LUCID Stage 3: Design

In LUCID Stage 3, the product's basic design, including overall navigation, screen layout and visual design and information or work-flow organization, is defined. The design is iterated through user and expert reviews until the team is satisfied that the design concept meets usability goals and is strong enough to be used for complete UI design specifications.

Goals

The key goal of Stage 3 is the development of the key screen prototype, which will be used as the basis for the interface design specifications in Stage Task 4. The key screen prototype demonstrates the software's basic navigation and 'look and feel,' as well as screen layout and critical screen objects. The key screen prototype typically includes the entry to the program; the basic home screen; major branches off the home screen; and examples of screen objects used to collect, manipulate or present data.

Prerequisites

LUCID Stages 1 and 2 have been completed.

Decisions

1. Will the product include a conceptual metaphor? While a conceptual model is required, a metaphor is optional.

2. Do you want to investigate several different design alternatives and/or 'looks' before creating the key screen prototype? Low fidelity paper prototypes can be used to compare several different approaches before making the selection for the key screen prototype.

3. What type of prototype will be created initially? At the end of this Stage? Three basic types are possible: low fidelity paper prototypes; medium fidelity on-screen prototypes; and high fidelity on-screen prototypes.

4. What methods will you use to obtain feedback on the prototype(s)? Possibilities include demonstration/user review, expert evaluation, and usability testing.

Tasks

Task 3.1 Develop the conceptual model and metaphor (if used)
Task 3.2 Develop and test the design concept
Task 3.3 Develop a low-fidelity prototype
Task 3.4 Conduct informal usability tests
Task 3.5 Develop a visual design
Task 3.6 Test the visual design (optional)
Task 3.7 Create the key screen prototype
Task 3.8 Usability test the key screen prototype
Task 3.9 Review the UI Roadmap, Product Concept and Requirements Analysis and revise them if necessary

Deliverables

The primary deliverable of LUCID Stage 3 is a key screen prototype that demonstrates the basic design of the software product.

The activities in Stage 3 focus on the development of the key screen prototype which documents the basic design concept and any design decisions which are made. The key screen prototype permits users, systems development staff, and other interested parties to react to the design. Typically, the prototype iterates from low to high fidelity, with user and expert reviews providing feedback between each iteration. What is critical is that the emerging design be expressed in a tangible form which represents the interface as the user will see it, rather than as a list of functional requirements or other abstract descriptions. At this point in the design process there may be two or more competing design alternatives. If so, it may be necessary to develop and evaluate several different design approaches. This is a decision based on budget, time, and the criticality of the application.

Types of prototypes

One of the decisions to be made during this stage concerns the choice of prototypes. Low-fidelity paper prototypes and medium- or high-fidelity on-screen prototypes are all possibilities.

Because they can be created quickly and easily, paper prototypes are especially useful when several different designs are being considered. Paper prototypes can be created either by hand, or by using a drawing program such as Visio.

Medium- or high-fidelity on-screen prototypes require more preparation time and resources, but are able give users a more accurate representation of the proposed ‘look and feel.’ However, this definition can also work against you by inhibiting suggestions and criticisms. Because they are more costly to produce, these prototypes are typically used when the design has been reasonably well defined.

A medium-fidelity on-screen prototype comprised of several screens will be able to demonstrate simple interactions (perhaps via a slide show approach), but will not be able to demonstrate complex interactions.

A high-fidelity on-screen prototype may be a program prototype. While they are more difficult and time-consuming to produce, program prototypes have the advantage of being able to stand alone, and are therefore easily evaluated by a large audience. This type of prototype is typically reserved for the key screen prototype. However, any code written for a prototype must be viewed as disposable; otherwise, the prototype will develop a 'life of its own' and lose its prototype status.

Review and evaluation methods

Several of the tasks in Stage 3 include 'Review and Evaluate' procedures. Review and evaluation can be performed in several ways:

- Demonstration/user walkthrough
- Expert evaluation
- Usability testing

Because they provide information that will be used to help form the product, these techniques are referred to as **formative** evaluations. (In contrast, evaluations performed on the finished product are **summative**; summative evaluation techniques are used in LUCID Stage 6.) A clear understanding of the goals for each review are important in selecting the correct technique.

Demonstration/User walk-through

In a walk-through, a user is guided through the interface by a demonstrator (normally a member of the design team). The focus of the walk-through may be an overview of the interface or a specific task-flow. Typically, this sort of demonstration is done early in the design process and is used to confirm assumptions made about user requirements or mental models.

In contrast to a usability test where the user is typically not prompted by the monitor, in a walk-through, the demonstrator may discuss specific design issues with the user, as well as encouraging the user to ask questions.

Expert evaluation

An expert evaluation is a review performed by usability specialists. Normally, two or three usability specialists participate, although it is possible (if less productive) for individuals to conduct expert reviews. Ideally, the specialists are not directly involved with the development of the product being reviewed.

There are several varieties of expert evaluations. Heuristic reviews relate problems in the user interface to design principles, while cognitive walk-throughs evaluate the interface based on assumptions about the user's mental model of the product.

In either case, the specialists use generally accepted usability principles to evaluate the user interface from the viewpoint of the targeted audience.

Usability testing

Broadly speaking, there are two types of usability tests: qualitative and quantitative.

> **Qualitative usability tests** are relatively informal. They are typically used as formative studies to guide the design process, and have the advantage of being relatively low-cost. The goal is to observe users working with the software and identify areas in which they have problems. Qualitative tests are not necessarily free form. The user is typically given specific tasks to accomplish, and performance is observed as the user attempts the tasks. One benefit of this is that only the paths to be tested need to be implemented in the software model used.
>
> Subject selection is often less precise than in the quantitative test, although this need not be the case. As a general rule, a representative sample should be chosen from the user population; however, the number of subjects may not be large enough to assure statistical validity.
>
> **Quantitative usability tests** are more rigorous than qualitative tests. They are used when several competing designs must be compared. This may relate to the overall approach of the software, but may also relate to specific elements which are not global but are considered critical to success of the project.
>
> For a quantitative usability test, subjects must be carefully selected from the target user groups. If the user group is segmented into different sub-groups, then each sub-group should be represented. In general, there should be at least six subjects for each cell in the quantitative test.

Although the details may vary with the specific test design, in both qualitative and quantitative studies, the subjects are provided with a task list and a working model of the system. A moderator (usually not a member of the usability team) acquaints each subject with the software and the tasks to be performed. The moderator then leaves the room , and the subject attempts to complete the tasks.

The subject is video taped; a time code generator is used to superimpose a time on each frame of the tape. Whenever a problem is observed by the usability team (who are watching through the one-way mirror or observing a video monitor), the time is noted. This permits specific frames to be located after the test. If the user becomes hopelessly confused, the moderator may return to the room and help out.

When the subject has completed the tasks, the moderator asks a series of questions to probe the subject's reactions and areas of confusion. The moderator may also ask the subject to complete one or more written scales which reflect their subjective reaction to the software. One widely used scale is the QUIS (Questionnaire of User Interface Satisfaction) developed at the University of Maryland. You can find more information about QUIS in the introduction to LUCID Stage 6.

Usability testing is a powerful tool. There is nothing quite so convincing to a usability engineer as watching a user successfully operate the software. However, despite its clear value, usability testing has three limitations which should be noted:

1. Subjects do not have time to cover all the options in a complex piece of software. Therefore the usability testing designer must make judgments about which elements to test. It is rare for more than a small section of the software to be tested.

2. The software is unfamiliar to the user. It is difficult to evaluate the usability of software until the user has had considerable experience with it. Thus, usability studies show whether the software design is comprehensible to a new user, but cannot tell you how users will fare once they become familiar with the product.

3. The usability test does not measure performance under actual job conditions. The effects of stress and workplace demands are not factored into the results. This can be critical in products designed for air traffic control, military, and medical applications. It can affect almost any application, however.

Task 3.1: Develop the conceptual model and identify any metaphor that will be used

At this transition from investigation to design, the team uses the requirements and task analysis from Stage 2 to develop a design concept and information model which will meet the goals laid out in the product concept.

Prerequisites The Stage 1 UI Roadmap and Stage 2 Requirements and Task Analysis have been completed.

Deliverables The primary deliverable of Task 3.1 is a description of the software's conceptual model which can be used as a basis for the development of the interface design. If any conceptual metaphors will be used, they should be described along with the conceptual model.

Overview The goal of Task 3.1 is the identification of the conceptual model that will be used for the product. This tasks iterates with Task 3.2 until a model is selected.

After the analysis of Stage 2 is complete, it can be tempting to dive directly into screen design. But, just as the product concept is the key to the overall software design, the conceptual design guides the interface development and must not be omitted.

Using a conceptual model has two advantages:

1. It provides a form of structural integrity for the design process. If the interface is modeled on a well-defined concept, the designers can test each element to see if it fits the model. In interfaces, as in the real world, objects are rarely used in isolation. A table setting includes plates, glasses, napkins and silverware. It would be reasonable to add a pair of chopsticks to the place setting, but really startling to place a hammer on the table. This is because a hammer violates our concept of tableware. In interactive designs, the conceptual model provides a test to see if objects and actions are consistent. For example, a trash can fits the office metaphor of many desktop programs, while a dump truck does not.

2. It assists the user in developing a mental map of the software. For example, many musicians now use computers to record and edit their material. Although the MIDI (Musical Instrument Digital Interface) and digital sound recording systems have little in common with traditional tape recorders, most of the software programs use an interface modeled on the controls of a tape recorder. Thus, the musician can think in the familiar terms of "rewind," "playback" and "record."

While it is possible to design elegant usable systems which have no special metaphor underlying them, all well-designed systems are based on a conceptual model that can be articulated,

reviewed and refined. Consider the difference in the interfaces which would be created from these two design metaphors:

- The interface will act as an "expert coach" guiding the user through the tasks.
- The interface will act as a central menu, allowing users to quickly select the tasks they need to complete.

Both will allow the user to complete a series of tasks, but the first implies an interface which uses wizards or embedded tutorials to sequence the users' actions while the second hints at a rich desktop a flat navigational structure.

The conceptual model vs. the technical model

Many interfaces fail because they do not distinguish the technical implementation model from the user's mental model and create a design based on the technical model. The technical implementation model is the software architecture of the program -- data structures and storage mechanisms, interface widget mechanics, etc.

One of the most common errors in interface design is to allow this model to "bleed" into the interface even when it is not part of the user's mental model. The goal of the interface design should always be to follow the user task model as closely as possible.

For example, many database programs distinguish between retrieving a record for *display* and for *editing*. The interfaces for these programs often require the user to take an action in order to edit a record displayed on the screen. This can be confusing to users who do not understand the underlying technical difference between the two modes.

When the interface uses a visual metaphor drawn from physical objects, such as a name-and-address database which uses a pocket organizer page as a metaphor, the problem is aggravated. In this case, the user is confused both by the technical task model and by the difference between this model and those in the physical world the metaphor is based on. For example, compare the tasks in reading and changing an entry in an address book in a "real" paper book, the user task model and the technical model:

Paper Organizer Tasks	User Task Model	Typical Technical Model
1. Open to correct page and read entry 2. Make a change or addition using pen or pencil. Old information can be erased or crossed out. 3. Close book. Any changes are permanent, and will be there the next time the entry is	1. Open to correct entry and read it. 2. Make changes or additions as desired, seeing the changes on the screen. 3. Switch to new entry or task. Changes are saved, and will be there the next time the entry is read.	1. Retrieve selected record and display it for reading. 2. Change to edit mode (refresh fields, and lock record for other editors). 3. Collect changes in the edit buffer. 4. Write changes to the record in order to save them.

read.		

The goal of the conceptual model and design metaphor is to bring the interface as close as possible to the mental model of the user. As the model is developed, it should be tested against task scenarios developed in Stage 2 to ensure that it is a good match.

Procedure

Step 1

Working as a group, brainstorm possible conceptual models for the software product. Be careful not to discard any ideas pre-maturely, and to have a way to capture and record all ideas.

Step 2

Review the list of models and metaphors generated in Step 1, looking for those that best match the user model. Discard any that are inappropriate, and select one model from those that remain for further development.

Step 3

Expand the conceptual model (and optional metaphor) using the Requirements Analysis from Stage 2. Make sure both the model and metaphor are able to accommodate all of the software requirements. If not, repeat the process using another model and/or metaphor.

Examples

Hewlett-Packard decided to include a online hypertext user guide with its LaserJet 4 printer. During the development of the LaserJet 4 Travel, the design team worked to find a simple navigational organization for all of the information about this new printer. The Guide would include not only instructions for using the printer, but tutorials using fonts and guides on paper selection. Many of the potential users were new to hypertext (in 1991, the Web had not yet made linking a common function); others were being introduced to advanced printer functions (such as type selection) for the first time. The team developed and tested three different conceptual models:

- The first used the metaphor of an airport (a play on the product name) as the conceptual organization. It was discarded early because it was difficult to fit all of the information and functionality into a navigational structure within the metaphor. It also proved impossible to develop the metaphor into a visual design which would translate into the many different languages and cultures required for this product.

- The second attempted to organize all of the information under a top level menu with only three items: Features, Use and Trouble-shooting. Early usability tests with screen mockups showed that users did not approach the manual with this model, which divided *what features* the printer offered from *how to use* those features. However, the usability tests did show that users liked having a single section with all problem-solving information.

- The final model divided the information functionally into four groups: information about the printer, controlling the printer, type, and printing tasks. A fifth group held problem-solving assistance. For each group, a descriptive menu identified 4-5 topics. In usability tests, users were able to quickly locate information on a topic, even when they were not sure of the correct term.

Task 3.2: Develop and test the design concept

The conceptual model selected in Task 3.1 is further developed into a design. A simple screen sketch, navigational diagram or description is produced and concept used to get preliminary user feedback. If there are several competing models, they might be developed and tested in parallel, or a several models might be tested in sequence in a rapid iteration of Tasks 3.1 and Task 3.2.

Prerequisites A conceptual model (and metaphor, if used) has been selected for further development.

Deliverables The primary deliverable of Task 3.2 is the results of user feedback on the design concept. A secondary deliverable, required to elicit the feedback is the sketch and description of the design concept.

Overview The goal of Task 3.2 is to rapidly test the design concept before further development takes place This tasks iterates with Task 3.1 until a model is selected.

The goal of this task is to get early feedback on the design concept before further development takes place. For each design project, the team must decide who should be included in this review, and how the concept will be presented. It can be difficult to find the right balance between too widely circulating a nascent idea and failing to get critical feedback at this early stage.

The format of the concept description will vary depending on the project. The design team must be aware of when they are proposing a design concept which might be controversial in some way – for example, by requiring new forms of user interaction, a new implementation technique, or a change in business process – and should be reviewed more carefully.

The description should also describe how this design concept will meet the goals of the product concept developed in Stage 1.

In almost all circumstances, the description of the concept should be accompanied by sketches which illustrate key aspects of the proposed design and how it will be used. These sketches are important because they are the first step in visualizing the finished design. They force the design team to *show*, rather than *explain* their ideas.

These sketches should not imply a finished design, and should be presented in a way that invites comment. One technique is to create “thumbnail” screen sketches which show only the key elements on a screen, rather than attempting to create a complete screen layout and visual design.

Design concept presentations might include:

- Short scenarios describing how users will accomplish key tasks
- A series of screen thumbnail sketches illustrating the concept in action
- An overview of the navigation as an annotated flow chart
- A sketch of the visual elements which will support a design metaphor

Procedure

Step 1

Create a description of the design concept which includes an explanation of how it supports users in meeting the goals for the product. Create screen or navigational sketches to illustrate the concept.

Step 2

Test the design concept with the extended design team, key decision-makers and users.

Step 3

Review the feedback from concept tests and decide whether to proceed with this concept or repeat Step 3.1 to develop a new concept.

If two concepts are being developed in parallel, decide which will be used. If elements of each will be merged, decide whether to repeat this task with the merged concept or proceed to the next.

Task 3.3: Develop a low-fidelity prototype

In this task, the design concept selected in task 3.2 is developed into a paper prototype. Paper prototypes allow easy distribution and rapid iteration of the design. They also allow the visual design to be developed in parallel as the screen layout requirements and navigational structure are fleshed out.

Prerequisites A design concept has been selected.

Deliverables The primary deliverable of Task 3.3 is a paper prototype

Overview The goal of Task 3.3 is to start designing the screens, deciding what elements go on each screen and beginning to create screen layouts. This prototype can be rapidly iterated to reflect feedback from users and other participants.

In this task, each design identified in Task 3.2 is represented by a paper prototype. Paper prototypes have several advantages at this stage in the development of an interface design:

- They invite comment in a way that a more finished presentation might not.
- They can be rapidly iterated in response to feedback.
- The design team may be more open to suggestions because the time (and emotional) investment have been minimized.
- They don't create unrealistic expectations for delivery timelines because the presentation makes clear that they are preliminary designs.

Paper prototypes can take several forms, and can be created by hand, or using a drawing program such as Visio. They generally do not try to create an exact picture of the screen, but simply indicate screen objects in outlined shapes. Whatever technique is chosen, it should support rapid changes.

Paper prototypes can include:

- Flow chart-style diagram of the overall system interaction
- Overall screen layouts showing relationship of elements such as banners, menus, buttons, etc.
- Sketches for each screen (or screen template) showing screen elements in detail
- Step-by-step thumbnails showing interaction details for unusual screen objects

Paper prototypes can also be an excellent way to collect and circulate on-screen text for approval. They also can be used to provide the software developers with an early version of the interface to inspect for technical requirements.

Procedure

Step 1

Sketch each screen or screen template showing overall screen layout and details of all objects on the screen.

Step 2

Circulate the prototype for comments, or use it as part of a team brainstorming session to refine the design.

Step 3

Revise the prototype based on feedback, and iterate Step 2.

Step 4

Identify any areas of the interface where user input is needed to make appropriate design choices, and identify any other questions that can be answered with a user review or other informal usability techniques. These issues will be addressed in Task 3.4

Task 3.4: Conduct informal usability Tests

Using the paper prototype completed in Task 3.3, informal usability tests or other user review is conducted. These reviews may be used to solve specific problems which have come up during the creation of the paper prototype, to confirm that the over-all navigation matches user expectations, or to refine key details such as menu organization.

Prerequisites A paper prototype has been completed, and design questions requiring user input have been identified

Deliverables The primary deliverable of Task 3.4 is a report on the user feedback required to answer any open design questions.

Overview The goal of Task 3.4 is to obtain user feedback required to solve any design problems which have arisen during the creation of the paper prototype.

Inevitably, during the initial design work in Task 3.3, there will be decisions for which the initial user investigations and requirements analysis provide no clear answers and where design heuristics do not provide sufficient guidance for the specific context. Informal usability techniques, using the paper prototype, can be used to collect the user input needed to answer these questions.

In addition, the paper prototype is the first design work which shows the whole scope of the program and is therefore ideal to use in a user walk-through to test assumptions about the users' task model which have been made in the design.

These tests do not have to be lengthy or exhaustive. Current practice in the industry suggests that 3-5 users may be sufficient for some tests. Because these tests can be quickly and cheaply conducted, they can also be done frequently or even on an ad-hoc basis. What is important is that user input is included in some form at this stage in the design to confirm that the work done matches user requirements and expectations.

There are many techniques for informal usability testing, some of general use and others which are useful for specific types of user investigation.

- Card sorting techniques work well to collect input on menu item or task groupings, and to see whether users can correctly match a functional element in the software to the menu item used to access it.

- User walk-throughs can be conducted using paper prototypes by asking users to point to the controls they would have selected on-screen.

Procedure

Step 1

Determine the correct technique to collect user input for each of the open design issues and design the test.

Step 2

Conduct user tests

Step 3

Document test results, and revise the paper prototype to reflect suggested changes. If the results show a need for substantial change – or even a re-evaluation of the design concept – repeat Task 3.3 or 3.2.

Task 3.5: Develop a visual design for the interface

In Task 3.5, several different approaches to creating a distinctive 'look and feel' for the interface are explored, and a visual design selected.

Prerequisites A design concept has been selected and any visual metaphor defined.

Deliverables The primary deliverable of Task 3.5 is a screen prototype of any 'look and feel' being considered for the product.

Overview The goal of Task 3.5 is to select a visual design for the interface prior to creating the key screen prototype.

The development of the visual design often happens in parallel with the initial design work done in creating the paper prototype, but it may also wait until the paper prototype is complete and some general screen layout designed.

The presentation of the look should include the visual style, icon and button style, color palette, fonts and text display. The graphic designer, working from screens in the paper prototype when available, creates sample screens for review and discussion. These screen should include sample content (greeking text can be used) or typical controls, so that any graphical elements can be seen in context.

- For an interface which will use a metaphor, key elements that support this metaphor must be shown.
- Any corporate logos should be included on the screen. If there are corporate identity guidelines or approval processes, they should be considered here, so that any required changes can be made early.
- For commercially released products, the marketing department may already be developing advertising material which needs to work with the interface design, and should be consulted. Brand logos or color schemes should also be considered.

The visual design should support screen readability, and should be reviewed on a target technical platform and under typical environmental conditions.

Accessibility must also be considered in the visual look – for example, will visual cues still be visible to the color-blind or on a monochrome screen.

Procedure

Step 1

Brainstorm possible visual styles, looking at sample designs from other interfaces, graphic sample books or sketches from the visual designer.

Step 2

Create sample screens for 2-3 of the styles selected from the brainstorming. These samples should show a typical screen in context. Graphic details can be incomplete as long as this does not interfere with an understanding of the style.

Step 3

Review the sample looks and either select one for further development or identify any user testing which will be done before selection.

Task 3.6: Test the visual design (optional)

In this task, the 'look and feel' selected in Task 3.5 is tested to collect user feedback. This feedback might test specific elements such as user comprehension of icons, or might be a general "likeability" test for the visual style. The tests can be comparative, looking at two or more solutions to a specific problem.

Prerequisites Sample visual designs have been completed in Task 3.5 and any areas for user feedback identified.

Deliverables The primary deliverable of Task 3.6 is a single visual design for the product.

Overview The goal of Task 3.6 is to collect any user feedback necessary to finalize the selection of the visual design.

For some issues requiring user input the visual design, like the paper prototype, can be tested with informal usability techniques. For others, especially user acceptance of a visual style, more formal testing techniques are required. For commercial software, the marketing department may wish to be involved in this testing and may use traditional market research techniques.

Procedure

Step 1

Prepare a test plan to meet the objectives for user testing of the visual design.

Step 2

Make any changes or additions to the sample screens from Task 3.5 required to support the test plan

Step 3

Conduct the tests

Step 4

Evaluate and report on the test results, and make changes to the visual design as suggested by the user feedback. If necessary, repeat Task 3.5, creating a new visual design for review and testing.

Examples

In a recent project to redesign a bank's automated teller machines, we developed six different preliminary 'looks.' To aid the selection we conducted an *intercept test* at three of the bank's branches. Three branches in diverse demographic locations were identified (inner city, business district, suburban). Customers visiting the branches were shown each design and asked which they preferred. Comments were also recorded and analyzed. On the basis of customer reaction, one 'look and feel' was selected for further development.

Task 3.8: Create the key screen prototype

In Task 3.8, the key screen prototype is created using the design concept, paper prototype and visual design created in earlier Stage 3 tasks. The key screen prototype typically includes the system logon or start-up screen; the basic home screen; major branches off the home screen; and examples of screen objects used to collect, manipulate or present data.

Prerequisites Tasks 3.1-3.7 have been completed.

Deliverables The primary deliverable of Task 3.8 is the key screen prototype, the model that demonstrates the basic design of the software product.

Overview The goal of Task 3.8 is to bring together the major screen elements identified in Stage 3 to create the key screen prototype.

In this task the major deliverable for Stage 3—the key screen prototype—is developed. A key screen prototype is like an architectural model. It does not show detail and, in fact, may not be functional at all. But it does show the major elements of the system—screens, screen objects and navigation. The key screen prototype permits users, systems development staff, managers and other interested parties to react to the design.

It is not necessary that the key screen prototype be fully functional software – in fact, the less detailed functionality included in the prototype the better. As with the paper prototype, the design team should be able to make rapid changes in the design and remain open to suggestions as the prototype is iterated.

Often the elements in a key screen prototype are carefully constructed to walk through the most typical user scenarios. Not every interface design problem needs to be completely solved in this prototype, but there should be an indication of the solution – and no critical design elements should be ignored.

Key screen prototypes can be created in many different ways:

- Simple slide shows sequencing from one static screen to another. These are the least satisfactory as they do not include possibilities for user interaction during a test, but can work as a preliminary stage.

- Enhanced slide shows created in a program like HTML or Director which allow some user interaction, especially high level navigation.

- Simple application prototypes, created in a rapid development environment like Visual Basic or Delphi. These allow a high fidelity screen representation for a GUI interface and can have a minimal level of interaction programmed with little difficulty.

- Web applications can be prototyped directly in HTML, using a screen builder such as NetObjects Fusion, SiteMill or Front Page.

- Detailed animation or programming prototypes may be created to show the specific interaction for unusual or critical user interaction or data collection devices

As you decide what kind of key screen prototype to create, consider not only the design and usability goals, but what kind of information other groups (marketing, programming, business units, etc.) may be seeking from this prototype.

One frequent mistake made in prototyping, especially when it is created in the application development environment, is to assume that this prototype is the beginning of the programming work. These prototypes should be "throw away" work that can be revised freely without worrying about the underlying software architecture. If it proves re-usable, great….but the design team must be free to make changes without worrying about interfering with the programming effort. The purpose of the prototype is to *illustrate* the design, not kick start development!

Production of the key screen prototype is an important milestone in the usability engineering process. It is the dividing line between the initial design exploration and the intensive design and refinement stage.

Procedure

Step 1

Decide what elements or tasks in the software will be illustrated in the key screen prototype. Create paper prototypes for any screens not included in the work of Task 3.3.

Step 2

Create any visual design elements required for these screens.

Step 3

Construct the screens, including sample material as required. Program any user interaction the prototype will support.

Step 4

Review the prototype with the design team, making any changes to refine it. If the team is large enough, members who did not work directly on the prototype can conduct expert reviews to correct obvious flaws before the prototype is usability tested.

Task 3.8: Usability test the key screen prototype

The creation of the key screen prototype is a major milestone in the development of the interface design. No matter how informal or extensive any previous user testing has been, the testing in this task is critical to the success of the interface.

Prerequisites The key screen prototype has been completed and refined by the design team.

Deliverables The primary deliverable of Task 3.9 is the usability test results of the key screen prototype and a refined version

Overview The goal of Task 3.9 is to refine the key screen prototype by correcting any major design flaws found in user testing.

Before the key screen prototype begins the intense iterative refinement process in LUCID Stage 4, it should be quickly evaluated for major flaws.

Procedure

Step 1

Use one or more of the techniques described in the Introduction to Stage 3 to evaluate the key screen prototype.

Step 2

Review the results of the usability testing and brainstorm design solutions to the design flaws.

Step 3

Refine the key screen prototype as required by the results of the evaluation process.

Task 3.9: Review the UI Roadmap, Product Concept and Requirements Analysis and revise them if necessary

Before proceeding with the detailed design refinement, the key design documents – UI Roadmap, Product Concept and Requirements Analysis are reviewed. Often discoveries during the initial design stage may suggest changes in scope or direction. This task is an opportunity to formalize those decisions which have been made on an ad-hoc basis.

Prerequisites The key screen prototype has been completed, evaluated through usability testing and refined.

Deliverables The primary deliverable of Task 3.10 is a revised or re-validated version of the UI Roadmap, Product Concept and Requirements Analysis.

Overview The goal of Task 3.10 is to make sure the these fundamental documents are still accurate and appropriate.

As a result of the initial review and evaluation of the key screen prototype in Task 3.9, you may decide that parts of the UI Roadmap, Product Concept and Requirements Analysis need to be revised. If revision is required, it should be done before moving on to LUCID Stage 4.

Procedure

Step 1

Review the Requirements Analysis generated in LUCID Stage 2. Watch for areas in need of revision as a result of Stage 3 activities and investigations.

Step 2

Revise the Requirements Analysis as needed.