Successful Product Interface Development

March 29, 2005

Echo Visualization, LLC.
500 Bishop Street, Suite F-9
Atlanta, GA 30318
404.389.0873 - Phone
404.389.0886 – Fax

create@echoviz.com
www.echoviz.com

Successful Product Interface Development

The Importance of Interactive Prototypes

Interactive applications and product interfaces -- whether delivered via the web, on CD-ROM/DVD-ROM or as installed executables -- are incredibly powerful representations of ideas and actions. If a picture is worth 1,000 words, an interactive piece is worth 100,000 words.

In terms of expressing a product idea or a concept, verbal or written descriptions accomplish a great deal. A picture or series of pictures is more descriptive. A physical model is *an order of magnitude* more descriptive. Allowing someone to interact in a meaningful and accurate (or nearly accurate) way with a device or application is the ultimate method of testing a concept, product or idea. People learn and understand by doing.

Planning is Critical

Designing a product interface is like any other part of the product development process: it should be iterative, tested and refined *before* development begins.

In an ideal world, an interface should be tested early and often with a small group of people not invested in the development cycle. Ideally, the testers should be actual users. A group of 3-5 people can be brought in 3-5 times to review index cards with images, PowerPoint slides of static screens, written scenarios of actions, iconography, colors, fonts, presentation and clarity of information, use and volume of sounds, etc. In the latter stages of this early testing, very simple interactive vignettes - snippets of activity - should be included in the evaluations.

With this early and often consideration, meaningful refinement to the design and usability of the product can take place before development is at the code level. At this point in the design process, these marketing and testing decisions can also be made:

- To what level of fidelity does the interactive application need to be developed?
- Will the fidelity evolve over multiple versions of the application?
- Who will use the interactive application?
- How will it be used?
- What type of usability testing will be employed?
- How will the interface be delivered - web, CD-ROM, DVD-ROM, installed application on a laptop, on a workstation, on a prototype of the product, etc.?
- Will the interface send or receive data?
- How will errors be triggered/simulated within the interface?
- Is error recovery critical to successful use of the product/interface? How will this be tested and evaluated?
- How will external input be triggered/simulated within the interface?
- Will it interact with actual, physical equipment (either legacy, new or supplemental)?
- After this round of testing/demonstration is complete, what's next?
- Will there be a Version 2 of the interactive piece?

- How many changes are anticipated for Version 2?
- What types of changes are anticipated for Version 2?
- Will there be a Version 3?
- Will the interactive piece serve as a virtual prototype for the programming team?
- Is there a desire to capture or reuse code from the application for the production software?
- Will this evolve into a training piece? What type of training will it support?

Ultimately, planning for the use and evolution of the interactive application is of paramount importance - even if there isn't follow through and implementation of all of the things that have been projected or planned. Once programming commences and code is involved, it can be costly to go back and make changes. Programmers generally charge between $120 and $250 an hour. There's no way to short-cut code; it's brainpower per hour.

Best Practices

At EchoViz, our best practices indicate that the following should be considered and budgeted for from the beginning of any product interface project:

- Plan for greater fidelity than you think you need. It'll be cheaper in the long run to add more fidelity if you've planned for it from the beginning.
- Plan for the possibility that the product will need to pass data to and from a database or other external component. Again, it'll be more cost effective to add this functionality later if planned from the beginning. The code will already support it - or at least will support the possibility.
- Determine early how inputs - like things that trigger errors or state changes - should be handled. Deciding after the fact that a "dumb" system suddenly needs to accept variables and data from an external source can be costly and time consuming to remedy. There are no short cuts if code has to be rewritten.
- Complete the thought. In other words, don't make changes along and along. Build it. Evaluate it. Modify it. Interaction relies on complex relationships. Build all of the relationships and then determine which ones require massaging. Continual changes throughout the product development process results in higher costs. Changing code over "here" might break good code over "there." Changes ALWAYS require testing and more testing to insure that everything works as expected. Even the simplest interfaces are complex systems.
- Insure that all of the stakeholders are on the same page prior to implementation. Manage the expectations and the budget from the beginning and the results should satisfy everyone.

Bottom line: If the product interface is an afterthought - in terms of priority and budget, the results will be acceptable at best. If it's considered an integral part of the product development process and appropriate attention is paid to it (with budget, time and personnel), the results can be spectacular.