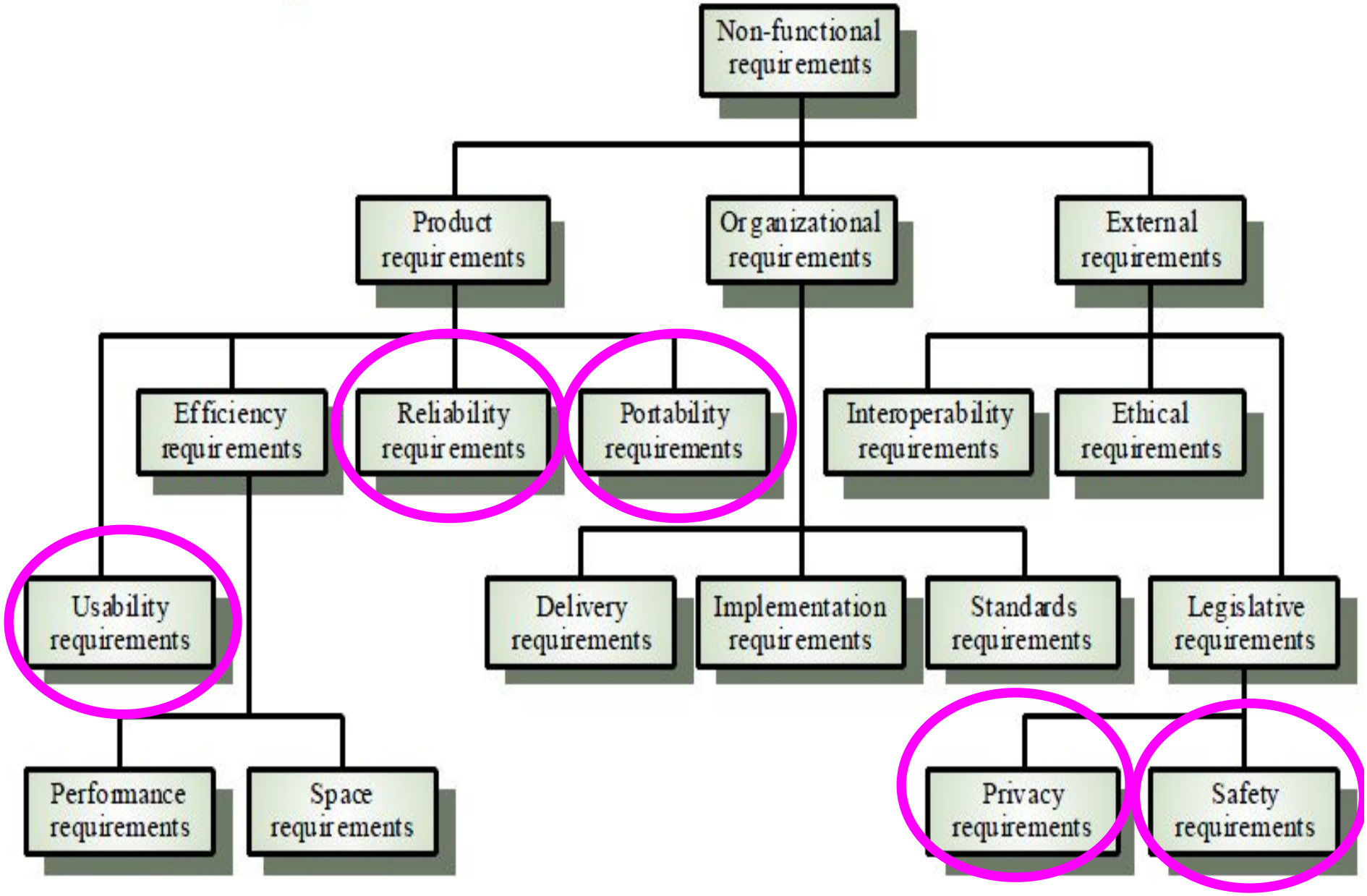
第9章. 非功能需求建模

主要内容

1. 什么是非功能需求？
2. 非功能需求建模的挑战
3. 非功能需求建模的方法

1. What are non-functional requirements ?

- Functional vs. Non-Functional
  - Functional requirements describe what the system should do
    - things that can be captured in use cases
    - things that can be analyzed by drawing interaction diagrams, statecharts, etc.
    - Functional requirements will probably trace to individual chunks(大块) of a program
  - Non-functional requirements are global constraints on a software system
    - e.g. development costs, operational costs, performance, reliability, maintainability, portability, robustness etc.
    - Usually cannot be implemented in a single module of a program


Non-functional requirements
Product requirements
Organizational requirements
External requirements
Efficiency requirements
Reliability requirements
Portability requirements
Interoperability requirements
Ethical requirements
Usability requirements
Delivery requirements
Implementation requirements
Standards requirements
Legislative requirements
Performance requirements
Space requirements
Privacy requirements
Safety requirements

Why it is hard to model non-functional requirements?

- The challenge of NFRs
  - Hard to model
  - Usually stated informally, and so are:
    - often contradictory,
    - difficult to enforce during development
    - difficult to evaluate for the customer prior to delivery
    - Hard to make them measurable requirements
  - We'd like to state them in a way that we can measure how well they've been met

Approaches to NFRs

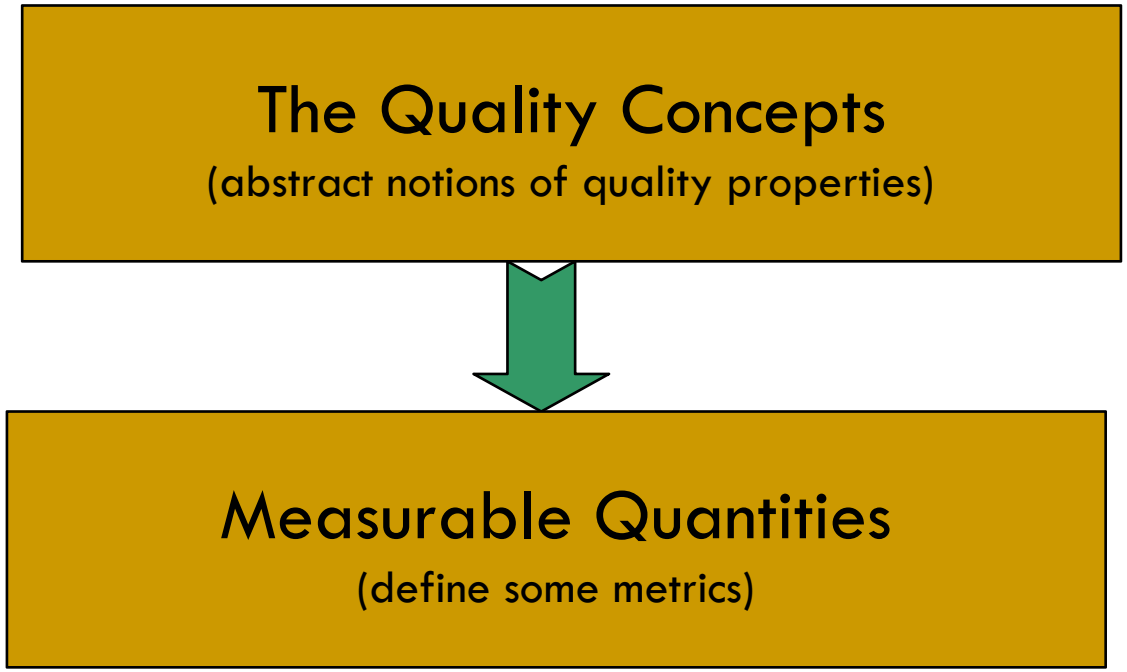
- Product vs. Process?
  - Product-oriented Approaches
    - Focus on system (or software) quality
    - Aim is to have a way of measuring the product once it's built
  - Process-oriented Approaches
    - Focus on how NFRs can be used in the design process
    - Aim is to have a way of making appropriate design decisions

Approaches to NFRs

- Quantitative vs. Qualitative?
  - Quantitative Approaches
    - Find measurable scales for the quality attributes
    - Calculate degree to which a design meets the quality targets
  - Qualitative Approaches
    - Study various relationships between quality goals
    - Reason about trade-offs etc.

Making Requirements Measurable

- We have to turn our vague ideas about quality into measurables (Quantification)



Quantification

- Non-functional requirements need to be measurable
  - Avoid subjective characterization: good, optimal, better...
- Values are not just randomly specified
  - Must have a rational (合理)
  - Stakeholder must understand trade-offs (平衡)
  - Important to rank and prioritize the requirements (排序;优先级)
- Precise numbers are unlikely to be known at the beginning of the requirement process
  - Do not slow down your initial elicitation process
  - Ensure that quality attributes are identified
  - Negotiate precise values later during the process

How to measure NFRs?

- Performance measurement
- Reliability measurement
- Availability measurement
- Security measurement
- Usability measurement
- Maintainability measurement
- Testability measurement
- Robustness measurement

Performance measurement

- Normally
  - Response time, number of events processed/denied in some interval of time, throughput(吞吐量), capacity, usage ratio, loss of information, latency...
  - Usually with probabilities, confidence interval
- Examples
  - The system shall be able to process 100 payment transactions per second in peak load;
  - In standard workload, the CPU usage shall be less than 50%, leaving 50% for background jobs;
  - Production of a simple report shall take less than 20 seconds for 95% of the cases;
  - Scrolling one page up or down in a 200 page document shall take at most 1 second;

Reliability measurement

- Definition: Probability that system will perform its required function for a specified interval under stated conditions
- Can be measured using
  - Mean-time to failure
  - Defect rate
  - Degree of precision for computations

- Examples
  - The precision of calculations shall be at least 10^{-6}.
  - The system defect rate shall be less than 1 failure per 1000 hours of operation.
  - No more than 1 per 1000000 transactions shall result in a failure requiring a system restart.

Availability measurement

- Definition: Percentage of time that the system is up and running correctly
- Can be calculated based on Mean-Time to Failure (MTBF) and Mean-Time to Repair (MTTR)
  - MTBF : Length of time between failures
  - MTTR : Length of time needed to resume operation after a failure
  - Availability = MTBF/(MTBF+MTTR)

Availability measurement

- Examples
  - The system shall meet or exceed 99.99% uptime(正常运行时间).
  - The system shall not be unavailable more than 1 hour per 1000 hours of operation.
  - Less than 20 seconds shall be needed to restart the system after a failure 95% of the time. (This is a MTTR requirement)

Security measurement

- There are at least two measures:
  1. The ability to resist unauthorized attempts at usage
  2. Continue providing service to legitimate users while under denial of service attack (resistance to DoS attacks)
- Measurement methods:
  - Success rate in authentication
  - Resistance to known attacks (to be enumerated)
  - Time/efforts/resources needed to find a key (probability of finding the key)
  - Probability/time/resources to detect an attack
  - Percentage of useful services still available during an attack
  - Percentage of successful attacks
  - Lifespan of a password, of a session
  - Encryption level

Security measurement

- Examples
  - The application shall identify all of its client applications before allowing them to use its capabilities.
  - The application shall ensure that the name of the employee in the official human resource and payroll databases exactly matches the name printed on the employee's social security card.
  - At least 99% of intrusions shall be detected within 10 seconds

Usability measurement

- In general, concerns ease of use and of training end users.
- The following more specific measures can be identified:
  - Learnability
    - Proportion of functionalities or tasks mastered after a given training time
  - Efficiency
    - Acceptable response time
    - Number of tasks performed or problems resolved in a given time
    - Number of mouse clicks needed to get to information or functionality
  - Memorability
    - Number (or ratio) of learned tasks that can still be performed after not using the system for a given time period
  - Error avoidance
    - Number of error per time period and user class
    - Number of calls to user support

Usability measurement

- Error handling
  - Mean time to recover from an error and be able to continue the task
- User satisfaction
  - Satisfaction ratio per user class
  - Usage ratio
- Examples
  - Four out of five users shall be able to book a guest within 5 minutes after a 2-hour introduction to the system.
  - Novice users shall perform tasks X and Y in 15 minutes.
  - Experienced users shall perform tasks X and Y in 2 minutes.
  - At least 80% of customers polled after a 3 months usage period shall rate their satisfaction with the system at 7 and more on a scale of 1 to10.

Maintainability measurement

- Measures ability to make changes quickly and cost effectively
  - Extension with new functionality
  - Deleting unwanted capabilities
  - Adaptation to new operating environments (portability)
  - Restructuring (rationalizing, modularizing, optimizing, creating reusable components)
- Can be measured in terms of
  - Coupling/cohesion(内聚耦合) metrics, number of anti-patterns, cyclomatic complexity
  - Mean time to fix a defect, mean time to add new functionality
  - Quality/quantity of documentation
- Measurement tools
  - code analysis tools such as IBM Structural Analysis for Java
  - http://www.alphaworks.ibm.com/tech/sa4j

Maintainability measurement

- Examples
  - Every program module must be assessed for maintainability according to procedure xx. 70% must obtain “highly maintainable” and none “poor”.
  - The cyclomatic complexity of code must not exceed 7. No method in any object may exceed 200 lines of code.
  - Installation of a new version shall leave all database contents and all personal settings unchanged.
  - The product shall provide facilities for tracing any database field to places where it is used.

Testability measurement

- Measures the ability to detect, isolate, and fix defects
  - Time to run tests
  - Time to setup testing environment (development and execution)
  - Probability of visible failure in presence of a defect
- Test coverage (requirements coverage, code coverage…)
  - May lead to architectural requirements
  - Mechanisms for monitoring
  - Access points and additional control
- Examples
  - The delivered system shall include unit tests that ensure 100% branch coverage.
  - Development must use regression tests allowing for full retesting in 12 hours.

Robustness measurement

- Measure ability to cope with the unexpected
  - Percentage of failures on invalid inputs
  - Degree of service degradation
  - Minimum performance under extreme loads
  - Active services in presence of faults
  - Length of time for which system is required to manage stress conditions
- Examples
  - The estimated loss of data in case of a disk crash shall be less than 0.01%.
  - The system shall be able to handle up to 10000 concurrent users when satisfying all their requirements and up to 25000 concurrent users with browsing capabilities.

本章小结

- 增强非功能需求建模的意识
- 非功能需求建模的两种方法
- 常用非功能需求的度量指标