

Collaborative Design with Use Case Scenarios

Lynne Davis
DLESE Program Center
University Corporation for Atmospheric Research
P.O. Box 3000
Boulder, CO. 80307-3000
303 497-8313
lynne@ucar.edu

Melissa Dawe
Center for LifeLong Learning and Design
Dept. Of Computer Science
University of Colorado at Boulder
303 492-4932
meliss@colorado.edu

1. ABSTRACT

Digital libraries, particularly those governed by a community, need to be collaboratively designed. In this paper, we compare our experience using two design methods: a Task-centered method that draws upon a group's strength for eliciting and formulating tasks, and a Use Case method that tends to require a focus on defining an explicit process for tasks. We discuss how these methods did and did not work well in a collaborative setting.

Categories and Subject Descriptors

H.3.7 [Information Storage and Retrieval]: Digital Libraries – *user issues*.

General Terms

Design, Experimentation, Human Factors.

Keywords

Use case, task-centered design, collaboration, design model, methodology.

2. INTRODUCTION

Designing a digital library system is difficult; doing so collaboratively through community participation is even more so. Nevertheless, for the last twelve months we have actively engaged the community in the design of the Digital Library for Earth System Education (DLESE). DLESE serves a broad base of educators and students interested in any scientific aspect of the Earth and as such requires a systematic design approach to define requirements. Because DLESE is envisioned, designed and sustained by its community of users, such an approach for defining requirements needs likewise to be a highly collaborative effort. As we approached this problem, two design models emerged as likely candidates to provide this system: Task-centered design [3] and Use Case development [1]. Because both models use natural language to define requirements, users can

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Conference '00, Month 1-2, 2000, City, State.

Copyright 2000 ACM 1-58113-000-0/00/0000...\$5.00.

read and understand them more easily than traditional software requirements documents. Also, both assume an iterative development model for rapid prototyping and flexibility.

But, while both profess to consider the needs of the users to be paramount in a successful design, they differ considerably in their methods. In July of 2000, at the DLESE Leadership Workshop at Montana State University in Bozeman, MT, our informed efforts to use the Use Case development method failed to result in the expected outcomes. We explain those methods and why we conclude that the Use Case methods were not appropriate for use in a highly collaborative design setting.

3. METHODOLOGIES

3.1 Task-centered Design

Task-centered design methodology [3], well known among the human-computer interaction (HCI) community and interface designers, provides techniques to elicit tasks from users that capture the user's goals and work setting. It centers on developing a small suite of tasks that represent, in essence, who will use the system and what they want to do. We used it before the workshop to elicit user tasks. Here is an excerpt from one:

Alan teaches an introductory undergraduate geology class. He wants to find a few photographs taken by the Sojourner of the surface of Mars and compare them to similar geographical areas on earth for his class.

Tasks are then used to guide design and testing of the system. The process of creating a user interface based on these tasks occurs through an iterative process of design and user testing. The precise steps a user would take to accomplish a task are determined while a tangible user interface develops. This iterative design method encourages consideration of alternatives.

3.2 Use Case Design

The Use Case design methodology as defined by Alistair Cockburn [1] is popular among software engineers to define requirements in terms of a specific sequence of steps. It begins as a narrative that is much like the tasks in the Task-centered design methodology. A major difference is that the narratives are then broken down into a sequence of execution steps, called a scenario. For each narrative, many scenarios describe the different outcomes a user might experience when executing a task. Thus, Use Cases try to provide more detail about the system. However, the model does not include contextual or

environmental factors that can influence use, and can infer commitment to a design prematurely [2], before design alternatives can be considered.

In preparing for the DLESE Leadership Workshop, the Task-centered design methodology appealed to the user interface designers, while the Use Case methodology appealed to software engineers, who wanted the system requirements to be in a form more readily usable by them. So, our design process involved elements of both.

4. SUITABILITY FOR COLLABORATION

Where these approaches really differ becomes apparent in our collaborative design setting. The Use Case approach assumes that the narrative for a goal is already written and is ready to be broken down into its various scenarios. At the Leadership Workshop, we started with representative tasks that were expressed by the user community previously. The participants were asked to form groups of 4-6 and over a one-hour period, begin to articulate step-by-step scenarios to accomplish these tasks. The process of creating Use Cases that conform to Cockburn's model requires skill to break down a Use Case into a formatted sequence of steps and assumes a readiness on the part of the collaborators to commit to those steps.

This approach was problematic in our collaborative setting. Few participants understood the process. To list a single sequence of steps the group had to exclude other interrelated steps, at least temporarily. The group had to accept assumptions about pre- and post-conditions to carry out the goal. Also, by starting with a given narrative, it disregarded a key need for the user community to contribute to the understanding of their tasks and work setting, and hence, for the designers to situate the system most appropriately. Finally, the collaborators were not ready to commit to a defined sequence of steps which made group feel uneasy and hampered successful collaboration.

The Task-centered design process, on the other hand, begins by trying to understand the user and the user's situation. In the very early stages of the DLESE design, we asked our community to give us stories about how they envision the library might support them in their work. What we received were highly visionary, detailed descriptions. This approach facilitated group participation that was useful to the designers for describing user settings and system context.

The user interface developers transformed task descriptions into tangible mockup designs. Participants then used the mockups to test whether they could accomplish a task. They had something tangible about which they might suggest improvements.

5. RESULTS AND DISCUSSION

Accepting the Use Case narratives as written and having to develop the scenarios proved to be very incongruous with the needs, expectations, skills and motivation of our workshop participants. While it did elicit valuable information, participants did not develop scenarios. Instead, they reformulated user tasks. We feel there are two main reasons why this exercise didn't work.

5.1 Preparation of Participants

Participants did not understand the processes and constraints of developing Use Case scenarios. The semi-formal process was foreign to them, and despite minimal training and guidance from the facilitators, they essentially ignored it.

There was a mismatch of purpose. The cognitive characteristics that are well suited to developing Use Case scenarios were incongruous with the motivation and expectations of the group. Participants wanted to describe ways that the system can support them in getting their job done and they were not ready to exclude possibilities. They expected to discuss issues in an open forum. They were motivated to contribute at a high level and had neither the skills, nor the inclination, to develop structured scenarios.

5.2 Readiness to Commit

Participants were asked to accept the Use Case narratives in order to proceed with developing the scenarios. The narratives originated from the community but participants had no direct ownership, or "buy-in" to them. They felt they were too narrow, and did not represent all possibilities. They wanted to discuss the narratives in terms of their experience. While this dialog was useful and valuable for expanding the existing Use Cases, it indicated a lack of readiness to commit to specific scenarios.

6. CONCLUSIONS

Defining requirements for a digital library can be done collaboratively in conjunction with community members. In our experiment, the community was better suited to participate successfully in task creation rather than scenario development. Task-centered design leads quickly to a tangible, user-testable object that can serve as a picture to see how Use Case scenarios might develop. Task-centered design appears to be better suited for design collaboration in the case where the participants are not skilled in the semi-formal process of developing Use Cases, and where they are neither motivated nor committed to do so.

Cockburn argues that user interface design should come after the development of Use Cases. Our experience in collaborative design suggests that user interface design should come first through the Task-centered design process, where design alternatives can be included rather than excluded. Then, when the mockups and low-level prototypes begin to pass user testing, and the users are ready to commit to a design, the scenarios could be formulated by software engineers or by those who are trained in this process to carry the design further.

7. REFERENCES

- [1] Cockburn, A., 1997. Goals and use cases. *Journal of object-oriented programming*, (Sep 1997), pp 35-40, and (Nov-Dec 1997), pp 56-62.
- [2] Green, T. R. G. and Petre, M., 1996. Usability analysis of visual programming environments. *J. Visual Languages and Computing*, 7, pp 131-174.
- [3] Lewis, C. & Reiman, J., 1993. Task-centered user interface design. Available via ftp.cs.colorado.edu/pub/cs/distrib/clewis/HCI-Design-Book