document" (OCD) [US DoD 1985]. The purpose of that DID was to describe the mission of the system, its operational and support environments, and the functions and characteristics of the computer system within an overall system. The OCD DID was folded into the System/Segment Design Document [US DoD 1988a] in the revised version of DoD-STD-2167 [US DoD 1988b].

In DoD-STD-2167A operational concepts were moved into section 3 of the System/Segment Design Document (SSDD) [US DoD 1988a], which tended to place emphasis on overall system concepts rather than software concepts. Because the OCD was no longer a standalone document in 2167A, many users of 2167A did not place sufficient emphasis on operational concepts. For software-only projects, use of the SSDD was often waivered. In these cases, there was no other place within the 2167A DlDs to record operational concepts for a software-intensive system. As a result, several other government agencies, including NASA and the Federal Aviation Administration, produced their own versions of the original 2167 DID for documenting operational concepts within the 2167A framework.

Another DOD standard, DOD-STD-7935A, for development of information systems, required that the functional description of the proposed information system be contained in section 2 of that document [US DoD 1988c]. The functional description in 7935A provided little guidance on how to develop a ConOps document; furthermore, it was very specific to the information systems domain, emphasized functionality only, and allowed little flexibility for new methods and techniques of software system development.

In recognition of the importance of well-defined operational concepts to successful development of a software system, MIL-STD-498 [US DoD 1995] for software development and documentation, which has replaced 2167A and 7935A, includes a data item description for an Operational Concept Document (OCD). The authors of this paper played a leading role in developing the draft version of the Operational Concept Document (OCD) for the Harmonization Working Group that prepared MIL-STD-498. The OCD in MIL-STD-498 is similar to the ConOps outline contained in Appendix A of this paper. IEEE Standard 1498 [IEEE 1995], the commercial counterpart of MIL-STD-498 (which currently exists in draft form) incorporates an OCD similar to the one in Appendix A.

The American Institute of Aeronautics and Astronautics (AIAA) published a document titled "Operational Concept Document (OCD) Preparation Guidelines" [AIAA 1992]. The AIAA OCD compares favorably with the ConOps presented in this paper; however, in the opinion of this paper's authors, the tone and language used in the AIAA OCD is biased to the developer's view of user needs rather than the users' view of operational needs. The AIAA OCD is also biased toward embedded, real-time systems.

A major goal for the ConOps presented here is to provide a means for users of information processing systems, who are knowledgeable in their application domain but not expert in software engineering, to describe their needs and wants from their point of view; in other words, the recommended Guide is more user-oriented than existing standards and guidelines, which tend to be systems-oriented and developer-oriented.

Another difference between existing standards and the ConOps recommended in this paper is that this paper emphasizes the importance of describing both the current system's and the proposed system's characteristics, even though that may result in some redundancy in the document. The advantages of redundancy are considered to outweigh the problems.

3. The concept analysis process

Concept analysis is the process of analyzing a problem domain and an operational environment for the purpose of specifying the characteristics of a proposed system from the users' perspective. The traditional system development process emphasizes functionality with little concern for how that functionality will be used. Concept analysis emphasizes an integrated view of a system and its operational characteristics, rather than focusing on individual functions or pieces of a system. A major goal of concept analysis is to avoid development of a system in which each individual function meets its specifications, but the system as a whole fails to meet the users' needs.

Concept analysis should be the first step taken in the overall system development process. It identifies the various classes of users and modes of operation and provides users with a mechanism for stating their needs and desires. Concept analysis is also useful to surface different user's (and user groups) needs and viewpoints, and to allow the buyer (or multiple buyers) to state their requirements for the proposed system. This process is essential to the success of the subsequent system development effort. Users have an opportunity to express their needs and desires, but they are also required to state which of those needs are essential, which are desirable, and which are optional. In addition, they must prioritize the desired and optional needs. Prioritized user needs provide the basis for establishing an incremental development process and for making tradeoffs among operational needs, schedule, and budget.

Concept analysis helps to clarify and resolve vague and conflicting needs, wants, and opinions by reconciling divergent views. In the case where several user groups

(or buyer groups) have conflicting needs, viewpoints, or expectations, concept analysis can aid in building consensus. In some cases, it may be determined that no single system can satisfy all of the divergent needs and desires of multiple user groups and buyer agencies. It is better to make that determination earlier rather than later.

Concept analysis is an iterative process that should involve various people. The concept analysis team should include representatives from the user, buyer, and developer organizations, plus any other appropriate parties such as training and operational support groups. In cases where a development organization has not been selected at the time of concept analysis the developer role can be filled by in-house development experts or consultants.

The results of concept analysis are recorded in the ConOps document, which serves as a framework to guide the analysis process and provides the foundation document for all subsequent system development activities (analysis, design, implementation, and validation). The ConOps document should say everything about the system that the users and buyer need to communicate to those who will develop the system.

The ConOps document should be reviewed, revised, reviewed, revised, repeatedly, until all involved parties agree on the resulting document. This iterative process helps surface viewpoints, needs, wants, and scenarios that might otherwise be overlooked.

4. The Concept of Operations document

The Concept of Operations (ConOps) document describes the results of the conceptual analysis process. The ConOps document should contain all of the information needed to describe the users' needs, goals, expectations, operational environment, work processes, and other appropriate characteristics.

Essential elements of a ConOps include:

- A description of the current system or situation.
- A description of the needs that motivate development of a new system or modification of an existing system.
- Modes of operation for the proposed system.
- User classes and user characteristics.
- Operational features of the proposed system.
- Priorities among proposed operational features.
- Operational scenarios for each operational mode and class of user.
- Limitations of the proposed approach.
- Impact analysis for the proposed system.

A detailed outline for a ConOps document containing these elements is provided in the Appendix to this paper.

A ConOps document should, in contrast to a requirements specification, be written in narrative prose, using the language and terminology of the users' application domain. It should be organized so as to tell a story, and should make use of visual forms (diagrams, illustrations, graphs, etc.) whenever possible. Although desirable, it is not necessary that the needs and wants expressed in a ConOps be quantified; i.e., users can state their desire for "fast response" or "reliable operation". These desires are quantified during the process of mapping the ConOps to the requirements specification and during the flowdown of requirements to the system architecture. During system development the impact of tradeoffs among quantified system attributes (such as response time and reliability) must be explored within the limits of available time, money, and the state of technology.

A ConOps document should be tailored for the application domain, operational environment, and intended audience. This means that the terminology, level of abstraction, detail, technical content, and presentation format should adhere to the objectives for that particular ConOps document. The following points are worth making in this regard:

(a) A ConOps document must be written in the users' language. This does not necessarily imply that it cannot use technical language; but rather that it should be written in the users' technical language if the users are experts in a technical domain. If the ConOps document is written by the buyer or developer the authors must avoid use of terminology associated with their own discipline.

(b) The level of detail contained in a ConOps should be appropriate to the situation. For example, there may be instances in which a high level description of the current system or situation is sufficient. In other instances, a detailed description of the current system or situation may be necessary. For example, there may be no current system and a detailed statement of the situation that motivates development of a new system, with extensive specification of operational scenarios for the envisioned system may be required. Or, the new system may be a replacement for an existing system to upgrade technology while adding new capabilities; in this case, a brief description of the existing system would be appropriate, with more detail on the new capabilities to be provided by the new system. The level of detail also depends on whether the ConOps document is for a system as a whole, or whether there will be separate ConOps documents for each system segment (e.g., checkout, launch, on-orbit, and ground support elements for a spacecraft system) with an umbrella ConOps that describes operational aspects of the entire system.

(c) The presentation format used in a ConOps document will vary, depending on the application of the document. In some user communities, textual documents are the tradition, while in others, storyboards are used. Examples of this difference can be seen by comparing the styles of communication in the information processing and command-and-control domains. The presentation format should be adjusted to accommodate the intended audience of the ConOps, although the use of visual forms is recommended for all audiences.

(d) The comprehensive outline of a ConOps document, as presented in the Appendix, may not apply to every system or situation. If a particular paragraph of the outline does not apply to the situation under consideration, it should be marked "Not Applicable (N/A)"; however, for each paragraph marked N/A, a brief justification stating why that paragraph is not applicable should be provided in place of the paragraph. "Not Applicable" should be used only when the authors of a ConOps are confident that the paragraph does not apply to the situation, and not simply because the authors don't have the required information. For example, if the authors do not know whether alternatives and trade-offs were considered (paragraph 8.3 of the ConOps outline), they should determine that fact. In the interim period, the paragraph can be marked "TBD". If they determine no alternatives or trade-offs were considered the paragraph can be marked "not applicable". In this case, a brief justification stating why alternatives and trade-offs were not considered should be included.

To summarize, the ConOps format presented in Appendix A should be tailored to produce an efficient and cost-effective mechanism for documenting user needs and for maintaining traceability to those needs throughout the development process.

5. Roles for ConOps documents

The ConOps document can fill one of several roles, or some combination thereof:

(1) To communicate users' and buyer's needs/requirements to the system developers. The ConOps author might be a buyer, presenting users' views to a developer, or a user presenting the users' view to a buyer and/or a developer. In this case, the ConOps is used by the developer as the basis for subsequent development activities.

(2) To communicate a developer's understanding to users and/or buyer. The developer might produce a ConOps document as an aid in communicating the technical requirements to users and buyer, or to explain a possible solution strategy to the users and/or buyer. In this case, the ConOps is reviewed by the users and buyer to determine whether the proposed approach meets their needs and expectations.

(3) To communicate a buyer's understanding of user needs to a developer. In this case, the buyer would develop the ConOps and use it to present user needs and operational requirements to the developers. (Role 3 is similiar to Role 1; the difference being that in Role 1, buyer and users work together; in Role 3, the buyer conveys perceptions of user needs to the developer.)

(4) To document divergent needs and differing viewpoints of various user groups and/or buyers. In this case, each user group and/or buyer might develop (or

commission development of) a ConOps to document their particular needs and viewpoints. This would be done as a prelude to obtaining a consensus view (see Role 5), or to determine that no single system can satisfy all of the various users' needs and buyers' requirements.

(5) To document consensus on the system's characteristics among multiple users, user groups, or multiple buyers. In this case, the ConOps provides a mechanism for documenting the consensus view obtained from divergent needs, visions, and viewpoints among different users, user groups, and buyers before further development work proceeds.

(6) To provide a means of communication between system engineers and software developers. In this case, the ConOps would describe user needs and operational requirements for the overall system (hardware, software, and people) and provide a context for the role of software within the total system.

(7) To provide common understanding among multiple system/software developers. In cases where multiple system development and/or software development organizations are involved, the ConOps can provide a common understanding of how the software fits into the overall system, and how each software developer's part fits into the software portion of the system. In this case, there may be multiple ConOps documents, related in a hierarchical manner that mirrors the system partitioning.

Variations on, and combinations of these roles might be found under differing circumstances. For example, the ConOps process might play Roles 4 and 5 to obtain and document consensus among user groups and buyers prior to selection of a developer; the consensus ConOps document would then fill Role 1 by providing the basis for subsequent development activities by the developer.

Additional roles for the ConOps include:

(8) Providing a mechanism to document a system's characteristics and the users' operational needs in a manner that can be verified by the users without requiring them to have any technical knowledge beyond what is required to perform their job functions.

(9) Providing a place for users to state their desires, visions, and expectations without requiring them to provide quantified, testable specifications. For example, the users could express their need for a "highly reliable" system, and their reasons for that need, without having to produce a testable reliability requirement.

(10) Providing a mechanism for users and buyer(s) to express their thoughts and concerns on possible solution strategies. In some cases, there may be design constraints that dictate particular approaches. In other cases, there may be a variety of acceptable solution strategies. The ConOps allows users and buyer(s) to record design constraints, the rationale for those constraints, and to indicate the range of acceptable solution strategies.

6. When should the ConOps be developed?

Development of a ConOps document should be the first step in the overall development process, so that it can serve as a basis for subsequent development activities.

The ConOps might be developed:

(1) Before the decision is made to develop a system. In this case, the ConOps document would be used to support the decision process.

(2) Before the request for proposals (RFP) or in-house project authorization is issued. The ConOps would be included in the RFP package or project authorization.

(3) As the first task after award of contract or in-house authorization, so that the developer can better understand the users' needs and expectations before subsequent system development activities are started.

In cases (1) and (2), development of the ConOps document will be initiated by the users or the buyer (although the author of the document might be a developer; possibly the developer who will later develop the system). In case (3), development of the ConOps can be initiated by, and/or developed by, the user, buyer, or developer.

Concept analysis and preparation of a ConOps document can also be quite useful even if initiated at a later stage of the system lifecycle. If, during system development so many diverging opinions, needs, visions, and viewpoints surface that the development process cannot continue successfully, a ConOps document can provide a common vision of the system. The ConOps document for the Hubble Space Telescope System is a good example of this situation [NASA 1983]. It was written after several attempts to develop a requirements specification; however, potential users of the space telescope could not agree on the operational requirements. The ConOps document provided the vehicle for obtaining a consensus view, which in turn provided a basis for generating detailed operational requirements.

The developer who is building a system might want to develop a ConOps document, even as the technical specifications are being generated. The developer might want the ConOps to serve as a high level overview and introduction to the system for the development team. Developers might develop a ConOps document to increase their confidence that they will develop a system that meets the users' needs and expectations.

A ConOps document might be developed during the operational phase of the system lifecycle to support users, operators, and maintainers of the system. It might happen that potential users of the system do not want to use it because they do not understand the operational capabilities of the system, or because they do not understand how the system would fit into their working environment. To solve these problems, the buyer or the developer might develop a ConOps document to "sell" the system to potential users. A ConOps is also helpful to new users, operators, and maintainers

who need to understand the operational characteristics of a system. The ConOps can also be used to explain the operational characteristics of a system to prospective buyers who were not involved in initial development of the system.

If the involved parties deem it to be useful, a ConOps document can be developed at any time during the system lifecycle; however, some major benefits of the document and the process of developing it are lost if it is developed after the requirements specification is baselined.

7. Scenarios for developing the ConOps

Ideally, concept analysis and development of the ConOps document should be done by the users. However, depending on the purpose and timing of development, the ConOps might be developed by the users, the buyer, or the developer. Regardless of who develops the ConOps, it must reflect the views of, and be approved by, the user community.

A high degree of user involvement in concept analysis and review of the ConOps document is crucial to a successful outcome, even if concept analysis and development of the ConOps document are done by the buyer or the developer. In these cases, the buyer or developer must engage the users in the process to ensure a correct and comprehensive understanding of the current system or situation and the users' needs, visions, and expectations for the new system. One way to ensure the necessary interactions is to establish an interdisciplinary team consisting of representatives from all user groups, from the buyer(s), and from the developer(s). However, the focus must never be allowed to shift from the users' operational perspective to the buyer's or developer's perspective.

One benefit of having the users write the ConOps document is that it ensures the focus will stay on user-related issues. However, the users may not know how to develop a ConOps document or be able to realistically envision what a new system can accomplish, i.e., they may not know the capabilities of existing technology. To reduce the impact of these problems, qualified personnel can be brought in to assist the users in developing the ConOps document.

One benefit of having the developers write the ConOps document is that they will, in most cases, have comprehensive knowledge of available technologies, and thus may be able to propose alternative (and better) ways of solving the users problems. Another benefit of a developer-produced ConOps is that the ConOps analysis process will provide the developer with a good understanding of the users' problems, needs, and expectations, which facilitates subsequent development activities.

An advantage of a buyer-developed ConOps is that the buyer may have a good understanding of the user community, the developer organization, the political realities of the situation, and the budgetary constraints that may exist. This knowledge can be invaluable in producing a ConOps for a system that will satisfy user needs and that can be delivered within political and budgetary constraints.

Regardless of who takes primary responsibility for producing the ConOps document, it is important that all parties (users, buyers, developers) be involved in the analysis process and that everyone contribute their particular viewpoint to development of the ConOps.

8. A development process for the ConOps

The approach described below is intended as a guideline. If the approach conflicts with what seems to be most appropriate in a specific situation, the guideline should be modified to fit that situation. For instance, there may be no current system; or the new system may be a modification of a current system; or the new system may be a total replacement for an outdated (manual or automated) system. Topics emphasized in the ConOps may be different in each situation.

(1) Determine the objectives, roles, and team members for the ConOps process. This will normally be determined by the situation that motivates development of the ConOps document.

(2) Tailor the recommended ConOps document format and obtain agreement on an outline for the ConOps document. This is important so that everyone understands the agreed-upon format and content areas of the document.

(3) Describe the overall objectives and shortcomings of the current system. Also determine and document the overall objectives for the new or modified system. If there is no current system, describe the situation that motivates development of a new system.

(4) If there is an existing system, describe the scope and boundaries of that system, and identify any external systems and the interfaces to them. Also establish and describe in general terms the scope and boundaries for the new or modified system, and identify the major external systems and interfaces to it.

(5) Describe operational policies and constraints that apply to the current system or situation and any changes to those policies and constraints for the new system.

(6) Describe the features of the current system or situation. This includes the system's operational characteristics, operational environment and processes, modes of operation, user classes, and the operational support and maintenance environments.

(7) State the operational policies and constraints that will apply to the new or modified system.

(8) Determine the operational characteristics of the proposed system, i.e., describe the characteristics the proposed system must possess to meet users' needs and expectations.

(9) Document operational scenarios for the new or modified system. Scenarios are specified by recording, in a step-by-step manner, the sequences of actions and interactions between a user and the system. The following approach can be used to develop and document operational scenarios:

   (a) Develop a set of scenarios that, to the extent possible, covers all modes of operation, all classes of users, and all specific operations and processes of the proposed system.

   (b) Walk through each scenario with the appropriate users and record information concerning normal operating states and unusual conditions that are relevant to the operation of the proposed system.

   (c) During the walkthroughs, establish new scenarios to cover abnormal operations such as exception handling, stress load handling, and handling of incomplete and incorrect data.

   (d) Establish new scenarios whenever a branch in the thread of operation is encountered. Typically, walking through the "normal" scenarios will uncover additional scenarios. Different users may also have different views of some scenarios. If these variations are significant, include them as separate scenarios.

   (e) Repeatedly develop scenarios until all operations, and all significant variations of those operations, are covered.

   (f) For each operational scenario, develop an associated test scenario to be used in validating the operational aspects of the delivered system in the user environment. Establish traceability between operational scenarios and test scenarios.

(10) After the scenarios have been developed, validate the description of the proposed system and the operational scenarios by walking through all of the scenarios with representatives from all user groups for all modes of operation.

(11) Obtain consensus on priorities among the operational scenarios and features of the proposed system. Group the scenarios and operational features into essential, desirable, and optional categories; prioritize scenarios and features within the desirable and optional categories. Also describe scenarios and operational features considered but not included in the proposed system.

(12) Analyze and describe the operational and organizational impacts the proposed system will have on users, buyer(s), developers, and the support/maintenance agencies. Also include significant impacts on these groups during development of the system.

(13) Describe the benefits, limitations, advantages, and disadvantages of the proposed system, compared to the present system or situation.

9. Recommended format of a ConOps document

The recommended format of a ConOps document accommodates the objective of describing a proposed system from the users' point of view, in user terminology. The following format is recommended. Appendix A contains a detailed version of this outline.

1. Introduction to the ConOps document and to the system described in the document.
2. List of all documents referenced in the ConOps document.
3. Description of the current system or situation, including scope and objectives of the current system, operational policies and constraints, modes of operation, classes of users, and the support environment for the current system. If there is no existing system, describe the reasons that motivate development of a new system.
4. Nature of proposed changes and/or new features, including the justification for those changes and/or features.
5. Operational concepts for the proposed system, including scope and objectives for the proposed system, operational policies and constraints, modes of operation, classes of users, and the support environment for the proposed system.
6. Operational scenarios describing how the proposed system is to perform in its environment, relating system capabilities and functions to modes of operation, classes of users, and interactions with external systems. Guidelines for developing operational scenarios have been developed in the fields of systems engineering, object-oriented analysis, and design of human-computer interfaces.
7. Operational and organizational impacts on the users, buyers, developers, and the support and maintenance agencies, during development of the system and after installation of the system.
8. Alternative and trade-offs considered but not included in the new or modified system; analysis of benefits, limitations, advantages, and disadvantages of the new or modified system.
9. Notes, acronyms and abbreviations, appendices, and glossary of terms.

This organization of a ConOps document provides a logical flow of information beginning with a description of the current system, transitioning through considerations of needed changes and the rationale for such changes, and leading to a description of the new or modified system. This will guide the reader through the description of the systems (both the current system or situation and the proposed system) in a simple and intuitive way.

10. Maintaining the ConOps

A ConOps should be a living document that is updated and maintained throughout the entire lifecycle (development process and operational life) of the software product. During system development, the ConOps document must be updated to keep users informed of the operational impacts of changes in requirements, the system design, operational policies, the operational environment, and other users' needs. During the operational life of the software product, the ConOps must be updated to reflect the evolutionary changes to the system.

It is important to maintain the ConOps document under configuration control, and to ensure that user and buyer representatives are members of the change control board for the ConOps. Placing the ConOps under configuration control will protect the document from uncontrolled changes, and through the formal process of updating and notification, help to keep all parties informed of changes. A major benefit of this approach is that users and buyers are involved in reviewing and approving the changes. This minimizes the surprise factor that can occur when a delivered system is not the same as the system users thought they agreed to at the requirements review.

The ConOps document should also be updated and maintained under configuration control throughout the operational life of the associated system. During the operational life of the system, a ConOps can aid the support, maintenance, and enhancement activities for the system in much the same way that it helped during development. Specifically, it can be used to communicate new operational needs and impacts that result in modifications, upgrades, and enhancements. Furthermore, the ConOps provides a communication tool to familiarize new personnel with the system and the application domain.

Traceability should be established and maintained among the ConOps document, the system/software requirements specifications, and the acceptance/regression test scenarios. It is important for the developer (or maintainer) to be able to demonstrate to the users, buyer, and themselves that every essential user need stated in the ConOps document, and the desirable and optional features implemented, can be traced to and from the system specifications and to and from the delivered capabilities in the final product.

11. Summary and conclusions

This paper has described the evolution of the ConOps approach, the conceptual analysis process, the Concept of Operations document, roles to be played by a ConOps, some guidelines on when to develop a ConOps, development scenarios and a development process for developing the ConOps, the recommended format for a ConOps, and some issues concerning the maintenance of a ConOps throughout the development process and operational life of a software system.

As software engineers, we become so involved in the technology of software development and modification that we sometimes forget our fundamental charter: to

develop and modify software-intensive systems that satisfy user needs, on time and within budget. Performing conceptual analysis and developing and maintaining a Concept of Operations document provides the bridge from users' operational requirements to technical specifications. All subsequent work products (requirements specs, design documents, source code, test plans, user manuals, training aids, maintenance guide, etc.) should flow from the ConOps. Maintaining the ConOps and the traceability of work products to the ConOps will not guarantee success; however it can increase the probability that we will develop systems that satisfy users needs for efficient and effective software-intensive systems that will help them to better accomplish their work activities.

Acknowledgements

The authors would like to acknowledge the support of the following individuals in preparing the ConOps Guide: Mr. Per Bjorke, Dr. Merlin Dorfman, Dr. Lisa Friendly, and Ms. Jane Radatz.

Appendix. Outline for a Concept of Operations document

1. Scope
   - 1.1. Identification
   - 1.2. System overview
   - 1.3. Document overview
2. Referenced documents
3. The current system or situation
   - 3.1. Background, objectives, and scope of the current system or situation
   - 3.2. Operational policies and constraints for the current system or situation
   - 3.3. Description of the current system or situation
   - 3.4. Modes of operation for the current system
   - 3.5. User classes for the current system
     - 3.5.1. Organizational structure
     - 3.5.2. Profiles of user classes
     - 3.5.3. Interactions among user classes
   - 3.6. Other involved personnel
   - 3.7. Support environment for the current system
4. Justification for and nature of proposed changes/new features
   - 4.1. Justification for changes and new features
   - 4.2. Description of needed changes and new features
   - 4.3. Priorities among changes and new features
   - 4.4. Changes and new features considered but not included
   - 4.5. Assumptions and constraints
5. Concepts of operations for the proposed system
   - 5.1. Background, objectives, and scope for the new or modified system

5.2. Operational policies and constraints
5.3. Description of the proposed system
5.4. Modes of operation for the proposed system
5.5. User classes for the proposed system
5.1.1. Organizational structure
5.1.2. Profiles of user classes
5.1.3. Interactions among user classes
5.1.4. Other involved personnel
5.6. Support environment for the proposed system
6. Operational scenarios for the proposed system
7. Summary of impacts
7.1. Operational impacts
7.2. Organizational impacts
7.3. Impacts during development
8. Analysis of the proposed system
8.1. Summary of improvements
8.2. Disadvantages and limitations
8.3. Alternatives and trade-offs considered
9. Notes
Appendices
Glossary

References

AIAA (1992), *AIAA Recommended Technical Practice, Operational Concept Document* (*OCD*), *Preparation Guidelines*, Software Systems Technical Committee, American Institute of Aeronautics and Astronautics (AIAA), March 1, 1992.

IEEE (1984), *ANSI/IEEE Standard 830-1984: IEEE Guide for Software Requirements Specifications*, The Institute of Electrical and Electronic Engineers, Inc., approved by the American National Standards Institute, July 20, 1984.

IEEE (1995), *IEEE Standard for Acquirer-Supplier Agreement for Software Life Cycle Process*, Version D2, The Institute of Electrical and Electronic Engineerings, Inc., March 15, 1995.

Lano, R.J. (1980), "A Structured Approach For Operational Concept Formulation (OCF)," *TRW-SS-80-02, TRW Systems Engineering and Integration Division, Redondo Beach, CA, January 1980*; also In *Tutorial: Software Engineering Project Management*, R. Thayer, Ed., Computer Society Press, 1988.

NASA (1983), *Science Operations Concept, Part 1* (*Final*), Space Telescope Science Institute, Prepared for NASA Goddard Space Flight Center, Greenbelt, MD, May 1983.

US DoD (1985), *Operational Concept Document* (OCD), DI-ECRS-8×25, DoD-STD-2167, US Department of Defense, June 4, 1985.

US DoD (1988a), *System/Segment Design Document* (SSDD), DI-CMAN-80534, DoD-STD-2167A, US Department of Defense, February 29, 1988.

US DoD (1988b), *Military Standard: Defense System Software Development*, DoD-STD-2167A, US Department of Defense, February 29, 1988.

US DoD (1988c), "Functional Description (FD)," *DoD Automated Information Systems* (*AIS*) *Documentation Standards*, DoD-STD-7935A, US Department of Defense, October 31, 1988.

US DoD (1995), *Software Development and Documentation Standard*, MIL-STD-498, US Department of Defense, December 5, 1995.