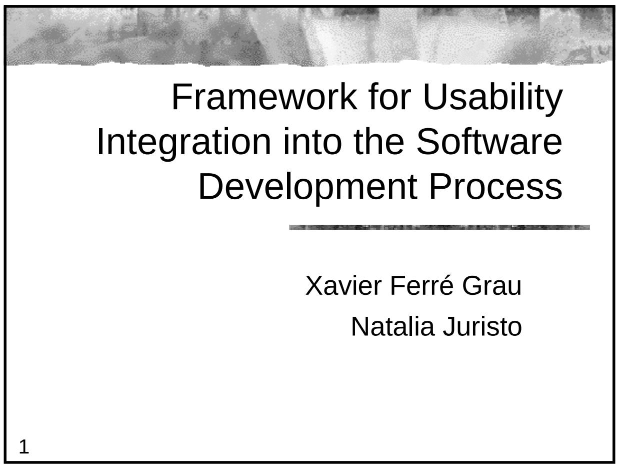
How to Integrate Usability into the Software Development Process

Tutorial H10

ICSE 2006 - Shanghai (China)
28 May, 2006

Natalia Juristo
Universidad Politecnica de Madrid (Spain)
natalia@fi.upm.es

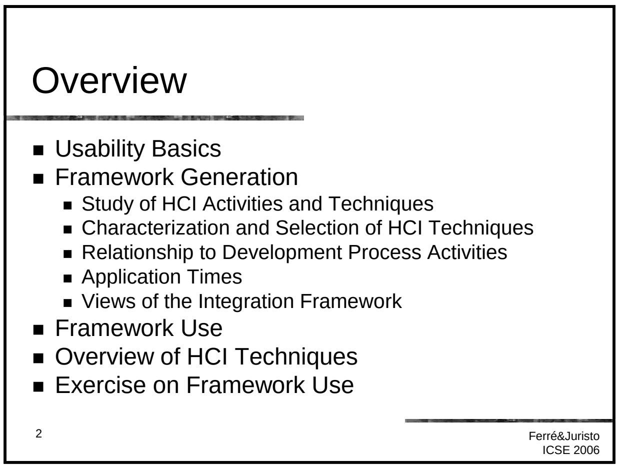
Xavier Ferre
Universidad Politecnica de Madrid (Spain)
xavier@fi.upm.es



Framework for Usability Integration into the Software Development Process

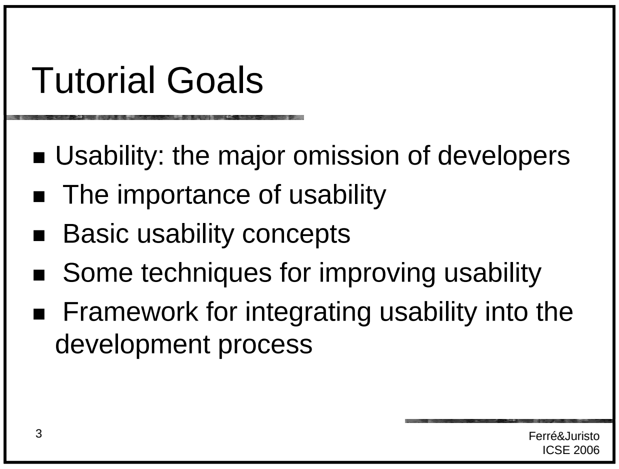
Xavier Ferré Grau
Natalia Juristo

1



Overview

- Usability Basics
- Framework Generation
  - Study of HCI Activities and Techniques
  - Characterization and Selection of HCI Techniques
  - Relationship to Development Process Activities
  - Application Times
  - Views of the Integration Framework
- Framework Use
- Overview of HCI Techniques
- Exercise on Framework Use



Tutorial Goals

- Usability: the major omission of developers
- The importance of usability
- Basic usability concepts
- Some techniques for improving usability
- Framework for integrating usability into the development process

Usability Basics

Natalia

4

Usability is a pending issue

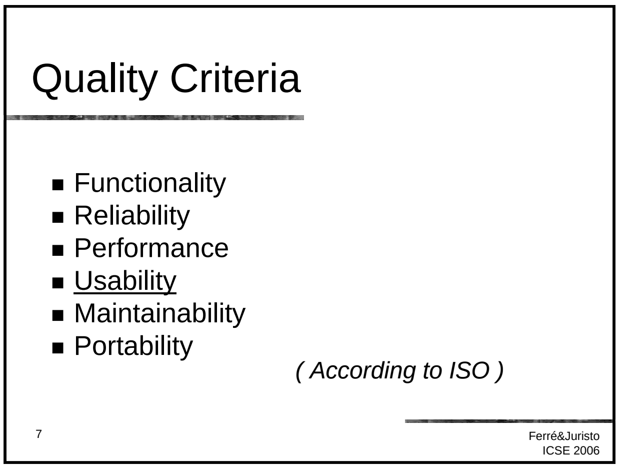
- There are so many software products with immature usability that we all acknowledge the low level of use of usability methods
- Usability is not properly addressed in most developments, on spite of its importance

5 Ferré&Juristo ICSE 2006

Quality includes usability

- Usability is a **basic** software system feature
- Several software quality attribute classifications agree on considering usability as a **quality attribute**

6 Ferré&Juristo ICSE 2006



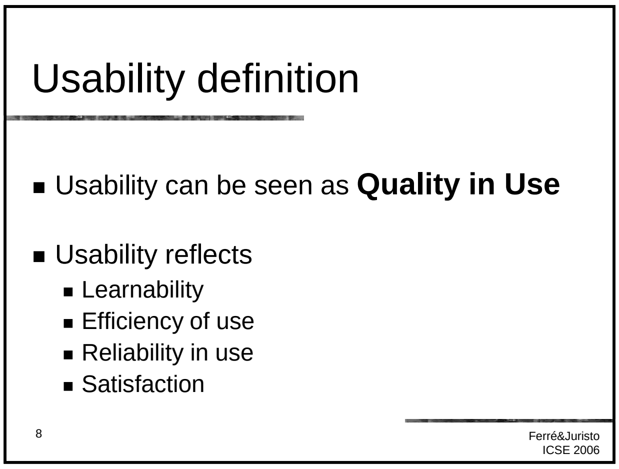
Quality Criteria

- Functionality
- Reliability
- Performance
- Usability
- Maintainability
- Portability

(According to ISO)

7

Ferré&Juristo
ICSE 2006

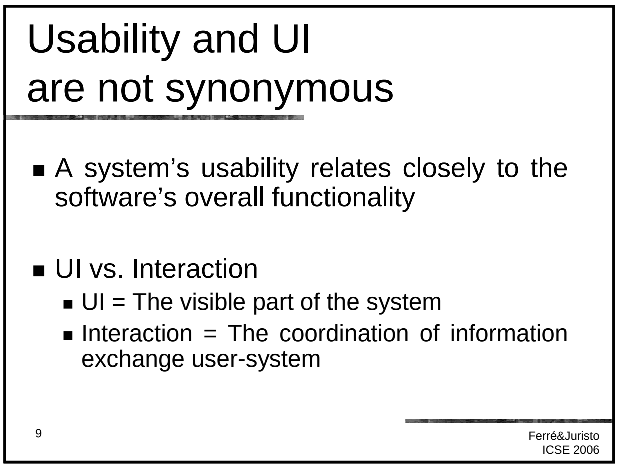


Usability definition

- Usability can be seen as **Quality in Use**
- Usability reflects
  - Learnability
  - Efficiency of use
  - Reliability in use
  - Satisfaction

8

Ferré&Juristo
ICSE 2006

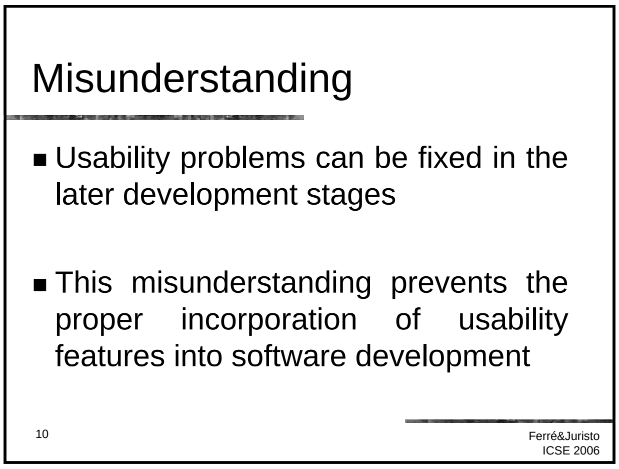


Usability and UI are not synonymous

- A system's usability relates closely to the software's overall functionality
- UI vs. Interaction
  - UI = The visible part of the system
  - Interaction = The coordination of information exchange user-system

9

Ferré&Juristo
ICSE 2006

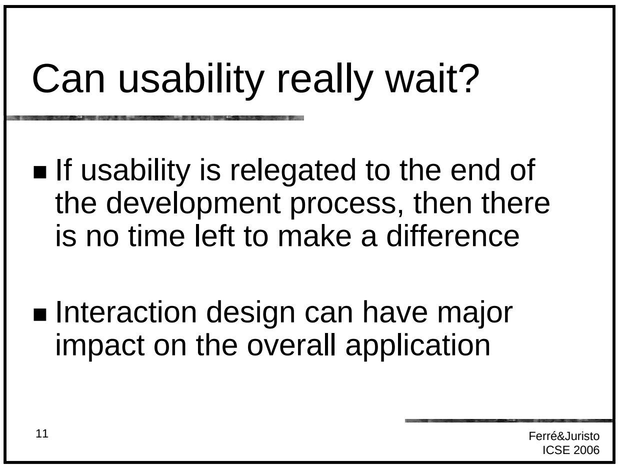


Misunderstanding

- Usability problems can be fixed in the later development stages
- This misunderstanding prevents the proper incorporation of usability features into software development

10

Ferré&Juristo
ICSE 2006

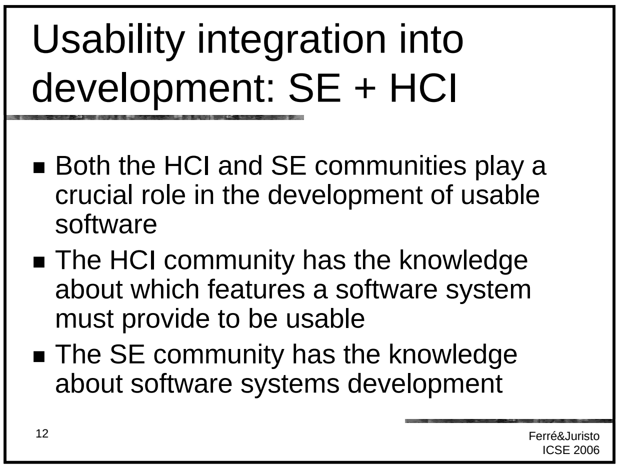


Can usability really wait?

- If usability is relegated to the end of the development process, then there is no time left to make a difference
- Interaction design can have major impact on the overall application

11

Ferré&Juristo
ICSE 2006

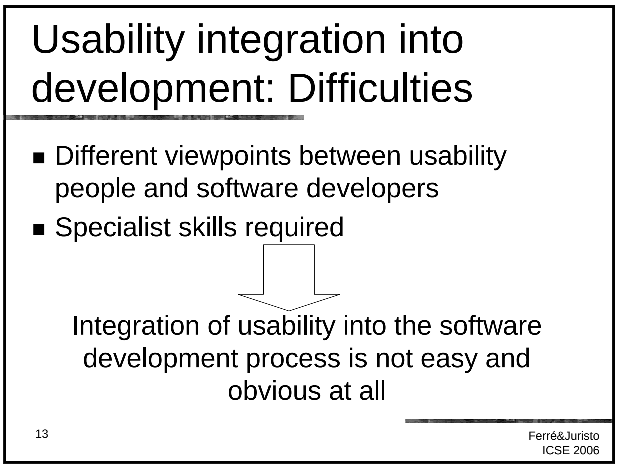


Usability integration into development: SE + HCI

- Both the HCI and SE communities play a crucial role in the development of usable software
- The HCI community has the knowledge about which features a software system must provide to be usable
- The SE community has the knowledge about software systems development

12

Ferré&Juristo
ICSE 2006



Usability integration into development: Difficulties

- Different viewpoints between usability people and software developers
- Specialist skills required

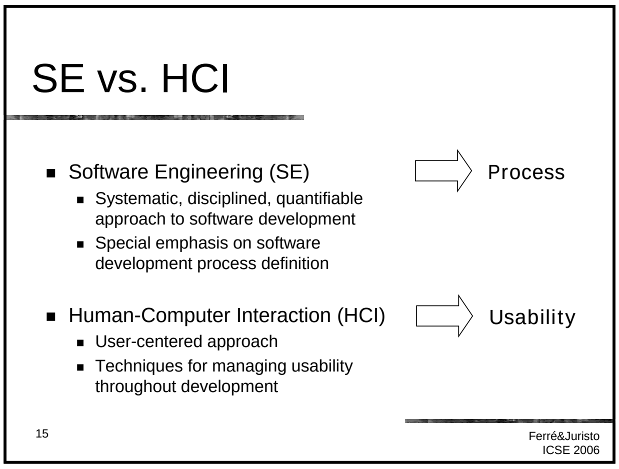
Integration of usability into the software development process is not easy and obvious at all

13

Ferré&Juristo
ICSE 2006

SE-HCI Integration Framework

14



SE vs. HCI

- Software Engineering (SE) ⇨ Process
  - Systematic, disciplined, quantifiable approach to software development
  - Special emphasis on software development process definition
- Human-Computer Interaction (HCI) ⇨ Usability
  - User-centered approach
  - Techniques for managing usability throughout development

15

Ferré&Juristo
ICSE 2006

Obstacles to HCI-SE Integration

- Strong differences between HCI and SE
  - **HCI processes sphere**
    - Example: Requirements
  - **Terminology**
    - Example: User Interface (UI) Design
  - **When to consider usability during development**
- HCI activities and techniques are not presented properly from a SE viewpoint

16

Ferré&Juristo
ICSE 2006

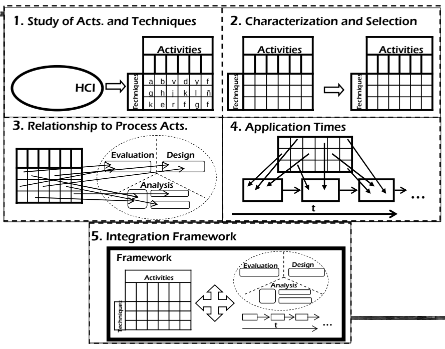
Development Process Conditions

- Characteristics of a user-centred process
  - Active user involvement
  - A proper understanding of user and task requirements
  - Multidisciplinary knowledge
  - Iterative development
- Only condition to be met
  - Iterative development

17

Ferré&Juristo
ICSE 2006

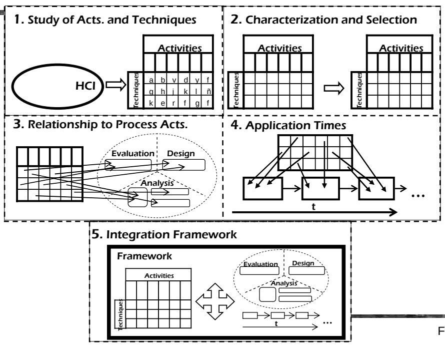
Steps to obtain the Framework



18

Ferré&Juristo
ICSE 2006

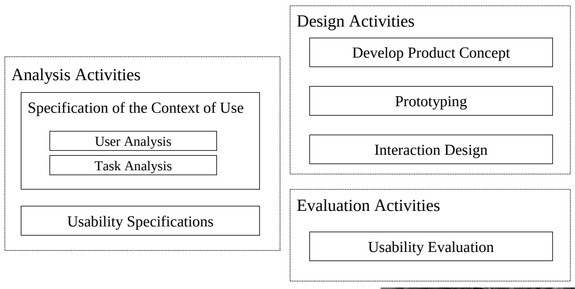
Study of Activities and Techniques in a User-Centered Process



Study of Usability Activities by Source

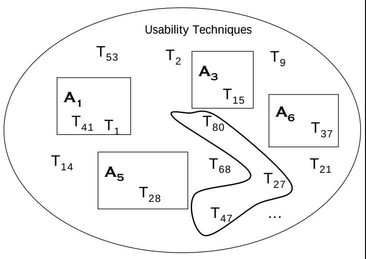
Activity	Nielsen93	ISO13407,99	Shneiderman98	Hix93	Preece94	Wixon97	Constantine99	Mayhew 99
SPECIFICATION OF THE CONTEXT OF USE	Know the user	Understand and specify the context of use	Perform research and needs analysis	Systems/ tasks / functional / user analysis	Task analysis / functional analysis	* Specify and categorise the users * Conduct a task analysis	Task modelling	* User profile * Contextual Task Analysis
USABILITY SPECIFICATIONS	Goal Settings	Specify the user and organisational requirements	Design concepts and key-screen prototype (create specific usability objectives based on user needs)	Requirements / Usability Specifications	Requirements specification	* Define quantitative usability goals * Set levels of desired usability for each goal	-	Usability goal setting
DEVELOP PRODUCT CONCEPT	-	-	Develop product concept	Conceptual design	Conceptual design / formal design	-	-	Conceptual model design
PROTOTYPING	Prototyping	Produce design solutions (make design solutions more concrete using simulations, models, mock-ups, etc.)	Design concepts and key-screen prototype	Rapid prototyping	Prototyping	-	-	Screen design standards prototyping
USABILITY EVALUATION	Interface Evaluation	Evaluate design against requirements	Do iterative design and refinement (conduct full-scale usability tests)	Usability evaluation	Evaluation	Test the product against usability goals	Usability inspection	Iterative detailed user interface design evaluation

Representative Activities in a User-Centered Process



Difficulties in the Study of HCI techniques from a SE viewpoint

- Diversity of HCI techniques
  - Differences between authors
  - Terminological diversity
  - Lack of assignment of techniques to activities

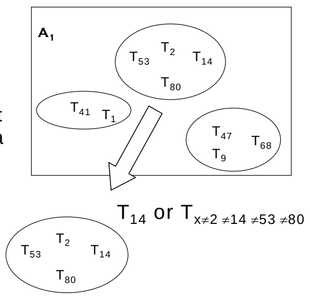


22

Ferré&Juristo
ICSE 2006

Identification and Classification Process

1. Compilation of the techniques mentioned in the considered sources
2. Grouping of techniques that refer to the same basic idea
3. Assignment to the HCI activity that they best fit
4. Choice of the most representative name for each basic technique



23

Ferré&Juristo
ICSE 2006

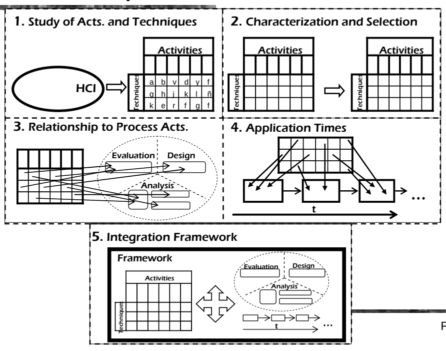
Technique Classification: Techniques related to the Specification of the Context of Use

Activity	Technique	Hix, 93	Nielsen, 93	Preece, 94	Shnei-derman, 98	Constan-tine, 99	Mayhew, 99
	Competitive Analysis		Competitive Analysis				
	Financial Impact Analysis		Financial Impact Analysis				
	Contextual Inquiry	Contextual Inquiry		Contextual Inquiry			Contextual Interviews
	Affinity Diagrams						Affinity Diagrams
	Ethnographic Observation			Ethnography	Ethnographic Observation		
	JEM[1]					JEM	
User Analysis	**User Profiles**	User Profiles	Individual user characteristics		User profiles	Structured Role Model	User profile questionn.
	User Role Map					User role map	
	Operational Modeling					Operational Modeling	Platform constraints and capabilities

24

Ferré&Juristo
ICSE 2006

Characterization and Selection of HCI Techniques



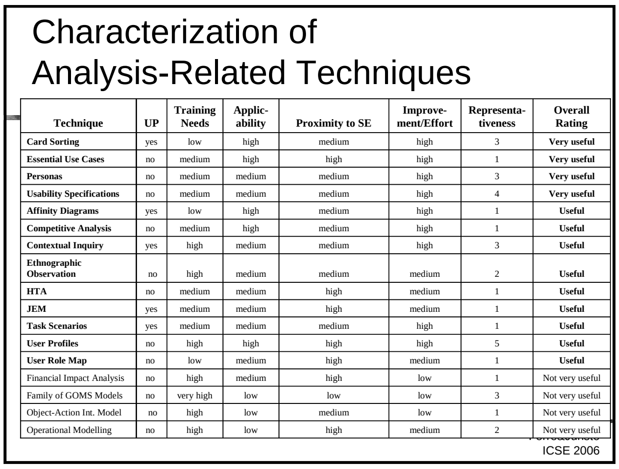
Technique Characterization

- Source: 94 compiled techniques
- Objective: To provide information about how useful each technique is
- Criteria
  - User Participation
  - Training Needs
  - General Applicability
  - Proximity to SE
  - Usability Improvement/Effort Ratio
  - Representativeness
  - Total Rating

Summary Criterion: Total Rating

- All the considered criteria are summarized in a criterion called “Overall Rating”

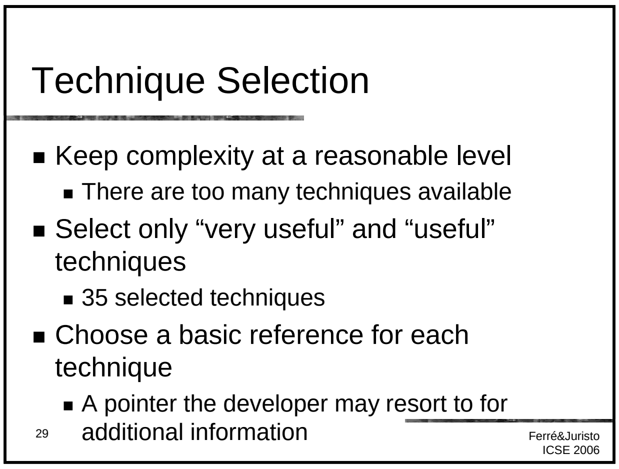
■ Very useful
■ Useful
■ Not very useful



Characterization of Analysis-Related Techniques

Technique	UP	Training Needs	Applic-ability	Proximity to SE	Improve-ment/Effort	Representa-tiveness	Overall Rating
Card Sorting	yes	low	high	medium	high	3	**Very useful**
Essential Use Cases	no	medium	high	high	high	1	**Very useful**
Personas	no	medium	medium	medium	high	3	**Very useful**
Usability Specifications	no	medium	medium	medium	high	4	**Very useful**
Affinity Diagrams	yes	low	high	medium	high	1	**Useful**
Competitive Analysis	no	medium	high	medium	high	1	**Useful**
Contextual Inquiry	yes	high	medium	medium	high	3	**Useful**
Ethnographic Observation	no	high	medium	medium	medium	2	**Useful**
HTA	no	medium	medium	high	medium	1	**Useful**
JEM	yes	medium	medium	high	medium	1	**Useful**
Task Scenarios	yes	medium	medium	medium	high	1	**Useful**
User Profiles	no	high	high	high	high	5	**Useful**
User Role Map	no	low	medium	high	medium	1	**Useful**
Financial Impact Analysis	no	high	medium	high	low	1	Not very useful
Family of GOMS Models	no	very high	low	low	low	3	Not very useful
Object-Action Int. Model	no	high	low	medium	low	1	Not very useful
Operational Modelling	no	high	low	high	medium	2	Not very useful

ICSE 2006

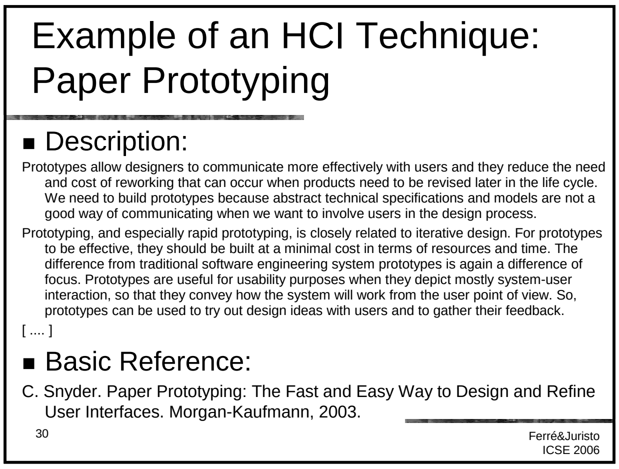


Technique Selection

- Keep complexity at a reasonable level
  - There are too many techniques available
- Select only “very useful” and “useful” techniques
  - 35 selected techniques
- Choose a basic reference for each technique
  - A pointer the developer may resort to for additional information

29

Ferré&Juristo
ICSE 2006



Example of an HCI Technique: Paper Prototyping

- Description:

Prototypes allow designers to communicate more effectively with users and they reduce the need and cost of reworking that can occur when products need to be revised later in the life cycle. We need to build prototypes because abstract technical specifications and models are not a good way of communicating when we want to involve users in the design process.

Prototyping, and especially rapid prototyping, is closely related to iterative design. For prototypes to be effective, they should be built at a minimal cost in terms of resources and time. The difference from traditional software engineering system prototypes is again a difference of focus. Prototypes are useful for usability purposes when they depict mostly system-user interaction, so that they convey how the system will work from the user point of view. So, prototypes can be used to try out design ideas with users and to gather their feedback.

[....]

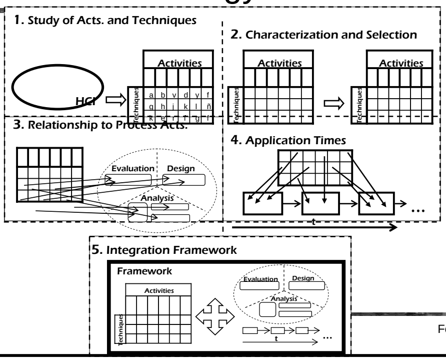
- Basic Reference:

C. Snyder. Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces. Morgan-Kaufmann, 2003.

30

Ferré&Juristo
ICSE 2006

Relationship Between HCI and SE Process Terminology



31

Ferré&Juristo
ICSE 2006

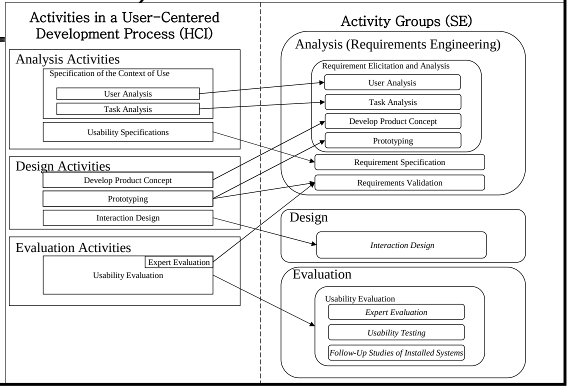
Relationship Between HCI and SE Process Terminology

- Employ SE process terminology
  - So developers may understand where they should apply usability techniques
- Take the following steps
  1. Map HCI activities to SE activities
  2. Assign HCI techniques to the SE activity group in which their application is more useful

32

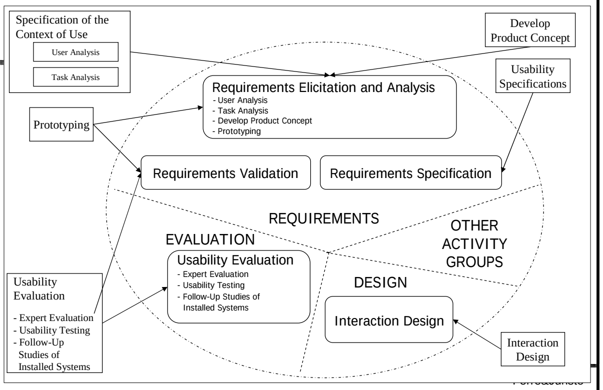
Ferré&Juristo
ICSE 2006

Analysis, Design & Evaluation (HCI vs. SE)

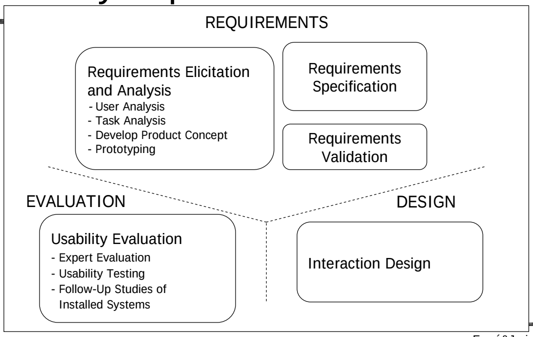


33

Mapping of HCI Activities to SE Activities



Development Activities with a Usability Impact



Assignment of HCI Techniques to SE Activities

- Technique assignment to activity types
  - Activity type whose objectives the technique fulfills
  - First candidate is the activity type to which its HCI activity maps
- Requirements Elicitation and Analysis have been grouped together
  - The HCI makes no clear distinction between Elicitation and Analysis
  - 15 HCI techniques assigned to the combined activity group

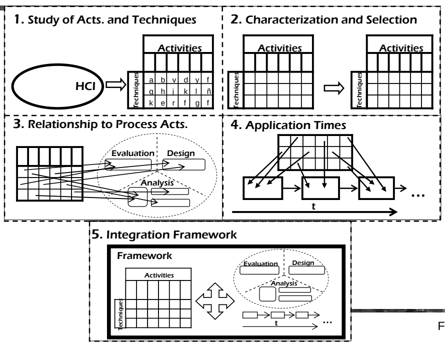
Assignment of HCI Techniques to SE Activities

Activity			Technique
Requirements	Requirements Elicitation and Analysis		Card Sorting
			Affinity Diagramming
			Competitive Analysis
			Contextual Inquiry
			Ethnographic Observation
			JEM
		User Analysis	Personas
			User Profiles
			User Role Map
		Task Analysis	Essential Use Cases
			HTA
			Task Scenarios
		Develop Product Concept	Scenarios and Storyboards
			Visual Brainstorming
		Prototyping	Paper Prototypes
	Requirements Specification		Usability Specifications
	Requirements Validation		Inspections
			Cognitive Walkthrough
			Collaborative Inspections

37

Ferré&Juristo
ICSE 2006

Study of Application Times



38

Ferré&Juristo
ICSE 2006

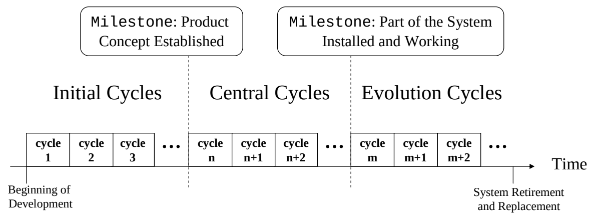
Usability-Significant Stages in an Iterative Process

- Not all cycles are the same
- Relevant milestones regarding HCI technique application
  - Product concept established
  - Part of the system installed and working

39

Ferré&Juristo
ICSE 2006

Division of Iterative Stages into Cycles



40

Ferré&Juristo
ICSE 2006

Assignment of HCI Techniques to Development Stages

- Aim
  - To identify the best application time for each technique
- Technique classification according to their fitness values
  - Especially well-matched
  - Neutral
  - Not usual

41

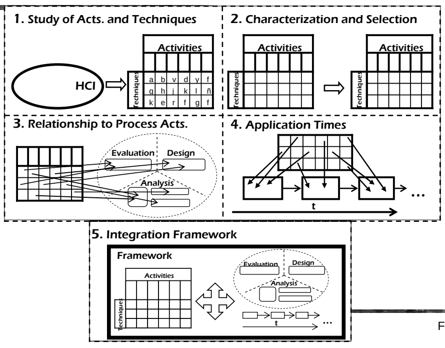
Ferré&Juristo
ICSE 2006

Suitability of Selected HCI Techniques for each Development Stage

Activities		Techniques	Initial Cycles	Central Cycles	Evolution C.
Design		Impact Analysis	■	▨	▨
		Organisation of Help by Use Cases	▨	▨	▨
	Interaction Design	Interface Content Model	▨	□	▨
		Menu Trees	▨	▨	▨
		Navigation Map	▨	▨	▨
		Product Style Guide	■	□	▨
		Transition Diagrams	▨	▨	▨
Evaluation	Expert Reviews	Cognitive Walkthrough	▨	▨	▨
		Collaborative Usability Inspections	□	▨	▨
		Heuristic Evaluation	▨	▨	▨
		Inspections	▨	▨	▨
		Pluralistic Walkthrough	□	▨	▨
	Usability Testing	Laboratory Usability Testing	■	▨	▨
		Measured Performance	■	▨	▨
		Post-Test Feedback	■	▨	▨
		Thinking Aloud	■	▨	▨
	Follow-Up Studies of Inst. Systems	Questionnaires, Interviews and Surveys	■	▨	□
		Logging Actual Use	■	▨	□
		User Feedback	■	■	□

ICSE 2006

Framework



43

Ferré&Juristo
ICSE 2006

Framework views

- Views
  - By HCI techniques
  - By development activities
  - By application times

44

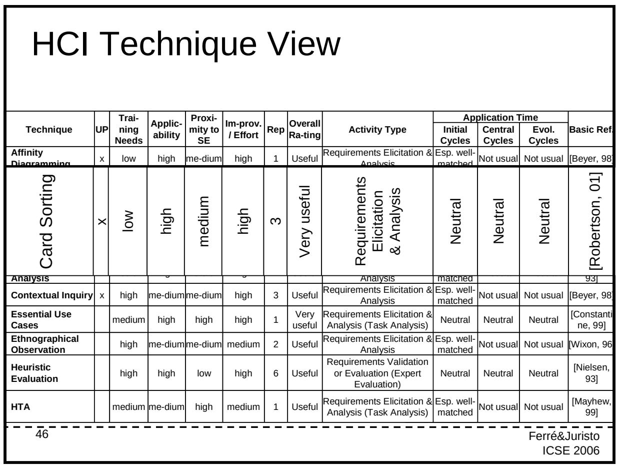
Ferré&Juristo
ICSE 2006

Integration Framework Overview

- All the knowledge generated for each technique is included in each view
  - Technique characterization
  - Activity type where it applies
  - Suitability for each moment in development time
  - Basic reference
- Each view organizes techniques according to their particular criteria
  - “very useful” vs. “useful” techniques

45

Ferré&Juristo
ICSE 2006

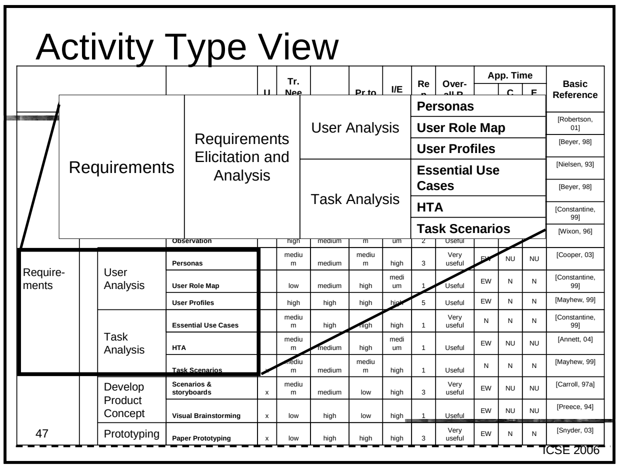


HCI Technique View

Technique	UP	Trai-ning Needs	Applic-ability	Proxi-mity to SE	Im-prov. / Effort	Rep	Overall Ra-ting	Activity Type	Application Time: Initial Cycles	Central Cycles	Evol. Cycles	Basic Ref.
Affinity Diagramming	x	low	high	me-dium	high	1	Useful	Requirements Elicitation & Analysis	Esp. well-matched	Not usual	Not usual	[Beyer, 98]
Card Sorting	x	low	high	medium	high	3	Very useful	Requirements Elicitation & Analysis	Neutral	Neutral	Neutral	[Robertson, 01]
Analysis								Analysis	matched			93]
Contextual Inquiry	x	high	me-dium	me-dium	high	3	Useful	Requirements Elicitation & Analysis	Esp. well-matched	Not usual	Not usual	[Beyer, 98]
Essential Use Cases		medium	high	high	high	1	Very useful	Requirements Elicitation & Analysis (Task Analysis)	Neutral	Neutral	Neutral	[Constantine, 99]
Ethnographical Observation		high	me-dium	me-dium	medium	2	Useful	Requirements Elicitation & Analysis	Esp. well-matched	Not usual	Not usual	[Wixon, 96
Heuristic Evaluation		high	high	low	high	6	Useful	Requirements Validation or Evaluation (Expert Evaluation)	Neutral	Neutral	Neutral	[Nielsen, 93]
HTA		medium	me-dium	high	medium	1	Useful	Requirements Elicitation & Analysis (Task Analysis)	Esp. well-matched	Not usual	Not usual	[Mayhew, 99]

46

Ferré&Juristo
ICSE 2006



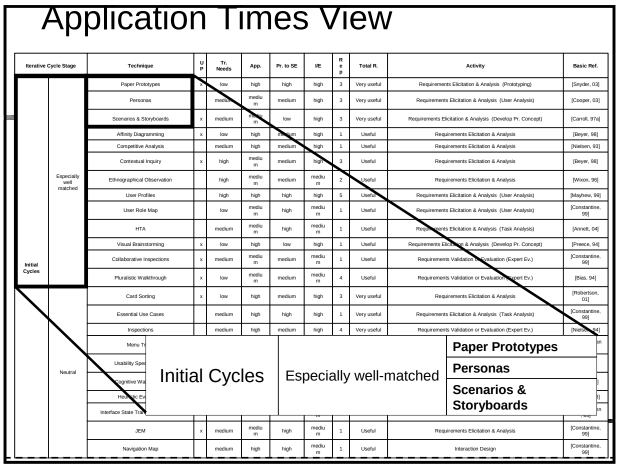
Activity Type View

Requirements	Requirements Elicitation and Analysis	User Analysis	Personas
			User Role Map
			User Profiles
		Task Analysis	Essential Use Cases
			HTA
			Task Scenarios

		Technique	U	Tr. Nee		Pr to	I/E	Re	Over-	App. Time	C	E	Basic Reference
													[Robertson, 01]
													[Beyer, 98]
													[Nielsen, 93]
													[Beyer, 98]
													[Constantine, 99]
		Observation		high	medium	m	um	2	Useful				[Wixon, 96]
Require-ments	User Analysis	Personas		medium	medium	medium	high	3	Very useful	EW	NU	NU	[Cooper, 03]
		User Role Map		low	medium	high	medium	1	Useful	EW	N	N	[Constantine, 99]
		User Profiles		high	high	high	high	5	Useful	EW	N	N	[Mayhew, 99]
	Task Analysis	Essential Use Cases		medium	high	high	high	1	Very useful	N	N	N	[Constantine, 99]
		HTA		medium	medium	high	medium	1	Useful	EW	NU	NU	[Annett, 04]
		Task Scenarios	x	medium	medium	medium	high	1	Useful	N	N	N	[Mayhew, 99]
	Develop Product Concept	Scenarios & storyboards	x	medium	medium	low	high	3	Very useful	EW	NU	NU	[Carroll, 97a]
		Visual Brainstorming	x	low	high	low	high	1	Useful	EW	NU	NU	[Preece, 94]
	Prototyping	Paper Prototyping	x	low	high	high	high	3	Very useful	EW	N	N	[Snyder, 03]

47

ICSE 2006



Application Times View

Iterative Cycle Stage		Technique	UP	Tr. Needs	App.	Pr. to SE	I/E	Rep	Total R.	Activity	Basic Ref.
Initial Cycles	Especially well matched	Paper Prototypes	x	low	high	high	high	3	Very useful	Requirements Elicitation & Analysis (Prototyping)	[Snyder, 03]
		Personas		medium	medium	medium	high	3	Very useful	Requirements Elicitation & Analysis (User Analysis)	[Cooper, 03]
		Scenarios & Storyboards	x	medium	medium	low	high	3	Very useful	Requirements Elicitation & Analysis (Develop Pr. Concept)	[Carroll, 97a]
		Affinity Diagramming	x	low	high	medium	high	1	Useful	Requirements Elicitation & Analysis	[Beyer, 98]
		Competitive Analysis		medium	high	medium	high	1	Useful	Requirements Elicitation & Analysis	[Nielsen, 93]
		Contextual Inquiry	x	high	medium	medium	high	3	Useful	Requirements Elicitation & Analysis	[Beyer, 98]
		Ethnographical Observation		high	medium	medium	medium	2	Useful	Requirements Elicitation & Analysis	[Wixon, 96]
		User Profiles		high	high	high	high	5	Useful	Requirements Elicitation & Analysis (User Analysis)	[Mayhew, 99]
		User Role Map		low	medium	high	medium	1	Useful	Requirements Elicitation & Analysis (User Analysis)	[Constantine, 99]
		HTA		medium	medium	high	medium	1	Useful	Requirements Elicitation & Analysis (Task Analysis)	[Annett, 04]
		Visual Brainstorming	x	low	high	low	high	1	Useful	Requirements Elicitation & Analysis (Develop Pr. Concept)	[Preece, 94]
		Collaborative Inspections	x	medium	medium	medium	medium	1	Useful	Requirements Validation or Evaluation (Expert Ev.)	[Constantine, 99]
		Pluralistic Walkthrough	x	low	medium	medium	medium	4	Useful	Requirements Validation or Evaluation (Expert Ev.)	[Bias, 94]
	Neutral	Card Sorting	x	low	high	medium	high	3	Very useful	Requirements Elicitation & Analysis	[Robertson, 01]
		Essential Use Cases		medium	high	high	high	1	Very useful	Requirements Elicitation & Analysis (Task Analysis)	[Constantine, 99]
		Inspections		medium	high	medium	high	4	Very useful	Requirements Validation or Evaluation (Expert Ev.)	[Nielsen, 94]
		Menu Tr									an
		Usability Spec									
		Cognitive Wa									]
		Heuristic Ev									3]
		Interface State Tra					m				hn , 85]
		JEM	x	medium	medium	high	medium	1	Useful	Requirements Elicitation & Analysis	[Constantine, 99]
		Navigation Map		medium	high	high	medium	1	Useful	Interaction Design	[Constantine, 99]

Initial Cycles	Especially well-matched	Paper Prototypes
		Personas
		Scenarios & Storyboards

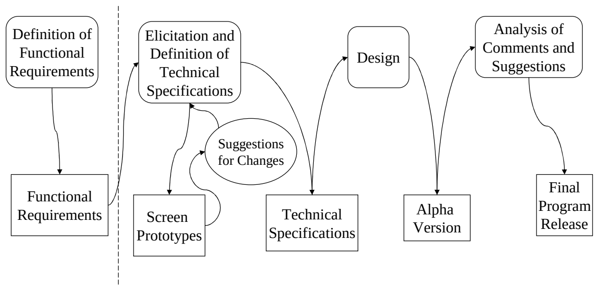
Usage of the Framework

- Views are complementary ways of accessing the knowledge
- They can be flexibly combined in any situation
  - Preliminary selection of what techniques to include
  - Planning of next cycle
  - ...

49

Ferré&Juristo
ICSE 2006

Exercise: Development Process



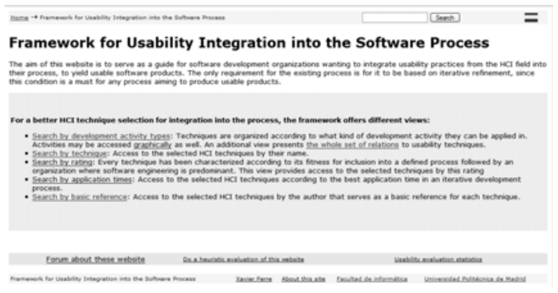
Definition of Functional Requirements
Functional Requirements
Elicitation and Definition of Technical Specifications
Suggestions for Changes
Screen Prototypes
Technical Specifications
Design
Alpha Version
Analysis of Comments and Suggestions
Final Program Release

50

Ferré&Juristo
ICSE 2006

Web Tool

http://is.ls.fi.upm.es/udis/miembros/xavier/usabilityframework/



Home → Framework for Usability Integration into the Software Process

Framework for Usability Integration into the Software Process

The aim of this website is to serve as a guide for software development organizations wanting to integrate usability practices from the HCI field into their process, to yield usable software products. The only requirement for the existing process is for it to be based on iterative refinement, since this condition is a must for any process aiming to produce usable products.

For a better HCI technique selection for integration into the process, the framework offers different views:

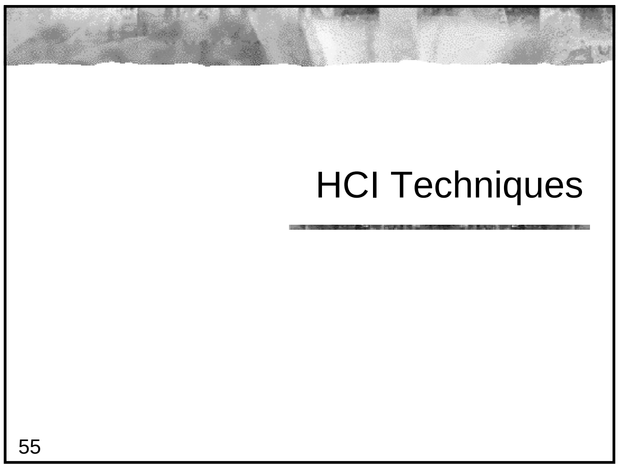
- Search by development activity types: Techniques are organized according to what kind of development activity they can be applied in. Activities may be accessed graphically as well. An additional view presents the whole set of relations to usability techniques.
- Search by technique: Access to the selected HCI techniques by their name.
- Search by rating: Every technique has been characterized according to its fitness for inclusion into a defined process followed by an organization where software engineering is predominant. This view provides access to the selected techniques by this rating
- Search by application times: Access to the selected HCI techniques according to the best application time in an iterative development process.
- Search by basic reference: Access to the selected HCI techniques by the author that serves as a basic reference for each technique.

Forum about these website   Do a heuristic evaluation of this website   Usability evaluation statistics

Framework for Usability Integration into the Software Process   Xavier Ferre   About this site   Facultad de informática   Universidad Politécnica de Madrid

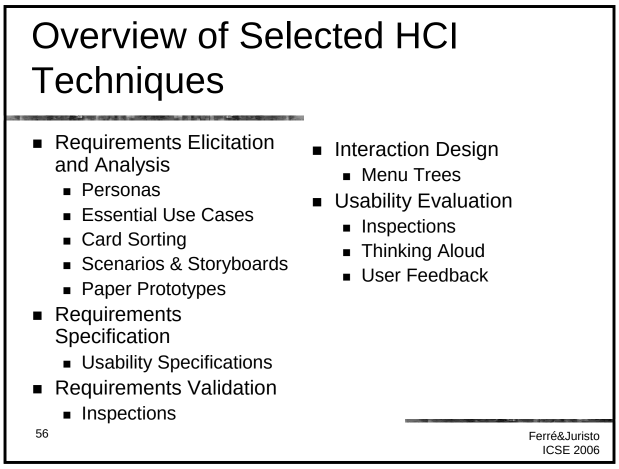
54

Ferré&Juristo
ICSE 2006



HCI Techniques

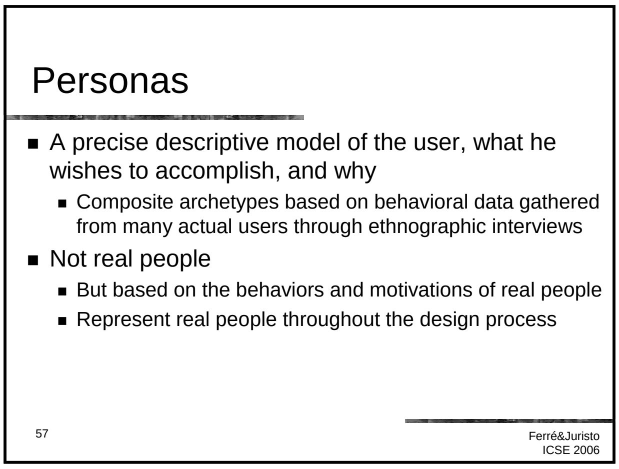
55



Overview of Selected HCI Techniques

- Requirements Elicitation and Analysis
  - Personas
  - Essential Use Cases
  - Card Sorting
  - Scenarios & Storyboards
  - Paper Prototypes
- Requirements Specification
  - Usability Specifications
- Requirements Validation
  - Inspections
- Interaction Design
  - Menu Trees
- Usability Evaluation
  - Inspections
  - Thinking Aloud
  - User Feedback

56 Ferré&Juristo ICSE 2006



Personas

- A precise descriptive model of the user, what he wishes to accomplish, and why
  - Composite archetypes based on behavioral data gathered from many actual users through ethnographic interviews
- Not real people
  - But based on the behaviors and motivations of real people
  - Represent real people throughout the design process

57 Ferré&Juristo ICSE 2006

Why Using Personas

- The best way to successfully accommodate a variety of users is to design for specific types of individuals with specific needs
  - Choose the right individuals to design for, ones whose needs represent the needs of a larger set of key constituents
  - Prioritize the design elements to address the needs of the most important users without significantly inconveniencing secondary users
- Personas provide a tool for
  - Understanding user needs
  - Differentiating between different types of users
  - Prioritizing which users are the most important target in the design of function and behavior

58

Ferré&Juristo
ICSE 2006

Personas: Example

- Costas
  - He is 35 years old, he is married with 2 children. He's got a diploma on Mathematics and a Masters degree on Economics. He uses a laptop for his work, and the main applications he uses are: MS Excel, Word, etc. He usually looks after information on the stock market in the web (he has some stocks of his company). He owns a cutting-edge mobile phone, that holds pictures of his children he may show to his friends. He manages a team of 3-10 people and he is accounted responsible for the productivity of his team. His main concerns are managing the tasks the people on his team are assigned to.

59

Ferré&Juristo
ICSE 2006

Use Cases in SE

- A use case is a case of use, or one kind of use to which a system can be put. It is:
  - Supplied functionality
  - An external, “black-box” view
  - A narrative description
  - Interaction between a user and a system
  - A use of the system that is completely meaningful to the user
- Extensively used in object-oriented analysis and design, but with a wrong focus:
  - Implicit design assumptions on the form of the UI
  - Too close to implementation
  - Away from the user sphere

60

Ferré&Juristo
ICSE 2006

Essential Use Cases

- Two levels of abstraction for describing use cases:
  - Essential use cases
  - Detailed or concrete use cases
- An Essential use case is based on the purpose or intentions of a user rather than on the concrete steps by which such intentions are carried out
- It has three components:
  - A statement of the overall user purpose or intention expressed within the use case
  - A two-part narrative that structures the interaction between user intentions and system responsibilities

Example of Essential Use Case

ESSENTIAL USE CASE		**DETAILED USE CASE**	
gettingCash		gettingCash	
USER INTENTION	SYSTEM RESPONSIBILITY	USER ACTION	SYSTEM RESPONSE
identify self		insert card	
	verify identity		read magnetic stripe
	offer choices		request PIN
choose		enter PIN	
	dispense cash		verify PIN
take cash			display transaction option menu
		press key	
			display account menu
		press key	
			prompt for amount
		enter amount	
			display amount
		press key	
			return card
		take card	
			dispense cash
		take cash	

Card Sorting

- Technique used to understand how users envision the organization of information.
- It consists on asking users to categorize a list of terms
  - Useful when a list of terms is already available
- Good for defining information architectures (for websites)
- Advantages:
  - Simple, easy to understand and apply
  - Inexpensive
  - Quick

Card Sorting: The Sorting Session

- Explain the card sorting process to the participants.
- Encourage them to organize the cards in a way that makes sense to them
- After the participants are done grouping the topics, they are asked to label each one of the resulting groups

Scenarios

- Personalized, fictional story with characters, events, products and environments
- Scenarios help the development team to explore ideas and the ramifications of design solutions in particular situations
- It is an encapsulated description of:
  - An individual user
  - Using a specific set of computer facilities
  - To achieve a specific outcome
  - Under specified circumstances
  - Over a certain time interval

Example of Scenario (I)

```
Path Smith has just arrived at Geneva International Airport en
    route to a large conference on Human-Computer Interaction.
    Pat is carrying a laptop and a large, heavy suitcase and
    needs to get to the conference centre quickly. Looking
    around for a bank in order to get some local currency, Pat
    sees the Eurochange machine with its blue flag style logo
    showing a circle of twelve stars.
Pat goes up the machine. It seems similar to the automatic
    teller machine that Pat uses regularly. Pat puts down the
    suitcase, takes out a credit card and inserts it into the
    slot. A message is displayed on the screen:
        Enter your PIN
Pat thinks for a few moments and then types a four-digit number
    on the numerical pad, listening to the reassuring beep that
    follows each number pressed. The machine pauses for a few
    seconds and then displays:
        Select currency required
```

Example of Scenario (II)

Pat pauses again. What is the currency in Switzerland? Pat browses the currencies available, sees “Swiss Franc (CHF)” and presses the key. The machine displays the message:

Exchange rate is 1.47 CHF to 1 EUR
Enter amount required in Swiss Francs in units of [10]
Press <Proceed>

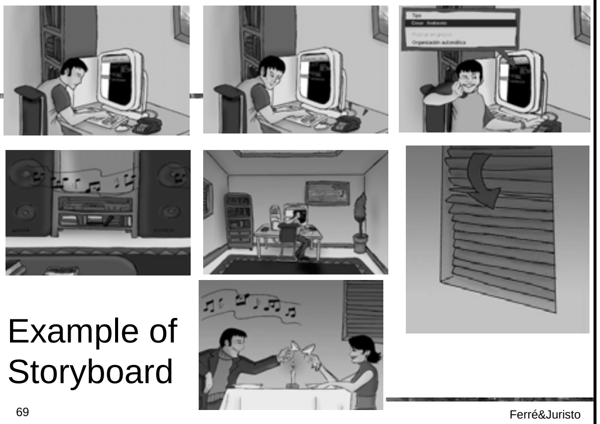
Pat types 253 and presses <Proceed>. A message is displayed:

Machine deals in bank notes only
Smallest bank note is [10] CHF
Enter new amount to obtain CHF or press <Cancel>

Pat enters 260 and presses <Proceed>. There is a whirring noise and a few other indeterminate clunks and clicks. The credit card is returned from the card entry slot and the money deposited in the delivery slot, with a printout of the transaction.

Storyboards

- Technique from the movie making industry
- A storyboard captures the procedure for doing a task pictorially
- Each frame in the storyboard captures a single scene
  - An interaction between two people
  - A person and the system
  - A person and an artifact
  - A system step
- It requires more drawing skills than paper prototypes



Example of Storyboard

Prototypes

- A prototype is a representation of all or part of a product or system that, although limited in some way, can be used for evaluation
- Good communication tool with users and other non-technical stakeholders
- Iterative development relies on prototyping techniques to a great extent
- HCI offers to software development a kind of cheap and quick prototypes, which are the less elaborate ones:
  - Paper and chauffeured prototypes
  - Wizard of Oz technique

70

Ferré&Juristo
ICSE 2006

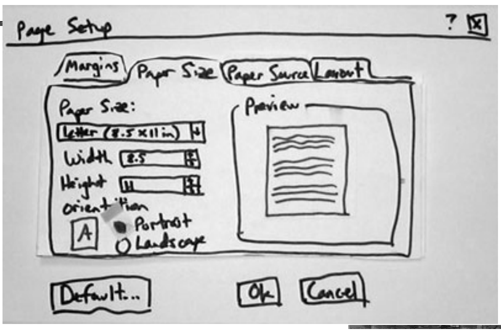
Paper Prototypes

- Rough prototypes in paper
- They serve to start the co-designing with users
- Paper prototypes are NOT demos
- They convey clearly to stakeholders that there is a lot of work to be done yet
  - And changes are easier to make because not a big amount of effort has been spent in their creation
- They allow for easy and quick changes

71

Ferré&Juristo
ICSE 2006

Paper Prototypes: Example



72

Ferré&Juristo
ICSE 2006

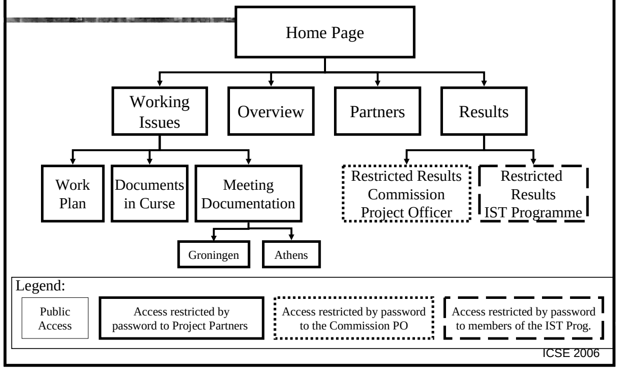
Chauffeured Prototypes

- Animation of the intended system behavior using paper prototypes
- Used for evaluation of prototypes with users, mimicking the usage of the system
- The user watches while a member of the development team “drives” the system
  - The user points out which actions he or she would take
  - The developer changes from one screen to another, and explains the intended system reaction to each user action

Menu Trees

- They represent
  - The menu navigational structure, or
  - The dialog boxes navigation
- The menu structure may be very big for medium-big sized projects
  - It can be mounted on a wall
- Card sorting may be used to decide on the menu structure
- For web development: Website maps

Website Map: Example


Home Page
Working Issues
Overview
Partners
Results
Work Plan
Documents in Curse
Meeting Documentation
Restricted Results Commission Project Officer
Restricted Results IST Programme
Groningen
Athens
Legend:
Public Access
Access restricted by password to Project Partners
Access restricted by password to the Commission PO
Access restricted by password to members of the IST Prog.

Website Map with Navigation Transitions: Example

76

Ferré&Juristo
ICSE 2006

Usability Inspections

- The goal of any inspection is to find defects
  - Usability inspections are aimed at identifying usability defects
- It is a systematic process (vs. heuristic evaluations which are less formal)
- Two types of inspections according to the focus:
  - Consistency inspections
  - Conformance inspections (with the style guide, or a standard)
- Collaborative usability inspection: A specific kind of inspection with participation of different stakeholders

77

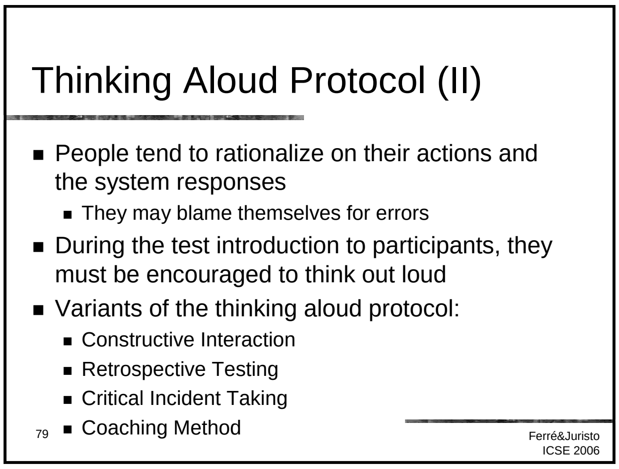
Ferré&Juristo
ICSE 2006

Thinking Aloud Protocol (I)

- The user is asked to talk out loud while trying to perform the tasks in a usability test session
  - What they are trying to do
  - Which problems they encounter and which strategy they devise in order to overcome them
  - Their (not fulfilled) expectations on the system behavior
  - How they interpret what the UI shows to them
- Not compatible with performance measures
  - Users are slowed down by the effort of verbalizing their thoughts

78

Ferré&Juristo
ICSE 2006

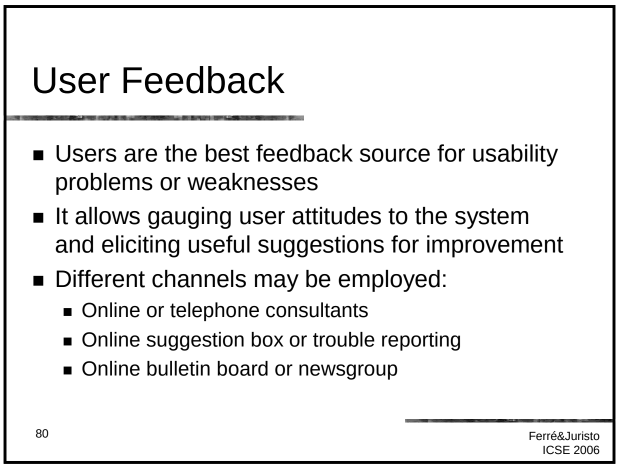


Thinking Aloud Protocol (II)

- People tend to rationalize on their actions and the system responses
  - They may blame themselves for errors
- During the test introduction to participants, they must be encouraged to think out loud
- Variants of the thinking aloud protocol:
  - Constructive Interaction
  - Retrospective Testing
  - Critical Incident Taking
  - Coaching Method

79

Ferré&Juristo
ICSE 2006



User Feedback

- Users are the best feedback source for usability problems or weaknesses
- It allows gauging user attitudes to the system and eliciting useful suggestions for improvement
- Different channels may be employed:
  - Online or telephone consultants
  - Online suggestion box or trouble reporting
  - Online bulletin board or newsgroup

80

Ferré&Juristo
ICSE 2006

HCI Technique Classification

1

Table 1 - Context of Use Specification-Related Techniques

ACTIVITY		TECHNIQUE	Hix, 93	Nielsen, 93	Preece, 94	Shneiderman, 98	Constantine, 99	Mayhew, 99
		Competitive Analysis		Competitive Analysis				
		Financial Impact Analysis		Financial Impact Analysis				
		Contextual Inquiry	Contextual Inquiry		Contextual Inquiry			Contextual Interviews
		Affinity Diagrams						Affinity Diagrams
		Ethnographic Observation			Ethnography	Ethnographic Observation		
		JEM[1]					JEM	
	User Analysis	**User Profiles**	User Profiles	Individual User Characteristics		Usage Profiles	Structured User Role Model	User Profiles Questionnaires
		User Role Map					User Role Map	
		Operational Modeling					Operational Modeling	Platform Capabilities and Constraints
		Personas	*This recent technique is not mentioned in any of the sources, but is included for the reasons detailed above in section 3.2.1*					
	Task Analysis	**Essential Use Cases**					Essential Use Cases	
		HTA[2]			HTA			
		GOMS[3]		GOMS	GOMS	GOMS		
		NGOMSL			NGOMSL	NGOMSL		
		Object-Action Interface Model				Object-Action Interface Model		
		Task Scenarios						Task Scenarios
		Task Sorting[4]						Task Sorting

Table 2 – Usability Specifications-Related Techniques

ACTIVITY	TECHNIQUE	Hix, 93	Nielsen, 93	Preece, 94	Shneiderman, 98	Constantine, 99	Mayhew, 99
Usability Specifications	**Usability Specification**	Usability Specifications	Usability Goals	Usability Specifications			Usability Goals
	Performance Goals	Objective Measures					Performance Goals
	Satisfaction Goals	Subjective Measures					Satisfaction Goals
	Usability Goal Line		**Usability Goal Line**				
	Preference Goals						Preference Goals

[1] JEM: Joint Essential Modeling
[2] HTA: Hierarchical Task Analysis
[3] GOMS: Goals, Operations, Methods and Selection Rules
[4] Task Sorting is a variation on the Card Sorting technique

ACTIVITY	TECHNIQUE	Hix, 93	Nielsen, 93	Preece, 94	Shnei-derman, 98	Const-antine, 99	Mayhew, 99
	Qualitative Goals						Qualitative Goals

Table 3 - Analysis-Related Techniques not Specific to any Activity

ACTIVITY	TECHNIQUE	Hix, 93	Nielsen, 93	Preece, 94	Shneider-man, 98	Constantine, 99	Mayhew, 99
Analysis generally	**Card Sorting**		Card Sorting			Card Sorting	
	Affinity Clustering					Affinity Clustering	
	Criteria Prioritization					Criteria Prioritization	
	Threshold Voting					Threshold Voting	
	Task Sorting						Task Sorting

Table 4 - Design-Related Techniques

ACTIVITY	TECHNIQUE	Hix, 93	Nielsen, 93	Preece, 94	Shneiderman, 98	Constantine, 99	Mayhew, 99
Develop Product Concept	**Scenarios and Storyboards**			Scenarios, Storyboards and Snapshots	Scenarios	Scenarios and Storyboards	
	Visual Brainstorming			Visual Brainstorming			
Prototyping	**Prototyping**	Prototyping	Prototyping				
	Scenario Prototypes		Scenarios				
	Active Prototypes			Requirements Animation		Active Prototypes	“High-Fidelity” Mock-ups
	Paper Prototypes			Low-Fidelity Mock-ups		Passive Prototypes	“Low-Fidelity” Mock-ups
	Chauffeured Prototypes			Chauffeured Prototyping			
	Wizard of Oz Prototypes			Wizard of Oz Prototypes			
Interaction Design	**Screen Pictures**	Scenarios and Screen Pictures					
	Product Style Guide						Product Style Guide
	Grammars				Grammars		
	UAN[5]	UAN			UAN		
	TAG[6]				TAG		
	Menu-Selection and Dialog Box Trees				Menu-Selection and Dialog Box Trees		
	Interface State Transition Diagrams	Interface State Transition Diagrams			Transition Diagrams		
	Statecharts				Statecharts		
	Content Model of Interface					Content Model of Interface	

[5] UAN: User Action Notation
[6] TAG: Task-Action Grammars

ACTIVITY	TECHNIQUE	Hix, 93	Nielsen, 93	Preece, 94	Shneiderman, 98	Constantine, 99	Mayhew, 99
	Context Navigation Map					Context Navigation Map	

Tabla 5 - Design-Related Techniques not Specific to any Activity

ACTIVITY	TECHNIQUE	Hix, 93	Nielsen, 93	Preece, 94	Shneider-man, 98	Constantine, 99	Mayhew, 99
Design generally	**Both-And Design**					Both-And Design	
	Parallel Design		Parallel Design				
	Impact Analysis	Cost / Importance Analysis	Impact Analysis	Impact Analysis			
	Organizing Help by Use Cases					Organizing Help by Use Cases	
	IBIS[7] and PHL[8]			IBIS and PHL			
	Design Space Analysis			Design Space Analysis			
	Claims Analysis			Claims Analysis			

Table 6 - Usability Evaluation Techniques for Expert Reviews

ACTIVITY	TECHNIQUE		Hix, 93	Nielsen, 93	Preece, 94	Shneider-man, 98	Constantine, 99	Mayhew, 99
Expert Reviews	**Heuristic Evaluation**		Heuristic Evaluation	Heuristic Evaluation	Heuristic Evaluation	Heuristic Evaluation	Heuristic Evaluation	Heuristic Evaluation
	Inspections	**Conformance Inspections**			Standards Inspections		Conformance Inspections	Standards Inspections
		Guideline Reviews				Guideline Reviews		Guideline Reviews
		Consistency Inspection			Consistency Inspection	Consistency Inspection	Consistency Inspection	Consistency Inspection
		Collaborative Usability Inspections					Collaborative Usability Inspections	
	Walkthroughs	**Cognitive**			Cognitive Walkthrough	Cognitive Walkthrough	Cognitive Walkthrough	Cognitive Walkthrough
		Pluralistic		Pluralistic Walkthrough	Pluralistic Walkthrough		Pluralistic Usability Walkthrough	Pluralistic Walkthrough

Table.7 - Usability Evaluation Techniques for Usability Testing

ACTI-VITY	TECHNIQUE		Hix, 93	Nielsen, 93	Preece, 94	Shnei-derman, 98	Constan-tine, 99	Mayhew, 99
Usability Testing	**Thinking Aloud**		Concurrent Verbal Protocol Taking	Thinking Aloud	Thinking Aloud Protocol		Talk to me (thinking aloud)	Formal usability inspections (in early stages)
		Constructive Interaction		Constructive Interaction				
		Retrospective Test	Retrospective Verbal Protocol Taking	Retrospective Testing	Post-Event Protocol		Deferred Reflection	

7 IBIS: Issue-Based Information Systems

8 PHI: Procedural Hierarchy of Issues

ACTI-VITY	TECHNIQUE		Hix, 93	Nielsen, 93	Preece, 94	Shnei-derman, 98	Constan-tine, 99	Mayhew, 99
		Critical Incident Taking	Criticial Incident Taking					
		Coaching Method		Coaching Method				
	Measured Performance				Benchmark Tasks		Measured Performance	Formal usability tests (in later stages)
	Post-Test Feedback						Post-Test Feedback	
	Laboratory Usability Testing		Laboratory Testing	Usability Laboratories		Usability Testing and Laboratories	Laboratory Usability Testing	
	Field Usability Testing		Field Testing				Field Testing	
	Video Recording		Video-taping	Video Recording	Video Recording			
	Verbal Protocol		Audio-taping		Verbal Protocol			
	Logging Actual Use		Internal Instrumentation of the Interface	Logging Actual Use	Software Logging	Continuous User-Performance Data Logging		
		Time-Stamped Keypresses			Time-Stamped Keypresses			
		Interaction Logging			Interaction Logging			
	Remote Control Evaluation							Remote Control Evaluation
	Remote Videoconferencing Evaluation							Remote Videoconferencing Evaluation

Table 8 - Usability Evaluation Techniques for Follow-up Studies of Installed Systems

ACTIVITY	TECHNIQUE		Hix, 93	Nielsen, 93	Preece, 94	Shnei-derman, 98	Constan-tine, 99	Mayhew, 99
Follow-up Studies of Installed Systems	**Direct Observation**			Observation	Direct Observation			
		Random Observation						Usage Studies - Random Observation
	Questionnaires and Surveys			Questionnaires	Questionnaires and Surveys	Surveys		
	Interviews			Interviews	Interviews	Interviews		
		Structured Interviews	Structured Interviews		Structured Interviews			
		Flexible Interviews			Flexible Interviews			
	Focus Groups			Focus Groups		Focus Group Discussions		
	Logging Actual Use		Internal Instrument ation of the Interface	Logging Actual Use	Software Logging	Continuous User-Performance Data Logging		Instrumented Remote Evaluation
		Time-Stamped Keypresses			Time-Stamped Keypresses			
		Interaction Logging			Interaction Logging			
		Random Activation Software Monitors						Usage Studies – Software Monitors
	User Feedback			User Feedback		Online Suggestion Box or Trouble Reporting		

ACTIVITY	TECHNIQUE		Hix, 93	Nielsen, 93	Preece, 94	Shnei-derman, 98	Constan-tine, 99	Mayhew, 99
		Online or Telephone Consultants				Online or Telephone Consultants		
		Online Bulletin Board or Newsgroups				Online Bulletin Board or Newsgroups		
		User Newsletters and Conferences				User Newsletters and Conferences		
		Semi-Instrumented Remote Evaluation						Semi-Instrumented Remote Evaluation

Table 9 - Other Usability Evaluation Techniques

ACTIVITY	TECHNIQUE			Hix, 93	Nielsen, 93	Preece, 94	Shneiderman, 98	Constantine, 99	Mayhew, 99
Usability evaluation generally	**Experimental Tests**					Traditional Tests	Controlled Psychologically Oriented Experiments		
	Predictive Metrics	**Procedural**						Procedural Metrics	
			Analytic Evaluation Methods			Analytic Evaluation Methods			
		Structural						Structural Metrics	
		Semantic						Semantic Metrics	
	Cooperative Evaluation					Cooperative Evaluation			

HCI Technique Characterization

2

Table 1 – Rating of Analysis-Related Techniques

Technique	U. P.	Training Needs	Applicability	Proximity to SE	Improvement/ Effort	Representa-tiveness	Overall Rating
Card Sorting	yes	low	high	medium	high	3	**Very useful**
Essential Use Cases	no	medium	high	high	high	1	**Very useful**
Personas	no	medium	medium	medium	high	3	**Very useful**
Usability Specifications	no	medium	medium	medium	high	4	**Very useful**
Affinity Diagrams	yes	low	high	medium	high	1	**Useful**
Competitive Analysis	no	medium	high	medium	high	1	**Useful**
Contextual Inquiry	yes	high	medium	medium	high	3	**Useful**
Ethnographic Observation	no	high	medium	medium	medium	2	**Useful**
HTA	no	medium	medium	high	medium	1	**Useful**
JEM	yes	medium	medium	high	medium	1	**Useful**
Task Scenarios	yes	medium	medium	medium	high	1	**Useful**
User Profiles	no	high	high	high	high	5	**Useful**
User Role Model	no	low	medium	high	medium	1	**Useful**
Financial Impact Analysis	no	high	medium	high	low	1	**Not very useful**
GOMS	no	very high	low	low	low	3	**Not very useful**
Object-Action Interface Model	no	high	low	medium	low	1	**Not very useful**
Operational Modeling	no	high	low	high	medium	2	**Not very useful**

Table 2 – Rating of Design-Related Techniques

Technique	U. P.	Training Needs	Applicability	Proximity to SE	Improvement/Effort	Representa-tiveness	Overall Rating
Menu-Selection and Dialog Box Trees	no	low	medium	high	high	1	**Very useful**
Paper Prototypes	yes	low	high	high	high	3	**Very useful**
Scenarios and Storyboards	yes	medium	medium	low	high	3	**Very useful**
Content Model of Interface	no	medium	high	medium	medium	1	**Useful**
Context Navigation Map	no	medium	high	high	medium	1	**Useful**
Impact Analysis	no	medium	medium	high	medium	3	**Useful**
Interface State Transition Diagrams	no	low	high	high	medium	2	**Useful**
Organizing Help by Use Cases	no	medium	medium	high	medium	1	**Useful**
Product Style Guide	no	high	medium	medium	medium	1	**Useful**
Visual Brainstorming	yes	low	high	low	high	1	**Useful**
Both-And Design	no	medium	medium	high	low	1	**Not very useful**
Grammars	no	medium	low	high	medium	1	**Not very useful**
Parallel Design	no	low	medium	high	low	1	**Not very useful**
Screen Pictures	no	medium	low	high	low	1	**Not very useful**
Statecharts	no	medium	low	high	low	1	**Not very useful**
TAG	no	very high	low	medium	low	1	**Not very useful**
UAN	no	very high	low	medium	low	2	**Not very useful**
Wizard of Oz Prototypes	yes	low	low	medium	medium	1	**Not very useful**

Table 3 – Rating of Evaluation-related Techniques

Technique	U. P.	Training Needs	Applicability	Proximity to SE	Improvement/Effort	Representa-tiveness	Overall Rating
Inspections	no	medium	high	medium	high	4	**Very useful**
Thinking Aloud	yes	medium	high	low	high	5	**Very useful**
User Feedback	yes	low	high	high	high	3	**Very useful**
Cognitive Walkthrough	no	high	medium	medium	medium	4	**Useful**
Collaborative Usability Inspections	yes	medium	medium	medium	medium	1	**Useful**
Heuristic Evaluation	no	high	high	low	high	6	**Useful**
Laboratory Usability Testing	yes	medium	medium	medium	medium	4	**Useful**
Logging Actual Use	no	high	medium	high	medium	5	**Useful**
Measured Performance	yes	medium	medium	medium	medium	3	**Useful**
Pluralistic Walkthrough	yes	low	medium	medium	medium	4	**Useful**
Post-Test Feedback	yes	medium	high	medium	high	1	**Useful**
Questionnaires, Interviews & Surveys	yes	medium	high	medium	medium	3	**Useful**
Audio-/Video-Taping	yes	medium	high	low	low	3	**Not very useful**
Cooperative Evaluation	yes	low	low	low	medium	1	**Not very useful**
Direct Observation	no	high	medium	low	low	3	**Not very useful**
Experimental Tests	yes	very high	low	low	low	2	**Not very useful**
Field Usability Testing	yes	medium	low	medium	medium	2	**Not very useful**
Focus Groups	yes	high	low	medium	medium	2	**Not very useful**
Predictive Metrics	no	very high	low	high	low	2	**Not very useful**

Assignment of HCI Techniques to SE Activity Groups

3

Table 1- Usability Techniques Grouped according to the SE Activity Group in which their Application is most Useful

SE Activity Type			Usability Technique
Analysis (Requirements Engineering)	Requirements Elictation and Analysis		Card Sorting
			Affinity Diagrams
			Competitive Analysis
			Contextual Inquiry
			Ethnographic Observation
			JEM
		User Analysis [*]	Personas
			User Profiles
			User Role Map
		Task Analysis [*]	Essential Use Cases
			HTA
			Task Scenarios
		Develop Product Concept [*]	Scenarios and Storyboards
			Visual Brainstorming
		Prototyping [*]	Paper Prototypes
	Requirements Specification		Usability Specifications
	Requirements Validation		Inspections
			Collaborative Usability Inspections
			Cognitive Walkthroughs
			Heuristic Evaluation
			Pluralistic Walkthroughs
Design	Interaction Design		Menu-Selection and Dialog Box Trees
			Context Navigation Map
			Content Model of Interface
			Interface State Transition Diagrams
			Product Style Guide
			Impact Analysis
			Organizing Help by Use Cases
Evaluation	Expert Reviews		Inspections
			Collaborative Usability Inspections
			Cognitive Walkthroughs
			Heuristic Evaluation
			Pluralistic Walkthroughs
	Usability Testing		Thinking Aloud
			Measured Performance
			Laboratory Usability Testing
			Post-Test Feedback
	Follow-up Studies of Installed Systems		User Feedback
			Logging Actual Use
			Questionnaires, Interviews and Surveys

[*] These are not SE activity types, but usability activities included to offer developers a structured view of the 15 usability techniques that can be applied for requirements elicitation and analysis.

4 Suitability of HCI Techniques to Development Stages

Table 1 – Technique Fit with Moment in Development Time

Content	Key
⬛	Especially Well-matched
▩	Neutral
⬜	Not Usual

Table 2 – How Selected Usability Techniques Fit the Iterative Development Stages

SE Activity Type			Usability Technique	Iterative Development Stages		
				Initial Cycles	Central Cycles	Evolution Cycles
Analysis (Requirements Engineering)	Requirements Elicitation and Analysis		Card Sorting	▨	▨	▨
			Competitive Analysis	■	▨	▨
			Affinity Diagrams	■	□	□
			Contextual Inquiry	■	□	□
			JEM	▨	▨	▨
			Ethnographic Observation	■	□	□
		User Analysis *	Personas	■	□	□
			User Role Map	■	▨	▨
			User Profiles	■	▨	▨
		Task Analysis *	Essential Use Cases	▨	▨	▨
			Task Scenarios	▨	▨	▨
			HTA	■	□	□
		Develop Product Concept *	Scenarios and Storyboards	■	□	□
			Visual Brainstorming	■	□	□
		Prototyping *	Paper Prototypes	■	▨	▨
	Requirements Specification		Usability Specifications	▨	▨	▨
	Requirements Validation		Inspections	▨	▨	▨
			Heuristic Evaluation	▨	▨	▨
			Collaborative Usability Inspections	■	▨	▨
			Cognitive Walkthroughs	▨	▨	▨
			Pluralistic Walkthroughs	■	▨	▨
Design	Interaction Design		Menu-Selection and Dialog Box Trees	▨	▨	▨
			Interface State Transition Diagrams	▨	▨	▨
			Product Style Guide	□	■	▨
			Context Navigation Map	▨	▨	▨
			Content Model of Interface	□	■	▨
			Impact Analysis	□	▨	▨
			Organizing Help by Use Cases	▨	▨	▨
Evaluation	Expert Reviews		Inspections	▨	▨	▨
			Heuristic Evaluation	▨	▨	▨
			Collaborative Usability Inspections	■	▨	▨
			Cognitive Walkthroughs	▨	▨	▨
			Pluralistic Walkthroughs	■	▨	▨
	Usability Testing		Thinking Aloud	□	▨	▨
			Post-Test Feedback	□	▨	▨
			Measured Performance	□	▨	▨
			Laboratory Usability Testing	□	▨	▨
	Follow-up Studies of Installed Systems		User Feedback	□	□	■
			Questionnaires, Interviews and Surveys	□	▨	■
			Logging Actual Use	□	▨	■

* These are not SE activity types but usability activities included to offer developers a structured view of the 15 usability techniques that can be applied for requirements elicitation and analysis.

Integration Framework Views

5

Table 1 – HCI Technique View: Proposed Usability Techniques in Alphabetical Order

Technique	U. P.	Training Needs	Applica-bility	Prox-imity to SE	Improve-ment/Ef-fort	Rep-resen ta-tive-ness	Overall Rating	Activity Type	Application Time			Basic Refer-ence
									Initial C.	Central C.	Evolution C.	
Affinity Diagrams	yes	low	high	medium	high	1	useful	Requirements Elicitation and Analysis	well-matched	not usual	not usual	[Beyer, 98]
Card Sorting	yes	low	high	medium	high	3	very useful	Requirements Elicitation and Analysis	neutral	neutral	neutral	[Robertson, 01]
Cognitive Walkthrough	no	high	medium	medium	medium	4	useful	Requirements Validation or Evaluation (Expert Reviews)	neutral	neutral	neutral	[Lewis, 97]
Collaborative Usability Inspections	yes	medium	medium	medium	medium	1	useful	Requirements Validation or Evaluation (Expert Reviews)	well-matched	neutral	neutral	[Constantine, 99]
Competitive Analysis	no	medium	high	medium	high	1	useful	Requirements Elicitation and Analysis	well-matched	neutral	neutral	[Nielsen, 93]
Content Model of Interface	no	medium	high	medium	medium	1	useful	Interaction Design	neutral	well-matched	neutral	[Constantine, 99]
Context Navigation Map	no	medium	high	high	medium	1	useful	Interaction Design	neutral	neutral	neutral	[Constantine, 99]
Contextual Inquiry	yes	high	medium	medium	high	3	useful	Requirements Elicitation and Analysis	well-matched	not usual	not usual	[Beyer, 98]
Essential Use Cases	no	medium	high	high	high	1	very useful	Requirements Elicitation and Analysis (Task Analy-sis)	neutral	neutral	neutral	[Constantine, 99]
Ethnographic Observation	no	high	medium	medium	medium	2	useful	Requirements Elicitation and Analysis	well-matched	not usual	not usual	[Wixon, 96]
Heuristic Evaluation	no	high	high	low	high	6	useful	Requirements Validation or Evaluación (Expert Reviews)	neutral	neutral	neutral	[Nielsen, 93]
HTA	no	medium	medium	high	medium	1	useful	Requirements Elicitation and Analysis (Task Analy-sis)	well-matched	not usual	not usual	[Annett, 04]
Impact Analysis	no	medium	medium	high	medium	3	useful	Design	not usual	neutral	neutral	[Hix, 93]
Inspections	no	medium	high	medium	high	4	very useful	Requirements Validation or Evaluation (Expert Reviews)	neutral	neutral	neutral	[Nielsen, 94]
Interface State Transition Diagrams	no	low	high	high	medium	2	useful	Interaction Design	neutral	neutral	neutral	[Wasserman, 85]
JEM	yes	medium	medium	high	medium	1	useful	Requirements Elicitation and Analysis	neutral	neutral	neutral	[Constantine, 99]
Laboratory Usability Testing	yes	medium	medium	medium	medium	4	useful	Evaluation (Usability Testing)	not usual	neutral	neutral	[Dumas, 99]
Logging Actual Use	no	high	medium	high	medium	5	useful	Evaluation (Follow-Up Studies of Installed Systems)	not usual	neutral	well-matched	[Shneiderman, 98]
Measured Performance	yes	medium	medium	medium	medium	3	useful	Evaluation (Usability Testing)	not usual	neutral	neutral	[Dumas, 99]

Technique	U. P.	Training Needs	Applica-bility	Prox-imity to SE	Improve-ment/Ef-fort	Rep-resen ta-tive-ness	Overall Rating	Activity Type	Application Time			Basic Refer-ence
									Initial C.	Central C.	Evolution C.	
Menu-Selection and Dialog Box Trees	no	low	medium	high	high	1	very useful	Interaction Design	neutral	neutral	neutral	[Shneiderman, 98]
Organizing Help by Use Cases	no	medium	medium	high	medium	1	useful	Design	neutral	neutral	neutral	[Constantine, 99]
Paper Prototypes	yes	low	high	high	high	3	very useful	Requirements Elicitation and Analysis (Prototyping)	well-matched	neutral	neutral	[Snyder, 03]
Personas	no	medium	medium	medium	high	3	very useful	Requirements Elicitation and Analysis (User Analy-sis)	well-matched	not usual	not usual	[Cooper, 03a]
Pluralistic Walkthrough	yes	low	medium	medium	medium	4	useful	Requirements Validation or Evaluation (Expert Reviews)	well-matched	neutral	neutral	[Bias, 94]
Post-Test Feedback	yes	medium	high	medium	high	1	useful	Evaluación (Usability Testing)	not usual	neutral	neutral	[Constantine, 99]
Product Style Guide	no	high	medium	medium	medium	1	useful	Interaction Design	not usual	well-matched	neutral	[Mayhew, 99]
Questionnaires, Inteviews and Surveys	yes	medium	high	medium	medium	3	useful	Evaluation (Follow-Up Studies of Installed Systems)	not usual	neutral	well-matched	[Mayhew, 99]
Scenarios and Storyboards	yes	medium	medium	low	high	3	very useful	Requirements Elicitation and Analysis (Develop Product Concept)	well-matched	not usual	not usual	[Carroll, 97]
Task Scenarios	yes	medium	medium	medium	high	1	useful	Requirements Elicitation and Analysis (Task Analy-sis)	neutral	neutral	neutral	[Mayhew, 99]
Thinking Aloud	yes	medium	high	low	high	5	very useful	Evaluation (Usability Testing)	not usual	neutral	neutral	[Nielsen, 93]
Usability Specifications	no	medium	medium	medium	high	4	very useful	Requirements Specification	neutral	neutral	neutral	[Hix, 93]
User Feedback	yes	low	high	high	high	3	very useful	Evaluation (Follow-Up Studies of Installed Systems)	not usual	not usual	well-matched	[Shneiderman, 98]
User Profiles	no	high	high	high	high	5	useful	Requirements Elicitation and Analysis (User Analy-sis)	well-matched	neutral	neutral	[Mayhew, 99]
User Role Map	no	low	medium	high	medium	1	useful	Requirements Elicitation and Analysis (User Analy-sis)	well-matched	neutral	neutral	[Constantine, 99]
Visual Brainstorming	yes	low	high	low	high	1	useful	Requirements Elicitation and Analysis (Develop Product Concept)	well-matched	not usual	not usual	[Preece, 94]

Table 2 – Activity Type View: Proposed Usability Techniques Ranked by Activity Type

Activity Type			Technique	U. P.	Training Needs	Applica-bility	Prox-imity to SE	Improve-ment/Ef-fort	Rep-resen-tative-ness	Overall Rating	Application Time			Basic Refer-ence
											Initial C.	Central C.	Evolution C.	
Analysis (Requirements Engineering)	Requirements Elicitation and Analysis		**Card Sorting**	yes	low	high	medium	high	3	very useful	neutral	neutral	neutral	[Robertson, 01]
			Competitive Analysis	no	medium	high	medium	high	1	useful	well-matched	neutral	neutral	[Nielsen, 93]
			Affinity Diagrams	yes	low	high	medium	high	1	useful	well-matched	not usual	not usual	[Beyer, 98]
			Contextual Inquiry	yes	high	medium	medium	high	3	useful	well-matched	not usual	not usual	[Beyer, 98]
			JEM	yes	medium	medium	high	medium	1	useful	neutral	neutral	neutral	[Constantine, 99]
			Ethnographic Observation	no	high	medium	medium	medium	2	useful	well-matched	not usual	not usual	[Wixon, 96]
		User Analysis [*]	**Personas**	no	medium	medium	medium	high	3	very useful	well-matched	not usual	not usual	[Cooper, 03a]
			User Role Map	no	low	medium	high	medium	1	useful	well-matched	neutral	neutral	[Constantine, 99]
			User Profiles	no	high	high	high	high	5	useful	well-matched	neutral	neutral	[Mayhew, 99]
		Task Analysis [*]	**Essential Use Cases**	no	medium	high	high	high	1	very useful	neutral	neutral	neutral	[Constantine, 99]
			HTA	no	medium	medium	high	medium	1	useful	well-matched	not usual	not usual	[Annett, 04]
			Task Scenarios	yes	medium	medium	medium	high	1	useful	neutral	neutral	neutral	[Mayhew, 99]
		Develop Product Concept [*]	**Scenarios and Storyboards**	yes	medium	medium	low	high	3	very useful	well-matched	not usual	not usual	[Carroll, 97]
			Visual Brainstorming	yes	low	high	low	high	1	useful	well-matched	not usual	not usual	[Preece, 94]
		Prototyping [*]	**Paper Prototypes**	yes	low	high	high	high	3	very useful	well-matched	neutral	neutral	[Snyder, 03]
	Requirements Specification		**Usability Specification**	no	medium	medium	medium	high	4	very useful	neutral	neutral	neutral	[Hix, 93]
	Requirements Validation		**Inspections**	no	medium	high	medium	high	4	very useful	neutral	neutral	neutral	[Nielsen, 94]
			Heuristic Evaluation	no	high	high	low	high	6	useful	neutral	neutral	neutral	[Nielsen, 93]

[*] These are not SE activity types, but usability activities included to offer developers a structured view of the 15 usability techniques that can be applied for requirements elicitation and analysis.

Activity Type		Technique	U. P.	Training Needs	Applica-bility	Prox-imity to SE	Improve-ment/Ef-fort	Rep-resen-tative-ness	Overall Rating	Application Time			Basic Refer-ence
										Initial C.	Central C.	Evolution C.	
		Collaborative Usability Inspec-tions	yes	medium	medium	medium	medium	1	useful	well-matched	neutral	neutral	[Constantine, 99]
		Cognitive Walkthroughs	no	high	medium	medium	medium	4	useful	neutral	neutral	neutral	[Lewis, 97]
		Pluralistic Walkthroughs	yes	low	medium	medium	medium	4	useful	well-matched	neutral	neutral	[Bias, 94]
Design	Interaction Design	**Menu-Selection and Dialog Box Trees**	no	low	medium	high	high	1	very useful	neutral	neutral	neutral	[Shneiderman, 98]
		Interface State Transition Dia-grams	no	low	high	high	medium	2	useful	neutral	neutral	neutral	[Wasserman, 85]
		Product Style Guide	no	high	medium	medium	medium	1	useful	not usual	well-matched	neutral	[Mayhew, 99]
		Context Navigation Map	no	medium	high	high	medium	1	useful	neutral	neutral	neutral	[Constantine, 99]
		Content Model of Interface	no	medium	high	medium	medium	1	useful	neutral	well-matched	neutral	[Constantine, 99]
		Impact Analysis	no	medium	medium	high	medium	3	useful	not usual	neutral	neutral	[Hix, 93]
		Organizing Help by Use Cases	no	medium	medium	high	medium	1	useful	neutral	neutral	neutral	[Constantine, 99]
Evaluation	Expert Reviews	**Inspections**	no	medium	high	medium	high	4	very useful	neutral	neutral	neutral	[Nielsen, 94]
		Heuristic Evaluation	no	high	high	low	high	6	useful	neutral	neutral	neutral	[Nielsen, 93]
		Collaborative Usability Inspec-tions	yes	medium	medium	medium	medium	1	useful	well-matched	neutral	neutral	[Constantine, 99]
		Cognitive Walkthroughs	no	high	medium	medium	medium	4	useful	neutral	neutral	neutral	[Lewis, 97]
		Pluralistic Walkthroughs	yes	low	medium	medium	medium	4	useful	well-matched	neutral	neutral	[Bias, 94]
	Usability Testing	**Thinking Aloud**	yes	medium	high	low	high	5	very useful	not usual	neutral	neutral	[Nielsen, 93]
		Post-Test Feedback	yes	medium	high	medium	high	1	useful	not usual	neutral	neutral	[Constantine, 99]
		Measured Performance	yes	medium	medium	medium	medium	3	useful	not usual	neutral	neutral	[Dumas, 99]
		Laboratory Usability Testing	yes	medium	medium	medium	medium	4	useful	not usual	neutral	neutral	[Dumas, 99]
	Follow-up Studies of Installed Systems	**User Feedback**	yes	low	high	high	high	3	very useful	not usual	not usual	well-matched	[Shneiderman, 98]
		Questionnaires, Interviews and Surveys	yes	medium	high	medium	medium	3	useful	not usual	neutral	well-matched	[Mayhew, 99]
		Logging Actual Use	no	high	medium	high	medium	5	useful	not usual	neutral	well-matched	[Shneiderman, 98]

Table 3 –Application Times View: Proposed Usability Techniques Ranked by Best Time for Application in Iterative Development

Stage in Iterative Cycle		Technique	U. P.	Training Needs	Applicability	Proximity to SE	Improvement/Effort	Representativeness	Overall Rating	Activity Type	Basic Reference
Initial Cycles	Especially Well-matched	Paper Prototypes	yes	low	high	high	high	3	very useful	Req. Elicitation and Analysis (Prototyping)	[Snyder, 03]
		Personas	no	medium	medium	medium	high	3	very useful	Req. Elicitation and Analysis (User Analysis)	[Cooper, 03a]
		Scenarios and Storyboards	yes	medium	medium	low	high	3	very useful	Req. Elicitation and Analysis (Develop Product Concept)	
		Competitive Analysis	no	medium	high	medium	high	1	useful	Req. Elicitation and Analysis	[Nielsen, 93]
		Affinity Diagrams	yes	low	high	medium	high	1	useful		[Beyer, 98]
		Contextual Inquiry	yes	high	medium	medium	high	3	useful		[Beyer, 98]
		Ethnographic Observation	no	high	medium	medium	medium	2	useful		[Wixon, 96]
		User Profiles	no	high	high	high	high	5	useful	Req. Elicitation and Analysis (User Analysis)	[Mayhew, 99]
		User Role Map	no	low	medium	high	medium	1	useful		[Constantine, 99]
		HTA	no	medium	medium	high	medium	1	useful		[Annett, 04]
		Visual Brainstorming	yes	low	high	low	high	1	useful	Req. Elicitation and Analysis (Develop Product Concept)	[Preece, 94]
		Collaborative Usability Inspections	yes	medium	medium	medium	medium	1	useful	Requirements Validation or Evaluation (Expert Reviews)	[Constantine, 99]
		Pluralistic Walkthrough	yes	low	medium	medium	medium	4	useful		[Bias, 94]
	Neutral	Card Sorting	yes	low	high	medium	high	3	very useful	Req. Elicitation and Analysis	[Robertson, 01]
		Essential Use Cases	no	medium	high	high	high	1	very useful	Req. Elicitation and Analysis (Task Analysis)	[Constantine, 99]
		Usability Specifications	no	medium	medium	medium	high	4	very useful	Requirements Specification	[Hix, 93]
		Inspections	no	medium	high	medium	high	4	very useful	Requirements Validation or Evaluation (Expert Reviews)	[Nielsen, 94]
		Menu-Selection and Dialog Box Trees	no	low	medium	high	high	1	very useful	Interaction Design	[Shneiderman, 98]
		JEM	yes	medium	medium	high	medium	1	useful	Req. Elicitation and Analysis	[Constantine, 99]
		Task Scenarios	yes	medium	medium	medium	high	1	useful	Req. Elicitation and Analysis (Task Analysis)	[Mayhew, 99]
		Heuristic Evaluation	no	high	high	low	high	6	useful	Requirements Validation or Evaluation (Expert Reviews)	[Nielsen, 93]
		Cognitive Walkthrough	no	high	medium	medium	medium	4	useful		[Lewis, 97]
		Interface State Transition Diagrams	no	low	high	high	medium	2	useful	Interaction Design	[Wasserman, 85]
		Context Navigation Map	no	medium	high	high	medium	1	useful		[Constantine, 99]
		Organizing Help by Use Cases	no	medium	medium	high	medium	1	useful	Design	[Constantine, 99]
Central Cycles	Especially Well-matched	Product Style Guide	no	high	medium	medium	medium	1	useful	Interaction Design	[Mayhew, 99]
		Content Model of Interface	no	medium	high	medium	medium	1	useful		[Constantine, 99]
	Neutral	Card Sorting	yes	low	high	medium	high	3	very useful	Req. Elicitation and Analysis	[Robertson, 01]
		Essential Use Cases	no	medium	high	high	high	1	very useful	Req. Elicitation and Analysis (Task Analysis)	[Constantine, 99]

Stage in Iterative Cycle		Technique	U. P.	Training Needs	Applica-bility	Prox-imity to SE	Improve-ment/Ef-fort	Rep-resen-tative-ness	Over-all Rat-ing	Activity Type	Basic Reference
		Paper Prototypes	yes	low	high	high	high	3	very useful	Req. Elicitation and Analysis (Prototyp-ing)	[Snyder, 03]
		Usability Specifications	no	medium	medium	medium	high	4	very useful	Requirements Specification	[Hix, 93]
		Inspections	no	medium	high	medium	high	4	very useful	Requirements Validation or Evaluation (Expert Reviews)	[Nielsen, 94]
		Menu-Selection and Dialog Box Trees	no	low	medium	high	high	1	very useful	Interaction Design	[Shneider-man, 98]
		Thinking Aloud	yes	medium	high	low	high	5	very useful	Evaluation (Usabil-ity Testing)	[Nielsen, 93]
		Competitive Analysis	no	medium	high	medium	high	1	useful	Req. Elicitation and Analysis	[Nielsen, 93]
		JEM	yes	medium	medium	high	medium	1	useful		[Constantine, 99]
		User Role Map	no	low	medium	high	medium	1	useful	Req. Elicitation and Analysis (User Analysis)	[Constantine, 99]
		User Profiles	no	high	high	high	high	5	useful		[Mayhew, 99]
		Task Scenarios	yes	medium	medium	medium	high	1	useful	Req. Elicitation and Analysis (Task Analysis)	[Mayhew, 99]
		Heuristic Evaluation	no	high	high	low	high	6	useful	Requirements Validation or Evaluation (Expert Reviews)	[Nielsen, 93]
		Collaborative Usability Inspections	yes	medium	medium	medium	medium	1	useful		[Constantine, 99]
		Cognitive Walkthrough	no	high	medium	medium	medium	4	useful		[Lewis, 97]
		Pluralistic Walkthrough	yes	low	medium	medium	medium	4	useful		[Bias, 94]
		Interface State Transition Diagrams	no	low	high	high	medium	2	useful	Interaction Design	[Wasserman, 85]
		Context Navigation Map	no	medium	high	high	medium	1	useful		[Constantine, 99]
		Impact Analysis	no	medium	medium	high	medium	3	useful	Design	[Hix, 93]
		Organizing Help by Use Cases	no	medium	medium	high	medium	1	useful		[Constantine, 99]
		Measured Performance	yes	medium	medium	medium	medium	3	useful	Evaluation (Usabil-ity Testing	[Dumas, 99]
		Post-Test Feedback	yes	medium	high	medium	high	1	useful		[Constantine, 99]
		Laboratory Usability Testing	yes	medium	medium	medium	medium	4	useful		[Dumas, 99]
		Questionnaires, Inteviews and Surveys	yes	medium	high	medium	medium	3	useful	Evaluation (Fol-low-up Studies of Installed Systems)	[Mayhew, 99]
		Logging Actual Use	no	high	medium	high	medium	5	useful		[Shneider-man, 98]
Evolu-tion Cycles	Espe-cially Well-matched	User Feedback	yes	low	high	high	high	3	very useful	Evaluation (Fol-low-up Studies of Installed Systems)	[Shneider-man, 98]
		Questionnaires, Inteviews and Surveys	yes	medium	high	medium	medium	3	useful	Evaluation (Fol-low-up Studies of Installed Systems)	[Mayhew, 99]
		Logging Actual Use	no	high	medium	high	medium	5	useful		[Shneider-man, 98]
	Neutral	Card Sorting	yes	low	high	medium	high	3	very useful	Req. Elicitation and Analysis	[Robertson, 01]
		Essential Use Cases	no	medium	high	high	high	1	very useful	Req. Elicitation and Analysis (Task Analysis)	[Constantine, 99]
		Paper Prototypes	yes	low	high	high	high	3	very useful	Req. Elicitation and Analysis (Prototyp-ing)	[Snyder, 03]
		Usability Specifications	no	medium	medium	medium	high	4	very useful	Requirements Specification	[Hix, 93]
		Inspections	no	medium	high	medium	high	4	very useful	Requirements Validation or Evaluation (Expert Reviews)	[Nielsen, 94]
		Menu-Selection and Dialog Box Trees	no	low	medium	high	high	1	very useful	Interaction Design	[Shneider-man, 98]

Stage in Iterative Cycle	Technique	U. P.	Training Needs	Applica-bility	Prox-imity to SE	Improve-ment/Ef-fort	Rep-resen-tative-ness	Over-all Rat-ing	Activity Type	Basic Reference
	Thinking Aloud	yes	medium	high	low	high	5	very useful	Evaluation (Usabil-ity Testing)	[Nielsen, 93]
	Competitive Analysis	no	medium	high	medium	high	1	useful	Req. Elicitation and Analysis	[Nielsen, 93]
	JEM	yes	medium	medium	high	medium	1	useful		[Constantine, 99]
	User Role Map	no	low	medium	high	medium	1	useful	Req. Elicitation and Analysis (User Analysis)	[Constantine, 99]
	User Profiles	no	high	high	high	high	5	useful		[Mayhew, 99]
	Task Scenarios	yes	medium	medium	medium	high	1	useful	Req. Elicitation and Analysis (Task Analysis)	[Mayhew, 99]
	Heuristic Evaluation	no	high	high	low	high	6	useful	Requirements Validation or Evaluation (Expert Reviews)	[Nielsen, 93]
	Collaborative Usability Inspections	yes	medium	medium	medium	medium	1	useful		[Constantine, 99]
	Cognitive Walkthrough	no	high	medium	medium	medium	4	useful		[Lewis, 97]
	Pluralistic Walkthrough	yes	low	medium	medium	medium	4	useful		[Bias, 94]
	Interface State Transition Diagrams	no	low	high	high	medium	2	useful	Interaction Design	[Wasserman, 85]
	Product Style Guide	no	high	medium	medium	medium	1	useful		[Mayhew, 99]
	Content Model of Inter-face	no	medium	high	medium	medium	1	useful		[Constantine, 99]
	Context Navigation Map	no	medium	high	high	medium	1	useful		[Constantine, 99]
	Impact Analysis	no	medium	medium	high	medium	3	useful	Design	[Hix, 93]
	Organizing Help by Use Cases	no	medium	medium	high	medium	1	useful		[Constantine, 99]
	Measured Performance	yes	medium	medium	medium	medium	3	useful	Evaluation (Usabil-ity Testing	[Dumas, 99]
	Post-Test Feedback	yes	medium	high	medium	high	1	useful		[Constantine, 99]
	Laboratory Usability Testing	yes	medium	medium	medium	medium	4	useful		[Dumas, 99]

Description and Basic Reference for HCI Techniques

6

Designing Usable Software

How to Enrich Software Development with Usability Activities and Techniques

Xavier Ferré

xavier@fi.upm.es

Universidad Politécnica de Madrid (Spain)

Table of Contents

Introduction

This document presents usability techniques that may be applied in any iterative development process, organized according to the kind of activity where they fit. For each usability technique a brief description is provided and a basic reference where additional information may be obtained.

I - Requirements Elicitation and Analysis

Requirements elicitation and analysis include the specification of the context of use. Its aim is to understand and record the implications of the context of use so that they can be considered during system design. There is no single best interaction style or approach suitable for just any user. Specific design alternatives that optimize the performance of some types of users may actually degrade performance of other types of users. Therefore, it is very important to know the particular users that the system is built for, and the kinds of tasks they want to perform using the system, and this is the aim of the specification of the context of use. Context of use is a broad term that comprises different interrelated aspects:

- The characteristics of the intended users. The identification of these characteristics is known as user analysis.
- The tasks the users are to perform. Task analysis deals with this issue.
- The environment in which the users are to use the system, including the hardware, software and materials to be used.

The development of the product concept is also part of requirements elicitation and analysis. The idea of a product concept is based on mental models: When the product concept is vague, ambiguous, inconsistent or obscure, there will be a divergence between the user mental model of the system and the design model that developers work with. Specifying a clear product concept ensures a proper communication between the members of the development team, and the production of a design that clearly conveys the model to the user.

Finally, prototypes allow designers to communicate more effectively with users and they reduce the need and cost of reworking that can occur when products need to be revised later in the life cycle. They are employed as part of requirements elicitation and analysis activities as well.

I.1 Affinity Diagrams

Affinity diagrams serve for grouping and understanding information.

In order to conduct an affinity diagram session, begin by handing out Post-It notes. Then, ask participants to write one issue on each note.

You can give participants some minutes for this activity, but ask them to stop when a large majority of participants have stopped. Get all participants to gather at a vertical surface suitable for Post-It notes. Windows are appropriate. Encourage participants to place notes, one at a time, on the surface. As each note is placed, other participants may add similar notes in close proximity. Depending on the amount of time, the information being analyzed, and on group dynamics (and patience) it may be worth spending some additional time considering and rearranging the groups. When all notes have been placed and grouped, you can optionally name each group.

If there are more than 8 people, gathering around a common area may not be convenient. In this case, you can handle all the note-placing yourself-get one note from each person in turn; all participants can then pass you any similar or related notes. This is not as satisfactory as having the group work together, since it is difficult to keep everyone focused on the task.

Affinity diagramming is best used if the work can be followed up quickly. For example, affinity diagramming of issues can lead into discussion of methods to address the issues. Nevertheless, the resultant groupings are arbitrary, so they should be used with a flexible approach (i.e. they are not the absolute truth).

Advice on conducting an affinity diagram session:

- You should allow all participants to contribute. There may be an individual who wants to take control of positioning and moving the notes. Do not allow this to happen. Do not move someone's note without their agreement. Discussion will often indicate that the participant wanted to articulate a different issue.
- Encourage participants to read their notes aloud while placing them on the surface.
- Do not allow the activity to continue past the point of tiredness or boredom, because it is a tiring effort. Avoid having more than two consecutive affinity diagramming sessions during a workshop.

Basic Reference: H. Beyer, K. Holtzblatt. *Contextual Design. Defining Customer-Centered Systems*. Morgan Kaufmann, 1998.

I.2 Card Sorting

At the beginning of any exercise of information architecture design, it is normal to be confronted by a very long list of potential subjects to include. The challenge is to organize this information in a way that is useful and meaningful for the users of the system. While careful investigation and analysis of the information may reveal some clues, making relevant users organize the subjects may be more cost-effective. Therefore, card sorting consists on asking users to categorize a list of terms. It is useful when such a list already exists.

Card sorting is very similar to affinity diagrams, but it is aimed to the design of an information structure (e.g., a website, or a menu tree).

When creating the list of topics some issues should be considered:

- The length of the list should be manageable
- Existing structures should not be reflected
- Do not put “clues” that will lead users to arrange the topics in a particular way
- Each topic should not be too general nor too specific
- The topics chosen must be meaningful to the participants in the session

First you should create the cards, and then select the participants. They should be the actual end-users of the system you are building. If your participant is the manager of the user instead of the end user himself or herself, the results will not reflect a mental model which is natural to the end user. Each session must be dedicated to a homogeneous group of users. If several different groups of users must be tackled, hold a different card sorting session for each one.

In the sorting session you should carefully explain the mechanics and the expected outcome of the session to participants. Then, you should encourage participants to organize the cards in a way that makes sense to them. When they are finished with the grouping, ask them to label each category with a card. You may allow a “to be determined” or “not sure” pile to be created, but do not tell them so at the beginning of the exercise.

Use the notebook to take notes about any important event arising during the session:

- User questions,
- user comments or suggestions, and
- user non-verbal behaviors.

Bear in mind that a card sorting exercise does not produce a finished information design, it is just an elaborate guidance, which is very useful because it has been built according to the user expectations.
Basic Reference: J. Robertson. *Information Design Using Card Sorting*. Step Two Designs Pty Ltd, 2001. http://www.steptwo.com.au/papers/cardsorting/

I.3 Competitive Analysis

Competitive analysis basically involves studying existing products to find out their strengths and weaknesses. These analyzed products may be competing products, but they may also be products from different fields that address issues that are similar to the ones the system will have to deal with. Commercial products that are widely known serve as good references for establishing the product concept, and a competitive analysis of their benefits from a usability point of view can help to focus the discussion and the decision-making process.
If several competing products are available for analysis, we can perform a comparative analysis of their differing approaches to support the user goals. This will provide ideas for the system we are developing, especially for developing the product concept. It can also provide a list of ad hoc guidelines for approaches to specific issues that seem to work, and things that should be avoided.
Basic Reference: J. Nielsen. *Usability Engineering*. AP Professional, 1993. pp. 78-79.

I.4 Contextual Inquiry

In a contextual inquiry interview, the developer sits beside the user and observes how he or she performs the work, interrupting him or her every now and then to ask why a particular action has been taken or what its purpose is. The interview should be performed in an environment that is as close as possible to the usual working environment of the user, because the developer is looking for first-hand knowledge, the kind of understanding about the work structure that the user cannot formulate, unless he or she is performing the work at that very time.
The interview should be a combination of watching and probing.
A sense of partnership should be formed between observer and user, in the sense that they are both looking to explain the internal logic behind the user's actions, as there can be a lot of tasks that the user does routinely and he or she cannot completely explain. Therefore, the observer must try to make out the work structure and find patterns and distinctions in the way people organize work. Not only the observer gains a better understanding of the user's work, the user himself or herself also acquires increased insight into his or her work by being forced to look at it from an external perspective. Users themselves are sometimes surprised about some of the actions they perform routinely when they look at them from an analytical point of view.
A good approach to contextual inquiry is the master / apprentice model, where the user is the master, and the observer is the apprentice that wants to learn how the work is done. This model of action for the observer aims to make the user focus on his or her work. Nevertheless, the master / apprentice model

should not be taken literally, since the observer must play an active role in probing the user every time he or she needs an explanation.

Note that competitive analysis is a useful technique for vendor organizations, while observational techniques are more suited for bespoke systems (nevertheless they may be useful as well for vendor organizations when a particular kind of user is targeted).

Basic Reference: H. Beyer, K. Holtzblatt. *Contextual Design. Defining Customer-Centered Systems*. Morgan Kaufmann, 1998. Chapters 3 and 4.

I.5 Ethnographic Observation

Observational techniques are based on performing elicitation at the customer organization, at the users' workplace, observing how they work. The work may be performed either manually, using a competitor product or using a previous version of the software product under development.

Ethnography is a method belonging to Anthropology for studying a particular tribe or culture. The ethnographer participates, overtly or covertly, in people's daily lives for an extended period of time. When this approach is used for software development purposes, it provides developers information about the context where the user performs his or her tasks, which would be very difficult to apprehend otherwise (for example, by means of interviews). The difference between developers acting as ethnographers and anthropologists is that, apart from trying to understand the user, developers observe the usage of existing software products for the purpose of changing and improving those products. Additionally, the available time for ethnographic observation in software development is a lot less than the time anthropologists spend immersed in a culture.

Basic Reference: [Wixon, 96] D. Wixon, J. Ramey (editors). Field Methods Casebook for Software Design. John Wiley and Sons, 1996.

I.6 JEM (Joint Essential Modeling)

JEM is a structured, facilitated, collaborative process for concurrent modeling of users and use cases. It is based on the JAD (Joint Application Development) technique.

In JEM, users and developers join in a collaborative effort to define the essential models and reach agreement on core requirements. The objective is to reach consensus on the tasks to be supported by the system under development. Although other models play some part, the principal medium of exchange are use cases, whether essential or detailed.

The preparation and consolidation process prepares materials and an agenda for the subsequent sessions and also generates a candidate list of user roles used as a guide for participation in later joint modeling. Following the development of role and task models, these models are audited for completeness, correctness, and consistency. To complete the process, use cases are prioritized and allocated to project iterations. These activities are carried out in a series of sessions as follows:

- Framing session: The purpose of this session is to establish the framework within which the joint modeling sessions will operate.
- Modeling sessions: User role and use case models are developed collaboratively during these sessions.

- Review session: First the group reviews the models to ensure that they are complete, correct, and consistent. Second, use cases are sorted to identify which capabilities are to be supported and when.

The main roles in JEM are the users, the lead analyst (who ensures appropriate technique leadership and expertise in modeling), the facilitator, and the scribe. Other potential participants include a sponsor, and other members of the development team.

Basic Reference: L. L. Constantine, L. A. D. Lockwood. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design*. Addison-Wesley, New York, NY, 1999. pp. 499-509.

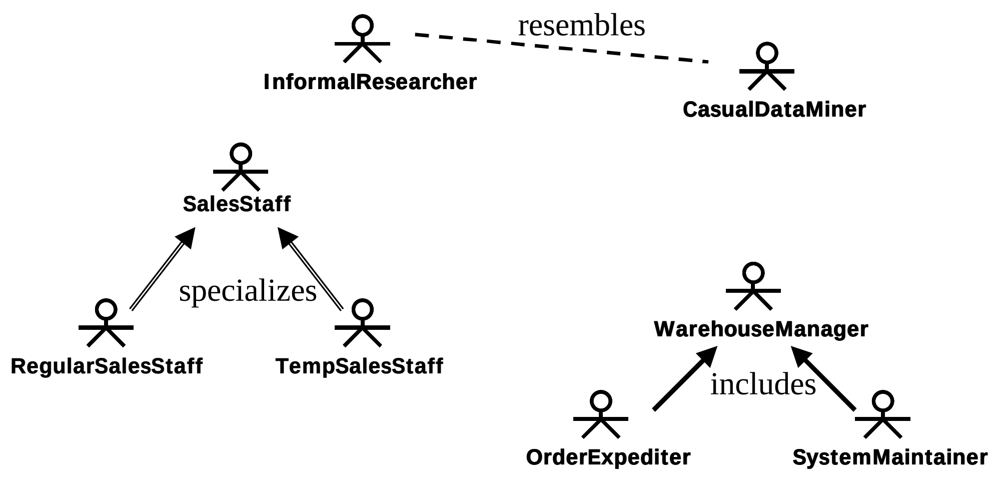
I.7 Structured User Role Model

The structured user role model is formed by profiles. The main profiles are as follows:

- Incumbents: Common characteristics that users who play a given role share. There are three categories into which the elements in this profile may fall: domain knowledge, system knowledge and other background knowledge.
- Proficiency: How usage proficiency is distributed over time and among users in a given role.
- Interaction: Patterns of usage associated with a given role. The kind of information in this profile falls in one or more of the following categories: frequency, regularity, continuity, concentration, intensity, complexity, predictability or locus of control.
- Information: Nature of the information manipulated by users in a role or exchanged between users and the system. The information in this profile may offer details on the input origins, the flow direction, the information volume and/or the information complexity.
- Usability criteria: Relative importance of specific usability attributes with respect to a given role.
- Functional support: Specific functions, features, or facilities needed to support users in a given role.

Some roles are highlighted as focal roles, which are the ones that are judged to be the most common or typical or that are deemed particularly important from a business perspective or from the standpoint of risk.

Any elicitation technique can be used to acquire the information about users to complete the profiles in the structured user role model. The more complex the user population for a system under development is, the more complete the profiles for a user role must be. For simple systems just some data on a few of the profiles may be enough to describe each user role, and then the user role model is not said to be structured.



This example of a user role map shows the user roles identified for a statistical analysis package.

We say that there is affinity between two roles, which is represented by a dashed line, if we identify some similarity or resemblance between them (InformalResearcher and CasualDataMiner have an affinity relationship). When a user role is a subtype of another, we say that the former specializes the latter, and this is represented by a double-lined arrow that goes from the more specific role to the more abstract one (RegularSalesStaff and TempSalesStaff are both subtypes of the general SalesStaff). Finally, there is a composition relationship when one role combines the characteristics or features of two or more other roles and is composed of these other roles, which we represent by a single-lined arrow (WarehouseManager includes both the OrderExpediter and the SystemMaintainer).

Basic Reference: L. L. Constantine, L. A. D. Lockwood. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design*. Addison-Wesley, New York, NY, 1999. Chapter 4. pp. 69-96.

I.8 Operational Modeling

Operational modeling is a collection of various operational and contextual influences that can play a role in usability:

- Operational Risk Profile: Operational risk refers to what is at stake if the user and the system fail to correctly complete tasks. For example, what are the consequences of an input error, a failure to complete a transaction, a system lockup, or a delay in processing. Where operational risk is higher in connection with particular roles or use cases, special attention needs to be paid to mechanisms that assure input accuracy and accurate interpretation of output.
- Device Constraints Profile: The device constraints profile identifies equipment characteristics associated with specific roles, use cases, or the system as a whole. There may be limitations on the input side, the output side, or both. These constraints include screen size, resolution, and color depth; keyboard or keypad size and layout; and special controls such as sliders, toggle switches, rotary knobs, or similar items. It is especially interesting to describe the device constraints in projects where the devices are fixed by economics or the user community.
- Environment Profile: The environment profile is formed of physical factors, such as the type of user location (office, home, factory, etc.), the level of ambient noise, the lighting conditions, temperature, humidity, or the presence of vibration. The information gathered may reflect any

kind of physical condition of the environment that may affect system usage. A key issue here is the level of distraction due to the physical environment. Distractions may be physical (like a noisy fan or repeated phone calls) or mental (like trying to remember to do something that must be temporarily postponed).

Basic Reference: L. L. Constantine, L. A. D. Lockwood. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design*. Addison-Wesley, New York, NY, 1999. pp. 308-313.

I.9 Personas

A persona is a fictional person we create to represent a particular class of real users or system participants. A persona is an archetypal user, who resembles real people (the people interviewed when user research is performed, for example), but does not exactly match any one of them.

This technique helps to determine what the product should do and how it should behave, to communicate with non-technical stakeholders because the natural language description of each persona is understandable by anyone. As it helps to build a common language, it helps reaching consensus in the team. Finally, personas may help in marketing and sales plans efforts.

There are three potential problems in interaction design that strongly affect to usability: The elastic user, self-referential design and the design for edge cases. The three of them are tackled when personas is used. Each persona has particular goals and behavior patterns.

Basic Reference: Cooper, A., Reimann, R. *About Face 2.0. The Essentials of Interaction Design*. Wiley Publishing, 2003. Chapter 5. pp. 55-74.

I.10 Essential Use Cases

A use case is a case of use, or one kind of use to which a system can be put. It is:

- Supplied functionality
- An external, “black-box” view
- A narrative description
- Interaction between a user and a system
- A use of the system that is completely meaningful to the user

In object-oriented analysis and design practice, the use-case model is very important in cycle planning, but once the cycle starts, use cases are regarded as a preliminary version of elements of the internal functionality design. When design elements are labeled as use-case realizations, we are shifting use cases to the design world and, therefore, away from the user realm, losing most user-centered advantages with that shift.

Each use case describes, in narrative form, an interaction that is complete, well defined, and meaningful to some users. The narrative of the use case is divided into two parts: the user action model, which shows the actions the user takes; and the system response model, which shows what the system does in response.

Depending on the level of abstraction at which the use case is described, there are two forms for use cases: essential and detailed or concrete. Essential use cases describe a generalized, abstract, technology-free and implementation-independent interaction, in the language of the application domain and of users. On the other hand, detailed use cases reflect the actual interaction as it happens between the user and the system, so they include restrictions imposed by internal design decisions and according to a particular user interface design. Detailed use cases can also be called concrete, since they reflect the concrete instantiation of the abstract description in its essential form. Detailed use cases are the ones usually employed in object-oriented software development, even if some recent software engineering reference books acknowledge the advantages of essential use case modeling.

Detailed use cases are useful as a technique for improving the usability of the software product, but not as the first approach to describing the interaction between the users and the system. Below we will describe essential use cases as an appropriate initial approach to interaction modeling, which will serve as basis for designing the best scheme to support interaction (using some of the other usability techniques described in this catalogue).

An example of a use case in both forms follows:

ESSENTIAL USE CASE

gettingCash

USER INTENTION	SYSTEM RESPONSIBILITY
identify self	
	verify identity
	offer choices
choose	
	dispense cash
take cash	

DETAILED USE CASE

gettingCash

USER INTENTION	SYSTEM RESPONSIBILITY
insert card	
	read magnetic strip
	request PIN
enter PIN	
	verify PIN
	display transaction option menu
press key	
	display account menu
press key	
	prompt for amount
enter amount	
	display amount
press key	
	return card
take card	
	dispense cash
take cash	

An essential use case is based on the purpose or intentions of a user rather than on the concrete steps or mechanisms by which that purpose or intention might be carried out.

The first step for the creation of the use case model is to identify the use cases that the system must support. The structured user role model is a starting point for use case identification. For each user role we can ask ourselves what these kinds of users are trying to accomplish, what they need to do in order to fulfill the role, or what capabilities are required to support whatever these users need to accomplish. Once the use cases have been identified, narratives are written for each use case, and the relationships between them are defined.

To write a use case narrative, we must identify the essential purpose or user intent embodied in the interaction. The name of the essential use case should be simple, it should imply purposeful, goal-directed action. Transitive gerunds, verbs of continuing action with a direct object, make good names for essential use cases. Examples of essential use case names are: findingCustomer, verifyingOrder or insertingMathSymbol. If the user purpose is not well expressed or fully implied by the name of the use case, then an explicit purpose clause should be added to the head of the narrative, describing and detailing the purpose or goal from the user perspective.

The use case diagram or use case map represents the use cases supported by the whole system, and the interrelationships among them and with the users. The use case map along with the narratives of use cases form the use case model.

Basic Reference: L. L. Constantine, L. A. D. Lockwood. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design*. Addison-Wesley, New York, NY, 1999. Chapter 5. pp. 97-123.

I.11 GOMS (Goals, Operations, Methods and Selection Rules)

Cognitive modeling aims to produce a computational model for how people perform tasks and solve problems, based on psychological principles. These models may be outlines of tasks written on paper or computer programs which enable us to predict the time it takes for people to perform tasks, the kinds of errors they make, the decisions they make, or what buttons and menu items they choose. Such models can be used to determine ways of improving the user interface so that a person's task has fewer errors or takes less time. They may be also used for deciding between two alternative designs, or between two systems for a purchase.

There are basically four different GOMS models: The Keystroke-Level Model (KLM), CMN-GOMS, NGOMSL, and CPM-GOMS.

The GOMS model is based on goals (edit document) and subgoals (change a word) that the user formulates; the operators available to users, like motor, perceptual or cognitive primitives (click the mouse, look at the menubar); the methods users compose out of sequences of these operators to achieve the goals or subgoals (selection is done by moving the cursor to point to the word and double-clicking the mouse); and the selection rules needed to decide what to do next if the user has several goals pending or if there are several methods that will accomplish a given goal (the word can be removed by selecting it an issuing a “cut” command or by backspacing over it).

GOMS can be used both quantitatively and qualitatively. Quantitatively, it gives good predictions of performance time and learning. Qualitatively, GOMS can be used to design training programs and help systems. The GOMS model is a careful description of the knowledge needed to perform a given task and thus it describes the content of task-oriented documentation. You only need to tell the new user what the goals are, what different methods could be used to achieve them, and when to use each method(selection rules). This approach has been shown to be an efficient way to organize help systems, tutorials, and training programs as well as user documentation.

Its main weakness is that GOMS assumes that the users are expert users (errorless performance is assumed). If the system is being developed for novice or intermediate users GOMS cannot be used.

In NGOMSL, learning time and execution time are predicted based on a program-like representation of the procedures that the user must learn and execute to perform tasks with the system. NGOMSL stands for Natural GOMS Language, because the notation used is a natural language structure to represent the user methods and selection rules.

NGOMSL starts after an initial task analysis has been performed, that is, after the user goals have been identified. The methods must be defined for each goal, by asking the question "how do you do it on this system?". Each method is described as a series of steps. If all the operators in a method are primitive, this is the final level of analysis. However, if some operators are high-level, they must be examined to decide whether a method of analysis is needed. Then we can calculate a time estimate for each goal.

The NGOMSL model below describes how to move an object in the Macintosh Finder tool:

```
Method for goal: move an object.
      Step 1. Accomplish goal: drag object to destination.
      Step 2. Return with goal accomplished.
```

There is a submethod for describing the dragging operation:

```
Method for goal: drag item to destination.
      Step 1. Locate icon for item on screen.
      Step 2. Move cursor to item icon location.
      Step 3. Hold mouse button down.
      Step 4. Locate destination icon on screen.
      Step 5. Move cursor to destination icon.
      Step 6. Verify that destination icon is reverse-video.
      Step 7. Release mouse button.
      Step 8. Return with goal accomplished.
```

Basic Reference: D. Kieras. "A guide to GOMS Model Usability Evaluation using NGOMSL", in *Handbook of Human-Computer Interaction. Second Edition*, edited by M. Helander, T. Landauer and P. Prabhu. North-Holland, 1997. Chapter 31. pp. 733-766.

I.12 Scenarios & Storyboards

A scenario details an interaction example illustrating the flow of specific user actions needed to get some result, concentrating on what the user will see, what the user must know, and what the user can do.

Scenarios are useful where there is no available data about the range and distribution of user task frequencies and sequences, especially for highly innovative systems. In these less well-defined projects, developers find day-in-the-life scenarios helpful to characterize what happens when users perform typical tasks. Scenarios can represent common or emergency situations, with both novice and expert users, and they are especially suited when multiple users must cooperate or multiple physical devices are used.

Storyboards (pictorial representations of scenarios, like the ones used by film directors) may provide additional support to the situations described in scenarios.

Scenarios may serve to convey a shared understanding of the product concept and the kind of users and tasks for which it is intended. It may be used as well to show to the customer what could be offered or provided if the system is actually developed.

An example of scenario for the Eurochange system (a system for currency exchange) follows:

> Path Smith has just arrived at Geneva International Airport en route to a large conference on Human-Computer Interaction. Pat is carrying a laptop and a large, heavy suitcase and needs to get to the conference center quickly. Looking around for a bank in order to get some local currency, Pat sees the Eurochange machine with its blue flag style logo showing a circle of twelve stars.
>
> Pat goes up the machine. It seems similar to the automatic teller machine that Pat uses regularly. Pat puts down the suitcase, takes out a credit card and inserts it into the slot. A message is displayed on the screen:
>
> *Enter your PIN*
>
> Pat thinks for a few moments and then types a four-digit number on the numerical pad, listening to the reassuring beep that follows each number pressed. The machine pauses for a few seconds and then displays:
>
> *Select currency required*
>
> Pat pauses again. What is the currency in Switzerland? Pat browses the currencies available, sees “Swiss Franc (CHF)” and presses the key. The machine displays the message:
>
> *Exchange rate is 1.47 CHF to 1 EUR*
>
> *Enter amount required in Swiss Francs in units of [10]*
>
> *Press <Proceed>*
>
> Pat types 253 and presses <Proceed>. A message is displayed:
>
> *Machine deals in bank notes only*
>
> *Smallest bank note is [10] CHF*
>
> *Enter new amount to obtain CHF or press <Cancel>*
>
> Pat enters 260 and presses <Proceed>. There is a whirring noise and a few other indeterminate clunks and clicks. The credit card is returned from the card entry slot and the money deposited in the delivery slot, with a printout of the transaction.

Storyboards (pictorial representations of scenarios, like the ones used by film directors) may provide additional support to the situations described in scenarios.

Basic Reference: J. M. Carroll. “Scenario-Based Design”, in *Handbook of Human-Computer Interaction. Second Edition,* edited by M. Helander, T. Landauer and P. Prabhu. North-Holland, 1997. Chapter 17. pp. 383-406.

I.13 Visual brainstorming

Brainstorming brings together a set of design and task experts to inspire each other in the creative, idea generation phase of the problem-solving process. It is used to generate new ideas by freeing the mind to accept any idea that is suggested, thus allowing freedom for creativity. The results of a brainstorming session are, it is hoped, a set of good ideas and a general feel for the solution area.

With visual brainstorming we try to generate visual ideas of abstract concepts. Visual ideas are a basis for discussion in a brainstorming session.

Paper prototypes may be used for exploring all kinds of design ideas, and they can help the development team to think about the organizing metaphor for a system. Visual brainstorming involves using exploratory paper prototypes as a means for facilitating communication in brainstorming sessions.

One of the first things you learn in design is to put forward a number of alternatives so that you can then compare them. Having a lot of display space is important for doing this because you can then make design ideas visual. One of the things you can do with visual things is superimpose them, or put them side by side and quite often when you start doing that you like one better than another. Until you have made a comparison, you have no idea why you prefer one to another. The criteria emerge from the comparison. It is not just picking the right idea, but recognizing the right idea in all the mess of different alternatives produced.

Evaluation also comes into brainstorming: when you stop generating ideas you have to start evaluating them. The best exploratory designs produced in the visual brainstorming can then be further developed by constructing more elaborated prototypes, which can be evaluated with users.

Basic Reference: J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, T. Carey. *Human-Computer Interaction*. Addison Wesley, 1994. pp. 456-461.

I.14 Paper Prototypes

Prototypes allow designers to communicate more effectively with users and they reduce the need and cost of reworking that can occur when products need to be revised later in the life cycle. We need to build prototypes because abstract technical specifications and models are not a good way of communicating when we want to involve users in the design process.

Prototyping, and especially rapid prototyping, is closely related to iterative design. For prototypes to be effective, they should be built at a minimal cost in terms of resources and time. The difference from traditional software engineering system prototypes is again a difference of focus. Prototypes are useful for usability purposes when they depict mostly system-user interaction, so that they convey how the system will work from the user point of view. So, prototypes can be used to try out design ideas with users and to gather their feedback.

From the product concept to full-detail design, prototypes of varying fidelity to the final system can be produced for use in usability evaluation activities. Therefore, the prototyping approach to interactive system development involves the production of at least one early version of the system that illustrates essential features of the later operational system. When used early in the development process, a prototype encourages user participation and involvement and allows developers to observe user behavior and reaction to the prototype. They may also serve for eliciting information from users: About the necessary functionality of the system, operation sequences, user support needs, required representations, and about the look and feel of the user interface.

HCI offers to software development a kind of cheap and quick prototypes, which are the less elaborate ones: Paper and chauffeured prototypes, and the Wizard of Oz technique.

Paper prototypes can be presented to users and members of the development team for comment and improvement. At the early stages of development, it may be undesirable to express design ideas by means of working software, since this may take a significant effort to build. Once there is a glossy version of an idea, it is too easy to get carried away thinking that this must be the only or the best solution to the problem.

Chauffeured prototyping involves the user watching while a member of the development team "drives" the system. It is a way to test whether the interface meets the user needs without the user actually having to carry out low-level actions with the system.

Basic Reference: C. Snyder. *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces*. Morgan-Kaufmann, 2003.

I.15 Wizard of Oz Technique

Wizard of Oz prototyping involves having some kind of behind-the-scenes manipulation to produce the responses of a working system, usually by means of a person providing the responses.

In a typical Wizard of Oz prototyping setting, the user interacts with a screen, but instead of a piece of software responding to the user requests, a developer is sitting at another computer (which is connected to the user computer through the network), answering the queries and responding to the real user. The user is unaware of the trick, so the perception of using a real working system is not spoiled. This kind of prototyping is widely used to prototype and test out user interface designs of many kinds, but especially for exotic or unusual configurations.

For example, the development team may want to try out a telephone-based interface that mixes limited voice recognition with telephone keypad responses. The behavior of the system is simulated by a person at the other end of the telephone line.

It is specially suited for systems including subsystems with some intelligence, like agents or advisors, or that include natural-language processing. That is, processing that can be easily done by a human, but requiring extensive programming efforts for it to be done by a computer.

Basic Reference: J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, T. Carey. *Human-Computer Interaction*. Addison Wesley, 1994. pp. 538-542.

II - Requirements Specification

II.1 Usability Specifications

Usability specifications are quantitative usability goals, which are used as a guide for ascertaining when a system has the proper usability level. They can be compared to non-functional requirements. Their concerns are user satisfaction and performance. Performance is interpreted as establishing the required performance of the new system formed by both the user and the software system, working together towards the achievement of certain goals. In this sense, usability specifications can also be called usability benchmarks.

The knowledge gathered in the specification of the context of use activity is the input for this activity. The usability specifications are defined according to the characteristics of the target user population and the goals and tasks identified in task analysis.

The set of usability specifications represent the system acceptance criteria from a usability point of view. Usability specifications are monitored at the end of each development cycle, establishing how much progress has been made towards the usability objective. They serve as criteria for determining when to stop iterating. While usability attributes are not directly measurable, usability specifications need to be.

Several authors prefer the term usability goal, as these specifications are established as a goal to be achieved in the system design. Usability goals drive design as information shared by the whole development team, which can provide decision criteria when different design alternatives are considered, not just as a test case that will be checked in the evaluation phase.

By establishing usability specifications early in the development process, and monitoring them at each iteration, you can determine whether your system is, indeed, moving towards an improved, more usable, result.

Usability Attribute	Measuring Instrument	Value to be measured	Current Level	Worst Acceptable Level	Planned Target Level	Best Possible Level	Observed Results
Performance in normal use	“Answer Request” task	Length of time taken to successfully perform the task (minutes and seconds)	2’ 53’’	2’ 53’’	1’ 30’’	50’’	
Performance in normal use	“Answer Request” task	Number of errors during task performance	0	0	0	0	
First Impression	Questionnaire	Average score (range –2 to 2)	-	0	1	2	

The first step is to identify the usability attributes or sub-attributes that we want to cater for. Depending on the kind of product developed, some attributes might be irrelevant, for example, efficiency could be a secondary goal for a walk-and-use kiosk, while learnability would have top priority.

Attributes like satisfaction or first impression may be evaluated by means of subjective measures, normally in the form of questionnaires.

For performance-related attributes, a set of benchmark tasks (use cases) must be selected and associated with each attribute. A benchmark task is a typical, representative use case a user will perform. Measuring a user's performance on a benchmark task provides an objective usability metric for the related usability attribute. Benchmark tasks should be as specific as possible, so that there is little variability in their enactment by different users. The benchmark task in the first row could be described as follows: "Suppose you are at the Help Desk counter, and you receive a request. You decide to answer the request, and you look for the answer in your knowledge base ...".

The value to be measured must be then decided. For a benchmark the main values we can collect are the time to complete a task, or the number of errors during task performance. It is usually sensible to measure both of them, and we would have two rows in the Usability Specification table that are based on the same task.

The last step in defining usability specifications involves establishing the range of levels:

- The current level may refer to the value for the usability attribute in question with the current version of the system or with a competitor we want to challenge with our more usable product. When we are automating a manual procedure, it may refer to the time required to manually perform the task. If the system is very innovative, this field could be left blank.
- The worst acceptable level is the lowest acceptable level of user performance. It means that if the system does not reach this minimum level for any of the attributes in the specifications, the system is unacceptable from a usability point of view. The value for the current version of the product is usually taken as a reference to establish this level, and the level will be higher if the current version is unsatisfactory.
- The best possible level is a realistic upper limit. It should be an attainable level, not a wild dream. A hypothetical expert user should be able to attain this level. You can use developers as users to establish the best possible level, since they are the ones who are better acquainted with the subtleties of the interaction design. Another possibility is to use GOMS to provide theoretical estimates of expert error-free task performance.
- The planned target level is the attainment of unquestioned usability success. It is the most important value, because it is the actual requirement equivalent to traditional requirements. The other values must be filled in beforehand to help to set the planned target level in a sensible range of values. If there is a competitive system with a high usability level, it may serve as a reference for setting the planned target level.

When a system prototype exists, usability testing may be used to establish these levels at reasonable values.

Much expertise in the issue is required to establish good levels for the usability specification table, and this is why a usability specialist might be needed to apply this technique the first times it is used.

The observed results column will be filled in when the specifications are tested by means of usability testing and questionnaires.

Basic Reference: D. Hix, H.R. Hartson. *Developing User Interfaces: Ensuring Usability Through Product and Process*. John Wiley and Sons, 1993. Chapter 8. pp. 221-248.

III - Requirements Validation

As usability practice involves a combination of user participation and iterative development, it includes techniques for validating preliminary models. Any kind of expert review could be employed to validate early prototypes, or models of user-system interaction.

Expert review techniques (Cognitive Walkthrough, Heuristic Evaluation, Pluralistic Walkthrough and Usability Inspections) are detailed in part V: Usability Evaluation.

IV - Design

Interaction design tackles the definition of the interaction environments and their behavior, including the design of the visual elements that form the graphical UI (User Interface).
Interaction design should be made to accommodate the definition of the interaction between the system and the environment that has been produced as part of requirements analysis activities. And the internal structure of the system should be designed to provide a good implementation of this interaction with the environment.
Before detailing each design technique some interaction design guidelines are presented.

IV.1 Interaction Design Guidelines

Each HCI expert has his or her own set of guidelines. In the table below are detailed Jakob Nielsen's heuristics [Nielsen, 93].

Nielsen's Usability Heuristics
• Use simple and natural dialogue
• Speak the users' language
• Minimize user memory load
• Be consistent
• Provide feedback
• Provide clearly marked exits
• Provide shortcuts
• Provide good error messages
• Prevent errors

Another well respected HCI expert, Ben Shneiderman, presents three main principles [Shneiderman, 98], of which one of them relates to eight rules for interface design, as follows.

- Principle 1: **Recognize diversity**. Before beginning a design we must make the characterization of the users and the situation as precise and complete as possible.
- Principle 2: Use the Eight Golden Rules of Interface Design.
    - **Strive for consistency**. Consistent sequence of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent colour, layout, capitalization, fonts and so should be employed throughout.
    - **Enable frequent users to use shortcuts**. Abbreviations, special keys, hidden commands and macro facilities

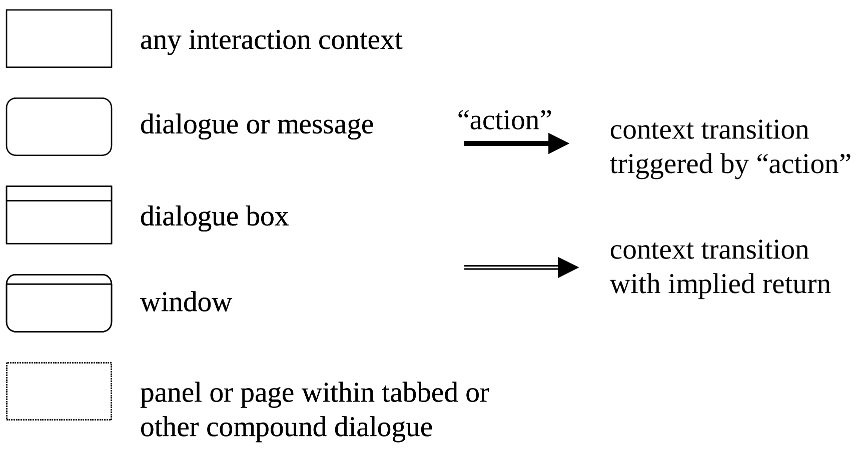
    - **Offer informative feedback**. For every user action, there should be system feedback.
    - **Design dialogs to yield closure**. Sequences of actions should be organized into groups with a beginning, middle, and end. The aim is to provide the user with a sense of accomplishment.
    - **Offer error prevention and simple error handling**. As much as possible, design the system such that users cannot make a serious error. Erroneous actions should leave the system unchanged, or the system should give instructions about restoring the state.
    - **Permit easy reversal of actions**. The units of reversibility may be a single action, a data-entry mask, or a complete group of actions.
    - **Support internal locus of control**. Users should feel they are in control of the system.
    - **Reduce short-term memory load**. Displays should be kept simple, multiple page displays be consolidated, window-motion frequency be reduced, and sufficient training time be allotted for codes, mnemonics, and sequence of actions. Where appropriate, online access to command-syntax forms, abbreviations, codes, and other information should be provided.
- Principle 3: **Prevent Errors**.
    - **Correct matching pairs**. This rule is directed to avoid syntactical errors when writing in a programming language or similar. It can be also applicable to word processors.
    - **Complete sequences**. Group different steps which are often used jointly . Example: The sequence of dialling up, setting communication parameters, logging on, and loading files. Another example: In am word processor, setting the alignment, the font, the font size an the letters in uppercase for section titles; it should not be necessary to set those characteristics every time a section title is entered.
    - **Correct commands**. Automatic command completion for command languages, and limiting the possible choices to permissible commands in a more developed interface.

IV.2 Context Navigation Maps

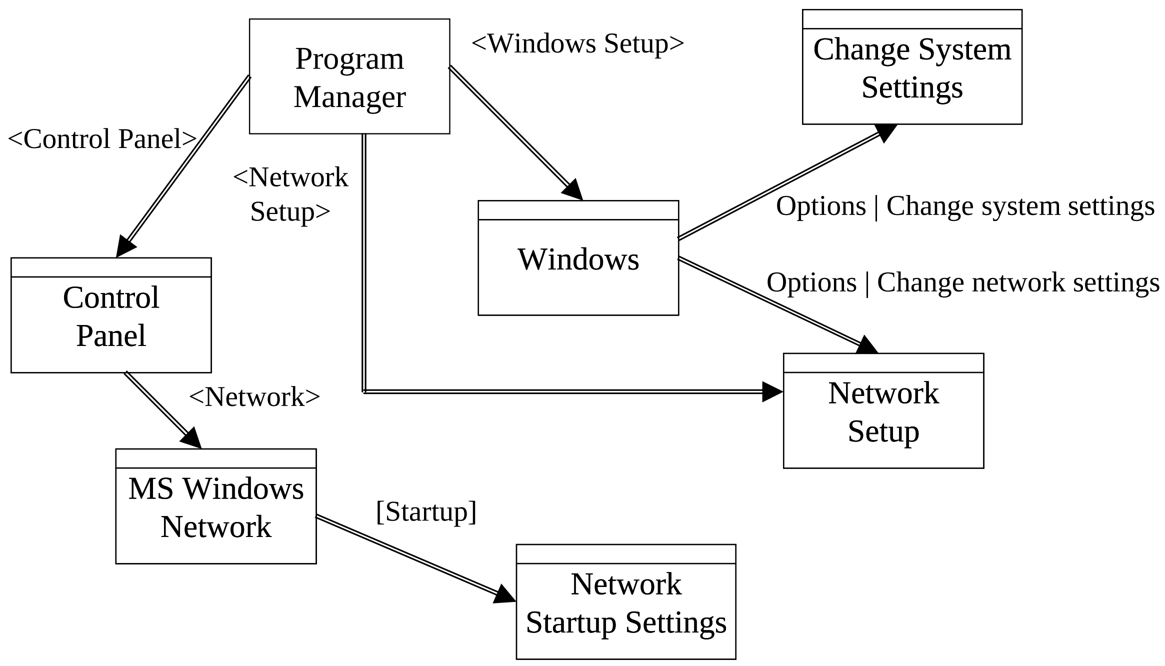
A context navigation map models the interconnections among the various interaction spaces of the user interface. It gives more dynamic expressiveness than tree menus, by specifying the transition between the different interaction contexts that occurs when a use case is enacted. A context navigation map (or navigation map for short) represents the structure of the user interface by modeling the relationships among interaction contexts.

The navigation map models the way users can navigate through the various interaction contexts within the user interface in the course of enacting use cases. When a single use case is represented, the navigation map models the behavioral view, and this is the most usual application of navigation maps. When the map combines all the behavioral views for the various use cases of the system in a single diagram, the result is called an architectural view. For big systems with a lot of interaction spaces the architectural view may get unwieldy, and too many transitions can lead to spaghetti-like diagrams.

Next figure details the notation for navigation maps.


any interaction context
dialogue or message
dialogue box
window
panel or page within tabbed or other compound dialogue
"action"
context transition triggered by "action"
context transition with implied return

The navigation map models the way users can navigate through the various interaction contexts within the user interface in the course of enacting use cases. When a single use case is represented, the navigation map models the behavioural view, and this is the most usual application of navigation maps. An example of a behavioural view is shown in the next figure:



When the map combines all the behavioural views for the various use cases of the system in a single diagram, the result is called an architectural view. For big systems with a lot of interaction spaces the architectural view may get unwieldy, and too many transitions can lead to spaghetti-like diagrams.

Basic Reference: L. L. Constantine, L. A. D. Lockwood. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design*. Addison-Wesley, New York, NY, 1999. pp. 135-145.

IV.3 Impact Analysis

Impact or cost/importance analysis is a technique for deciding between design options, by relating the options to the usability problems that are affected and choosing the ones that address the most important usability problems. It is performed once we have a set of usability problems identified in any kind of usability evaluation activity. For each usability problem, we can propose a design decision, and use impact analysis to prioritize these decisions in order to undertake the redesign effort.

It is a tool for making the trade-offs necessary in any design process, which are based, in this case, on the usability issues that are addressed.

In an impact analysis, the development team considers the relative importance of the usability problems found, and the cost of the solutions as listed in a table like the following one:

Usability Problem	**Effect on User Performance**	**Importance**	**Solution(s)**	**Cost**	**Resolution**
Too much window manipulation	10 of 35 minutes	High	Fix window placement automatically, but allow user to reposition it	6 hours	
Black arrow on black background	N/A	Low	Reverse arrow to white on black	1 hour	

Actually, impact analysis begins once all columns except the Resolution column have been completed for all observed problems. Depending on the number of design decisions to be considered, tools may be used for decision making, such as a graphical representation of problem distribution on the importance vs. cost scale. In principle, highly important problems should be tackled first, but the development team must also consider the resources allocated for the design activity and act accordingly. The process of deciding between design improvements that need to be made is not easy, and all this technique does is provide information in a structured manner so that the development team can make an informed decision.

Basic Reference: D. Hix, H.R. Hartson. *Developing User Interfaces: Ensuring Usability Through Product and Process*. John Wiley and Sons, 1993. pp. 316-330.

IV.4 Menu-selection and Dialog Box Trees

In a menu-based system, menu trees represent the structure of menu navigation. Menu trees are powerful as a specification tool since they show users and other stakeholders the complete and detailed coverage of the system. Like any map, a menu tree shows high-level relationships and low-level details.

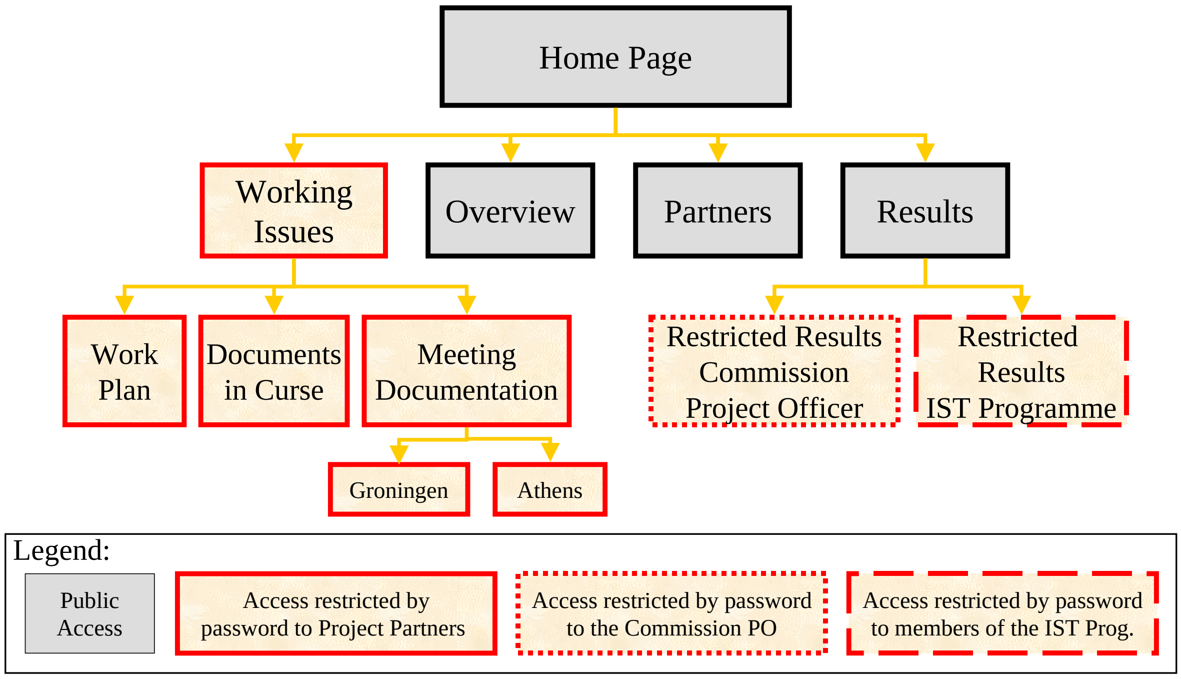
Similar comments apply to dialog-boxes. Printing out the dialog boxes and showing their relationships by mounting them on a wall is very helpful for gaining an overview of the entire system to check for consistency and completeness.

When we cannot create an interaction scheme based on direct manipulation (as in the desktop metaphor in the Mac and Windows operating systems), we can use menu selection. If the menu items are written using familiar terminology and are organized in a convenient structure and sequence, users can select an item easily.

When a collection of items grows and becomes difficult to maintain under intellectual control, designers can form categories of similar items, creating a tree structure. Menu trees represent this structure. With large systems, the menu tree may have to be laid out on a large wall or floor, but it is important to be able to see the whole structure in one go to check for consistency, completeness, and lack of ambiguity or redundancy.

It is difficult to group menu items in a tree so that they are comprehensible to users and match the task structure. Problems include overlapping categories, extraneous items, conflicting classifications in the same menu, unfamiliar jargon, and generic terms. The members of the development team may discuss all these issues while they all share a view of the complete tree structure represented in a menu-selection tree.

Websites that are organized in a highly hierarchical structure can be easily represented by means of tree menus. The following figure shows the menu tree of the STATUS project website:


Home Page
Working Issues
Overview
Partners
Results
Work Plan
Documents in Curse
Meeting Documentation
Restricted Results Commission Project Officer
Restricted Results IST Programme
Groningen
Athens
Legend:
Public Access
Access restricted by password to Project Partners
Access restricted by password to the Commission PO
Access restricted by password to members of the IST Prog.

Basic Reference: B. Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 1998. pp. 247-252.

IV.5 Organizing Help by Use Cases

The design of the help subsystem is important for the overall usability of the software product. Good help will not turn an unusable product into a highly usable one, but well-written, well-organized and accessible help can compensate to some extent for limitations in software.

Essential use cases express what the users may want to accomplish with the system, so they are an excellent basis for organizing help contents. This kind of structure is useful for providing procedural help, that is, help with how to perform a task.

Additionally, the design of the help subsystem may be undertaken for a set of general help use cases (or help cases), which give a response to common help requests made by users of all kind of systems: *seekingIdentification*, *seekingInstruction*, *seekingClarification*, *seekingElaboration*, *seekingReminder*, *seekingLocation*, and *exploringFeatures*.

Procedural help is most helpful when it is organized by use cases that are titled and written in the ordinary language of the users and the application domain and are well indexed. Use cases are a natural way of organizing and providing access to help because they represent the basic intents of users. Each essential use case is a complete and well-defined task based on something a user might try to accomplish. If the essential use case model has been well constructed, it will reflect how users think about and conduct their work. Each use case then becomes an entry in the help file.

To design support for other kinds of help, we can focus on the common help requests described as help cases (use cases for help seeking). Note that procedural help is expressed by means of the help case *seekingInstruction*. The following table details common user questions and the corresponding help case:

User Question	**Help case**	
What is this?	seekingIdentification	
	indicate object	brief description
How do I...?	seekingInstruction	
	identify task	operational sequence
What should I do?	seekingSuggestion	
	request	hint
What do you mean?	seekingClarification	
	request	different explanation
Tell me more	seekingElaboration	
	request	details or advanced
Remind me about...	seekingReminder	
	identify feature/task	brief synopsis
Where is...?	seekingLocation	
	identify feature	give place, routing
What can I do?	exploringFeatures	
	request	overview, topic map

Basic Reference: L. L. Constantine, L. A. D. Lockwood. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design*. Addison-Wesley, New York, NY, 1999. Chapter 11. pp. 231- 264.

IV.6 Product Style Guide

Consistency is a very important asset of any UI from a usability point of view. The Product Style Guide document is a way to assure that all UI elements are built according to a consistent pattern. The style guide specifies the standard for the creation of UI elements that will be followed in any screen of the product. It contains the basic templates, controls, and rules of design for a product, or family of products.

The following table shows the main benefits provided by a Product Style Guide to different stakeholders:

End Users	Developers	Business Team
● Reduced errors ● Less frustration ● Increased morale ● Improved efficiency ● Increased confidence ● Reduced resistance to new technology	● Maintain control over look and feel ● Minimize re-invention ● Capitalize on learning ● Enable production of reusable software ● Reduce development time ● Reduce arbitrary design decisions	● Produce usable systems that reduce support costs and increase user satisfaction ● Increase market awareness ● Increase product awareness ● Reduce training costs ● Improve staff retention ● Increase user acceptance of new systems

Typical elements of a style guide include:

- General presentation rules for products/processes
- Characteristics of standard screen types (browser, form, search,...)
- Rules for consistent usage of colors, dimming, bold, reverse video, fonts, sound, etc.
- Rules for controls and organization and layout of dialog box fields. Details may be included about the ordering of elements, group boxes, alignment, and use of white space.
- Keyboard shortcut schema
- Rules for message boxes (errors, warnings)
- Rules for the display of status information

The writing of a style guide is not an easy task, as it requires some expertise in this kind of issues. Nevertheless, it usually pays off when undertaken along with other user-centered activities, since the products of usability techniques give the basis for the decisions to take in the production of a style guide.

Start with high-level architectural guidelines and standards that will have the most impact (templates for Web pages or major features like search). Have small groups with the relevant expertise write draft sections and have the entire group review them and provide feedback. After defining the high-level pages or screens, work on lower level issues and general issues like color, use of controls, and text guidelines. Base these more detailed issues on the high-level components so the design is internally consistent.

Decide what processes will be implemented to publicize, distribute, update, and enforce the guidelines and standards in the style guide. Convince the product team to implement consistency inspections and make consistency part of everyone’s objectives, bonuses, and job descriptions. Conduct style guide training with managers, developers, and other key groups.

For a style guide to be useful it must be created in a way that ensures its usability. Prepare drafts and test them with UI implementers, refining it until it is fit for purpose. It is a waste of time and money producing a style guide that remains on the shelf and is never used.

Also, for the success of a style guide, it is very important to establish a process of consistency inspections that ensure that the style guide specifications are actually followed in the resulting software product, or family of products.

It is sometimes tempting to think that when having a style guide all usability problems go away. A UI style guide is a foundation for design but does not guarantee the success of a design. A style guide is only one link in the chain of user-centered design activities that are important for product success.

Requirement analysis, user profiles, task analysis, usability evaluation, and iterative design must accompany a style guide.

Basic Reference: D. J. Mayhew. *The Usability Engineering Lifecycle*. Morgan Kaufmann, 1999. Chapter 14. pp. 311-324.

V - Evaluation

Usability is a very complex concept, due to the complex nature of humans. Without doing some form of evaluation, it is impossible to know whether or not the design or system fulfils the needs of the users and how well it fits the physical, social and organizational context in which it will be used. No matter how much we stress the performance of user-centered activities in the development process, we will not be able to exactly predict the usability level of the system in advance. For this reason, we need to perform usability evaluation at the end of every iterative design cycle to find out where the product is in usability terms and how much improvement is needed in order to reach the previously specified usability goals.

V.1 Cognitive Walkthrough

In a cognitive walkthrough, correct sequences of actions are analyzed, asking if they will actually be followed by users. The cognitive walkthrough analyst identifies problems by tracing the likely mental processes of a hypothetical user. The analysis considers matters like user background knowledge that influence mental processes but are not part of the user interface. The technique aims to identify likely usability problems in the user interface and to suggest reasons for these problems.

Cognitive walkthroughs were developed for systems that can be learned by exploratory browsing, but they are useful even for systems that require substantial learning.

The cognitive walkthrough analyst must begin by defining the assumed user background. The user structure role model should provide this information. Then the analyst must choose a representative task and devise a realistic usage of this task. If usage scenarios have been created, they can be a good source for realistic usage of tasks.

The analyst determines one or more correct sequences of actions for the chosen task. A correct sequence of actions is one that developers would be happy to see users use. Often there will be more than one acceptable way of performing a task. It these variations are important a cognitive walkthrough can be done on more than one, but often it will be sensible to choose the most common, or perhaps the most problematic.

The final step in the preparation of the cognitive walkthrough is to work out as fully as possible what the user will see at each step of the sequence or sequences to be examined. This may sometimes force the developers to create a partial design that is detailed enough to indicate the key interface features along the path. Screen sketches and /or dialogue flow (use cases) are usually enough to perform a cognitive walkthrough.

In the walkthrough itself, the analyst works through the sequence of correct actions, considering the state of the interface before and after each action, trying to determine how likely it is that users will follow that path. The kind of questions the analyst must try to answer are:

- Will the user be trying to achieve the right effect?
- Will the user notice that the correct action is available?
- Will the user associate the correct action with the desired effect?
- If the correct action is performed, will the user see that progress is being made?

For each correct action, the analyst must construct a success or failure story. If all the answers to all the questions are “yes”, including an explanation, then it is a success story. If the answer to one or more of the questions is “no” or “not always”, the analyst has a failure story. The explanation of this answer will tell the development team why the analyst expects that some users will have trouble at this point.

Basic Reference: C. Lewis, C. Wharton. “Cognitive Walkthroughs”, in *Handbook of Human-Computer Interaction. Second Edition*, edited by M. Helander, T. Landauer and P. Prabhu. North-Holland, 1997. Chapter 30. pp. 717-732.

V.2 Heuristic Evaluation

Heuristic evaluation is performed to identify the usability problems of a system, so that they can be attended to as part of an iterative design process. It involves having a small set of evaluators examine the interaction design and judge its compliance with recognized usability principles (the heuristics).

It can be used as a complement to usability testing with users, since it usually reveals different kinds of usability problems than usability testing.

Each evaluator inspects the interaction design alone. After the evaluations have been completed, the evaluators may gather to report their findings. This procedure is important in order to ensure independent and unbiased evaluations from each evaluator.

During the evaluation session, the evaluator goes through the interaction scheme (screen sketches and/or use case description) several times and inspects the various dialogue elements and compares them with the list of recognized usability principles. These heuristics are general rules that seem to describe common properties of usable interfaces, like the ones described in section IV.1. In addition to the checklist of general heuristics to be considered for all dialogue elements, the evaluator is also allowed to consider any additional usability principles or results that come to mind that may be relevant for any specific dialogue element.

The number of evaluators to be employed depends on the criticality of system usability, but it is clearly better to combine evaluations by several evaluators than have a single evaluator. Experts recommend using about five evaluators and certainly at least three.

Unlike other evaluation methods, such as walkthroughs, the evaluators decide on their own how they want to proceed with evaluating the interface, instead of following the predefined paths given by use cases.

The output from heuristic evaluation is a list of usability problems in the interaction design, annotated with references to the usability principles that were, in the opinion of the evaluator, violated by the design in each case.

Basic Reference: J. Nielsen. *Usability Engineering*. AP Professional, 1993. pp. 155-162.

V.3 Pluralistic Walkthrough

A pluralistic usability walkthrough is a collaborative process involving users, developers and other stakeholders, where all participants are expected to play the role of users. The participants evaluate the interaction design by trying to perform a given task, and they stop at each step to have a group discussion about its usability. The goal of the technique is coordinated empathies to help developers to put themselves in the shoes of users.

A pluralistic walkthrough is driven by a task scenario chosen in advance and for which a storyboard, a series of screen sketches or paper prototypes representing the various contexts or a working prototype have been prepared. For each step in the task, all participants independently decide on what action or actions they would take next and note these on their own copies of the storyboard. No discussion takes place until all participants have completed a given step. When the discussion of the step begins end users speak first to prevent the developers dominating the discussion.

This technique is relatively slow, since all participants have to be at the same step at the same time. But this technique has some appealing advantages, cited by participants in this kind of evaluations, such as:

- They feel that their viewpoints have been heard,
- their expertise was valued, and
- their design concerns were remedied to satisfaction.

Basic Reference: R. G. Bias. "The Pluralistic Walk-Through: Coordinated Empathies", in *Usability Inspection Methods*, edited by J. Nielsen and R. L. Mack. Wiley, 1994.

V.4 Usability Inspections

Inspections have a long history in software development. The goal of all inspections is to find defects. Usability inspections are aimed at identifying usability defects. The object of inspection may be a finished product, a design or a prototype. Usability inspections refer to systematic processes for inspection, as opposed to heuristic evaluation, which is a less formal usability assessment technique.

When different stakeholders perform the inspection in a collaborative effort, it is called collaborative usability inspection. In this case, the review process is a team effort that includes software developers, end users, application or domain experts and usability specialists, collaborating to perform a thorough and efficient inspection.

There are two variants of inspection, which have a specific focus: consistency inspections and conformance inspections.

In consistency inspections, the goal is to identify inconsistencies across interaction contexts and their contents. The evaluators check for consistency of terminology, color, layout, input and output formats, and so on. When the product belongs to a family of products, teams of designers, at least one from each project, meet to inspect the usability of the different products of the family.

In conformance inspections, the participants inspect the system interaction for compliance with specified standards or with style guidelines. All participants must be familiar with the applicable standards and/or style guidelines.

Collaborative usability inspections, if well conducted, can be more productive than expert inspections. The focus needs to be kept on the user perspective, in order to identify the usability problems that might arise. Developers need to adopt the mindset of an impatient and intolerant user. The presence of actual users in inspections helps to catalyze taking the user perspective. Two special roles in the inspection team are the lead reviewer, who organizes the inspection meetings and moderates the process; and the inspection recorder, who maintains a complete log of identified defects. Another special role in the team may be the continuity reviewer, who has special responsibility for identifying inconsistencies. Apart from members of the development team, it may be useful to have some developers who have not participated in the development effort, because they bring a fresh perspective into the inspection. Members of the

development team are not allowed to defend, explain, excuse or rationalize any aspect of their design or the decisions leading to it. Developers should also avoid making implied promises to the users. The comments and inputs from users should be given special weight in the inspection process, without allowing these to dictate interaction design decisions. The lead reviewer should encourage user participation and protect users from criticism or antagonistic questioning. Users and domain experts should be regarded as authorities, but not as arbiters. Finally, usability experts may also contribute to collaborative usability inspections.
Basic Reference: J. Nielsen, R.L. Mack. *Usability Inspection Methods*. Wiley, 1994.

V.5 Thinking Aloud Protocol

Thinking aloud is a technique for performing usability tests with users. The evaluator asks participants (users) to talk out loud while working during a usability testing session, indicating what they are trying to do, or why they are having a problem, what they expected to happen that did not, what they wished had happened, and so on. By verbalizing their thoughts, test participants enable the developer to understand how they view the system, and this helps to identify the major user misconceptions.
The strength of thinking out loud is on qualitative data and not on performance measures. The idea is to get the user’s impression while using the system to avoid later rationalizations. The aim of this kind of testing is to detect the parts of the dialogue that are more problematic from a usability point of view, along with the real causes of the problems.
There are some variants of this technique: constructive interaction, retrospective testing, critical incident taking and coaching method.
Thinking aloud is can be employed in any usability test. There is no difference in the test preparation with performance measurement usability testing. But, before starting with the test, the evaluator must encourage the test participant to think out loud, maintaining a running monologue about what he or she is doing as it is being done.
The evaluator may find that some participants are not good at thinking aloud while they work. They will not talk much, and the evaluator will have to prod them constantly to find out what they are thinking or trying to do. This has an impact on performance measures, so we do not advise combining thinking aloud with performance measurement.
User comments are sometimes indicators of personal user likes or dislikes, so developers should take care not to change part of the system just because of a comment by a single user. It is the responsibility of the evaluator to interpret the user comments and not just accept them indiscriminately. For example, users using a mouse for the first time will often direct a large proportion of their comments toward aspects of moving the mouse and pointing and clicking. In this case, the evaluator should try to abstract from the mouse problems in the dialogue and focus on other system issues.
The following techniques are variants of the basic think-aloud protocol:

- **Constructive interaction**: It involves having two test users use a system together. It is also called codiscovery learning. It aims to overcome the problem of shy test participants, who do not verbalize easily. This is based on the fact that people are used to verbalizing when they are trying to solve a problem in a collaborative effort.

- **Retrospective testing**: The usability testing session is recorded on a videotape and the participant is requested to review the recording. Participant comments while reviewing the tape are sometimes more extensive than comments while performing the task in the test. The evaluator can stop the tape and ask the participant questions at any time, without fear of interfering with the test, which has essentially been completed already. This variant may be useful when the usability testing involves some kind of performance measurement that could be distorted by the dialogue with the reviewer.
- **Critical incident taking**: This variant implies recording both negative incidents (signs of frustration, either with remarks or actions), and positive incidents (satisfaction or closure expressions). Negative incidents help to identify the more important usability problems, while positive incidents help to identify metaphors or details to be used more thoroughly in the user interface because of their success
- **Coaching method**: The evaluator (or "coach") steers the participant in the right direction while using the system. The participant can ask the evaluator questions, and the questions may show up usability problems that would remain uncovered otherwise. The evaluator will answer to the best of his or her ability.

Basic Reference: J. Nielsen. *Usability Engineering*. AP Professional, 1993. pp. 195-200.

V.6 Performance Measurement / Laboratory Usability Testing

Performance measurement through usability testing is used for assessing whether usability goals set in usability specifications have been met. It can be used as well for comparisons with competing products.

Performance is measured by having a group of users perform a predefined set of test tasks while collecting time and error data. When the test is performed in a special room prepared for usability testing, it is called laboratory usability testing. A laboratory is usually composed of two rooms separated by a one-way mirror: the evaluation room where participants carry out the tests and the main evaluator gives instructions; and the control room, where additional evaluators and other members of the development team can observe the test, without disturbing the test participant. Usual equipment for a usability laboratory includes a video camera to record the screen, another one for recording the participant, tools for software logging and monitors to show in the control room what is happening in the evaluation room.

The opposite to laboratory testing is field testing, where the system is taken to the user environment instead of taking the participant to the system, and the usability test is performed in the user organization.

Before performing any test, the first step is to develop the experiment. The participants must be selected, trying to get a representative sample of the total user population. Information in the structured user role model should serve to select an adequate distribution of test participants. The tasks to be employed in the tests must be defined as well, the benchmark tasks that appear in the usability specifications must be tested, but additional representative tasks may be tested as well, not to get performance measurements but to identify usability problems. The evaluator should write down the tasks in the order that the participant will be asked to perform them. This list of tasks may be either given to the participant or read out loud by the evaluator one task at a time. Finally, the evaluator must define the protocol and procedures for the test. This includes the preparation of introductory instructional remarks for participants, which should briefly explain the purpose of the experiment, the system to be tested and what the participant will be expected to do. It is important to make clear to all participants that the purpose of the test session is to evaluate the system, not the participant. An informed consent form should also be prepared for participants to sign, stating that the participant is volunteering for the experiment, that the data may be used if the participant's name or identity is not associated with the data, that the participant understands that the experiment is in no way harmful and that the participant may discontinue the experiment at any time.

It is advisable to perform a few pilot tests with three or four participants to ensure that all parts of the experiment are ready. Pilot testing may show up inadequate wording of the tasks that the participants are being asked to perform, or that some part of the procedure needs to be changed. After pilot testing the test plan is refined in order to proceed with the greater part of the testing effort using an improved test plan.

For the test session, the evaluator will usually be sitting beside the participant, especially when qualitative data needs to be collected. The test participant is asked to perform the tasks defined for the test, and both the number of errors and the performance time are measured for each task. It may be necessary to prompt the participant during the session, primarily during qualitative data collection, to get the desired information. The think-aloud technique and its variants may be applied for this purpose in any usability test.

The data collected during the test sessions must then be analyzed. Quantitative data will be formed by performance times, error rates, and also by the user preference expressed in questionnaires. Qualitative data will come from the user comments that the evaluator has taken down or extracted from an audio or video recording of the session. The information gathered in usability tests can tell the development team whether or not the development is going in the right direction (that is, whether we are coming closer to the goals in the usability specifications or not), and it can point out the issues in the interaction dialogue that are a source of usability problems. After an impact analysis, decisions are taken about which usability problems will be tackled first in the next cycle redesign effort.

Basic Reference: J.S. Dumas, J.C. Redish. *A Practical Guide to Usability Testing. Revised Edition*. Intellect, 1999.

V.7 Questionnaires and Surveys

Questionnaires are used to determine a user's subjective satisfaction with the system. Measuring user satisfaction provides a subjective (but, nevertheless, quantitative) usability metric for the related usability attribute. Some usability specifications will be related to user satisfaction, and questionnaires are the way to check whether the level specified for this attribute has been reached. Questionnaires are usually administered to usability test participants after the test has taken place, so they can give their opinion about specific parts of the user interface and about the overall system.

When questionnaires are distributed to a lot of users, they are called surveys. While questionnaires issued to usability test participants may contain questions about specific parts that have been used in the test, surveys usually gather opinions on more generic issues. Additional information that is usually collected has to do with individual user characteristics, such as background (age, gender, education), experience with computers, familiarity with specific features (virtual reality, macros, shortcuts), and so on.

It is advisable to do a pilot study before sending questionnaires to a large number of users in order to ensure that it is well designed. Care must be taken to make the questions unambiguous, and the questionnaire in general should be as simple as possible to increase the chance of respondents completing and returning the questionnaire.

Questions may be open, where the respondent is free to provide his or her own answer, or closed, where the respondent is asked to select an answer from a choice or alternative replies. Closed questions usually have some form of associated rating scale. The most commonly used scale for HCI studies is the semantic differential scale. This scale is based on bipolar adjectives (such as easy-difficult, clear-confusing) at the end points of the scale and respondents rate on a scale between these paired adjectives. This is the scale used in the questions in the table below.

Once the questionnaires have been given to the selected population, the responses obtained on the different rating scales are converted into numerical values and statistical analysis is performed. The main statistics used in the analysis of surveys data are means and standard deviations.

The table below shows sample questions belonging to a questionnaire to be administered to test participants after a usability testing session. The tool to be tested provided facilities for managing problem resolution tasks in a Help Desk. A differential semantic scale with five choices was used for each question, centering the scale around zero. As a mid-scale reading, zero is an appropriately neutral value. Negative scale readings correspond to negative user opinions and positive readings to positive opinions. The final category of questions is focused on overall user reactions.

	unsatisfactory				satisfactory
General satisfaction	-2	-1	0	1	2
	not suitable				suitable
Suitability for problem solving tasks	-2	-1	0	1	2
	worst				better
General comparison with existing system	-2	-1	0	1	2
	too little				enough
Feedback provided to user actions	-2	-1	0	1	2
Overall opinion about the system:	terrible				wonderful
	-2	-1	0	1	2
	frustrating				satisfying
	-2	-1	0	1	2
	dull				stimulating
	-2	-1	0	1	2
	difficult				easy
	-2	-1	0	1	2
	rigid				flexible
	-2	-1	0	1	2

Beta-testing is a survey-based form of evaluation. In beta-testing, a working but not completely finished version is supplied to a big pool of customers who are willing to test the product using it to perform their work (or to fulfill their goals). In addition to questions on possible system failures, beta-testers may be asked to answer preference questions after their usage of the system.
Basic Reference: D. J. Mayhew. *The Usability Engineering Lifecycle*. Morgan Kaufmann, 1999. Chapter 17. pp. 353-399.

V.8 Interviews

Interviews involve having an interviewer read questions to a respondent and writing down the responses. For the creation of the questionnaire, refer to section V.7 Questionnaires and Surveys.
After usability testing the evaluator may interview the participant to get the user's subjective opinion, instead of letting the participant fill in a written questionnaire. Interviews are more flexible, since the evaluator may ask follow-up questions that not were in the script.
Interviews need to be planned for them to yield useful results, much in the same way questionnaires must be carefully planned before being administered to users.
There are two main kinds of interviews: structured, where the questions are predetermined and flexible interviews, where the interviewer is free to follow the interviewee's replies and to find out personal attitudes. Flexible interviews are less formal, and they are adequate for requirements elicitation and for gauging users' opinions about a particular idea. No matter how flexible the interview is going to be, a rough plan of the topics to be discussed is still needed.
The interviewer should make the interviewee feel comfortable, establishing interviewer-interviewee rapport. For example, some people feel embarrassed when they criticize a system, particularly when they have to describe their own difficulties in using it.
When the interviewer has a set of questions prepared in case the interviewee digresses or does not say much, it is called a semi-structured interview. A variant for drawing out more information from the interviewee is prompted interviewing, where the interviewer stimulates the interviewee by saying things like "... and can you tell me a bit more about that" or "...and what do you mean by...". Alternatively, prompting may take the form of showing the interviewee an alternative item such as a screen design, in order to promote further discussion or generate new ideas for discussion.
The trade-off to be considered in structured vs. flexible interviewing is that the less structured the interview is, the more scope there is for picking up relevant issues but the harder it is for the interviewer.
Flexible interviews on usability issues have been predominantly used to determine the user's understanding of the interaction scheme. An issue to consider is that the interviewer should avoid asking leading questions that beg a particular response.
As for questionnaires and surveys, when preparing an interview with domain experts (who are usually a scarce resource), it is better to do a small pilot study to be able to refine the interview script.
Basic Reference: J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, T. Carey. *Human-Computer Interaction*. Addison Wesley, 1994. pp. 628-631.

V.9 Direct Observation and Video/Audio Recording

Individual users may be directly observed doing specially devised tasks or doing their normal work, with the observer making notes about interesting behavior or recording their performance in some way, such as timing sequences of actions. This is called direct observation. When the observation takes place in the user organization, it is called field usability evaluation.

Video recording can be either an alternative to direct observation or a backup for what happens in a usability evaluation session. For field usability evaluation, audio recording can be useful as well to record the user comments.

The evaluator should be prepared to take copious notes as activities proceed during a usability evaluation session. It may be useful to have a second evaluator also observing the session in order to help take notes. Especially for usability testing sessions, the first evaluator may be in charge of conducting the session (giving instructions, prompting the user) and timing tasks where necessary, while the second evaluator may be in charge of just taking notes.

Even if the evaluator (or evaluators) is fast at note taking, the record of the observation will usually be incomplete. Direct observation only allows for one go at data collection, so the evaluator rarely gets a full record of user activity for detailed analysis. The evaluator has to make decisions about what is important to record and has no chance to revise that decision and look at alternative data later on. For these reasons, if a permanent record is needed, video recording equipment may be used to record usability evaluation sessions. Usability laboratories are usually equipped with video cameras and perhaps some video editing equipment as well. The main advantage of videotaping is to capture every detail that occurs during the session. If multiple cameras are available, one can be aimed at the participant's hands and the screen, and another at a broader view including the participant's face. Audiotaping may be done when videotaping is not available, as, for example, in field testing. Just having a record of all the user's comments may prove invaluable for later data analysis.

The main disadvantage of videotaping is the time it takes to edit the taped material. A ratio of 5:1 (analysis time to recording time) is often cited, that is, it usually takes five hours to analyze one hour of videotape. When more than one camera is used, the editing is also very time consuming, since synchronization problems may arise.

Short video clips of users experiencing problems with a given software product can have a big influence on a development team, especially, if the development team is reluctant to make changes to what they consider to be their already perfect design. These same video clips can also be useful for convincing management that there is a usability problem in the first place.

Basic Reference: D. Hix, H.R. Hartson. *Developing User Interfaces: Ensuring Usability Through Product and Process*. John Wiley and Sons, 1993. pp. 309-313.

V.10 Focus Groups

Focus groups are a somewhat informal technique that can be used to assess user needs and feelings after the system has been in use for some time. Focus groups often bring out spontaneous reactions and ideas from users through the interaction between the participants and have the major advantage of allowing some group dynamics and organizational issues. Focus groups are especially appropriate for limited user communities.

In a focus group, a group of users are brought together to discuss new concepts and identify issues over a period of about two hours. Each group is run by a moderator who is responsible for maintaining the focus of the group on whatever issues are of interest. From the user perspective, a focus group session should feel free flowing and relatively unstructured, but, in reality, the moderator has to follow a preplanned script for what issues to bring up.

To prepare a focus group, the moderator needs to prepare a list of the issues to be discussed and set goals for the kinds of information that are to be gathered. During the group session the moderator has the difficult job of keeping the discussion on track without inhibiting the free flow of ideas and comments. Also, the moderator needs to ensure that all members of the group get to contribute to the discussion and guard against having the opinions of any single participant dominate unduly. After the session, data analysis can be as simple as having the moderator write a short report summing up the prevailing mood in the group, illustrated with a few colorful quotes.

Focus group discussions may be held after a set of individual user interviews have been conducted. Then, focus-group discussions may be valuable to ascertain the universality of comments. Individual interviews are costly and time consuming, so usually only a small fraction of the user community is involved. On the other hand, group discussions offer more representative results.

Basic Reference: D. J. Mayhew. *The Usability Engineering Lifecycle*. Morgan Kaufmann, 1999. pp. 364-366.

V.11 Logging Actual Use

Logging involves having the computer automatically collect statistics about the detailed use of the system. It is mainly used to collect information about the field use of a system after release, but it can also be used as a supplementary method during usability testing to collect more detailed data. It is unobtrusive, so it does not interfere with the user’s normal usage of the system.

When the actual use of the system is logged, this information is particularly useful because it shows how users perform their actual work and because it is relatively easy to automatically collect data from a large number of users working under different circumstances. Typically, an interface log will contain statistics about the frequency with which each user has used each feature in the system, and the frequency with which various events of interest (like, for example, error messages) have occurred.

When undertaking a major redesign for a system that has been in use, it is very helpful to rely on interaction log information to guide the redesign effort.

For this technique to be applied, the software architecture should make it easy for system managers to collect data about the patterns of system usage, speed of user performance, rate of errors or requests for online assistance.

There are different software logging tools that can be employed for logging actual use, but there are two main categories: time-stamped keypresses and real-time interaction logging. Logging time-stamped keypresses simply provides a record of each key that the user presses along with the exact time of the event. Interaction logging is similar, except that the recording includes real-time information, which means that it can be replayed in real time so the observer can see the interaction between the user and the computer exactly as it happened.

Logging may be well intentioned, but user rights to privacy should be respected. Links to specific user names should not be collected, unless necessary. When logging aggregate performance crosses over to monitoring individual activity, managers must inform users of what is being monitored and how the information will be used.

It is usual to combine video, audio and keypresses or interaction logging. The advantage of using combinations of data capture techniques is that evaluators can relate revealing data about body language and comments with records of the actual human-computer interaction. The main disadvantage of this approach is the cost of setting up this kind of synchronized equipment.

Basic Reference: B. Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 1998. pp. 146-147.

V.12 User Feedback Facilities

Once the system is in use, the user community is the best source for information on the usability weaknesses of the system. Feedback from the users can be collected by giving them access to special electronic mail addresses, network newsgroups, or bulletin boards. Users can send their complaints and requests for change or improvement.

Offering a help line or a communication channel with users can be implemented in different forms. These are the main ways of gathering user-initiated feedback:

- **Online or Telephone Consultants**: They can provide extremely effective and personal assistance to users who are experiencing difficulties. Many users feel reassured if they know that there is a human being whom they can address if problems arise. These consultants are an excellent source of information about problems users are having and can suggest improvements and potential extensions. Some organizations offer a toll-free number for users, while others charge for consultation by the minute.
- **Online Suggestion Box or Trouble Reporting**: Email can be employed to allow users to send messages to the maintainers or designers. Such an "online suggestion box" encourages some users to make productive comments, since writing a letter may be seen as requiring too much effort.
- **Online Bulletin Board or Newsgroup**: Users may have questions about the suitability of a software package for their application, or may be seeking someone who has had experience using an interface feature. They do not have any individual in mind, so email does not serve their needs. Then bulletin boards and newsgroups can be helpful. Electronic bulletin boards or newsgroups allow the posting of open messages and questions. Mailing lists may be used as well for this purpose.

By soliciting user feedback by any of these ways, the development team can gauge user attitudes and elicit useful suggestions. Furthermore, users may have more positive attitudes towards the system if they see that the software development organization genuinely desires comments and suggestions on the piece of software they are using.

Basic Reference: B. Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 1998. pp. 147-149.

Bibliography

[Beyer, 98] Beyer, H., Holtzblatt, K. *Contextual Design. Defining Customer-Centered Systems*. Morgan Kaufmann, 1998.

[Bias, 94] Bias, R. G. “The Pluralistic Walk-Through: Coordinated Empathies”, in *Usability Inspection Methods*, edited by J. Nielsen and R. L. Mack. Wiley, 1994.

[Constantine, 99] Constantine, L. L., and Lockwood, L. A. D. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design*. Addison-Wesley, 1999.

[Cooper, 03] Cooper, A., Reimann, R. *About Face 2.0. The Essentials of Interaction Design*. Wiley Publishing, 2003.

[Dumas, 99] Dumas, J.S., Redish, J.C. *A Practical Guide to Usability Testing. Revised Edition*. Intellect, 1999.

[Hix, 93] Hix, D., Hartson, H.R. *Developing User Interfaces: Ensuring Usability Through Product and Process*. John Wiley and Sons, 1993.

[Kieras, 97] Kieras, D. “A guide to GOMS Model Usability Evaluation using NGOMSL”, in *Handbook of Human-Computer Interaction. Second Edition*, edited by M. Helander, T. Landauer and P. Prabhu. North-Holland, 1997. Chapter 31. pp. 733-766.

[Lewis, 97] Lewis, C., Wharton, C. “Scenario-Based Design”, in *Handbook of Human-Computer Interaction. Second Edition*. ed. by M. Helander, T. Landauer, P. Prabhu. Elsevier North-Holland, 1997. Chapter 17. pp. 383-406.

[Mayhew, 99] Mayhew, D.J. *The Usability Engineering Lifecycle*. Morgan Kaufmann, 1999.

[Nielsen, 93] Nielsen, J. *Usability Engineering*. AP Professional, 1993.

[Nielsen, 94] Nielsen. J., Mack, R. L. *Usability Inspection Methods*. Wiley, 1994.

[Preece, 94] Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., Carey T. *Human-Computer Interaction*. Addison Wesley, 1994.

[Robertson, 01] Robertson, J. *Information Design Using Card Sorting*. Step Two Designs Pty Ltd, 2001. http://www.steptwo.com.au/papers/cardsorting/

[Shneiderman, 98] Shneiderman, B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 1998.

[Snyder, 03] Snyder, C. *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces*. Morgan-Kaufmann, 2003.