Module – V

Interface Testing

Objective

- To understand the testing features of Interface and selecting proper testing tool
- To know about various layout design principles to implement
- To acquire knowledge about hypermedia, www and visualization
- To know about various software tools

Organize and Layout Windows and Pages

- In graphical user interfaces, components to be included on windows include a title, screen controls, headings, other screen content, and possibly instructional messages.
- On Web pages, components to be included consist of elements such as the page title, textual content, graphics, headings, screen controls, links, and other necessary components.

General Guidelines

- Amount of information:
  — Present the proper amount of information on each screen.
    • Too little is inefficient.
    • Too much is confusing.
  — Present all information necessary for performing an action or making a decision on one screen, whenever possible.
- Organization:
  — Provide an ordering that:
    • Is logical and sequential.
    • Is rhythmic, guiding a person's eye through the display.
    • Encourages natural movement sequences.
    • Minimizes pointer and eye movement distances.
- Control placement:
  — Position the most important and frequently used controls at the top left.
  — Maintain a top-to-bottom, left-to-right flow.
  — If one control enables or affects another, the enabling control should be above or to the left of the enabled control.
  — Place the command buttons that affect the entire window horizontally, and centered, at the window's bottom.
- Navigation:
  — The flow of interaction should:
    • Require as little cursor and pointer travel as possible.
    • Minimize the number of times a person's hand has to travel between the keyboard and the mouse.
  — Assist users in navigating through a screen by:
    • Aligning elements.
    • Grouping elements.
    • Using line borders.

- Aesthetics:
  — Provide a visually pleasing composition through:
    • Adequate use of white space.
    • Balance.
    • Groupings.
    • Alignment of elements.
- Visual clutter:
  — Avoid visual clutter by:
    • Maintaining low screen density levels.
    • Maintaining distinctiveness of elements.
- Focus and emphasis:
  — Provide visual emphasis to the most important screen elements, its data or information.
  — Sequentially, direct attention to items that are:
    1. Critical.
    2. Important.
    3. Secondary.
    4. Peripheral.
- Consistency:
  — Provide consistency.
    • With a person's experiences and cultural conventions.
    • Internally within a system.
    • Externally across systems.

Organization Guidelines

Organizational guidelines to be addressed include those relating to groupings, borders, dependent controls, alignment, and balance.

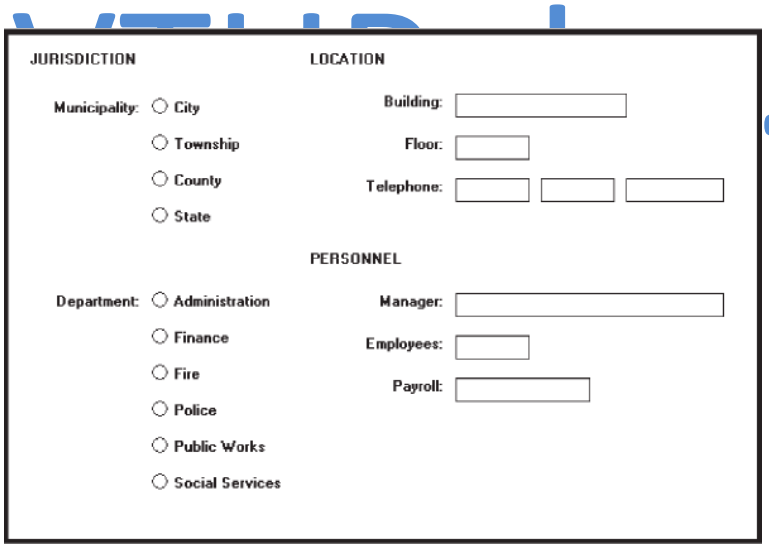
Creating Groupings

- General:
  — Provide groupings of associated elements.
    • Elements of a radio button or check box control.
    • Two or more related fields or controls.
  — Create groupings as close as possible to 5 degrees of visual angle.
- White space:
  — Provide adequate separation of groupings through the liberal use of white space.
  — Leave adequate space:
    • Around groups of related controls.
    • Between groupings and window borders.
  — The space between groupings should be greater than the space between fields within a grouping.
- Headings:
  — Provide section headings and subsection headings for multiple control groupings.
  — Provide headings that meaningfully and concisely describe the nature of the group of related fields.
- Borders:

— Enhance groupings through incorporation of borders around:
    • Elements of a single control.
    • Groups of related controls or fields.
— Individual control borders should be visually differentiable from borders delineating groupings of fields or controls.
    • Provide a border consisting of a thin line around *single* controls.
    • Provide a border consisting of a slightly thicker line around *groups* of fields or controls.
— Do not place individual field or control borders around:
    • Single entry fields.
    • Single list boxes.
    • Single combination boxes.
    • Single spin boxes.
    • Single sliders.
— Do not place borders around command buttons.

Borders

- Groupings can be further enhanced through the use of borders. Inscribe line borders around elements of a single control such as a radio button or check box and/or groups of related controls or fields.
- Individual control borders should be visually differentiable from borders delineating groupings of fields or controls.
- Provide a border consisting of a thin line around single controls and a slightly thicker line around groups of fields or controls.



Control Borders

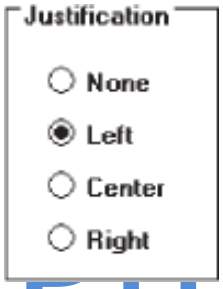
- Incorporate a thin single-line border around the elements of a selection control.
- For spacing:
  — Vertically, leave one line space above and below the control elements.
  — Horizontally:

• Leave at least two character positions between the border and the left side of the control elements.
• Leave at least two character positions between the border and the right side of the longest control element.

— Locate the control caption in the top border, indented one character position from the left border.

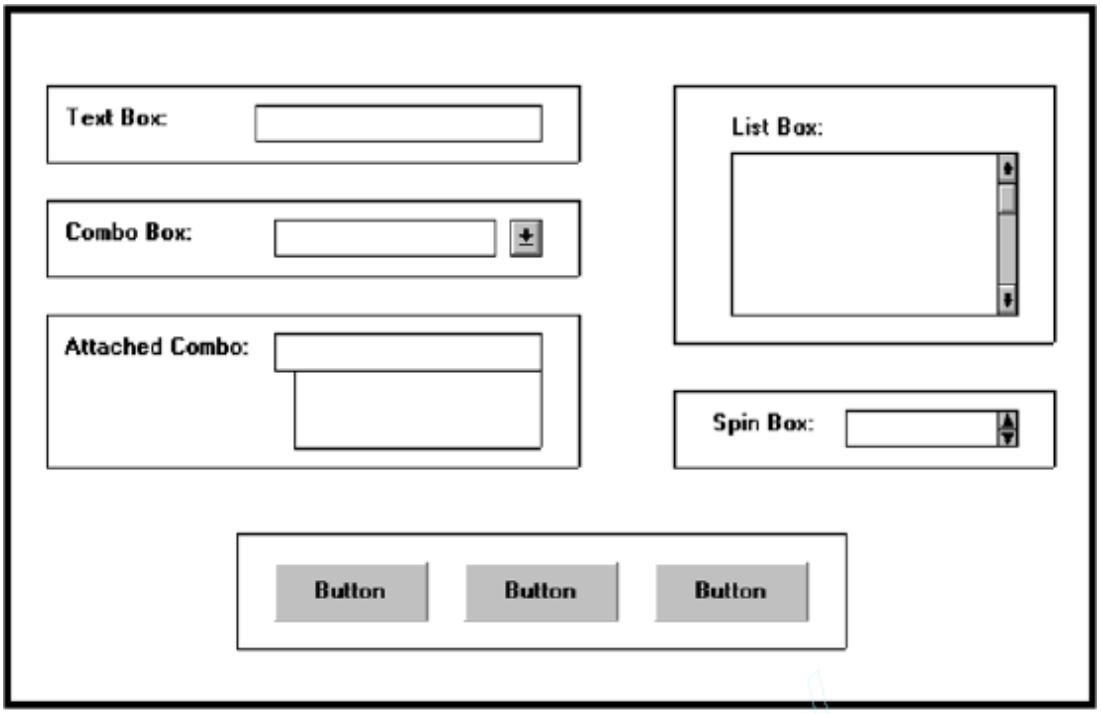
• Alternately, locate the caption at the upper left of the box.

— If the control caption exceeds the length of the choice descriptions, extend the border two character positions to the right of the caption.



Section Borders

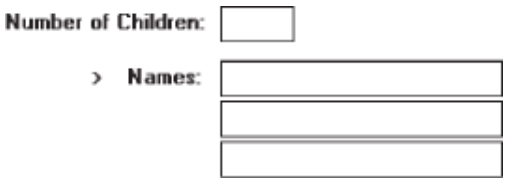
- Incorporate a thicker single-line border around groups of related entry or selection controls.
- For spacing:
  — Vertically, leave one line space between the top and bottom row of the entry or selection control elements.
  — Horizontally, leave at least four character positions to the left and right of the longest caption and/or entry field.
- Locate the section heading in the top border, indented two character positions from the left border.



Dependent Controls

- Position a conditional control, or controls:
  — To the right of the control to which it relates.

  — Alternately, position it below the control to which it relates.



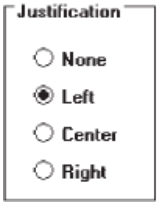
- Either:
  — Display these conditional controls in a subdued or grayed out manner.
    • When a control is relevant, return it to a normal intensity.
  — Do not display a conditional control until the information to which it relates is set.
- Inscribe a filled-in arrow between the selected control and its dependent controls to visually relate them to each other.

Aligning Screen Elements

- Minimize alignment points on a window.
  — Vertically.
  — Horizontally.

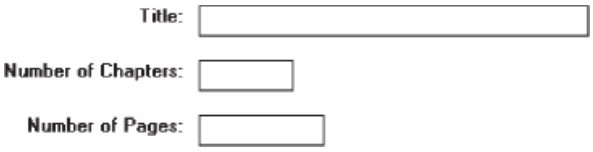
Vertical Orientation and Vertical Alignment

- Radio buttons/check boxes:
  — Left-align choice descriptions, selection indicators, and borders.
  — Captions:
    • Those inscribed within borders must be left-aligned.

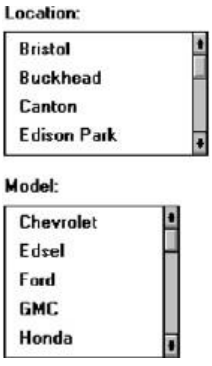


    • Those located at the left may be left- or right-aligned.

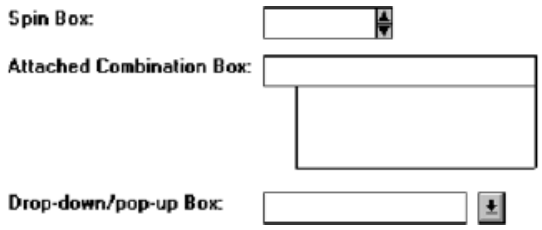
- Text boxes:
  — Left-align the boxes. If the screen will be used for inquiry or display purposes, numeric fields should be right-aligned.
  — Captions may be left- or right-aligned.



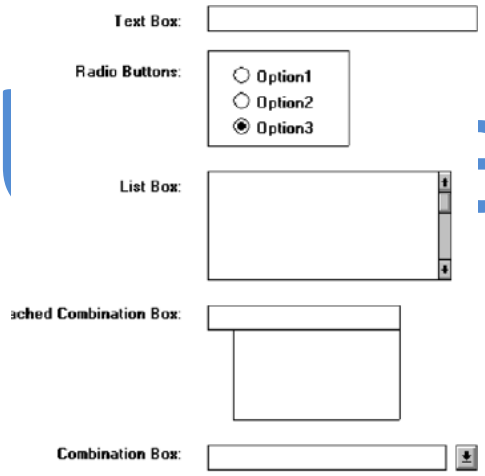
- List boxes:
  — Left-align fixed list boxes.
  — Captions:
    • Those located above the boxes must be left-aligned.



    - Those located at the left may be left- or right-aligned.
- Drop-down/pop-up boxes, spin boxes, combo boxes:
  — Left-align control boxes.
  — Field captions may be left- or right-aligned.



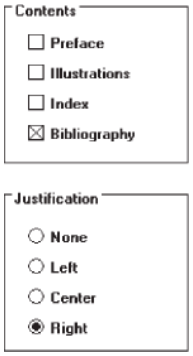
- Mixed controls
  — Left-align vertically arrayed:
    - Text boxes.
    - Radio buttons.
    - Check box boxes.
    - Drop-down/pop-up list boxes.
    - Spin boxes.
    - Combination control boxes.
    - List boxes.
  — Captions may be left- or right-aligned.



Balancing Elements

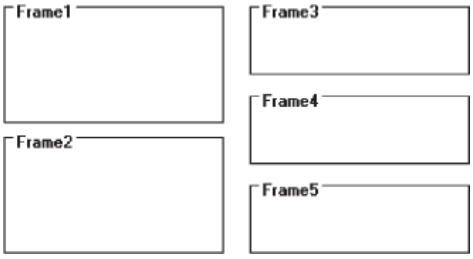
- General:
  — Create balance by:
    - Equally distributing controls, spatially, within a window.
    - Aligning borders whenever possible.
- Individual control borders:
  — If more than one control with a border is incorporated within a column on a screen:
    - Align the controls following the guidelines for multiple-control alignment.
    - Align the left and right borders of all groups.

• Establish the left and right border positions according to the spacing required for the widest element within the groups.



— With multiple groupings and multiple columns, create a balanced screen by:

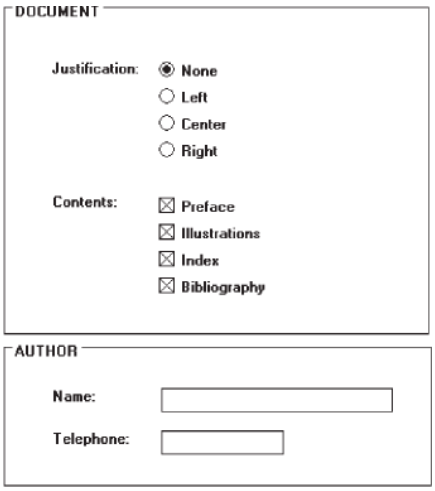
• Maintaining equal column widths as much as practical.
• Maintaining equal column heights as much as practical.



- Section borders:

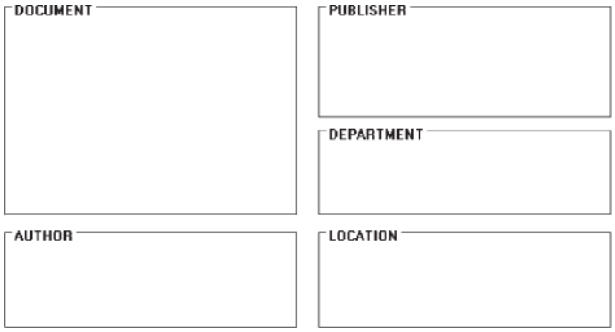
— If more than one section with borders is incorporated within a column on a screen:

• Align the left and right borders of all groups.
• Establish the left and right border positions according to the spacing required by the widest element within the groups.



— With multiple groupings and multiple columns, create a balanced screen by:

• Maintaining equal column widths as much as practical.
• Maintaining equal column heights as much as practical.



Window Guidelines

- Organization:
  — Organize windows to support user tasks.
  — Present related information in a single window whenever possible.
  — Support the most common tasks in the most efficient sequence of steps.
  — Use:
    • Primary windows to:
    — Begin an interaction and provide top-level context for dependent windows.
    — Perform a major interaction.
    • Secondary windows to:
    — Extend the interaction.
    — Obtain or display supplemental information related to the primary window.
    • Dialog boxes for:
    — Infrequently used or needed information.
    — "Nice-to-know" information.
- Number:
  — Minimize the number of windows needed to accomplish an objective.
- Size:
  — Provide large enough windows to:
    • Present all relevant and expected information for the task.
    • Not hide important information.
    • Not cause crowding or visual confusion.
    • Minimize the need for scrolling.
    • Less than the full size of the entire screen.
  — If a window is too large, determine:
    • Is all the information needed?
    • Is all the information related?

Test, Test, and Retest

Testing steps to be reviewed are:

- Identifying the purpose and scope of testing.
- Understanding the importance of testing.

- Developing a prototype.
- Developing the right kind of test plan.
- Designing a test to yield relevant data.
- Soliciting, selecting, and scheduling users to participate.
- Providing the proper test facility.
- Conducting tests and collecting data.
- Analyzing the data and generating design recommendations.
- Modifying the prototype as necessary.
- Testing the system again.
- Evaluating the working system.

The Purpose of Usability Testing

- First, it establishes a communication bridge between developers and users. Through testing, the developer learns about the user's goals, perceptions, questions, and problems.
- Second, testing is used to evaluate a product. It validates design decisions. It also can identify potential problems in design at a point in the development process where they can be more easily addressed.

The Importance of Usability Testing

A thorough usability testing process is important for many reasons,

- Developers and users possess different models.
- Developer's intuitions are not always correct.
- There is no average user.
- It's impossible to predict usability from appearance.
- Design standards and guidelines are not sufficient.
- Informal feedback is inadequate.
- Problems found late are more difficult and expensive to fix.
- Advantages over a competitive product can be achieved.

Scope of Testing

- Testing should begin in the earliest stages of product development and continue throughout the development process.
- It should include as many of the user's tasks, and as many of the product's components, as reasonably possible.

Prototypes

- A prototype is primarily a vehicle for exploration, communication, and evaluation. Its purpose is to obtain user input in design, and to provide feedback to designers.
- A prototype is a simulation of an actual system that can be quickly created.

- A prototype may be a rough approximation, such as a simple hand-drawn sketch, or it may be interactive, allowing the user to key or select data using controls, navigate through menus, retrieve displays of data, and perform basic system functions.
- A prototype may have great breadth, including as many features as possible to present concepts and overall organization, or it might have more depth, including more detail on a given feature or task to focus on individual design aspects.

Hand Sketches and Scenarios

- Description:
  — Screen sketches created by hand.
  — Focus is on the design, not the interface mechanics.
  — A low-fidelity prototype.
- Advantages:
  — Can be used very early in the development process.
  — Suited for use by entire design team.
  — No large investment of time and cost.
  — No programming skill needed.
  — Easily portable.
  — Fast to modify and iterate.
  — A rough approximation often yields more substantive critical comments.
  — Easier to comprehend than functional specifications.
  — Can be used to define requirements.
- Disadvantages:
  — Only a rough approximation.
  — Limited in providing an understanding of navigation and flow.
  — A demonstration, not an exercise.
  — Driven by a facilitator, not the user.
  — Limited usefulness for a usability test.
  — A poor detailed specification for writing the code.
  — Usually restricted to most common tasks.
- **Sketch Creation Process**
  - Sketch (storyboard) the screens while determining:
    — The source of the screen's information.
    — The content and structure of individual screens.
    — The overall order of screens and windows.
  - Use an erasable medium.
  - Sketch the screens needed to complete each workflow task.
  - Try out selected metaphors and change them as necessary.
  - First, storyboard common/critical/frequent scenarios.
    — Follow them from beginning to end.
    — Then, go back and build in exceptions.
  - Don't get too detailed; exact control positioning is not important, just overall order and flow.
  - Storyboard as a team, including at least one user.
  - Only develop online prototypes when everyone agrees that a complete set of screens has been satisfactorily sketched.

Interactive Paper Prototypes

- Description:
  — Interface components (menus, windows, and screens) constructed of common paper technologies (Post-It notes, transparencies, and so on).
  — The components are manually manipulated to reflect the dynamics of the software.
  — A low-fidelity prototype.
- Advantages:
  — More illustrative of program dynamics than sketches.
  — Can be used to demonstrate the interaction.
  — Otherwise, generally the same as for hand-drawn sketches and scenarios.
- Disadvantages:
  — Only a rough approximation.
  — A demonstration, not an exercise.
  — Driven by a facilitator, not the user.
  — Limited usefulness for usability testing.

Programmed Facades

- Description:
  — Examples of finished dialogs and screens for some important aspects of the system.
  — Created by prototyping tools.
  — Medium-fidelity to high-fidelity prototypes.
- Advantages:
  — Provide a good detailed specification for writing code.
  — A vehicle for data collection.
- Disadvantages:
  — May solidify the design too soon.
  — May create the false expectation that the “real thing” is only a short time away.
  — More expensive to develop.
  — More time-consuming to create.
  — Not effective for requirements gathering.
  — Not all of the functions demonstrated may be used because of cost, schedule limitations, or lack of user interest.
  — Not practical for investigating more than two or three approaches.

Prototype-Oriented Languages

- Description:
  — An example of finished dialogs and screens for some important aspects of the system.
  — Created through programming languages that support the actual programming process.
  — A high-fidelity prototype.
- Advantages:

— May include the final code.
— Otherwise, generally the same as those of programmed facades.

- Disadvantages:
  — Generally the same as for programmed facades.

Kinds of Tests

A test is a tool that is used to measure something. The "something" may be:

- Conformance with a requirement.
- Conformance with guidelines for good design.
- Identification of design problems.
- Ease of system learning.
- Retention of learning over time.
- Speed of task completion.
- Speed of need fulfillment.
- Error rates.
- Subjective user satisfaction.

Guidelines Review

- Description:
  — A review of the interface in terms of an organization's standards and design guidelines.
- Advantages:
  — Can be performed by developers.
  — Low cost.
  — Can identify general and recurring problems
  — Particularly useful for identifying screen design and layout problems.
- Disadvantages:
  — May miss severe conceptual, navigation, and operational problems.

Heuristic Evaluation

- Description:
  — A detailed evaluation of a system by interface design specialists to identify problems.
- Advantages:
  — Easy to do.
  — Relatively low cost.
  — Does not waste user's time.
  — Can identify many problems.
- Disadvantages:
  — Evaluators must possess interface design expertise.
  — Evaluators may not possess an adequate understanding of the tasks and user communities.
  — Difficult to identify system wide structural problems.
  — Difficult to uncover missing exits and interface elements.

— Difficult to identify the most important problems among all problems uncovered.
— Does not provide any systematic way to generate solutions to the problems uncovered.

- Guidelines:
  — Use 3 to 5 expert evaluators.
  — Choose knowledgeable people:
    • Familiar with the project situation.
    • Possessing a long-term relationship with the organization.
- **Heuristic Evaluation Process**
  - Preparing the session:
    — Select evaluators.
    — Prepare or assemble:
      • A project overview.
      • A checklist of heuristics.
    — Provide briefing to evaluators to:
      • Review the purpose of the evaluation session.
      • Preview the evaluation process.
      • Present the project overview and heuristics.
      • Answer any evaluator questions.
      • Provide any special evaluator training that may be necessary.
  - Conducting the session:
    — Have each evaluator review the system alone.
    — The evaluator should:
      • Establish own process or method of reviewing the system.
    — provide usage scenarios, if necessary.
      • Compare his or her findings with the list of usability principles.
      • Identify any other relevant problems or issues.
      • Make at least two passes through the system.
    — Detected problems should be related to the specific heuristics they violate.
    — Comments are recorded either:
      • By the evaluator.
      • By an observer.
    — The observer may answer questions and provide hints.
    — Restrict the length of the session to no more than 2 hours.
  - After the session:
    — Hold a debriefing session including observers and design team members where:
      • Each evaluator presents problems detected and the heuristic it violated.
      • A composite problem listing is assembled.
      • Design suggestions for improving the problematic aspects of the system are discussed.
    — After the debriefing session:
      • Generate a composite list of violations as a ratings form.
      • Request evaluators to assign severity ratings to each violation.
      • Analyze results and establish a program to correct violations and deficiencies.
- **Heuristic Evaluation Effectiveness**

- One of the earliest papers addressing the effectiveness of heuristic evaluations was by Nielsen (1992). He reported that the probability of finding a major usability problem averaged 42 percent for single evaluators in six case studies. The corresponding probability for uncovering a minor problem was only 32 percent.
- Heuristic evaluations are useful in identifying many usability problems and should be part of the testing arsenal. Performing this kind of evaluation before beginning actual testing with users will eliminate a number of design problems, and is but one step along the path toward a very usable system.

- Research based set of heuristics

1. Automate unwanted workload. • Free cognitive resources for high-level tasks. • Eliminate mental calculations, estimations, comparisons, and unnecessary thinking.
2. Reduce uncertainty. • Display data in a manner that is clear and obvious.
3. Fuse data. • Reduce cognitive load by bringing together lower-level data into a higher-level summation.
4. Present new information with meaningful aids to interpretation. • Use a familiar framework, making it easier to absorb. • Use everyday terms, metaphors, and so on.
5. Use names that are conceptually related to functions. • Context-dependent. • Attempt to improve recall and recognition.
6. Group data in consistently meaningful ways to decrease search time.
7. Limit data-driven tasks. • Reduce the time needed to assimilate raw data. • Make appropriate use of color and graphics.
8. Include in the displays only that information needed by a user at a given time. • Allow users to remain focused on critical data. • Exclude extraneous information that is not relevant to current tasks.
9. Provide multiple coding of data where appropriate.
10. Practice judicious redundancy. • To resolve the conflict between heuristics 6 and 8.

From Gerhardt-Powals (1996).

Cognitive Walkthroughs

- Description:
  — Reviews of the interface in the context of tasks users perform.
- Advantages:
  — Allow a clear evaluation of the task flow early in the design process.
  — Do not require a functioning prototype.
  — Low cost.
  — Can be used to evaluate alternate solutions.
  — Can be performed by developers.
  — More structured than a heuristic evaluation.
  — Useful for assessing “exploratory learning.”
- Disadvantages:

— Tedious to perform.
— May miss inconsistencies and general and recurring problems.

- Guidelines:
  — Needed to conduct the walkthrough are:
    • A general description of proposed system users and what relevant knowledge they possess.
    • A specific description of one or more core or representative tasks to be performed.
    • A list of the correct actions required to complete each of the tasks.
  — Review:
    • Several core or representative tasks across a range of functions.
    • Proposed tasks of particular concern.
  — Developers must be assigned roles of:
    • Scribe to record results of the action.
    • Facilitator to keep the evaluation moving.
  — Start with simple tasks.
  — Don't get bogged down demanding solutions.
  — Limit session to 60 to 90 minutes.

Think-Aloud Evaluations

- Description:
  — Users perform specific tasks while thinking out load.
  — Comments are recorded and analyzed.
- Advantages:
  — Utilizes actual representative tasks.
  — Provides insights into the user's reasoning.
- Disadvantages:
  — May be difficult to get users to think out loud.
- Guidelines:
  — Develop:
    • Several core or representative tasks.
    • Tasks of particular concern.
  — Limit session to 60 to 90 minutes.

Usability Test

- Description:
  — An interface evaluation under real-world or controlled conditions.
  — Measures of performance are derived for specific tasks.
  — Problems are identified.
- Advantages:
  — Utilizes an actual work environment.
  — Identifies serious or recurring problems.
- Disadvantages:
  — High cost for establishing facility.
  — Requires a test conductor with user interface expertise.
  — Emphasizes first-time system usage.

— Poorly suited for detecting inconsistency problems.

Classic Experiments

- Description:
  — An objective comparison of two or more prototypes identical in all aspects except for one design issue.
- Advantages:
  — Objective measures of performance are obtained.
  — Subjective measures of user satisfaction may be obtained.
- Disadvantages:
  — Requires a rigorously controlled experiment to conduct the evaluation.
  — The experiment conductor must have expertise in setting up, running, and analyzing the data collected.
  — Requires creation of multiple prototypes.
- Guidelines:
  — State a clear and testable hypothesis.
  — Specify a small number of independent variables to be manipulated.
  — Carefully choose the measurements.
  — Judiciously select study participants and carefully or randomly assign them to groups.
  — Control for biasing factors.
  — Collect the data in a controlled environment.
  — Apply statistical methods to data analysis.
  — Resolve the problem that led to conducting the experiment.

Focus Groups

- Description:
  — A discussion with users about interface design prototypes or tasks.
- Advantages:
  — Useful for:
    • Obtaining initial user thoughts.
    • Trying out ideas.
  — Easy to set up and run.
  — Low cost.
- Disadvantages:
  — Requires experienced moderator.
  — Not useful for establishing:
    • How people really work.
    • What kinds of usability problems people have.
- Guidelines:
  — Restrict group size to 8 to 12.
  — Limit to 90 to 120 minutes in length.
  — Record session for later detailed analysis.

Choosing a Testing Method

- Beer, Anodenko, and Sears (1997) suggest a good pairing is cognitive walkthroughs followed by think-aloud evaluations.
- Using cognitive walkthroughs early in the development process permits the identification and correction of the most serious problems. Later, when a functioning prototype is available, the remaining problems can be identified using a think-aloud evaluation.
- A substantial leap forward in the testing process would be the creation of a software tool simulating the behavior of people. This will allow usability tests to be performed without requiring real users to perform the necessary tasks.
- In conclusion, each testing method has strengths and weaknesses. A well-rounded testing program will use a combination of some, or all, of these methods to guarantee the usability of its created product.
- It is very important that testing start as early as possible in the design process and, continue through all developmental stages.

Developing and Conducting the Test

- A usability test requires developing a test plan, selecting test participants, conducting the test, and analyzing the test results.

The Test Plan

- Define the scope of the test.
  - A test's scope will be influenced by a variety of factors.
  - Determinants include the following issues:
  - The *design stage:* early, middle, or late—the stage of design influences the kinds of prototypes that may exist for the test,
  - the *time available* for the test—this may range from just a few days to a year or more,
  - *finances allocated* for testing—money allocated may range from one percent of a project's cost to more than 10 percent,
  - the project's *novelty* (well defined or exploratory)—this will influence the kinds of tests feasible to conduct,
  - expected *user numbers* (few or many) and *interface criticality* (life-critical medical system or informational exhibit)—much more testing depth and length will be needed for systems with greater human impact, and finally, the development team's *experience* and testing knowledge will also affect the kinds of tests that can be conducted.
- Define the purpose of the test.
  — Performance goals.
  — What the test is intended to accomplish.
- Define the test methodology.
  — Type of test to be performed.
  — Test limitations.
  — Developer participants.
- Identify and schedule the test facility or location.

- The location should be away from distractions and disturbances. If the test is being held in a usability laboratory, the test facility should resemble the location where the system will be used.
- It may be an actual office designated for the purpose of testing, or it may be a laboratory specially designed and fitted for conducting tests.

- Develop scenarios to satisfy the test's purpose.

Test Participants

- Assemble the proper people to participate in the test.

Test Conduct and Data Collection

- To collect usable data, the test should begin only after the proper preparation. Then, the data must be properly and accurately recorded.
- Finally, the test must be concluded and followed up properly.

Usability Test Guidelines

- Before starting the test:
  — Explain that the objective is to test the software, not the participants.
  — Explain how the test materials and records will be used.
  — If a consent agreement is to be signed, explain all information on it.
  — If verbal protocols will be collected, let participants practice thinking aloud.
  — Ensure that all participants' questions are answered and that participants are comfortable with all procedures.
- During the test:
  — Minimize the number of people who will interact with the participants.
  — If observers will be in the room, limit them to two or three.
  — Provide a checklist for recording:
    • Times to perform tasks.
    • Errors made in performing tasks.
    • Unexpected user actions.
    • System features used/not used.
    • Difficult/easy-to-use features.
    • System bugs or failures.
  — Record techniques and search patterns that participants employ when attempting to work through a difficulty.
  — If participants are thinking aloud, record assumptions and inferences being made.
  — Record the session with a tape recorder or video camera.
  — Do not interrupt participants unless absolutely necessary.
  — If participants need help, provide some response.
    • Provide encouragement or hints.
    • Give general hints before specific hints.
    • Record the number of hints given.
  — Watch carefully for signs of stress in participants:
    • Sitting for long times doing nothing.
    • Blaming themselves for problems.

    - Flipping through documentation without really reading it.
  — Provide short breaks when needed.
  — Maintain a positive attitude, no matter what happens.
- After the test:
  — Hold a final interview with participants; tell participants what has been learned in the test.
  — Provide a follow-up questionnaire that asks participants to evaluate the product or tasks performed.
  — If videotaping, use tapes only in proper ways.
    - Respect participants' privacy.
    - Get written permission to use tapes.

Analyze, Modify, and Retest

- Compile the data from all test participants.
- List the problems the participants had.
- Sort the problems by priority and frequency.
- Develop solutions for the problems.
- Modify the prototype as necessary.
- Test the system again, and again.

Evaluate the Working System

- Collect information on actual system usage through:
  — Interviews and focus group discussions.
  — Surveys.
  — Support line.
  — Online suggestion box or trouble reporting.
  — Online bulletin board.
  — User newsletters and conferences.
  — User performance data logging.
- Respond to users who provide feedback.

Hypermedia

- **Hypermedia** is used as a logical extension of the term hypertext in which graphics, audio, video, plain text and hyperlinks intertwine to create a generally non-linear medium of information.
- This contrasts with the broader term *multimedia*, which may be used to describe non-interactive linear presentations as well as hypermedia.
- Hypermedia should not be confused with hypergraphics or super-writing which is not a related subject. It is also related to the field of Electronic literature. A term first used in a 1965 article by Ted Nelson.
- The World Wide Web is a classic example of hypermedia, whereas a non-interactive cinema presentation is an example of standard multimedia due to the absence of hyperlinks.

Hypermedia development tools

- Hypermedia may be developed a number of ways. Any programming tool can be used to write programs that link data from internal variables and nodes for external data files.
- Multimedia development software such as Adobe Flash, Adobe Director, Macromedia Authorware, and MatchWare Mediator may be used to create stand-alone hypermedia applications, with emphasis on entertainment content.
- Some database software such as Visual FoxPro and FileMaker Developer may be used to develop stand-alone hypermedia applications, with emphasis on educational and business content management.
- Hypermedia applications may be developed on embedded devices for the mobile and the Digital signage industries using the Scalable Vector Graphics (SVG) specification from W3C (World Wide Web Consortium).
- Software applications such as Ikivo Animator and Inkscape simplify the development of Hypermedia content based on SVG. Embedded devices such as iPhone natively support SVG specifications and may be used to create mobile and distributed Hypermedia applications.
- Hyperlinks may also be added to data files using most business software via the limited scripting and hyperlinking features built in.
- Documentation software such as the Microsoft Office Suite allows for hypertext links to other content within the same file, other external files, and URL links to files on external file servers.
- For more emphasis on graphics and page layout, hyperlinks may be added using most modern desktop publishing tools. This includes presentation programs, such as Microsoft Powerpoint, add-ons to print layout programs such as Quark Immedia, and tools to include hyperlinks in PDF documents such as Adobe InDesign for creating and Adobe Acrobat for editing.
- Hyper Publish is a tool specifically designed and optimized for hypermedia and hypertext management. Any HTML Editor may be used to build HTML files, accessible by any web browser.
- CD/DVD authoring tools such as DVD Studio Pro may be used to hyperlink the content of DVDs for DVD players or web links when the disc is played on a personal computer connected to the internet.

Visualization

- **Visualization** is any technique for creating images, diagrams, or animations to communicate a message.
- Visualization through visual imagery has been an effective way to communicate both abstract and concrete ideas since the dawn of man.
- Examples from history include cave paintings, Egyptian hieroglyphs, Greek geometry, and Leonardo da Vinci's revolutionary methods of technical drawing for engineering and scientific purposes.
- Visualization today has ever-expanding applications in science, education, engineering (e.g. product visualization), interactive multimedia, medicine , etc.
- Typical of a visualization application is the field of computer graphics.

- The invention of computer graphics may be the most important development in visualization since the invention of central perspective in the Renaissance period. The development of animation also helped advance visualization.

Fields of visualization

- A scientific visualization of an extremely large simulation of a Raleigh-Taylor instability caused by two mixing fluids.
- As a subject in computer science, **data visualization** or scientific visualization is the use of interactive, sensory representations, typically visual, of abstract data to reinforce cognition, hypothesis building and reasoning.

Educational visualization

- Educational visualization is using a simulation normally created on a computer to create an image of something so it can be taught about. In the Roman times, this is very useful when teaching about a topic which is difficult to otherwise see, for example, atomic structure, because atoms are far too small to be studied easily without expensive and difficult to use scientific equipment.
- It can also be used to view past events, such as looking at dinosaurs, or looking at things that are difficult or fragile to look at in reality like the human skeleton, without causing physical or mental harm to a subjective volunteer or cadaver.

Information visualization

- Information visualization concentrates on the use of computer-supported tools to explore large amount of abstract data.
- The term "information visualization" was originally coined by the User Interface Research Group at Xerox PARC and included Dr. Jock Mackinlay. Practical application of information visualization in computer programs involves selecting, transforming and representing abstract data in a form that facilitates human interaction for exploration and understanding.
- Important aspects of information visualization are dynamics of visual representation and the interactivity. Strong techniques enable the user to modify the visualization in real-time, thus affording unparalleled perception of patterns and structural relations in the abstract data in question.

Knowledge visualization

- The use of visual representations to transfer knowledge between at least two persons aims to improve the transfer of knowledge by using computer and non-computer based visualization methods complementarily.
- Examples of such visual formats are sketches, diagrams, images, objects, interactive visualizations, information visualization applications and imaginary visualizations as in stories. While information visualization concentrates on the use of computer-supported tools to derive new insights, knowledge visualization focuses on transferring insights and creating new knowledge in groups.

- Beyond the mere transfer of facts, knowledge visualization aims to further transfer insights, experiences, attitudes, values, expectations, perspectives, opinions, and predictions by using various complementary visualizations.

Product Visualization

- Product Visualization involves visualization software technology for the viewing and manipulation of 3D models, technical drawing and other related documentation of manufactured components and large assemblies of products.
- It is a key part of Product Lifecycle Management. Product visualization software typically provides high levels of photorealism so that a product can be viewed before it is actually manufactured. This supports functions ranging from design and styling to sales and marketing.
- *Technical visualization* is an important aspect of product development. Originally technical drawings were made by hand, but with the rise of advanced computer graphics the drawing board has been replaced by computer-aided design (CAD).
- CAD-drawings and models have several advantages over hand-made drawings such as the possibility of 3-D modeling, rapid prototyping and simulation.

Visual communication

- Visual communication is the communication of ideas through the visual display of information. Primarily associated with two dimensional images, it includes: alphanumerics, art, signs, and electronic resources.
- Recent research in the field has focused on web design and graphically oriented usability.

Visual analytics

- Visual analytics focuses on human interaction with visualization systems as part of a larger process of data analysis. Visual analytics has been defined as "the science of analytical reasoning supported by the interactive visual interface" [2].
- Its focus is on human information discourse (interaction) within massive, dynamically changing information spaces. Visual analytics research concentrates on support for perceptual and cognitive operations that enable users to detect the expected and discover the unexpected in complex information space.
- Technologies resulting from visual analytics find their application in almost all fields, but are being driven by critical needs (and funding) in biology and national security.

Visualization techniques

The following are examples of some common visualization techniques:

- Constructing isosurfaces
- direct volume rendering
- Streamlines, streaklines, and pathlines

- table, matrix
- charts (pie chart, bar chart, histogram, function graph, scatter plot, etc.)
- graphs (tree diagram, network diagram, flowchart, existential graph, etc.)
- Maps
- parallel coordinates - a visualization technique aimed at multidimensional data
- treemap - a visualization technique aimed at hierarchical data
- Venn diagram
- Euler diagram
- Chernoff face
- Hyperbolic trees

WWW

The **World Wide Web** (commonly shortened to **the Web**) is a system of interlinked hypertext documents accessed via the Internet. With a Web browser, a user views Web pages that may contain text, images, videos, and other multimedia and navigates between them using hyperlinks.

The World Wide Web was created in 1989 by British scientist Sir Tim Berners-Lee, working at the European Organization for Nuclear Research (CERN) in Geneva, Switzerland, and released in 1992.

Since then, Berners-Lee has played an active role in guiding the development of Web standards (such as the markup languages in which Web pages are composed), and in recent years has advocated his vision of a Semantic Web.

[edit] How it works

Viewing a Web page on the World Wide Web normally begins either by typing the URL of the page into a Web browser, or by following a hyperlink to that page or resource. The Web browser then initiates a series of communication messages, behind the scenes, in order to fetch and display it.

First, the server-name portion of the URL is resolved into an IP address using the global, distributed Internet database known as the domain name system, or DNS. This IP address is necessary to contact and send data packets to the Web server.

The browser then requests the resource by sending an HTTP request to the Web server at that particular address.

In the case of a typical Web page, the HTML text of the page is requested first and parsed immediately by the Web browser, which will then make additional requests for images and any other files that form a part of the page.

Statistics measuring a website's popularity are usually based on the number of 'page views' or associated server 'hits', or file requests, which take place.

Having received the required files from the Web server, the browser then renders the page onto the screen as specified by its HTML, CSS, and other Web languages. Any

images and other resources are incorporated to produce the on-screen Web page that the user sees.

Standards

Many formal standards and other technical specifications define the operation of different aspects of the World Wide Web, the Internet, and computer information exchange.

Many of the documents are the work of the World Wide Web Consortium (W3C), headed by Berners-Lee, but some are produced by the Internet Engineering Task Force (IETF) and other organizations.

Usually, when Web standards are discussed, the following publications are seen as foundational:

- Recommendations for markup languages, especially HTML and XHTML, from the W3C. These define the structure and interpretation of hypertext documents.
- Recommendations for stylesheets, especially CSS, from the W3C.
- Standards for ECMAScript (usually in the form of JavaScript), from Ecma International.
- Recommendations for the Document Object Model, from W3C.

Additional publications provide definitions of other essential technologies for the World Wide Web, including, but not limited to, the following:

- *Uniform Resource Identifier* (URI), which is a universal system for referencing resources on the Internet, such as hypertext documents and images. URIs, often called URLs, are defined by the IETF's RFC 3986 / STD 66: *Uniform Resource Identifier (URI): Generic Syntax*, as well as its predecessors and numerous URI scheme-defining RFCs;
- *HyperText Transfer Protocol (HTTP)*, especially as defined by RFC 2616: *HTTP/1.1* and RFC 2617: *HTTP Authentication*, which specify how the browser and server authenticate each other.

Java

A significant advance in Web technology was Sun Microsystems' Java platform. It enables Web pages to embed small programs (called applets) directly into the view. These applets run on the end-user's computer, providing a richer user interface than simple Web pages.

Java client-side applets never gained the popularity that Sun had hoped for a variety of reasons, including lack of integration with other content (applets were confined to small boxes within the rendered page) and the fact that many computers at the time were supplied to end users without a suitably installed Java Virtual Machine, and so required a download by the user before applets would appear.

Adobe Flash now performs many of the functions that were originally envisioned for Java applets, including the playing of video content, animation, and some rich GUI

features. Java itself has become more widely used as a platform and language for server-side and other programming.

JavaScript

JavaScript, on the other hand, is a scripting language that was initially developed for use within Web pages. The standardized version is ECMAScript.

While its name is similar to Java, JavaScript was developed by Netscape and has very little to do with Java, although the syntax of both languages is derived from the C programming language.

In conjunction with a Web page's Document Object Model (DOM), JavaScript has become a much more powerful technology than its creators originally envisioned.[*citation needed*] The manipulation of a page's DOM after the page is delivered to the client has been called Dynamic HTML (DHTML), to emphasize a shift away from *static* HTML displays.

In simple cases, all the optional information and actions available on a JavaScript-enhanced Web page will have been downloaded when the page was first delivered.

Ajax ("Asynchronous JavaScript and XML") is a group of interrelated web development techniques used for creating interactive web applications that provide a method whereby parts *within* a Web page may be updated, using new information obtained over the network at a later time in response to user actions.

This allows the page to be more responsive, interactive and interesting, without the user having to wait for whole-page reloads. Ajax is seen as an important aspect of what is being called Web 2.0. Examples of Ajax techniques currently in use can be seen in Gmail, Google Maps, and other dynamic Web applications.

Publishing Web pages

Web page production is available to individuals outside the mass media. In order to publish a Web page, one does not have to go through a publisher or other media institution, and potential readers could be found in all corners of the globe.

Many different kinds of information are available on the Web, and for those who wish to know other societies, cultures, and peoples, it has become easier.

The increased opportunity to publish materials is observable in the countless personal and social networking pages, as well as sites by families, small shops, etc., facilitated by the emergence of free Web hosting services

WWW prefix in Web addresses

The letters "www" are commonly found at the beginning of Web addresses because of the long-standing practice of naming Internet hosts (servers) according to the services they provide.

This use of such prefixes is not required by any technical standard; indeed, the first Web server was at "nxoc01.cern.ch",[32] and even today many Web sites exist without a "www" prefix. The "www" prefix has no meaning in the way the main Web site is shown. The "www" prefix is simply one choice for a Web site's host name.

Some Web browsers will automatically try adding "www." to the beginning, and possibly ".com" to the end, of typed URLs if no host is found without them. ll major web browser will also prefix "http://www." and append ".com" to the address bar contents if the Control and Enter keys are pressed simultaneously.

Interface Design Tools

- Tools used for designing the interface, development environments for writing code, and toolkits of graphical user interfaces.

Epark

- EnhancementPak™ (EPak) is an advanced set of OSF/Motif™ widgets extensively tested in hundreds of large scale commercial software applications.
- EPak provides developers with tested, easy-to-use components that eliminate the time, cost, and risk of custom widget development.
- Key Features:
    - Windows95-like controls
    - Easy to use geometry managers and containers
    - Data presentation, application
    - Binary and source code available
    - No royalties or runtime fees

Converter

- At Integrated Computer Solutions we are committed to your success.
- That's why we offer several products that will let you migrate from software that does not meet your needs to the state-of-the-art, industry standard that will help you retain the value of your code.
- Our code conversion is more than simply translating to UIL, you need to know how to avoid creating more problems than you solve.

BX PRO

- Builder Xcessory PRO™ (BX PRO) provides C, C++ or Java developers with everything they need to develop and manage GUI projects of any size from start to finish.
- Benefits

    - Speed GUI developlment
    - Eliminate unmaintainable code
    - Quickly implement changes

ICE Interface

- ICEfaces is a standards-compliant extension to JavaServer Faces (JSF) for building and deploying rich web applications. ICEfaces™ creates rich user interaction and provides superior presentation characteristics like:
- Smooth, incremental page updates with in-place editing and no full page refresh
- User context preservation during page update, including scrollbar positioning and user focus
- Asynchronous page updates driven from the application in real time
- Fine-grained user interaction during form entry that augments the standard submit/response loop
- ICEfaces is an industrial strength solution that leverages the JSF application framework and the entire J2EE ecosystem of tools and execution environments. Rich application features are developed in pure Java, and in a pure thin-client model.
- There are no Applets, browser plugins, or JavaScript-ladened pages thanks to ICEfaces' breakthrough Direct-to-DOMTM rendering technology. And because ICEfaces applications are JSF applications, your J2EE and JSF development skills apply directly with no learning curve.

Software Testing Tools

Quick Test Professional (QTP)

Quick Test Professional (QTP) is an automated functional Graphical User Interface (GUI) testing tool that allows the automation of user actions on a web or client based computer application. It is primarily used for functional regression test automation. QTP uses a scripting language built on top of VBScript to specify the test procedure, and to manipulate the objects and controls of the application under test.

WinRunner

WinRunner, Mercury Interactive's enterprise functional testing tool. It is used to quickly create and run sophisticated automated tests on your application. Winrunner helps you automate the testing process, from test development to execution. You create adaptable and reusable test scripts that challenge the functionality of your application. Prior to a

software release, you can run these tests in a single overnight run- enabling you to detect and ensure superior software quality.

LoadRunner

LoadRunner is divided up into 3 smaller applications:

The Virtual User Generator allows us to determine what actions we would like our Vusers, or virtual users, to perform within the application. We create scripts that generate a series of actions, such as logging on, navigating through the application, and exiting the program.

The Controller takes the scripts that we have made and runs them through a schedule that we set up. We tell the Controller how many Vusers to activate, when to activate them, and how to group the Vusers and keep track of them.

The Results and Analysis program gives us all the results of the load test in various forms. It allows us to see summaries of data, as well as the details of the load test for pinpointing problems or bottlenecks.

TestDirector

TestDirector, the industry's first global test management solution, helps organizations deploy high-quality applications more quickly and effectively. Its four modules Requirements, Test Plan, Test Lab, and Defects are seamlessly integrated, allowing for a smooth information flow between various testing stages. The completely Web-enabled TestDirector supports high levels of communication and collaboration among distributed testing teams, driving a more effective, efficient global application-testing process

Silk Test

Silk Test is a tool specifically designed for doing **REGRESSION AND FUNCTIONALITY** testing. It is developed by Segue Software Inc. Silk Test is the industry's leading functional testing product for e-business applications, whether Window based, Web, Java, or traditional client/server-based. Silk Test also offers test planning, management, direct database access and validation, the flexible and robust 4Test scripting language, a built in recovery system for unattended testing, and the ability to test across multiple platforms, browsers and technologies.

You have two ways to create automated tests using silktest:

1. Use the **Record Testcase command** to record actions and verification steps as you navigate through the application.

2. Write the testcase manually using the **Visual 4Test scripting language**.

1. Record Testcase

The Record / Testcase command is used to record actions and verification steps as you navigate through the application. Tests are recorded in an object-oriented language **called Visual 4Test**. The recorded testreads like a logical trace of all of the steps that were completed by the user. The Silk Test point and click verification system allows you to record the verification step by selecting from a list of properties that are appropriate for the type of object being tested. For example, you can verify the text is stored in a text field.

2. Write the Testcase manually

We can write tests that are capable of accomplishing many variations on a test. The key here is re-use. A test case can be designed to take parameters including input data and expected results. This "data-driven" testcase is really an instance of a class of test cases that performs certain steps to drive and verify the application-under-test. Each instance varies by the data that it carries. Since far fewer tests are written with this approach, changes in the GUI will result in reduced effort in updating tests. A data-driven test design also allows for the externalization of testcase data and makes it possible to divide the responsibilities for developing testing requirements and for developing test automation. For example, it may be that a group of domain experts create the Testplan Detail while another group of test engineers develop tests to satisfy those requirements.

In a script file, an automated testcase ideally addresses one test requirement. Specifically, a 4Test function that begins with the test case keyword and contains a sequence of 4Test statements. It drives an application to the state to be tested, verifies that the application works as expected, and returns the application to its base state.

A script file is a file that contains one or more related testcases. A script file has a .t extension, such as find .t

Rational Robot

Rational Robot is a complete set of components for automating the testing of Microsoft Windows client/server and Internet applications running under Windows NT 4.0, Windows 2000, Windows 98, and Windows 95.

The main component of Robot lets you start recording tests in as few as two mouse clicks. After recording, Robot plays back the tests in a fraction of the time it would take to repeat the actions manually.

Other components of Robot are:

- Rational Administrator Use to create and manage Rational projects, which store your testing information.
- Rational TestManager Log Use to review and analyze test results.
- Object Properties, Text, Grid, and Image Comparators Use to view and analyze the results of verification point playback.
- Rational SiteCheck Use to manage Internet and intranet Web sites.