chapter 6

HCI in the software process

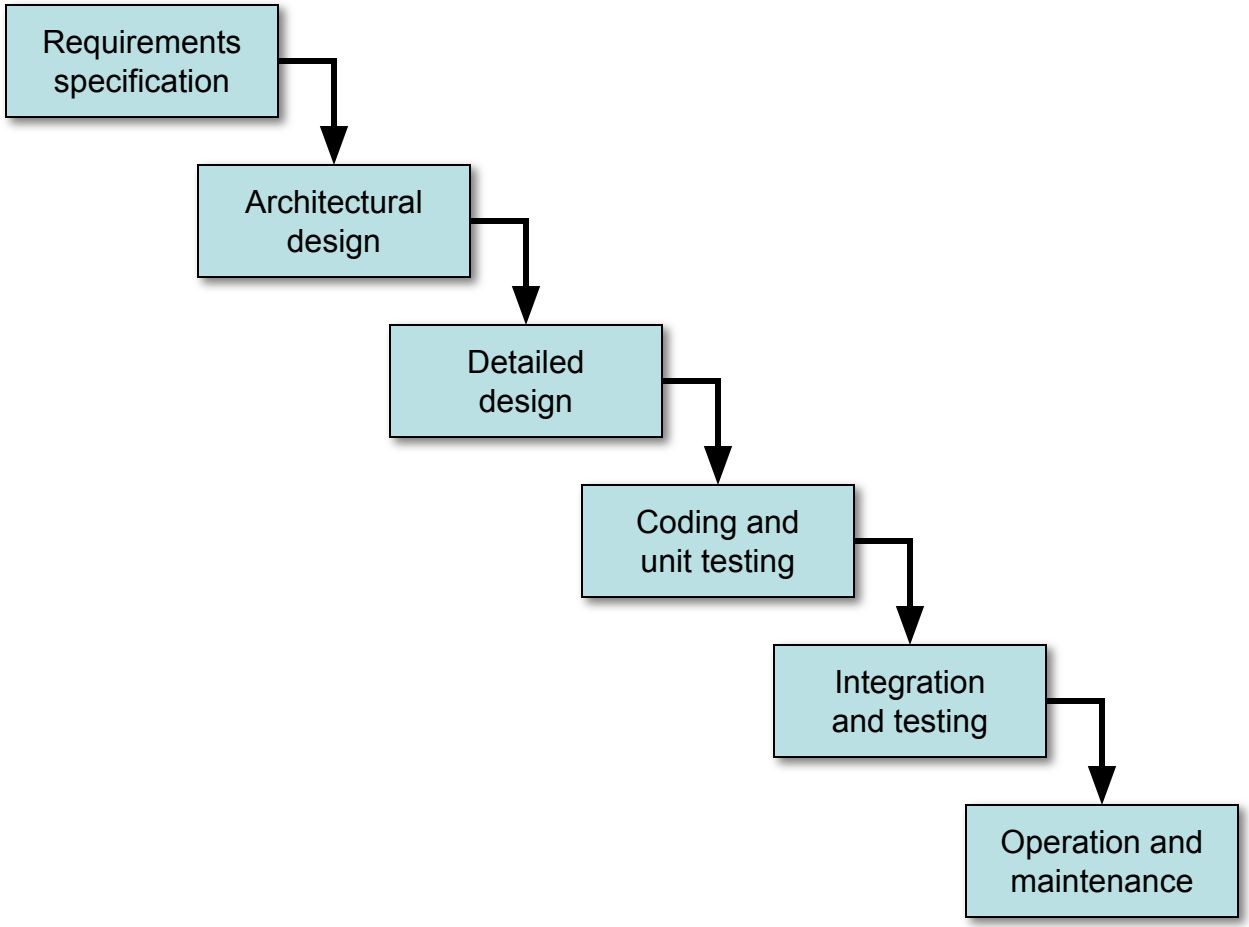
HCI in the software process

- Software engineering and the design process for interactive systems
- Usability engineering
- Iterative design and prototyping
- Design rationale

the software lifecycle

- Software engineering is the discipline for understanding the software design process, or life cycle
- Designing for usability occurs at all stages of the life cycle, not as a single isolated activity

The waterfall model



Activities in the life cycle

Requirements specification

- designer and customer try capture what the system is expected to provide can be expressed in natural language or more precise languages, such as a task analysis would provide

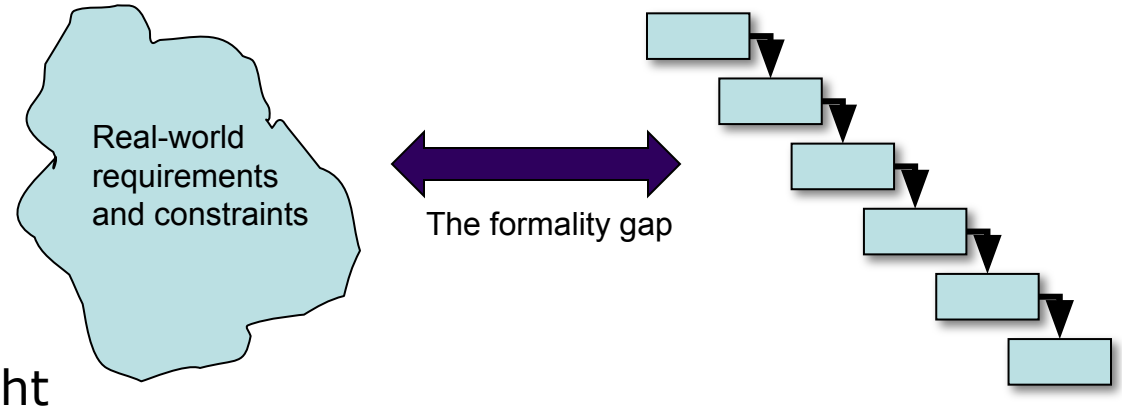
Architectural design

- high-level description of how the system will provide the services required factor system into major components of the system and how they are interrelated needs to satisfy both functional and nonfunctional requirements

Detailed design

- refinement of architectural components and interrelations to identify modules to be implemented separately the refinement is governed by the nonfunctional requirements

Verification and validation



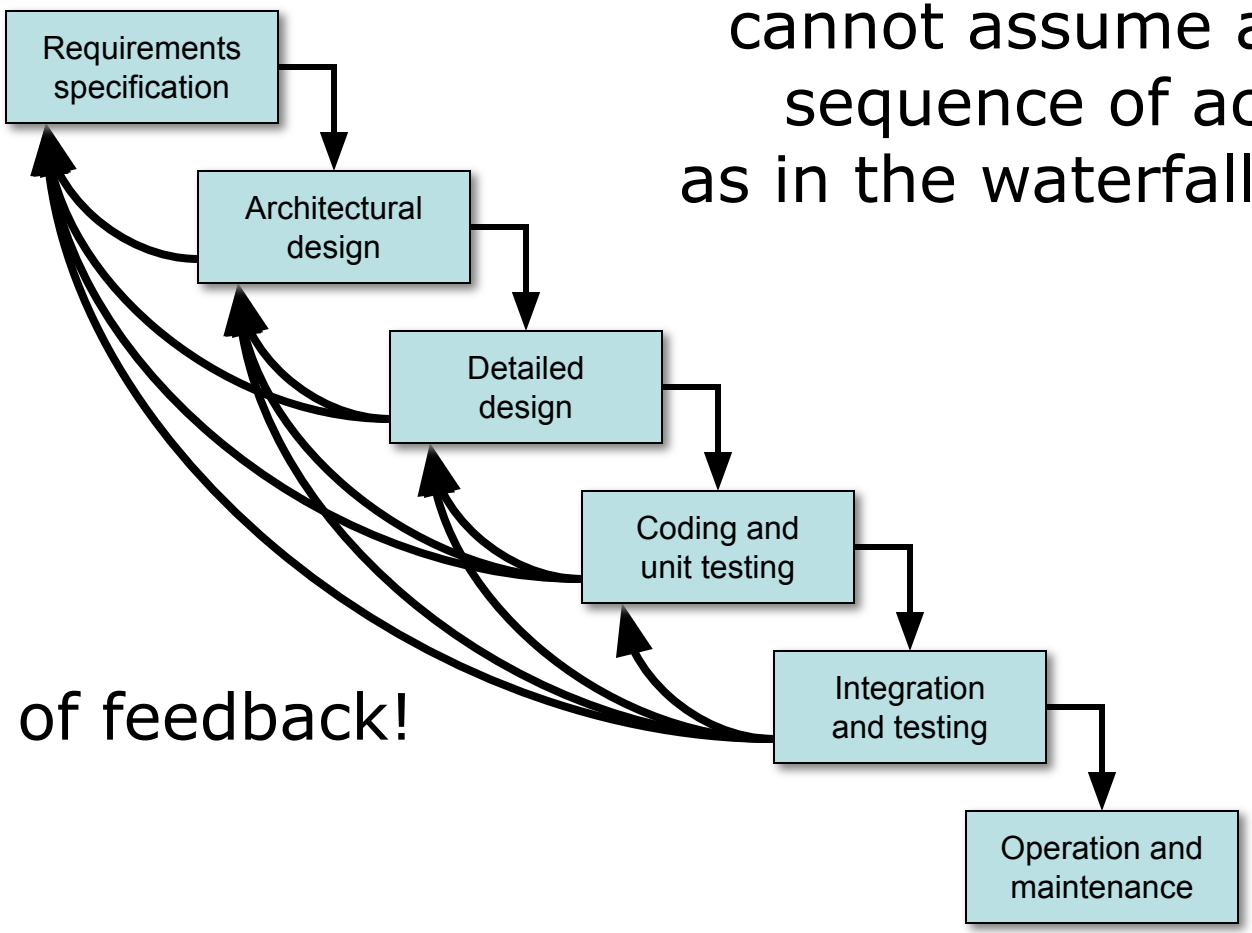
Verification
designing the product right

Validation
designing the right product

The formality gap
validation will always rely to some extent on subjective means of proof

Management and contractual issues
design in commercial and legal contexts

The life cycle for interactive systems



cannot assume a linear sequence of activities as in the waterfall model

lots of feedback!

Usability engineering

The ultimate test of usability based on measurement of user experience

Usability engineering demands that specific usability measures be made explicit as requirements

Usability specification

- usability attribute/principle
- measuring concept
- measuring method
- now level/ worst case/ planned level/ best case

Problems

- usability specification requires level of detail that may not be
- possible early in design satisfying a usability specification
- does not necessarily satisfy usability

part of a usability specification for a VCR

Attribute: Backward recoverability
Measuring concept: Undo an erroneous programming sequence
Measuring method: Number of explicit user actions to undo current program
Now level: No current product allows such an undo
Worst case: As many actions as it takes to program-in mistake
Planned level: A maximum of two explicit user actions
Best case: One explicit cancel action

ISO usability standard 9241

adopts traditional usability categories:

- effectiveness
  - can you achieve what you want to?
- efficiency
  - can you do it without wasting effort?
- satisfaction
  - do you enjoy the process?

some metrics from ISO 9241

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

Iterative design and prototyping

- Iterative design overcomes inherent problems of incomplete requirements

- Prototypes
  - simulate or animate some features of intended system
  - different types of prototypes
    - throw-away
    - incremental
    - evolutionary

- Management issues
  - time
  - planning
  - non-functional features
  - contracts

Techniques for prototyping

Storyboards
- need not be computer-based
- can be animated

Limited functionality simulations
- some part of system functionality provided by designers
- tools like HyperCard are common for these
- Wizard of Oz technique

Warning about iterative design
- design inertia – early bad decisions stay bad
- diagnosing real usability problems in prototypes….
- …. and not just the symptoms

Design rationale

Design rationale is information that explains why a computer system is the way it is.

Benefits of design rationale

- communication throughout life cycle
- reuse of design knowledge across products
- enforces design discipline
- presents arguments for design trade-offs
- organizes potentially large design space
- capturing contextual information

Design rationale (cont'd)

Types of DR:

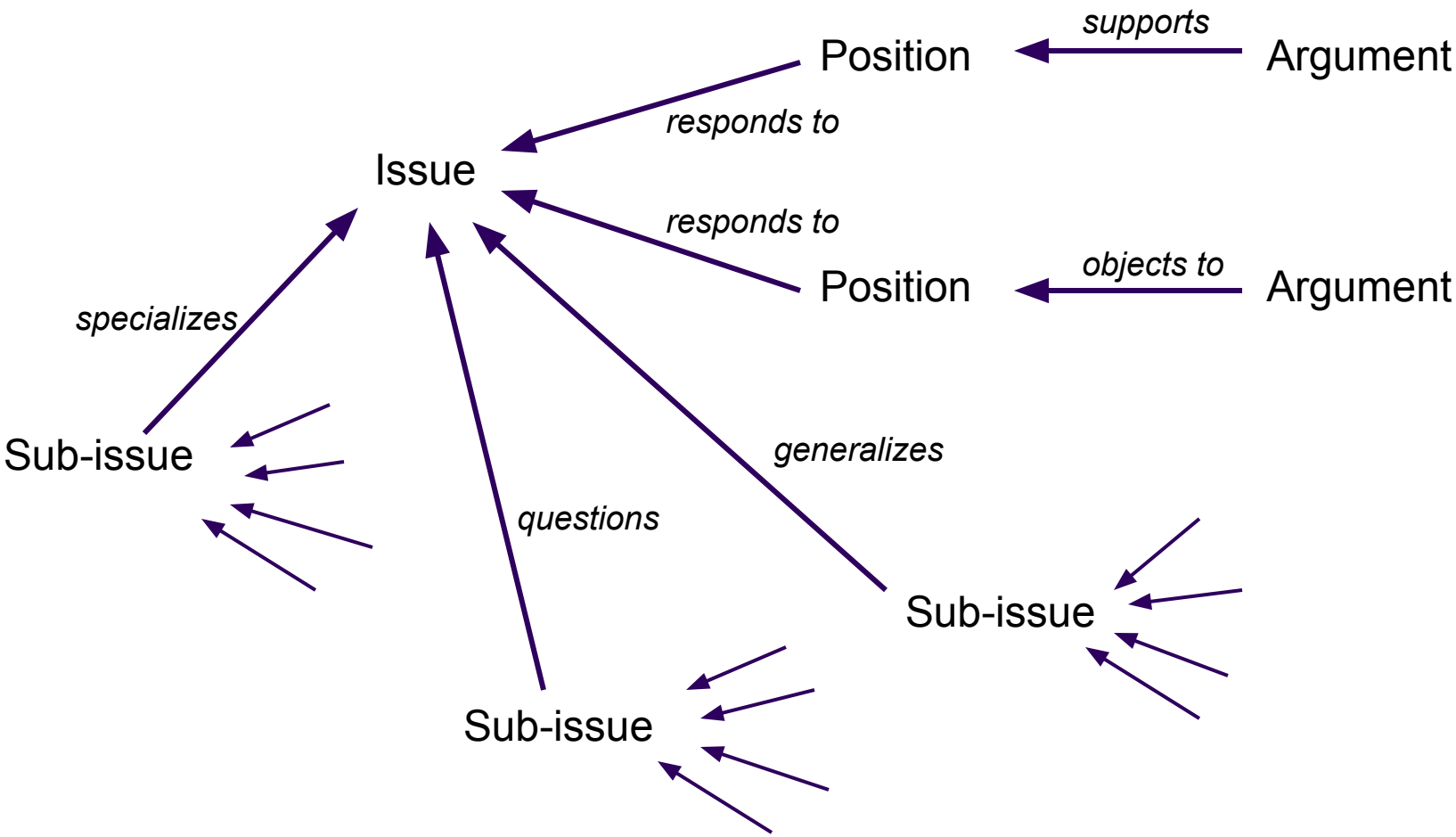
- Process-oriented
  - preserves order of deliberation and decision-making
- Structure-oriented
  - emphasizes post hoc structuring of considered design alternatives

- Two examples:
  - Issue-based information system (IBIS)
  - Design space analysis

Issue-based information system (IBIS)

- basis for much of design rationale research
- process-oriented
- main elements:
  - issues
    - hierarchical structure with one 'root' issue
  - positions
    - potential resolutions of an issue
  - arguments
    - modify the relationship between positions and issues
- gIBIS is a graphical version

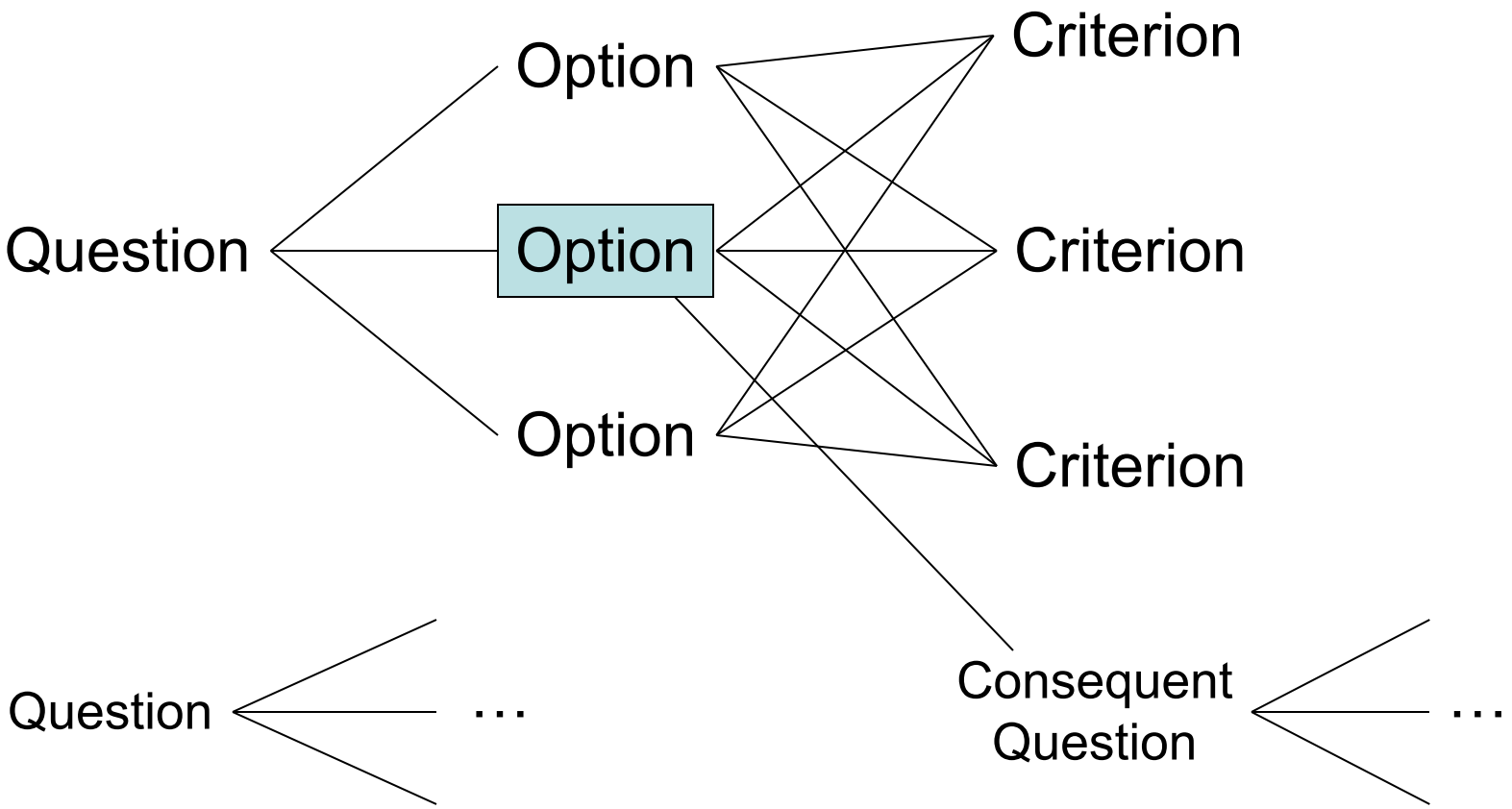
structure of gIBIS



Design space analysis

- structure-oriented
- QOC – hierarchical structure:
  - questions (and sub-questions)
    - represent major issues of a design
  - options
    - provide alternative solutions to the question
  - criteria
    - the means to assess the options in order to make a choice
- DRL – similar to QOC with a larger language and more formal semantics

the QOC notation



Psychological design rationale

- to support task-artefact cycle in which user tasks are affected by the systems they use
- aims to make explicit consequences of design for users
- designers identify tasks system will support
- scenarios are suggested to test task
- users are observed on system
- psychological claims of system made explicit
- negative aspects of design can be used to improve next iteration of design

Summary

The software engineering life cycle
- distinct activities and the consequences for interactive system design

Usability engineering
- making usability measurements explicit as requirements

Iterative design and prototyping
- limited functionality simulations and animations

Design rationale
- recording design knowledge
- process vs. structure