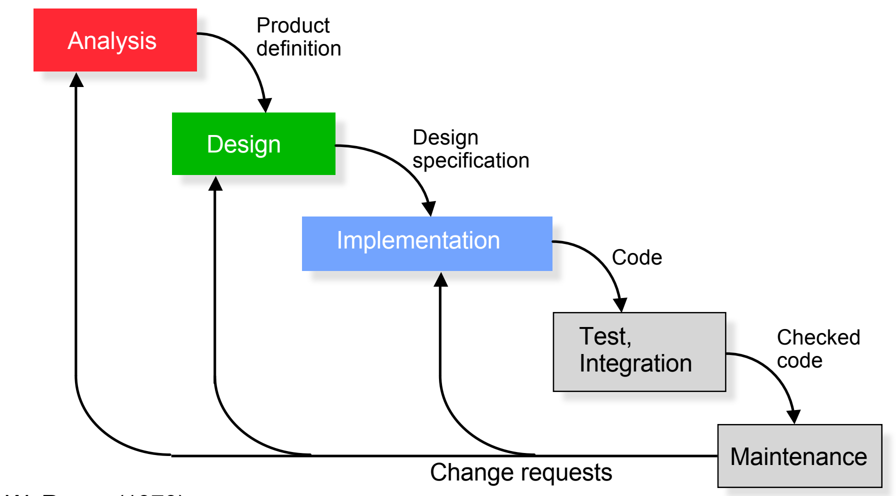
4 User-Centered Development Process

Software Development Process Model

- ***Process model***
  - Segmentation of the overall (team) activity of software development into smaller portions of work
    - High-level structure: phases
    - Low-level structure: steps, activities
  - Definition of an order for carrying out work units
  - Guideline for the production of intermediate results
- Basic activities covered in all models:
  - Analysis
  - Design
  - Implementation
  - Validation (in particular Test, Integration)
  - Evolution (in particular Maintenance)

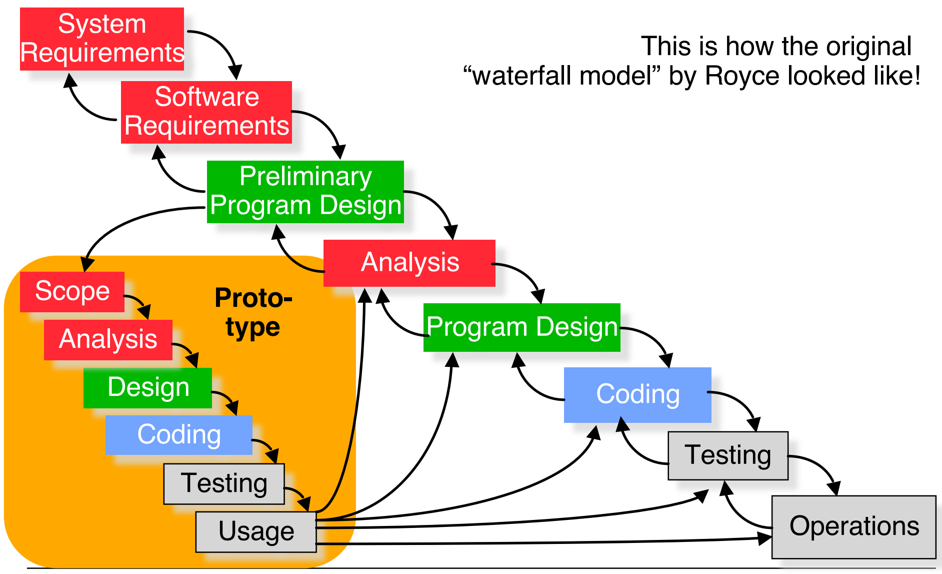
“Pure” Waterfall Model



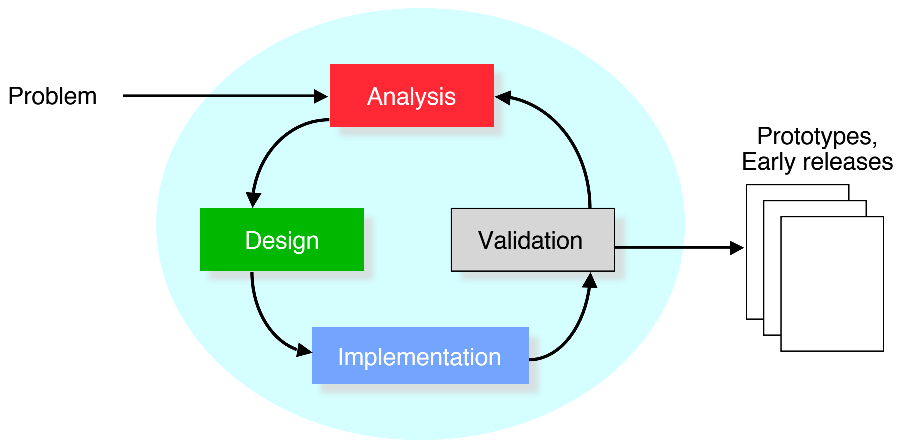
W. Royce (1970)

“Waterfall” with Feedback Loops and Prototyping

This is how the original “waterfall model” by Royce looked like!

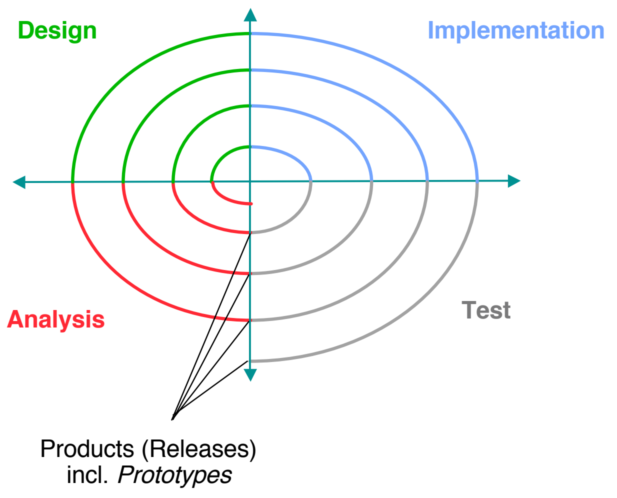


Evolutionary Development

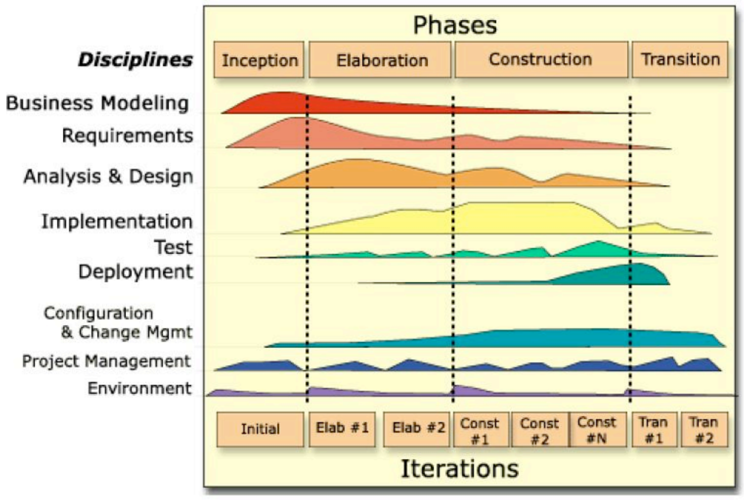


- Typical for small projects and experimental systems
- Technological progress (e.g. object orientation) may have improved scalability to large systems

Object Oriented Spiral Model

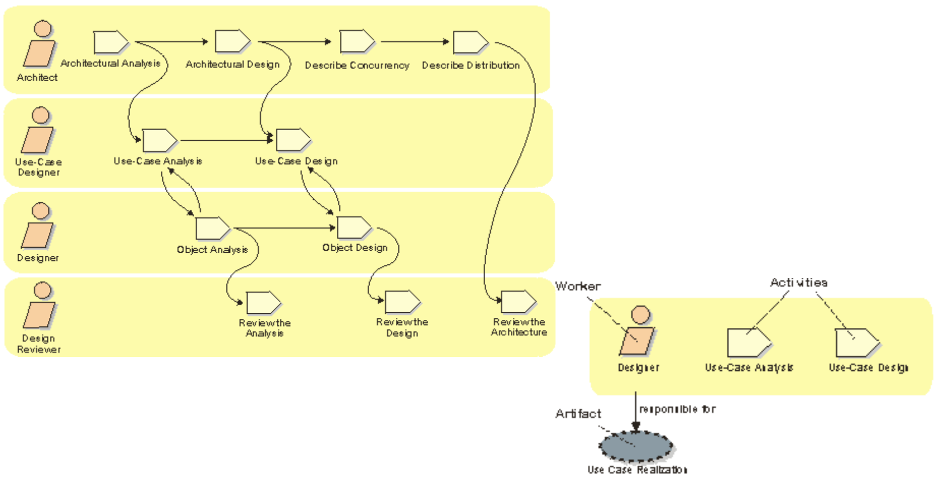


Rational Unified Process (RUP)



Software Process framework is a commercial product of Rational, now IBM.

Detailed Prescriptions in RUP



- Developers often consider this as not flexible enough fo creative work.

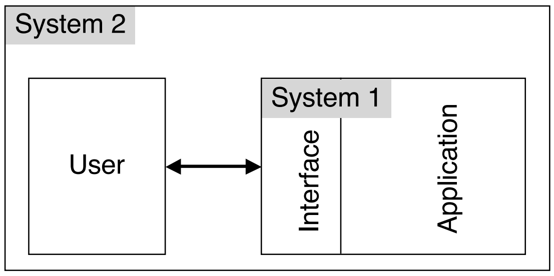
Agile Development

- "Agile" Software development (www.agilemanifesto.org):
  - E.g. Extreme Programming (XP), Crystal, Scrum
- Recent trend in software development processes
  - Radical evolutionary development
- Key characteristics of agile development:
  - Individuals and interactions (rather than processes and tools)
  - Working software (code rather than extensive documentation)
  - Customer collaboration (instead of contract negotiations)
  - Responding to change (instead of following a plan)
- Agile development is not just “hacking along”!
  - Clear and strict rules
- Mixed information about success in practice
  - Good experiences in small and innovative projects
  - Large-scale projects tend to stay “conservative”, mainly due to transparency for project management

4 User-Centered Development Process

Usability Aspects are Mostly Ignored by Software Engineers

- Example:
  - IEEE “SWEBOK” body of knowledge definition for SE mentions HCI as “related discipline” under the name “software ergonomics”
- System perspectives
  - SW Engineers take the “System 1” perspective
  - Usability Engineers take the “System 2” perspective (following examples)



Seffah/Desmarais/Metzker

Separation between Interaction Design and Technical Design

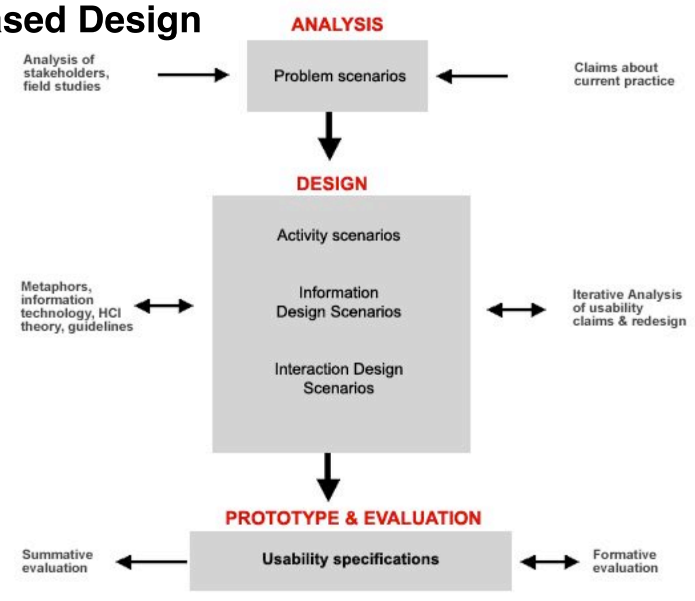
- For interactive applications a separation into a two stage process is often advisable
- 1st – Interaction design (iterative)
  - concept
  - Interaction analysis
  - Prototypes
  - Evaluation
  - Stable and tested design
- 2nd – technical realization
  - Technical analysis
  - Technical specification (e.g. architecture, platform)
  - Implementation
  - Evaluation and Quality management

LUCID Development Process

- Logical User Centered Interactive development Methodology (LUCID)
  - http://www.cognetics.com/lucid/index.html
- Stage 1: **Envision**
  - Develop UI Roadmap which defines the product concept, rationale, constraints and design objectives.
- Stage 2: **Analyze**
  - Analyze the user needs and develop requirements.
- Stage 3: **Design**
  - Create a design concept and implement a key screen prototype.
- Stage 4: **Refine**
  - Test the prototype for design problems and iteratively refine and expand the design.
- Stage 5: **Implement**
  - Support implementation of the product making late stage design changes where required. Develop user support components.
- Stage 6: **Support**
  - Provide roll-out support as the product is deployed and gather data for next version.

Scenario-Based Design

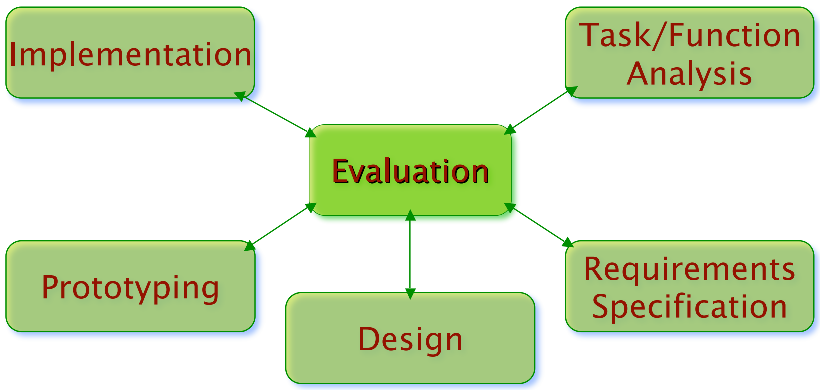
- Mary Beth Rosson, John M. Carroll: Usability Engineering - Scenario-Based Development of Human-Computer Interaction, Academic Press 2002


ANALYSIS
Analysis of stakeholders, field studies
Problem scenarios
Claims about current practice
DESIGN
Activity scenarios
Information Design Scenarios
Interaction Design Scenarios
Metaphors, information technology, HCI theory, guidelines
Iterative Analysis of usability claims & redesign
PROTOTYPE & EVALUATION
Summative evaluation
Usability specifications
Formative evaluation

Scenarios and Claims

- Scenario
  - Scenarios describe an existing or envisioned system from the perspective of one or more real or realistic users.
    - » Example: "A person turned on a computer; the screen displayed a button labeled Start; the person used the mouse to select the button."
- Claim
  - Claims are psychologically motivated design rationales that express the advantages and disadvantages of a design as a usability issue
  - Relating properties of the artifact with specific psychological consequences, under the scope of a basic task usage situation
    - » Example: Including open-ended exercises in an instruction manual supports learning-by-exploration
    - » Example: "Returning the user to the Create or Revise menu is adequate feedback that an option change attempt is successful *(but may not be enough feedback for users who are unsure or confused)*"
  - Encourages designer to reason about trade-offs rather than accepting a single guideline or principle

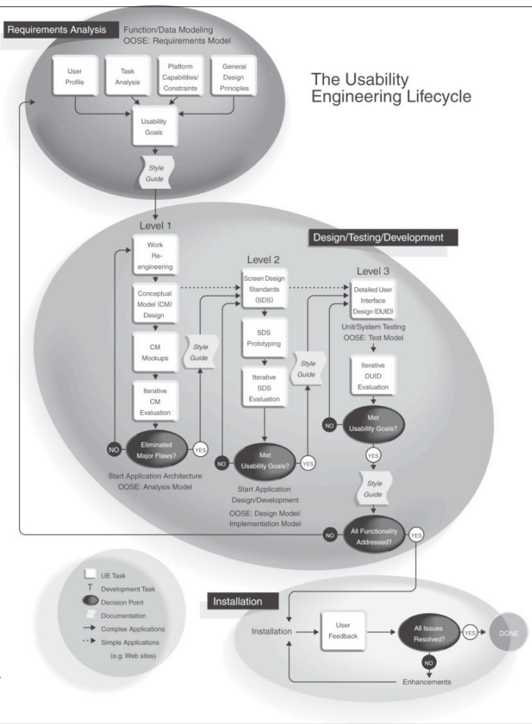
Star Lifecycle



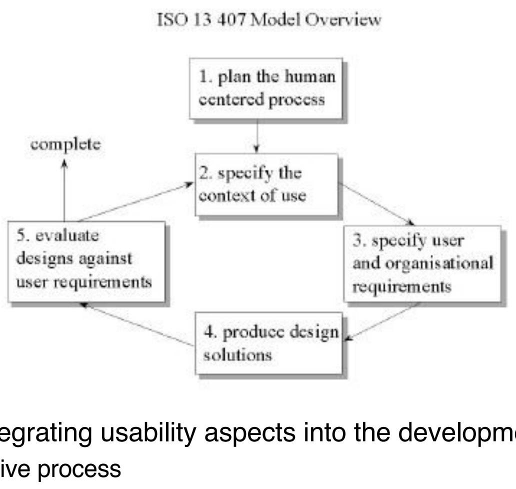
- Hix, Hartson 1993
  - Non-sequential: Any order of activities
  - Evaluation-centric: Every activity is evaluated
  - Interconnected: Evaluation connects everything

Usability Engineering Lifecycle

- D. Mayhew 1999
- Concrete guidance for projects
  - Complex but rather complete
  - Integrated with the "OOSE" development methodology



ISO 13407



ISO 13 407 Model Overview
1. plan the human centered process
2. specify the context of use
3. specify user and organisational requirements
4. produce design solutions
5. evaluate designs against user requirements
complete

- Guidelines for integrating usability aspects into the development process
  - Proposes iterative process
  - Stresses evaluation
  - Design solutions cover also lightweight prototypes, mock-ups etc.
- See e.g. http://www.ucc.ie/hfrg/emmus/methods/iso.html

Problems of User Centered Design

- Users may be wrong
- Users may be resistant to change
- Users may expect disadvantages (e.g. being replaced by software)

- Be aware – you are expected to create an optimal system with regard to ***the goals specified***
  - this is unfortunately NOT necessarily the system users would like to have (e.g. trade-off between employers and employees)

4 User-Centered Development Process

Concurrent Workflows, Competing Cultures

- User Interaction Design and Software/System Design are concurrent activities
  - One depends on the other, one influences the other
- Separate cultures led to competing models of the development process
  - Software Engineering: Artefact-centric (e.g. design documents), disciplined order of steps, quantitative evaluation (metrics, tests), highly compatible to project management needs
  - User-Centered Development: Interdisciplinary, loose (e.g. rough guidelines), flexible in order of steps, open to late changes, continuous qualitative evaluation (e.g. user tests with prototypes), difficult to “sell” to project managers
- Ambiguous overlaps in terminology
  - The same terms are used in many methods with differently defined or weakly defined semantics
    - » E.g. “scenario”, “use case”, “test”
- Integration of process models
  - “Interface development is transitioning from an artistic exercise into an engineering discipline.” (Curtis/Hefley)

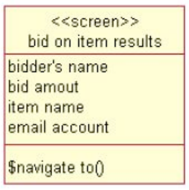
Concurrency of UI and SW Engineering

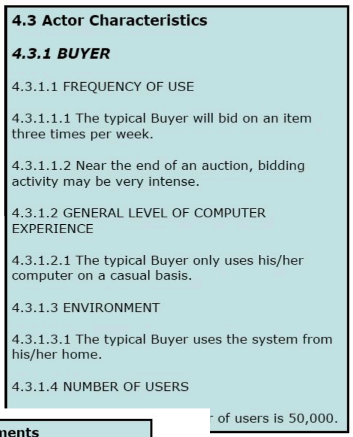
User Interface Engineering		Development Phase		Software Engineering
User & task analysis	⇐	**Requirements analysis**	⇒	Application design
Human vs. Machine	⇐	**Requirements allocation**	⇒	Hardware vs. Software
Dialogue design	⇐	**Preliminary design**	⇒	Architectural design
Screen design	⇐	**Detailed design**	⇒	Logical design
Coding	⇐	**Implementation**	⇒	Coding
Usability lab	⇐	**Implementation testing**	⇒	Unit & Integration testing
Contextual observation	⇐	**System testing**	⇒	System testing
Human performance	⇐	**Optimization**	⇒	Machine performance

(Curtis/Hefley)

User Experience “Plugin” Rational Unified Process

- Extensions of roles, activities and (UML) artifacts
  - Use cases extended by “use case storyboards”
  - UI Prototyping as a specific activity
  - Screens as special cases of classes (derived from Conallen’s UML-based Web Design Method)
- http://www-128.ibm.com/developerworks/rational/library/content/RationalEdge/nov03/f_usability_jh.pdf





4.3 Actor Characteristics

4.3.1 BUYER

4.3.1.1 FREQUENCY OF USE

4.3.1.1.1 The typical Buyer will bid on an item three times per week.

4.3.1.1.2 Near the end of an auction, bidding activity may be very intense.

4.3.1.2 GENERAL LEVEL OF COMPUTER EXPERIENCE

4.3.1.2.1 The typical Buyer only uses his/her computer on a casual basis.

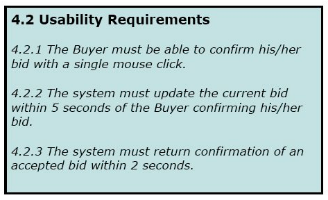
4.3.1.3 ENVIRONMENT

4.3.1.3.1 The typical Buyer uses the system from his/her home.

4.3.1.4 NUMBER OF USERS

of users is 50,000.

the Bid on Item use case



4.2 Usability Requirements

4.2.1 The Buyer must be able to confirm his/her bid with a single mouse click.

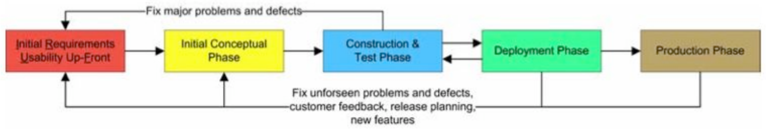
4.2.2 The system must update the current bid within 5 seconds of the Buyer confirming his/her bid.

4.2.3 The system must return confirmation of an accepted bid within 2 seconds.

Figure 7: Usability requirements for the Bid on Item use case

User-Centered Design and Agile Development

- Is a user-centered approach promoted or hindered by agile values?
  - Negative: Users are ***not*** at the center of agile development:
    Focus on core functionality, no distinction between client and (end) user
  - Positive: High value of communication and customer collaboration support usability
  - Positive: Iterative development and “response to change” are consistent with user-centered processes
  - Positive: Simplicity is a common key value for user-centered design and agile development (“design the simplest solution possible”)
- Attempts to integrate user-centered design and agile development
  - Results similar to non-agile approaches integrating SW and UI engineering?
  - Example: “Agile Usage-Centered Software Lifecycle”
    (Gundelsweiler/Memmel/Reiter, in: Mensch&Computer 2004)



Fix major problems and defects
Initial Requirements Usability Up-Front
Initial Conceptual Phase
Construction & Test Phase
Deployment Phase
Production Phase
Fix unforseen problems and defects, customer feedback, release planning, new features

References

- Ahmed Seffah, Jan Gulliksen, Michel C. Desmarais (eds.): Human-Centered Software Engineering - Integrating Usability in the Development Process, Springer 2005
- Mary Beth Rosson, John M. Carroll: Usability Engineering - Scenario-Based Development of Human-Computer Interaction, Academic Press 2002
- Deborah Hix and H. Rex Hartson: Developing User Interfaces Ensuring Usability Through Product & Process, John Wiley 1993
- Bill Curtis, Bill Hefley: A WIMP no more: the maturing of user interface engineering, ACM *interactions* 1(1), January 1994, 22-34
- John Armitage: Are agile methods good for design?, ACM *interactions* 11(1), Jan/Feb 2004, 14-23

4 User-Centered Development Process

Design Cycles & Prototyping

- Creating prototypes is important to get **early** feedback
  - from the project team (prototypes help to communicate)
  - from potential users
- Different types of prototypes
  - Low-fidelity prototypes (e.g. paper prototypes)
  - Hi-fidelity prototypes (e.g. implemented and semi-functional UI)
  - Fidelity is referring to detail
- Tools & Methods
  - Sketches & Storyboards
  - Paper prototyping
  - Using GUI-builders to prototype
  - Limited functionality simulations
  - Wizard of Oz

Testing prototypes to choose among alternatives

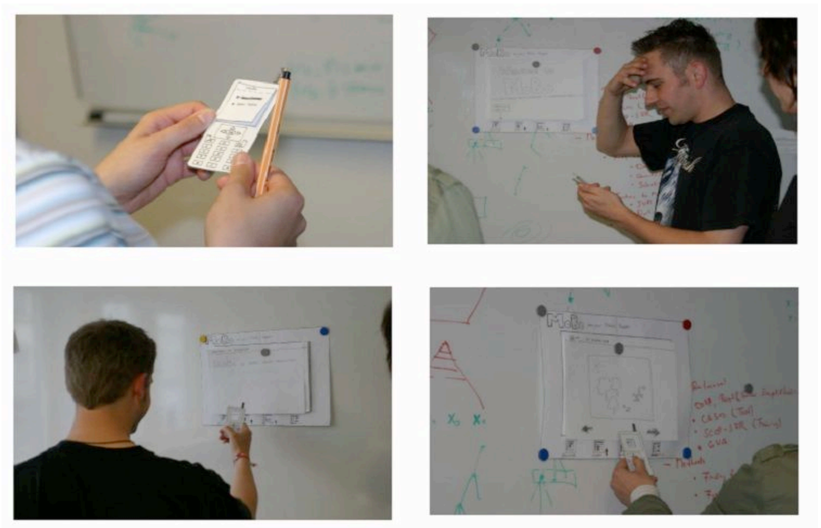
Low-Fidelity Prototyping

- Advantages of paper prototypes
  - Cheap and quick – results within hours!
  - Helps to find general problems and difficult issues
  - Make the mistakes on paper and make them before you do your architecture and the coding
  - Can save money by helping to get a better design (UI and system architecture) and a more structured code
  - Enables non-technical people to interact easily with the design team (no technology barrier for suggestions)
  - Provisional presentation invites observers to propose changes
- Get users involved!
  - To get the full potential of paper-prototypes these designs have to be tested with users
  - Specify usage scenarios
  - Prepare tasks that can be done with the prototype

Paper Prototypes

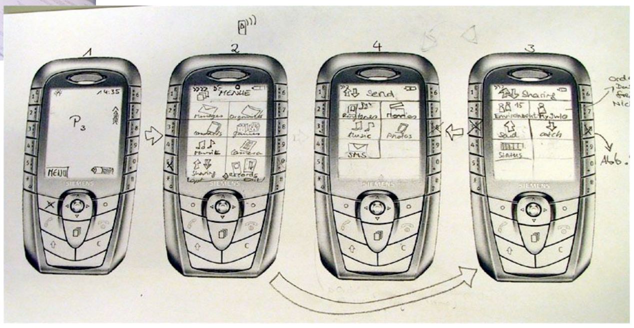
- Specify the set of tasks that should be supported
- Create a paper prototype using office stationery
  - Screens, dialogs, menus, forms, …
  - Specify the interactive behavior
- Use the prototype
  - Give users a specific task and observe how they use the prototype
  - Ask users to “think aloud” – comment what they are doing
  - At least two people
    - » One is simulating the computer (e.g. changing screens)
    - » One is observing and recording
- Evaluate and document the findings
  - What did work – what did not work
  - Where did the user get stuck or chose alternative ways
  - Analyze comments from the user
- Iterate over the process (make a new version)

Paper Prototyping – Example I

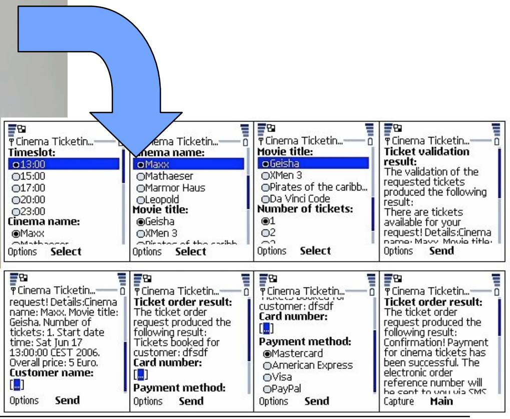


Sketches & Storyboards

- Storyboards as for movies
  - A picture for each key scene
- Sketch out the application
  - Key screens
  - Main interaction
  - Important transitions
- Helps to communicate and validate ideas
  - Easy to try out different option, e.g. document base vs. application based
- Ignore details, e.g.
  - what font to use, how icons will look like



Paper Prototyping – Example II



Cinema Ticketin...
Timeslot:
◉13:00
○15:00
○17:00
○20:00
○23:00
Cinema name:
◉Maxx
Options **Select**

Cinema Ticketin...
Cinema name:
◉Maxx
○Mathaeser
○Marmor Haus
○Leopold
Movie title:
◉Geisha
○XMen 3
Options **Select**

Cinema Ticketin...
Movie title:
◉Geisha
○XMen 3
○Pirates of the caribb...
○Da Vinci Code
Number of tickets:
◉1
○2
Options **Select**

Cinema Ticketin...
Ticket validation result:
The validation of the requested tickets produced the following result:
There are tickets available for your request! Details:Cinema
Options **Send**

Cinema Ticketin...
request! Details:Cinema name: Maxx. Movie title: Geisha. Number of tickets: 1. Start date time: Sat Jun 17 13:00:00 CEST 2006. Overall price: 5 Euro.
Customer name:
[...]
Options **Send**

Cinema Ticketin...
Ticket order result:
The ticket order request produced the following result:
Tickets booked for customer: dfsdf
Card number:
[...]
Payment method:
Options **Send**

Cinema Ticketin...
customer: dfsdf
Card number:
[...]
Payment method:
◉Mastercard
○American Express
○Visa
○PayPal
Options **Send**

Cinema Ticketin...
Ticket order result:
The ticket order request produced the following result:
Confirmation! Payment for cinema tickets has been successful. The electronic order reference number will
Capture **Main**

Minimize the time for design Iterations
Make errors quickly!

- Idea of rapid prototyping
- Enables the design team to evaluate more design options in detail
- If you go all the way before evaluating your design you risk a lot!
- Sketches and paper prototypes can be seen as a simulation of the real prototype

- Without paper prototyping:
  - Idea – sketch – implementation – evaluation

Slow Iteration

- With paper prototyping:
  - Idea – sketch/paper prototype – evaluation – implementation - evaluation

Quick Iteration **Slow Iteration**

High-fidelity Prototype

- Looks & feels like the final product to the user
  - Colors, screen layout, fonts, …
  - Text used
  - Response time and interactive behavior
- The functionality however is restricted
  - Only certain functions work (vertical prototype)
  - Functionality is targeted towards the tasks (e.g. a search query is predetermined)
  - Non-visible issues (e.g. security) are not regarded
- Can be used to predict task efficiency of the product
- Feedback often centered around the look & feel
- Standard technologies for implementation
  - HTML, JavaScript
  - Flash, Director, presentation programs
  - GUI Builder (e.g. Visual Basic, Delphi, NetBeans)