Clarifying the Fidelity Dimensions of Prototypes

Version 1.0 Dec 8, 2000

James F. Zdralek
Department of Psychology; Carleton University
Ottawa, Canada

Abstract

In this paper the multiple dimensions of prototyping will be clarified. The position of prototyping in the iterative design process will be located and debates about the relative merit of Low vs. High fidelity Prototyping will be summarized.

Communications and Design

Why prototype? Prototypes are the manifestation of ideas that are in germination. As the prototype is built the details must be decided upon. The concept within the designer's brains contains the overall idea. The manifestation acts as a communication device to show other people the idea and gather consensus amongst the team members. As feedback is gained from other people the prototype changes as improvements for the idea are suggested.

> A low-fidelity prototype can be the communication medium by which requirements can be articulated. A prototype can serve as the common language to which users and developers can relate.
> (Rudd 1996)

Prototyping is a communications device with three main communication paths. Prototyping helps designers communicate amongst themselves. To build consensus amongst the team and visually show developers the product. Prototyping also helps designers communicate with reality. Many interfaces can look good in a sketch or on paper but confounding details can show up when things start to move and interact or constraints are added. Building a prototype can help clarify the reality of how the interface really works.

> “The process of building a high-fidelity prototype always helps to identify weakness and omissions in a user interface specification.”
> (Virizi Sokolov and Karis)

Prototyping also helps designers communicate with the user. This is the communication that has to have the most clarity. Any breakdown in the communication at this point will prevent the user from understanding the software. The designers cannot communicate directly with the user therefore all communication is done through the interface and the prototypes are the first attempts at a dialog.

Low vs. High

When the tools for interactive and digital prototyping became simpler to use debate about the fidelity of prototyping emerged. Jim Rudd gives a definition of high and low fidelity prototyping.

> Low-Fidelity prototypes generally require a facilitator, who knows the application thoroughly, to demonstrate or to test the application. Interactivity by the user is somewhat restricted. The user is dependent on the facilitator to respond to the user's commands to turn cards or advance screens to simulate the flow of the application. In contrast, high-fidelity prototypes are fully interactive.
> (Rudd 1996)

An agreed upon definition of what a high fidelity prototype has not been found but this definition by Tullis has a good point. Problems surface because of the lack of clarity as to what the true definitions are for the terms Low Fidelity and High fidelity.

Interaction

Paper vs. Screen is one of the most common of the fidelity dimensions debated. The issues are the rapid ease with which paper can be edited but the limits to which a user can interact with the prototype.

There are many benefits of using paper as a design medium and, with fast iteration, it is one of the keys to good design. There are sacrifices to taking the prototype beyond paper. The ability to eliminate bugs on the fly in a paper prototype is illustrated by Rettig and shows how the shortcuts taken to write quick code for interactive prototypes can have problems.

> A single bug in a hi-fi prototype can bring a test to a complete halt... I often see teams correcting "bugs" in a paper prototype while the test is in progress.
> (Rettig 1994)

Thinking that paper prototypes are close enough to the end result and that the details will work is a bad trap to fall in. As stated by Rudd the testing of prototypes should be interactive. Scenarios and storyboards fall apart when attempting to specify final designs. It is screen based prototypes and interactive mediums that can be used at these stages where design tests are used for confirmation of details rather than confirmation of concept to which story boards, paper prototypes and “day in the life” videos are useful.

> Low-fidelity prototype may not provide a good forum for user evaluation. Because they are often demonstrated to, rather than exercised by the user, it is more difficult to identify design inconsistencies and shortcomings.
> (Rudd 1996)

This is where the first “Fidelity Dimension” is defined with paper being “Low”. “Medium” can be defined as being a computer screen and “High” defined as being the physical product or site location where the interface would be used. The first fidelity dimension “Location” is separated out from the general Low and High Fidelity definition so that the confusion that occurs because people associate paper prototypes and screen prototypes with only interactive testing situations. Scenario development can also evolve from paper to screen and often become an animated spec. These linear, presented media are very useful to show how the software would work in certain situations. “Participation” is another fidelity dimension that has “Low” defined as being “Presented” where the user is shown a situation and watches the action take place. “Medium” can be defined as being a “scripted interaction” where the user is asked to do a specific task. High can be defined as an “explored” interface where the user can familiarize them selves with the interface by “trying it out”.

These two Fidelity Dimensions are closely associated and typically defines the type prototype developed. These types can be Paper prototypes, Storyboards, Interactive Demos, Simulations, Sketches or “Day in the life” Videos.

With “Presented” selected as the “Participation” dimension and the “Location” dimension varied from paper to computer to site. The prototype can be a storyboard then a day in the life scenario and then a presentation to a customer. At the “Scripted” level the prototype becomes a Paper Prototype, an interactive demo or a work place usability study. At the “Explored” level the prototype becomes an extensive Paper Prototype, a detailed simulation or a field test.

Who evaluates the design is the third Fidelity Dimension that determines “Interaction”. At the concept development stage where ideas are being germinated and things change rapidly the analysis is the blunt comment and sharp opinion of colleagues and coworkers. As paper prototypes and storyboards evolve through the project, heuristic evaluation and cognitive walkthroughs are useful for analysis of historic interface problems that have occurred before.

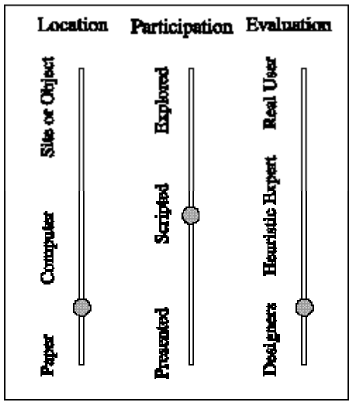

Figure 1 Interaction

These methods use experts other than the user that are trained to spot usability problems based on prior experience and standards. These are very good methods that are cost effective and can be applied to interface prototypes that are either paper-based sketches or fully interactive simulations. They find problems that are historic and standard but once a few of these discount usability studies have been completed the next iterations should be evaluated differently. Heuristics and cognitive walkthroughs are valuable but not a substitute for testing prototypes with real users in a usability study.

Spread

The division of prototypes into fidelity levels beyond hi and low can also be done in the form of vertical and horizontal prototypes. Vertical prototypes are solidly constructed sections of the interface often involving code that gives the user the ability to do work with the interface. These prototypes commonly have dead ends in other areas that the user cannot test. Horizontal prototypes are broadly defined interfaces that allow some exploration of the scope of the interface but no depth or real functionality. They cover more areas than vertical prototypes but do not have the true deep functionality of a vertical prototype. The over lap is that low fidelity is often horizontal and high fidelity is often vertical. The Vertical and Horizontal designation of prototypes are not exclusive aspects. The prototype may allow exploration by covering the surface aspects of the features but may also allow for deep exploration into one of the features or use cases. The subtlety is that a paper prototype can be generated that is horizontal, covers the range of general functions, but is also vertical, looks in great detail at a specific section of the prototype. Likewise a software-coded interface can be a horizontal prototype because it involves the entire interface but the supporting code to do real work is lacking.

Tullis contends that the fidelity of a prototype is judged by how it appears to the person viewing it, and not by its similarity to the actual application. In other words, the degree to which the prototype accurately represents the appearance and interaction of the product is the determining factor in prototype fidelity, not the degree to which the code and other attribute invisible to the user are accurate.
(Nielsen 1993)

Horizontal prototypes can be used to test general concepts and organization but vertical prototypes are more useful for proving coding capability and to test specific interface issues.

The main advantages of horizontal prototype are that they can often be implemented fast with the use of various prototyping and screen design tools and they can be used to assess how well the entire interface "hangs together " and feels as a whole.
(Nielsen 1993)

The effort to build prototypes is done to find mistakes before they are built into the final product.

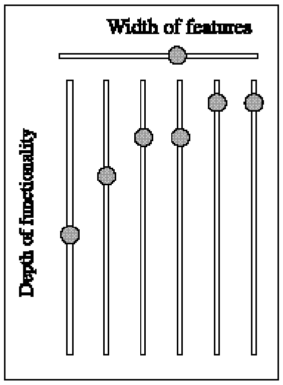

Figure 3 Spread

A lot of problems can be discovered by testing a compete prototype covering both the width and depth of features but this thoroughness must be balanced with cost and complexity. Care should be taken that the interface does not fail in the scenario user comes across 20% of the time or the scenario that 20% of your users come across.

On the other hand, unrealistic scenarios and data can severely damage the credibility of your design.
(Rettig 1994)

At first the fidelity dimensions of vertical and horizontal seems related to the participation aspect where an “explored” interface is a wide horizontal prototype and a scripted prototype can be vertically going deeply down one path. Horizontal and vertical are neither exclusive nor absolute. The “Spread” can vary in the degree of either or both of the fidelities of vertical and horizontal.

Refinement

Polished vs. Messy is another of the fidelity definition problems that garners debate. One side of the argument is that a graphically well balanced and visually pleasing interface will present the user with something less to pick apart freeing up time to look at structure. Arguably if a user is looking at an unbalanced screen then they may have to filter out or process confusing information.

With a slick software prototype, you are just as likely to hear criticisms about your choice of fonts, color combinations, and button sizes.
(Rettig 1994)

Another aspect of the debate is that a graphically balanced interface may cause the user to like the interface even although there are structural problems while the user may end up disliking a graphically unbalanced interface for similar reasons. Whether these dimensions of fidelity interfere with the discovery of usability issues is unclear but a study by Wiklund, Thurrott and Dumas, J. looks promising.

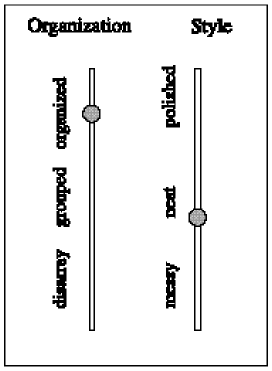

Figure 2 Refinement

Will a polished slick prototype prevent people from identifying possible design problems within a design? The theory is that with a high quality visual interface problems will not be as obvious when using a paper prototype or a computer simulation. Users will be less critical of graphically polished prototypes causing a limiting effect on suggestions. The fidelity dimensions stripped out of the general definition of Low and High Fidelity are the “Style” and the “Organization”. A prototype can be well organized but messy or can be badly organized but slick and polished.

Extreme Definitions

The Highest Fidelity would be a user explored interface that is site located on the device it would be used, covering all general areas and going deep into common tasks with well-organized and graphically polished screens. The Lowest Fidelity would be messy, unorganized paper sketches, presented to the design team, covering a few main screens. There is an obvious trend to the prototype fidelities. As the project evolves the fidelity measures increase from near the lowest toward the highest with occasional and spontaneous reduction at points where rapid evaluation and design iteration are merited.

Fidelity Escalation

The first stages of an idea usually exist in the form of sketches. As the prototype evolves the details are added and it becomes more refined. Often the prototype evolves to a stage where interaction is possible through multimedia or rapid development programs. This process of refining improves the solidity, completeness and quality of the prototype. As problems are discovered with the prototype the quality degrades momentarily as the solutions offered for the problem are sketched out and storyboarded. An arguable linear progression of fidelity could be diagramed as follows.

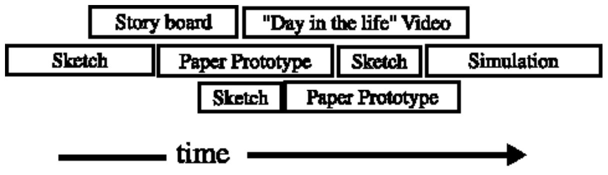

Figure 4 Time line

After the designers agree upon the design of the prototype the first functional use should be to view it and apply heuristic evaluation. The second is to watch and listen to users exercise it to help find hidden issues. The third would be to collect hard data for usability comparisons. As the prototypes function within a project evolves the fidelity dimensions across the full range should increase.

> Remember that the goal of low-level prototyping is to evaluate design alternatives and depict concepts. Do not get bogged down in the details. Evaluate broad-brushed approaches to particular user-interface design approaches.
> (Rudd 1996)

Do not start to code the product if the prototype is too vague or incomplete. Design the details and solve the issues so the programmers don't have to wing it as they go. Good interface design is in the details. The broad brush can be used at the storyboard, paper prototype and video mock up stages. It is better to decide on as much of the details as possible before the developers start investing time. If the design is solid and without vagueness the cost of development should be reduced and less time shall be wasted on correcting the design late in the process or supporting a badly designed product.

> Do not use low-fidelity prototyping anytime after the product requirements have been decided upon and coding has started. At this point in time, the prototype is too vague and too incomplete to give the programmers the guidance they need in developing the product.
> (Rudd 1996)

In the initial phases the prototype is typically a sketch on the drawing board shown to a few users. It gradually evolves to a highly polished usability-testing prototype shown to many people. The types of prototypes range in quality and use. The fidelity dimensions change with the time line of the design cycle.

This progression is not necessarily a single line through which the prototype evolves. The prototype could start as printed screen shots, be developed as a video presentation, return to pencil sketches then to animated screen shots and then a coded interface. These prototype fidelities could also exist in parallel. A coded interface prototype may be used to test the defined areas while paper prototypes of undefined areas are rapidly iterated.

The iterative escalation should be toward higher fidelities on the separate dimensions where the specific design details are very concrete. Lower fidelity prototypes can be used to get very broad definitions of the interface and requirements but you can be fooling yourself and hiding some problems. Developing higher fidelity prototypes should be done but it should not interfere with the lower fidelity focus of quick testing of alternate designs both at the beginning of the process and as it cycles through the timeline.

Lower fidelity prototypes can be useful for graphic layout, navigation, mental models, and structure but higher fidelity prototypes are useful for gathering detailed specs.

Usability Testing

The closer the Usability tester, Designer and Prototyper the better. The terminology for timing of testing is "formative evaluation" when you are still in the development stages versus "summary evaluation" when it is too late to make changes but confirmation that the design is correct is sought. The formative evaluation stage is avoided because of perceived costs and consumption of time. Products that has not undergone iterative prototype development and lacks formative evaluation of these prototypes will likely have hidden costs such as customer support or worse, lost customers. The usability tests can be the designers trying the prototype, a heuristic Expert evaluating the design or real users exercising the interface. The best thing is to cycle as rapidly as possible. Predetermining a schedule will help keep the project on track and more importantly the prototype in front of evaluators. If weekly or biweekly user evaluations are scheduled then whatever has been built up to that point can be evaluated. Someone should be responsible for finding and scheduling the availability of these users. This schedule of evaluations can also prevent excuses that partial or buggy prototypes can not be used for evaluation. Finding usability problems early is the key and putting it off prevents finding things that should be changed. Early usability testing may find issues that will scrap the area that is buggy or partially finished.

Iteration

Burning up your users is a problem some have mentioned. “Don't waste your users” is heard on occasion when the prototype is not yet developed enough or there are no clear gains. If you don't have enough users that you can “use up” a few on testing paper prototypes then try combining the usability tests with market research. If you cannot bring people in to try the prototype then going to them or even remote evaluations such as web site telephone interviews may be a viable solution.

> SET A DEAD LINE - If you know four users are coming tomorrow expecting to work with your prototype, you'll have far less trouble with fixations on minor details.
> (Rettig 1994)

When your prototype is closing in on the evaluation day run an evaluation with discount methods such as heuristic evaluation or cognitive walkthrough. Follow this with a test using a colleague. Try one user the day before a group is lined up, you may find some holes where a college knew what to do but a fresh user may be naive to the concept. Finally bring in the full set of users.

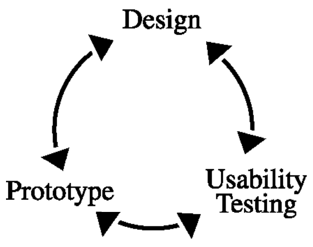

Figure 5 Cycle

Once it has gone through the usability testing the feedback is integrated into the design. At this point the prototype fidelity dimensions should be evaluated.

> If quality is partially a function of the number of iterations and refinements a design undergoes before it hits the street, low-fidelity prototyping is a technique that can dramatically increase quality.
> (Rettig 1994)

As the prototype progresses, it goes through several iterations. The design benefits the most if the prototype is tested with real users. The number of iterations of the design-prototype-test cycle improves the quality of the resulting design.

Long Term Communications

If the design of interface is the manifestation of the idea, then a document or record should be generated that can be the manifestation of the idea justifications. Ideas or concepts that were tried but found to be lacking would be recorded so that the next iteration of the interface, which may be developed by different people, will not follow that path without the gained knowledge of the previous versions. The instantaneous decisions made using a rapid cycle or lower fidelity prototypes may not be well thought out.

> Given the speed, in which rapid prototypes can be generated, there may be the tendency to make rash decisions regarding design options without ensuring that the proper validations are conducted.
> (Rudd 1996)

Histories of design changes and decisions are sadly lacking in the design community. A list of the decisions and the justifications behind the concept is needed. Both to identify areas that are pitfalls as well as areas that were glossed over and "left for the next version". Problems occur when things are not changed because the designers think "they must have had a good reason to do it that way" or things are changed to the worse because the designers think, "they probably didn't have time to come up with this idea". A “design justification and decision history" is a document that is not part of a spec but is a historic document. This history would identify whether an option was decided to be lacking because of consensus or whether the option was empirically evaluated and found to be the lessor.

> A prototype is a form of design specification, and the final implementation of a user interface is often performed with the prototype as a major way of communicating the design to developers. Unfortunately, the prototype can be over specified in some aspects that are not really intended to be part of the design.
> (Nielsen 1993)

Testing Objectives

As the fidelity escalates the focus broadens or narrows. Specific objectives in terms of known usability areas can be prototyped in detail while the general feel of a product can exposed details lacking in the spec. This list of considerations and advantages from Nielsen is valuable but reinterpretation of some of them should be done within the context of multiple Fidelity Dimensions.

Considerations
- Cost Constraints
- Define Market Requirements

Schedule Constraints
Screen Layout
Navigation and Flow
Proof Of Concept
Communications Medium
Training Overview
Training Tool
Usability Test
Basis for Coding
User Driven
Facilitator Driven
Data Collection
Look And Feel Of Product

Low fidelity advantages

Lower Development Costs
Evaluate Multiple Project Design Concepts
Useful Communications Device
Address Screen Layout
Useful For Identifying Market Requirements
Proof-Of-Concept

Low-fidelity disadvantages

Limited Error Checking
Poor Detailed Specification to Code To
Facilitator Driven
Limited Utility after Requirements Established
Limited Usefulness for Usability Tests
Navigational and Flow Limitations

Hi-fi advantages

Complete Functionality
Fully Interactive
User driven
Clearly Defines Navigational Scheme
Use For Exploration and Test.
Look And Feel Of Final Product
Serves As A Living Specification
Marketing And Sales Tool

Hi-fi disadvantages

More Expensive To Develop
Time Consuming to Create
Inefficient for Proof of Concept Designs
Not Effective For Requirements Gathering.

(Rudd 1996)

Higher fidelity prototypes require much more development time than lower fidelity prototypes. A balance must be struck as to how much you gain from lower fidelity prototypes and what must be confirmed with a higher fidelity prototype that acts like and is in the same environment as the final product.

Programming errors and system difficulties will sidetrack the test session from the focus on the user's task and the interface...
(Nielsen 1993)

As the prototype escalates further up the fidelity dimensions the attributes of the prototype change. The cost in terms of time and money increases. The difficulty in altering the deign increases. The pace of iteration slows. The coverage of design issues broadens and the ability to transfer this information increases.

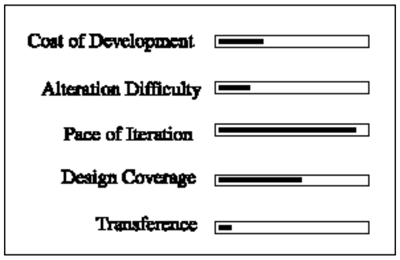

Figure 6 Attributes

Interface specs can be hundreds of pages of dialog describing a function. A linear prototype provides an animated spec that the developers can watch. Simulations can have good transference as well since they can give the developers a feel for the design and potential customers can be sold on an idea at a trade show. For the task of testing using a fully interactive simulation is the best but transference is sometimes better with an animated spec or “day in the life” video. For scenarios or describing situations it is best if linear animated specs are used to give to developers or for tutorials for manuals or for marketing material for trade shows.

Politics of Prototyping

A common pitfall is to work too hard on the prototype and start to defend it rather than take suggestions. This is the reverse to most designers’ main premise. You are trying to improve the work and so suggestions are what you are seeking. The point of a prototype is to have something for people to point at. The more details the more they will have to point at and find wrong. It may be difficult to stomach the number of prototypes trashed during a project.

Developers resist changes. They are attached to their work because it was so hard to implement. Spend enough time crafting something and you are likely to fall in love with it.
(Rettig 1994)

Other objectives go beyond good design. Promoting usability and iterative prototyping within an organization means a design team has to look professional.

“High fidelity prototypes are more successful at convincing managers and development that a particular user interface approach will work”. (Rudd and Isensee 1994)

Highly polished fully interactive testing simulations can be very influential and impressive. The danger comes when these types of prototypes are started too soon in the design process and are used to wrongly promote a design that has not been fully tested.

Being able to produce material that is reusable by marketing or training is a promotional bonus that could free up more money for the iterative design process. This reuse has a thorny edge to it that is another debate within prototyping. Reuse by marketing and training is fine since they are looking to only show the effects. The sharp edge is exposed when the prototype becomes the product.

The lower level the language used to code the prototypes the more chance of bugs creeping in. The speed of the iteration increases the likelihood that the code will be buggy and inefficient. Using higher level languages and card flipping programs to design interfaces ensures a limited complexity of code and also ensures that code used for the prototype, which is bung and thrown together for a user test, is not used in the final product.

If the code is started and written for the prototype the pitfalls can occur when programmers take shortcuts in the prototype but reuse this bung code in the final version of the product. Prototypes should be created in a language completely separate from the final language used for the final product. A sticky situation comes with generating vertical prototypes. The code used to give the user the ability to do real work may be written by a developer and may be compromised if rushed to write it in time for the prototype testing. If, to save money and development time, the code from the prototype is be included in the final product then disaster can strike.

A distinction must be made that prototypers are not developers. Prototypers should be adept at “faking it” and “rapid iteration” not writing robust efficient code. Prototypers should concentrate on the full range of prototyping techniques and not on the code used for development.

Fidelity Dimension

Prototypes are not the polar extremes of Low or High and they do not progress linearly from Low to High. There are a plethora of fidelities that can exist in parallel.

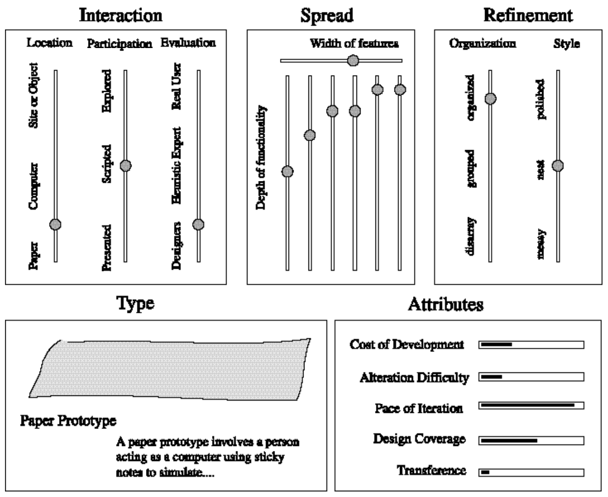

Figure 7 Fidelity Dimensions.

Fidelity dimensions of prototypes should be considered at the beginning of a project to plan a time line that can remind people when they should throw out the prototype they are working on and start the next iteration in the design-prototype-test cycle.

References

Alavi, M. (1984) *An assessment of the prototyping approach to information systems development.* Communications of the ACM. 27(6), 556-563.

Boehm, B. W., Gray, T. E., Seewaldt, T. (1984) *Prototyping versus specifying: A multi-project experiment.* IEEE Transactions on Software Engineering. , 10(3), 290-302.

Greenberg, S. (2000) *Prototyping for Design and Evaluation*, CPSC 681 Research Methodologies in HCI, Website

Gould, J., Conti, J., and Hovanyecz, T. (1981) *Composing letters with a simulated listening typewriter.* In Proceedings of the ACM Conference on Human Factors in Computing Systems, p367-370. ACM Press.

Muller, M.J. (1991) *Pictive: An exploration in participatory design.* In Proceedings of the ACM Conference on Human Factors in Computing Systems, p225-231, ACM Press.

Nielsen, J. (1993). Extract-Chapter 4.8: *Prototyping.* In Usability Engineering, p93-101, Academic Press.

Overmyer, S. P. (1987). *Rapid prototyping in the design of large-scale interactive information systems.* Human Factors Society Bulletin, 30, 3-4.

Rettig, M. (1994). *Prototyping for tiny fingers.* Communications of the ACM, 37(4), ACM Press.

Rudd, J., and Isensee, S. (1994). *Twenty-Two Tips for a Happier, Healthier Prototype.* Columns: Methods and Tools / interactions v.1 n.1 p.35-40

Rudd, J., Stern, K. and Isensee, S. (1996) *Low vs. high fidelity prototyping debate.* Interactions 3(1), p76-85, ACM Press.

Seminara, J. L. (1985) *Use of models and mock-ups in verifying man-machine interfaces.* In IEEE third Conference on Human Factors and Power Plants, pp.38-40).

Wiklund, M., Thurrott, C., Dumas, J. (1992) *Does the Fidelity of Software Prototypes Affect the Perception of Usability?* COMPUTER SYSTEMS: Usability and Rapid Prototyping Proceedings of the Human Factors Society 36th Annual Meeting v.1 p.399-403

Wilson, J., and Rosenberg, D. (1988) *Rapid prototyping for User Interface Design.* In Helander (1988), 859-875

Virzi, R., Sokolov, J., Karis, D. (1996) *Usability Problem Identification Using Both Low and High Fidelity Prototype* PAPERS: Evaluation, Proceedings of ACM CHI 96 Conference on Human Factors in Computing Systems v.1 p.236-243