The UI Design Process

Planning, Managing, and Documenting UI Design Work: A CHI 99 Workshop

Paul McInerney and Rick Sobiesiak

Introduction

The root cause of many user interface (UI) design deficiencies is not a lack of knowledge about human-computer interaction principles nor a lack of information on user needs. Rather, many UI deficiencies arise because the UI design process is ad hoc and the design is not communicated successfully to the programmers who will implement it. Many UI designers are seeking and discovering ways to plan, manage, and document UI design work more effectively. This workshop provided an opportunity for participants to share lessons learned and obtain advice from other participants.

In the weeks leading up to the workshop, participants selected the specific topics that were of prime concern to them. As a result, we narrowed the focus of the workshop to the following topics:

- Division of UI design activities into stages
- Division of labor and interdisciplinary collaboration
- Collaborating in geographically-dispersed projects
- Writing the UI specification
- Defining the maturity of the UI design process.

The following sections summarize the results of the workshop activities for each of these topics.

Division of UI Design Activities into Stages

As with any complex activity or process, it is useful to divide the UI design process into stages. Stages provide a means to focus on a subset of issues (that is, divide and conquer), communicate progress (for example, the design has reached stage 2 of 3), and provide checkpoints to adjust the design direction. As well, the project environment in which UI

Aspect	*Early in the Project*	*Late in the Project*
Information about requirements needed as input into UI design	Unknown & unstable	Known & stable
Focus of UI designer effort	Ensuring coverage of requirements in the UI design	Finding workarounds to late-breaking implementation problems
Nature of UI design solutions sought	Exploring alternate designs	Removing defects & evolving a single design option

Table 1: Comparison of aspects of UI design

design takes place evolves from the early to later stages, as summarized in the table below.

Participants varied widely in how they divided the UI design process into stages and labeled these stages. Some participants described a spiral approach in which each iteration addressed a portion of the system and its UI. In one such scheme, steps included: develop high level design alternatives, evaluate and select high level design, detailed design, and prototype/implementation. Another approach was divided into the more traditional waterfall phases: analysis, design, implement, evaluate. The design phase included: object design, navigation design, low-fidelity prototype, and the UI specification document. Some other approaches included:

- Conceptual design (including concepts, user objects and actions), logical design (screen definition and navigation) and physical design (detailed design of each screen).
- Initial design (addressing mental model and navigation), revised design, and final design.

Despite the variation in approaches, participants generally felt comfortable with the definition of stages shown below. Also shown below are estimates of the percentage of total UI design team effort allocated to each stage. It is interesting to note that up to a third of the UI design effort is performed in the last stage after the design is supposedly complete. Participants found there is considerable work in addressing issues that arise late in the project and in working with the implementation team to ensure the design is implemented as it was intended.

Some additional discussion points were:

- One interesting stage that was discussed was that of a "concept car" version of the UI design. This is a visionary prototype created early in the project to rally a team around a common vision of the UI design direction.
- The use of low-fidelity prototyping was discussed. While participants like low-fidelity prototypes, other stakeholders (clients, development management) expressed a preference for high-fidelity prototypes because they considered low-fidelity prototypes unprofessional. One participant found a middle ground by having a graphic designer render his low-fidelity sketches, giving them a professional polish and at the same time conveying the preliminary stage of the design.

Other issues raised but left for future discussion include:

- The nature of iteration in the UI design process.
- Allocation of specific UI design activities to either the UI architecture or UI detailed design phases.
- Interleaving UI design activities with other usability engineering activities such as user/task analysis and usability evaluation.

Division of Labor and Interdisciplinary Collaboration

This segment was facilitated through a discussion of the interdisciplinary collaboration model of the architecture profession. Given that architecture is a more mature and established discipline than UI design, we felt we could learn from understanding the collaboration model used in architecture and why it differs from the UI design model.

The key elements of the architecture collaboration model are:

- Clients engage architects directly for a project. The architect acts as the chief liaison to the client for the entire project and is responsible for overall project management.
- Architects, in turn, engage the engineers and subcontractors required for the project. Furthermore, the architect maintains an overall supervisory role over the work of the other professions involved in the project.

UI Design Phase	*Description*	*Effort*
UI architecture	Defining the UI at a gross level, defining the key design direction	20 – 30%
UI detail design	Defining the UI at a level detailed enough to serve as instructions for the programmers	50%
UI design change management & verification	Design rework to address issues that arise after the UI design freeze; also includes work to verify that the UI was implemented as specified	20- 30%

Table 2: Stages of UI design and the relative effort allocated to each stage

- During the design stage, architects focus their efforts on building models of the artifact (for example, an office building or a house) under consideration. These models eventually result in producing a full specification of the artifact before the construction of the artifact begins. The architect will typically engage engineers during the design process to ensure the technical feasibility of the design, since the goal is to ensure that the construction phase focuses on implementing rather than reworking the design.

The workshop participants discussed how the architectural model can apply to UI design. Some of the key points were:

- Technical invention is often required in UI design. Architecture typically involves the use of standard components (for example, I-beams) that are well understood in terms of their properties and behavior. Furthermore, architecture is typically focused on human tasks (for example, mass movement through an office building) that are relatively well understood. UI design often involves the invention of new components (for example, custom graphical controls) for relatively new tasks. We do not have as much history and insight to draw upon in UI design for understanding how the artifacts we design will be used. Consequently, considerable innovation is often required.
- At least one of the workshop participants works as a UI designer who is the chief liaison to the client for the overall project, similar to the architectural model. It was noted that the chief technical architect for the project needs to collaborate with the UI designer from the very start of the project for this model to succeed.
- Several workshop participants noted that their roles as UI designers and human-computer interaction (HCI) professionals have broadened over time to encompass higher-level responsibilities, including those typically associated with a business analyst. Such roles tend to be focused more on requirements and user needs than on detailed design issues such as the layout of UI controls. Furthermore, this seems to be a logical extension of the HCI professional's role. Thus, within the workshop group we saw some evidence that the scope of the work of UI design professionals is growing in the direction of the architectural model, although there is a considerable difference between how the two disciplines are practiced.

Collaborating in Geographically-Dispersed Projects

For this topic we examined two case studies: a short-cycle web site development environment, and a large-scale commercial software product. Both of these case studies focused on issues related to collaborating in geographically dispersed project situations. Both cases involved project teams which were spread across more than one site and time zone.

A key point that emerged is that cross-site collaboration is not free; it requires an extra investment of time and resources that would not otherwise be necessary. This investment needs to be considered when planning cross-site design work.

Some of the other key points were:

- Many of the challenges associated with cross-site projects are similar to those faced by single-site projects. For example, integration of a solution across components can be a challenge regardless of whether or not the component designers are located in one site. However, cross-site projects can magnify this challenge.
- Communicating designs across sites is a key issue. Formal design specifications become an important tool for creating a common understanding of designs across the team. Informal, ad hoc design communication is prone to miscommunication.
- Regular meetings (both face-to-face and remote) across sites are important vehicles for communicating overall project status as well as design information.
- Face-to-face meetings are important for team building and for facilitating detailed design work. Despite the drawbacks associated with travel, the general practice seems to be that some amount of regular face-to-face collaboration is required.
- Various commercially available software tools for remote/synchronous collaboration are commonly used across many organizations, often for exchanging design information.
- E-mail and phone communications are very common in cross-site projects. Videoconference meetings are used somewhat less frequently. Although the value of videoconference meetings for communicating emotive content is recognized, the overhead associated with arranging and holding videoconference meetings restricts their use in many cases.
- Various benefits can arise from cross-site collaboration. For one, the diversity of design experience is often enriched by spreading work across sites and organizational cultures. In addition, there can be various opportunities to leverage the resources across sites.

Writing the UI Specification

The UI design needs to be communicated to those who will implement the UI as well as to other members of the project team (for example, technical writers). The entire description of the UI design usually consists of several artifacts including a prototype, a style guide, and a UI specification document. Sometimes usage-based descriptions (for example, scenarios) also form part of the description of the UI. In this part of the workshop, participants compared and discussed document templates and notations for the UI specification.

We examined a wide variety of approaches which were geared to the needs of particular projects. The practices that were discussed included:

- A tool called Storyboard Pro which is an add-in to Microsoft Visual Basic^{TM}. UI designers can create a mockup in Visual Basic and attach the behavior description directly to each UI. Creating the mockup and documenting the behavior becomes an integrated activity performed in a single design/development environment.
- A spreadsheet workbook. One sheet is created to document each application window and spreadsheet columns are

dedicated to information needed by the various disciplines in the project. Project members can show or hide columns of interest to them. With this approach, all the information about a window is available in one place rather than in discipline-specific locations.
- A custom database record. This record includes various fields for entering information about each UI screen in a standard format. The format includes checkboxes for selecting tests that should be performed on this screen as part of software testing. This information is part of a larger repository that includes the entire product specification.
- A set of web pages, one per screen. Contents of a web page include a screen shot, links to other screens that call this screen, behavioral description of each control, and additional notes.

Participants were divided on the merits of writing a stand-alone, distinct UI specification versus contributing UI aspects to an overall system specification. Arguments for a distinct UI specification included:

- Provides the UI design team with a tangible output that demonstrates their contribution to the project. This is especially a concern in consulting situations where a client is paying for a UI design. There needs to be a deliverable to show that the work was completed.
- Allows for better control over the document organization and content. Having a distinct document allows the UI team to optimize the organization of their content rather than slotting their work into a project-wide scheme that may not have been designed with UI design in mind.

Arguments for integrating the UI specification in a system-wide document or organizational scheme included that it provides various disciplines with a holistic view of the system. Integrating the UI design specification for a subsystem (or component) with all the other documentation for that subsystem enables the reader to be able to conveniently read about all the facets of that subsystem. Splitting out the UI design information (or any other facet) increases the chance that discrepancies will be unnoticed until late in the development cycle.

Other issues raised but left for future discussion included how to:

- Communicate changes as the UI design is updated
- Use the specification to support verifying the implementation
- Separate out common versus local design elements
- Distribute various items of UI design information between a written specification, prototype, and style guide.

Defining the Maturity of the UI Design Process

In a manner analogous to the Capability Maturity Model (Paulk, 1995) for software engineering, it would be desirable for UI design teams to be able to self-assess their UI design process against a common benchmark to identify areas for improvement. This part of the workshop began the process of creating such a metric. The first step, defining the levels of maturity is shown below.

The second step involved selecting the attributes of UI design process maturity. As shown below, the potential list is extensive.

The final step was to define the attribute levels. Participants selected a few attributes from the list above and created some scales. The results are very preliminary; an example is shown below.

The following steps were identified for continuing the development of this metric in the future:

1. Selecting a manageable list of attributes to measure
2. Creating the scale for each measurement.

Level of Maturity	*Description - the typical UI design process*
0 - Baseline	No user-centered design (UCD) orientation
1 - Getting Started	New to UCD
2 - Mainstream UCD	With a mature UCD skill set following mainstream , generally accepted UCD practices
3 - Best Practice	Demonstrating an exemplary execution of a UI design process with innovative practices
4 - Ideal/Next Generation	Experimenting with techniques that are not yet proven for full-scale project development

Table 3: Levels of UI design maturity.

People and Organization Factors Organizational structure Multiple disciplines Specialized training Roles and responsibilities *Processes & Procedures Factors* Design lifecycle definition/iteration Work planning Sizing Change control Design reviews Implementation verification Document revision control & updating Central repository	*Notation/Documentation Format Factors* Organization of information Guidelines and style guides Prototypes Design rationale & history Requirements traceability *Tool Support Factors* Level of automation Capability to support sophisticated design process

Table 4: Possible attributes of UI design process maturity

Level of Quality/Maturity	*Description of Aspect: Design Rationale*
0 - Baseline	Use intuition; not recorded
1 - Getting Started	Informal group consensus; expert opinion; design decisions noted
2 - Mainstream UCD	Some key aspects have an explicitly recorded rationale (for example, a design rationale document); some attempt to refer to data in rationale
3 - Best Practice	Formal process/procedure in place for recording rationale
4 - Ideal/Next Generation	High percent of design decisions have a recorded rationale; rationale available for use by future projects

Table 5: Sample draft scale

Conclusion

This workshop was an initial attempt to cover the topic of planning, managing, and documenting UI design work. There remains much to do in this important aspect of UI design practice. We kept the workshop manageable by focusing on five topics: division of UI design activities into stages, division of labor and interdisciplinary collaboration, collaborating in geographically-dispersed projects, writing the UI specification, and defining the maturity of the UI design process. Even within these well-bounded topics, we identified more issues than there was time to discuss.

A common theme throughout the various topics was that of project planning and management. The key point that emerged was that a UI design team needs to plan projects and continually revise the schedules and sizings at each step along the way. It's not an up-front activity that is performed once, but rather an ongoing process.

References

Paulk, Mark (1995). The Capability Maturity Model: Guidelines for Improving the Software Process. New York: Addison-Wesley.

About the Authors

Paul McInerney has worked as a human-computer interaction specialist since 1984 and holds a graduate degree in human factors. He currently works at IBM and has taught a CHI tutorial on writing UI specifications.

Rick Sobiesiak has worked in the IBM Toronto Laboratory since 1984 in various disciplines, including user-centered design, software development, graphic design, and information development. He currently manages a user-centered design department that focuses on database technology products. He holds a graduate degree in computer science.

Authors' Address

Paul McInerney
IBM Toronto Laboratory, Station 4T
1150 Eglinton Avenue East
Toronto, Ontario
Canada M3C 1H7
e-mail: paulmci@ca.ibm.com

Rick Sobiesiak
IBM Toronto Laboratory, Station 2B
1150 Eglinton Avenue East
Toronto, Ontario
Canada M3C 1H7
e-mail: ricks@ca.ibm.com