GUI Design Methodology

December 18, 1996

1. Introduction

The BellSouth GUI Design Methodology provides a complete process and set of job aids/tools for building a Graphical User Interface in any environment and for any size project.

The primary components of this methodology include:

- Recommend roles, responsibilities and skill set requirements for the GUI development/design team
- Define the User Profile of the Primary and Secondary user groups
- Define Usability Objectives for the product
- Implement iterative User Centered Design
- Implement Object Oriented GUI design principles
- Implement BellSouth and Industry standards/conventions for design
- Perform Usability testing to verify design and initial objectives for usability

The objectives of this GUI Design Methodology are as follows:

- Reduce the amount of time needed to design and build a GUI
- Enhance GUI flexibility, functionality, and usability by implementing the OO GUI design framework. The OO GUI design facilitates adding or changing functionality and GUI components to better respond to changes in the business. Design and code reuse can also be realized within and between related projects.
- Deliver *user centered*, not *data or process centered* GUI designs that meet user expectations for usability and functionality.
- Involve users from start to finish to enhance acceptance and buy-in to your design. Users are active members of the design team.

This methodology is supported by IT-Architecture and Standards. The following products are available from A&S to all projects for use in GUI design:

- *BellSouth GUI Style Guide V4.1*
- User Centered Design training and design workshop
- GUI Requirements training and template
- *Usability Planning Guide*

All BellSouth projects should implement this GUI design process in order to ensure a complete and usable GUI design. This methodology should also minimize the design and development cycle for the GUI.

Below are the BellSouth contacts for this GUI Design Methodology:

Jim Berney
675 W. Peachtree Street
13L71
Atlanta, Georgia
404-529-0346

Tara Powell
1876 Data Drive
S-304
Birmingham, Alabama
205-988-6805

2. Organization - GUI Design Team

This section describes the recommended GUI Design Team roles and skill sets. Whereas large projects should staff each position, smaller projects may combine roles as appropriate. It is essential that key team members have the necessary skills and experience upfront.

2.1 Roles/Skills within the GUI Design Team

CORE GUI Design Team - The CORE GUI Team has complete responsibility for GUI design, development, and maintenance throughout the life of the application. This team is composed of the Chief GUI Architect, GUI Design Team leads, Human Factors/Usability Engineer(s), key user SME's, and management responsible for user interface design. The CORE Team must be empowered to make all GUI design decisions. The turn-over rate within this team should be very low throughout the project.

CORE GUI Team should meet on a regular basis to:

- Review product design (prototypes and completed design) - review for style guide compliance, consistency with other windows, and adherence to design heuristics for usability.
- Review product and usability goals
- Review usability data - ensure that the GUI design meets usability goals and is acceptable to the user
- Ensure Style Guide (BellSouth and Application) compliance by all design teams
- Ensure that each GUI component is smoothly integrated into the overall GUI design and plan for navigation.
- Make routine GUI design decisions
- Update the Application User Interface Handbook

During initial project phases, the CORE GUI team should meet frequently to establish standards, direction, and develop early prototype design. As GUI design moves into implementation, the CORE team will meet less frequently.

CORE GUI Team Leader - Responsible for overall coordination of the GUI Design team, scheduling of design and review meetings, creating and updating the Application User Interface Handbook and communicating all design changes/standards/conventions to all design teams and participants. The CORE team leader is an active participant in each design team and solicits problems and coordinates resolution to all design and style guide conformance issues.

The CORE Team Leader should be the librarian for the GUI Requirements document and all UCD workshop artifacts (task flows, task object descriptions, window prototypes). The CORE Team leader should work with the Chief GUI Architect and each GUI Design Team Lead to update and maintain the GUI Requirements document. As this document is updated, it should be re-distributed to all GUI design team members.

As librarian, the CORE Team Leader should:

- Maintain the *GUI Requirements document* that contains a complete specification for GUI design
- Maintain *Task Flow documentation* (index cards, task flow-charts) developed in the UCD workshops. Document task flow changes resulting from follow-up discussions or subsequent UCD workshops. Modify and/or create new Task Flows as driven by iterative design
- Maintain *GUI/Task Object descriptions* (index card, stickies) developed in the UCD workshops. As requirements mature and as Object Views are added or modified, the GUI/Task Object descriptions must be reviewed and updated accordingly.
- Maintain *Primary and Secondary Window designs* developed in the UCD Workshop and refined within each GUI Design Team. Throughout design and construction, it is critical that the GUI Requirements document and Window prototypes be updated with design enhancements so that the changes can be effectively communicated to all design teams.
- Maintain any supplemental documentation from UCD workshops

CORE Team Representative from the Requirements Group

Iterative design and testing will always introduce changes and insertions into requirements that must be carefuly coordinated and documented. Large projects have formal processes for modifying requirements once “baselined” and changes to requirements are often frozen to meet product delivery dates.

The team responsible for building, documenting, and maintaining formal application requirements should have a representative on the CORE GUI Design Team. This representative is responsible for initiating change requests to the formal requirements baseline as a result of GUI design enhancements/changes.

Whereas the UCD workshops do not identify every detailed Task Flow of the application, the workshops do identify the principle flows that are the basis for GUI design. These flows are also critical to ensure design completeness and for GUI testing and should supplement scenarios typically developed for usability testing. Thus, the Requirements Representative and the Usability Engineer should communicate frequently.

Work with each GUI Design group to ensure that they understand and concur with the Task Flows that pertain to their group. Note that some Task Flows may pertain to multiple GUI Design groups. Throughout the design, the Requirements Representative should work closely with each GUI Design Team to identify changes and extensions to the Task Flows. Any Task Flow update must be communicated to all affected Design Teams. The Task Flow documentation should also be updated routinely to communicate the latest design baseline.

CORE Team Representative from Training & Documentation

Representatives from the organizations responsible for providing training and documentation to the end-user should be a member of the CORE GUI team. Not only may these individuals provide valuable input into your design, they should be involved early so that they can begin planning to implement appropriate packages and delivery mechanisms for training and documentation. In fact, these representatives should plan parallel sessions to gather requirements and specifications for training and documentation/help.

On-line help and training packages may be tested for usability in parallel to the application. Training and documentation should complement and help achieve the usability goals of your application.

Management Representative

The manager responsible for GUI development and the management representative from the owner/user organization responsible for the project should to permanent members of the GUI design team. Whereas they may not participate in every activity and meeting, they should have the following roles:

- maintain a current knowledge of the issues, the GUI design, and all project schedules
- provide direction to the team
- obtain resources as needed
- provide access to users

Chief GUI Design Architect

Responsible for overall GUI design and all coordination between the GUI design teams (Theme Teams). The Chief GUI Architect acts as a mentor for each GUI design team and should have extensive experience in designing and building GUI applications.

The Chief GUI Architect is responsible for the following:

- Identify and design any application *Metaphor*
- Ensure that the overall *GUI navigation model* meets the user's task flow requirements and that this model is used consistently across design and GUI design teams.
- As GUI Objects and GUI Object Views are added or modified, the GUI/Task Object descriptions must be reviewed and updated accordingly. You must identify and document the impact of these changes on other GUI Objects and Views in the design. You must also ensure that the new or modified GUI objects are consistent with documented Task/GUI Object descriptions and that the established heirarchy and relationships between these Objects is maintained.
- Ensure that all changes to Primary and Secondary Window are effectively communicated to all GUI design teams.
- Ensure that all window designs are reviewed to ensure style guide consistency.
- Work with Graphic Artists to effectively integrate graphic art into the design

Human Factors/Usability Engineer

The Usability Engineer uses Task Flows identified in the UCD Workshops and Usage Scenarios to test usability and verify product and usability objectives of the prototype and fully functional GUI design. Usage scenarios are developed to test specific aspects of the interface in actual use by a typical user.

The Usability Engineer should notify the CORE Team Leader, GUI Team Design Lead, and Requirements Representative of all usability problems identified within a Task Flow.

The GUI Design Team will need to work with the Requirements Group to assess and resolve the usability problem. Refer to the original UCD workshop documentation to ensure that no information was lost or misinterpreted by the GUI Design Team.

Usability Coordinator

Representative from the User Organization responsible for the following:

- Schedule routine informal usability evaluations (refer to the Usability Planning Guide for detailed information of planning and executing informal usability evaluations.)
- Schedule and coordinate formal usability evaluations after key deliverable (refer to the Usability Planning Guide for detailed information of planning and executing formal usability evaluations.)
- Select users for the evaluations and coordinate transportation
- Help analyse usability results
- Work with the Usability Engineer to summarize and communicate usability results to the GUI Design Teams

GUI Design Team

Responsible for building the application GUI.

Each major component of the application is represented by a Design Team. This is an effort to break down the development process into smaller manageable components. Each Design Team is responsible for design and development of their component. The lead GUI analyst from each Design Team should be a member of the CORE GUI design team. Each Design Team must ensure that all GUI components are integrated together so that the navigation and hierarchy of the GUI is natural to the user. Each Design Team must adhere to application style standards and must integrate reusable GUI dialogs and components where appropriate.

A large project may have multiple GUI design teams (Theme Teams) that each design a component of the GUI.

- A GUI Design team member should participate in all UCD workshops that pertains to their GUI component. It is recommended that the same team member participate in all UCD workshops.
- As the GUI Design Team builds and modifies the interface, they should verify their design with the Task Flows and original window prototypes developed in the UCD workshops. Informal task walk-throughs will help identify design gaps, navigational problems, and missing information/functionality needed to complete the task.

GUI Design Team Lead - Each GUI Design Team has a lead analyst that is the spokesperson and design lead for that team. The Team lead should be the most experienced member of the design team and should have expertise in the development/production environment. The GUI Design Team Lead has the following responsibilities:

- GUI Team Lead is the representative on the CORE GUI Team. In larger projects where there are multiple GUI design teams, each team lead should be on the CORE GUI Design team.
- Track and champion the team's GUI design throughout development
- Update the GUI Requirements document
- Argue for features critical to meeting usability/product objective and future enhancements
- Proactively exploit easily supported functionality requested by the user
- Ensure GUI design consistency between all GUI design teams
- Ensure BST and application style guide consistency

GUI Design/Coding Analysts - Responsible for implementing and constructing the GUI design prototype and production product. Each GUI Design Team is composed of one or more GUI analysts. These individuals must have expertise in the development platform and production environment (e.g., MS Windows, OSF Motif).

3. GUI Development & Design by System Development Lifecyle (SDLC) Phase

3.1 Planning Phase

During the Planning Phase, you should perform the following key tasks and provide the following documentation deliverables for the User Interface component:

1. *User Profile Definition* - Begin to identify individual user characteristics (Know thy user.....) of all primary and secondary users of your application. The User Profile should be finalized in the Analysis Phase.
2. Identify *Usability Goals* (benchmarks) for design that can be measured and validated in usability testing
3. Diagram the user's *current and desired task flow (high level)* - Determine at a high level what the users expect the system to do (capture user, task, and job needs). Describe each major task.
4. *GUI Team organization* (roles and responsibilities) - Establish a GUI Development Team and a model for design and development.
5. *Post Deployment Impact Analysis* - Project the evolution of the user and the job and the expected organizational impact of your application into requirements

The following individuals/roles should activitely participate in the Planning Phase:

1. User Subject Matter Experts (SME's) - must be knowledgeable of the business domain
2. GUI Analysts - should have knowledge of the production environment and business domain
3. Facilitators and experts familiar with the chosen methodology for performing task analysis (e.g., use cases)
4. Analysts skilled at working with User SMEs to gather requirements
5. Usability Engineer to assist in defining usability objectives and defining the user profile
6. Project and/or Product Manager accountable for the project and its implementation
7. Representatives from Training

3.1.1 Define the GUI Design Team

Refer to section 2 of this document for details on the GUI Design team.

3.1.2 Define the User Profile

Know Thy User! - Examine the knowledge, background, experience, objectives, and motivations of your primary and secondary user groups. This information may be collected using a combination of surveys, interviews, and on-site visits. A high level view of the User Profile should be delivered in the Planning Phase. The final User Profile containing adequate detail on each user category should be delivered in the Analysis Phase and is used as input into GUI requirements.

Refer to the BellSouth GUI Style Guide section 2.2 and the Usability Planning Guide section 6 for more information and job aids for creating and documenting the User Profile.

The profile should identify each *user category* and provide a *profile summary* of all user types within the application's customer base.

- List all user categories (primary and secondary users)
- Describe each user category (who makes up category, job/task being performed, user objectives/motivations, constraints)
- Provide an example *snapshots* of a real user in each category. Includes video and an actual interview with the user.
- Provide the User Profile data to the Requirements and GUI Design Team via summary reports and verbal presentations

3.1.3 Identify Usability Goals

Usability goals are *non-functional requirements* used within the context of GUI design and should be part of the project requirements. These goals must be objective, support the product goals, be measurable, and be testable. Quantitative usability goals are measured during usability evaluations in order to help determine when GUI design is complete or where additional work is needed.

The deliverable is a set of Usability Goals. Define high level usability goals during the Planning Phase. Refine these goals and provide additional detail during the Analysis and Design Phases. Each goal should include:

1. Usability Attribute (e.g., ease-of-use, ease-of-learning, task performance, support of product goals)
2. Usability Measurement - Examples of usability measurements include "time to complete a task", "number of errors".
3. Usability Criteria (e.g., 85% of users can complete the task within 2 minutes with no errors) - range of values assigned to each usability measurement
   - worse than expected - typically the "expected" criteria for an existing or competing product that your product will replace
   - expected - value generally accepted by most users and product designers as the design target
   - better than expected - typically a target value for the next version of your product.

Attr. Ref. Index	Usability Attribute	Measurement (define unit)	Usability Criteria			Comments
			Less Than	Expected	Better Than	
101	Find target information in on-line help	seconds	20 or more	10	8 or less	

Refer to section 8 of the Usability Planning Guide for information and job aids for defining Usability Goals.

3.1.4 Define the High Level Task Flow

Part 1 of the User Centered Design workshop should be performed to build an overall High Level Task Flow of your application. Whereas a UCD workshop will be scheduled to complete the design of each component of the GUI, the first step is to perform a workshop to define the high level task flow, major GUI components, high level navigation between components, and integration with the user desktop. During this initial workshop you will not design the GUI detail or narrow your scope to individual components of the GUI/application.

The following High Level Task flows should be defined:

- *Need Analysis* - Define the user's needs, the big picture of the work flow (should be in the context in which the subject task scope is performed) - what is the job or task performed?
- *Current Task Flow* - Identify how the task is currently performed.
  - Identify issues & bottlenecks in the current task flow. What needs to be fixed?
- *Blue Sky Task Flow* - Using the Current Task Flow and the issues/bottlenecks identified, design an ideal task flow, even if it is unrealistic
  - Priorities & constraints: Label Blue Sky task flow with user priorities and developer constraints. What's important to the user vs. what can be built and delivered?
- *Realistic Task Flow* - Modify Blue Sky task flow so it is realistically attainable (both the user and the developers must agree on the task flow). The Realistic Task Flow becomes the basis for your design. At a high level include inputs (triggers) into the task flow, process (what happens to the input), and what is produced (output). The Realistic Task Flow is in the form of task flow diagrams composed of index cards. Each card represents an individual task (data object and action). The cards are laid out on a big piece of paper, on which lines and labels are drawn to show the cards' relations to each other.

3.1.5 Post Deployment Impact Analysis

Assess the impact of your design on the user and user organization and user goals and objectives. The results of this assessment should be inputed into project's Change Management process and plans.

3.1.6 Object Oriented Design Process & Deliverables

The OO design process documented in this methodology is based upon the use case methodology and the Object Modeling Technique (OMT). Refer to the OO Methodology Handbook for more details.

The High Level Object Model (business terms only, no implementation detail) is produced during the Planning Phase:

1. Object definition
2. Environment description

4. Analysis Phase

The Analysis Phase produces the following deliverables:
1. Set of *baselined* project requirements.
2. System Architecture must also be defined that describes the application layers, the components of each layer, and the hardware environment.
3. Approved project contract that contains a good estimate of costs must also be produced.

The Analysis Phase is typically performed by Analysts skilled with in object modeling and data modeling techniques. User SMEs ensure that the Analysts maintain an accurate understanding of the requirements and business.

During the Analysis Phase, the following key tasks are performed and the following documentation deliverables are provided:
1. *Usability Objectives* - Finalize Usability objectives for the interface.
2. *Usability Testing Plan & Schedule* - Establish user interface design reviews, a plan for executing usability testing, and a schedule for usability testing (Informal and Formal Usability Testing)
3. *Usability Testing Scenarios* - Build initial Usability testing scenarios
4. Build and/or acquire custom GUI controls if needed for the development/production environment.
5. *Application User Interface Handbook* - Begin development of the Application User Interface Handbook.
6. Specify GUI design constraints (e.g., performance, security, reliability, portability, maintainability)
7. Select GUI design tools
8. *Detailed Task Flow Definitions* - Use Part 1 of the UCD workshop to define detailed task flows for all components of the application. Refine the High Level Task Flow documented in the Planning Phase as necessary. Document how the user will use the application to perform their job. The task flows should describe work inputs (triggers), process (what happens once input is received), and output (what is produced). Details of the Window design do not begin until Part 3 of the UCD Workshop. Concentrate on Task Flow analysis rather than design and construction. Ensure that you do not limit creativity in design or restrict the number of possible solutions to a problem.
9. *Task/GUI Object Definition* - Use Part 2 of the UCD Workshop to define the GUI objects (attributes, methods, containment, relationships between GUI objects).
10. *GUI Window Prototypes based on Task/GUI Objects* - Use Part 3 of the UCD Workshop to evolve the task/GUI objects into Primary and Secondary windows. Paper and pencil prototypes are produced for each window. Integration between the GUI and User Desktop is also defined.

11. *GUI Requirements Documentation* - Begin formal documentation of GUI Requirements using the template provided by IT-A&S.
12. Define relationships within the requirements specification
13. Perform low-fidelity prototype *Usability testing* of the user interface
14. Conduct scheduled *GUI design reviews* to ensure standards compliance and consistency of the user interface

The following individuals/roles participate in Analysis Phase:

1. User Subject Matter Experts (SME's) - must be knowledgeable of the business domain
2. GUI Analysts - should have knowledge of the development/production environment and business domain
3. Facilitators and experts familiar with the methodology (e.g., user case methodology) used to gather and document requirements.
4. Analysts skilled at working with User SMEs to gather requirements
5. Human Factors/Usability Engineer
6. Project and/or Product Manager accountable for the project and its implementation
7. Analysts responsible for database design and architecture
8. Representatives from the group responsible for installation and deployment
9. Representatives from Training
10. Representatives from Infrastructure groups (e.g., security, network, hardware)
11. Representatives from the Testing group

4.1 Adhere to GUI Standards

All BST GUI design should adhere to Industry and BST standards and conventions. Style guides should be common knowledge to the GUI design team.

- BellSouth GUI Style Guide V4.1
- Industry Style Guides for Windows® and/or Motif™
- Application User Interface Handbook

4.2 Select GUI Design Tools

Determine the GUI Design Tools that will be used in construction.

- Select the standard GUI design tool (e.g., MS Visual Basic, MS Visual C/C++, Visual Edge UIM/X for Motif)
- Select any add-on products to the GUI design tool (e.g., widgets not supported by the native tool, VB and VC++ add-ons, On-line help tool)
- Identify and build reusable GUI components (e.g., GUI controls, dialogs, windows, menu framework)

4.3 Detailed Task Flow Definition (UCD Workshop Part 1)

Define the task flow for each component of your application, including the high level task flow that defines the overall navigation between tasks and application components. You will define the steps and information needed by the user when performing the task/job that will be supported by your GUI.

This Task Flow Analysis is an interative, top-down process that is repeated until every component of your application GUI is defined. The initial UCD workshops to define your high level task flow and the task flow of your principle GUI components should be performed sequentially with the same participants. Once your high level components and component navigation is defined, the lower level sessions may be done in parallel by different SMEs.

This is a participatory exercise that draws upon the knowledge of users, designers, developers, and SMEs. Available Requirements documentation may also be used to help define task flows.

For each category of users defined in your User Profile and for each component of your application, identify the following:

- Diagram and describe the task flows. Identify which user group(s) (reference user profile) will perform the task. This diagram should capture the overall job that must be performed and the high-level tasks required to complete that job. Identify the following:
  - what triggers the task (task input)
  - where does the task fit in the overall job/task flow
  - tasks that precede, follow, or interrupt the task
  - task output (what is produced?)
- Briefly describe the each task (user perspective)
- Ensure that all information required to complete the task is available to the user
- Identify any additional documentation or tools required to complete the task and ensure that they are available to the user.
- Consider the impact of the following on the task flow and design the flows accordingly:
  - how frequently will the task be performed by the user group?
  - what are the task constraints?
  - are there any constraints in the physical environment where the task will be performed?
- Identify current problems in performing the task (user and system) and how your task flow will resolve these problems.

4.3.1 UCD Workshop Part 1: Task Flow Definition

Objective: Translate user needs for the task into requirements that reflect the task flow.

The following Task flows should be defined:

- *Need Analysis* - Define the user's needs, the big picture of the work flow (should be in the context in which the subject task scope is performed) - what is the job or task performed? What are the business needs for this task/job?
- *Current Task Flow* - Identify, in a moderate level of detail, the task steps the user currently does to do this particular work. Use arrows to indicate within the task flow the flow between task steps.
  - Identify issues & bottlenecks in the current task flow. What needs to be fixed? What is the priority of the fix or feature? Since this exercise may expand and complicate the scope of the UCD workshop, consider the following:
    - ♦ importance of the issues and bottlenecks
    - ♦ how much time is allocated for the workshop
    - ♦ whether the necessary SMEs and users are available and/or participating in the workshop
    - ♦ what parts of the workflow is the team authorized to work on
    - ♦ what resources (time, people) are available for design and development after the workshop.
- *Blue Sky Task Flow* - Using the Current Task Flow and the issues/bottlenecks identified, design an ideal task flow, even if it is unrealistic
  - Priorities & constraints: Label each step of the Blue Sky task flow with a user priority and developer constraint. What's important to the user vs. what can be built and delivered? Based on these decisions, the group will jointly decide what will be included in the final task flow.
- *Realistic Task Flow* - Modify the Blue Sky task flow so it is realistically attainable (both the user and the developers must agree on the task flow). The Realistic Task Flow becomes the basis for your design and is used as input into Part 2 of the UCD Workshop. At a high level include inputs (triggers) into the task flow, process (what happens to the input), and what is produced (output). The Realistic Task Flow is in the form of task flow diagrams composed of index cards. Each card represents an individual task (data object and action). The cards are laid out on a big piece of paper, on which lines and labels are drawn to show the cards' relations to each other.
- Usability Test your Realistic Task Flow to verify the completeness your design.

Output:

The output of the Task Flow definition are the user requirements for doing the task that your application GUI must support. You will produce a script of what users do, in the form of a task flow diagram composed of index cards. An index card is used to represent

each step, with removable sticky arrows between cards (steps), and the entire flow taped to a piece of flip chart paper.

Note:

- This stage of system development is independent of a specific GUI/platform environment; that is, it is the same regardless of whether the delivery platform is a GUI (Windows®, Motif™), a character user interface, a personal digital assistant, pieces of paper, or a new business process.

4.4 Define Task/GUI Objects (UCD Workshop Part 2)

Based on the Realistic Task Flow defined in Part 1 of the UCD Workshop, Part 2 defines the Task/GUI objects and the relationships between these objects. These Task/GUI objects are then used in Part 3 to build Primary and Secondary windows. Below are the steps towards defining Task/GUI objects:

1. Define the GUI Conceptual Design:
   - Identify Task/GUI Objects: Identify the GUI objects that make sense in terms of the user interface requirements for doing the task. GUI objects and object hierarchy should match the user's mental model of how they perform the job. GUI object names should be derived from the user task and should be described in the user's language.
     - GUI object description
     - define GUI object actions
     - define GUI object attributes
     - determine whether a UI metaphor will be used. If so, define the metaphor

     For example, each index card used in the task flow from Part 1 may be identified as having one data object (e.g., customer, bill, network element) and one action on that object (e.g., add customer, print bill). Also identify the attributes of those objects, the actions on those objects, and actions those objects take.

   - Identify Object relationships: Identify static relationships (peer and hierarchical relations),. The relationship between GUI objects in the design should match the way users think about the objects they use to do their job. The object hierarchy should match the user's mental model of their job . For example, a customer service rep may view a Customer to have a Line which has Network features. Line is a child object of Customer, and Network features is a child object of Line.
2. Represent Task/GUI Objects (index cards and stickies)
3. Document the GUI Conceptual Design in the GUI Requirements template
4. Review the GUI Conceptual Design (usability testing and formal design review). All design issues and changes in requirements should be incorporated back into the GUI Conceptual Design. This is an iterative process until the Conceptual Design model is correct.

4.4.1 UCD Workshop Part 2 - Task to GUI Object Mapping

Objective: Map task flow into task/GUI objects

Output:

The output of this stage is a set of task-related GUI objects represented as index cards and groups of index cards. Stickies that describe the GUI object, list object attributes (child objects, properties), object actions (e.g, create, delete, print, save), and object relationships (containment) are attached to each index card representing a task/GUI object. There will be several of these layouts: one describing the association of all objects, the other describing the hierarchical relations among objects. In addition, the attributes and relationships of the objects will be represented in tables.

Note:

- This stage of system development is independent of the particular interface; that is, it is the same regardless of whether the delivery platform is a graphical user interface, a character user interface, a personal digital assistant, pieces of paper, or a new business process.
- The focus is on defining GUI objects from the end-user's view, not from any object-oriented programming view; GUI objects are those that make sense in the context of the user requirements the task flow.
- Nor are the objects in this stage of system development the objects of the interface such as windows and icons.

4.5 Design the GUI Windows (UCD Workshop Part 3)

Based on the task objects defined in Part 2 of the UCD Workshop, the design team uses the methods in Part 3 to identify GUI objects and to design Primary and Secondary windows for each GUI object. This stage of system development is dependent on the particular type of the user interface.

The GUI model for representation, navigation, and interaction: Object orientation for GUI design

Below are steps in Part 3 of the UCD Workshop:

- Hardware and software constraints: Identify type of user interface for system product, for example, graphical versus character interface.
- Identify which Task Objects will become GUI Objects.
- Identify the type of each GUI object in your design.
- Identify GUI object actions that must be implemented
- Define GUI objects containment. For each GUI object...
  - ⇒ Identify all other GUI objects that it can be contained in (every Task Object should be contained in something, even if only the desktop)
  - ⇒ Create additional container & device objects to help organize GUI objects, using task object cards.
  - ⇒ Identify how each GUI object can appear when in the closed state
- Mapping objects to windows: Identify primary windows and dialog boxes.
- Identify window type (Primary or Secondary)
- Define the GUI Object Views to be displayed: Window views: Different ways of looking at an object's information, for example, list versus icon views
  - ⇒ For each GUI object, identify types of views available (structured, unstructured). Identify each view on the task object card. Design the principle control used to display data in the View (e.g., Notebook - tab control, Scrolled List, Text fields) in each window.
  - ⇒ Using window sheets, design the views for each GUI object that will be delivered to the user.
  - ⇒ Identify whether user will be able to create new views for a GUI object, and will be able to delete existing views. Mark this capability on the task object card.
- For each Window, define the menu pull-down structure and control buttons:
- Mapping object methods to menu bar controls: For example, deleting an object by using the Delete action

For all GUI objects viewed in primary windows:

⇒ Design System/Control menu, and Window menu (if appropriate)
⇒ Design menu pull-downs (follow industry and BST standards and conventions for menus)
⇒ File (if appropriate)
⇒ View (if appropriate)
⇒ Selected menu (if appropriate)
⇒ Edit menu
⇒ Help menu

For windows containing pushbutton controls:

⇒ Design System/Control menu
⇒ Design the buttons in command area

- Mapping object relationships to window families: For example, objects that depend on each other are in the same window family.

4.5.1 UCD Workshop Part 3 - OBJECT-TO-INTERFACE MAPPING

Objective: Implementing the application design by mapping the system objects to interface objects

Output:

The output of Part 3 of the UCD Workshop is a paper prototype (e.g., flip chart paper for the desktop, typing paper for a window, stickies for dialog boxes and menus) of the interface per se, with its data objects, folder objects, and device objects represented as windows, icons, and text strings. The dynamics of the interface may be captured by videotaping the paper prototype being used to do the task. This combination of paper prototype and video of its use is one part of the user requirements that are given to developers.

Benefit:

A benefit of Part 3 of the UCD Workshop is creating a design that reflects the user's mental model of the desirable way to do the task, when that task model was created without consideration of the particulars of the type of user interface.

> The UCD Workshop allows participants to quickly go from user requirements to GUI designs that are portable across platform look-&-feels. What enables these workshops to give participants those abilities is the blending of highly effective, inexpensive, low-tech methods such as PICTIVE with an object orientation used to describe user needs, requirements, and resulting system designs.

4.6 GUI Requirements Documentation

A MS Word template is available for documenting GUI Requirements. This template provides a detailed structure for writing GUI requirements. The output of the UCD Workshops, Parts 1, 2, and 3 may be directly entered into this template. The template may be modified and/or extended to fit application specific needs. This template is available from IT-Architecture & Standards, Tara Powell (205) 988-6085.

During the Analysis Phase, you should complete the following sections of the GUI Requirements template. This document should be circulated to all design team members for review.

- Brief overview of the application, user population, tasks performed by the user, work environment, and product goals.
- Product and usability goals
- Assumptions, dependencies, and contraints
- User Population (User Profile information)
- High level tasks that represent the user's job. Create a separate task list for each category of user
- Work environment
- Related products used to accomplish the job
- Realistic Task Flow from the UCD workshops, Part 1 - Task Flow Definition.

4.7 Refine the Low Fidelity GUI Prototype

During Part III of the UCD workshop, a paper and pencil prototype of each Primary and Secondary Windows is constructed**.** This paper and pencil prototype should capture high level window design that includes GUI object attributes and actions performed by the user, pull-down menus for the window, and interaction technique. Each GUI Object is represented by a Primary Window and possibly one or more Secondary Windows.

In this step, the low fidelity paper prototype from Part III of the UCD workshop is refined. At this point, the prototype should remain low fidelity and should not be built using a GUI tool (e.g., Visual Basic, UIM/X). The dynamics of the low fidelity prototype make it easier to continue to refine the GUI design. Continue usability testing to validate the GUI paper-and-pencil prototype designs - window design and navigation must support user goals, product objectives, and the business model.

The GUI (MS Windows®, OSF Motif™) environment must be identified in this step. The prototype should reflect this GUI environment (e.g., desktop, GUI controls, window management) to ensure that the prototype design can be constructed.

In this step, the low fidelity GUI prototype is refined to include the following.

⇒ how GUI/Task Objects will be displayed in both an Open (e.g., window) and Closed (e.g., icon, list item) state

⇒ overall look-and-feel of the GUI including integration with the desktop

⇒ metaphor design (if any), include ideas for icons and graphics

⇒ alternative window designs (in early design stages, you will likely have several design alternatives). Low fidelty prototypes allow the design team to demonstrate alternative designs and make quick design changes. The design team may then select the most appropriate design for the application.

⇒ high level interaction model for your GUI. This model should describe how users will use input devices (e.g., mouse, keyboard) for selection and data entry. Internal consistency is critical. External consistency promotes learning and is critical if your GUI will be marketed outside BST. Attempt to demonstrate interaction in the low fidelity prototype. The target GUI environment must support the interaction model.

- ♦ drag and drop
- ♦ interaction using a pointing device (mouse) - selection, activate an object/GUI control,
- ♦ keyboard equivalence (specify in the GUI Requirements)
- ♦ selection policy

4.8 Develop Usability Testing Scenarios

Develop test scenarios for usability requirements. These are typical usage scenarios (scripts) that describe how the GUI and overall system design will used by the user to perform tasks identified in task analysis. Test/usability scenarios must represent a cross-section of on-the-job tasks that will be performed by the user.

Test/usability scenarios must also demonstrate that Usability and Product objectives are being met by the GUI design. These scenarios should drive Usability testing in order to verify that the GUI design supports each task. Usability testing results should be integrated back into the GUI and product design.

4.9 Document the Initial User Interface Model

The Initial User Inteface Model includes the GUI analysis and screen flows. Model should map usage scenarios to screen flows.

1. The Initial User Interface Model is a compilation of the following deliverables:
    - GUI Analysis documentation (task flows and GUI object definitions defined in UCD workshops Parts 1 and 2)
    - *GUI Requirements* documentation - Formal description of Window design
        - ⇒ Paper-and-pencil prototypes of Window design
        - ⇒ Window flow diagram & definition
        - ⇒ Relationship & hierarchy of windows
    - Refined Usage Scenarios for testing built in the Planning Phase. These scenarios will be finalized for the Story Board prototype.
    - Description of how the Window design/flow meet the needs of the Usage Scenarios
2. User review and sign off of the Initial User Interface Model - This is a formal review of the model by a selected group of Users, User Management/Sponsor, and GUI design team. Feedback from this review should be incorporated back into the Interface Model before releasing the final document.

4.10 Identify Constraints on the GUI

Identify GUI design constraints. Constraints may be organizational and technological.

4.11 Application User Interface Handbook

The Application UI Handbook is a "living document" continually enhanced throughout design. Every member of the GUI design team should have a copy of this handbook. The objective is to ensure that design is consistent throughout the application and meets industry standards and conventions for the target GUI environment. The application handbook should not duplicate the Industry or BST Style Guides. Instead, include look-and-feel and design conventions specific to the application GUI. This ensures "internal consistency" within the GUI. "External consistency" is achieved by adhering the Industry and BST standards.

At this point, an Initial version of the Application User Interface Handbook should be developed. The handbook will be refined and finalized during the Design Phase.

This handbook should be edited by the GUI CORE team leader with input from the entire GUI team. All versions of the handbook should be reviewed by the entire GUI CORE team. Include as many figures and window snapshots as possible to illustrate design. *The BellSouth GUI Style Guide contains a template that may be used in developing the Application Handbook.*

- The GUI platform (e.g., Microsoft Windows®, OSF Motif™) previously selected for the application should be the basis for this handbook and the application look-and-feel. The interface should look and behave like other interfaces available in the industry and within BST for the target GUI platform.
- If the interface will run under multiple GUI environments, the look-and-feel must still be consistent with other native applications for that GUI. Thus, the GUI design team must use multiple platform style guides in their design. If you plan to use the "common denominator" approach to cross-platform GUI design using JAVA or some other cross-platform design tool, avoid the following temptation:

  ⇒ Limit your design to GUI controls and interaction methods available in all target GUI platforms. This may make your appear to be less functional than other native GUI applications and may be unacceptable to the user.
- In addition to the GUI platform style guide, use the BellSouth GUI Style Guide for design standards and conventions. The BellSouth style guide covers both Microsoft Windows® and OSF Motif™ and can be used in cross-GUI development.

4.12 Develop / Test / Refine a Storyboard Prototype

Useful to verify design against representative task flows. High fidelity storyboards can be used to verify interaction techniques. This is the last step before a working prototype is built.

- Build a Storyboard that illustrates key & typical user tasks identified in during Task Analysis and documented in the Usage Scenarios. Use Cases, if available, may be used to develop the Storyboard flow.
  - ⇒ Use the low fidelity window prototypes developed in the UCD workshop Part III and refined for the Initial User Interface Model. This prototype should be paper-and-pencil at this stage of design. Resist all temptation to construct a working prototype!
  - ⇒ Integrate the Interaction Model identified in Initial User Interface Model into the storyboad

- Perform a design walk-through with the GUI Design Team to identify problems with the Initial User Interface Model. Revise the Window Prototypes, Conceptual Design Model, Interaction Model, GUI Object Model, etc. as necessary. The system analyst should verify that data used in the UI design can be obtained and maintain expected performance.
  - ⇒ Task Analysis documentation, Usage Scenarios, and Use Cases may also be verified during the Walk-through

- Perform Informal Usability Testing on the Storyboard - Test Usage scenarios against the Storyboard design to determine if the GUI design is complete and acceptable.
  - ⇒ Usability testing implies that "actual end users" will perform the usage scenarios to test the GUI. Project Managers and Management from the User Organizations may also participate in the Usability Testing.
  - ⇒ Feedback and results from Usability testing should be analysed by the GUI design team, summarized, and incorporated back into the design.
  - ⇒ Repeat this iterative design and testing cycle until the design is complete and correct. Completeness and correctness is defined by the Usability and Product Objectives.

- Obtain User Sign-Off and CORE GUI Team Sign-off on the Story Board prototype before continuing to the next step.

4.13 Construct a Navigational Prototype

Build a Navigational Prototype of the GUI using a GUI development tool specific to your target GUI environment (e.g., Visual Basic, UIM/X). The Story Board prototype and Initial User Interface model is used as input into this step.

- Development of a Navigational Prototype is an iterative task.
  Design ---> Test ---> Refine
- Prototype will later be thoroughly tested by the users for Usability and completeness.
- Prototype must address:
  - ⇒ High frequency Tasks and Usage Scenarios
  - ⇒ Components of the GUI where design is questionable (incomplete requirements, poor preliminary usability)
  - ⇒ Features (e.g., actions, dialogs) of the design that are repeated throughout the GUI and used frequently by the user.
  - ⇒ Design approaches (e.g., metaphor, custom GUI controls) used throughout the GUI (include the Notebook and Hierarchical List controls).
  - ⇒ Drag and drop; GUI object selection
- Prototype should include:
  - ⇒ Complete Primary and Secondary (dialog) window design
  - ⇒ Message dialogs (warning, error, information)
  - ⇒ Principle help dialogs to illustrate how help is invoked and displayed
  - ⇒ Icons (application, desktop) - preliminary design
  - ⇒ Full navigation between windows to demonstrate flow
  - ⇒ Actual and representative data

4.14 Test the Navigational Prototype

Testing and design walk-throughs by the GUI design team should address the following:

- Problems with window navigation
- Data presented is the wrong window or out of sequence for the task
  ⇒ difficult to access data
  ⇒ data availability
  ⇒ data included with the appropriate GUI object
- Problems with the Object Hierarchy, containment
- GUI does not fit task requirements efficiently
- Missing windows (secondary)
- Window or menu hierarchy too complex
  ⇒ too many windows
  ⇒ doesn't match task flow
- Terminology used in menus, buttons, etc. not within the user's knowledge or job domain
- Inappropriate message wording
- Task inefficiency - too many steps, windows, etc.
- Confusing or cluttered window design
- Unclear or missing feedback
- Non-compliance to industry and BST GUI standards
- Internal consistency (common look and feel)

4.14.1 Perform a Design Walk-through

Objective is to review and enhance the prototype before Formal Usability Testing in a laboratory.

4.14.2 Perform Usability Testing

Both Formal and Informal Usability Testing should be performed. Informal testing is typically performed during analysis and design, and formal testing is performed after the Navigational Prototype is complete and after each major GUI component is completed. The objective is to have real users verify that your GUI design meets Usability Objectives developed in the Planning Phase and GUI Requirements.

- Use the Usability Test Plan as a basis for testing
- Implement the steps documented in the BST Usability Evaluation Guide
- Usability Testing should address:
  - ⇒ High frequency Tasks and Usage Scenarios
  - ⇒ Components of the GUI where design is questionable (incomplete requirements, poor usability)
  - ⇒ Features (e.g., actions, dialogs) of the design that are repeated throughout the GUI
  - ⇒ Design approaches (e.g., metaphor, GUI controls) used throughout the GUI
- Results of Usability Testing should be incorporated back into the Navigational Prototype and documented in GUI Requirements.
- Retest the modified Navigational Prototype to verify changes
- Repeat the design --> test --> change cycle until Usability Testing results comply with previously established Usability Objectives. Allow for 3 - 4 iterations of this design cycle.

Below are the basic steps for Usability Testing (refer to the Usability Planning Handbook for a full description) of each step.

- Develop a Usability Test Plan - Refer to the "Usability Handbook" for a complete guide to planning and implementing Usability Tests. Below are the major steps:
  - ⇒ Select users to participate in the testing
  - ⇒ Use Usage Scenarios from the UI Story Board to test the Navigational Prototype
  - ⇒ Prepare Support Materials to be used during the test (e.g., job artifacts)
  - ⇒ Ensure that data needed for the test is built into the Navigational Prototype
  - ⇒ Identify test location and obtain equipment/software/communications needed for the test
  - ⇒ Schedule a series of testing sessions for the prototype
  - ⇒ Contact and schedule users to participate in each session

4.15 Refine the User Interface Model

This is a "Living Design Specification" of the GUI. Update the following:

- The Application User Interface Handbook (the BST GUI Style Guide contains a template) is also used to document the GUI.
- Update the GUI Requirements document with design enhancements and changes identified in the Navigational Prototype. Include the following:
  - ⇒ Descriptions of all windows, position within the window hierarchy, window title
  - ⇒ Snapshot of each window (specify in the GUI Requirements)
  - ⇒ Description of accelerators, mnemonics, tool-bar icons, and tool-tip pop-ups

4.16 Object Oriented Design Process & Deliverables

The OO design process documented in this methodology is based upon the use case methodology and the object modeling technique (OMT).

During Analysis Phase, the GUI Analyst should provide high-level User Interface Diagrams to design teams to assist in requirements definition. In the Analysis Phase, the GUI Analyst should participate in UCD workshop Part 2 to identify GUI Objects (attributes, methods, containment). Early window prototypes produced during the Analysis Phase in the UCD Workshop Part 3 should be paper and pencil vs. on-line using a GUI Builder or Prototyping tool. This allows easier and more "hands-on" design changes. Once on glass, the design is harder to change.

4.16.1.1 Actor Action Definitions

Identifies the action that a user can perform within the context of a Use Case Specification. AADs contain the "events" created as the Actor performs the Use Case.

- When can it occur? Identify the conditions or states when the action can take place.
- Who initiates the action and how? Identify whether the Actor or the system initiates
- What is it? Describe the nature of what occurs, information content, choices, changes, and activities that can occur during the action. Do not specify GUI details or design at this point.
- Expected Outcome - Identify possible outcomes of the action, state of the information recorded, state of the system when the action is complete.

The AAD should not designate “How” the action is invoked (e.g., menu, command button). Instead the AAD should capture the nature of the event triggered by the actor. Subject Matter Experts (SMEs) build the Actor Action Definitions (AADs).

Link the AAD to the Model

- Create a link (hyperlink if possible) between the Use Case and the action

Sample Actor Action Definition

Actor Action Name:	View Backup Line Service
[When will this action occur?]	
Who Initiates?	Customer Representative
Nature of Action	On the request of the Customer Representative, the system displays the Back-up Line state for the requested Customer Account, with all of its associated features and characteristics.
Outcome(s)	System displays all of the required data and waits for further action by the Customer Representative.

5. Design Phase

The goal of the Design Phase is to transform the Analysis Phase Model into a model of how the system will be constructed to meet requirements. All environmental and implementation features and constraints that affect the GUI must be considered and resolved during the Design Phase.

Analysts evolve and expand information gathered in the Analysis Phase to specify what exactly must be done to satisfy the requirements. *You must obtain user sign-off in design review.*

During the Design Phase, you must perform the following key tasks:
1. Build or acquire re-usable OO GUI Framework
2. Build the detailed and final User Interface Model.

At the end of the Design Phase, you should provide the following documentation:
- User Interface Model (UIM) which includes User Interface Diagrams (UIDs) and Actor Action Definitions (AADs).
- Expanded User Interface Model (UIM) - consists of an Analysis Prototype, completed Actor Action Definitions, Application Windows and associated Window Definitions
- Final GUI Requirements Document

The following individuals/roles participate in Design Phase:
1. User Subject Matter Experts (SME's) - must be knowledgeable of the business domain
2. GUI Analysts - should have knowledge of the production environment and business domain
3. Facilitators and experts familiar with the methodology (e.g., use case methodology) used to gather and document requirements.
4. Usability Engineer
5. Project and/or Product Manager accountable for the project and its implementation
6. Analysts responsible for database design and architecture
7. Representatives from the group responsible for installation and deployment
8. Representatives from Training
9. Representatives from Infrastructure groups (e.g., security, network, hardware)
10. Representatives from the Testing group

5.1 Develop the Detailed User Interface Model

This is an iterative refinement of deliverables from the Analysis Phase of the GUI Design Methodology. The Detailed User Interface Model describes the final GUI look-and-feel, interface attributes and actions, metaphor design, functionality, and navigation between windows.

The objective is to document the Detailed User Interface Model and to allow users, management, and the design team to review and provide final input into the design.

Team members involved in documenting this deliverable include business analysts, system and GUI analysts, GUI CORE team lead, and GUI design team. Management, users, and key project team members should be involved in reviewing this deliverable.

The Detailed User Interface Model includes refined versions of the documentation deliverables listed below:

1. Update the Usage Scenarios - Usage Scenarios developed in the Analysis Phase are enhanced and updated to show how users will interact with the system.
   - GUI design team, users, and management should verify (perform usability testing) the storyboard prototype and the Navigational Prototype against usage scenarios.
   - project design team should incorporate results from testing and other user feedback into the usage scenarios. The objective is to better match the usage scenarios with actual on the job product usage (describe how the actual end-user interacts with the system).

2. Update the User Interface Attributes and Actions - During the UCD Workshop Part 2 in the Analysis Phase, User Interface objects are identified. Attributes (e.g., data, child objects contained within the object) and actions (e.g., actions performed on the object) are defined for each UI object. User feedback and results from usability testing should be used to refine the UI object attributes and actions. Update the GUI requirements, UCD Workshop documentation, etc. accordingly to reflect all changes.

3. Update the User Interface Navigational Prototype - Prototype developed in the Analysis Phase is enhanced and finalized.
   - Ensure that the prototype includes Metaphor design (if any). *Note that it is not necessary to design a metaphor for your GUI. Metaphors must enhance usability and ease of learning. Inappropriate metaphors will slow learning, mislead users, and negatively affect user productivity.*
   - Refine each GUI Component, including all Primary and Secondary Windows
     $\Rightarrow$ complete design for each window - finalize layout, selection of GUI controls

⇒ finalize how information will be formatted in each window (e.g., GUI control used to display information, column headings, order in which columns are displayed)
⇒ finalize labels used for buttons, menus, fields, lists, and any other GUI control
⇒ define use of highlighting (bold, color)
⇒ define use of color within the window (GUI control, graphics)
⇒ define use of business graphics/charts, structured graphics, graphic art design
⇒ finalize fonts (e.g., style, size, weight)
⇒ finalize icon design

Use the UCD Workshop Part 3 if necessary to further define each window and/or further decompose a GUI component.

- GUI design team, users, and management should review (perform usability testing) the navigational prototype. Ensure that Usability and Product objectives are being met.
- GUI design team should modify the Navigational Prototype as needed based on feedback from Usability testing and user review.
- GUI design team should incorporate results from testing and other user feedback back into the Navigational Prototype. The objective is to better refine the prototype. Note that this testing is in addition to testing already performed and approvals already obtained for the navigational prototype.

4. Finalize GUI Requirements Document - Update the GUI Requirements document to reflect changes and enhancements to the Navigational Prototype.

5.2 Define Reusable Components

The objective is to ensure that business objects exist to perform functions invoked through the GUI. Below are the deliverables for this step:

- Develop a Set of User Interface Reusable Classes - GUI design team should identify "GUI" components of the Detailed User Interface Model that may be reused. In an OO application, these components should be constructed as reusable classes.

- Develop a Set of Business Reusable Classes - business analysts with the project design team should identify "business" components of the Detailed User Interface Model that may be reused. In an OO application, these components should be constructed as reusable classes.

5.3 Perform Usability Testing

The Analysis and Design Phase prototype should be tested thoroughly for usability where the user performs "real life" usage scenarios. Iterative usability testing is a valuable tool to refine your prototype design and requirements specification. Three iterations of usability testing is typically required. After each iteration, all problems and gaps in requirements should be resolved prior to the next test. Incomplete and non-functional code can be used for usability testing.

5.3.1 Formal Usability Testing

Formal Usability is a well defined exercise to access the usability and acceptability of your product. Informal usability is simply an abbreviated version of the formal evaluation process.

Formal usability evaluations should be used to evaluate the following for a product or major product component:

- *End-to-end* functionality
- Overall usability and acceptability
- Overall product flow
- Navigation between product components

Formal evaluations should be performed on the User Interface prototypes. It is best to evaluate fully functional prototypes and to use realistic data (e.g., test or training databases). Prototypes with canned data or missing functionality will affect your evaluation if these components are key to completing a task.

Schedule formal evaluations to coincide with the delivery of key product components or a prototype. Because of the resources and time to perform a formal evaluation, they should be built into the product design schedule.

Formal evaluations are categorized by the following:

- Usability lab equipped with video and sound recording
- Dedicated period of one week to evaluate and analyze product design
- Well planned and documented evaluation plan
- Well organized usability team and facilitator
- Observers from outside the usability and design teams
- Documented evaluation results and summary of findings for management

Formal evaluations should not be the only usability performed on your product. Many informal evaluations should already have been performed during iterative design.

Therefore, the scope of formal usability is much broader and includes modules previously evaluated in informal evaluations. Formal usability should focus on overall product flow and functionality. Hopefully, there should be no surprises by the time you reach a formal evaluation.

Formal usability is much more controlled than informal evaluations. It is appropriate and key to your evaluation to invite management and key members of the project team to observe the evaluation.

5.3.2 Informal Usability Testing

It is not necessary to go through a formal evaluation to obtain user feedback. Instead, informal evaluations can be performed on early design concepts, screen sketchs, and screen prototypes. Informal usability takes place early during functional and high level design.

Usability is an iterative process to define, refine, and verify design. It is not a one time evaluation near completion of your product. A typical product component should be evaluated for usability 2 or more times before it is considered complete and ready for development. Informal usability provides an easy and well-defined process for evaluating design without involving alot of resources.

You can evaluate paper sketches of user interfaces, documentation, and training to verify design assumptions and to ensure that product requirements are complete and provide the needed functionality. These are early sanity checks to verify requirements and design.

It is a good practice to schedule an informal usability every two weeks throughout the design period. This scheduled routine keeps usability in the forefront and helps ensure that more product components are evaluated and that evaluations are timely. Members of the design team submit items for evaluation to the person coordinating the evaluations. Don't worry about not having something to evaluate every two weeks.

It is a good idea for key design team members to meet on a regular basis to discuss usability changes and issues. These meetings can be useful to obtain buy-in for changes and to discuss ongoing design issues.

Usability can also be performed on an existing or competitive product. Identifies product weaknesses, missing functionality, and problems with ease of use that will become requirements for the new product.

Consider grouping two users together, each taking turns performing the task scenarios. Encourage both users to comment and talk out loud throughout the tasks. This method is very effective when time is limited.

5.4 Refine the Application User Interface Handbook

During the Design Phase, the GUI team should finalize the Application User Interface Handbook started in the Planning Phase and updated during the Analysis Phase.

This handbook should capture all *application specific* design conventions. This handbook should not duplicate either Industry or the BellSouth GUI Style Guide.

The Application User Interface Handbook should address the following:

- Color coding and highlighting conventions
- Use of non-industry standard button labels
- Conventions for tabbing and keyboard navigation
- Method for accessing on-line help
- Font styles
- Special data field formats (e.g., telephone number, social security number)
- Window management and size conventions
- Conventions for Error Handling
- Error, Warning, and Informational Messages
- Abbreviations and accronyms used within the application
- Terminology specific to the application

6. Construction Phase

During the Construction Phase, you must perform the following key tasks:

- Complete detailed design of the user interface
- Review usability objectives
- Build the GUI using a design that supports the production environment
- Conduct scheduled design reviews of the user interface to ensure usability and consistency with standards.
- Conduct ongoing high-fidelity Usability testing of the user interface
- Input Usability test results back into GUI requirements as needed.
- Perform Formal Usability Testing on key component deliverables.
- Facilitate the design of training materials and user interface documentation

The following individuals/roles participate in Construction Phase:

1. User Subject Matter Experts (SME's) - must be knowledgeable of the business domain
2. GUI Analysts - should have knowledge of the production environment and business domain
3. Facilitators and Experts familiar with the Methodology (e.g., Jacobson's User Case Methodology) used to gather and document requirements.
4. Analysts skilled at working with User SMEs to gather requirements
5. Usability Engineer
6. Project and/or Product Manager accountable for the project and its implementation
7. Analysts responsible for database design and architecture
8. Representatives from the group responsible for installation and deployment
9. Representatives from Training
10. Representatives from Infrastructure groups (e.g., security, network, hardware)
11. Representatives from the Testing group

7. Implementation Phase

The goal of the Implementation Phase is to transform the Design Phase model into a working product.

The following individuals/roles participate in Implementation Phase:
1. User Subject Matter Experts (SME's) - must be knowledgeable of the business domain
2. GUI Analysts - should have knowledge of the production environment and business domain
3. Facilitators and Experts familiar with the Methodology (e.g., Jacobson's User Case Methodology) used to gather and document requirements.
4. Analysts skilled at working with User SMEs to gather requirements
5. Usability Engineer
6. Project and/or Product Manager accountable for the project and its implementation
7. Analysts responsible for database design and architecture
8. Representatives from the group responsible for installation and deployment
9. Representatives from Training
10. Representatives from Infrastructure groups (e.g., security, network, hardware)
11. Representatives from the Testing group

7.1 Object Oriented Design Process & Deliverables

During the Implementation Phase, the WDs and AADs are used during the construction of the user interface (e.g., menus, GUI interaction, GUI navigation).

Application components are unit tested for conformance with GUI guidelines.

Perform Usability Testing
The lead GUI analyst participates in formal Usability testing. Usability should be scheduled throughout the Implementation Phase.

GUI analyst maintains Application Windows, Window Definitions (WDs), and physical information in the Actor Action Definitions (AADs).

8. Maintenance Phase

8.1 Object Oriented Design Process & Deliverables