Non-functional Requirements

Objectives

- To introduce non-functional requirements
- To explain the schemes used to classify non-functional requirements
- To illustrate various derivation techniques for non-functional requirements
- To demonstrate the importance of non-functional requirements in critical systems

Non-functional requirements (NFR)

- Non-functional requirements define the overall qualities or attributes of the resulting system
- Non-functional requirements place restrictions on the product being developed, the development process, and specify external constraints that the product must meet.
- Examples of NFR include safety, security, usability, reliability and performance requirements.

Functional and Non-functional requirements

- There is no a clear distinction between functional and non-functional requirements.
- Whether or not a requirement is expressed as a functional or a non-functional requirement may depend:
  - on the level of detail to be included in the requirements document
  - the degree of trust which exists between a system customer and a system developer.

Example

- The system shall ensure that data is protected from unauthorised access.
  - Conventionally, this would be considered as a non-functional requirement because it does not specify specific system functionality which must be provided. However, it could have been specified in slightly more detail as follows:
  - The system shall include a user authorisation procedure where users must identify themselves using a login name and password. Only users who are authorised in this way may access the system data.
  - In this form, the requirement looks rather more like a functional requirement as it specifies a function (user login) which must be incorporated in the system.

Types of Non-functional requirements

- The 'IEEE-Std 830 - 1993' lists 13 non-functional requirements to be included in a Software Requirements Document.
  - Performance requirements
  - Interface requirements
  - Operational requirements
  - Resource requirements
  - Verification requirements
  - Acceptance requirements

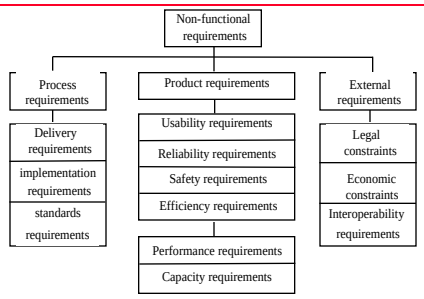
Types of NFRs (contd.)

- Documentation requirements
- Security requirements
- Portability requirements
- Quality requirements
- Reliability requirements
- Maintainability requirements
- Safety requirements

Classification of NFRs

- NFRs may be classified n terms of qualities that a software must exhibit (Boehm)
- A more general classification distinguishes between product, process and external requirements

Classification of NFRs (contd.)



Product requirements

- Specify the desired characteristics that a system or subsystem must possess.
- Most NFRs are concerned with specifying constraints on the behaviour of the executing system.

Specifying product requirements

- Some product requirements can be formulated precisely, and thus easily quantified
  - Performance
  - Capacity
- Others are more difficult to quantify and, consequently, are often stated informally
  - Usability

Examples of product requirements

- The System service X shall have an availability of 999/1000 or 99%. This is a reliability requirement which means that out of every 1000 requests for this service, 999 must be satisfied.
- System Y shall process a minimum of 8 transactions per second. This is a performance requirement.
- The executable code of System Z shall be limited to 512Kbytes. This is a space requirement which specifies the maximum memory size of the system.

Source code requirements

- There are product requirements which relate to the source code of the system
- Examples
  - The system shall be developed for PC and Macintosh platforms. This is a portability requirement which affects the way in which the system may be designed
  - The system must encrypt all external communications using the RSA algorithm. This is a security requirement which specifies that a specific algorithm must be used in the product

Conflicts in product requirements

- Product requirements are often conflict. For example:
  - A requirement for a certain level of performance may be contradicted by reliability and security requirements which use processor capacity to carry out dynamic system checking
  - A requirement on the space utilisation of the system may be contradicted by another requirement which specifies that a standard compiler which does not generate compact code must be used
- The process of arriving at a trade-off in these conflicts depends on:
  - The level importance attached to the requirement
  - The consequence of the change on the other requirements and,
  - The wider business goals

Process requirements

- Process requirements are constraints placed upon the development process of the system
- Process requirements include:
  - Requirements on development standards and methods which must be followed
  - CASE tools which should be used
  - The management reports which must be provided

Examples of process requirements

- The development process to be used must be explicitly defined and must be conformant with ISO 9000 standards
- The system must be developed using the XYZ suite of CASE tools
- Management reports setting out the effort expended on each identified system component must be produced every two weeks
- A disaster recovery plan for the system development must be specified

External requirements

- May be placed on both the product and the process
- Derived from the environment in which the system is developed
- External requirements are based on:
  - application domain information
  - organisational considerations
  - the need for the system to work with other systems
  - health and safety or data protection regulations
  - or even basic natural laws such as the laws of physics

Examples of external requirements

- *Medical data system* The organisation's data protection officer must certify that all data is maintained according to data protection legislation before the system is put into operation.
- *Train protection system* The time required to bring the train to a complete halt is computed using the following function:

The deceleration of the train shall be taken as:

$$\gamma_{train} = \gamma_{control} + \gamma_{gradient}$$

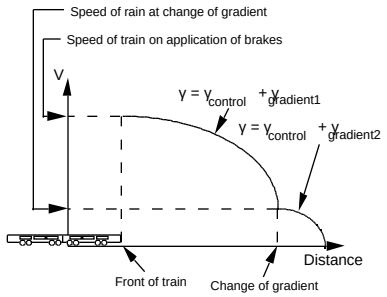
where:

External requirement example (contd.)

$\gamma_{gradient}$ = 9.81 ms^{-2} * compensated gradient / alpha and where the values of *9.81 ms^{-2}/ alpha* are known for the different types of train.

$\gamma_{control}$ is initialised at 0.8 ms^{-2} - this value being parameterised in order to remain adjustable. The illustrates an example of the train's deceleration by using the parabolas derived from the above formula where there is a change in gradient before the (predicted) stopping point of the train.

External requirement example (contd.)



External requirement example (contd.)

- The first of the the requirements described comes from the need for the system to conform to data protection legislation
- The second comes from the application domain and is a specification of the physical braking characteristics of a train.
- External requirements rarely have the form "the system shall..." or 'the system shall not...". Rather, they are descriptions of the system's environment which must be taken into account.

Deriving NFRs

- NFRs are difficult to express
- A number of important issues contribute to the problem of expressing non-functional requirements:
  - Certain constraints are related to the design solution that is unknown at the requirements stage
  - Certain constraints, are highly subjective and can only be determined through complex empirical evaluations
  - Non-functional requirements tend to be related to one or more functional requirements
  - Non-functional requirements tend to conflict and contradict
  - There are no 'universal' rules and guidelines for determining when non-functional requirements are optimally met.

Stakeholder concerns

- Stakeholders normally have a number of 'concerns'
- Concerns are typically non-functional
- Examples include:
  - Critical business objectives
  - Essential system characteristics (e.g. security)
  - Safety, performance, functionality and maintainability
- Vaguely defined user concerns may be related to NFRs

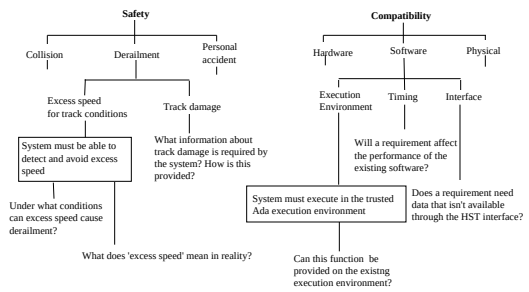
Relationships between user needs, concerns and NFRs

User's need	User's concern	Non-functional requirement
Function	1. Ease of use 2. Unauthorised access 3. Likelihood of failure	1. Usability 2. Security 3. Reliability
Performance	1. Resource utilisation 2. Performance verification 3. Ease of interfacing	1. Efficiency 2. Verifiability 3. Interoperability
Change	1. Ease of repair 2. Ease of change 3. Ease of transport ? 4. Ease of expanding or upgrading capacity or performance ?	1. Maintainability 2. Flexibility 3. Portability 4. Expandability

Concerns

- A way of expressing critical ‘holistic’ requirements
- Concerns may be broken down into sub-concerns and finally into specific questions
- Questions act as a check list to ensure that specific requirements do not conflict with global priorities

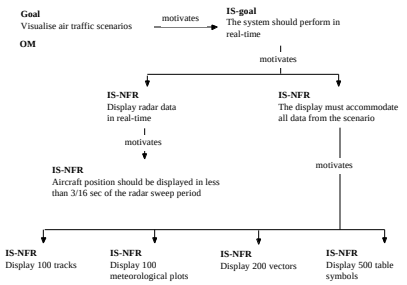
Concern decomposition



Goal-based derivation

- Relates non-functional requirements to the goals of the enterprise
- Goal-based NFR derivation is a 3 step approach:
  - Identify the enterprise goal
  - Decompose of the goal into sub-goals
  - Identify non-functional requirements.

Example of goal-based derivation



Testable NFRs

- Stakeholders may have vague goals which cannot be expressed precisely
- Vague and imprecise ‘requirements’ are problematic
- NFRs should satisfy two attributes
  - Must be objective
  - Must be testable (Use measurable metrics)
- It is not always possible to express NFRs objectively

Examples of measurable metrics for NFRs

Property	Metric
Performance	1. Processed transactions per second 2. Response time to user input
Reliability	1. Rate of occurrence of failure 2. Mean time to failure
Availability	Probability of failure on demand
Size	Kbytes
Usability	1. Time taken to learn 80% of the facilities 2. Number of errors made by users in a given time period
Robustness	Time to restart after system failure
Portability	Number of target systems

Requirements for critical systems

- Systems whose ‘failure’ causes significant economic, physical or human damage to organisations or people.
- There are three principal types of critical system:
  - Business critical systems
  - Mission critical systems
  - Safety critical systems

NFRs for critical systems

- The principal non-functional constraints which are relevant to critical systems:
  - Reliability
  - Performance
  - Security
  - Usability
  - Safety

Reliability

- Constraints on the run-time behaviour of the system
- Can be considered under two separate headings:
  - Availability - is the system available for service when requested by end-users.
  - Failure rate - how often does the system fail to deliver the service expected by end-users.

Performance

- Constrain the speed of operation of a system
- Types of performance requirements:
  - Response requirements
  - Throughput requirements
  - Timing requirements

Security

- Security requirements are included in a system to ensure:
  - Unauthorised access to the system and its data is not allowed
  - Ensure the integrity of the system from accidental or malicious damage
- Examples of security requirements are:
  - The access permissions for system data may only be changed by the system's data administrator
  - All system data must be backed up every 24 hours and the backup copies stored in a secure location which is not in the same building as the system
  - All external communications between the system's data server and clients must be encrypted

Usability

- Concerned with specifying the user interface and end-user interactions with the system
- Well structured user manuals, informative error messages, help facilities and consistent interfaces enhance usability

Usability metrics

- Measurable attributes of usability requirements include:
  - *Entry requirements* Measured in terms of years of experience with class of applications or simply age
  - *Learning requirements* Denotes the time needed to learn the facilities of the system. This could be measured in terms of speed of learning, say hours of training required before independent use is possible
  - *Handling requirements* Denotes the error rate of the end-users of the system. This could be measured in terms of the errors made when working at normal speed
  - *Likeability* Denotes 'niceness' to use. The most direct to measure user satisfaction is to survey actual users and record the proportion who 'like to work with the product'

COQUAMO approach to measuring usability

- Provides an automated support facility for setting and measuring usability
- Provides a set of 'templates' which have to be filled in

COQUAMO usability template

Classification	General quality - application dependent
Level required	High
Associated qualities	
synonyms	learnability, operability, user friendliness
related concepts	understandability
Definition 1	ease with which users can learn to use system
Explosion 1.1	average time for specific classes of user to achieve level of competence with system
Measurement unites	days
measuring too/data source	survey forms and results of survey
Measurement conditions	new graduate intake number required for test is 20
Worst case	8
Planned level	4
Best case	2
Current level	10
Justification	improvement specifically requested by XYZ
Consequence of failure	cost

Safety

- No consensus in the system's engineering community about what is meant by the term 'safety requirement'
  - Sometimes considered to be all requirements (functional and non-functional) on safety-related systems
  - Sometimes the sub-set of these requirements which are directly related to ensuring safe operation and sometimes requirements on protection systems which are designed to protect against accident
- Usage of the term often depends on the culture and practice of the organisation

Safety requirement definition

- Informal definition:
  Safety requirements are the 'shall not' requirements which exclude unsafe situations from the possible solution space of the system

Examples of safety requirements

- The system shall not permit operation unless the operator guard is in place
- The system shall not allow the sedative dose delivered to the patient to be greater than the maximum value which is determined by the patient's physician
- The system shall not operate if the external temperature is below 4 degrees Celsius

Requirements engineering for safety-related systems

- The requirements engineering process for safety-related systems should incorporate a specific safety-analysis activity
- The widely accepted model of the system critical systems life cycle (BCS and IEE 1989) identifies two stages in the safety analysis process:
  - *Safety requirements discovery*
  - *Safety validation*

Hazard analysis

- As part of the process of safety analysis, there are various activities such as hazard identification and analysis
- Various methods such as fault-tree analysis and Petri net analysis have been developed for safety validation

Integrating safety analysis with RE

- We illustrate in the following sections a simple technique for integrating safety analysis with a requirements engineering method
- The technique is intended to support the process of requirements discovery (safety requirements) and validation

Integrating safety analysis with RE (contd.)

- The starting point for specifying the system is a set of abstract organisational needs and constraints
- It is important that the approach used incorporates a systematic approach to dealing safety issues. These include:
  - Identifying hazards, risks and risk criteria
  - Identifying the necessary risk reduction to meet the risk criteria
  - Defining the overall safety requirements specification for the safeguards necessary to achieve the required risk reduction

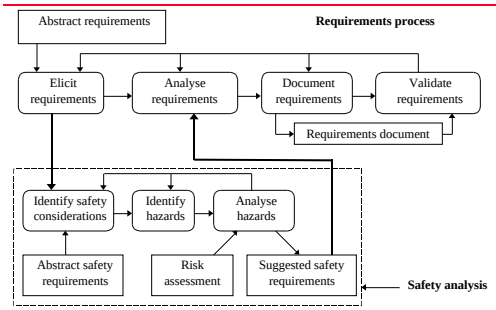
Integrating safety analysis with RE (contd.)

- The requirements process, shown in , has been extended to incorporate an explicit safety analysis activity
- The safety analysis process is based on requirements information drawn from the requirements elicitation and documentation process
- A set of abstract safety requirements serves as a reference model for identifying initial safety considerations or concerns

Integrating safety analysis with RE (contd.)

- The safety analysis process includes:
  - The identification of safety considerations
  - Hazard identification
  - Hazard analysis
  - Risk analysis and derivation of safety requirements
- Proposed safety requirements are analysed together with other system requirements to ensure that safety is not compromised

Integrating safety analysis with RE (contd.)



Key points

- Non-functional requirements define the overall qualities or attributes of the resulting system
- Non-functional requirements may be classified into three main types; product, process and external requirements
- Product requirements specify the desired characteristics that the system or subsystem must posses
- Non-functional requirements tend to conflict and interact with other system requirements
- The principal non-functional constraints which are relevant to critical systems are reliability, performance, security, usability and safety