Middle-Out Design

Audrey Crane
Dubberly Design Office
2501 Harrison Street, #7
San Francisco, CA 94110 USA
audcrane@dubberly.com

Abstract

In 2004, Dubberly Design Office (DDO) was contracted by “HandScript” to design a product that enables physicians to enter orders on a handheld device.

HandScripts’ engineers had been working for a year on an alpha prototype and would continue development during the design of the beta. HandScripts’ physicians were supplying content using a tool that mimicked an early interface for the product and enjoyed their roles as designers. The client had a limited budget and needed usability questions answered immediately.

As in many design projects, there was not time for a top-down or bottom-up design process. DDO had to work “middle-out”.

This case study describes how DDO borrowed the software quality assurance cycle and applied it to managing interaction design—resolving both large conceptual questions and detailed, screen-level questions. This “middle-out” approach used a familiar process to achieve fast, quality work.

Keywords

Analysis, Concept Design, Design Planning, Handheld Devices and Mobile Computing, Health Care, Health, Interaction Design, Process Improvement, Product Design, Usability Research, User-Centered Design / Human-Centered Design, User Experience, User Interface Design.

Note: “HandScript” is a pseudonym. Our client has asked to remain anonymous.

Project / problem statement

Conventionally, physicians write orders freehand on paper. Occasionally specific forms exist for common orders. Our client, “HandScript”, was developing a product that would enable physicians to enter orders for patients (such as ordering a specific test, bed rest, medication, etc.) using a handheld device connected wirelessly to a database.

The content being compiled for the system had to include (or at least allow for) every type of order a physician might want to place—1000’s of orders were possible. HandScript was organizing the content for all possible orders into a hierarchy of forms, an enormous task.

The project as HandScript presented it to Dubberly Design Office (DDO) was to contribute design work for their beta product that would:

1. meet the user’s needs in providing design work that was usable for all types of physicians in spite of the very complex nature of the application.
2. meet our client’s needs, resolving engineer’s questions about solving design problems immediately.

Additionally, DDO wanted to do work that was up to our own standards: logical design based on a well-thought-out premise of what the product is and how people want to use it.

In addition to the product design (interaction) problem, we faced a considerable process management challenge—the client asked us to start in the middle.

Background

The Dubberly Design Office team consisted of:

- Greg Baker
  Designer
  Responsible for designing all aspects of the product: concept, interaction, and visual.
- Audrey Crane
  Design and Project Manager
  Responsible for process, communication and some documentation, and schedule.

The HandScript team consisted of two physicians, a product manager, and two engineers (one of whom had experience designing voice interaction products and would eventually take on the design management role).

The project was initiated in August 2004 and ended in February 2005. The deliverables were scoped as a specific number of user task flow diagrams, visual design mockups, a style guide, and usability study reports.

Challenge

The client came to us in the middle of the project having already invested a year in design, content development, and engineering. The client was understandably looking to move forward—they were not interested in starting over or even in a lengthy reassessment. HandScript (a self funded start-up) wanted to move forward as quickly as possible.

Dubberly Design Office's challenge was to work within the context of three sets of parameters: our clients', our own, and the users' (the problem presented to us by our client).

Client Context

The vast majority of employees at HandScript were engineers. They had been working on a prototype (an alpha) and would not stop development on the beta product while we resolved open design issues. HandScript did not feel that they had time to do extensive research and product concepting. (There was a basic product concept in place, "Our product helps physicians enter orders…" but nothing specific, e.g., "The system consists of a navigation hierarchy, forms, etc…")

Design work for the product had been mainly focused on visual design—colors, font sizes, icons, etc. The designer, content developers and engineers had only began to consider how to organize screens and content. DDO had to address specific user interface (UI) issues that were holding up engineering immediately. Engineering had to be involved in the design decisions so that they could implement the work as it was completed.

The physicians at HandScript entered content via a tool that mimicked the interface of an early prototype, so they had been forced to deal with UI issues as well as content. They had spent a great deal of time on this and did not want to see all their work thrown out. They questioned how DDO might do better than they had at resolving the UI issues.

So our contribution had to work within the context of our client's business:

- Immediately resolve usability issues that engineers were "stuck" on (e.g., How do I fit all this content on the screen? How do I deal with orders that are complex statements?)
- Develop credibility with the physicians on staff.
- Build on the work of the one designer at HandScript (who had recently resigned).
- Show demonstrable progress to a team who felt frustrated by a need to show advancement quickly.
- Work within a limited budget.

DDO Context

At Dubberly Design Office, we ideally use a "top-down" approach for such a complex product. We interview users, develop a product concept, and test and refine that concept. We derive the product flow (interaction) from the interplay of product concept and user goals, and finally design pages, page elements ("widgets") and look and feel. We also play a role in documenting and organizing the content for the pages, in this case order forms (information architecture).

The modeling of the product concept is especially important to us. We have consistently found that very complex problems leave working teams in a sort of muddle—each team or employee only truly understands one small part of the problem. It is the old 'Blind Men and the Elephant' analogy: One man feels the tail and declares an elephant to be like a rope. Another feels its side and declares an elephant to be like a wall, and so on. What each piece of the product is, how it fits with the other pieces, and what the sum of the part adds up to: These concepts are often missing or lost. We've found that working with our clients to create a diagram of the concepts that are key to the product helps us understand the product deeply and thus make logical decisions based on a shared understanding. We've also found that such models help our clients' teams think differently about their products, to organize and understand their work better.

Unfortunately, we could not schedule time up front to model the product concept. Instead DDO had to work on the run while remaining true to our own beliefs about good design. We wanted to:

- Create a system that was clear and well-defined. DDO wanted the product to be consistent and to be as simple as possible (to follow the principle of least means).
- Create a system that was extendible. Developing such a system would ensure that DDO's work would be useful to HandScript as they came across new problems after our contract ended.
- Make suggestions based on clear guiding principles rooted in a thoughtful and logical product concept.

User Context

In order for Dubberly Design Office to help our client be successful, we had to find a way to contribute design thinking that worked for various types of physician-users in such a way that:

- Key concepts were clear (e.g., What is a form? What is an order? Where am I?).
- Expectations were clear (e.g., What am I supposed to do first? What does this thing do?).
- Content was discoverable and legible (given the vast amount of content and the small size of the screen).

Solution

A. Process:

Before beginning work, the DDO team discussed what process would best meet HandScript's needs. Handscript had presented questions from "How can we fit all these words on the screen?" to "How should we navigate the information hierarchy?" to "What should an order consist of?" It was clear that a "normal" design process would not meet the challenges we faced. A top-down process would require most of HandScripts' employees to put development on hold until we reached the point that details about interaction and screen function were documented. A bottom-up approach (simply working out the details as they came up) might be faster but ran the risk of resulting in a disjointed product that was difficult to use.

A possible approach was to persuade the client of the usefulness of a typical design process: Couldn't they see how critical it was to follow the process that we knew was best? Couldn't they spare just a few weeks for research and analysis?

Instead, we tried to experiment with a new and approach. What if, instead of wrestling them into one of our design processes, we found a way to answer their questions when they needed them answered?

What we needed was a kind of "middle-out" approach that would both address details quickly and address larger conceptual questions—so that the detailed work sprang from a logical foundation and resulted in a cohesive product. And the approach had to be something that our client was comfortable with, not a heavyweight or complicated process that would take time to explain and get accepted.

We decided to borrow from our experience in the quality assurance (QA) cycle of software development. Specifically, we introduced the bug tracking process, re-cast to address design issues or "design bugs".

Everyone at Dubberly Design Office has experience with software development. In particular, I have worked as a quality assurance engineer and managed software through QA and release. The QA cycle is a well known iterative process used in nearly all software development: Problems with the product are kept in a list, or more commonly in a bug database. Bugs are given priorities, usually P1 through P5 where P1 is the most critical. Each priority level is explicitly defined. Bugs are assigned to appropriate members of the development team. Bugs are fixed, regressed (to ensure that the bug as reported is actually fixed), and closed on an ongoing basis. The bug list is regularly reviewed—new bugs are added and priorities are assessed. The bug list can be stored in anything from a dedicated database to a simple spreadsheet. Several companies like Elementool or Bugzero provide web-based bug tracking tools that anyone can rent for a relatively modest fee.

The QA cycle is a process that our clients at HandScript were comfortable with. The bug list is a lightweight tool that was also familiar to HandScript. A list of "design bugs", then, would be easy to compile and track, and our client could immediately understand and participate in the process of using such a list to guide the design work.

With that in mind, we decided on a process that would support the “Design Bug” tracking approach:

1. Create a Prioritized Issues List—make a list of design bugs (or, to use a more neutral word, issues)
2. Document “Good Design” Criteria—the criteria by which we’d decide whether a particular issue could be closed
3. Develop Personas and Articulate Their Goals
4. Audit One Section of the Application—to learn about the product and contribute to the Issues List
5. Develop Libraries—make a record of how HandScript had resolved design bugs to date
6. Resolve Issues and Document Outcome

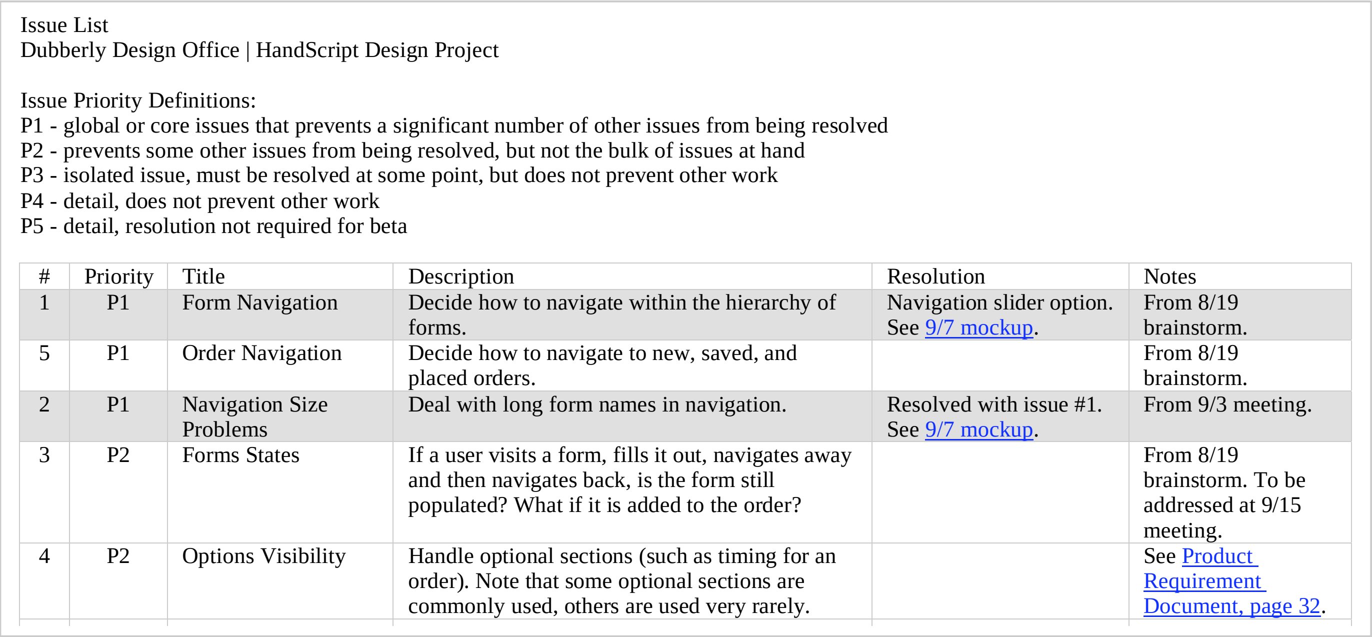
Details about each step follow.

B. Solution details:

Create a Prioritized Issues List

Dubberly Design Office saw problems we wanted to address (“design bugs”, or issues) from very early in the project—even while reviewing the alpha product during the project negotiation. We began keeping a list of those issues. HandScript’s engineers and physicians had their own pressing questions, which were not documented. In a brainstorming session, DDO and HandScript developed a “seed list” of issues we needed to work on, ranging from the very specific (e.g., What do I do about this word that won’t fit on the screen?) to the very general (e.g., What is an order?). After the brainstorming session we reviewed some of HandScript’s documents (e.g., Product Requirements Document, etc.) and found additional open issues. Each issue was listed with an issue number, title, brief description, and origin. The original list included 20 issues (see Figure 1).

DDO intended to use the Issues List to visibly focus our limited time on the most important problems. We would also use it as a tool for setting aside tangential discussions gracefully, without the HandScript team feeling that we were ignoring the inevitable myriad of details (or anyone’s current pet peeve).



Issue List
Dubberly Design Office | HandScript Design Project

Issue Priority Definitions:
P1 - global or core issues that prevents a significant number of other issues from being resolved
P2 - prevents some other issues from being resolved, but not the bulk of issues at hand
P3 - isolated issue, must be resolved at some point, but does not prevent other work
P4 - detail, does not prevent other work
P5 - detail, resolution not required for beta

#	Priority	Title	Description	Resolution	Notes
1	P1	Form Navigation	Decide how to navigate within the hierarchy of forms.	Navigation slider option. See 9/7 mockup.	From 8/19 brainstorm.
5	P1	Order Navigation	Decide how to navigate to new, saved, and placed orders.		From 8/19 brainstorm.
2	P1	Navigation Size Problems	Deal with long form names in navigation.	Resolved with issue #1. See 9/7 mockup.	From 9/3 meeting.
3	P2	Forms States	If a user visits a form, fills it out, navigates away and then navigates back, is the form still populated? What if it is added to the order?		From 8/19 brainstorm. To be addressed at 9/15 meeting.
4	P2	Options Visibility	Handle optional sections (such as timing for an order). Note that some optional sections are commonly used, others are used very rarely.		See Product Requirement Document, page 32.

Figure 1. A sample from the issues list.

to be most effective, (again we borrowed from the quality assurance discipline) DDO introduced issue priorities. The issue priorities were similar to the priorities that might be assigned to software bugs. Our experience told us that the priorities should be defined and documented within the Issues List (also see Figure 1).

Once we had prioritized the list, it was fairly simple to agree on which issues DDO should work on first. We held weekly meetings with HandScript. Each meeting's agenda followed roughly the same outline:

1. Distribute updated Issues List; note issues added from previous week's discussion.
2. Review DDO's materials for proposed solutions to selected issues from previous week.
3. Agree on which issues are closed, which require further revision.
4. Decide which issues to tackle for the following week.

The List was updated weekly and kept online so that it could be hyperlinked to other documents. Each diagram or mockup we produced referenced the issue number that it addressed.

In retrospect, the Issues List was the most important tool for the success of the project. It did several things:

- It helped DDO focus our limited time on the most important issues.
- It helped us gracefully set aside the inevitable new but minor tangential issues that came up. Adding the new issues to the list and prioritizing them assured the client that their concerns were heard and would not be forgotten, and kept discussions focused.
- It gave us a way to track progress for our client and ourselves.
- It documented the date and manner in which each issue was closed. This process gave us a very clear way to say, “That issue was closed, and this was the resolution. Do you want us to re-open it?” This helped us avoid any vague continuing messing-around with issues that we thought were closed—what Jim Barksdale calls playing with dead snakes. The client explicitly determined whether to let the resolution stand when it was questioned, or to reopen it and ask us to spend more time (and money) on it.

In the beginning of the project, we were concerned about jumping into the middle of a work-in-progress without taking the time to work out a product concept with the client. In the end, DDO found that some issues simply couldn't be resolved without modeling the product concept. We reached that conclusion *with the client*, from the perspective of trying to resolve a specific issue. As a result, we never had to “sell” modeling the product concept. The HandScript team saw the need for themselves.

The product concept model, a simple one-page diagram (that cannot be included in this case study for proprietary reasons) named the parts of the system and described relationships between them. (For example, there are physicians, patients, forms, order sheets, and orders. The physician chooses one patient from a list. The physician starts an order from a blank order sheet. Etc.) Resolution of even the few issues listed in the example on the previous page required that the team develop a clear, shared model of those relationships. I cannot overemphasize the usefulness of the model in clarifying both teams' understanding of the product, the open issues, and the priorities.

In a few cases DDO couldn't quickly reach agreement with the HandScript team on how an issue should be closed. We conducted one round of usability testing and brought those issues to the test. Conducting usability testing gave us a chance to close some sticky disagreements. Most of the HandScript team had never seen a usability study—they were impressed and excited by the lab setting, the coordination of the test and the professionalism of the moderator. And we garnered some (more) credibility based on the very positive responses of the users.

In order to make the middle-out process successful, we conducted a number of "foundation-building" activities:

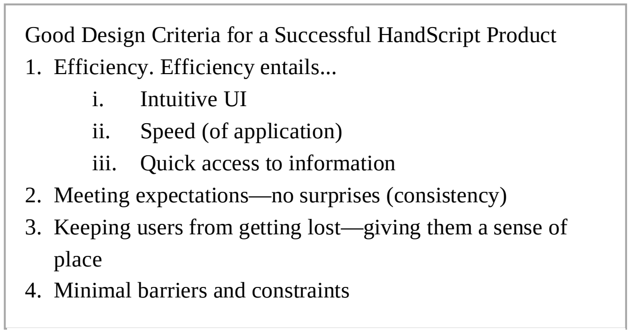
Document "Good Design" Criteria

Our experience is that people use words like "good design" without qualifying what they mean. Theoretically our clients bring us on board because they trust our opinion on what "good design" is, but often we find that someone on the team has different ideas. Working out those differences can be time consuming.

HandScript was eager to work with us; most of the team were happy to have experts helping. Nevertheless, initial general comments about areas that might be improved elicited defensiveness. In particular, the physicians had spent considerable time developing content using a tool that looked very much like the interface of the alpha product, and thus dealing with UI issues. They wondered why the rest of the team felt a design consultancy was needed.

Agreeing on and documenting "good design" in the context of the HandScript product was critical. Few designers haven't had the experience of getting some way into a project before discovering that one team member doesn't agree that consistency is important, for example. Better to get these on the table at the outset. (in fact, we did go through a miniature lesson in why consistency was good in interaction design to get it included in the criteria.) Issues would be closed when good solutions were identified. Having the criteria on paper would help focus discussions and move our work along more quickly. It would also de-personalize any decisions to change the UI. (It was easy for the physicians to take any UI changes personally; after all, they'd spent a lot of time on it. Such discussions may take less time if they can be framed more objectively.)

After several hours of discussion with the client, we reached agreement on a prioritized list of design criteria (see Figure 2). While these were not DDO's ideal criteria or necessarily the way we'd prioritize them, the list did encompass ideas that we'd seen were missing in the alpha. And we understood early what differences existed between what we felt was good design and what HandScript felt was good design.



Good Design Criteria for a Successful HandScript Product

1. Efficiency. Efficiency entails...
   - i. Intuitive UI
   - ii. Speed (of application)
   - iii. Quick access to information
2. Meeting expectations—no surprises (consistency)
3. Keeping users from getting lost—giving them a sense of place
4. Minimal barriers and constraints

Figure 2. The list of HandScript design criteria.

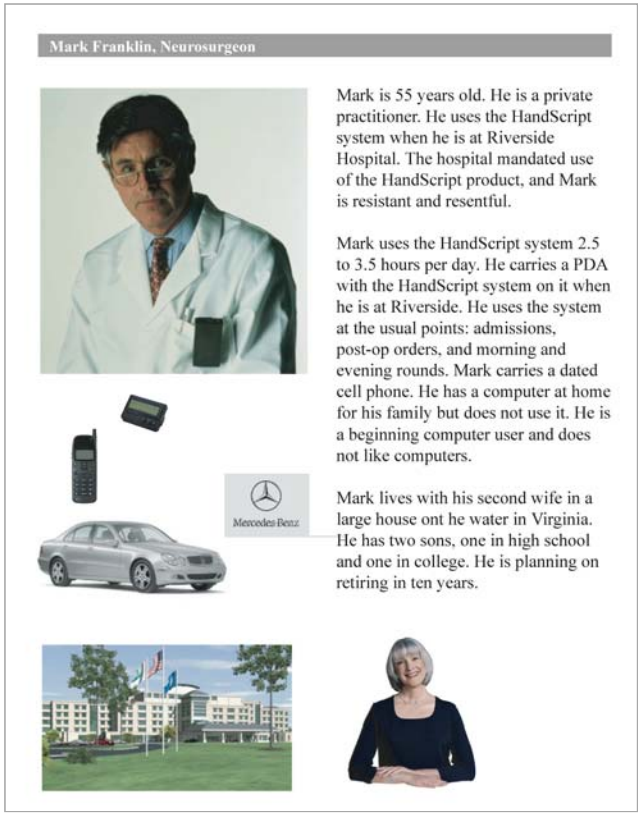
The criteria were posted in our regular meeting room and on the project web site as a constant reminder of our collective goals. Of course later there were discussions about what “intuitive” means or what users’ expectations might be, but obviously we spent less time in these conversations than we might have without any criteria at all. And DDO’s work could be more efficient since we knew the criteria by which the team would be deciding whether an issue could be closed.

Develop Personas

The development of personas was another important step to help de-personalize decisions that changed the UI of the product and hopefully close issues more quickly. (Of course personas serve many other valuable purposes as described by Alan Cooper and Robert Reimann, [1] [2].) DDO created three personas in a brainstorming session with HandScript—there was neither time nor budget to do more research (see Figure 3).

We were gratified that the predictions about the personas came to life in usability testing—HandScripts’ ideas about types of physicians were very accurate. The personas ranged from Jennifer, a young, tech-savvy woman, to Mark, an older physician who hated technology. The HandScript physicians were no longer the only users; we stopped hearing, “I designed this screen like this because I like it when…” in our weekly meetings. The collective team could now talk about our personas Jennifer, Mark, and Steve.

Mark was a particularly useful persona as we continued to stress consistency and giving the users a sense of where they were (and where they’d been, and where they could go next). This was important because HandScript was constantly tempted to resolve questions at a screen-by-screen level in an effort to move quickly. The personas were posted prominently in our conference room and on the project web site along with the “Good Design” Criteria.



Mark Franklin, Neurosurgeon

Mark is 55 years old. He is a private practitioner. He uses the HandScript system when he is at Riverside Hospital. The hospital mandated use of the HandScript product, and Mark is resistant and resentful.

Mark uses the HandScript system 2.5 to 3.5 hours per day. He carries a PDA with the HandScript system on it when he is at Riverside. He uses the system at the usual points: admissions, post-op orders, and morning and evening rounds. Mark carries a dated cell phone. He has a computer at home for his family but does not use it. He is a beginning computer user and does not like computers.

Mark lives with his second wife in a large house ont he water in Virginia. He has two sons, one in high school and one in college. He is planning on retiring in ten years.

Figure 3. A detail from a persona.

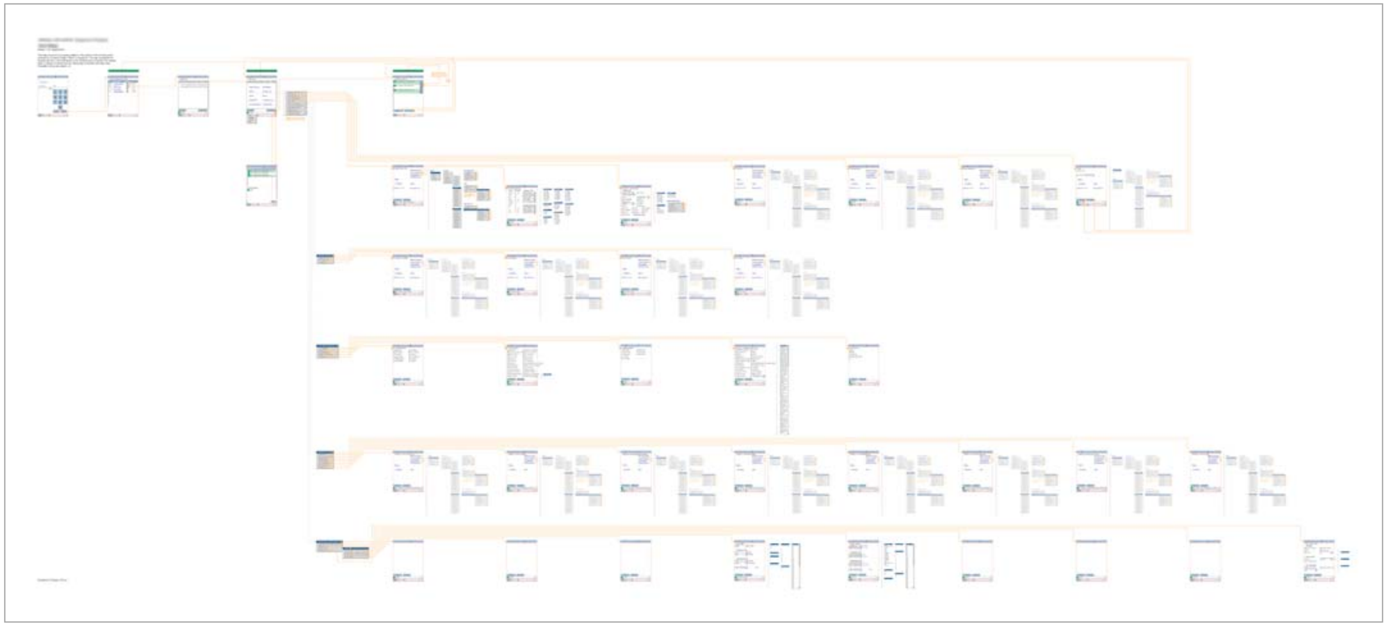
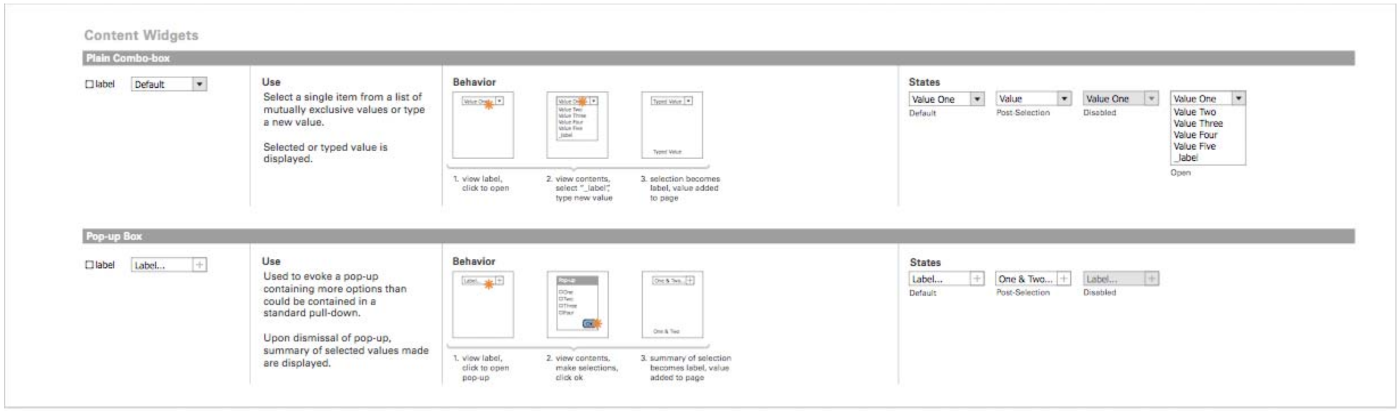


Figure 4. A map of one seventeenth of the alpha product.

Audit One Section of the Application

While the personas and design criteria were being hammered out, we mapped one of the 17 sections of the alpha application (see Figure 4) in order to understand the current state of the product.

The alpha product consisted of seventeen categories of possible orders. Each category was comprised of a navigation tree and the order forms. The categories contained anywhere from 30 to 100 forms; the system as a whole contained roughly 1500 forms. Physicians were continuing to enter content for the product, and engineers were continuing to “port” that content to the alpha product. HandScript did not have the time or budget for a complete audit, nor could we justify mapping the entire alpha product, in spite of all we thought DDO would learn from such an exercise. Instead, we mapped the section of the alpha that was most nearly complete. Our partial map turned out to be exceedingly useful in many ways: It uncovered design bugs to add to our Issues List. Some of the issues were areas that we could immediately improve, thus contributing to our credibility. The engineering team was thrilled to see in one place what a big, complicated thing they had built. DDO became deeply familiar with the product. Finally, the map helped DDO begin to talk with HandScript about the bigger picture—the overall product concept as opposed to the screen-level or even pixel-level issues that they were caught up in.



Content Widgets

Plain Combo-box

□ label

Use
Select a single item from a list of mutually exclusive values or type a new value.

Selected or typed value is displayed.

Behavior
1. view label, click to open
2. view contents, select "_label", type new value
3. selection becomes label, value added to page

States

Pop-up Box

□ label

Use
Used to evoke a pop-up containing more options than could be contained in a standard pull-down.

Upon dismissal of pop-up, summary of selected values made are displayed.

Behavior
1. view label, click to open pop-up
2. view contents, make selections, click ok
3. summary of selection becomes label, value added to page

States

Figure 5. A sample from the baseline Widget Library.

Develop Libraries
DDO used the map to begin developing two libraries: a library of page types and a library of "widgets", functional objects, used within the pages (see Figure 5). Starting these libraries from the map of the alpha product reinforced with HandScript that we were building on the work they had already done.

DDO used the libraries to understand how HandScript had been solving page-level and widget-level design issues to date. Any concerns we had about HandScripts' resolutions were added to the Issues List.

As the design work proceeded, we updated the libraries with any issue resolutions that were relevant to page types or widgets.

The libraries served as a toolkit that could be used as needed to resolve problems with the least number of tools necessary (following the principle of least means, avoiding unnecessary complexity). The rigorousness of the libraries prevented HandScript from acting on the temptation to build unique solutions on a screen-by-screen basis. Finally, DDO updated the libraries as decisions were made, so we had a document that was essentially a continually evolving final deliverable.

C. Results (measured against goals)

In the end, the project budget ran out before all of the issues were resolved. Out of 72 issues, the Dubberly Design Office/HandScript team resolved 48. Another 15 issues were labeled low priority (P5, see Figure 1 for a definition), and did not need to be resolved in advance of the beta.

The HandScript team was left with about 10 open issues to resolve, and none of those were critical (no P1s or P2s).

Both DDO and HandScript felt that the project left HandScript in a good position to complete the remaining work on their own.

- We'd resolved all major issues that were holding up development.
- We had developed a set of core documents, including the design principles, personas, product concept, page and widget libraries that would inform later decisions.
- The libraries comprised a modular toolkit that could be used to solve any number of usability problems, and the rules for extending the toolkit (the principals of consistency and least means) were well established for the team.
- The team had developed a disciplined, methodical approach for tracking and resolving remaining and new usability questions and issues.

HandScript continues to use the documents and process we created to finish the user interface for the beta. We are sometimes frustrated when we can't see a design through to product completion. In some ways, though, what we did for HandScript is more satisfying than completing the design ourselves: we accomplished a knowledge transfer that enabled the clients to resolve their own issues (taught them to fish, as it were…).

Before our engagement began, HandScript struggled with a tangled nest of tough UI issues, and yet some members of the team were skeptical that interaction design was even necessary. After we left, HandScript had a firm foundation for both process and design, with an orderly queue of issues and the tools for tackling them. And we had a good time working on the project.

On future projects, we will consider the medium of the Issues List carefully. The list for this project was maintained by the project manager in a spreadsheet, and then published as HTML to the project web site. This had the disadvantage of being time-consuming to update and post weekly. On the other hand, it had the advantage of keeping the list in one person's hand, ensuring that issues weren't duplicated and that they were documented clearly and briefly. In some cases, it may make more sense to use a web-based bug tracking tool so that all team members can enter issues, run reports of issues assigned to them, and close them when they're resolved. This would be useful if the team included more members capable of resolving issues, if the issues themselves were more diverse (some product management issues, some UI issues, some technical issues, etc.) or if the team was more geographically diverse.

In the final analysis, nearly all of the projects we work on are "Middle-out" design problems—it is unfortunately very rare to have an opportunity to start design during the product concepting stage. Our project with HandScript was so clearly starting in the middle of the software development process that we had the perspective to tackle it in a unique way. It never occurred to us to try to wrestle a "perfect" process into an imperfect situation. We will continue to explore and develop the middle-out design process and its tools, "good design" criteria, design bugs, and the issues list.

A note on the origins of middle-out design:
Middle-out design is not an entirely new idea. Good project managers always track issues. And as mentioned above, software QA is built around the software bug tracking process. In addition, a number of software developers have studied the development process to understand and improve the process of identifying issues, making decisions, and tracking them (an area of research known as design rationale). [3]

The middle-out design process proposed here differs from these other methods. It's more than just tracking issues; it involves the whole team (including engineers) in discussions, decisions, and tracking. It's quite similar to software QA but tracks both design bugs and also issues for which no design solutions have yet been proposed. And finally it differs from most design rationale efforts in that it is a light-weight process, based on widely understood practices (from QA), and relies on readily available software tools. [4] [5]

Nevertheless, middle-out design shares an important trait with design rationale projects. Middle-out design assumes the design process is essentially political and argumentative—the building of an argument (and agreement) about goals and means. In this regard it has roots (as do most design rationale efforts) in the work of Horst Rittel. [6] [7].

References

[1] Cooper, Alan and Reimann, Robert. *About Face 2.0: The Essentials of Interaction Design*. Indianapolis: Wiley Publishing Inc., 2003.

[2] Cooper, Alan. *The Inmates are Running the Asylum: Why High Tech Products Drive Us Crazy and How to Restore the Sanity*. 2nd ed. Indianapolis: Sams Publishing, 2004.

[3] MacLean, A., Young, R. M., and Moran, T. P. Design Rationale: The Argument Behind the Artifact, CHI'89 Proceedings, (Austin, Texas, May 1989), ACM, 247-252.

[4] Shum, S. "A Cognitive Analysis of Design Rationale Representation, (Chapter 1- Design Rationale's Research Roots)." Doctoral Dissertation, Department of Psychology, University of York, 1991.

[5] Regli, W. C., Hu, X., Atwood, M., and Sun, W. *A Survey of Design Rationale System: Approaches, Representation, Capture and Retrieval, Engineering with Computers* (2000) 16: 209-235, Springer-Verlag, London, 2000.

[6] Rittel, H. W. J., and Webber, M. M. "Dilemmas in a General Theory of Planning." *Policy Sciences, 4* (1973), 155-169.

[7] Rittel, H. W. J. "Issues as Elements of Information Systems, Working Paper No. 131." Berkeley, CA: Institute of Urban and Regional Development, University of California, 1970.

Acknowledgements

Greg Baker's extraordinarily clear thinking and imaginative problem solving provided some of the best interaction design work we've done to date.

The rest of the staff at Dubberly Design Office, Robin Bahr, Hugh Dubberly, Ryan Reposar, and Paul Souza, patiently helped us out whenever they were needed, whether it was concept development, design decisions, production, or reviewing this paper.

Our clients at HandScript gave us an invaluable opportunity to work on a fascinating problem in a challenging environment. They are extremely dedicated to their work, and were open to new solutions to difficult problems.

Richard Anderson's contributions during usability testing were graceful, articulate, and enormously helpful.