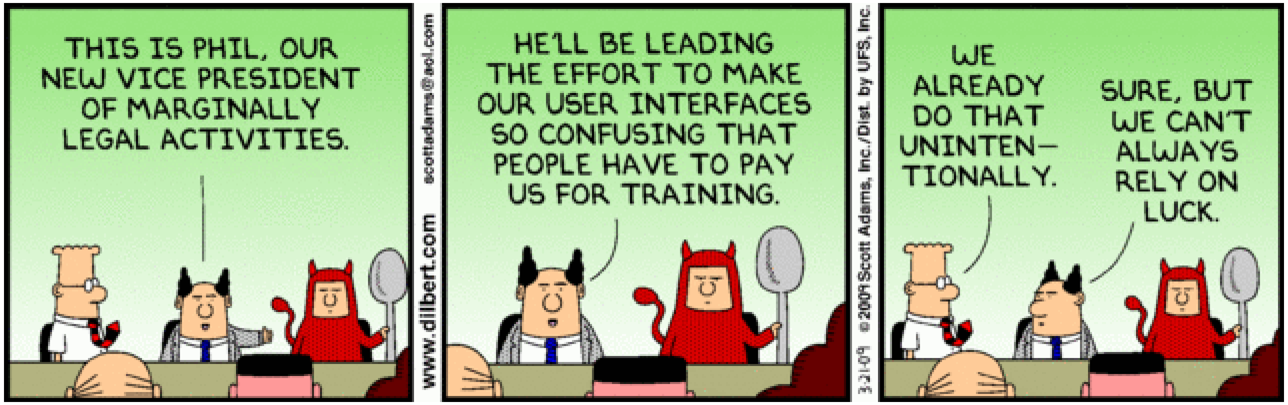
# USABILITY ENGINEERING& EVALUATION

Lecture 7

# Dillbert … on usability


THIS IS PHIL, OUR NEW VICE PRESIDENT OF MARGINALLY LEGAL ACTIVITIES.
HE'LL BE LEADING THE EFFORT TO MAKE OUR USER INTERFACES SO CONFUSING THAT PEOPLE HAVE TO PAY US FOR TRAINING.
WE ALREADY DO THAT UNINTENTIONALLY.
SURE, BUT WE CAN'T ALWAYS RELY ON LUCK.
www.dilbert.com
scottadams@aol.com
3-21-09 © 2009 Scott Adams, Inc./Dist. by UFS, Inc.

# Jakob Nielsen…

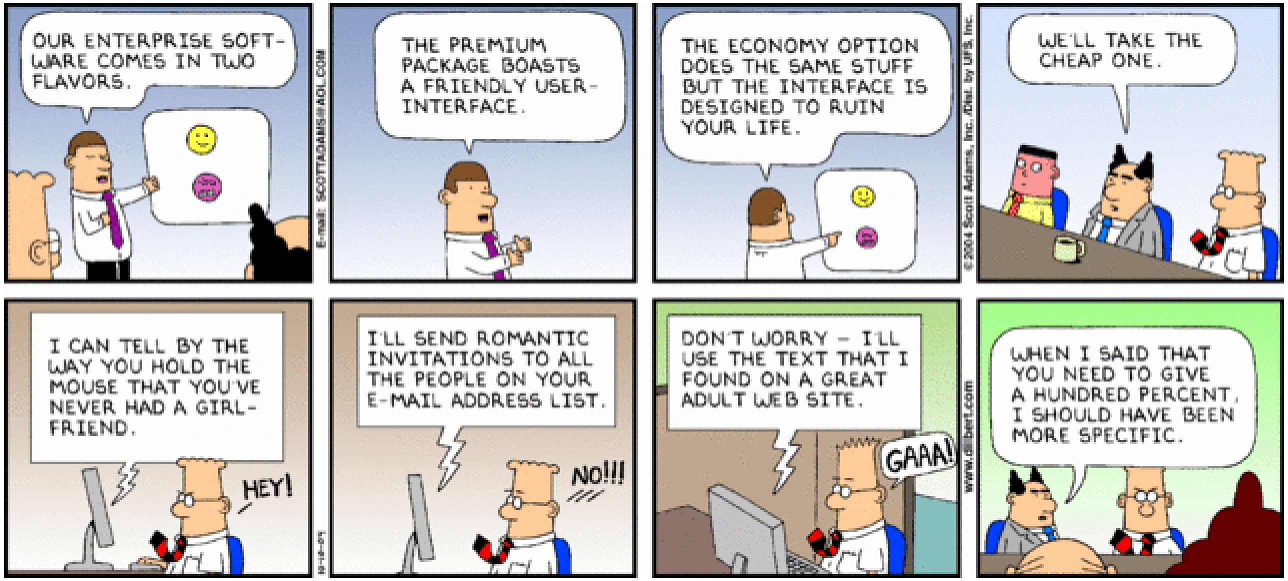
- **Q:** *How many programmers does it take to change a light bulb?*
  **A:** *None; it is a hardware problem!*

- When asking how many usability specialists it takes to change a light bulb, the answer might well be four:
  - Two to conduct a field study and task analysis to determine whether people really need light,
  - one to observe the user who actually screws in the light bulb,
  - and one to control the video camera filming the event.

# Usability benefits

- **The benefits of good web design by examples**
  - **Creative Good**
  - Creative Good offered the striking revelation that a dollar spent on advertising during the 1998 holiday season produced $5 in total revenue, while **a dollar spent on customer experience improvements yielded more than $60**.
  - **IBM**
  - On IBM's website, the most popular feature was the **search** function, because the site was difficult to navigate. The second most popular feature was the **'help**' button, because the search technology was so ineffective. IBM's solution was a 10-week effort to redesign the site, which involved more than 100 employees at a cost estimated 'in the millions.' The result: In the first week after the redesign, use of the 'help' button decreased **84** per cent, while sales increased **400** per cent.
  - **Jakob Nielsen**
  - Alert Box, June 2000. It's quite normal for e-commerce **sites to increase sales by 100% or more as a result of usability**

# www.dillbert.com


OUR ENTERPRISE SOFTWARE COMES IN TWO FLAVORS.
THE PREMIUM PACKAGE BOASTS A FRIENDLY USER-INTERFACE.
THE ECONOMY OPTION DOES THE SAME STUFF BUT THE INTERFACE IS DESIGNED TO RUIN YOUR LIFE.
WE'LL TAKE THE CHEAP ONE.
I CAN TELL BY THE WAY YOU HOLD THE MOUSE THAT YOU'VE NEVER HAD A GIRLFRIEND.
HEY!
I'LL SEND ROMANTIC INVITATIONS TO ALL THE PEOPLE ON YOUR E-MAIL ADDRESS LIST.
NO!!!
DON'T WORRY – I'LL USE THE TEXT THAT I FOUND ON A GREAT ADULT WEB SITE.
GAAA!
WHEN I SAID THAT YOU NEED TO GIVE A HUNDRED PERCENT, I SHOULD HAVE BEEN MORE SPECIFIC.
E-mail: SCOTTADAMS@AOL.COM
© 2004 Scott Adams, Inc./Dist. by UFS, Inc.
www.dilbert.com

# Usability Engineering

- usability engineering describes a process of user interface development, sometimes referred to as **user centered design.**

- a lifecycle process that puts an early emphasis on user and task analysis and actual user involvement in the design and testing of a product

- a product developed with such a user centered process is likely to be a more usable product than one that is developed independent of user considerations and involvement.

# Usability Evaluation

- Usability evaluation is itself a process that entails many activities depending on the method employed.
- Common activities include:
  - **Capture:** collecting usability data, such as task completion time, errors, guideline violations, and subjective ratings;
  - **Analysis:** interpreting usability data to identify usability problems in the interface;
  - **Critique**: suggesting solutions or improvements to mitigate problems.

# Usability Evaluation Methods

- **Testing**: an evaluator observes users interacting with an interface (i.e., completing tasks) to determine usability problems.
- **Inspection**: an evaluator uses a set of criteria or heuristics to identify potential usability problems in an interface.
- **Inquiry**: users provide feedback on an interface via interviews, surveys.
- **Analytical Modeling**: an evaluator employs user and interface models to generate usability predictions.
- **Simulation**: an evaluator employs user and interface models to mimic a user interacting with an interface and report the results of this interaction

# Usability metrics

- A **metric** is a way of measuring or evaluating a particular phenomenon or thing
- We can say something is longer, taller, or faster because we are able to measure or quantify some attribute of it, such as distance, height, or speed
- Every industry, activity, and culture has its own set of metrics (auto industry: horse power, gas milage; computers: processor speed, memory)
- usability metrics are based on a reliable system of measurement:
- using the same set of measurements each time something is measured should result in comparable outcomes

# Usability metrics

- Must be:
    - **Observable** – directly or indirectly (a task has completed)
    - **Quantifiable** - they have to be turned into a number or counted in some way
    - the thing being measured **represent some aspect of the user experience**, presented in a numeric format (65 percent of the users are satisfied with using a product, or that 90 percent of the users are able to complete a set of tasks in less than one minute)

- A usability metric reveals something about the interaction between the user and the thing:
    - **effectiveness** (being able to complete a task),
    - **efficiency** (the amount of effort required to complete the task)
    - **satisfaction** (the degree to which the user was happy with his or her experience while performing the task).

- measure something about people and their behavior or attitudes

# Usability metrics

- Usability metrics can answer these critical questions:
  - Will the users like the product?
  - Is this new product more efficient to use than the current product?
  - How does the usability of this product compare to the competition?
  - What are the most significant usability problems with this product?
  - Are improvements being made from one design iteration to the next?

# Usability metrics

- Usability metrics offer a way to estimate the number of users likely to experience a typical problem

- Knowing the magnitude of the problem could mean the difference between delaying a major product launch and simply adding an additional item to the bug list with a low priority.

- Without usability metrics, the magnitude of the problem is just a guess.

- Usability metrics show whether you’re actually improving the user experience from one product to the next

# Usability metrics

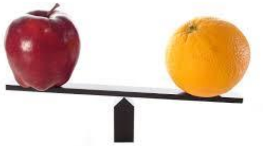
- Usability metrics are a key ingredient in calculating a ROI
- As part of a business plan, you may be asked to determine how much money is saved or how revenue increases as a result of a new product design.
- Without usability metrics, this task is impossible.
- With usability metrics, you might determine that a simple change in a data input field on an internal website could reduce data entry errors by 75 percent, reduce the time required to complete the customer service task, increase the number of transactions processed each day, reduce the backlog in customer orders, cut the delay in customer shipments, and increase both customer satisfaction and customer orders, resulting in an overall rise in revenue for the company.

# Designing a usability study

- Questions you have to answer:
  - What type of participants do I need?

  - How many participants do I need?

  - Am I going to compare the data from a single group of participants or from several different groups?



  - Do I need to counterbalance (adjust for) the order of tasks?

# Selecting participants

- Decisions based on factors such as:
    - cost
    - availability
    - appropriateness
    - study goals

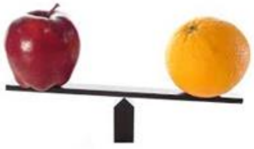
- Questions:
  - **how well your participants should reflect your target audience**
  - **are you going to divide your data by different types of participants.**
    - If you plan to separate your participants into distinct groups, think about what those groups are and about how many participants you want in each group.
    - few common types of groups or segments:
      - Self-reported expertise in some domain (novice, intermediate, expert)
      - Frequency of use (e.g., number of web visits or interactions per month)
      - Amount of experience with something relevant (days, months, years)
      - Demographics (gender, age, location, etc.)
      - Activities (use of particular functionality or features)
  - **sampling strategy**: random, systematic, stratified, of convenience (anyone willing to participate)

# Sample size

- There's no rule that says if you don't have at least x number of participants in a study, the data won't be valid.

- The sample size should be based on two factors:
  - the **goals** of your study and
  - your **tolerance** for a margin of error.
- If you are interested **only in identifying major usability issues** as part of an iterative design process, you can get useful feedback from **three or four representative participants**
- During the early stages of design, you need fewer participants to identify the major usability issues
- As the design gets closer to completion, you need more participants to identify the remaining issues

# Comparing data



- Within-Subjects or Between-Subjects Study?

- **Within-subjects** – comparing different data for each participant (such as success rates for different designs of the product)

- **Between-Subjects** – comparing data from each participant to the other participants (such as success rates for different age groups).

# Within-subjects

- within-subjects (repeated measure) - when you want **to evaluate how easily a participant can learn to use a particular product.**
- By comparing metrics such **as task completion times** or **errors** across several trials with the same set of participants, you can determine how quickly and easily the participant becomes familiar with the product
- does not require as large a sample size, and you don’t have to worry about differences across groups.
- Because each participant is being compared to himself, the differences you observe in the data cannot be attributed to differences between participants
- Possible disadvantage: “carryover effects,” where performance in one condition impacts performance in another condition - might be the result of practice (improving performance) or fatigue (decreasing performance).

# Between-subjects

- used to compare results for different participants, such as differences in satisfaction between novices and experts or in task completion times for younger versus older participants

- Participants are randomly assigned to groups that receive different treatments: different designs of the same product

- requires a larger sample size

- the elimination of carryover effects, because any potential carryover effects would impact both groups equally.

# Mixed- design

- A mixed design contains a between-subjects factor, such as gender, and a within-subjects factor, such as three trials distributed over time.

- For example, you could use a mixed-design study to find out if there is a difference in the way men and women perform some task across several trials

# Counterbalancing

- Sometimes the order in which participants perform their tasks has a significant impact on the results.

- Participants usually learn the product as their experience with it grows

- you must consider the order in which the data are collected, which is usually the order of tasks. It’s possible that you may see improvement in performance or satisfaction as the usability session continues.

- Can help you determine if the improvement occurs because the fifth task is easier than the first task or if some learning takes place between the first and fifth tasks that makes the fifth task easier to perform

# Counterbalancing

- Counterbalancing involves simply changing the order in which different tasks are performed

- You can randomize the order of tasks by “shuffling” the task order prior to each participant

- you can create various orders ahead of time so that each participant performs each task in a different order

# Counterbalancing

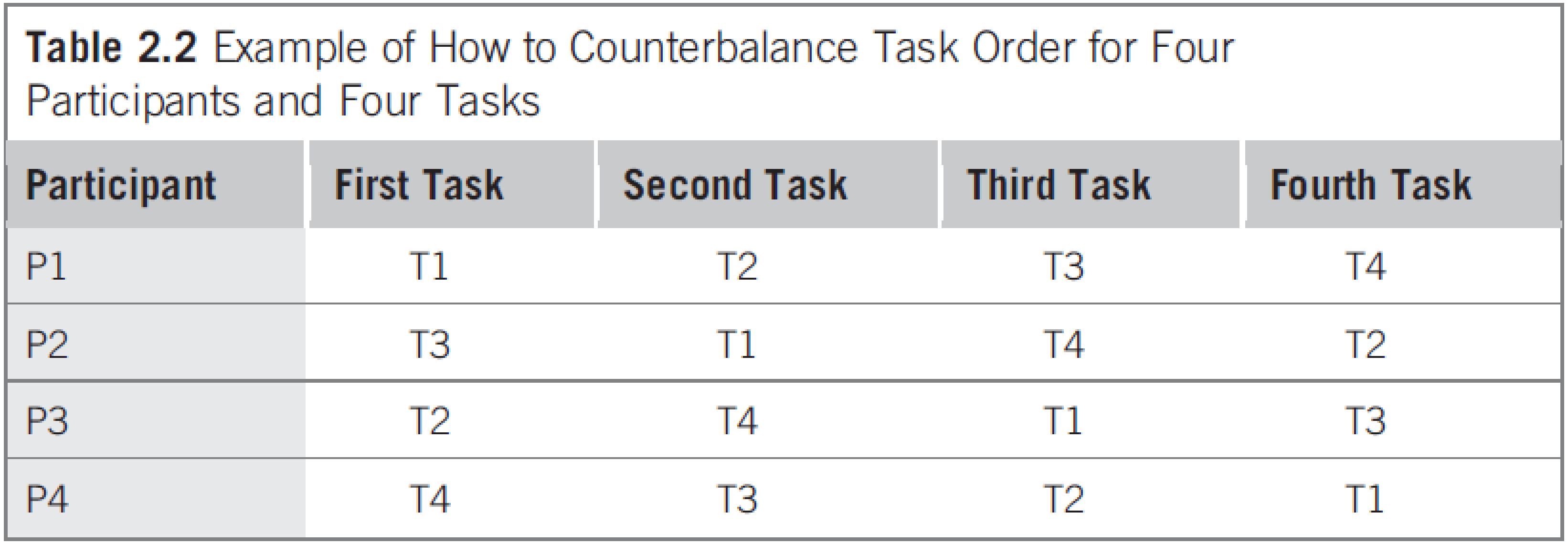


**Table 2.2** Example of How to Counterbalance Task Order for Four Participants and Four Tasks

| Participant | First Task | Second Task | Third Task | Fourth Task |
|---|---|---|---|---|
| P1 | T1 | T2 | T3 | T4 |
| P2 | T3 | T1 | T4 | T2 |
| P3 | T2 | T4 | T1 | T3 |
| P4 | T4 | T3 | T2 | T1 |

# Counterbalancing

- if the tasks are totally unrelated to each other, learning between tasks is unlikely
- counterbalancing is not appropriate when a natural order of tasks is present
- Sometimes the order cannot be juggled because the test session would not make sense.

# Independent and dependent variables

- An independent variable of a study is an aspect that you manipulate.
- Choose the independent variables based on your research questions.
- For example, you may be concerned with differences in performance between males and females, or between novices and experts, or between two different designs.
- Dependent variables (also called outcome or response variables) describe what happened as the result of the study.
- A dependent variable is something you measure as the result of, or as dependent on, how you manipulate the independent variables.
- Dependent variables include metrics or measurements such as success rates, number of errors, user satisfaction, completion times, and many more
- you must have a clear idea of what you plan to manipulate (independent variables) and what you plan to measure (dependent variables).

# Types of data

- 4 types of data:
  - nominal
  - ordinal
  - interval
  - ratio

# Nominal data

- are simply **unordered** groups or categories

- might be characteristics of different types of users, such as Windows versus Mac users, users in different geographic locations, or males as opposed to females.

- typically independent variables that allow you to segment the data by these different groups.

- Nominal data also include some commonly used dependent variables, such as task success.

- Nominal data could also be the number of participants who clicked on link A instead of link B, or participants who chose to use a remote control instead of the controls on a DVD player itself

# Nominal data

- counts and frequencies: for example, you could say that 45 percent of the participants are female, or 200 participants have blue eyes, or 95 percent were successful on a particular task

- when you work with nominal data is important how you code the data:
  - Female -1 ,male – 0 -  average has no meaning
  - Task success – 1/0 – average could have meaning

# Ordinal data

- ordinal data are **ordered** groups or categories
- the data are organized in a certain way.
- the intervals between the measurements are not meaningful (if something is on the 1st position and other thing is on the 4th position, you can not say that the first object is 4 times better than the second)
- ordinal data comes from self reported data on questionnaires. For example, a participant might rate a website as **excellent, good, fair, or poor** – relative rankings (the distance between excellent and good is not necessarily the same distance between good and fair)
- severity ratings are another example of ordinal data
- the most common way to analyze ordinal data is by looking at **frequencies**. For example, you might report that 40 percent of the participants rated the site as excellent, 30 percent as good, 20 percent as fair, and 10 percent as poor.
- **calculating an average ranking may be tempting, but it's statistically meaningless**.

# Interval data

- Interval data are continuous data **where the differences between the measurements are meaningful** but there is no natural zero point.

- An example: **temperature**, either Celsius or Fahrenheit

- Interval data allow you to calculate a wide range of descriptive statistics (including averages, standard deviation, etc.).

# Ratio data

- Ratio data are the same as **interval data, with the addition of an absolute zero.**
- the zero value is not arbitrary, as with interval data, but has some inherent meaning.
- With ratio data, the differences between the measurements are interpreted as a ratio.
- Examples of ratio data are **age, height, and weight**.
- In each example, **zero indicates the absence of age, height, or weight**.
- Example: time to completion – zero seconds left would mean no time or duration remaining.
- Ratio data let you say something is twice as fast or half as slow as something else.
- For example, you could say that one participant is twice as fast as another user in completing a task.
- Calculations similar to interval data + geometric mean - measuring differences in time

# Data types - synthesis

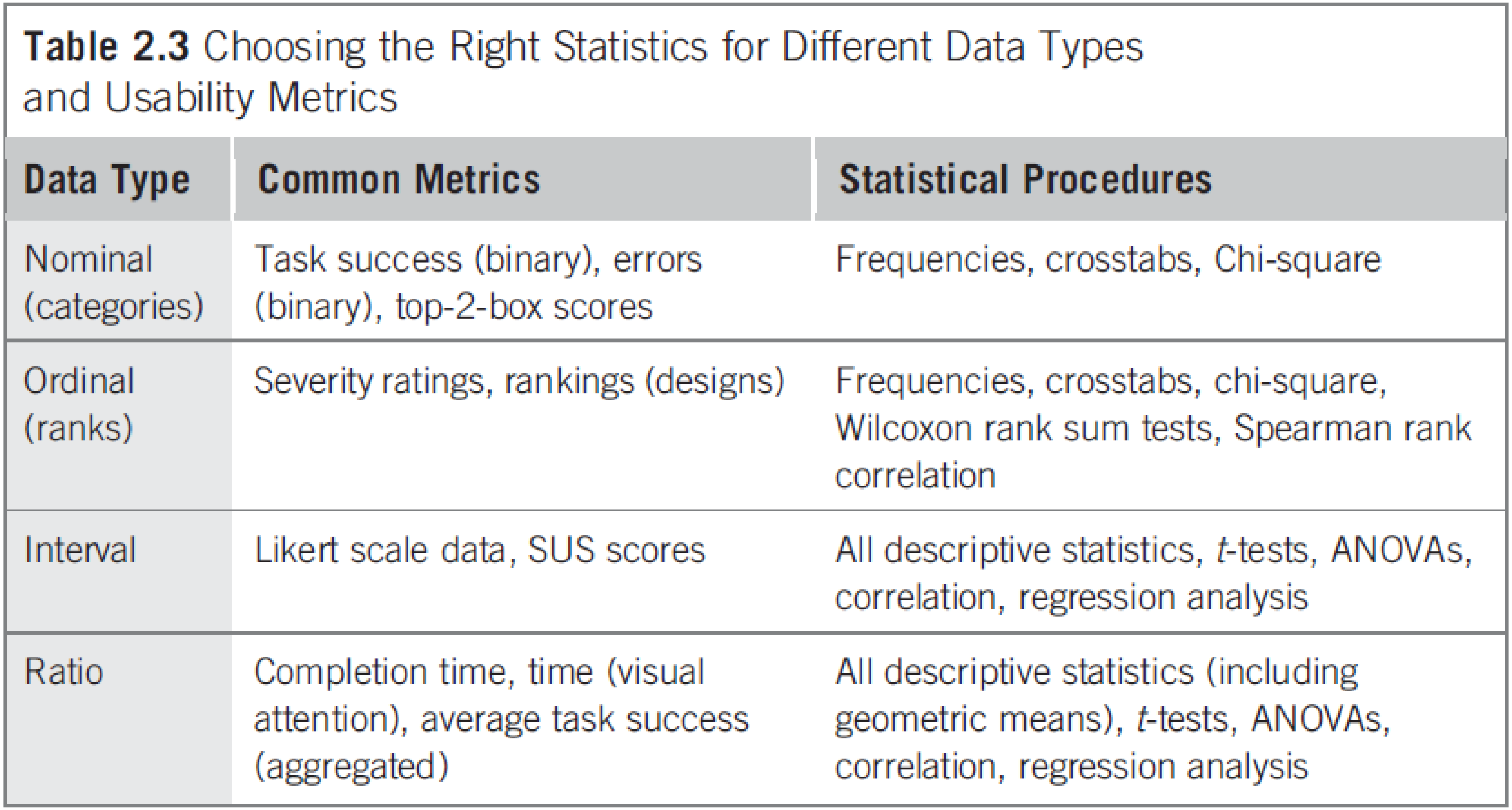
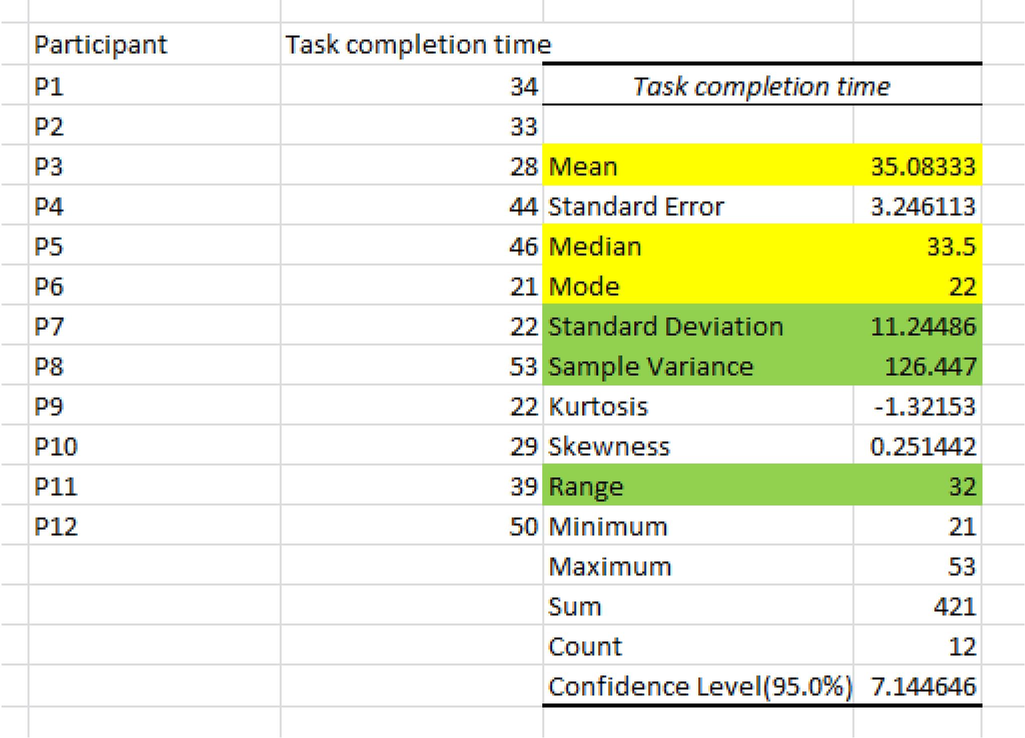


**Table 2.3** Choosing the Right Statistics for Different Data Types and Usability Metrics

| Data Type | Common Metrics | Statistical Procedures |
|---|---|---|
| Nominal (categories) | Task success (binary), errors (binary), top-2-box scores | Frequencies, crosstabs, Chi-square |
| Ordinal (ranks) | Severity ratings, rankings (designs) | Frequencies, crosstabs, chi-square, Wilcoxon rank sum tests, Spearman rank correlation |
| Interval | Likert scale data, SUS scores | All descriptive statistics, *t*-tests, ANOVAs, correlation, regression analysis |
| Ratio | Completion time, time (visual attention), average task success (aggregated) | All descriptive statistics (including geometric means), *t*-tests, ANOVAs, correlation, regression analysis |

# Descriptive statistics in Excel



| Participant | Task completion time | | |
|---|---|---|---|
| P1 | 34 | *Task completion time* | |
| P2 | 33 | | |
| P3 | 28 | Mean | 35.08333 |
| P4 | 44 | Standard Error | 3.246113 |
| P5 | 46 | Median | 33.5 |
| P6 | 21 | Mode | 22 |
| P7 | 22 | Standard Deviation | 11.24486 |
| P8 | 53 | Sample Variance | 126.447 |
| P9 | 22 | Kurtosis | -1.32153 |
| P10 | 29 | Skewness | 0.251442 |
| P11 | 39 | Range | 32 |
| P12 | 50 | Minimum | 21 |
| | | Maximum | 53 |
| | | Sum | 421 |
| | | Count | 12 |
| | | Confidence Level(95.0%) | 7.144646 |

Central tendency measures

Variability measures

# Central tendency measures

- The **mean** of most usability metrics is extremely useful and is probably the most common statistic cited in a usability report.
- The **median** is the midway point in the distribution: Half the participants are below the median and half are above the median – example: the median is equal to 33.5 seconds: Half of the participants were faster than 33.5 seconds, and half of the participants were slower than 33.5 seconds.
- In some cases, the median can be more revealing than the mean (salaries, median salaries for a company are more commonly reported because the higher executive salaries will skew the mean value so much that the average salary appears much higher than the majority really are. In such cases involving possible extreme values (as is sometimes the case with time data), consider using the median.
- The **mode** is the value that appears most often in a set of data
- The **mode** is the most commonly occurring value. In example  the mode is 22 seconds: Two participants completed the task in 22 seconds.
- It's not common to report the mode in usability test results, but it may be useful to know it.

# Variability measures

- Show how the data are spread or dispersed across the range of all the data.

- help answer the question “Do most users have similar completion times, or is there a wide range of times?”

- Determining the variability is critical if you want to know how confident you can be of the data

- The less the variability or spread, the more confidence you can have in relating the findings to a larger population.

- There are three common measures of variability: the range, the variance, and the standard deviation.

# The range

- The range is the distance between the minimum and maximum data points
- When you study completion times, the range is very important because it will identify “outliers” (data points that are at the extreme top and bottom of the range)
- Looking at the range is also a good check to make sure that the data are coded properly (if the range is supposed to be from one to five, and the data include a seven, you know there is a problem)

# Variance

- tells you how spread out the data are relative to the average or mean

- The formula for calculating variance measures the difference between each individual data point and the mean, squares that value, sums all of those squares, and then divides the result by the sample size minus 1

# Standard deviation

- The standard deviation is simply the square root of the variance.

- The standard deviation in the example shown 11 seconds.

- Interpreting this measure of variability is a little easier than interpreting the variance, since the unit of the standard deviation is the same as the original data (seconds in this example).

# Usability metrics

| **Usability objective** | **Effectiveness measures** | **Efficiency measures** | **Satisfaction measures** |
|---|---|---|---|
| Suitability for the task | Percentage of goals achieved | Time to complete a task | Rating scale for satisfaction |
| Appropriate for trained users | Number of power features used | Relative efficiency compared with an expert user | Rating scale for satisfaction with power features |
| Learnability | Percentage of functions learned | Time to learn criterion | Rating scale for ease of learning |
| Error tolerance | Percentage of errors corrected successfully | Time spent on correcting errors | Rating scale for error handling |

# Automatic Usability Evaluation

- Usability findings can vary widely when different evaluators study the same user interface, even if they use the same evaluation technique
- Less than a 1% overlap in findings among four and eight independent usability testing teams for evaluations of two user interfaces.
- a lack of systematic approach or predictability in the findings of usability evaluations
- usability evaluation typically only covers a subset of the possible actions users might take - usability experts often recommend using several different evaluation techniques
- Solutions:
  - Increase the number of evaluators and participant`s to cover more aspects of the system
  - Automate some aspects of usability evaluation

# Automatic Usability Evaluation Advantages

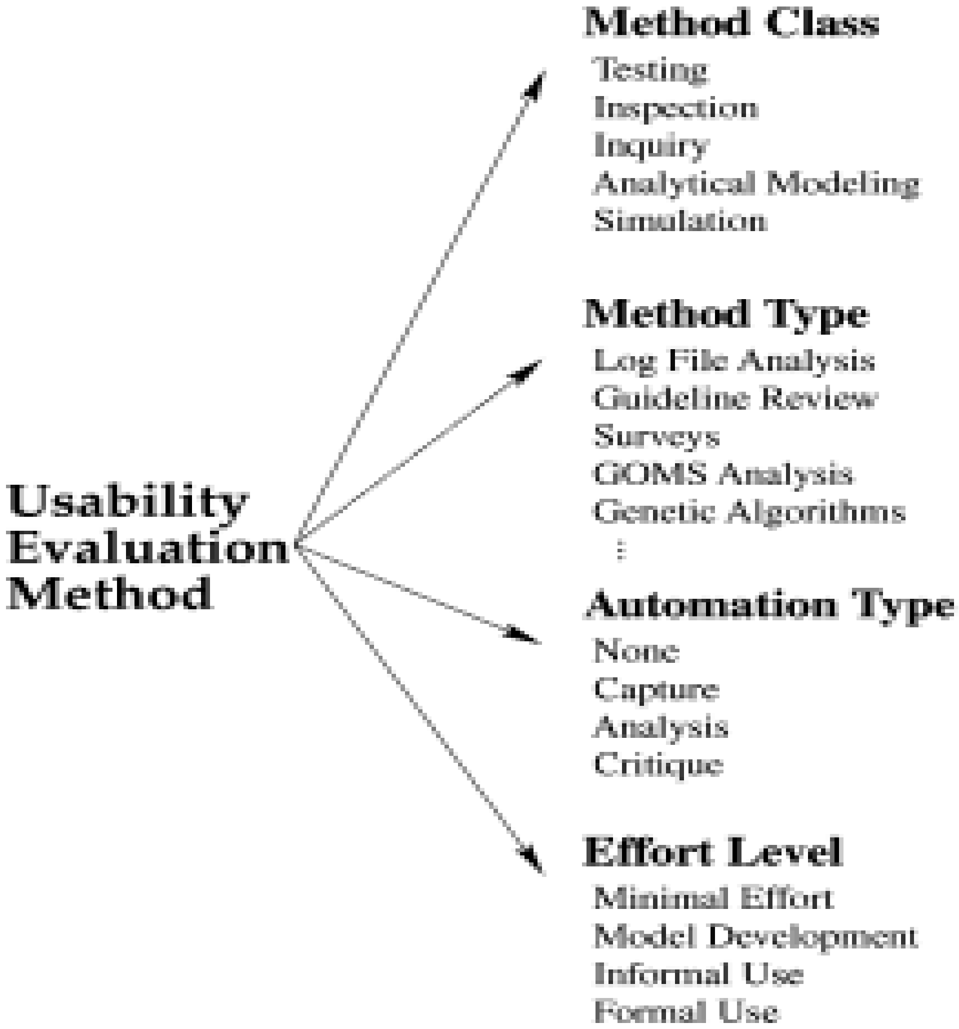
- Reducing the cost of usability evaluation (time and money spent)- logging tools
- Increasing consistency of the errors uncovered –
  - It is possible to develop models of task completion within an interface, and software tools can consistently detect deviations from these models
  - It is also possible to detect usage patterns that suggest possible errors, such as immediate task cancellation
- Reducing the need for evaluation expertise among individual evaluators.
- Increasing the coverage of evaluated features
- Enabling comparisons between alternative designs
- Incorporating evaluation within the design phase of UI development, as opposed to being applied after implementation

# Automatic Usability Evaluation

- automation to be a useful complement and addition to standard evaluation techniques such as heuristic evaluation and usability testing—not a substitute

- Some aspects of usability can not be automatically measured : satisfaction

# Taxonomy for Usability Evaluation Methods (Ivory, Hearst)



Usability Evaluation Method

**Method Class**
- Testing
- Inspection
- Inquiry
- Analytical Modeling
- Simulation

**Method Type**
- Log File Analysis
- Guideline Review
- Surveys
- GOMS Analysis
- Genetic Algorithms
- ⋮

**Automation Type**
- None
- Capture
- Analysis
- Critique

**Effort Level**
- Minimal Effort
- Model Development
- Informal Use
- Formal Use

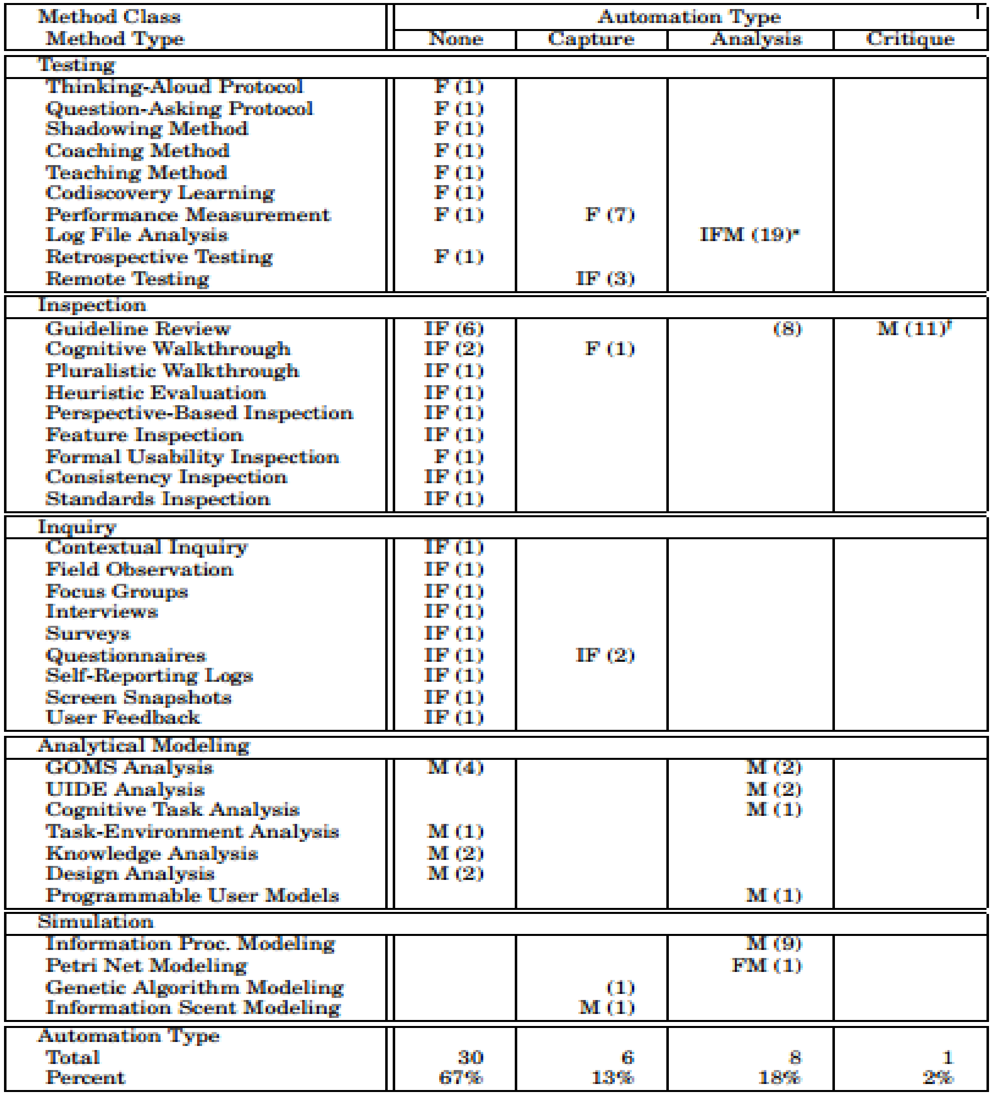
# Taxonomy for Usability Evaluation

- **Automation Type**: used to specify which aspect of a usability evaluation method is automated.
    - **None**: no level of automation supported (i.e., evaluator performs all aspects of the evaluation method);
    - **Capture**: software automatically records usability data (e.g., logging interface usage);
    - **Analysis**: software automatically identifies potential usability problems; and
    - **Critique**: software automates analysis and suggests improvement

# Taxonomy for Usability Evaluation

- **Effort level** - indicates the human effort required for method execution:
  - **Minimal Effort**: does not require interface usage or modeling.
  - **Model Development**: requires the evaluator to develop a UI model and/or a user model in order to employ the method.
  - **Informal Use**: requires completion of freely chosen tasks (i.e., unconstrained use by a user or evaluator).
  - **Formal Use**: requires completion of specially selected tasks (i.e., constrained use by a user or evaluator).

# Automation Support for WIMP and Web UE Methods



| Method Class | Automation Type | | | |
|---|---|---|---|---|
| Method Type | None | Capture | Analysis | Critique |
| **Testing** | | | | |
| Thinking-Aloud Protocol | F (1) | | | |
| Question-Asking Protocol | F (1) | | | |
| Shadowing Method | F (1) | | | |
| Coaching Method | F (1) | | | |
| Teaching Method | F (1) | | | |
| Codiscovery Learning | F (1) | | | |
| Performance Measurement | F (1) | F (7) | | |
| Log File Analysis | | | IFM (19)* | |
| Retrospective Testing | F (1) | | | |
| Remote Testing | | IF (3) | | |
| **Inspection** | | | | |
| Guideline Review | IF (6) | | (8) | M (11)† |
| Cognitive Walkthrough | IF (2) | F (1) | | |
| Pluralistic Walkthrough | IF (1) | | | |
| Heuristic Evaluation | IF (1) | | | |
| Perspective-Based Inspection | IF (1) | | | |
| Feature Inspection | IF (1) | | | |
| Formal Usability Inspection | F (1) | | | |
| Consistency Inspection | IF (1) | | | |
| Standards Inspection | IF (1) | | | |
| **Inquiry** | | | | |
| Contextual Inquiry | IF (1) | | | |
| Field Observation | IF (1) | | | |
| Focus Groups | IF (1) | | | |
| Interviews | IF (1) | | | |
| Surveys | IF (1) | | | |
| Questionnaires | IF (1) | IF (2) | | |
| Self-Reporting Logs | IF (1) | | | |
| Screen Snapshots | IF (1) | | | |
| User Feedback | IF (1) | | | |
| **Analytical Modeling** | | | | |
| GOMS Analysis | M (4) | | M (2) | |
| UIDE Analysis | | | M (2) | |
| Cognitive Task Analysis | | | M (1) | |
| Task-Environment Analysis | M (1) | | | |
| Knowledge Analysis | M (2) | | | |
| Design Analysis | M (2) | | | |
| Programmable User Models | | | M (1) | |
| **Simulation** | | | | |
| Information Proc. Modeling | | | M (9) | |
| Petri Net Modeling | | | FM (1) | |
| Genetic Algorithm Modeling | | (1) | | |
| Information Scent Modeling | | M (1) | | |
| **Automation Type** | | | | |
| Total | 30 | 6 | 8 | 1 |
| Percent | 67% | 13% | 18% | 2% |

a

A number in parentheses indicates the number of UE methods surveyed for a particular method type and automation type. The effort level for each method is represented as: minimal (blank), formal (F), informal (I), and model (M).

*

Indicates that either formal or informal interface use is required. In addition, a model may be used in the analysis.

†

Indicates that methods may or may not employ a model.

# USABILITY TESTING

- Automation has been used predominantly in two ways within usability testing:
  - automated capture of use data
  - automated analysis of these data according to some metrics or a model

# Automating Usability Testing Methods: Capture Support

- Many usability testing methods require the recording of the actions a user makes while exercising an interface.

- This can be done by an evaluator taking notes while the participant uses the system, either live or by repeatedly viewing a videotape of the session

- both are time-consuming activities

- Automated capture techniques can log user activity automatically.
  - information that is easy to record but difficult to interpret (e.g., keystrokes)
  - information that is meaningful but difficult to automatically label, such as task completion.

# Automating Usability Testing Methods: Capture Support

- automated capture of usage data is supported by two method types:
  - performance measurement and
  - remote testing.

- Both require the instrumentation of a user interface, incorporation into a user interface management system (UIMS), or capture at the system level
- Tools: KALDI, UsAGE, IDCAT

# Automating Usability Testing Methods: Analysis Support

- Log file analysis methods automate analysis of data captured during formal or informal interface use

- four general approaches for analyzing WIMP and Web log files:
  - metric based
  - pattern-matching
  - task-based
  - Inferential

# Automating Usability Testing Methods: Analysis Support—Metric-Based Analysis of Log Files.

- Generate quantitative performance measurements

- DRUM enables the evaluator to review a videotape of a usability test and manually log starting and ending points for tasks.

- DRUM processes this log and derives several measurements, including: task completion time, user efficiency (i.e., effectiveness divided by task completion time), and productive period (i.e., portion of time the user did not have problems).

- DRUM also synchronizes the occurrence of events in the log with videotaped footage, thus speeding up video analysis.

# Automating Usability Testing Methods: Analysis Support—Metric-Based Analysis of Log Files.

- MIKE UIMS enables an evaluator to assess the usability of a UI specified as a model that can be rapidly changed and compiled into a functional UI.

- MIKE captures usage data and generates a number of general, physical, logical, and visual metrics, including performance time, command frequency, the number of physical operations required to complete a task, and required changes in the user's focus of attention on the screen

- AMME employs Petri nets to reconstruct and analyze the user's problem-solving process.

- It requires a specially formatted log file and a manually created system description file (i.e., a list of interface states and a state transition matrix) in order to generate the Petri net.

- It then computes measures of behavioral complexity (i.e., steps taken to perform tasks), routinization (i.e., repetitive use of task sequences), and ratios of thinking versus waiting time

# Automating Usability Testing Methods: Analysis Support—Pattern-Matching Analysis of Log File

- Pattern-matching approaches, such as MRP (maximum repeating pattern) analyze user behavior captured in logs

- MRP detects and reports repeated user actions(e.g., consecutive invocations of the same command and errors) that may indicate usability problems.

- Studies with MRP showed the technique to be useful for detecting problems with expert users, but additional data prefiltering was required for detecting problems with novice users

# Automating Usability Testing Methods: Analysis Support—Task-Based Analysis of Log Files

- Task-based approaches analyze discrepancies between the designer's anticipation of the user's task model and what a user actually does while using the system
- IBOT - evaluators can use the system to compare user and designer behavior on these tasks and to recognize patterns of inefficient or incorrect behaviors during task completion
- QUIP (quantitative user interface profiling) tool and KALDI provide more advanced approaches to task-based, log file analysis for Java-based UIs.
- QUIP aggregates traces of multiple user interactions and compares the task flows of these users to the designer's task flow.
- QUIP encodes quantitative time- and trace-based information into directed graphs

# Automating Usability Testing Methods: Analysis Support—Task-Based Analysis of Log Files

- USINE employs the ConcurTaskTrees notation to express temporal relationships among UI tasks

- USINE looks for precondition errors (i.e., task sequences that violate temporal relationships) and also reports quantitative metrics (task completion time) and information about task patterns, missing tasks, and user preferences reflected in the usage data.

- RemUSINE is an extension that analyzes multiple log files (typically captured remotely) to enable comparison across users.

# Automating Usability Testing Methods: Analysis Support—Inferential Analysis of Log Files

- includes both statistical and visualization techniques

- Statistical approaches include traffic-based analysis (e.g., pages per visitor or visitors per page) and time based analysis(e.g., clickstreams and page-view durations)

- The evaluator must interpret reported measures in order to identify usability problems.

- Statistical analysis is largely inconclusive for Web server logs, since they provide only a partial trace of user behavior and timing estimates may be skewed by network latencies

# USABILITY INSPECTION METHODS

- an evaluation methodology whereby an evaluator examines the usability aspects of a UI design with respect to its conformance to a set of guidelines.

- rely solely on the evaluator’s judgment.

- A large number of detailed usability guidelines have been developed for WIMP interfaces and Web interfaces

- Common inspection techniques are:
  - heuristic evaluation
  - cognitive walkthroughs

- automation has been predominately used within the inspection class to objectively check guideline conformance

- Software tools assist evaluators with guideline review by automatically detecting and reporting usability violations and in some cases making suggestions for fixing them

# Automating Inspection Methods: Capture Support

- During a cognitive walkthrough, an evaluator attempts to simulate a user’s problem-solving process while examining UI tasks.
- At each step of a task, the evaluator assesses whether a user would succeed or fail to complete the step
- There was an early attempt to “automate” cognitive walkthroughs by prompting evaluators with walkthrough questions and enabling evaluators to record their analyses in HyperCard
- Evaluators found this approach too cumbersome and time consuming to employ

# Automating Inspection Methods: Analysis Support

- Several quantitative measures have been proposed for evaluating interfaces.

- size measures (overall density, local density, number of groups, size of groups, number of items, and layout complexity)

- five visual techniques: physical composition, association and dissociation, ordering, and photographic techniques (Vanderdonkt), which identified more visual design properties than traditional balance, symmetry, and alignment measures

- functional feedback, interactive directness, application flexibility, and dialog flexibility (Rauterberg)

# Automating Inspection Methods: Analysis Support

- AIDE (semi-automated interface designer and evaluator) - tool that helps designers assess and compare different design options using quantitative task-sensitive and task-independent metrics, including efficiency (i.e., distance of cursor movement), vertical and horizontal alignment of elements, horizontal and vertical balance, and designer-specified constraints (e.g., position of elements)

- AIDE also employs an optimization algorithm to automatically generate initial UI layouts

- Sherlock is another automated analysis tool for Windows interfaces.

- focuses on task-independent consistency checking (e.g., same widget placement and labels) within the UI or across multiple Uis

- Sherlock evaluates visual properties of dialog boxes, terminology (e.g., identify confusing terms and check spelling), as well as button sizes and labels

# Automating Inspection Methods: Analysis Support—Web UIs

- The Rating Game - an automated analysis tool using a set of easily measurable features: an information feature (word to link ratio), a graphics feature (number of graphics on a page), a gadgets feature (number of applets, controls, and scripts on a page)

- Design Advisor enables visual analysis of Web pages. The tool uses empirical results from eye-tracking studies designed to assess the attentional effects of various elements, such as animation, images, and highlighting, in multimedia presentations

# Limitations

- the tools cannot assess UI aspects that cannot be operationalized, such as whether the labels used on elements will be understood by users

- The tools compute and report a number of statistics about a page (e.g., number of links, graphics, and words)

- The effectiveness of these structural analyses is questionable, since the thresholds have not been empirically validated

# Automating Inspection Methods: Critique Support

- Critique systems give designers clear directions for conforming to violated guidelines and consequently improving usability

- Following guidelines is difficult, especially when there are a large number of guidelines to consider.

- Automated critique approaches, especially ones that modify a UI provide the highest level of support for adhering to guidelines.

# Automating Inspection Methods: Critique Support - WIMP UIs

- The KRI/AG tool (knowledge-based review of user interface) is an automated critique system that checks the guideline conformance of X-Window UI designs created using the TeleUSE UIMS

- contains a knowledge base of guidelines and style guides, including the Smith and Mosier guidelines

- IDA (user interface design assistance) also embeds rule-base

- SYNOP [Balbo 1995] is a similar automated critique system that performs a rule-based critique of a control system application. SYNOP also modifies the UI model based on its evaluation d (i.e., expert system) guideline checks within a UIMS

- CHIMES (computer-human interaction models) assesses the degree to which NASA's space-related critical and high-risk interfaces meet human factors standards.

- Ergoval - organizes guidelines into an object-based framework (i.e., guidelines that are relevant to each graphical object) in order to bridge the gap between the developer's view of an interface and how guidelines are traditionally presented (i.e., checklists).

# Automating Inspection Methods: Critique Support—Web UIs

- LIFT Online and LIFT Onsite perform usability checks as well as checking for use of standard and portable link, text, and background colors, the existence of stretched images, and other guideline violations.

- LIFT Online suggests improvements, and LIFT Onsite guides users through making suggested improvements

- Cooper - HTML analysis tool that checks Web pages for their accessibility to people with disabilities

- WebEval provides a framework for applying established WIMP guidelines to relevant HTML components.

# AUTOMATING INQUIRY METHODS

- Similar to usability testing approaches, inquiry methods require feedback from users and are often employed during usability testing
- the focus is not on studying specific tasks or measuring performance
- the goal of these methods is to gather subjective impressions (i.e., preferences or opinions) about various aspects of a UI
- Evaluators use  inquiry methods, such as surveys questionnaires, and interviews, to gather supplementary data after a system is released; this is useful for improving the interface for future releases
- Inquiry methods vary based on whether the evaluator interacts with a user or a group of users or whether users report their experiences using questionnaires or usage logs, possibly in conjunction with screen snapshots

# Automating Inquiry Methods: Capture Support

- developed to assist users with filling in questionnaires.
- Software tools enable the evaluator to collect subjective usability data and possibly make improvements throughout the life of an Interface.
- Questionnaires can be embedded into a WIMP UI to facilitate the response capture process.
- Typically dialog boxes prompt users for subjective input and process responses (e.g., saves data to a file or emails data to the evaluator).
- UPM (the user partnering module)  uses event-driven triggers (e.g., errors or specific command invocations) to ask users specific questions about their interface usage.
- This approach allows the evaluator to capture user reactions while they are still fresh

# Automating Inquiry Methods: Capture Support

- Several validated questionnaires are available in Web format
- QUIS (questionnaire for user interaction satisfaction)
- NetRaker Index (a short usability questionnaire) for continuously gathering feedback from users about a Web site.
- NetRaker’s tools are highly effective for gathering direct user feedback, but potential **irritations** caused by the NetRaker Index’s pop-up survey window is possible
- Automatic Inquiry Methods **do not support** automated analysis or critique of interfaces.

# The Evolution of Usability Engineering in Organizations (Nielsen)

- **Usability does not matter.** The main focus is to wring every last bit of performance from the iron. This is the attitude leading to the world-famous error message, "beep.“

- Usability is important, but **good interfaces can surely be designed by the regular development staff** as part of their general system design. At this stage, no attempt is made at user testing or at acquiring staff with usability expertise.

- The desire to have the **interface blessed by the magic wand** of a usability engineer. Developers recognize that they may not know everything about usability, so they call in a usability specialist to look over their design and comment on it. The involvement of the usability specialist is often too late to do much good in the project, and the usability specialist often has to provide advice on the interface without the benefit of access to real users.

# The Evolution of Usability Engineering in Organizations (Nielsen)

- **GUI panic strikes** , causing a sudden desire to learn about user interface issues. Currently, many companies are in this stage as they are moving from character-based user interfaces to graphical user interfaces and realize the need to bring in usability specialists to advise on graphical user interfaces from the start. Some usability specialists resent this attitude and maintain that it is more important to provide an appropriate interface for the task than to blindly go with a graphical interface without prior task analysis

- **Discount usability engineering *sporadically* used.** Typically, some projects use a few discount usability methods (like user testing or heuristic evaluation), though the methods are often used too late in the development lifecycle to do maximum good. Projects that do use usability methods often differ from others in having managers who have experienced the benefit of usability methods on earlier projects. Thus, usability acts as a kind of virus, infecting progressively more projects as more people experience its benefits.

# The Evolution of Usability Engineering in Organizations (Nielsen)

- **Discount usability engineering *systematically* used.** At some point in time, most projects involve some simple usability methods, and some projects even use usability methods in the early stages of system development. Scenarios and cheap prototyping techniques seem to be very effective weapons for guerrilla HCI in this stage.

- **Usability group and/or usability lab founded.** Many companies decide to expand to a deluxe usability approach after having experienced the benefits of discount usability engineering. Currently, the building of usability laboratories is quite popular as is the formation of dedicated groups of usability specialists.

- **Usability permeates lifecycle.** The final stage is rarely reached since even companies with usability groups and usability labs normally do not have enough usability resources to employ all the methods one could wish for at all the stages of the development lifecycle. However, there are some, often important, projects that have usability plans defined as part of their early project planning and where usability methods are used throughout the development lifecycle.

# Resources

- Smith and Mosier Design guidelines
- Jeffrey Rubin, Handbook of Usability Testing. How to plan, design, and conduct effective test