Use-Cases and Personas: A Case Study in Light-Weight User Interaction Design for Small Development Projects

Gary Randolph
Purdue University, Kokomo, Indiana, USA

gbrandolph@puk.indiana.edu

Abstract

While user interface design is generally considered important to the success of an information system, in both design and practice traditional methodologies it is often left to the last when most of the rest of the information system design has been finalized. What are needed are tools to design a logical model of user interaction requirements. This paper presents how two tools, personas and use-cases, were used together to specify the logical user interaction requirements in the actual design of a small information system. While these tools would not be appropriate for large projects, they worked very well in this case and provided the client with a successfully implemented product.

Keywords: user interaction design, user interface design, use-cases, personas, case study

Introduction

An information system is an arrangement of people, data, processes, interfaces, networks, and related technologies that interact to support the information needs of an organization (Whitten, Bentley, & Dittman, 2000). Organizations devote large amounts of effort and resources to developing these information systems. Once they are developed, it becomes crucial both to the success of the information system and the success of the organization that users use the information system appropriately and effectively.

Yet users of information systems often find those systems unusable. One review of software research found that the average software program has forty design flaws that impair employees' ability to use it, costing up to 720% in lost productivity (Kreitzberg, 1996). The discipline of systems analysis and design benefits from many established tools and techniques, including Project Evaluation and Review Technique (PERT) for project management, Entity Relationship Diagrams (ERD) for data design, Data Flow Diagrams (DFD) for process design, and Unified Modeling Language (UML) to name a few. One area that generally lacks formal tools and techniques is user interface design.

Numerous approaches of user interface design have been proposed, so many that it is unlikely that "one size fits all." This paper presents the results of a case study project to develop a small information system. For this small project two tools were used for the user interface design: use-cases and personas. This paper will review the literature on user interaction design leading to the selection of these two tools for this particular case study. The paper will discuss how these tools were applied in this development project. Finally, the paper will discuss the results.

Literature Review

In human-computer interaction the interface is what users see and work with to use a product (Hackos & Redish, 1998). However, the human and the computer do not share equal roles in the interface. The user is flexible and adaptable (Mayhew 1992), while the system is neither. One result of this inequity is that the responsibility for the interaction is placed on the user. Another result is that the design of the interface is an important issue in how users perceive the system. Barki and Hartwick (1994) concluded that user participation and involvement towards a system affect their productivity and attitude in the workplace.

User communication with the system is expressed in two languages: the action language and presentation language (Gerlach & Kuo, 1991). The user communicates through actions to tell the computer what tasks or commands need to be done. The computer communicates through a presentation language to ask about the tasks and objects and to respond to requests with the result of the operations. Both of these languages allow interaction and communication about common tasks and domains.

Foley and van Dam (1982) devised a four-level user interface model. The conceptual level describes the domain of tasks through which the user and the system interact, such as the objects that are to be manipulated and their relationships. The semantic level defines the meanings of words, incorporating objects that are part of the presentation language of the system, such as menus, dialog boxes, etc. The syntactic level describes the grammar that allows combining words into a meaningful context. This includes the appearance and disappearance of objects on the screen and also the order of actions done by the user. The fourth level is the lexical level, dealing with how the words are expressed. This includes font types, colors, lines, etc. Satzinger and Olfman (1998) demonstrated that interface consistency is an important element in user interface design, based on the Foley and van Dam model. Information presentation and display formats has an impact on user preferences and decisions (Tractinsky & Meyer, 1999).

The GOMS (goals, operators, methods, and selection rules) cognitive model was developed by Card, Moran, and Newell (1983). The model states that humans posses goals, such as creating a document in a spreadsheeet, and subgoals, such as entering numbers in the spreadsheet). These goals are attained by using methods or tasks provided by the user interface. Operators are basic perceptual, motor, or cognitive tasks, such as clicking a mouse on a browse button and recalling folder and filenames. The selection rules are the control mechanisms for selecting among the available methods to be used to achieve a goal.

Shneiderman (1998) proposed earlier the object-action interface (OAI) model, which emphasizes the visual manipulation of user objects and actions. In the OAI model the first step is to understand the action or task to be accomplished. The task is then broken into individual steps. Once the steps have been defined, a metaphor can be applied. A metaphor is a non-user interface construct applied to the user interface, such as the desktop, and recycle bin metaphors used in Microsoft Windows. Kendall and Kendall (1993) describe metaphors as "the cognitive lenses we use to make sense of all situations" (p. 149). Metaphors affect the ease of learning and using software (Madsen, 1994).

The process of user interface design can be defined as "the process of elaborating and the documentation of a design which leads to program code which implements the Presentation Layer, Views or 'screens' within a computer system" (Anderson, 2000, pg. 1). This is known by varying names: user interface design, user interaction design, human-centered development, user centered design, and more. Whatever it is called, it can be defined as the process of designing the interaction between users and a computer information system. It includes not just "look and feel" issues, but all aspects of system workflow and system requirements for human interaction (Randolph, 2002). "There is more to interfaces than windows, icons, pull-down menus, and mice" (Raskin, 2000, pg. 1)

While interface design is usually considered important to the success of an information system, traditional design methodologies generally leave interface design as a last step to be completed (Burns & Madey, 2001). By this time the options for user interface design are extremely limited. The budget and most of the schedule have been spent. Reworking major parts of the design for the sake of user interface simply is not a possibility (Raskin, 2000). Yet major design and project management books often place user interface design as being part of or even coming after the technical design phase (Eriksson & Magnus, 1998).

The design process in general involves two creative leaps: a leap from system requirements to the design implications and then a second leap from those implications to the specific features and an implementation solution (Beyer & Holtzblatt, 1998). Thus user data requirements are first distilled from interviews and investigation to build a logical data model. Then an appropriate data methodology is selected based on the user requirements and the logical data model is transformed to a physical data model ready for implementation.

This two-leap process is often applied to designing data, networks, and information system processes. Both leaps are rarely applied to user interface design. The leap that is normally left out is the development of the logical model of user interaction needs. Thus when designers finally get around to designing user interaction with the information system (a) the implementation technology has already been selected, (b) all logical models have already been solidified and, in many cases, already been transformed to physical models, and (c) no logical design exists to guide the design of the physical user interface.

User Interaction Requirements

This paper will use the term user interaction design to refer to the process of designing the logical model of how a user will interact with an information system. It will refer to the process of designing the physical implementation of that model in a computer interface as user interface design. What would be the components of a logical model of user interaction requirements?

Chen and Sharma (2001) stress work environment (physical demands, skill demands, risk demands, time demands), psychosocial environment (social and cultural style), and ergonomic environment (hardware design, anthropometrics, and biomechanics).

Hackos and Redish (1998) state that usable interfaces reflect the workflows that are familiar or comfortable, support the user's learning styles, are compatible in the users' working environment, encompass a design concept or metaphor that is familiar to users, have a consistency of presentation, and use language that are familiar to users.

Beyer and Holtzblatt (1998) declare that the first problem for design is to understand the customers, their needs, their desires, and their approach to the work. Dias (2001) found usefulness and ease of use of primary importance to professional users of information systems. Staggers and Norcio (1993) note a need for development to focus on human activities and to put users' needs ahead of technology considerations. Quesenbery (2001) describes five Es – efficient, effective, engaging, error tolerant, and easy to learn – as requirements for good usability.

Raskin (2000) states that an interface is humane if it is responsive to human needs and considerate of human frailties. Raskin points out that the user population shares common mental attributes and that designers can minimize their work by exploiting what is common to all humans with regard to interface-design requirements. Yet Raskin decries the overemphasis on human commonalities that happens when designers abdicate their design responsibility by simply adopting industry standards. Raskin points to three useful considerations of interface designers: exploiting what is common to all humans with regard to interface design requirements, accommodating differences in design needs between various individuals and groups, and satisfying the requirements of users' tasks.

Finally we will mention Cooper's declaration that the number one goal of all computer users is to not feel stupid (1999). Cooper contends that focusing solely on tasks to be performed without meeting user needs and goals is a recipe for failure. This is evident in so many of the cell phones, remote controls, and digital cameras that we struggle to operate every day. They accomplish many tasks but are so complex to operate that they fail to satisfy our needs or meet our goals.

The recurring themes of these lists appear to be (1) common human physical and mental capacities, (2) individual users' needs and goals, and (3) users' task requirements. Common human physical and mental capacities by definition do not vary between different information systems. All information systems are operated by humans. All humans have these same physical and mental abilities and tendencies. A logical model of these factors would always be the same. Therefore, there is no need to model them. However, they must not be forgotten or neglected. They should be part of the system designer's training and should shape both the logical and physical design.

Logical User Interaction Design Tools

But if we need not model human commonalities, what tools are available for modeling user needs and goals and user task requirements? Hackos and Redish (1998) describe several tools including workflow diagrams, task sequences, task hierarchies, user/task matrices, task scenarios, and affinity diagrams. Beyer and Holtzblatt (1998) suggest the use of affinity diagrams, cultural models, sequence models, and user environment design, among others.

While these tools are appropriate for large projects, they may not be as appropriate for small projects with a relatively small scope and undertaken by a small design team. The standard tool for small projects seems to be prototypes. However user interface prototypes generally assume an implementation technology, which is inappropriate at the logical design stage. Simple paper or whiteboard mock-ups could be logical design prototypes. But the prototype still needs to be based upon some logical analysis of user requirements. What we are searching for is the logical model that lies behind the prototype and guides the prototype's development. To build the prototype simply on intuition and seat-of-the-pants experience is to invite design oversights. Users are unlikely to catch subtle oversights in the prototype, but they will eventually discover them in the implemented information system leading to frustration and lost productivity.

In an effort to identify appropriate logical user design tools for small projects, the author applied two tools – personas and use-case models – in a typical small design and development project.

Personas

A design technique that specifically targets user interaction design is *Goal-Directed Design* developed by Alan Cooper (1999). The principle tool of goal-directed design is the creation of what Cooper calls p*ersonas*. With personas, the goal-directed designer develops “a precise description of our user and what he wishes to accomplish” (pg. 123). A more definition might be, "User models, or personas, are fictional, detailed archetypical characters that represent distinct groupings of behaviors, goals and motivations observed and identified during the research phase" (Calde, Goodwin, & Reimann, 2002). Personas can be thought of as hypothetical users – fictional people who represent classes of users. Persona design begins with brainstorming on the types of people who will use the system. These characters are then named and fleshed out with back-story and an understanding of their goals for using the system until they become like real people.

Personas are defined by their needs and goals. These include their personal goals as well as their goals for the system. A goal-directed design project may, and probably will, have multiple personas because different kinds of users with different goals will use the system. The system may not be designed for all personas. However, each system will have at least one *primary persona*. A primary persona is someone who must be satisfied with the system for it to be considered a success and who cannot be satisfied with an interaction designed for another persona. The user interaction designed for each primary persona should be based on the needs and goals of that persona.

Personas can be confused with actors used in use-cases, as discussed below. They are really very separate concepts. Personas represent users of the system. Use-case actors represent "anything that needs to interact with the system to exchange information. An actor is a user, a role, which could be an external system as well as a person" (Whitten, Bentley, & Dittman, 2000). While time or an external information system could be a use-case actor, they would never be personas. Another difference between personas and actors is the degree of detail in their definition. A use-case actor might be defined with one or two words, such as member or marketing. Personas require back-story, needs, and goals.

Cooper cites several cases in which the development of personas guided the user interaction design process. In the design of an airline seat entertainment system, the persona process identified a non-computer-literate senior traveler, a frequent business traveler, and a pre-teen among the cast of characters. As they designed for these personas, the system interface gained a physical scroll knob that senior citizens could understand and operate even with arthritis, a touch-screen menu that frequent flyers can use for shortcuts, and views of movie posters that would be entertaining and inviting to all three personas.

Use-Cases

While personas can aid in designing user interaction that meets a user's needs and goals, it often seems a rather nebulous tool for detailing the specifics of users' task requirements. For that task, use-cases can be a simple but useful tool.

Use-cases are part of the definition of Unified Modeling Language (UML) and its methodology (Fowler & Scott, 1997). A use-case "describes a sequence of actions a system performs that yields a result of value to a particular actor" (Leffingwell & Widrig, 2000). In practice, use-cases bridge the gap between the unstructured descriptions of the system that are gleaned from user interviews and a more formal model of the information system (Woo & Robinson, 2002).

Kenworthy (1997) outlines eight steps to developing use-cases.

1. Identify who is going to be using the system directly. These are the actors.
2. Pick one of those actors.
3. Define what that actor wants to do with the system. Each thing that the actor does with the system becomes a use-case.
4. For each use-case, decide on the most usual course of events when that actor is using the system. This is the basic course.
5. Describe that basic course in the description for the use case. Describe it in terms of what the Actor does and the system does in response. Use implementation-independent terms.
6. When the basic course is described, consider alternates courses of events and add those as extending use-cases.

7. Extract common courses as "used" use-cases.
8. Repeat steps 2 - 7 for all other actors.

Methods

Community Hospital Anderson is a medium-sized hospital in a town of 50,000 inhabitants in central Indiana. With more than 200 beds and 1100 employees, Community Hospital Anderson requires that employees go through extensive and on-going training. Some 2000 different training classes are offered to some or all employees each year.

Obviously managers need a way to track which employees have received which trainings and also which employees still need to receive which trainings. In-house personnel had developed an application in Microsoft Access that was meeting most of their needs. However, several shortcomings were noted in the system:

1. The user-interface, which was originally created by and for the expert users, was difficult for casual users, such as department managers, to operate.
2. The system was often slow to respond over the corporate network.
3. The look-and-feel of the system was inconsistent with corporate intranet sites.
4. Management had a desire to move the application to web programming on the intranet provided that the move was consistent with the other goals.

The scope of the proposed system was such that a design and development team of one person could accomplish the task. Thus, it was a good candidate for a case study of logical user design for a small information system.

An analysis of the proposed system was done by studying the existing Microsoft Access application and interviewing key users. It was determined that the underlying data structure was sound but the user-interface was less than optimal. It was also determined that Microsoft Access was the wrong tool for the number of users working with the system. It was also determined that the hospital had in place the necessary database tools and infrastructure to support a successful web implementation of the system. A decision was made to continue with the project.

A logical data structure for the proposed system was designed. Since the Microsoft Access data design was basically sound, it was the basis for the model but was revised based upon interviews with key users and upon the user interaction design as it was developed.

Persona Design

Personas (see Figure 1) were developed based upon interviews with key users. Because this was a fairly simple system, only three personas were determined to be needed. These three fictional, archetypical users represented all the users of the proposed system. The personas, once developed, were shared with the key users who checked them for accuracy of the assumptions.

Andrea Mohler became the archetypical power user of the system. As an educational specialist she would be a frequent user. She is both comfortable with computers and will quickly become familiar with the system. Andrea's biggest need is to be able to navigate quickly through the system and for it to work effectively for her. She will often need to process groups of records in mass as when a class is given to thirty employees at one time or a new employee is given a battery of trainings.

PERSONAS

PROJECT: **Inservice Tracking System**	**PROJECT MANAGER:** **Gary Randolph**
CREATED BY: **Gary Randolph**	**LAST UPDATED BY:**
DATE CREATED: **06/17/2002**	**DATE LAST UPDATED:**

Name	Job Function	Description	Personal Goals	Goals for System	Interface Implications
Andrea Mohler	Education Specialist Group	Frequent (even daily) user of system. Computer savvy.	1. Sees the system as a reflection on her. She wants it to be well thought of.	1. Needs to process groups of training records in mass	Power-user functionality
Bill Fernandez	Manager Group	Infrequent user of system. Has computer on desk but not very computer savvy. Very busy.	1. Limited time. System has to make Bill's life easier or it won't be used.	1. Easy to use and error proof 2. Needs to process training records for multiple employees in department.	Simple interface with links to access all available functionality from all pages
Betty Jenkins	Employee Group	Health professional. Not overly computer savvy but familiar with Internet. Interested in making sure the system has correct information on her. Has access to a shared computer.	1. Wants to make sure that she gets credit for every training she attends 2. Needs to be thinking about health care and not computers	1. Wants to check information in system on her 2. Needs to enter special activities for self.	Simple interface

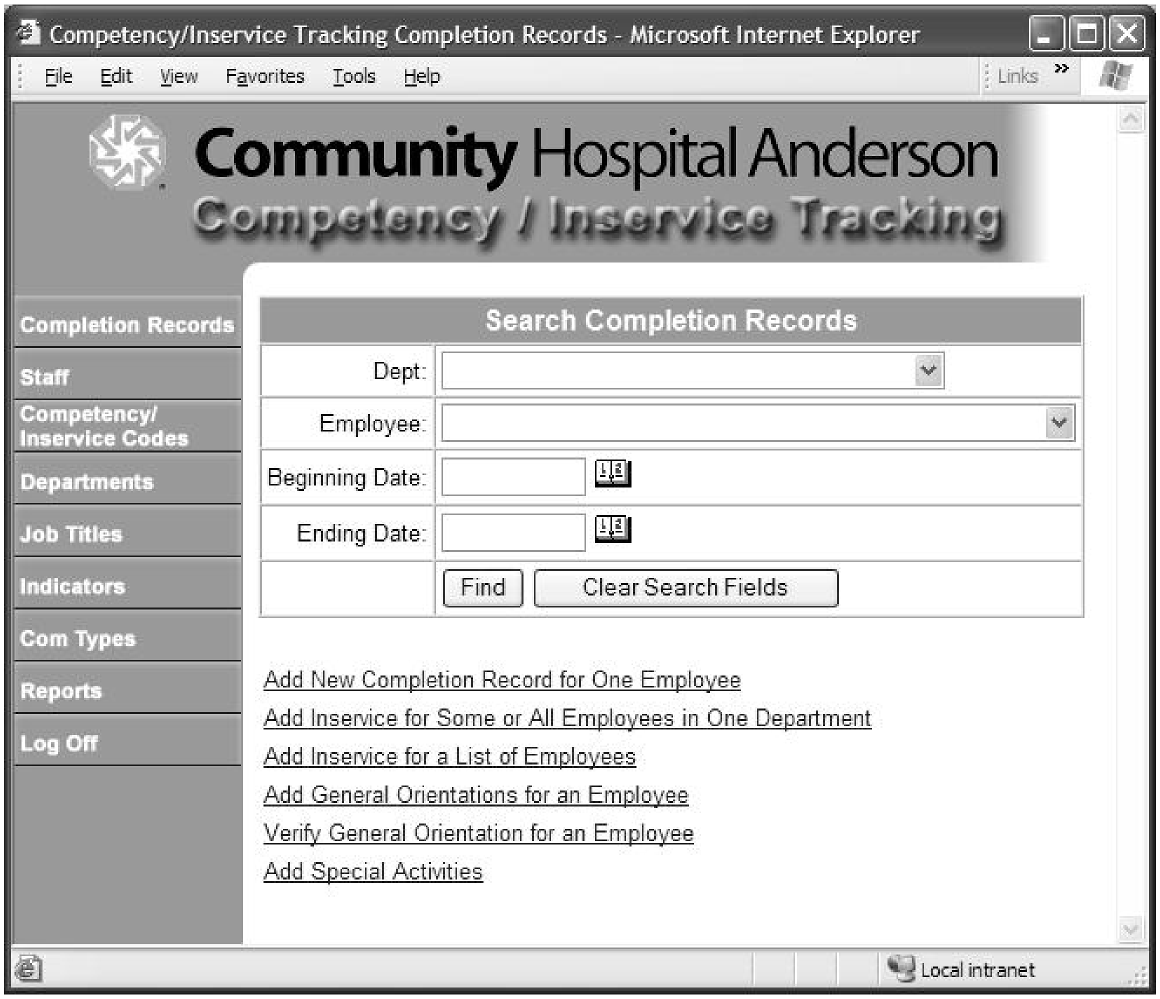
Figure 1: Personas created for case study as part of logical interaction design.

Bill Fernandez was the name given to the manager persona. Bill doesn't use a computer often and will use the system even less frequently. When he does use the system he needs to get in and out quickly. He needs consistent and clean user interaction.

The final persona was Betty Jenkins. Betty is a very competent health care professional but a computer novice. When she uses the system it will be on a shared computer in a public space. She is well aware that using this system is not part of her job but that she has to use it sometimes to fulfill certain job requirements. She is also aware that her peers may be watching her use it. She needs a simple user interaction even more than Bill does.

During the design phase when program specifications were written and the coding begun, the personas were not forgotten. They guided many implementation decisions. For example, the need for clean and consistent user interaction for Bill and Betty led to the following choices:

1. After users log in, all functionality not available to that user would be hidden to minimize clutter.
2. The expert users would receive more options on their screens after login. But given their better familiarity with the program, they should be able to navigate through those options.
3. The navigation buttons at the left of the screen (see Figure 2) allowed casual users a constant reference point and an ability to always get to any part of the program with a few clicks. These navigation buttons also allowed expert users to quickly go from one task to another.
4. The various screens were kept obsessively consistent. Each section of the program had the same Search and Add New options. The results of a search were always shown in a browse window with Edit, Delete, and Add New options always in the same place. This would be especially useful for casual users to make all parts of the user interaction as easy to navigate as possible.


Competency/Inservice Tracking Completion Records - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Links
Community Hospital Anderson
Competency / Inservice Tracking
Completion Records
Staff
Competency/ Inservice Codes
Departments
Job Titles
Indicators
Com Types
Reports
Log Off
Find
Clear Search Fields
Local intranet

Search Completion Records

Dept:

Employee:

Beginning Date:

Ending Date:

Add New Completion Record for One Employee

Add Inservice for Some or All Employees in One Department

Add Inservice for a List of Employees

Add General Orientations for an Employee

Verify General Orientation for an Employee

Add Special Activities

Figure 2: Part of completed user interface

Use-Case Design

During the design phase the personas were used as actors for the development of use-case models (see Figure 3). Use-case models were not created for all housekeeping chores required in the system but for the major, complex interactions between user and system. As with the personas, the Use-Cases were shared with the key users who checked them for accuracy of the assumptions.

After the implementation technology was selected, these use-cases were expanded into pseudocode models of the routines needed to accomplish them. This light-weight approach was successful because of the small size of the project and the small size of the development team.

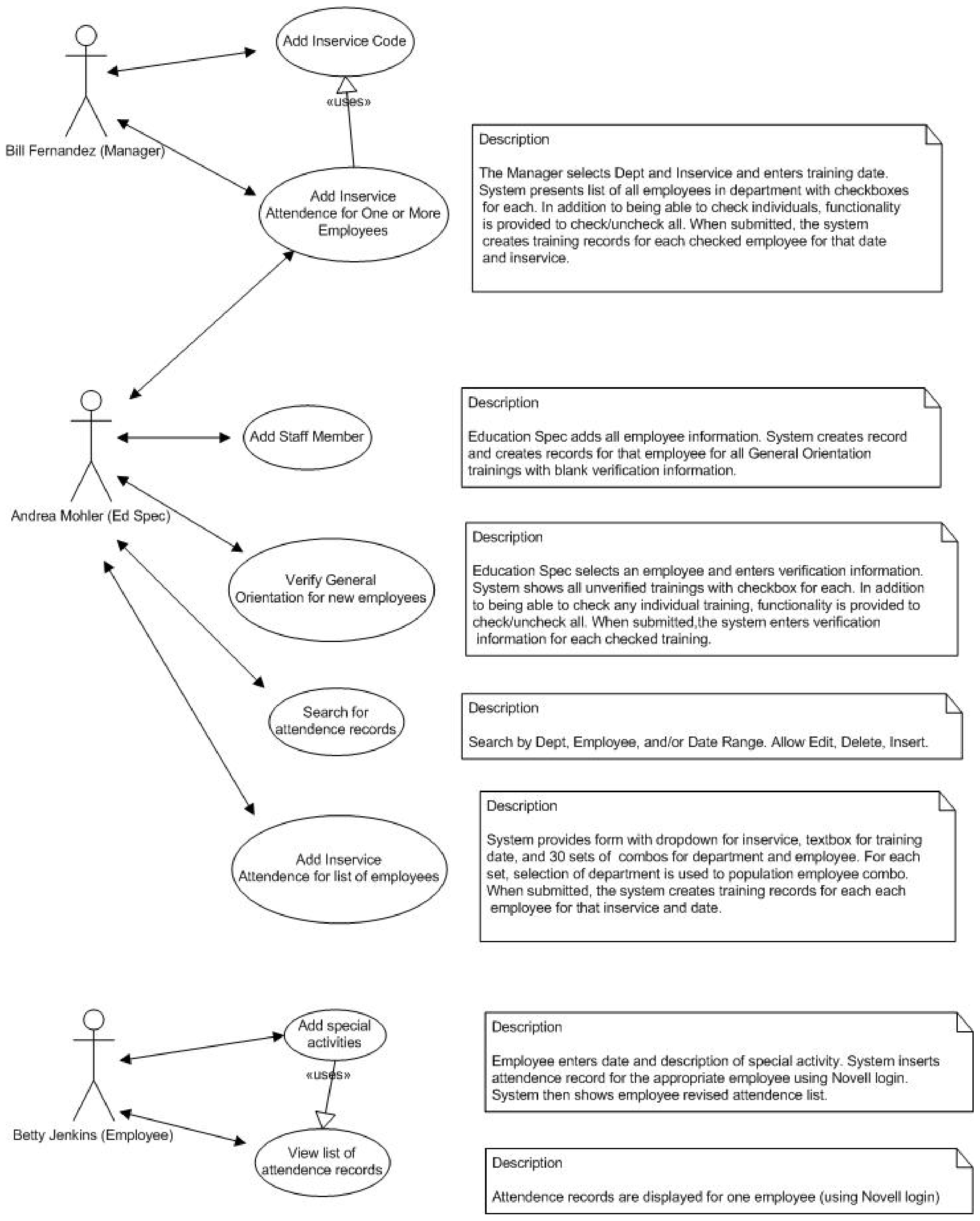

Add Inservice Code
«uses»
Bill Fernandez (Manager)
Add Inservice Attendence for One or More Employees
Description
The Manager selects Dept and Inservice and enters training date. System presents list of all employees in department with checkboxes for each. In addition to being able to check individuals, functionality is provided to check/uncheck all. When submitted, the system creates training records for each checked employee for that date and inservice.
Add Staff Member
Andrea Mohler (Ed Spec)
Description
Education Spec adds all employee information. System creates record and creates records for that employee for all General Orientation trainings with blank verification information.
Verify General Orientation for new employees
Description
Education Spec selects an employee and enters verification information. System shows all unverified trainings with checkbox for each. In addition to being able to check any individual training, functionality is provided to check/uncheck all. When submitted,the system enters verification information for each checked training.
Search for attendence records
Description
Search by Dept, Employee, and/or Date Range. Allow Edit, Delete, Insert.
Add Inservice Attendence for list of employees
Description
System provides form with dropdown for inservice, textbox for training date, and 30 sets of combos for department and employee. For each set, selection of department is used to population employee combo. When submitted, the system creates training records for each each employee for that inservice and date.
Add special activities
«uses»
Betty Jenkins (Employee)
View list of attendence records
Description
Employee enters date and description of special activity. System inserts attendence record for the appropriate employee using Novell login. System then shows employee revised attendence list.
Description
Attendence records are displayed for one employee (using Novell login)

Figure 3: Use-Cases developed as part of logical user interaction design

Results and Discussion

Following implementation of the system, users of the system were surveyed as to the usability of the system. Users were asked eleven questions that they responded to with a five-point Likert-type scale. The results are shown in Table 1. Questions 1-5 deal with general user acceptance, while questions 6-11 address specific features within the system. The weighted averages of all the questions indicate good user acceptance.

These results, however, need to be interpreted with an understanding that this was a real-world system, not a controlled laboratory experiment. The programming design discussed in this paper was one part of the final system that was implemented. That system included tutorial videos, written instructions, and additional programming functionality. In the survey, the users could not segregate their evaluation of any one part of the system. Still it does appear that this design methodology worked well for this particular small system.

Personas and use-cases are not the Holy Grail of logical user interaction design tools. In some projects, especially large projects, they would be totally inadequate. Anderson (2001) has argued, in fact, that use-cases can be the "death" of good design, mainly because of who writes them. Some theorists assign use-cases to the business analyst to describe what the proposed system must do. Other theorists assign them to programmers during the design phase. Anderson contends that neither business analysts nor programmers are competent to do user interaction design. Of course with the present case study and its one-person design team, that was not an issue. But it is an important criticism for large projects. Managers of large projects can perhaps insist that use-cases be (1) done early, (2) be written at a high-level in non-implementation specific language, and (3) be passed to competent user interaction designers at an early date for whatever user interaction design tools are appropriate for that project.

Table 1: Results of User Survey

Question	**Sample Size**	**Weighted Average***
1. I have used the system.	59	1.78
2. The system meets my needs.	59	2.31
3. I understand how to use the system.	59	2.34
4. The look of each screen in the system is similar.	59	1.92
5. The order of the steps on each screen makes sense.	59	2.08
6. I understand how to use the Signature Sheet Request function.	59	1.93
7. I understand when to use the Signature Sheet Request function.	59	1.80
8. The Signature Sheet form is easy to understand and follow.	59	1.81
9. I understand how to use the Code Request function.	59	1.98
10. I understand when to use the Code Request function.	59	2.00
11. The Code Request Form is easy to understand and follow.	59	1.97
* 1=Strongly Agree, 2=Agree, 3=Neutral, 4=Disagree, 5=Strongly Disagree		

This case study reports one project in which personas and use-cases were useful and successful tools for guiding the user design process and for discussing user design requirements with users. These simple tools would probably not be sufficient for projects of larger scope.

In this era of extreme programming and emphasis on rapid development, all modeling and documentation must be justified. But we should remember that in the case of design and modeling, less is not more; it is always less. It is just that more is not always appropriate or cost-effective. On the other hand, no user interaction and interface design is always a mistake. The tools used in this case study are definitely better than no tools at all and no logical user interaction design at all. In this particular small project they proved to be just the tools that were needed.

References

Anderson, D. (2000). *Extending UML for UI*. Retrieved 11 Dec. 2002 from http://www.uidesign.net/2000/papers/TUPISproposal.html

Anderson, D. (2001). *Are use cases the death of good UI design?* Retrieved 11 Dec. 2002 from http://www.uidesign.net/1999/imho/feb_imho.html

Barki, H. & Hartwick, J. (1994). Measuring user participation, user involvement, and user attitude. *MIS Quarterly*, *18* (1) 59-82.

Beyer, H., & Holtzblatt, K. (1998). *Contextual design*. San Francisco: Morgan Kaufman.

Burns, J., & Madey, G. (2001). A framework for effective user interface design for web-based electronic commerce applications. *Informing Science, 4* (2), 67-75.

Chen, Q., & Sharma, V. (2001). Human factors in interface design: An analytical survey and perspective. In M. Khosrowpour (Ed.), *Managing information technology in a global economy: Proceedings of the 2001 Information Resources Management Association International Conference* (pp. 10-13). Hershey, PA: Idea Group Publishing.

Calde, S., Goodwin, K., & Reimann, R. (2002). SHS Orcas: The first integrated information system for long-term healthcare facility management. *Conference on Human Factors and Computing Systems, Case studies of the CHI2002/AIGA Experience Design Forum.* New York, NY: ACM Press.

Cooper, A. (1999). *The inmates are running the asylum*. Indianapolis: Sams.

Dias, D. (2001). Motivation for using information technology. In M. Khosrowpour (Ed.), *Managing information technology in a global economy: Proceedings of the 2001 Information Resources Management Association International Conference* (pp. 326-328). Hershey, PA: Idea Group Publishing.

Eriksson, H., & Magnus, P. (1998). *UML (Unified Modeling Language) toolkit*. New York: Wiley.

Foley, J. and van Dam, A. (1982). *Fundamentals of interactive computer graphics.* Reading, MA: Addison-Wesley.

Fowler, M., & Scott, K. (1997). *UML distilled: Applying the standard object modeling language*. Boston: Addison Wesley.

Gerlach, J. H. & Kuo, F. (1991). Understanding human-computer interaction for information systems design. *MIS Quarterly*, *15* (4), 526-549.

Hackos, J., & Redish, J. (1998). *User and task analysis for interface design*. New York: Wiley.

Kendall, J. E. & Kandall, K. E. (1993). Metaphors and methodologies: Living beyond the systems machine. *MIS Quarterly*, *17* (2) 149-171.

Kenworthy, E. (1997). *Use case modeling: Capturing user requirements*. Retrieved 9 December 2002 from http://www.zoo.co.uk/~z0001039/PracGuides/pg_use_cases.htm

Kreitzberg, C. (1996). *Ending the software struggle: A strategic analysis*. Retrieved 10 December 2002 from http://www.cognetics.com/presentations/charlie/p_struggle.html

Leffingwell, D., & Widrig, D. (2000). *Managing software requirements: A unified approach*. Boston: Addison Wesley.

Madsen, K. H. (1994). A guide to metaphorical design. *Communications of the ACM*, *37* (12), 57-62.

Mayhew, D. J. (1992). Principles and guidelines in software user interface design. Engelwood Cliffs, NJ: Prentice Hall.

Quesenbery, W. (2001). *What does usability mean: Looking beyond 'ease of use'*. Retrieved 10 December 2002 from http://www.cognetics.com/presentations/whitney/more-than-ease-of-use.html

Randolph, G. (2002). Classroom exercises for improving understanding of user interaction design. In *Proceedings of the 5th annual conference of the Southern Association for Information Systems* (pp. 226-230). Savannah: SAIS.

Raskin, J. (2000). *The humane interface: New directions for designing interactive systems*. Boston: Addison Wesley Longman.

Satzinger, J. W. & Olfman, L. (1998). User interface consistency across end-user applications: The effects on mental models. *Journal of Management Information Systems*, *14* (4), 167-193.

Shneiderman, B. (1998). *Designing the user interface: Strategies for effective human-computer interaction*. Reading, MA: Addison-Wesley.

Staggers, N., & Norcio, A. (1993). Mental model: Concepts for human-computer interaction research. *International Journal of Man-Machine Studies, 38 (*4), 587-605.

Tractinsky, N. & Meyer, J. (1999). Chartjunk or goldgraph? Effects of presentation objectives and content desirability on information presentation. *MIS Quarterly*, *23* (3).

Whitten, J. L., Bentley, L. D., & Dittman, K. C. (2001). *Systems analysis and design methods* (5th ed.). New York: McGraw-Hill.

Woo, H., & Robinson, W. (2002). A light-weight approach to the reuse of use-cases specifications. In *Proceedings of the 5th annual conference of the Southern Association for Information Systems* (pp. 330-336). Savannah: SAIS.

Biography

Gary B. Randolph is associate professor of computer technology for Purdue University. His teaching and research interests include user interface design and curriculum design. In addition to teaching, he is active in consulting for a variety of companies across the nation. Following a career in industry, he earned a M.S. degree from Ball State University. He is a member of AIS, AITP, PASS, and SIGITE.