Usability Maturity: A Case Study in Planning and Designing an Enterprise Application Suite

Jeremy Ashley and Kristin Desmond

500 Oracle Parkway, MS 2op10,
Redwood Shores, California 94065
Jeremy.ashley@oracle.com, Kristin.desmond@oracle.com

Abstract. Although user experience professionals look to the user-centered design process (UCD) as the overarching set of principles for the research, design, and testing of usable products that meet customer needs, the application of these principles varies significantly depending on the type and scale of design challenges to be solved and the level of usability maturity that a company practices. This paper describes a case study of how one organization went from a Usability Maturity Model level of *Implemented* to a level of *Integrated* while it worked through a design cycle for a large enterprise application suite. This paper also discusses lessons learned along the way.

1 The Usability Maturity Model

The Usability Maturity Model, as synthesized by Jonathan Earthy (1998), describes six levels of capability in human-centered processes at which an organization can exist. The concept of the scale is derived from earlier scales created for quality and software development. The six levels are briefly described here:

- *Unrecognized* or "ignorance": Usability is not discussed as an issue.
- *Recognized* or "uncertainty": The organization sees that it has problems with usability but does not know what to do about these problems.
- *Considered* or "awakening": The organization understands that UCD processes are necessary to improve usability and begins to implement some of these processes.
- *Implemented* or "enlightenment": The full complement of UCD processes are used and deliver good results on a per product or module basis.
- *Integrated* or "wisdom": UCD processes are included in the development lifecycle methodology, results are tracked, and performance goals are met.
- *Institutionalized* or "certainty": The organization is human centered, in both its internal function and its product design.

For this paper, we have informally assessed the levels in use in our organization; however, the complete methodology as presented by Earthy includes a more formal recording form and scale.

M. Kurosu (Ed.): Human Centered Design, HCII 2009, LNCS 5619, pp. 579–584, 2009.

2 The Problem

UCD professionals, while highly trained in their individual subdisciplines, are often unprepared for the challenge of fully delivering UCD to a finished product. Occupied as they are with user research, design, prototyping, and usability testing, UCD professionals are often shocked when some or all of their work is not applied. This disconnect occurs when UCD professionals do not take into account the readiness of the organization as a whole for their work and do not undertake the non-UCD processes that ensure success.

3 A Case Study

About four years ago, Oracle started a new development cycle to design and build a next-generation integrated enterprise application suite. At the time, the company had just acquired PeopleSoft/JD Edwards and would continue to acquire significant companies during the cycle. The newly appointed applications development VP had previous positive experiences working with the UCD process and had a growing awareness that the outputs of UCD would make strategic differentiators in the suite. He decided to build a centralized team, gathering all of the enterprise applications UCD staff into one organization that reported directly to him, with dotted-line responsibility into his subteams, which were divided according to product functionality. As further evidence of his commitment to the team's success, he also granted the organization a generous number of new positions.

3.1 Initial Analysis

At the start of this process, the UCD team analyzed the historical effectiveness of its various working styles and approaches and noted areas where the organization needed to improve. While there were differences in degree, in general, the newly joined teams had all achieved a level of *Implemented* in their practices, according to the Usability Maturity Model. They had dedicated teams of UCD professionals who performed a more or less complete cycle in conjunction with individual product teams. In addition, all the teams had company-wide user interface (UI) guidelines and standards to which all products were expected to adhere. Some had managed to institute UI code reviews to ensure compliance to these standards. However, these processes were applied inconsistently across products. Some development teams were still saying that they didn't have time to incorporate all of the UCD team's recommendations. Users were still complaining that products were too technology driven and that the software was hard to learn and use. There were also numerous places in the products where the UCD teams had identified missing functionality that if implemented would greatly improve user productivity, efficiency, and satisfaction.

While the organization did not consciously use the Usability Maturity Model levels in its thinking at that time, it is clear in hindsight that the push to function fully at the next level (*Integrated*) would be a key determinant in the success of the overall effort. At the Integrated level, the following things occur:

- UCD processes are integrated with other processes in the development cycle, and UCD requirements are communicated in a way that developers understand and can implement.
- UCD processes are used to inform changes to the developing products in a timely manner, and these changes are implemented.
- Design and design solutions are treated iteratively, and goals for scope and quality are clearly stated and adhered to.

In the beginning, the technique that our organization adopted was to push harder on the UCD front. We knew that the teams needed more UCD to improve their products; however, the teams resisted the move toward this more unified approach to UCD, and we were having trouble understanding why. After all, everyone was saying that we needed to improve the products. At this discouraging moment, we decided to use some of our headcount to hire program managers and developers. We had seen that one issue that our UCD teams faced was that employees were spending too much time on process management or technical problems—not their area of expertise. We figured that by hiring product managers and developers, we could free up our UCD people to focus on their specialty. This assumption turned out to be true, but the real transformation came when these new team members changed our organization and opened up a way for us to transform the way our UCD organization functioned within the larger process.

3.2 Learning More about Our Internal Audience

We used the ethnographers in our organization who normally worked with users and customers to conduct an internal ethnographic study to help us better understand what our product management and development counterparts thought about UCD, about how our team functioned, and about how UCD could help them make better products. From this we learned three key things:

- Product managers and developers wanted a stake in the research and design process. They wanted to know that they were contributing meaningfully to the UI design. They wanted the solutions to match the requirements and to understand how the UCD team arrived at these requirements.
- Product managers and developers wanted all user experience guidelines, standards, and specifications to be in synch with delivered technology. Previously, synching user experience design standards and technology had been somewhat of a moving target, with UI designs running ahead of what was available in the technology. Developers wanted to build what they saw in prototypes and guidelines.
- Developer productivity was key. We scoured the bug database for the most annoying, time consuming, and frequently logged UI bugs. Fortunately, from one of the legacy product lines, we had data from mandatory UI reviews that we could mine. One thing we found was a problem with how key notation was generated. When developers marked fields as required, using a blue asterisk icon, they were also supposed to insert the key notation string as part of the header component. We found more than 1,000 bugs showing that developers were not

inserting the key notation because it had to be coded separately. Consider that it takes two to three hours to file and fix a bug. To optimize their process, we identified that placement of the key notation string should happen automatically when a field was marked as required, thus saving hours of extra coding or bug filing and fixing and improving overall quality and consistency. By automating the process, we enabled developers to focus on building the parts of the product that would differentiate it from our competitors.

3.3 Effect of Our Program Managers and Developers

Our program manager team members helped us understand what we needed to do to become fully integrated with the formal development process. Because the development organization was redefining this process in conjunction with a major release, the timing was ideal. In the beginning, a lot of what we had to do was to explain our process and requirements to the team responsible for defining the new development process. Once others began to understand our role and include us in the process, the tables turned, and we had to go back and look at the quality and consistency of our deliverables, ensuring that these deliverables were presented in a way that all audiences could understand, that these deliverables addressed the concerns of all stakeholders, and that all members of our organization understood how to create these deliverables consistently.

Because the development project included new UI's for every product, our team would not be able to do all the work. We needed to document the mandatory user experience deliverables in such a way that product managers could complete them as well. Had we not done the prior ethnographic work, it would have been much more difficult to do this well, and as it turned out, as teams worked through the design cycle, we did have to go through a couple of design iterations on the user experience part of the process. We were finally successful in defining a three-tier support structure to which all teams had to adhere. We then properly used the bug filing and tracking process for all user experience issues. When there were issues on which we disagreed with the design direction or approach of a particular team, the development process itself ensured that we discussed the issues and resolved them appropriately and on time. We made our process visible to other functions using the same tracking and metrics as the rest of the organization. Our program managers then helped us translate our UCD process to our executives and other team members and then translated their "language" back to us, ensuring that everyone was clear on the process.

The following illustration shows what the process evolved to over time. We were instrumental in generating this diagram, which became an effective teaching tool for all the teams involved.

Our developer and UCD architect team members participated as members of functional and technical working groups. Their work enabled us to understand how the UI would be delivered and what we needed to provide to the technical teams to ensure that we could build the UI that our users needed. Their work also gave us the opportunity to influence the technical direction, and because we were participating on the relevant working teams, our input was taken seriously, not as the agitations of clueless outsiders. We facilitated communication and cooperation among development, strategy, and product management and became in many cases the main conduit that led to the understanding of the solutions.

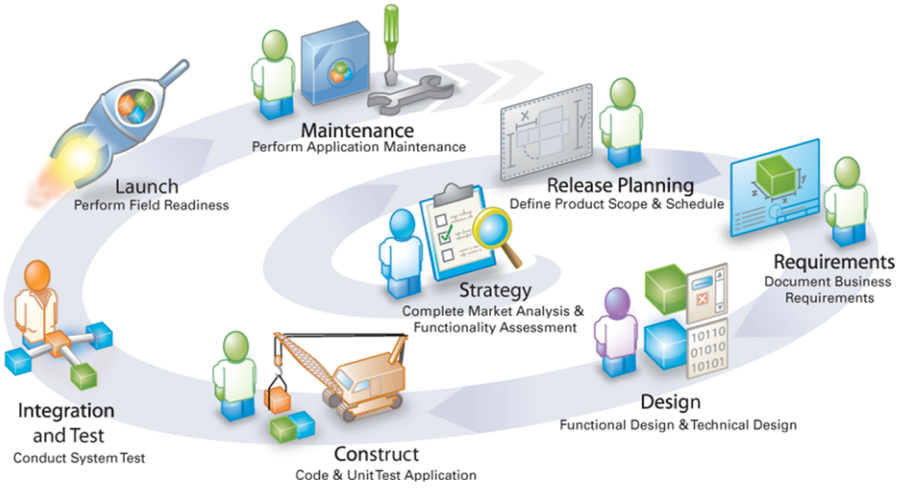

Fig. 1. The development cycle

As a result of these actions, we have successfully carried out the UCD process and avoided wasting time and effort justifying our process and resolving interorganizational problems. The design of the application suite itself, a thing of many moving parts, is now significantly more integrated in terms of the way the user experiences it. And the company now better understands the value and use of UCD. At all points in the process, UCD has a voice and a role to play.

4 Ensuring That UCD Provides Maximum Value

4.1 Get Top-Level Commitment to UCD

Top-level commitment to UCD really means that the organization recognizes that "usability" and UCD are strategic elements of the product offering. The organization may not understand all of the implications of applying these elements, but it knows enough to make an investment. Top-level commitment is demonstrated in three ways:

- Executive buy-in: Building executive buy-in takes time, unless you have a sponsor who comes in with previous successful experience with UCD. Part of your job is to increase the trust that your executives have in UCD through successful outcomes with more and more teams. The greater the number of people who have had a positive experience with UCD in an organization, the more momentum you can build around UCD.
- Adequate resources: An organization must appropriately budget for customer research, labs, test subjects, and staff for the number of necessary projects. If the major features of a product have not gone through some form of UCD, then the user experience will reflect the assumptions of the product builder, not the product users.

- Positioning of the UCD team: The UCD team must be positioned appropriately within the company to be effective. Both centralized and decentralized models can work. Our experience with an enterprise application suite is that the UCD team should be located in the development organization, as they are then seen to have the same level of buy-in as other members. As a centralized team, the UDC team can leverage expertise across the team, easily handle cross-suite issues, and load balance as needed.

4.2 Integrate with the Process

Integration with the process means that UCD is not a side activity whose outcomes may affect the product only if time permits at the end of the cycle, but instead that UCD activity is a full partner in the development process with all of the other players: strategy, development, product management, testing, and so on.

To ensure integration of UCD in the development process, a company must:

- Include UCD milestones and gating factors in the development process (whatever model your company follows), standardize deliverables, and ensure that all stakeholders understand these deliverables.
- Hire the appropriate staff to support the total effort. In order to integrate with the process, you must have people on your team who understand the perspectives and challenges of the other members of the organization Thus, you must go beyond UCD disciplines and include program management, developers, architects, and marketing specialists.

5 Conclusion

Instead of attempting to directly convert our organization to the language of UCD, we learned to communicate in the terms of product managers and developers by incorporating their disciplines on our team. We were able to meet these counterparts in the middle. Then we were able to introduce by example more detailed aspects of UCD as a normal and accepted part of the process. As a result, we are now a natural part of the process, from strategy to planning to execution. As we look forward, we see our organization beginning to function at the Institutionalized level, where the organization as a whole is human centered in its focus.

Reference

Earthy, J.: Usability Maturity Model: Human Centeredness Scale, Version 1.2 (1998), http://www.idemployee.id.tue.nl/g.w.m.rauterberg/lecturenotes/USability-Maturity-Model%5B1%5D.PDF