How Designers Design and Program Interactive Behaviors

Brad Myers, Sun Young Park*, Yoko Nakano, Greg Mueller, Andrew Ko
*Human Computer Interaction Institute; *School of Design*
Carnegie Mellon University
bam@cs.cmu.edu; sunyoun1@andrew.cmu.edu; yokonakano@alumni.cmu.edu; gmueller@andrew.cmu.edu; ajko@cs.cmu.edu

Abstract

Designers are skilled at sketching and prototyping the look of interfaces, but to explore various behaviors (what the interface does in response to input) typically requires programming using Javascript, ActionScript for Flash, or other languages. In our survey of 259 designers, 86% reported that the behavior is more difficult to prototype than the appearance. Often (78% of the time), designing the behavior requires collaborating with developers, but 76% of designers reported that communicating the behavior to developers was more difficult than the appearance. Other results include that annotations such as arrows and paragraphs of text are used on top of sketches and storyboards to explain behaviors, and designers want to explore multiple versions of behaviors, but today's tools make this difficult. The results provide new ideas for future tools.

1. Introduction

Designing user interfaces differs from designing static pages and movies in that user interfaces involve *interactivity*. Users click on buttons and links, fill in fields, directly manipulate graphical objects, and thereby control the results in a variety of ways. There have been many previous studies of the processes, techniques and tools that are used by designers, but none have focused on how the interactive behavior of the interface is created and communicated. Creating the behaviors inherently deals with programming concepts, such as conditionality (e.g., behaviors that only happen *if* the user interface is in a certain mode), and abstraction (e.g., a behavior should happen when users click on any object of a particular type). Designers can be classified as *end-user programmers* (EUP), since they do not write programs as a primary goal, but rather in support of designing a web page or an animation [12, 13].

The tools that have been made for designers either focus on the static design (such as Photoshop or Illustrator), provide only limited interactivity such as rollovers and pages changes for web pages (such as in Dreamweaver), or are based on conventional languages and tools aimed at professional programmers (for example, the ActionScript language in Flash is basically Javascript). Our intuition was that none of these approaches was meeting the needs of designers.

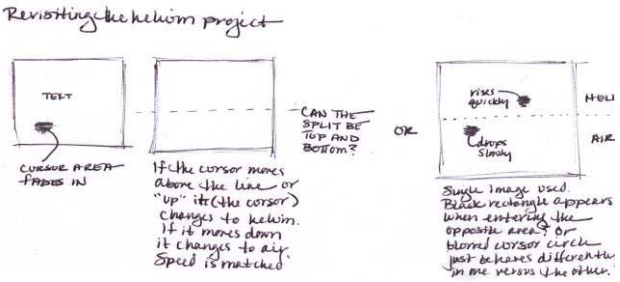

Figure 1: Sketches by a contextual inquiry participant showing two different options being investigated for an interaction, with lines and textual annotations to explain what is intended.

To evaluate our intuition, we conducted field studies of 13 designers, followed by a web-based survey, which received 259 responses, to investigate the design and programming of interactive behaviors. We particularly focused on people who are trained and work on interaction design, graphics design, information architecture, experience design, visual design, user interface design, or equivalent.

In our survey, we defined a behavior as:

> "...anything that an application does. If you think of "look and feel," we are mainly focusing on the "feel." Another way to look at it is that we are focusing on what you cannot draw in Photoshop, but instead need to describe in other ways, such as using storyboards, code, etc. Or, if you were developing in Flash or Director, the behavior would be anything that required using the timeline or scripting."

Since these behaviors happen in response to user inputs in the target applications, we classify them as *interactive behaviors*. In this paper, we use *behaviors* and *interactive behaviors* interchangeably.

Many of our findings confirmed what others have reported in previous surveys (e.g., [4, 6, 14, 18]), for example that designers prefer to start by sketching (about 97% in our survey), and most designers (88%) also use storyboards. However, we did find five interesting new results that have not been previously reported:

1-4244-2528-0/08/$25.00 ©2008 IEEE

1. By a large margin, the participants in our survey agreed that programming the behaviors is more difficult than designing the appearance.
2. The behaviors that the designers wanted were quite complex and diverse, beyond what could plausibly be provided by a system that provided only a few built-in behaviors or a selection of predefined widgets, and therefore seemingly requires full programming capabilities.
3. Sketches and storyboards cannot adequately convey the behaviors by themselves, so designers must augment them with annotations such as arrows and many textual descriptions of the desired behaviors (see Figure 1).
4. The purpose of implementing the interactive behaviors, and for annotating pictures, is often primarily to serve as documentation and specifications for others. Almost all designers worked in teams, and communicating with others is a key part of their jobs. Communicating the design of behaviors to developers was reported to be more difficult than the appearance by 76% of the designers in our study.
5. As reported for other kinds of design [4, 9, 17], the designers in our survey agreed that the design of interactive behaviors emerge through the process of exploration. In other words, designers do not have a final conception of the behavior before they start. However, whereas iterating on the look of the interface can be easily done by sketching, designers felt it difficult to iterate on the behavior. Today's authoring tools make it difficult or impossible to compare two implementations of behaviors side-by-side, and even keeping around and reverting to old versions of code is difficult.

2. Related Work

It is well known that designers prefer sketching for early phases of design [4, 14, 18]. Bill Buxton has devoted an entire book to this subject [4], which makes clear that sketching is an important technique for determining what should be designed, as well as for determining what the design should be. However, that book says little about determining the *behavior* of the designs, devoting only one page to interaction design. It gives some requirements for sketching behaviors, without any hint of how to achieve them [4, p. 136].

A recent survey of 370 practitioners, both designers and developers, reported that "evolutionary prototyping" was the most common development process [5]. Tools used included paper and pencil, whiteboards, html editing, analysis and modeling tools, visual interface builders, etc. Other studies have looked at designers for particular domains, such as web authoring and animation. A study of 11 designers showed extensive use of informal tools in the early design phases of web sites [14]. None of these designers were involved in the programming of the final version of the web sites. To collaborate with developers, they used site maps, storyboards, annotations on top of sketches, and detailed Photoshop renderings. This study did not differentiate difficulties that designers had with the appearance from the behaviors.

A later study of 334 web developers who did not have professional training in programming showed that the collaborative group in which the developers worked was a greater contributor to their successes and failures than the tools they used [15]. Virtually all of the participants taught themselves at least some of the programming skills they needed, and they often wanted to use behaviors that they could not successfully implement.

Other surveys have looked at designers of animation and multimedia. One study interviewed 7 animators and 8 non-animators [6]. About half of the animators used paper sketching and storyboards before switching to digital tools. Another study interviewed 12 and surveyed 13 professional multimedia designers [2], and showed that a script was often the first thing created for multimedia, which is not an artifact mentioned for creating other types of design. The participants rated scripts, storyboards, sketching, and prototypes as important, but inadequate to relate important aspects of the dynamic aspects such as interactivity or timing.

Many people have created tools to help designers with sketching and authoring behaviors, often informed by the studies mentioned above. For example, tools for early design focused on use by designers who are not programmers include SILK [11], Electronic Cocktail Napkin [8], DENIM [14], SketchWizard [7], DEMAIS [2], Designer's Outpost [10], and many others. However, these tools mainly provide a limited fixed set of behaviors and are focused on particular domains. The upcoming Adobe Thermo product provides a menu of 19 behaviors that can be applied to graphics [1].

3. Method

Our first study used the Contextual Inquiry (CI) methodology [3] with 13 participants. CIs involve observing and interviewing participants while they are in the process of working in their natural work environments to understand their practices and problems. For this study, we employed retrospective and artifact walkthroughs to investigate transitory work processes that would have been too time-consuming and impractical to study using the traditional CI method. We asked participants to "walk through" their recent projects involving interactivity using the resulting artifacts.

Each CI took about 90 minutes, and participants were not compensated.

We then assessed the generalizabity of the CI data using a widely distributed survey. We sent requests to fill out the survey to our alumni mailing lists, to CHI-Announcements, CHI-RESOURCES, and the Interaction Design Association (IxDA) mailing list. We received 259 responses, of which 203 (78%) completed all 47 questions. The complete survey took about 30 minutes to finish. The survey participants were entered into a raffle for five $25 gift certificates. To encourage respondents to answer the essay questions, we doubled the odds for those who filled them out completely, which worked pretty well – about 40% of the participants wrote comments in all of the fields.

4. Results

Across both studies, there seemed to be a high interest in this topic, and participants seemed very willing to help. Many said they were hopeful that there would eventually be interesting new tools as a result. After presenting the demographics and tool use of the participants, we then describe our five most interesting findings.

4.1. Demographics and Tool Use

The participants in both studies had a wide variety of backgrounds and degrees. The CI participants were university faculty, a Flash instructor, master's students, and professional designers. Of the professionals, most worked as consultants, but one was an in-house designer for a small company.

Most participants in the survey (about 45%) had a Masters degree, with 31% having a bachelor's as their highest degree. The area of the degree varied widely, and included psychology, human factors, anthropology, English, computer science, engineering, HCI, and many others. A few (17%) had a CS degree, although 63% of them also had one or more degrees in the other areas mentioned. Regarding work related experience, 9% had more than 20 years, 26% had 11-12 years, 36% had 6-10 years, 27% had 1-5 years, and 2% had less than one year.

Many worked in a design department of a company that consulted on various products (about 25%) or worked directly in a product division (24%). About 21% worked for a design consultancy (a company that mostly consults with other companies), and about 13% worked alone. Many of the rest worked in universities and did design consultancy on the side. 60% worked for companies bigger than 100 people. About 55% spent most their time on websites, 9% on desktop applications, and 1% on phones.

Their job titles at work included Interaction Designer (32%), User Interface Designer (20%), Information Architect (12%), and many others. About 6% were managers of groups that included designers.

In terms of their tool use and expertise, we saw that designers use a wide variety of tools, depending on the stage of the design process and their skills. For ideation sketches, all of our CI participants used paper and whiteboards. In the more collaborative work settings, ease of use and ability to combine parts created by different people determined which tool to use. In one company, PowerPoint was used since everyone could share the results, although the designer complained that updating changes was a hassle, since they are not reflected in all of the files. In our survey, we saw a wide variety of tools mentioned. Figure 2 shows the tools participants use often.

Because the CI participants mentioned annoyance with differences in the user interfaces among all the tools they use, we asked about this on the survey. For the statement, "I find the differences in how different tools work to be frustrating," 18% of the participants in the survey strongly agreed, 42% agreed (totaling 60%), with only 6% strongly disagreeing and 14% disagreeing (totaling 20%), with the rest (20%) neutral.

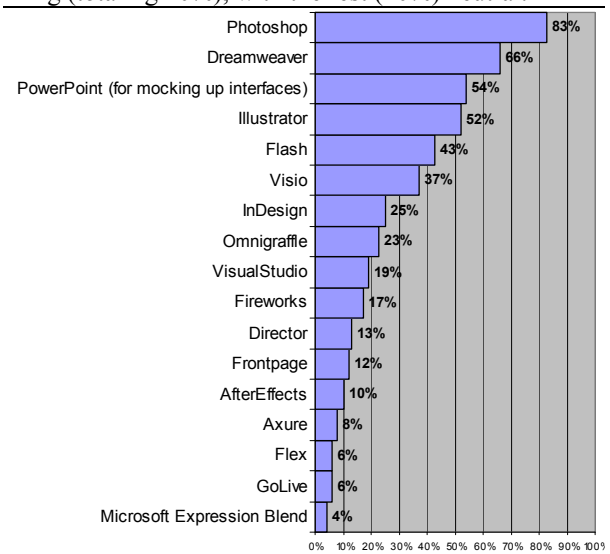

Figure 2: Percent of participants in survey that use each tool at least often (N=259)

4.2. Programming Behaviors is Difficult

Our CIs suggested that designers have more difficulty programming interactive behaviors (feel) compared to the appearance (look). Designing the behaviors was reported to be an ongoing process while designing appearance is a simpler process of creating a single static image. Behaviors were said to often re-

main ill-defined until the application is actually implemented.

These results were supported by the survey, where 86% of the participants listed behavior as more difficult to prototype than appearance (see Figure 3). 91% of participants with CS degrees answered that the "Behavior" was harder, compared to 84% of participants without a CS degree.

We asked participants how they divided their time between the behavior and the appearance, and the results ranged from 0% to 100% for each. Participants reported spending significantly more time on behavior ($t=6.8$, $p < .0001$). The two distributions for appearance and behavior were normally distributed. The mean percent of time (with the standard deviation) spent on the appearance was 39% (± 21), compared to 61% (± 21) for behavior.

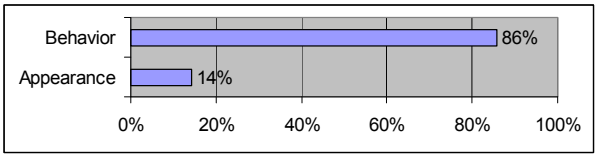

Figure 3: "Regarding Appearance versus Behavior, which is more difficult to prototype?"

To explore the source of difficulties with behaviors, we first wanted to understand how the participants were programming. When asked on the survey, "Is programming (also called 'scripting') a regular part of your job?" 63% answered "no." Interestingly, of the people who had a CS degree, 51% still answered "no," and for people without a CS degree, 65% answered "no." However, about 79% said they were good or expert with HTML, and 95% knew some other programming language at least to the level that they "can do a few things." Many programming languages and tools were mentioned, including (in decreasing order of popularity) Javascript, PHP, ActionScript for Flash, VisualBasic, Java, C++, Lingo for Director, .Net, C#, Ruby on Rails, CSS, "Processing" (from www.processing.org), Perl, Cocoa & Objective C, Python, XML, ASP, MAX/MSP, SQL, and a few others.

When participants in the survey had trouble understanding how to implement something, they usually "use Google or another search engine to search for examples" (91% do this "at least sometimes"). All of the other options we listed were also popular: "I go to online tutorials or on-line documentation" (90%), "I look for examples in code that I have around" (89%), "I ask a colleague how to do it" (81%), and "I go to the manual or books and look it up" (80%).

Participants gave a variety of reason for why behaviors were more difficult to prototype. Here are some categories of problems and supporting quotes:

- Interactions must be specified at a low-level of detail:
  - "Details are important, and you never have them all until full implementation."
  - "There are so many factors that can influence the behavior. It isn't a controlled event and [it's] very difficult to communicate the entire experience via a prototype.
  - "Because of the many different states that must be demonstrated and their dependence on different conditions preceding those states."
  - "It's more complex with more variables and constraints attached. There are also more stakeholders and coworkers involved"
  - "It's harder for people to fill in the gaps with imagination. Small changes make big differences in experiential outcomes so if something is not quite right it can cloud whole thing."
  - "There's no such thing as low-fidelity interaction, it has to be right."
- Difficulty with today's tools:
  - "I work in mobile, so often I have to wait until developers have something working before I can actually test out how a new behavior 'feels' on a phone. What it looks like is easy to test by just putting a picture of the mockup onto the phone."
  - "Current tools for defining behavior suck."
  - "I can represent very exactly the desired appearance. However, I can only approximate the backend behaviors."
- The richness of the interactions themselves (see next section).

4.3. Desired Behaviors are Complex

A goal of some prior research systems and commercial tools has been to make it easier to investigate interactive behaviors (e.g., [1, 2, 7, 11, 14]), usually by simplifying the kinds of behaviors that can be expressed. This is in conflict with the wide variety of behaviors we saw in our investigation. For example, in our CIs, participants were creating interfaces that use 3D rotation, physical simulations of dominos falling onto each other, interactions among graphical objects on the screen such as bouncing off each other, graphics that changed based on various sensors, novel physical devices, etc.

In the survey, we asked for ideas of what a future tool should support, and participants requested many things, including "character animation (using jointed inverse kinematics)," support for "more dynamic content, like AJAX," "database or CMS integration; for example, showing different content to different users based on business rules," "data-driven interactions," "ability to create RIA functionality" (Rich Internet

Applications – web applications that act like traditional desktop applications), and "ease of ... connect[ing] to live data sources."

55% of the participants said that they had had an interactive behavior they wanted to explore on paper but could not, and therefore had to explore by implementing it. We asked them to describe what they wanted to do, and participants listed 107 behaviors. Here are some representative quotes:

- "Rich interactions, such as a sliding dock or an object that can be dragged and dropped. These are hard to illustrate in static documents."
- "Interactive maps"
- "Camera interaction."
- "Complex transitions / animations."
- "Synchronised behaviours."
- "Issues around scrolling long pages."
- "A button that changes color to draw attention to itself."
- "Window/panel transparency within the OS and various applications."
- "Dynamic navigation systems (e.g. accordion menus), 3D or graphical navigational systems."
- "Mobile interaction."
- "More advanced interactions like drag-and-drop, column sorting, resizing on rollover, etc."
- "Hover effect on a graphic, that when clicked, also selected a visible tab."
- "Detailed keyboard-level interactions."
- "Interactive art - engaging experiences."
- "User experience of interaction with graphs."
- "The exact timing of certain interactions."
- "An animated 'lens effect' list UI."
- "Gaming behaviour."
- "Multi-dimensional selections that impact the display of other controls and data."
- "Dynamic layout based on user preferences."

4.4. Annotating Sketches and Storyboards

Consistent with previous studies (e.g., [14]), we found that virtually all of our participants used sketches and storyboards as part of their work (see Figure 4). However, participants also agreed that sketches and storyboards were not sufficient for exploring interactive behaviors, consistent with previous studies (e.g., [2, 4]). We observed during our CIs that designers extensively use annotations on sketches, storyboards, wireframes, and formal design documents to describe the behaviors (see Figure 1 and Figure 6). The kinds of annotations we observed included labels, arrows, and narrative textual descriptions.

Our survey confirmed that annotations are important to designers, used by 97% at least sometimes (see Figure 5); 88% used storyboards. In the comments on this question, one participant emphasized: "[I] SCAN my drawings to PDF and combine with design notes: I NEED to TYPE and SKETCH on the same paper." Another said: "Word is probably our most important tool because the text support allows us to easily describe UI behavior in great detail and everyone can open the file and edit it if needed... However, formatting in Word is a huge pain."

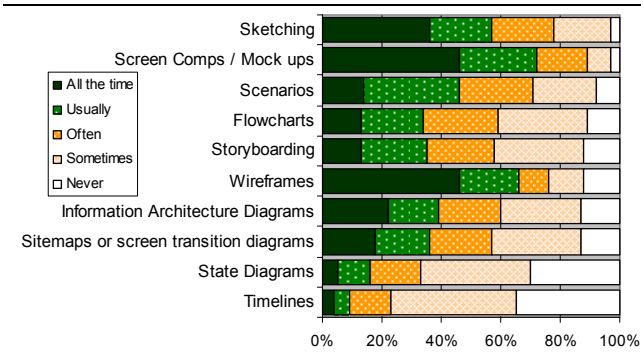

Figure 4: How often participants in the survey typically used each of these techniques when working on a project (N=210).

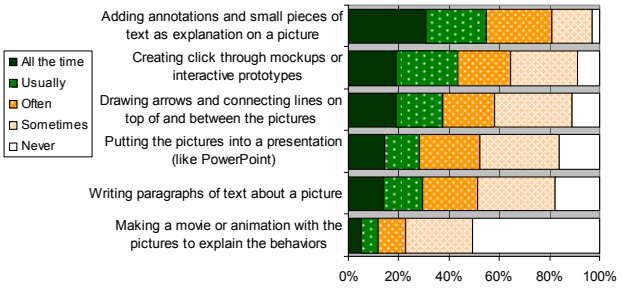

Figure 5: How often participants in the survey used each of these techniques to help explain how a digital drawing behaves (N=210).

4.5. Purpose is Communication

Our CIs suggested that a primary job of the designer is to collaborate with a developer and communicate their designs. Most participants preferred to work face-to-face, but often their final deliverable would be a detailed design document to hand off to the developer. These documents typically use many paragraphs of text and description of details around the pictures to explain the behaviors (see, for example, Figure 6). After delivering the documents, the designers would assume more of a support role, clarifying designs to developers as they implemented them. The CI participants emphasized that they considered the detailed design document to be an important deliverable, and they spent significant time making sure it was clear and looked polished. The behaviors were reported to be more difficult to

communicate because people can imagine the final detailed appearance even from a vague sketch; however, it is very hard to communicate behavior and to get a sense of the final version from a prototype.

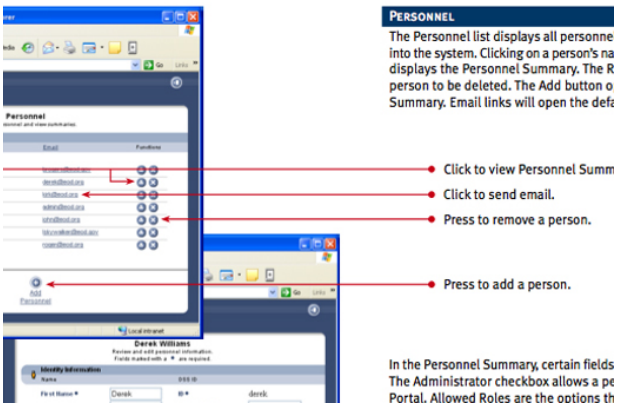

Figure 6: Annotations from a CI participant on a detailed design to serve as a specification for developers.

These findings were confirmed by the survey. On the participant's "current or most recent project," 91% said they were working as part of a team, which usually (78% of the time) included a developer. The designers usually (37%) met with developers "multiple times throughout the week", "work side-by-side" (18%), or "meet multiple times per day" (13%).

In the survey, we investigated what techniques the participants used to communicate with developers. Although sketching is a key tool for designers, apparently it is *not* used as a communication tool: 56% of the participants *never* "draw sketches on paper then hand them off to the developer to build," with only 8% saying they did this "usually" or "all the time." In contrast, the most popular techniques used for communication were to "write textual descriptions of how the application will work" (78% do this at least "often"), followed by "static designs digitally" (66%), "collaborating around a whiteboard or sketchpad" (56%), and "semi-functional, interactive prototypes (like in Flash)" (33%). So, designers seem to sketch and annotate the sketches for themselves (as in Figure 1), but use annotations on more formal drawings (as in Figure 6) to communicate.

76% of the designers felt that the behavior was more difficult to *communicate* to the developer than the appearance. Two of the reasons that participants mentioned for this problem were that developers not were not native English speakers, and the lack of details in the UI specifications which seem to be solved by face-to-face, phone, IM, and email conversations. One interesting comment was that "we publish our spec as a wiki – thereby allowing developers to add their annotations, questions and comments which we can in turn address. This makes deliverables more of 'living documents' than static paper weights."

Designers emphasized that they often were able to design the final appearance, but the developers were in charge of the code for the behaviors. Some participants reported that it would be important for a new tool to make sharing easier and be able to export designs into formats that can be easily viewed and commented on by others.

4.6. Iteration and Exploration for Behaviors

It is well known that designers sketch multiple variations when exploring designs (see Figure 1). Buxton quotes Linus Pauling saying that "the best way to a good idea is to have lots of ideas" [4, p. 121]. Other studies have also documented this [14]. We were interested in how multiple versions were used for exploring interactive behaviors. In the CIs, one participant noted that "most ideas are bad and you want to get the bad ones out quickly" by iterating. Designers also frequently mentioned that because of the difficulties involved in communicating behavior to developers, versions of the behavior are much harder to iterate on. In contrast, variations in appearance are easy to iterate on and can even be changed at the very end of the design process.

Participants mentioned that interactive behaviors had to be explored throughout the design process, including during the early stages when navigational structures being worked out because interactivity is often involved in changing among states. However, the subtle details of interactive elements are tweaked until the end, even after the final graphics are implemented.

As shown in Figure 1, we saw in our CIs that designers frequently wanted to have multiple designs side-by-side, either in their sketchbooks, on big displays, or on the wall. However, this is difficult to achieve for behaviors – there is no built-in way in today's implementation tools to have two versions of a behavior operating side-by-side. Therefore we asked in the survey, "How important is it to you to be able to compare alternatives to decide which one to use?" Figure 7 shows the responses. The behavior-related items (the last three) are at the bottom of the list, possibly reflecting the participant's lack of ability to do this now.

Programmers rarely start a new project from scratch, but instead reuse old code as a starting point. We wondered if this applied to designers as well. In the CIs, we saw one participant having a library of templates in tools such as Photoshop and Illustrator to save time creating standard widgets.

In the survey, we asked how often design elements are reused for a later project. Figure 8 shows the results. As expected, sketches are almost never reused –

only 5% said "often," 4% said "usually," and 1% said "all the time." In contrast, widgets and controls are reused frequently. Code and scripts are reused about half of the time.

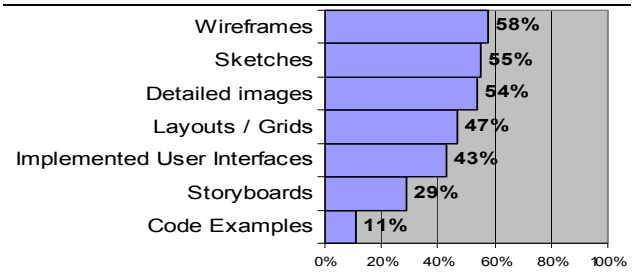

Figure 7: Percent of participants in survey who said it was "very important" or "crucial" to be able to compare alternatives to decide which one to use (N=210)

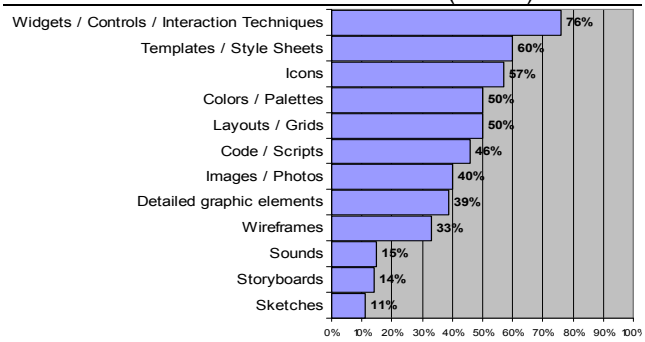

Figure 8: How often various kinds of design elements are reused in a later project (N=210)

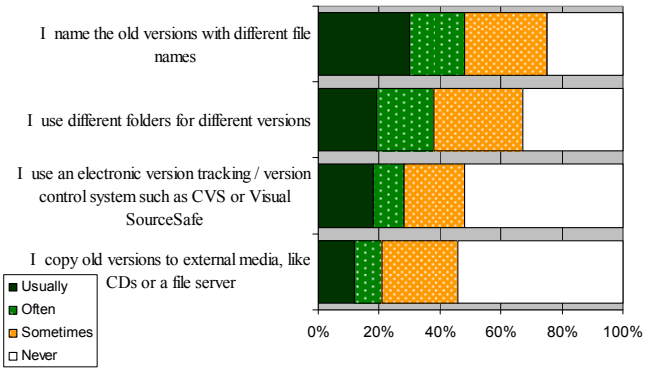

Figure 9: How often participants use these techniques to keep track of previous versions of the code (N=167).

We saw in the CIs that users did not have any good way of saving and reusing different versions of their code, either to enable comparisons in the current project, or to save for future projects. They had to use various naming schemes and file structures of their own invention. This was evident when a participant had a hard time locating the file he wanted to show the researchers during a CI.

Therefore, we wanted to see in the survey what participants did to keep track of previous versions of the code for a project, as shown in Figure 9. This shows that at least 75% of designers do some kind of version control on their own, and since few (28%) frequently use formal version-control software, they have to do this by hand.

5. Discussion

Our results suggest that there is definitely a need among designers for quickly and easily prototyping interactive behaviors. Many do not program, and have difficulty prototyping behaviors. Consequently, they resort to heavy use of annotations and other means for communicating their intentions to people who do program. Specifying behaviors textually is often inadequate, time-consuming and sometimes leads to miscommunication and confusion between designers and developers. Since designers strive to be creative and explore new and complex behaviors, this is a significant pain point for them both today and in the future.

5.1. Threats to Validity

There are a number of reasons our results may not generalize. In our CIs and survey, we tried to focus on interaction designers, but we cannot be sure who filled out our survey, and whether they are representative of the entire field of interaction design. Our means of reaching designers may have biased the results towards the views taught at our university (since we targeted our alumni), or who are members of a particular organization (ACM SIGCHI) and mailing list (IxDA). However, since our results match those previously reported, and since we seemed to have quite a wide variety of degrees and jobs represented, we feel comfortable relying on the results. Another limitation is that the discussion about communication and annotations should include the perspective of what the *receivers* want from designers, so it might be wise to also survey developers. It is also important to note that many developers do not collaborate with designers.

5.2. Implications for Future Tools

All but one person out of the 272 we interviewed or surveyed was interested in a new tool. Our results provide clear requirements for such tools, which go beyond what any of today's commercial or research tools offer. For example, designers are very comfortable sketching and using digital tools for the look of the interface, but still are severely lacking in appropriate tools for exploring behaviors. They would like to iterate and explore multiple behaviors themselves, but are inhibited by today's tools. However, just providing a few behaviors in menus for the designers to pick from is not sufficient, since designers want to explore quite complex behaviors.

Future tools should also support exploring multiple versions of the behaviors. Today's tools make the user be responsible for naming multiple versions of files, and often do not even let designers compare multiple versions side-by-side. Terry provides an example of how this might be done [16].

There are also significant opportunities for supporting tasks for behaviors beyond programming them. No tool today makes it easy to annotate and describe the behaviors, which is important for facilitating communication about the behaviors with developers. And since many designers work in teams with other designers, tools to facilitate collaboration around the exploration might be helpful. Many designers expressed the need for their coding tools to better integrate with drawing tools, like with Visio and Photoshop, to avoid redundant work.

6. Conclusions

Our contextual inquiries and surveys have confirmed what others have reported, but also revealed new requirements and ideas. Although time-consuming, these kinds of user research are very important to do before embarking on new tool efforts to ensure that the results will actually be helpful for the target audience. We hope the information presented here will enable us and others to produce such tools.

7. Acknowledgments

We thank Jodi Forlizzi, John Zimmerman, Ilpo Koskinen, Jeff Wong, Thomas LaToza, Chris Scaffidi, Jeff Stylos, and all the anonymous participants and reviewers for help with this work. Thanks to Adobe for financial support and advice for this research. This research was supported by Adobe and by the NSF under Grant No. IIS-0757511. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation.

8. References

[1] Adobe Systems Incorporated, "Product Codename: 'Thermo'," 2008. http://labs.adobe.com/wiki/index.php/Thermo.

[2] Bailey, B.P., Konstan, J.A., and Carlis, J.V. "Supporting Multimedia Designers: Towards more Effective Design Tools," in *8th International Conference on Multimedia Modeling*. 2001. pp. 267-286.

[3] Beyer, H. and Holtzblatt, K., *Contextual Design: Defining Custom-Centered Systems*. 1998, San Francisco, CA: Morgan Kaufmann Publishers, Inc.

[4] Buxton, B., *Sketching User Experiences: Getting the Design Right and the Right Design*. 2007, San Francisco, CA: Morgan Kaufmann.

[5] Campos, P. and Nunes, N.J., "Practitioner Tools and Workstyles for User-Interface Design." *IEEE Software*, January/February, 2007. **24**(1): pp. 73-80.

[6] Davis, R.C. and Landay, J.A. "Informal Animation Sketching: Requirements and Design," in *AAAI 2004 Fall Symposium on Making Pen-Based Interaction Intelligent and Natural*. October 21-24, 2004. pp. 42-48.

[7] Davis, R.C., Saponas, T.S., Shilman, M., and Landay, J.A. "SketchWizard: Wizard of Oz Prototyping of Pen-based User Interfaces," in *Symposium on User Interface Software and Technology: UIST 2007*. October 7-10, 2007. Newport, RI: pp. 119 - 128.

[8] Gross, M.D. and Do, E.Y.L. "Demonstrating the electronic cocktail napkin: a paper-like interface for early design," in *Conference companion for CHI'96: Human factors in computing systems*. 1996. Vancouver, British Columbia, Canada: pp. 5-6.

[9] Henderson, K., *On Line and On Paper: Visual Representations, Visual Culture, and Computer Graphics in Design Engineering*. 1999, Cambridge, MA: MIT Press.

[10] Klemmer, S.R., Thomsen, M., Phelps-Goodman, E., Lee, R., and Landay, J.A. "Where do web sites come from? capturing and interacting with design history," in *Proceedings of CHI'2002: the SIGCHI conference on Human factors in computing systems*. 2002. Minneapolis, Minnesota: pp. 1-8.

[11] Landay, J. and Myers, B., "Sketching Interfaces: Toward More Human Interface Design." *IEEE Computer*, March, 2001. **34**(3): pp. 56-64.

[12] Myers, B.A., Ko, A.J., and Burnett, M.M. "Invited Research Overview: End-User Programming," in *Extended Abstracts, CHI'2006*. April 22-27, 2006. Montreal, Canada: pp. 75-80.

[13] Myers, B.A., Smith, D.C., and Horn, B. "Report of the `End-User Programming' Working Group," in *Languages for Developing User Interfaces*. 1992. Boston, MA: Jones and Bartlett. pp. 343-366.

[14] Newman, M.W. and Landay, J.A. "Sitemaps, Storyboards, and Specifications: A Sketch of Web Site Design Practice," in *Designing Interactive Systems, DIS 2000*. August, 2000. New York City: pp. 263-274.

[15] Rosson, M.B., Ballin, J., and Rode, J. "Who, What, and How: A Survey of Informal and Professional Web Developers," in *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05)*. 2005. pp. 199 - 206.

[16] Terry, M. and Mynatt, E. "Recognizing creative needs in user interface design," in *C&C'02: Proceedings of the ACM Conference on Creativity and Cognition*. 2002. Loughborough, United Kingdom: pp. 38-44.

[17] Tversky, B., Suwa, M., Agrawala, M., H. J, C.S., Hanrahan, P., Phan, D., Klingner, J., Daniel, M., Lee, P., and Haymaker, J., "Human behavior in design: Individuals, teams, tools," in *Sketches for Design and Design of Sketches*, 2003, Springer. pp. 79–86.

[18] Wong, Y.Y. "Rough and Ready Prototypes: Lessons from Graphic Design," in *Extended Abstracts, SIGCHI'92*. May, 1992. Monterey, CA: pp. 685.