

Critical Review of Schrage's Serious Play

In Serious Play, Schrage explores the sociological aspects of product design by investigating how both individual and collective behaviors within a given organization affect that organization's ability to innovate. He argues that the primary catalyst for innovative ideas is this behavior or interaction, which results in successful products that deliver more value to an organization's customers. Based on several case studies and the discussions on organizational culture, Schrage advocates the use of prototypes as a principal technique for facilitating collaborative interactions.

Throughout the text, Schrage supports his theories with historical trends and anecdotal evidence from product-centric organizations that claim to value innovation. While examining Schrage's arguments, this reader encountered many difficult questions. Namely:

- Will prototyping become central to organizations and affect their culture as quickly and significantly as Schrage implies?
- Is prototyping really the best method for designing innovative products?

Subsequent sections of this paper examine these questions in detail, and include commentary from this author's perspective, as one who works in an organization that does not readily employ the use of prototypes and yet whose organization appears to be successful in designing innovative and valuable software products.

The Emergence of Prototyping Cultures

Schrage indicates that because the costs associated with prototyping tools and techniques continue to diminish, organizations will become increasingly eager to employ prototyping as a way to generate new and innovative ideas. Further, Schrage implies that the interactions taking place between people (innovators) and prototypes will create a new dynamic within organizations and thus change the landscape of the business culture (12).

As Schrage illustrates in "A Spreadsheet Way of Knowledge," the introduction of new tools and techniques historically results in changes to organizational structures and processes. It does seem reasonable to assume that an increased focus on prototyping tools and techniques throughout product development lifecycles will trigger some

significant modifications in an organization's culture. However, the idea that such a shift will be perceived as affordable or will be as widely adopted as Schrage suggests is uncertain.

Although affordability has resulted in rapid adoption of some technology by both individuals and organizations, there is more involved in encouraging organizations to prototype than the availability or cost of prototyping tools. As Schrage admits, effective prototyping requires a deep examination of an organization's fundamental ideology, including politics, assumptions, values, and perceptions (Schrage, 21; 32; 61). Prototyping organizations must also learn to manage themselves differently and learn how to encourage "serious play" (61). Unfortunately, it is inherently more expensive to introduce changes into an organizational culture than it is to maintain the current state of affairs. Making way for prototyping in an already established and/or seemingly successful organizational culture requires a concerted effort from all levels within that organization, and time is money. As such, the adoption of prototyping in organizations may prove more difficult and more expensive than Schrage implies.

Therefore, as with many of the assertions made in *Serious Play*, this author views the discussions of emerging prototyping cultures as a paradoxical situation. The paradox appears when one realizes that the integration of prototyping tools and techniques into a culture that will ultimately be changed by that integration also requires initial support from the existing organizational culture if prototyping is to establish itself as a valid tool in the first place. If one carries this idea in mind while reading the text, the arguments made start to feel irrelevant and the author seems to be oblivious to the real issues at hand.

Although Schrage discusses the somewhat obvious implications of prototyping on organizational cultures effectively, he may have been better off presenting more practical advice for organizations about how to adjust their organizational culture in preparation for a shift to prototyping, or with possible tactics for managing some aftereffects of this initial integration attempt.

Prototypes: the Ideal Technique for Innovation?

Given the difficulties of preparing an organizational culture for the introduction and aftereffects of prototyping as a method for innovation, one wonders whether it is really worth the effort. Throughout the text, Schrage advocates prototyping as the ideal method for "improvising with the unanticipated in ways that create new value" (2). He states, "tomorrow's innovators will invest more in playing with prototypes, modeling marketplaces, and simulating scenarios because that will become *the best way* to create

new value and profitably deliver it to customers” (Schrage, 15). In Schrage’s view, “companies that want to build better products and services *must* learn how to build better prototypes and simulations” (64). These are just a few examples that highlight Schrage’s belief in prototypes as “the” technique for organizations that want to innovate and design successful products.

While it is probable that prototypes promote atmospheres in which innovative behavior can grow and flourish, two questions remain. Can an organization that does not use prototyping as the center of innovation still create innovative products? If so, what other methods or techniques do successful organizations use to create the interactions that lead to innovation?

In the section “Culture of Prototyping at BEA Systems,” I describe my company as a spec-driven prototyping culture characterized by limited customer involvement throughout the development lifecycle and little of what Schrage calls “serious play.” Since one of Schrage’s primary conclusions is that prototyping is the best mechanism for collaborative discussion and for fostering innovation, it would appear as though BEA Systems should have difficulty creating innovative products that provide value for their customers. However, in an upcoming Java Developer’s Journal (JDJ) Edge Conference, BEA Systems will receive a number of awards for our products. The award most relevant to this discussion is the award for our WebLogic Commerce Server, which was recognized as the first runner-up in the “Most Innovative Java Product” category.

Based on Schrage’s arguments regarding prototyping and innovation, how does one reconcile this apparent contraction in theory versus practice? This author believes that Schrage also provides the answer when he states: “innovation is less the product of how innovators think than a by-product of how they behave” (Schrage, 1). Of course, when Schrage speaks of “behavior,” he is speaking of the interactions that result from creating and discussing prototypes. Schrage describes prototypes as mechanisms for “externalizing thought and sparking conversation,” and indicates that a prototyping culture can facilitate collaborative design practices by encouraging team members to discuss and resolve important issues before those issues are embedded deep inside the product architecture (Schrage, 14; 20). According to Schrage, prototypes also help “unearth choices” that can be evaluated early in the development lifecycle. Designers can use prototypes to “define the context for trade-offs” and concretely illustrate reasons behind design decisions to others in the organization (Schrage, 35). However, it is possible that successful organizations like BEA have already established mechanisms for interaction among team members that foster innovative behavior and allow for open discussion of alternatives and issues. Surely prototyping is not the only

way organizations can design innovative products that provide value to their customers, and Schrage seems to offer little proof that prototyping is in fact the best way.

Alternatively, one can argue that the lack of prototyping within BEA's development process is the reason why we are the first runner-up (as opposed to the "Winner") in the "Most Innovative Java Product" category. Our organizational culture at this moment in time is to be the leader in the market, often referred to by executives as TWD, or "Total World Domination." If TWD is the end-goal of every individual within this organization, then it clearly follows that runner-up is nice, but not good enough. If Schrage's conclusions about prototyping, innovation, and product value are correct, then BEA must make more room for prototyping in its culture. And this, unfortunately, brings us back to the paradox and the unanswered question "how?"

Conclusion

"There is a fundamental tension between thinking and doing: thinking impedes progress in doing, and doing obstructs thinking" (Carroll, 2).

Although taken from a reading about scenario-based prototyping techniques, the previous quote has special significance when one writes a critical review of Schrage's text. Serious Play provides an interesting look at organizational culture, innovative behaviors, and the creation of prototypes. However, with the exception of the chapter on "Measuring Prototyping Paybacks" and the "User's Guide," it does not offer much practical advice for organizations looking to move toward truly prototyped-based cultures.

As someone who works in the field and actively participates in a software company's development process, I believe that the challenge lies not in touting the benefits of prototyping or academically discussing the cultural implications of prototyping. It is unlikely that many engineers, managers, or others involved in the development process would argue against the principles of prototyping when presented with Schrage's (or another author's) arguments. Rather, the real challenge lies in preparing organizations for the integration of prototyping into their culture, and in helping organizations deal with the cultural implications of this integration. Those who examine organizations' social structures need to offer practical guidelines that will help organizations balance prototyping with the pressures and responsibilities inherent in delivering a quality product in a short amount of time.

The implementation of prototyping theories and techniques in an organization is no doubt quite challenging, and perhaps the problem is that no one really has any answers about how best to accomplish this task. Sadly, like the designer who stares at a blank piece of paper instead of creating some (any) prototype for users to evaluate, the failure of authors like Schrage to *attempt* practical solutions to these issues will only delay improvements further (Rettig, 23-24).

Culture of Prototyping at BEA Systems

As a Senior Technical Writer who has worked at BEA Systems, Inc. for a little over a year, I have experienced four releases of our e-business software. My knowledge of BEA's prototyping process described here is fairly limited, and comes solely from interacting with one product manager and one human factors/usability specialist in the e-Commerce Applications (eCA) business unit who, in the current and previous software releases, have begun to focus on designing graphical user interfaces for the highly technical consumers of our e-business product line.

While each release finds our software development team (product management, engineering, documentation, quality assurance, and so on) more aware of usability issues and more eager to address them, we are also faced with shorter times to market, rampant scope creep, and more ineffectual project management. These negative factors impact the ability of the few human factors/usability specialists from BEA's new Ease of Use Center or hired into individual engineering groups (like those I described above) to consider design principles and execute techniques that everyone in the organization (at least superficially) believes would improve our product development lifecycle. One of the techniques typically performed by the human factors/usability specialist that remains neglected at BEA is prototyping.

Based on Schrage's discussion, I would characterize the present culture of prototyping at BEA as a spec-driven culture. A subset of senior product managers, marketing specialists, and technical architects meet to determine the requirements for a subsequent product release while the software development team is finishing up the current one. After (sometimes) allowing the team to catch their breath, the various specifications that result from these high-level meetings are distributed to the entire organization via internal newsgroups. Specifications include Marketing Requirements Documents (MRDs), Product Requirements Documents (PRDs), and Engineering Requirements Documents (ERDs). It is probably safe to say that the purpose and

audience for each document is not entirely clear to everyone on the development team, and many sections (especially those dealing with ease of use, internationalization, and similar aspects) are left blank.

Nevertheless, these specifications are used as a basis for discussion among engineers on the development team. If the scope of the project is not realistic given the desired release date (which most of the time is inflexible due to quarterly earnings goals), engineering pushes back in an attempt to reduce the project scope. During this time, the human factors/usability specialist also reviews the specifications to determine the level of effort required to design any accompanying graphical user interfaces.

Following review of the specifications for feasibility, engineers on the team start delve into technical issues and discuss possible solutions. Simultaneously, the human factors/usability specialist attempts to meet with the engineers to discuss probable use cases, then designs some windows, dialog boxes, and other portions of the interface using a prototyping tool (most likely Visual Basic). The use cases are based on both the specifications and ideas the internal BEA engineers have about how they might use the product if they were customers. Much time is spent trying to keep track of new engineering decisions that are no longer updated in the specifications, and there is no direct contact with customers by members of the development team (even by the human factors/usability specialist) during this phase; it is assumed that the audience for the product has already been well defined in the specifications and is targeted to those with backgrounds similar to those developing the product.

Once the human factors/usability specialist designs some portions of the interface and models its interactions, this information is posted on an internal Web site for review. These mini-prototypes are professional in appearance and therefore seem finished. It is not known whether the engineers provide much in the way of interface design review comments. However, when coding of the front-end begins, the engineers attempt to follow the designs provided by the human factors/usability specialist. Because the engineers work within the NetBeans environment, however, the objects appearing in the prototype (or more accurately, the initial “draft” of the product) may not be available. In such cases, the human factors/usability specialist collaborates with the engineers to find an appropriate substitute from within the NetBeans framework. In other words, modifications are made to the design to accommodate the capabilities of the development environment.

After a number of iterations in the interface design (that are often brought to light because of technical issues), the “development complete” version of the product may be shown to some potential customers or a brief usability conducted to elicit feedback. Like other bugs in the software, issues are collected and prioritized using an internal bug tracking software. Tier 1 usability bugs are treated with the same importance (I

believe) as bugs in the underlying code. In other words, if the usability issue identified is characterized as a “show stopper,” it is highly likely to be fixed prior to General Availability (GA) of the software. Given the short time frame between development complete and GA, however, it is also likely that many important problems with the interface are not resolved.

Given that just two releases ago (approximately 8 months), BEA employed no human factors/usability specialists and did not participate in any prototyping activities, I am optimistic that the culture of prototyping at BEA will improve. However, it is also clear that we have a long road ahead. As Schrage suggests, BEA would probably benefit from placing more focus on prototyping and less on developing specifications in a black box. The design phase in our development lifecycle is still remarkably weak, and much more collaboration among engineers, human factors/usability specialists, and *actual* customers prior to and during coding phases is desperately needed. It is unclear as to whether loosening of the tight deadlines for product releases will have any affect on the prototyping culture, but such loosening is planned and therefore, only time will tell.

