



From eight-tracks to iPods – designing a customer experience

By **Howard Tiersky**, Vice President,
Interactive Solutions, Capgemini

Meet the competition for your next enterprise application: Apple iPod, Amazon, and Google. These products have set the standard: simple elegance, smart suggestions, and one-click answers. They have helped create a savvy audience that will not settle for the old days of fussing with eight-track tapes and trying to track down a reference librarian. They have succeeded in creating interactions that provide a sense of communion: experiences that consumers love.

These consumers are your target audience. They are both the end-users and business stakeholders of your internal and external enterprise applications. And not only do these users have high expectations, the evidence demonstrates that building applications that these users love is good business.

Capgemini designs a wide range of applications: CRM systems for call centers, web interfaces to enable consumers to manage their financial accounts, portals to support insurance agents' day-to-day sales and service activities, and sites for shopping for everything from automobiles to office supplies to electronics and apparel. Across this wide range of initiatives, we have seen a two-part phenomenon play out repeatedly. The first part: successfully creating applications that truly meet users' needs has a huge impact on business results. But the second part is perhaps even more important. Creating these results does not require magic or even genius. It's the result of the application of a set of highly repeatable, highly predictable activities—a process called "User-Centered Design" or UCD.



Making User-Centered Design work requires two basic commitments:

- The implementation of a set of structured activities which gather user "requirements" through interaction with representative users, followed by later tests of potential solution designs directly with users.
- The discipline to integrate information gathered regarding users' priorities and preferences into the decision-making process in a serious way.

Let's look at the benefits. For most enterprise applications, the "return on investment" for the solution is tightly correlated to the adoption of the solution by the targeted users. The degree to which they use the system, the completeness of their use and their level of satisfaction will be scaling factors of how much benefit the business receives from implementing the system. Whether you are trying to lower call center costs by moving your customers to web self-service or trying to encourage your sales force to leverage a CRM system to drive cross-sell, the need to drive desired user behavior is often at the core of whether an initiative will ultimately succeed or fail.

How does UCD impact user adoption?

- Users much more readily adopt applications designed through the UCD process, so risk of adoption failure is significantly reduced and less effort (and cost) is required on change management.
- Users can generally learn UCD-designed applications much more quickly, so training costs are lowered.
- Users tend to make fewer errors and require less support, again reducing costs.

Finally, our research has shown that IT solutions developed through a UCD process have lower risk of over-runs and fewer defects in the development process. Since UCD activities generally take no more than 15% of the budget of a given initiative and reliably demonstrate a far greater cost benefit in reduction of re-work and simplification of features, they typically pay for themselves through increased efficiency in the development process alone.

By following a proven set of activities, summarized below, and committing to seriously weighting your decision process in consideration of user needs, highly predictable adoption results can be achieved.

Implementing User-Centered Design to Meet Users' Needs

Across a wide range of applications, our research shows that the three most powerful user needs are:

- **Match my mindset:** “When I start using the application, it should be ‘intuitive’—meaning ‘it should think like me.’ Features and content should be where I would expect to find them if I simply guessed. Do not require me to learn and then remember how the system is organized.”
- **Pamper me:** “Don’t waste my time with steps that take longer than they should. If I provide you data, remember it. Don’t ask me again. Don’t create artificial barriers to moving through a given task, for example by requiring me to go to a different application and copy/paste my order number to check shipping status.”
- **Empower me:** “As I learn more about using the system, let me work faster and smarter in ways that fit my own personal patterns. Enable me to save key preferences and to customize my work environment easily and in the ways that matter.”

Making the right design and feature choices to achieve these three goals requires serious insight into how an application’s users go about accomplishing their tasks. Most applications fail in these three areas because they are designed in a conference room on a white board by business “stakeholders” who are over-confident in their knowledge of what their users really need and lack a clear, proven process for defining requirements.

For example, Capgemini frequently works for large insurance and brokerage clients. Before we begin the design of an application, we spend several weeks in “the field”—shadowing users of existing systems and observing how business is actually done. Inevitably, we come back from these anthropological expeditions with a treasure-



“Customers want applications that are intuitive—that think like them”

trove of insight of which the business stakeholder teams are unaware. Very frequently, these insights lead to clear hypotheses about which specific features are, or are not, needed by the users in the field and how individual features can be best assembled into an effective work flow.

The process of gaining this insight fits into a structured UCD process which follows eight basic steps:

1 Definition of business goals: Every initiative should begin with a clear set of measurable goals. It’s best not to define goals in terms of features (“to provide a personalized portal”), but rather, business goals should be stated in the metrics of your business—to reduce cost, to increase revenue, to increase retention, etc. The UCD process should then be used to determine what specific functions and features will help achieve those objectives.

2 User segmentation and profiling: The user base for any application is usually made up of several different audiences. These user segments may be distinguished based on their roles, activities, experience, or other criteria. A sales application may be used by field sales personnel, the call center, and sales force management—each having different needs. In addition, users may be segmented by level of experience, geography, product focus, or any number of other dimensions. It’s important to prioritize these user groups and to document clear thinking about how their needs differ. Data gathered from field studies, and possibly other research activities such as surveys, provides the basis for creating personas for each user segment.



User personas are thumbnail sketches that give us a better idea of each different user-type and what their key requirements are. These personas contain demographic information and often a fictional name, photo and bio of a “prototypical” user for each group. Understanding our segments allows us to design our application for “real” people. Some of our clients will print the personas on poster boards and place them around the office to remind the team who they are building the application for.

3 User observation and requirements gathering: Finding out users’ needs can often be less straightforward than one might expect. We have found that, when asked, the vast majority of users cannot articulate their needs. What’s worse, those few who can articulate them tend to represent a “tech-savvy” minority whose needs may not represent most users. Counter-intuitively, building the solutions that users say they want is almost always a recipe for over-engineering and delivering something most users actually do not like at all. As mentioned above, asking “business stakeholders” about customer’s needs, while worthwhile, is not a reliable gauge.

What does work is field observation (also called ethnography). This approach basically involves being a “fly on the wall,” observing users accomplishing their tasks in their “native” environments (office, living room, store, vehicle, as the case may be). Field research requires clear structure and planning, however, to define the key “research questions” to be answered, develop an observation protocol, and carefully select subjects.

Field observations give us the opportunity to see how users interact with the existing systems, what processes they use to accomplish their tasks, and the documents and items they typically use throughout the day. It gives us the chance to observe activities that can not be

uncovered by interviews. For example, the opportunity to see the work-arounds developed to existing systems, be they sticky-notes on computer monitors, cheat sheets, or ways of bypassing annoying required fields by entering dummy data. Seeing the “real world” in all its glory enables realistic design.

4 Requirements prioritization: The single factor most attributable to solution effectiveness and development efficiency is requirements definition. There are two primary steps to developing good requirements:

- Making good decisions and then...
- Documenting them in sufficient detail to reduce or eliminate “interpretation” at the time of development.

Making good decisions about solution requirements involves balancing four competing forces:

- **User needs:** the features and application characteristics which will motivate users to utilize the system and then help them achieve their goals. Ease-of-use, good search tools, and integrated functions are common user needs.
- **Business needs:** characteristics of the application that support the goals of business stakeholders, which may differ from what users primarily care about. Common business needs include cross-sell, fraud prevention, gathering marketing data, and driving self-service.
- **Compliance requirements:** needs that may relate to tax law, privacy, or other legal requirements across industries as well as specialized compliance issues in regulated industries such as brokerage, insurance, utilities and pharmaceutical.
- **Technology constraints or influences:** we don’t want technology driving our requirements. However, as we balance different potential features, the level of difficulty of implementation as well as performance and scalability issues need to be considered.

Optimizing the balance between these forces is critical to making the right decisions in the requirements process. It’s at this point that the true commitment to integrating the view of the user becomes essential, especially when it may conflict with business requirements or technology influences. If a shared commitment to UCD has been agreed to amongst the stakeholder group, hard data from field research should help drive and streamline decision-making—focusing the discussion on what we know is going to impact users and drive behavior in a meaningful way, rather than having a “battle of opinions” about what users “really want.”

At the same time, not all customer needs must be met 100% in order for the initiative to be successful. The first iPod was popular even though it was more expensive and heavier than what consumers really wanted.

5 Prototyping and user testing: Field observations provide deep insight into users’ needs; however, making the leap from diagnosis to treatment is not always immediately obvious. User Centered Design relies on an iterative approach to solution design whereby an

initial solution hypothesis is created and then a series of prototypes are tested with users via highly structured techniques. Initial prototypes may simply be sheets of paper with outlined diagrams of the proposed solutions (called “wireframes”). Later, fully clickable, interactive prototypes that actually connect to data sources using simulation tools such as iRise may be utilized. Usability testing focuses on observing representative users attempting to accomplish key tasks with the prototypes and documenting their successes and failures. Similar to field testing, we want to observe the user “in action.” We have found that its not especially effective to “demo” a prototype to users and then ask “what do you think?” Rather, users are given fictional task scenarios and asked to demonstrate how they would use the prototype to accomplish a series of “real world” goals.

There are a wide range of variants of testing protocols from task analysis to card sorting, but in all cases, the goal is to gather objective data that can be compiled into a predictive model of how the application, as designed, would perform if actually built and deployed, and to identify areas which are in need of remediation. Arriving at a prototyped solution which has been proven to meet users’ needs prior to development is several orders of magnitude less expensive than “rapidly” building an application based on “assumptions” about users’ needs and then experiencing project failure due to incorrect assumptions or needing to engage in significant development re-work.

6 Detailed requirements definition: Once the basic features and user flows are defined, detailed use cases are typically developed to provide the level of specifications necessary to drive development activities. The UCD function continues to be critical during this phase of work, even though much of the user interface may be finalized. The process of defining detailed requirements addresses alternate task flows, error messages, and various special-case situations. The knowledge of user needs and the principles by which the “large scale” requirements decisions were made need to continue to be applied down to detailed decisions.

7 Rollout planning: As an application moves through the development process, it is critical to thoughtfully consider how and when it will be introduced to users and how they will be motivated to use it. Two specific areas of consideration benefit greatly from the UCD process: release planning and user adoption support.

Most web applications are deployed in a series of releases with increasing functionality from release to release. Key strategic questions faced by solution teams are—how much of the application needs to be built before it is “worth” releasing? And, what sets of features and capabilities make sense to be grouped together into a single user release? Unfortunately, the decision of when to roll out the first release is often driven by external timeframe considerations that may not relate to the question of how much of the application is needed to be “useful.” Decisions regarding which features are best grouped together are often made based on technology dependencies and development expedience rather than what features “make sense” for users to receive together. The risk here is very real. The goal should be for users to gain benefit from

the first release of an application and look forward to future, even better releases. All too often, initial releases of enterprise applications result in user dissatisfaction and backlash which can result in a negative spiral from which it can be difficult to recover.

Through the UCD process, we can reasonably predict (and test) what sets of functionality are dependent on each other from a user’s perspective. For example, if an insurance company releases an application to provide insurance quoting, but is not yet ready to provide a new application to actually write the policy, it may mean users then have to re-enter data into the “old” system to complete the transaction. Under these circumstances, it may make sense instead to first release an application that allows both quoting and policy initiation, but only for one type of policy. While this does not fully meet users’ needs, at least users can experience a positive, complete workflow for one discrete task, and await the next release when the same benefits will be applied to others.

Finally, the release of even the best-designed application needs to be accompanied by appropriate activities to support user adoption. These typically include communication, training, support, and incentives. The insights gained about users’ needs from the various forms of research and testing performed to this point are a very valuable source of direction for these activities. What kinds of training will best serve users? What issues are they most likely to have with the application? What are the aspects of the solution they are likely to be most enthusiastic about and what concerns need to be alleviated before they will feel comfortable?

8 Build and deploy: One question we are often asked at Capgemini is how much time the UCD process typically adds to the development lifecycle. Our answer is that sometimes it adds a small number of weeks to the initial plan, but typically cuts a greater number of weeks off the actual delivery schedule by streamlining decisions, encouraging simplification of the solution definition, and documenting clear design standards to avoid re-work. The impact on budget is similar; it may add 10-15% to the project budget, but subtract 30-300% from the actual project cost had the UCD process not been employed. We have studied specific examples of projects developed with and without the UCD process and therefore have data to back up these assertions.

Conclusions/Recommendations

Our experience has shown that companies that integrate a systematic approach to building a “customer experience” into their technology process substantially lower development risks, reduce deployment costs, and have higher adoption rates.

In the real-world environment of limited budgets and resources, not every application can be the next iPod, Amazon, or Google. However, that does not mean we have to settle for a contemporary equivalent of the eight-track tape. Instead, by partnering with the future users of an application, we can make substantial progress toward creating solutions that meet, or even exceed, the expectations of today’s saavy web consumer.