A concept of operations Storyboard Tool was developed to assist authors in building a concept of operations for a new system, refining it with stakeholders, and using it to support subsequent development activities. It was developed iteratively, testing iterations by using the tool to support ongoing research and development projects at NASA Johnson Space Center. In this paper, we present lessons learned about integrating sketches and descriptions for clearer communication, the benefits of organizing descriptive information as structured data, and assisting the process of concept development. We also discuss the value of supporting workflow process and the role of human factors in systems engineering.

INTRODUCTION

It is difficult to build software systems that provide effective support to users in performing their tasks. Much of the difficulty is translating a good idea about user task support into smoothly running software applications that provide useful and usable support. In the course of expressing the concept for a new system, stakeholders often become distracted from the original concept: support for human task performance (Thronesbery & Malin, 1998).

To address these difficulties, system engineers have developed a Concept of Operations (ConOps) document that serves as a clear statement of how the new system is intended to help the end user perform tasks. Standard document formats (IEEE Std 1362-1998, 1998; ANSI/AIAA G-043-1992, 1993; DI-IPSC-81430, 1994) remind authors of critical information that should be included. Completed ConOps documents can be consulted while developing requirements and designing the systems so that developers are less likely to lose sight of critical end user task performance. ConOps documents do help and are becoming increasingly accepted as a part of the system acquisition process.

However, static ConOps documents are not fulfilling all the desired functions. First, they are not as widely used as expected, possibly because standard formats daunting. In the course of being thorough, the designers of these documents have included paragraph headings for an extensive amount of information. This alone could explain a reluctance to write a ConOps document. Yet, creating a simplified format could leave many projects with an incomplete system concept. Second, while the document formats emphasize the importance of illustrative scenarios, they do not provide much assistance for building them. Third, while these standard formats encourage illustrative sketches, they do not assist the author to distinguish critical parts of the sketch that represent key concepts from those that are incidental. Hence, early concept discussions can become sidelined by implementation details when they should focus on useful task support.

The ConOps Storyboard Tool was developed to assist ConOps authors in building a concept of operations for a new system, refining it with stakeholders, and using it to support subsequent development activities.

We describe the development of a tool to assist with the authoring, refinement, and usage of a concept of operations. We also describe the lessons learned from this iterative development effort, lessons that might be useful to human factors engineers participating in system acquisition activities. Because it was an iterative development effort, many of the lessons were learned because of false starts in early iterations and successes in later ones. Most of these false starts are reported as positive lessons for simplicity of presentation. Our means of testing iterations was to use the tool to support small, ongoing, software research and development projects at NASA Johnson Space Center.

INNOVATION

The ConOps Storyboard Tool was developed to help concept of operation authors describe information about the intended use of a new system, as well as build illustrative scenarios. In addition, the tool is designed to assist authors in presenting this information to stakeholders for comments and refining the ConOps in response to those comments. After information has been entered, a ConOps document can be printed from the tool and design information can be extracted that supports subsequent efforts like requirements development and system design.

The fundamental concept of the storyboard tool is the integration of sketches with text describing the scenario and clarifying the important parts of the sketches. Figure 1 shows the integrated view of sketch and description that is presented for stakeholder review. The authoring view is shown in Figure 2, where information is entered under category headings. Data entered by the user in the forms shown in Figure 2 are linked to an image file of the design sketch. Figure 1 illustrates how this linkage is used to construct a display for stakeholders.

Descriptive text is entered into the tool as structured data enabling machine reasoning over this information to provide improved user tools. This practice follows an example from a number of system engineering tools, e.g., DOORS, Cradle,
Rational Rose, SpecTRM (Leveson, 1997), that are moving from document-centered products to data-centered products.

Figure 1. Storyboard Tool integrates text and sketches.

The data structure used to represent the concept of operations can be considered a ConOps data model. It defines all the properties of information that can be used to describe a ConOps. It includes a user-configurable setting for each property that permits identifying a subset of these properties as relevant to a particular project or intended use. The tool uses this setting to determine which properties are shown to the user. It also defines a rich set of relationships among these information properties. Properties are organized into hierarchies that aid reasoning about a related set of information. For example, one can inspect and manipulate the properties of a scenario as a set. This hierarchy also is reflected in a tree-structured user interface, permitting sets of information to be revealed and hidden as the user's focus of attention changes. Some properties have ordinal relationships among them. Specifically, the steps of a scenario are represented as an ordered sequence of actions. This ordinal information affects the choice of user interface for steps. Some properties, such as the user group, are linked to multiple points in the data model. Thus, a group of users can be defined for the overall system and members of that group can be reused when specifying the actions performed in a scenario for that system. Finally, some properties define links to external media such as image files. The software utilizes this linkage information to select software for viewing the contents of the property. Thus, the tool can automatically determine the most appropriate way to view the property value (e.g., in a text box or in an image viewer).

Figure 2. Storyboard Tool reminds authors of useful information.

There are a number of benefits from this structured data approach. Initially, the structure reminds authors of the information that will help them to describe the intended use of the system. Information categories have been fashioned after the paragraph headings recommended in standard ConOps documents. However, while those document formats emphasize the importance of illustrative scenarios, they do not give much guidance about what information to include. The Storyboard Tool goes beyond the standard document formats by reminding the author of useful information to describe

Figure 3. Detailed, situated help is easier to provide with structured data.
scenarios, steps, and sketches. A second benefit of the structured information is the ability to provide the author with situated help for each information category, as illustrated in Figure 3. A third benefit is something we call vertical linking. Some information at multiple levels of description is related. For instance, a full list of users should be kept at the system level of description, but any given scenario or step within a scenario may involve only a subset of those users. The tool assists the author in maintaining a master list of users and in naming the users consistently throughout the ConOps. A fourth benefit of the structured information is that it can be tailored to the unique demands of a new domain by editing a configuration file. Thus, if a domain has unique safety considerations, they can be added to the structured data by editing the configuration data file for that domain. Another benefit is flexibility in using ConOps information for subsequent development activities. We have designs for links to pre-requirements, requirements, UML diagram components, and other design concepts, as well as to modeling information and multimedia illustrations.

Perhaps the greatest benefit for ConOps authoring support is the ability to associate selected ConOps products with specific information categories. For instance, if the author is interested in testing out the simplest of storyboard scenarios, or “story sketches”, on a few colleagues to see if the system concept merits additional work, only a small fraction of the structured data is initially relevant. Figure 4 illustrates the selection of the “story sketches” product in the left hand panel and the resulting identification of relevant information categories in the center panel. The panel on the far right indicates to the author how many of the selected categories have been completed so the author can track work progress. While entering information under categories, the author can turn on the ViewFilter so that only the selected information categories are shown.

Finally, a printable ConOps Document can be extracted from the Storyboard Tool. The document can be used to satisfy documentation requirements or it can be used to solicit comments from stakeholders or to guide subsequent development activities. Alternatively, the document can be put into a web page format for paperless publication. Because the basic format of the storyboard data is formatted using the eXtensible Markup Language (XML) format, an XML Stylesheet (XSL) transformation has been created that transforms the storyboard information into a data format that can be displayed in a web page that resembling the Storyboard Tool user interface. An example of this web page format is illustrated in Figure 5.

LESSONS LEARNED

We employed an iterative development approach by first implementing the core user support for the ConOps author. We tested the resulting prototype by supporting ongoing software projects at NASA Johnson Space Center. Based on the results of using the prototype, we modified it and incorporated additional user support. This iterative development and testing was employed from the beginning of the project in January of 2006 until the end in March of 2009. In this section, we discuss lessons learned for the Storyboard Tool during this testing.

One of our original hypotheses was that integrating text with sketches would help stakeholders understand the intended system usage. The importance of aiding communications among people with varying backgrounds is emphasized by the practice of concurrent engineering (Cutchofsky, et al., 1993) and observations of impediments to good design by Holden, et al. (1998). The work of Rettig (1994) and Snyder (2003) indicates that a dynamic, graphical communication medium might be successful for this task. This hypothesis received some confirmation as stakeholders in our test groups seemed to become oriented more quickly. It was further confirmed as discussions with stakeholders became focused quickly on the
desired system usage rather than being sidetracked by incidental sketch artifacts. For instance, discussions centered on which tasks should be supported and in what order as opposed to whether a radio button or checkbox was more appropriate. Unlike requirements and design documents, this storyboard medium seemed to communicate equally well with all stakeholders regardless of background. End user representatives were able to make valuable refinements about what the new system should do and developers were able to suggest variations that could be more easily developed. While these discussions represent difficult system concept development work, the teams seemed to reach critical issues more quickly so that resolutions could be found.

Another early hypothesis was that treating the descriptive information as structured data would help with exporting selected contents of the ConOps for subsequent development activities like requirements and design. This practice follows the lead of popular system development software like Cradle, DOORS, Rational Rose, and SpecTRM (Leveson, et al., 1997), which encourage a data focus to system development rather than a more traditional document focus. We have found abundant support for this hypothesis, and continue to find new ways of exporting ConOps for use in subsequent activities. For example, we expect a web page display of ConOps information like the one illustrated in Figure 5 to be a useful way of distributing the concept of operation to a geographically dispersed set of stakeholders.

Our third major lesson was not part of our original set of hypotheses: workflow assistance is needed to show simple tool uses while enabling wide flexibility of use. About midway through our project, we encountered increasing objections from tool users and prospective tool users that the large number of information categories was daunting. Some users objected that it is just too much information to provide, while others bravely tried to enter information for every information category even when the category did not match the needs of the project. While it was never intended that every project should have information for every information category, it was clear that we had not communicated this idea to our prospective users and that the display of these categories in the tool user interface complicated the entry and review of ConOps information. Initially, we planned to describe the intended relationship between project goals and individual information categories in the situated help files, tutorials, and user manuals. Because this placed a burden on the tool user, we made explicit links between the intended ConOps products and the associated information categories, and we integrated those links into the tool. We added a display for inspecting these relationships that provides workflow assistance to the user. Using this feature, users can see what information categories best support their intended use of the tool. In retrospect, it is possible to consider these project goals as use cases for the Storyboard Tool. As such, the concept of workflow assistance can be viewed as arising from a high-level analysis of the storyboard author’s tasks.

**SUMMARY AND DISCUSSION**

To summarize, we have developed a proof-of-principle prototype of a tool to support authoring, refining, reporting, and using concept of operations information that describes how a new system will support its end users. The prototype is usable and useful, but needs additional refinement to have the smooth quality of a commercial product. Early uses of the tool confirm the value of integrating text and graphics to illustrate how a new system will support its users. Those uses also underscore the value of entering descriptive information under specialized data categories to support flexible uses of the information. Finally, a product-oriented workflow assistance view of the tool enabled users to see simple uses of the tool tailored to their project goals.

In addition to the findings based on direct experience with supporting ongoing projects at NASA Johnson Space Center, we have additional observations about this tool and about software development in general.

First, the storyboard tool has a wide range of potential uses. We tested it in the context of small, innovative software projects at Johnson Space Center. We believe it can be easily extended to support much larger projects. In fact, larger projects probably benefit more from developing ConOps information than smaller ones because of the difficulty of gaining consensus in larger groups. Also, we tested with space-related projects, but it is applicable to a wide variety of domains. ConOps documents were originally developed to support system acquisition for defense projects, so it is rather certain that this tool based in ConOps document organization could support military applications. Also, we tested it with software projects, but believe that with minor refinements, it could support other engineering projects. In fact, we have specified information category names in a configuration file so that alternate application domains can add or change categories specific to their needs without software changes to the tool. The use of XML to represent ConOps information permits defining XML stylesheets to format the data to meet documentation requirements of specific domains. Thus, we think the storyboard tool can be applied to a wide range of application domains.

Also, over the course of developing and testing the storyboard tool, we received suggestions that the storyboard tool can be used for more than representing concept of operations information. An obvious suggestion is that it is good for generating use case scenarios. To build use case scenarios, an author would simply ignore the ConOps document heading information categories and concentrate only on the scenario definitions and illustrations. The easy way of doing this is to select the “Story Sketches” product in the workflow assistance and turn on the ViewFilter settings so that no other information categories appear as a distraction. Other suggested uses require a bit more creativity. Some see it as supporting project management, describing the project with both abstract descriptions and concrete illustrations, and
tracking status of the project by observing the completion status in the workflow and by tracking the illustrations from sketches to screen snapshots. While we can see project management information in the tool, we think managers will need many other tools in addition to this one. Another use we have made of the tool is in formulating relatively inexpensive proposals in terms understood by all so that stakeholders can make clear agreements about potential work.

Another observation is that workflow assistance is often needed for software tools that have great flexibility and complexity. The trigger for recognizing the need for workflow assistance is when adding flexibility makes the complexity of a tool daunting to its users. The workflow assistance that is needed can be illuminated by identifying simple usage scenarios and making those simpler uses of the software tool more apparent. It is certainly more helpful to the user to implement workflow assistance in the software rather than simply documenting special uses in a tutorial.

A final observation is that concept development aids like the Storyboard Tool help to establish the role of the human factors engineer early in system specification. Such tools assist in building a concept of operations and establishing a consensus about it among the stakeholders. These activities are an excellent opportunity for human factors engineers to establish their roles with the system development team (Thronesbery, et al., 2006). Human factors engineers have unique skills for working with prospective users and providing a task-oriented view of a system. These early system concept definition activities can provide human factors engineers with the opportunity to build good working relationships with the other stakeholders and to ensure that the ConOps has a strong user task orientation. This orientation is important to the upper levels of Forsberg and Mooz’s “Vee” model of system engineering (1995) to support the building of the system concept and the plans for validation testing and later for the execution of the validation and acceptance testing.

ACKNOWLEDGEMENTS

This work was performed under a U.S. Government contract (NNJ07JA21C) in the NASA Small Business Innovative Research program. We wish to thank Jane Malin, of NASA/Johnson Space Center, who served as the Contracting Officer’s Technical Representative and provided valuable guidance, direction, and consultation. We would also like to thank Kevin Kusy, who took some sketches and storyboards and converted them into functioning Java code to construct the proof-of-principle prototype.

REFERENCES


