What is a requirement ?

- Statement about intended product that specifies what is should do and how it should perform
- Specific, unambiguous, clear
  - Time to download any page is less than 5 seconds on a DSL line of 20Mbits/second
  - The playback bar should always indicate the position in the audio stream
- Too vague
  - Appealing to teenage girls (too vague)
  - The buttons should be engaging

Different kinds of requirements

- Functional
  - What the system should do
  - Historically the main focus of requirements activities
  - Examples: word processor must support variety of formatting styles
- Non-functional
  - Run on a variety of platforms
  - Function on 64M of RAM
  - Delivered in 6 months
- Data
  - What kinds of data need to be stored ? Database ?

Different kinds of requirements

- Environment or context of use
  - Physical
    - Dusty ? Noisy ? Vibrations ? Light ? Heat ? Humidity ?
  - Social
    - Sharing of files, displays, privacy, locking
  - Organizational
    - Hierarchy, IT departments, user support, communications structure, availability of training
  - Users
    - Novice : step-by-step, constrained, clear information
    - Expert : flexibility, access, power
    - Frequent: short cuts
    - Casual/infrequent: clear instructions, menu paths

Different requirements

- Usability
  - Learnability, flexibility, throughput,
- IMPORTANT
  - User requirements and usability requirements refer to different things
- Kinds of requirements
  - What factors (enviornmental, user, usability) ?
    - Self-service cafeteria - paying using credit system
    - Nuclear plant control room
    - Distributed car design teams

An exampe

- **Self-service cafeteria**
- *Functional*: Calculate total cost of purchases
- *Data*: Access to price of products in cafeteria
- *Environmental*: Noisy and busy environment, users maybe talking while using the system
- *User*: Majority of users under 25, comfortable dealing with technology
- *Usability:* Simple for new users, memorable for frequent users, efficient, deal easily with user errors

Data gathering techniques (1)

- Questionnaires
  - A series of questions designed to elicit specific information
  - Questions may require different kinds of answers
    - Yes/No, multiple choice, general comments
  - Often used in conjuction with other techniques
  - Can give quantitative and qualitative data
  - Good for answering specific questions from a large, dispersed group of people

Data gathering (2)

- Interviews
  - Forum for talking to people
  - Structured, unstructured or semi-structured
  - Pros, e.g. Sample scenarios of use, prototypes, can be used in interviews
  - Good for exploring issues
  - Time consuming, infeasible for large user or dispersed user populations

Data gathering (3)

- Workshop or focus groups
  - Group interviews
  - Good at gaining a consensus view and/or highlighting areas of conflict

Data gathering (4)

- Naturalistic observation
  - Spend time with stakeholders in their day-to-day tasks, observing work as it happens
  - Gain insights into stakeholder's tasks
  - Good for understanding the context and nature of tasks
  - But, it requires time and commitment from a member of the design team, and it can result in a huge amount of data
  - Ethnography is one form

Data gathering (5)

- Studying documentation
  - Procedures and rules are often written in manuals
  - Good source of data about the steps involved in an activity, and any regulations governing a task
  - Not to be used in isolation
  - Good for understanding legislation, and getting background information
  - No stakeholder time, which is a limiting factor in the other techniques

Data gathering (6)

- Data logging
- Instrument software to record user's activity in a log that can be examined later
- RECORD EVERYTHING
- Easy to do
- Requires a prototype or existing software

Choosing between techniques

- Data gathering techniques differ in two main ways
  - Amount of time, level of detail and risk associated with the findings
    - Naturalistic observation (2 days of effort, 3 months of training)
    - Interview (1 day of effort, 1 month of training)
  - Knowledge the analyst requires
- Sequential steps, overlapping subtasks ?
- Complex or simple information ?
- Layman or skilled practitioner ?

Problems with data gathering (1)

- Identifying and involving stakeholders, managers, developers, customer reps etc.
- Involving stakeholders: workshops, interviews, workplace studies
- “Real” users not managers
  - Traditionally a problem in software engineering , better now

Problems with data gathering (2)

- Requirements management: version control, ownership
- Communication between parties
  - Within development team
  - With customer/user
  - Between users ... different parts of an organization use different terminology
- Domain knowledge distributed and implicit:
  - Difficult to dig up and understand
  - Knowledge articulation: how do you walk ?
- Availability of key people

Problems with data gathering (3)

- Political problems with the organization
- Dominance of certain stakeholders
- Economic and business environment changes
- Balancing functional and usability demands

Some basic guidelines

- Focus on identifying the stakeholders needs
- Involve all the stakeholder groups
- Involve more than one representative from each stakeholder group
- Use a combination of data gathering techniques
- Support process with props
- Run pilot session
- Carefully record data – always better to have more

Data interpretation and analysis

- Start soon after data gathering section
- Initial interpretation before deeper analysis
- Different approaches emphasize different elements e.g class diagrams for object-oriented systems, entity-relationship diagrams for data intensive systems

Task descriptions

- Scenarios
  - An informal narrative story, simple, “natural”, personal, not generalizable
- Use cases
  - Assume interaction with the system
  - Assume detailed understanding of the interaction
- Essential use cases
  - Abstract away from the details
  - Doesn't have the same assumptions as use cases

Scenario for shared calendar

“The user types in all the names of the meeting participants together with some constraints such as the length of the meeting, roughly when the meeting needs to take place, and possibly where it needs to take place. The system then checks against the individuals’ calendars and the central departmental calendar and presents the user with a series of dates on which everyone is free all at the same time. Then the meeting could be confirmed and written into people’s calendars. Some people, though, will want to be asked before the calendar entry is made. Perhaps the system could email them automatically and ask that it be confirmed before it is written in.”

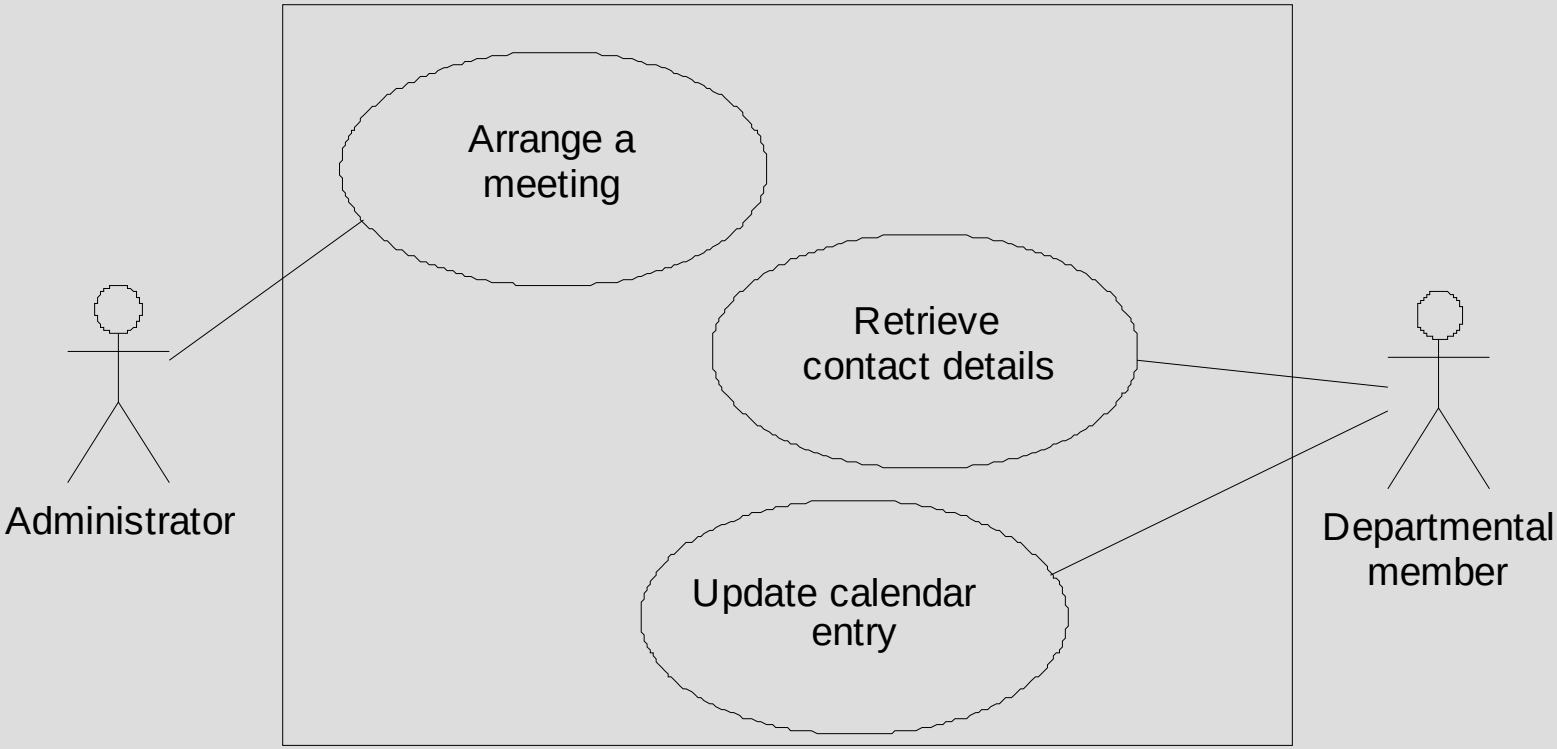
Use case

1. The user chooses the option to arrange a meeting.
2. The system prompts user for the names of attendees.
3. The user types in a list of names.
4. The system checks that the list is valid.
5. The system prompts the user for meeting constraints.
6. The user types in meeting constraints.
7. The system searches the calendars for a date that satisfies the constraints.
8. The system displays a list of potential dates.
9. The user chooses one of the dates.
10. The system writes the meeting into the calendar.
11. The system emails all the meeting participants informing them of them appointment

Alternative courses

- 5: If the list of people is invalid
  - 5.1 The system displays an error message
  - 5.2 The system returns to step 2
- 8: If no potential dates are found
  - 8.1 The system displays a suitable message
  - 8.2 The system returns to step 2

Example use case for shared calender



Essential use case for shared calendar

ArrangeMeeting

USER INTENTION	SYSTEM RESPONSIBILITY
arrange a meeting	
	request meeting attendees & constraints
identify meeting attendees & constraints	
	search calendars for suitable dates
	suggest potential dates
choose preferred date	
	book meeting

Task analysis

- Task descriptions are often used to envision new systems or devices
- Task analysis is mainly used to investigate an existing situation
- It is important not to focus on superficial acitivities
  - What are people trying to achieve ?
  - Who are they trying to achieve it ?
  - How are they going about it ?
- Many techniques, the most popular is Hierarchical Task Analysis (HTA)

Hierarchical Task Analysis

Annet and Duncan, 1967

- Involves breaking down a task into subtasks, then sub-sub-tasks and so on. These are grouped as plans which specify how tasks might be performed in practice
- HTA focuses on physical and observable actions, and includes looking at actions not related to software or an interaction device
- Start with the user goal which is examined and the main tasks for achieving it are identified
- Tasks are subdivided into subtasks

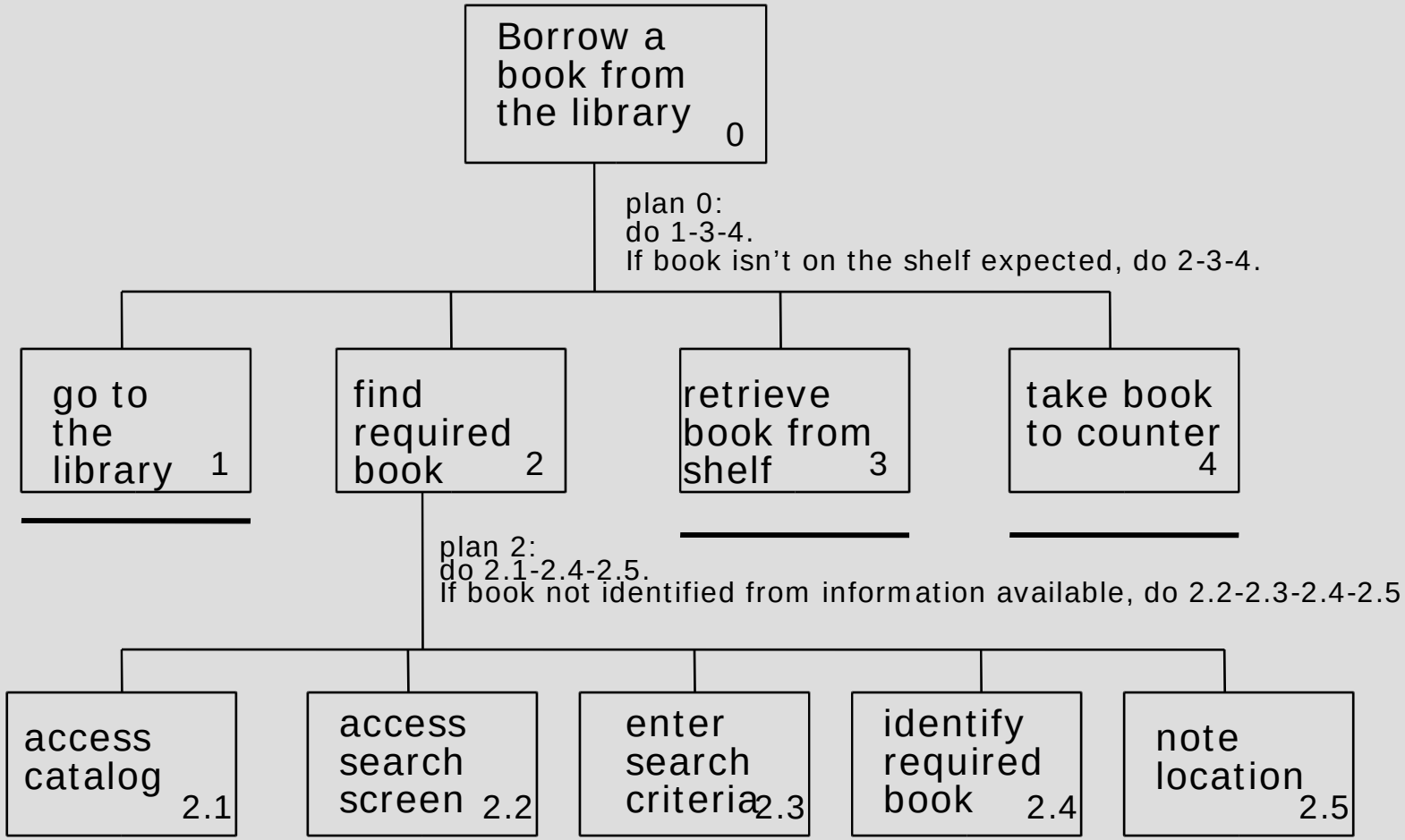
Example HTA

- 0: In order to borrow a book from a library
  - 1: go to the library
  - 2: Find the required book
    - 2.1 Access library catalog
    - 2.2 Access the search screen
    - 2.3 Enter search criteria
    - 2.4 Identify required book
    - 2.5 Note location
  - 3: Go to correct self and retrieve book
  - 4: Take book to checkout counter

Example HTA plans

- Plan 0: do 1-3-4 If book isn't on the self expected do 2-3-4
- Plan 2: do 2.1-2.4-2.5 If book not identified do 2.2-2.3-2.4

Example of graphical HTA



Based on the slides available at www.id-book.com

Summary

- Getting requirements right is crucial
- There are different kinds of requirements, each is significant for interaction design
- The most commonly-used techniques for data gathering are: questionnaires, interviews, focus groups and workshops, naturalistic observation, studying documentation
- Scenarios, use cases and essential use cases can be used to articulate existing and envisioned work practices
- Task analysis techniques such as HTA help to investigate existing systems and practices