Non-Functional Requirements

Lawrence Chung
Department of Computer Science
The University of Texas at Dallas

Why Non-Functional Requirements (NFRs)?

• Consider a brochure from an automobile manufacturer:
  – When you buy our car, you can now drive to a store…

• Consider a brochure from a cellular phone manufacturer:
  – When you buy our cellular phone, you can now call your friend.
  – Well, …

Why NFRs?

• With automobiles:
  – The basic function is transportation from one location to another.
  – "With premium luxury, outstanding safety features and superior off-pavement capability, … continue to exceed the high expectations of its owners, … continue to set the standard for premium luxury in its segment."

• With cellular phones:
  – The basic function is communication with another party
  – "… enhancements enable the best possible operation of your mobile … in various conditions. … The earpiece fits in either ear allowing for convenient and discreet access to all basic call controls. … To maximize call security, the headset also supports encryption of the wireless connection for compatible … models.

• With home networking:
  – "is the total home networking solution … linking variety of digital home appliances as one. It enables you to enjoy convenient, pleasant, and comfortable living environment at any time and any place.

• With CASE tool software:
  – The basic function is provision of some services
  – "… is a powerful, easy-to-use application definition platform used by business experts to quickly assemble functionally rich simulations of Web-based applications in a matter of hours. … Using the easy to learn, drag-and-drop paradigm, business people can quickly lay out the page flow of simulations and create high fidelity pages that precisely mimic not only the look and feel of the final…"
Non-functional requirements (NFRs) are a software requirement that describes how the software will do it, not what the software will do. In software system engineering, a software requirement that describes how the software will do it, for example, software performance requirements, software external interface requirements, design constraints, and software quality attributes. Nonfunctional requirements are difficult to test; therefore, they are usually evaluated subjectively.

**General Observations**

Non functional requirement – generally informally stated, often contradictory, difficult to enforce during development and evaluate for the customer prior to delivery.

**What are Non-Functional Requirements?**

- **-ilities**: understandability, usability, modifiability, interoperability, reliability, portability, maintainability, scalability, (re-)configurability, customizability, adaptability, variability, volatility, traceability, ...
- **-ities**: security, simplicity, clarity, ubiquity, integrity, modularity, nomadicy, ...
- **-ness**: user-friendliness, robustness, timeliness, responsiveness, correctness, completeness, conciseness, cohesiveness, ...
- ...and many other things: performance, efficiency, accuracy, precision, cost, development time, low coupling, ...

**NFRs: Some classification schemes - 1**

- [Roman, IEEE Computer 1985]
  - Interface requirements: describe how the system is to interface with its environment, users and other systems. E.g., user interfaces and their qualities (e.g., user-friendliness)
  - Performance requirements: describe performance constraints involving time/space bounds, such as workloads, response time, throughput and available storage space. E.g., “system must handle 100 transactions/second”
  - Reliability involving the availability of components and integrity of information maintained and supplied to the system. E.g., “system must have less than 1hr downtime/3 months”
  - Security, such as permissible information flows
  - Survivability, such as system endurance under file, natural catastrophes
- [Sommerville 1992]
  - Life Cycle: can be classified under two subcategories
    - quality of the design: measured in terms such as maintainability, enhanceability, portability.
    - limits on development: such as development time limitations, resource availability, methodological standards, etc.
  - External requirements: immediate and/or long-term costs
- Political requirements

**NFRs: Some classification schemes - 2**

- Process, Product and External considerations
  - Influences on system qualities
  - Process considerations
  - Product considerations
  - External considerations
  - Development method
  - Integration
  - Implementation environment
  - Security
  - Capacity
  - Test coverage
  - Cost
  - Design choices
  - Etc.
  - Social factors
  - Economic factors
  - Contract factors
  - Political factors

Lawrence Chung
NFRs:
Some classification schemes - 3

McCall’s NFR list

- Functionality
- Feature set capabilities, security, generality
- Usability
- Human factors aesthetics, consistency, documentation
- Reliability
- Frequency/severity of failure, recoverability, predictability, accuracy, MTBF
- Performance
- Speed efficiency, resource usage, throughput, response time
- Supportability
- Testability, Extensibility, Adaptable, Maintainability, Compatibility, Configurability, Serviceability, Installability, Localizability, Robustness

NFRs:
Some classification schemes - 4

- Uncionality
- Feature set capabilities, security, generality
- Sability
- Human factors aesthetics, consistency, documentation
- Eiability
- Frequency/severity of failure, recoverability, predictability, accuracy, MTBF
- Eformance
- Speed efficiency, resource usage, throughput, response time
- Supportability
- Testability, Extensibility, Adaptable, Maintainability, Compatibility, Configurability, Serviceability, Installability, Localizability, Robustness

NFRs:
Some classification schemes - 5

Software Quality Tree [Boehm 1976]

NFRs & RE Processes:
Why?

- Garbage in garbage out, so get the right requirements
- Garbage thru garbage out, so get the right process

Evolution is inevitable – traceability is a virtue
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 – metrics
  - **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

- **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

NFRs & RE Processes:

- Before FRs?
- After FRs?
- At the same time with FRs?
- ...and what about Business objectives/goals, system architectures, system models, SS, SRS, ...?

But, should we perhaps better know about the various relationships between NFRs and such and such, before answering these questions, more clearly, understandably, concisely, precisely, agreeably, ...?

Product-oriented approaches

Making Requirements Measurable

We have to turn our vague ideas about quality into measurable

The Quality Concepts (abstract notions of quality properties)

- Reliability
- Complexity
- Usability

Measurable Quantities (define some metrics)

- Mean time to failure?
- Information flow between modules?
- Time to learn how to use?

Counts taken from Design Representations (realization of the metrics)

- Run it and count crashes per hour??
- Count procedures calls??
- Minutes taken for some user task??

Quality Metrics:

<table>
<thead>
<tr>
<th>Property</th>
<th>Relative weight</th>
<th>Rating</th>
<th>Weighted Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
<td>3</td>
<td>6</td>
<td>18</td>
</tr>
<tr>
<td>Size</td>
<td>4</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>Ease of use</td>
<td>1</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Overall quality</td>
<td>10</td>
<td></td>
<td>50</td>
</tr>
</tbody>
</table>

Product-oriented approaches

Lawrence Chung

Lawrence Chung

Lawrence Chung
NFRs: Portability

- The degree to which software running on one platform can easily be converted to run on another platform
- E.g., number of target statements (e.g., from Unix to Windows)
- Hard to quantify, since it is hard to predict what a “next generation” platform might be like
- Can be enhanced by using languages, OSs and tools that are universally available and standardized.

E.g., C/C++/C#/Java

Lawrence Chung

NFRs: Reliability

- the ability of the system to behave consistently in a user-acceptable manner when operating within the environment for which the system was intended.
- theory and practice of hardware reliability are well established; some try to adopt them for software
- one popular metric for hardware reliability is mean-time-to-failure (MTTF)

“Bathtub” curve characterizes

Infant mortality:
Given a large population of a particular component, many will fail soon after development due to inaccuracies in the manufacturing process;
- Issues:
  - Do 2 different software copies have similar characteristics?
  - Does software wear & tear by decomposition?

NFRs: Efficiency

- refers to the level at which a software system uses scarce computational resources, such as CPU cycles, memory, disk space, buffers and communication channels
- can be characterized along a number of dimensions:
  - Capacity: maximum number of users/terminals/transactions ...
  - Degradation of service: what happens when a system with capacity X widgets per time unit receives X+1 widgets?
    - Let the system handle the load, perhaps with degraded performance
    - Let the system crash

Testing constrains:
Let stimulus refer to an action performed by the user/environment, and response refer to an action generated by the system.

- stimulus-response, e.g., “the system will generate a dial tone within 10 secs from the time the phone is picked up”
- response-stimulus, e.g., “the system will respond to the dial tone with a dial tone within 1 micro second after it had detected a dial tone”
- stimulus-observation, e.g., “the user will type her name and press enter from the time she inputs her name”
- response-observation, e.g., “the user will start dialing the phone number within 1 minute of getting the dial tone”
NFRs: Usability

- **Broadly** - quality fit to use
  - **narrowly** - good UI

- Usability inspection:
  - Finding usability problems in UI design, making recommendations for fixing them, and improving UI design.

- **Heuristics**: A set of criteria against which usability of UI design is evaluated
  - "9 usability heuristics" [Nielsen 90]
    - Promptness: No undue delay in accepting input items and responding to requests.
    - Tolerance: No hang-ups against errors, delays, unexpected behavior, etc.
    - Guidance: Providing guidance for correcting errors, generating reminders, etc.
    - Coherence: ... ...

- "10 usability heuristics" [Molich and Nielsen 90]
  - **Simple** and **natural** dialogue: Speak the user’s language.
  - **Minimize** the user’s memory: Consistency; Feedback.
  - **Clearly** marked exits: Shortcuts.
  - **Precise** and **constructive** error messages: Prevent errors.

NFRs: Usability

- All users will be satisfied with the usability of the product.
- 95% of all users will be satisfied with the usability of the product.
- 95% of the users will be able to complete representative tasks without requiring assistance (e.g., modifying exclusion date set).
- 95% of the users will be able to complete representative tasks by the third attempt without requiring assistance.
- 95% of the users will be able to complete tasks X Y Z by the third attempt without requiring assistance.
- 95% of the users will be able to complete task Y in less than 10 minutes without requiring assistance.
- 95% of the users will be able to complete task Z in less than 5 minutes.

Dependability

- **Dimensions of Dependability**
  - The ability of the system to deliver services when requested.
  - **Reliability**: The ability of the system to deliver services as specified.
  - **Safety**: The ability of the system to operate without catastrophic failure.

- **Cost of development**: Geometric rise in cost from low dependability to highest.

- **Effects of low dependability**
  - Often unused.
  - Failure recovery costs may be high.
  - Difficult to retrofit dependability.
  - Loss of information.

- Repeatable improvement process helps
  - CMM -SEI
  - More later.

Dependability

- **Cost of failure**
  - Direct:
    - Loss of life / injury.
    - Loss of business.
    - Indirect.
    - Legal.
    - Good will.

- **Availability and Reliability**
  - Factors affecting:
    - Environment office versus university.
    - Perception (frequency of occurrence).

- **Degrees**
  - Failure - service that is expected is not delivered.
  - Error - behavior that does not conform to the specification.
  - Fault - incorrect state anticipated.
  - Human error.

- **Critical Systems**
  - Safety critical.
  - Mission critical.
  - Business critical.

- **Dependability a key aspect**
  - A system failure causes:
    - Significant economic loss.
    - Physical damage.
    - Threat to or loss of human life.

- **Errors of this type are random**
  - Remain after testing due to unforeseen combinations of input or use.
  - Random based on user methods:
    - Not all inputs done the same.
    - Learn to avoid.
  - Therefore removal of some faults will not improve perception.

- **Improve reliability**
  - Fault avoidance.
  - Fault detection and removal - testing and debugging.
  - Fault tolerance - self checking and redundancy.

- **Designs**
  - Error - behavior that does not conform.
  - Fault - incorrect state.

Lawrence Chung
 Dependability - Safety

- Ability to operate normally or abnormally without threat to life or environment
  - Classes
    - Primary safety critical
      - Embedded as controller
    - Secondary
      - Output could effect indirectly other processes (CAD)
- Reasons for less than 100% certainty of fault tolerant/free
  - Incomplete specification
  - Hardware malfunction -- causing exceeded limits in software
  - Incorrect input

- Methods to lessen chance of safety failure
  - Hazard avoidance
  - Hazard detection and removal
    - Scenarios for known causes and cause preventive action
  - Damage limitation (control)
    - Freewalls and other protective reactions to results

- Classes
  - Primary safety critical
    - Embedded as controller
  - Secondary
    - Output could effect indirectly other processes (CAD)

- Reasons for less than 100% certainty of fault tolerant/free
  - Incomplete specification
  - Hardware malfunction -- causing exceeded limits in software
  - Incorrect input

- Methods to lessen chance of safety failure
  - Hazard avoidance
  - Hazard detection and removal
    - Scenarios for known causes and cause preventive action
  - Damage limitation (control)
    - Freewalls and other protective reactions to results

- Terms
  - Accident
  - Hazard
  - Damage
  - Hazard Severity
  - Hazard Probability

- Dependability - Security

Lack of security comprise to availability and reliability
- Types
  - Lack of service
  - Corruption of programs or data
  - Unauthorized disclosure
- Terms
  - Exposure
  - Vulnerability
  - Attack
  - Threats
  - Controls
- Methods
  - Vulnerability avoidance
  - Detection and neutralization
  - Damage limitation

- Security Specification
  - Similar to safety
  - Impractical to specify
    - Usually are "shall not"
- Cycle in General
  - Asset ID and evaluation
  - Degree of importance
  - Threat analysis and risk assessment
  - Threat assignment lists all threats against each asset
  - Technology analysis what is available to counteract

- Specification

- Safety
  - IEC 61508 safety life cycle
    - Concept to death
      - Hazard analysis
      - Safety requirements definition
      - Planning, validation, development, and execution
      - Separate safety validation -- installation and commissioning
      - OEM
      - Decommissioning
  - Hazard and Risk Analysis
    - Iterative process
      - Hazard identification
        - Hazard description
        - Risk analysis and hazard classification
        - Risk assessment
      - Hazard decomposition
        - Analysis, on to potential causes
      - Risk reduction analysis
      - Hazard and Risk Analysis
        - Prehensive safety requirement

- Fault tree
  - Deduction -- start with a hazard
  - Inductive -- start with failure
  - Fault tree starts with the failure and works backwards to potential causes

- Risk assessment
  - Classifications
    - Intolerable
    - Admissible
    - Acceptable
  - For each hazard
    - Frequency
    - Severity
    - Estimated risk

- Risk reduction
  - Avoidance
  - Detection and neutralization
  - Damage limitation

- Security Specification
  - Similar to safety
  - Impractical to specify
    - Usually are "shall not"

- Specification

- Requirements specification
  - Functional for error detection and recovery
  - Non functional for reliability and availability
  - Shall not requirements

- Reliability specification
  - Functional
    - Hardware
    - Software
    - Operator
  - Non functional
    - Hardware
    - Software
    - Operator

- Decrease probability of failure
  - For series of dependent components Pt = sum of Pi
  - But if there are n replicated (redundant) and independent components then the Pt = Pa to the nth

- Metrics for reliability
  - FOD probability of failure on demand .001 = 1 on 1000
  - ROCOF Rate of Failure occurrence 2/1000
  - MTTF Mean time to Failure avg time between observed failures 2000 = avg of 1 in 500 time units
  - MTBF Mean time to Failure avg time between observed failures 5000 = avg of 1 in 500 time units

- Non-functional reliability requirements
  - ID type of failure to occur
    - Define the appropriate requirement (metrics)
      - E.g. recoverable with intervention -- POPDE
      - E.g. permanent -- PDCS
## NFRs: With Rational Unified Process and UML

### Home Appliance Control System

**Vision**

**Description**

**Author**

<table>
<thead>
<tr>
<th>Revision History</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
</tr>
<tr>
<td>1.0</td>
</tr>
<tr>
<td>1.1</td>
</tr>
<tr>
<td>1.2</td>
</tr>
<tr>
<td>1.3</td>
</tr>
</tbody>
</table>

### Table of Contents

- PROJECT DRIVERS
  - Purpose
  - Business Opportunity
  - Problem Statement
  - Product Position Statement
- 4. Stakeholder and User Descriptions
  - 4.1 Homeowner
  - 4.2 Stakeholder Summary
  - 4.3 User Environment
  - 4.4 User Summary
- 5. Stakeholder Profiles
  - 5.1 Homeowner
  - 5.2 Business Owner
  - 5.3 Customer Care
  - 5.4 Add a new group of sequences
- 6. User Profiles
  - 6.1 Security
  - 6.2 Usability
  - 6.3 Responsiveness
  - 6.4 Capacity
  - Appendix A: COTS Components

### Table: Functional and Data Requirements

<table>
<thead>
<tr>
<th>Description</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Look and Feel</td>
<td>Easy to use (especially safety related features)</td>
</tr>
<tr>
<td>Cultural and Political</td>
<td>A description of valent features of the product that are related to the way a potential customer will see the product.</td>
</tr>
<tr>
<td>Communication</td>
<td>Interface design may overlap the requirements gathering process.</td>
</tr>
<tr>
<td>Usability and Humanity</td>
<td>This section contains requirements relating to spirit of the interface.</td>
</tr>
<tr>
<td>Accessibility</td>
<td>To ensure that the appearance of the product conforms to the organization's expectations.</td>
</tr>
<tr>
<td>Environment</td>
<td>The product shall comply with corporate branding standards.</td>
</tr>
<tr>
<td>Heritage</td>
<td>The product shall be attractive to a teenage audience.</td>
</tr>
<tr>
<td>Security</td>
<td>The product shall appear authoritative.</td>
</tr>
<tr>
<td>Compatibility</td>
<td>The product shall be in line with corporate branding standards.</td>
</tr>
</tbody>
</table>

### Constraints

- Security
- Look and Feel
- Accessibility
- Usability
- Compatibility
- Environment
- Communication
- Heritage
- Security
- Look and Feel
- Accessibility
- Usability
- Compatibility
- Environment
- Communication
- Heritage

### Functional and Data Requirements

- Look and Feel
  - Usability and Humanity
  - Accessibility
  - Environment
  - Communication
  - Heritage
- Accessibility
  - Usability and Humanity
  - Accessibility
  - Environment
  - Communication
  - Heritage
- Environment
  - Usability and Humanity
  - Accessibility
  - Environment
  - Communication
  - Heritage
- Communication
  - Usability and Humanity
  - Accessibility
  - Environment
  - Heritage
  - Heritage
- Heritage
  - Usability and Humanity
  - Accessibility
  - Environment
  - Communication

### Considerations

- Interface design may overlap the requirements gathering process. This particularly true if you are using prototyping as part of your requirements process. As prototypes develop it is important to capture the requirements that relate to the look and feel. In other words, be sure that you understand your client’s intentions for the product’s look and feel. Record these as requirements instead of merely having a prototype to which the client has added the look and feel. |

### Motivation

- A description of valent features of the product that are related to the way a potential customer will see the product. For example, if your client wants the product to appeal to the business executive, then a look and feel requirement is that the product has a conservative and professional appearance. Similarly, if the product is for sale to children, then the look and feel requirement is that it is colorful and look like it is intended for children. |

### Content

- Given the state of today’s market and potential customers’ expectations, it is often the appearance of products that determines whether they are successful or not...
Non-Functional Requirements

Practices and Recommendations:
A Brief Synopsis

Why

What

- Some Classification Schemes
- NFRs and RE Processes
- Some Individual NFRs
- With Rational Unified Process and UML
- With Volere Requirements Specification Templates

Lawrence Chung

Non-Functional Requirements

What - Essential Concepts

Lawrence Chung

Non-Functional Requirements: With Volere Requirements Specification Template

11 Usability and Humanity Requirements

11a. Ease of use.

Content

This section describes your client’s aspirations for how easy it will be for the intended users of the product to operate it. The product’s usability is derived from the abilities of the expected users of the product and the complexity of its functionality.

The usability requirements should cover such things as:

Efficiency of use - how quickly or accurately the user can use the product.
Ease of remembering - how much is the casual user expected to remember about using the product.
Error rates - for some products it is crucial that the user commits very few, or no, errors.
Overall satisfaction in using the product - this is especially important for commercial, interactive products where there is a lot of competition. Web sites are good examples of this.
Feedback - How much feedback does the user need in order to feel confident that the product is actually accurately doing what the user expects. The necessary degree of feedback will be lower for some products (eg: safety critical) than in others.

Motivation

To guide the product’s designers into building a product that will meet the expectations of its eventual users.

Examples

- The product shall be easy for 11 year-old children to use.
- The product shall help the user to avoid making mistakes.
- The product shall make the users want to use it.
- The product shall be used by people with no training, and possibly no understanding of English.

Fit Criterion

These examples may seem simplistic, but they do express the intention of the client. To completely specify what is meant by the requirement it is necessary to add a measurement of acceptance. We call this a fit criterion. The fit criterion for the above examples would be:

- [An agreed percentage, say 90%] of a test panel of 11 year olds shall be able to successfully complete [list of tasks] within [specified time].
- One month’s use of the product shall result in a total error rate of less than [an agreed percentage, say 2%].
- An anonymous survey shall show that [an agreed percentage, say 75%] of the users are regularly using the product after [an agreed time] familiarization period.

Lawrence Chung

An anonymous survey shall show that [an agreed percentage, say 75%] of the users are regularly using the product.

Lawrence Chung

Lawrence Chung

Lawrence Chung
**NFRs: functional vs. non-functional: a mathematical perspective**

- **(mathematical) function:**
  \[ f(x, y) = f_1(f_2(x), f_3(y)) \]

- **non-functional:**
  \[ nf(x, y) = nf_1(nf_2(x), nf_3(y)) \]
  \[ nf(x, y) = nf_1(nf_2(n(x)), nf_3(n(y))) \]

**Global nature**

Lawrence Chung

---

**NFRs: subjective, graded, interacting**

- **Subjective vs. objective:**
  subjective \[ \Rightarrow \] objective

- **Graded:**
  - worse \[ \Rightarrow \] better
  - expensive \[ \Rightarrow \] cheaper
  - slower \[ \Rightarrow \] faster

- **Interacting:**
  - Conflicting: the whole is less than the sum of its parts
  - Synergistic: the whole is more than the sum of its parts

Lawrence Chung

---

**NFRs: subjective in both definitions & solutions**

**Classification 1** (Roman, IEEE Computer 1985)

- **Interface requirements:**
  - describe how the system is to interact with its environment, users and other systems. E.g., web interfaces and data queries (e.g., web searches)

- **Performance requirements:**
  - throughput: how many users the system handles (e.g., the number of users the system can handle in a given time)

- **Operating requirements:**
  - maintenance: how easy it is to maintain the system (e.g., system absence, user accessibility, ease of use)

- **Lifespan requirements:**
  - quality of the design and code (e.g., reusability, maintenance)

- **Economic requirements:**
  - development cost

- **Political requirements:**
  - privacy considerations

**Classification 2 - Process, Product and External considerations**

[Sommerville 1992]

**Classification 5 - Software Quality Tree**

[Boehm 1976]

**Consider “security” – problem is subjective**

- Protection of data alone, fine with Chris
- Protection of data, and data availability, fine with Pat
- Protection of data, and data availability, and data accuracy, fine with Alex
- Protection of data, and data availability, and data accuracy, and filtering of viruses, fine with Neo
- Protection of data, and data availability, and data accuracy, and filtering of viruses, and blocking adware, fine with Gail

**Consider “security” – solutions are subjective**

- A password authentication fine with Chris
- A password authentication, with periodic change, fine with Pat
- A password, together with a fingerprint verification, fine with Alex
- A password, with a fingerprint verification rechecked every hour, fine with Neo
- A password, with a fingerprint verification rechecked every hour, and co-presence of two people, fine with Gail

Lawrence Chung
Non-Functional Requirements

What - Essential Concepts

- non-functional,
- subjective,
- graded,
- interacting
- and relative
- in both definitions & solutions

- Conflicting: the whole is less than the sum of its parts
- Synergistic: the whole is more than the sum of its parts

- Protection of data alone good
- A password authentication alone bad
- Protection of data alone << Protection of data, and data availability
- A password authentication << A password, together with a fingerprint verification

Lawrence Chung
Non-Functional Requirements

How 1 - Essential Tasks

NFRs:

functional vs. non-functional: a mathematical perspective

- (mathematical) function:
  \[ f_1: I \rightarrow O \]
  \[ f_2: I_1 \times I_2 \rightarrow O \]
  e.g.: sum: \( R \times R \rightarrow R \)
  \[ f(x, y) = f_1(f_2(x), f_3(y)) \]

non-functional:

- How fast can it be done? Fast, Fast(f), Fast(f_2)
- How precise is the answer? Precise, Precise(f), Precise(O)
- How easy is it to figure out how to use it? Easy-to-learn, Easy-to-learn(f), Easy-to-learn(x)
- How robust is the input? Robust, Robust(I_1), Robust(I_2)
- Who can use it? Security, Security(f), Security(O), Security(f_2), Accessibility, Accessibility(f), Accessibility(O)
- Can it be changed easily? Changeability, Changeability(f), Changeability(f_2)
- How much would it cost? Cost, Design-cost(f), Implementation-cost(f), Testing-cost(f)

The NFR Framework

- Based on traditional framework for problem solving in AI [Nilsson]
  - Establish the goals
  - Introduce sub-goals to satisfy the goal where the relationship is AND or OR
    - AND goal is satisfied when all of sub-goals are satisfied
    - OR goal is satisfied when any of the sub-goals are met
    - Continue until you cannot decompose further
  - Softgoal: no clear-cut definition and or criteria as to whether it is satisfied or not, since NFRs are subjective, relative, and interdependent
    - Introduce concept of satisficing
      - Provide basis for saying the softgoal can contribute positively or negatively, fully or partially, to some degree in satisfying other softgoals (i.e., achieved not absolutely but within acceptable limits)
  - Softgoal Interdependency Graphs (SIGs)
    - For modeling non-functional requirements and interdependencies between them
  - Introduces Catalogues of NFRs much like patterns for design are built

Qualitative in nature, Process oriented
**The NFR Framework**

Softgoal Interdependency Graph (SIG)

**Goal types:**
- Non-functional Requirement
- Satisfying Technique (e.g. a design choice)
- Claim (supporting or opposing a choice)

**Contribution Types:**
- AND links (decomposition)
- OR links (alternative)

**Evaluation of goals:**
- Satisfied
- Denied
- Conflicting
- Undetermined

---

**NFRs:** subjective in both *definitions* & solutions

- **Decompose:** Know at least what you mean
- **Operationalize:**
- **Argument:**

---

**Example:** A small portion of a hospital model for requirements analysis:
Lawrence Chung

NFRs:
subjective in both definitions & solutions

Know at least what you mean as precisely as possible - as many decompositions as needed

- Security
  - Integrity
  - Confidentiality
  - Availability
- Performance
  - Time-P
  - Space-P
- Responsive
  - Accuracy
  - Completeness
  - Response-T
  - Throughput

+ Coverage
- Coverage

- Authentication
  - Password
  - Fingerprint
  - Password+
  - Fingerprint

- Indexing
  - Single-level
  - Multi-level
  - Single
  - Multiple
  - Category-based
  - Multiple checks
  - Several different passwords
  - Keyword-based

Operationalize

NFRs:
subjective in both definitions & solutions

Know at least what you mean as precisely as possible - as many decompositions as needed

- Security
  - Integrity
  - Confidentiality
  - Availability
- Performance
  - Time-P
  - Space-P
- Responsive
  - Accuracy
  - Completeness
  - Response-T
  - Throughput

+ Coverage
- Coverage

- Authentication
  - Password
  - Fingerprint
  - Password+
  - Fingerprint

- Indexing
  - Single-level
  - Multi-level
  - Single
  - Multiple
  - Category-based
  - Multiple checks
  - Several different passwords
  - Keyword-based

Operationalize

NFRs:
non-functional …and… functional

Know at least what you mean - decompose
Relate Functional and Non-functional sides

- Security
  - Integrity
  - Confidentiality
  - Availability
- Performance
  - Time-P
  - Space-P
- Responsive
  - Accuracy
  - Completeness
  - Response-T
  - Throughput

+ Coverage
- Coverage

- Authentication
  - Password
  - Fingerprint
  - Password+
  - Fingerprint

- Indexing
  - Single-level
  - Multi-level
  - Single
  - Multiple
  - Category-based
  - Multiple checks
  - Several different passwords
  - Keyword-based

Operationalize
NFRs: non-functional ...and...functional
- Know at least what you mean – decompose
- Relate Functional and Non-functional sides
- Be as specific about the scope/topic/parameter from global to local

NFRs: non-functional ...and...subjective in both definitions & solutions
- Know at least what you mean – decompose
- Relate Functional and Non-functional sides
- Different functional operationalizations contribute differently

NFRs: graded in both definitions and solutions – and relative
- Explore alternatives – some are better/worse than others

NFRs: graded in both definitions and solutions – and relative
- Explore alternatives – some are better/worse than others
- Different alternatives may have different degrees of contributions

NFRs: interacting

- Conflicting: the whole is less than the sum of its parts
- Synergistic: the whole is greater than the sum of its parts

NFRs: interacting – graded and relative

- Different techniques have different impacts (cf. fr-operationalizations)

NFRs: interacting – graded/relative

- Through functional choices (fr-operationalizations)

- Priorities
NFRs: interacting – graded and relative

- Through functional choices
  
  **Prioritize**

```
Claims
Evaluate
```

- **Prioritize**
  - Information security
  - Integrity
  - Confidentiality
  - Availability

```
Home networking
authentication
password
fingerprint
Password+
```

```
Garage Door
Oven
```

**Softgoal Interdependency Graph (SIG):**

**Summary of Modeling Concepts**

- Softgoals: NFR Softgoals, Operationalizing Softgoals, Claim Softgoals
- Contributions:
  - Make >> Help >> Hurt >> Break
  - "Satisficing"

**Proposition = Softgoal U Contribution**

```
AND : Propositions \( \land \) satisfied \( \land \) satisfied \( \land \) satisfied \( \land \) satisfied \( \land \) satisfied \( \land \) satisfied \( \land \) satisfied \( \land \) satisfied

OR : Propositions \( \lor \) denied \( \lor \) denied \( \lor \) denied \( \lor \) denied \( \lor \) denied \( \lor \) denied \( \lor \) denied \( \lor \) denied
```

```
Softgoal Interdenpency Graph (SIG): Semantics
```

Lawrence Chung
Softgoal Interdependency Graph (SIG): Semantics

\[ \text{if } \text{satisficed}(G_0, G_1) \text{ then there exist propositions } G_2, \ldots, G_n \text{ such that} \]
\[ \text{und}(G_0, G_1) = \text{MAKE}(G_0, G_1) \land \text{HELP}(G_0, G_1) \land \text{HURT}(G_0, G_1) \land \text{BREAK}(G_0, G_1) \]

**Non-Functional Requirements**

**How 2 – Dos and Don’ts**

---

**NFRs – Dos & Don’ts**

- **Dos**
  - Relate to FRs
  - Clarify scopetopic
  - Identify agents, whenever useful
  - Discover relationships between definitions of NFRs
  - Discover relationships between solutions to NFRs
  - Refine definitions as many times as needed
  - Refine solutions as many times as needed
  - Prioritize
  - Discover conflicts
  - Safeguard against conflicts
  - Discover synergies
  - Discover operationalizations as reasons for conflicts/synergies
  - Determine strengths of contributions
  - Justify strengths of contributions
  - Explore alternatives
  - Discover solutions from requirements
  - Discover requirements from solutions
  - Consider use of multiple solutions
  - Consider scenarios
  - If necessary, quantify
  - Evaluate, ...subjectively, ...objectively Lawrence Chung

- **Don’ts**
  - Absolute security, absolute reliability, absolute safety...
  - One definition fits all
  - One solution solves all problems
  - The contribution is such and such, since I say so
  - Refine the definition only once
  - They are falling down from the sky
  - Dissociate from FRs
  - May be more important than FRs, but should consume less resources
  - You name it; our system does it
  - No quantification, no existence
  - Everybody needs the same
  - Be only pessimistic
  - Asking why + reveals ignorance
  - Beg the question
  - Evaluate & only evaluate
  - Brainwash nothing but objectivity
Conflict resolution 1

- Delete email w. any zip file attachment
  - misunderstanding betw. sender and receiver
    - move email w. any zip file attachment into a junk file folder
      - if the receiver does not check the junk file folder, still misunderstanding
        - at the time the file is moved, notify this to the receiver
          - if the receiver still does not check the junk file folder or checks it late, still misunderstanding
            - at the time the file is moved, notify the sender too
              - if the receiver checks the junk file folder and opens it and the file is an attack, still a security breach
              - Delete email w. any zip file attachment and block any future email from the same sender

Lawrence Chung

Conflict resolution 2

- If the receiver opens email w. zip file and the file is an attack, a security breach
- Delete email w. any zip file attachment
  - misunderstanding betw. sender and receiver
    - move email w. any zip file attachment into a junk file folder
      - if the receiver does not check the junk file folder, still misunderstanding
        - at the time the file is moved, notify this to the receiver
          - if the receiver still does not check the junk file folder or checks it late, still misunderstanding
            - at the time the file is moved, notify the sender too
              - if the receiver checks the junk file folder and opens it and the file is an attack, still a security breach
              - Delete email w. any zip file attachment and block any future email from the same sender

Conflict resolution 3

- Security(PC) -> S(email) -> S(sender) ^ S(recipient) ^ S(body) ^ S(attachment)
- Denied (S[attachment]) -> denied (S[email]) -> denied (S[PC])
- Zip(attachment) ^ attack(attachment) ^ open(attachment) -> denied (S[attachment])
  - if the receiver opens email w. zip file and the file is an attack, a security breach
  - Zip(attachment) v attack(attachment) v open(attachment) -> denied (S[attachment]) helps denied(S[email])
- Delete email w. any zip file attachment
  - misunderstanding betw. sender and receiver
    - move email w. any zip file attachment into a junk file folder
      - if the receiver does not check the junk file folder, still misunderstanding
        - at the time the file is moved, notify this to the receiver
          - if the receiver still does not check the junk file folder or checks it late, still misunderstanding
            - at the time the file is moved, notify the sender too
              - if the receiver checks the junk file folder and opens it and the file is an attack, still a security breach
              - Leave the email but delete the attachment only
                - Leave the email but delete the attachment only if it is an attack
                - Leave the email but change the name of the attachment to “…renameToZip”

Conflict resolution 4

- Security(PC) -> S(email) -> S(sender) ^ S(recipient) ^ S(body) ^ S(attachment)
- Denied (S[attachment]) -> denied (S[email]) -> denied (S[PC])
- Zip(attachment) ^ attack(attachment) ^ open(attachment) -> denied (S[attachment])
  - if the receiver opens email w. zip file and the file is an attack, a security breach
  - Zip(attachment) v attack(attachment) v open(attachment) -> denied (S[attachment]) helps denied(S[email])
- Delete email w. any zip file attachment, at the time of reception
  - misunderstanding betw. sender and receiver
    - move email w. any zip file attachment into a junk file folder
      - if the receiver does not check the junk file folder, still misunderstanding
        - at the time the file is moved, notify this to the receiver
          - if the receiver still does not check the junk file folder or checks it late, still misunderstanding
            - at the time the file is moved, notify the sender too
              - if the receiver checks the junk file folder and opens it and the file is an attack, still a security breach
              - Leave the email but delete the attachment only
                - Leave the email but delete the attachment only if it is an attack
                - Leave the email but change the name of the attachment to “…renameToZip”

Lawrence Chung
NFRs – Where

Wherever better/cheaper/faster/happier matters

- Requirements Engineering
- System Architecting
- Software Architecting
- Design
- Implementation
- Validation & Verification
- Testing
- Maintenance
- Software Process
- Project Planning and Management
- Configuration Management
- Decision making

Lawrence Chung

NFRs – How to represent

- Bring in FRs
- Clarify scope/topics
- Identify agents, whenever useful
- Discover relationships between definitions of NFRs
- Discover relationships between solutions to NFRs
- Refine definitions as many times as needed
- Prioritize
- Discover conflicts
- Safeguard against conflicts
- Discover synergies
- Discover operationalizations as reasons for conflicts/synergies
- Determine strengths of contributions
- Justify strengths of contributions
- Explore alternatives
- Discover solutions from requirements
- Discover requirements from solutions
- Consider use of multiple solutions
- Consider scenarios
- If necessary, quantify
- Evaluate
- Evaluate subjectivity
- Evaluate objectively
- Establish traceability

Lawrence Chung

NFRs: With Rational Unified Process and UML

Home Appliance Control System
Vision
Revision History

Table of Contents

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Introduction</td>
<td>5</td>
</tr>
<tr>
<td>1.1</td>
<td>Purpose</td>
<td>5</td>
</tr>
<tr>
<td>1.2</td>
<td>Scope</td>
<td>5</td>
</tr>
<tr>
<td>1.3</td>
<td>Definitions, Acronyms, and Abbreviations</td>
<td>5</td>
</tr>
<tr>
<td>1.4</td>
<td>References</td>
<td>5</td>
</tr>
<tr>
<td>2.</td>
<td>Positioning</td>
<td>5</td>
</tr>
<tr>
<td>2.1</td>
<td>Business Opportunity</td>
<td>5</td>
</tr>
<tr>
<td>2.2</td>
<td>Business Overview</td>
<td>5</td>
</tr>
<tr>
<td>2.3</td>
<td>Product Position Statement</td>
<td>5</td>
</tr>
<tr>
<td>3.</td>
<td>Stakeholder and User Descriptions</td>
<td>6</td>
</tr>
<tr>
<td>3.1</td>
<td>Market Demographics</td>
<td>6</td>
</tr>
<tr>
<td>3.2</td>
<td>Stakeholder Summary</td>
<td>6</td>
</tr>
<tr>
<td>3.3</td>
<td>User Summary</td>
<td>6</td>
</tr>
<tr>
<td>3.4</td>
<td>User Environment</td>
<td>7</td>
</tr>
<tr>
<td>3.5</td>
<td>Stakeholder Profiles</td>
<td>7</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Homeowner</td>
<td>7</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Business Owner</td>
<td>8</td>
</tr>
<tr>
<td>3.5.3</td>
<td>Customer Care</td>
<td>8</td>
</tr>
<tr>
<td>3.6</td>
<td>Use Profiles</td>
<td>9</td>
</tr>
<tr>
<td>3.7</td>
<td>Key Stakeholders or Use Needs</td>
<td>9</td>
</tr>
</tbody>
</table>

Lawrence Chung

Appendix

- RUP Specification
- Volere Specification
- How to Augment UML

Lawrence Chung
NFRs: With Rational Unified Process and UML

<table>
<thead>
<tr>
<th>Product Overview</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Product Overview</td>
<td>9</td>
</tr>
<tr>
<td>4.2 Summary of Capabilities</td>
<td>10</td>
</tr>
<tr>
<td>4.3 Assumptions and Dependencies</td>
<td>10</td>
</tr>
<tr>
<td>4.4 Design Philosophy</td>
<td>11</td>
</tr>
<tr>
<td>4.5 Licensing and Installation</td>
<td>11</td>
</tr>
<tr>
<td>5. Product Features</td>
<td>11</td>
</tr>
<tr>
<td>5.1 Start system</td>
<td>11</td>
</tr>
<tr>
<td>5.2 Shutdown system</td>
<td>11</td>
</tr>
<tr>
<td>5.3 View status of system</td>
<td>11</td>
</tr>
<tr>
<td>5.4 High a new group of sequences</td>
<td>12</td>
</tr>
<tr>
<td>5.5 Modify an existing group of sequences</td>
<td>12</td>
</tr>
<tr>
<td>5.6 Delete an existing group of sequences</td>
<td>12</td>
</tr>
<tr>
<td>5.7 Categorize a group</td>
<td>12</td>
</tr>
<tr>
<td>5.8 Schedule a group</td>
<td>12</td>
</tr>
<tr>
<td>5.9 Start a group</td>
<td>12</td>
</tr>
<tr>
<td>5.10 Stop a group</td>
<td>12</td>
</tr>
<tr>
<td>5.11 View the status of whole system</td>
<td>12</td>
</tr>
<tr>
<td>5.12 View the status of indoor lights</td>
<td>12</td>
</tr>
<tr>
<td>5.13 View the status of outdoor lights</td>
<td>12</td>
</tr>
<tr>
<td>5.14 View the status of entertainment equipment (radios, CD players, televisions)</td>
<td>12</td>
</tr>
<tr>
<td>5.15 View the status of the safety system</td>
<td>12</td>
</tr>
<tr>
<td>5.16 View the status of the security system</td>
<td>12</td>
</tr>
<tr>
<td>5.17 Make a new sequence</td>
<td>13</td>
</tr>
<tr>
<td>5.18 Modify an existing sequence</td>
<td>13</td>
</tr>
<tr>
<td>5.19 Delete an existing sequence</td>
<td>13</td>
</tr>
<tr>
<td>5.20 Schedule a sequence</td>
<td>13</td>
</tr>
<tr>
<td>5.21 Start a sequence</td>
<td>13</td>
</tr>
<tr>
<td>5.22 Stop a sequence</td>
<td>13</td>
</tr>
</tbody>
</table>

6. Constraints

| 6.1 Security | 14 |
| 6.2 Usability | 15 |
| 6.3 Responsiveness | 15 |
| 6.4 Capacity | 15 |

5.29 Schedule a sequence | 13 |
5.30 Start a sequence | 13 |
5.31 Stop a sequence | 13 |
5.32 Turn on outdoor lights (all) | 13 |
5.33 Turn off outdoor lights (all) | 13 |
5.34 Turn on selected outdoor lights | 13 |
5.35 Make a new sequence | 13 |
5.36 Modify an existing sequence | 13 |
5.37 Delete an existing sequence | 13 |
5.38 Schedule a sequence | 13 |
5.39 Start a sequence | 13 |
5.40 Stop a sequence | 13 |
5.41 Turn on radios, CD players, television (all) | 13 |
5.42 Turn off radios, CD players, television | 13 |
5.43 Turn on selected radio, CD player, television | 13 |
5.44 Turn off selected radio, CD player, television | 13 |
5.45 Automatic notification of emergency | 14 |
5.46 Make a new sequence | 14 |
5.47 Modify an existing sequence | 14 |
5.48 Delete an existing sequence | 14 |
5.49 Schedule a sequence | 14 |
5.50 Start a sequence | 14 |
5.51 Stop a sequence | 14 |
5.52 Turn on security system (all features) | 14 |
5.53 Turn off security system (all features) | 14 |
5.54 Turn on safety system (all features) | 14 |
5.55 Turn off safety system (all features) | 14 |
5.56 Turn on selected features of security system | 14 |
5.57 Turn off selected features of security system | 14 |
5.58 Turn on selected features of safety system | 14 |
5.59 Turn off selected features of safety system | 14 |
The usability requirements should cover such things as:

- Overall satisfaction in using the product - this is especially important for commercial, interactive products where there is a need to keep the user interested and engaged.

- Error rates - for some products it is crucial that the user commits very few, or no, errors. For example, a product that involves financial transactions must be absolutely error-free.

- Feedback - how much feedback does the user need in order to feel confident that the product is actually accurately doing what the user expects. The necessary degree of feedback will be higher for some products (e.g. safety-critical) than in others. For example, a medical device must provide clear and immediate feedback to avoid errors.

- Personal configuration options - there are a myriad of these which can be specified by the user. Consider the locations of the potential customers and users of your product. Any out of country users will participate more closely with your organization, as well as giving them their own personal user experience. Examples include:
  - Currencies including the symbols and decimal conventions
  - Languages, spelling preferences, language idioms
  - Personal preferences or choice of language. The personalization requirements should cover such things as:

- Efficiency of use - how quickly or accurately the user can use the product.

- Ease of use.

- This section describes your client's aspirations for how easy it will be for the intended users of the product to operate it. The product's usability is derived from the abilities of the expected users of the product and the complexity of its functionality.

- Feedback - how much feedback does the user need in order to feel confident that the product is actually accurately doing what the user expects. The necessary degree of feedback will be higher for some products (e.g. safety-critical) than in others. For example, a medical device must provide clear and immediate feedback to avoid errors.

- The section contains requirements relating to spirit of the interface. Your client may have given you particular demands such as corporate branding, style, colors to be used, degree of interaction and so on. This section captures the requirements for the interface rather than the design for the interface.

- Interface:

  - To ensure that the appearance of the product conforms to the organization's expectations.
  - The product shall comply with corporate branding standards. The product shall be attractive to a teenage audience.
  - The product shall appear authoritative.

- Considerations:

  - Interface design may overlap the requirements gathering process. This particularly true if you are using prototyping as part of your requirements process. An prototype developed is important to capture the requirements that relate to the look and feel. In other words, be sure that you understand your client's expectations for the product's appearance and feel.

- Interface design:

  - To ensure that the appearance of the product conforms to the organization's expectations. Example:

    - The product shall be attractive to a teenage audience.
    - The product shall appear authoritative.

- Motivation:

  - To guide the product's designers into building a product that will meet the expectations of its eventual users.

- Examples:

  - This product shall be easy for 11 year old children to use.
  - This product shall help the user avoid making mistakes.
  - This product needs the user to use it.

- Considerations:

  - This section describes the way in which the product can be altered or configured to take into account the user's personal preferences or choice of language. The personalization requirements should cover such things as:

    - Corporate branding requirements such as fonts, colors, etc.
    - User documentation and training requirements.
    - Hardware and software interface design considerations. This particularly true if you are using prototyping as part of your requirements process. An prototype developed is important to capture the requirements that relate to the design of the interface. This section captures the requirements for the interface rather than the design for the interface.

- Considerations:

  - This section describes the way in which the product can be altered or configured to take into account the user's personal preferences or choice of language. The personalization requirements should cover such things as:

    - Corporate branding requirements such as fonts, colors, etc.
    - User documentation and training requirements.
    - Hardware and software interface design considerations. This particularly true if you are using prototyping as part of your requirements process. An prototype developed is important to capture the requirements that relate to the design of the interface. This section captures the requirements for the interface rather than the design for the interface.
**NFRs: With Volere Requirements Specification Template**

**Content**

A statement of how easy it should be to learn to use the product. This will range from zero time for products intended for placement in the public domain (for example, a parking meter or a web site) to a considerable time for complex, highly technical products. (We know of one product where it was necessary for graduate engineers to spend 18 months in training before being qualified to use the product.)

**Motivation**

To quantify the amount of time that your client feels is allowable before a user can successfully use the product. To avoid forcing the user to learn terms and concepts that are part of the product’s internal construction and are not relevant to the users’ world. To make the product more comprehensible and thus more likely to be adopted by its intended users.

**Examples**

- The product shall be easy for an engineer to learn.
- The product shall be used by engineers who will attend 5 weeks of training before using the product.

**Fit Criteria**

Fit criterion for the above example requirements are:

- The project shall be easy for an engineer to learn.
- A stark shall be able to be productive within a short time.
- The product shall be able to be used by members of the public who will receive no training before using it.
- The product shall be used by engineers who will attend 5 weeks of training before using the product.

**NFRs: With Volere Requirements Specification Template**

**Examples**

Any interface between a user and the automated system shall have a maximum response time of 2 seconds.

The response shall be fast enough to avoid interrupting the user’s flow of thought.

The product shall poll the sensor every 10 seconds.

The product shall achieve (specified percentage) pass rate from the final examination of the training.

**Fit Criterion - Unit of measurement**

Required range of values

**Examples**

- The product shall not emit noxious gases that damage people’s health.
- The product shall hide the details of its construction from the user.
- The product shall be usable by partially sighted users.

**NFRs: With Volere Requirements Specification Template**

**Examples**

- The product shall be expected to operate within the maximum management budget for a minimum 5 years.
- The product shall be expected to operate within the maximum maintenance budget for a minimum 5 years.

**NFRs: With Volere Requirements Specification Template**

**Examples**

- The product shall continue to operate in local mode whenever it loses its link to the central server.
- It should provide 10 minutes of emergency operation should it become disconnected from the electricity source.

**NFRs: With Volere Requirements Specification Template**

**Examples**

- The product shall be expected to operate within the maximum maintenance budget for a minimum 5 years.
- The product shall be expected to operate within the maximum management budget for a minimum 5 years.

**NFRs: With Volere Requirements Specification Template**

**Examples**

- The product shall be expected to operate within the maximum maintenance budget for a minimum 5 years.
- The product shall be expected to operate within the maximum management budget for a minimum 5 years.
NFRs: With Volere Requirements Specification Template

13 Operational Requirements
13a. Expected physical environment
   Examples:
   - The product shall be used by a worker, standing up, outside in cold, rainy conditions.
   - The product shall be used in noisy conditions with a lot of dust.

13b. Expected technological environment
13c. Partner applications
   Examples:
   - We must be able to interface with any HTML browser.

13d. Productization Requirements
   Examples:
   - The product shall be distributed as a ZIP file.
   - The product shall be able to be installed by an untrained user without recourse to separately printed instructions.

NFRs: With Volere Requirements Specification Template

14 Maintainability and Support Requirements
14a. Maintenance Requirements
   Examples:
   - New MIS reports must be available within one working week of the date the requirements are agreed.
   - A new weather station must be able to be added to the system overnight.

14b. Are there special conditions that apply to the maintenance of this product?
   Examples:
   - The maintenance releases will be offered to end-users once a year.
   - Only registered user will have access to our help site via the Internet.

14c. Supportability Requirements
14d. Adaptability requirements
   Examples:
   - The maintenance releases will be offered to end-users once a year.
   - The product shall make all functionality available to the managing director.

14e. Installation requirements
   Examples:
   - The product shall be able to be installed in the specified environment within 2 working days.

NFRs: With Volere Requirements Specification Template

15 Security Requirements
15a. Access requirements
   Examples:
   - Only direct managers can see the personnel records of their staff.
   - Only holders of current security clearance can enter the building.

15b. Integrity requirements
   Examples:
   - The product shall prevent key data from being introduced.
   - The product shall protect itself from internal abuse.

15c. Privacy requirements
   Examples:
   - The product shall notify customers of changes to its information policy.

15d. Audit requirements
15e. Immunity requirements
   Examples:
   - The requirements for which the product has to do to protect itself from infection by unauthorized or undesirable software programs, such as viruses, worms, Trojan horses and others.

NFRs: With Volere Requirements Specification Template

16 Cultural and Political Requirements
16a. Cultural requirements
   Examples:
   - The product shall not be offensive to religious or ethnic groups.
   - The product shall be able to distinguish between French, Italian and British road numbering systems.

16b. Political requirements
   Examples:
   - The product shall be developed using XYZ standards.

Considerations:
Did you intend to develop the product on a Macintosh, when the office manager has laid down a edict that only Windows machines are permitted?
Is a director also on the board of a company that manufactures products similar to the one that you intend to build?
Whether you agree with these political requirements has little bearing on the outcome. The reality is that the system has to comply with political requirements even if you can find a better/more efficient/more economical solution.
A few probing questions here may save some heartache later. The political requirements might be purely concerned with the politics inside your organisation. However there are situations where you need to consider the politics inside your customer’s organisations or the national politics of the country.
NFRs:
With Volere Requirements Specification Template

17 Legal Requirements
17a. Compliance requirements
Examples:
- Financial information shall be implemented so as to comply with the data protection act.
- Lawyers' opinion that the product does not break any laws.

Considerations:
- Consider consulting lawyers to help identify the legal requirements.
- Are there any copyrights/intellectual property that must be protected? Alternatively, do any competitors have copyrights that you might be in danger of infringing?

17b. Standards requirements
Example:
The product shall comply with MilSpec standards.
The product shall comply with insurance industry standards.
The product shall be developed according to SSADM standard development steps.

Fit Criterion:
The appropriate standard-keeper certifies that the standard has been adhered to.
Considerations:
- It is not always apparent that there are applicable standards because their existence is often taken for granted. Consider the following:
  - Are there any industry bodies that have applicable standards?
  - Has the industry a code of practice, watchdog or ombudsman?
  - Are there any special development steps for this type of product?

Lawrence Chung

NFRs:
With Rational Unified Process and UML

17 Legal Requirements
17a. Compliance requirements
Examples:
- Personal information shall be implemented so as to comply with the data protection act.

Considerations:
- Consider consulting lawyers to help identify the legal requirements.
- Are there any copyrights/intellectual property that must be protected? Alternatively, do any competitors have copyrights that you might be in danger of infringing?

17b. Standards requirements
Example:
The product shall comply with MilSpec standards.
The product shall comply with insurance industry standards.
The product shall be developed according to SSADM standard development steps.

Fit Criterion:
The appropriate standard-keeper certifies that the standard has been adhered to.
Considerations:
- It is not always apparent that there are applicable standards because their existence is often taken for granted. Consider the following:
  - Are there any industry bodies that have applicable standards?
  - Has the industry a code of practice, watchdog or ombudsman?
  - Are there any special development steps for this type of product?

Lawrence Chung

NFRs:
With Rational Unified Process and UML

What Are Use Cases?

System
Actor
Generalized Actor
Specialized Actor

Use Case
Specialized Use Case
Generalized Use Case
Actor-Use Case Association

System in question
external entity
Use case
Use case specializations
Actor-Use Case Association

Textual description for NFRs embedded in the use case special requirements section – not 1st class citizens

Inadequate Handling of NFRs

Problems:

Textual description for NFRs embedded in the use case special requirements section – not 1st class citizens

Lawrence Chung
### NFRs:

**With Rational Unified Process and UML**

<table>
<thead>
<tr>
<th>Author(s)</th>
<th>Method</th>
<th>Alternative Constructs</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimitrov’s [4]</td>
<td>Use cases</td>
<td>Use cases, Sequence diagram, State chart, Activity diagram</td>
<td>Using the existing use case modeling for FR elicitation</td>
</tr>
<tr>
<td>Moreira’s [3]</td>
<td>Use cases</td>
<td>Use cases</td>
<td>Using use cases (FR constructs) to represent NFRs. No organizational constructs.</td>
</tr>
<tr>
<td>Lee’s [2]</td>
<td>Not using the preferred use case modeling for FR elicitation</td>
<td>Use cases, Class/ERD extensions</td>
<td>Not using the preferred use case modeling for FR elicitation</td>
</tr>
<tr>
<td>Cysnerios’s [1]</td>
<td>Drawbacks</td>
<td>NFR Modeling Constructs Integration Point Method</td>
<td>No single scheme providing all of:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Use case driven</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Modeling constructs for representing and organizing NFRs</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Preserving underlying use case principles (e.g., ovals for FRs but not for NFRs)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>General: for a wide range of NFRs</td>
</tr>
</tbody>
</table>

### Table Notes:

- Use case driven
- Modeling constructs for representing and organizing NFRs
- Preserving underlying use case principles (e.g., ovals for FRs but not for NFRs)
- General: for a wide range of NFRs