Week2: Introduction to Requirement Engineering

Dr. Rasha Kashef

Previous Week

- The Software…
- Software Engineering
- The software problem
- Software Development Process
- Software Process Models

Agenda

- Definition and Importance of Requirements
- Requirement Engineering activities
- Types of Requirements
- Requirements Quality
- Business Examples
- Modeling

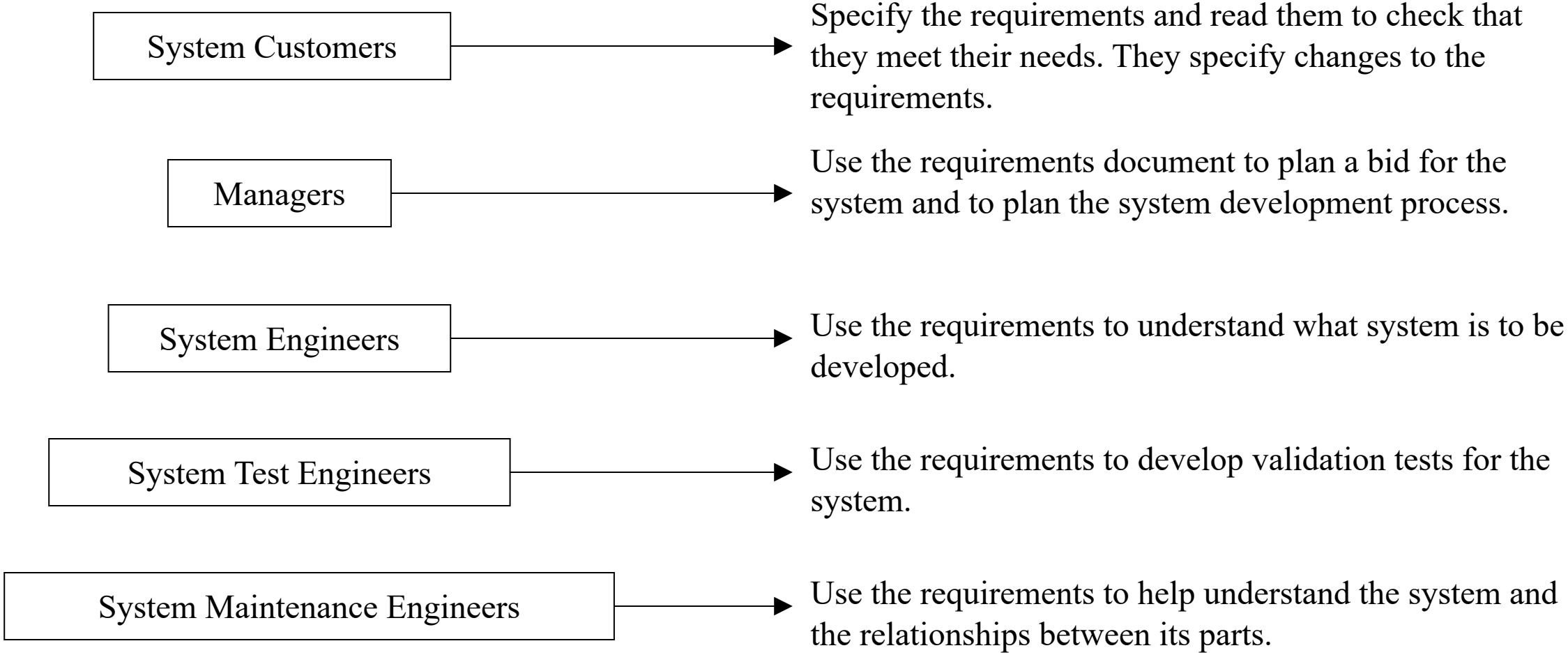
Requirements

- A requirement is:
  - A requirement is a simple statement of what the system must do or what characteristics does it have.
  - Capturing the purpose of a system
  - An expression of the ideas to be embodied in the system or application under development
  - A statement about the proposed system that all stakeholders agree must be made true in order for the customer's problem to be adequately solved
    - Short and concise piece of information
    - Says something about the system
    - All the stakeholders have agreed that it is valid
    - It helps solve the customer's problem

Requirements in IEEE 830-1993

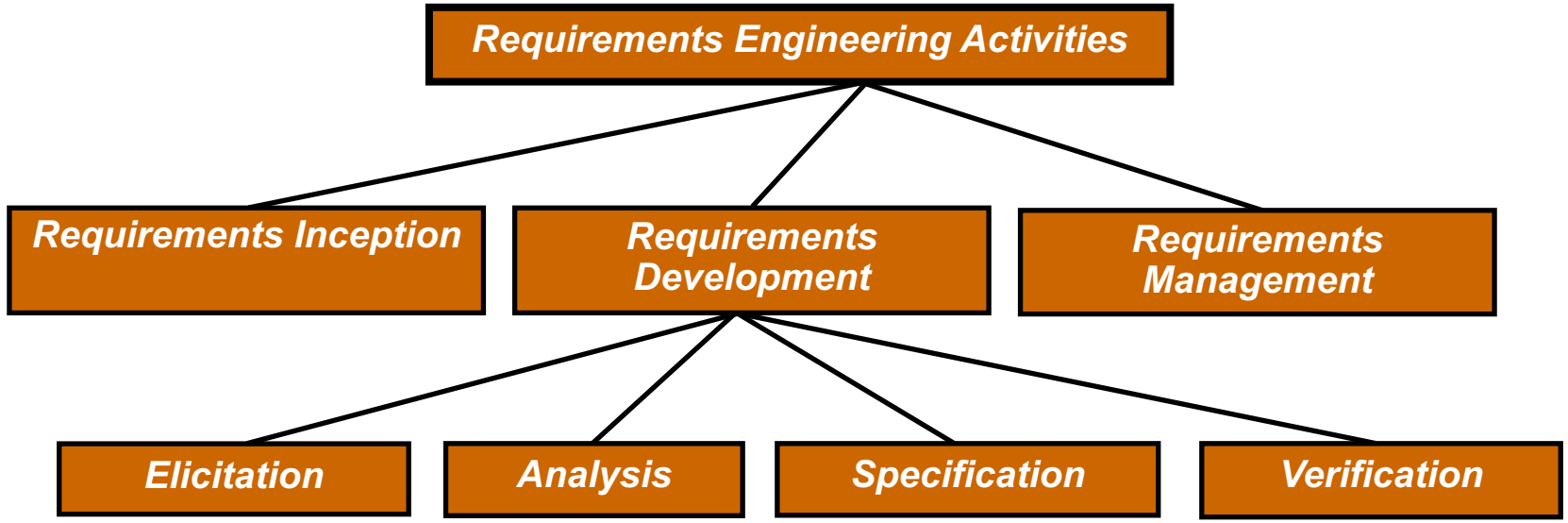
- A requirement is defined as:
  - A condition or capability needed by a user to solve a problem or achieve an objective
  - A condition or a capability that must be met or possessed by a system … to satisfy a contract, standard, specification, or other formally imposed document …

Users of the Requirements (Sommerville, 2000)



Requirement Engineering

- Requirements Engineering (RE) is:
  - **The activity of development, elicitation, specification, analysis, and management of the stakeholder requirements, which are to be met by a new or evolving system**
  - **RE is concerned with identifying the purpose of a software system… and the contexts in which it will be used**



Source: Larry Boldt, Trends in Requirements Engineering People-Process-Technology, Technology Builders, Inc., 2001

Requirement Engineering Activities

- **Inception:** Start the process (identification of business need, new market opportunity, great ideas)
- **Requirements elicitation:** Requirements discovered through consultation with stakeholders
- **Requirements analysis and negotiation:** Requirements are analyzed and conflicts resolved
- **Requirements specification:** A precise requirements document is produced
- **Requirements validation:** The requirements document is checked for consistency and completeness
- **Requirements management:** Needs and contexts evolve, and so do requirements!

1. Requirement Inception

- Start the process
  - Identification of business need
  - New market opportunity
  - Great idea
- Involves
  - Building a business case
  - Preliminary feasibility assessment
  - Preliminary definition of project scope
- Stakeholders
  - Business managers, marketing people, product managers...
- Examples of techniques
  - Brainstorming, Joint Application Development (JAD), meeting…

2. Requirement Elicitation (1)

- Gathering of information
  - About problem domain
  - About problems requiring a solution
  - About constraints related to the problem or solution
- Questions that need to be answered
  - What is the system?
  - What are the goals of the system?
  - How is the work done now?
  - What are the problems?
  - How will the system solve these problems?
  - How will the system be used on a day-to-day basis?
  - Will performance issues or other constraints affect the way the solution is approached?

2. Requirement Elicitation (2)

- Why is Requirements Elicitation hard?
  - Customers / Users are not always good at describing what they want or need
  - Software Engineers are not always good at understanding someone else's concerns
  - In certain application domains, software engineers and customers have completely different backgrounds and use a different terminology
  - Volatility: requirements change over time.
- Overview of different techniques
  - Brainstorming: Moderated meeting with trigger questions
  - Interviews: Ask specific details: boundaries, exceptions, anticipated changes, vision of future, alternatives, etc
  - Task observations: Observe users at work, Obtain subtle information not told by customer (forgotten, didn't think it was important, didn't understand implication)
  - Use cases / scenarios: include all the use-cases and various scenarios of the systems components.
  - Prototyping, Questionnaire and More…

3. Requirement Analysis

- The process of studying and analyzing the needs of stakeholders (e.g., customer, user) in view of coming up with a “solution”. Such a solution may involve:
  - A new organization of the workflow in the company.
  - A new system (system-to-be, also called solution domain) which will be used in the existing or modified workflow.
  - A new software to be developed which is to run within the existing computer system or involving modified and/or additional hardware.
- Objectives
  - Detect and resolve conflicts between requirements (e.g., through negotiation)
  - Discover the boundaries of the system / software and how it must interact with its environment
  - Elaborate system requirements to derive software requirements

4. Requirement Specifications

- The invention and definition of the behavior of a new system (solution domain) such that it will produce the required effects in the problem domain
- A requirement specification is a **description** of a software system to be developed.
- Requirements Analysis has defined the problem domain and the required effects
- Software Requirements Specification Document (SRS)
  - A document that clearly and precisely describes, each of the essential requirements (functions, performance, design constraints, and quality attributes) of the software and the external interfaces

- It may include a set of use cases that describe user interactions that the software must provide.
- Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on what the software product is to do as well as what it is not expected to do.
- Software requirements specification permits a rigorous assessment of requirements before design can begin and reduces later redesign.

5. Requirement Validation and verification

- Validation and verification
  - Both help ensure delivery of what the client wants
  - Need to be performed at every stage during the process
- Validation: checks that the right product is being built (refers back to stakeholders)
- Verification: checks that the product is being built right
  - During design phase: refers back to the specification of the system or software requirements
  - During RE: mainly checking consistency between different requirements, detecting conflicts
- Techniques used during RE
  - Simple checks
  - Formal Review
  - Logical analysis
  - Prototypes and enactments
  - Design of functional tests
  - Development of user manual

6. Requirements Management

- Necessary to cope with changes to requirements
- Requirements change is caused by:
  - Business process changes
  - Technology changes
  - Better understanding of the problem
- Traceability is very important for effective requirements management

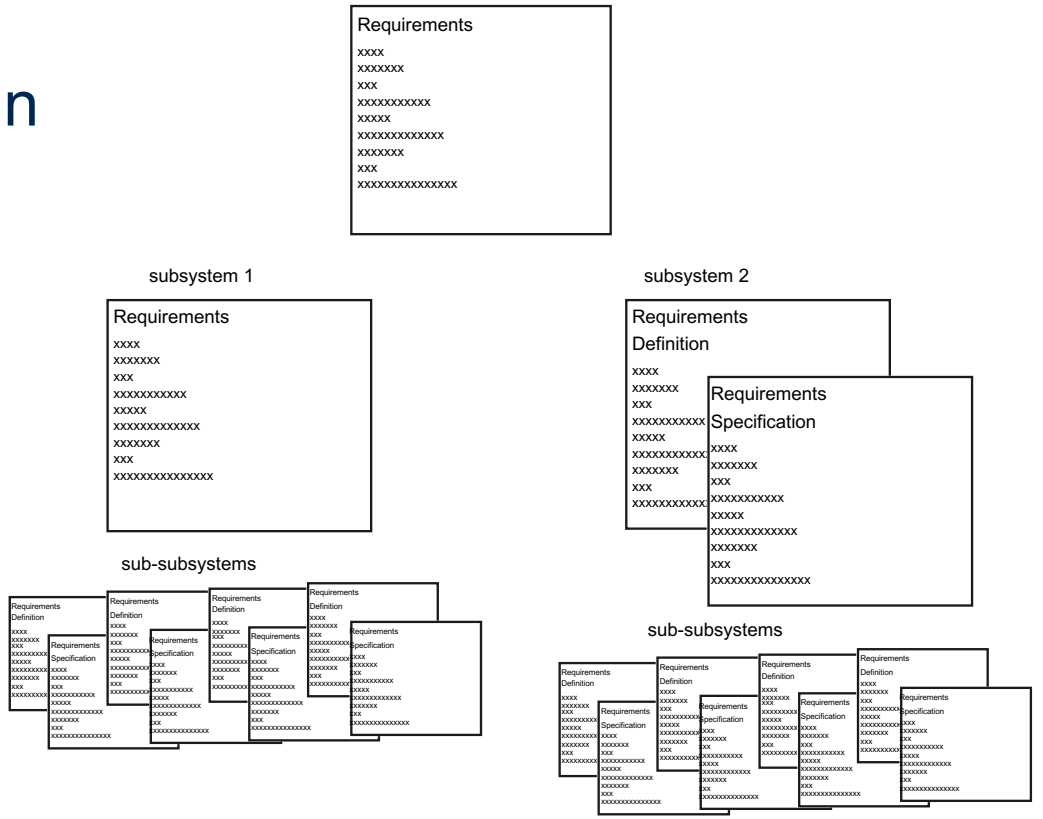
Requirements Documents

- Vision and Scope Document
- Elicitation notes: (Raw) requirements obtained through elicitation; often unstructured, incomplete, and inconsistent
- (Problem domain) Requirements document
- System requirements document
- Software requirements document
  - The software is normally part of a system that includes hardware and software. Therefore, the software requirements are normally part of the system requirements.

Types of Requirements Documents

Two extremes:

- An informal outline of the requirements using a few paragraphs or simple diagrams
  - This is called the requirements definition
- A long list of specifications that contain thousands of pages of intricate requirements describing the system in detail
  - This is called the requirements specification
- Requirements documents for large systems are normally arranged in a hierarchy



Challenges in the Requirements Process

- Lack of the right expertise (software engineers, domain experts, etc.)
- Initial ideas are often incomplete, wildly optimistic, and firmly entrenched in the minds of the people leading the acquisition process
- Difficulty of using complex tools and diverse methods associated with requirements gathering may negate the anticipated benefits of a complete and detailed approach

Statistics from NIST Report

- NIST (National Institute of Standards and Technology) has published a comprehensive (309 pages) and very interesting report on project statistics and experiences based on data from a large number of software projects[1]
  - 70% of the defects are introduced in the specification phase
  - 30% are introduced later in the technical solution process
  - Only 5% of the specification inadequacies are corrected in the specification phase
  - 95% are detected later in the project or after deliver,y where the cost for correction on average is 22 times higher compared to a correction directly during the specification effort
  - The NIST report concludes that extensive testing is essential, however testing detects the dominating specification errors late in the process.

Requirements Categories

- During a systems development project, requirements will describe
  - what the business needs (**business requirements**);
  - what the users need to do (**user requirements**);
  - what the software should do (**functional requirements**);
  - The list of characteristics the system should have (**non-functional requirements**);

Business Requirements

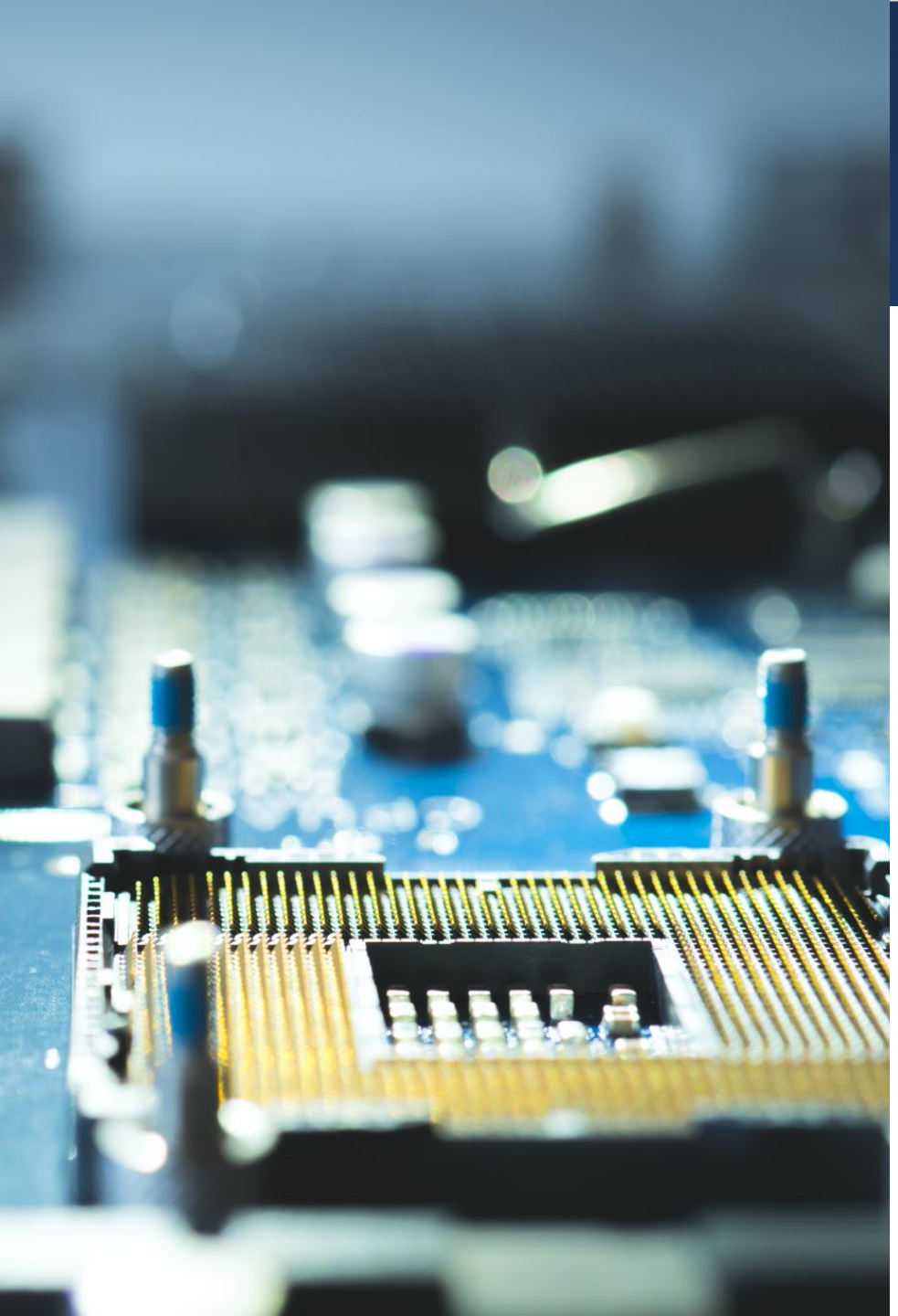
- Business requirements are **what** must be delivered to provide **value**.
  - e.g. “Increase market share”; “Shorten order processing time”; “Reduce customer service costs”; “Lower inventory spoilage”; “Improve responsiveness to customer service requests”; “Provide account access to mobile customers.”
- When the systems development project is complete, success will be measured by **evaluating** whether the stated business requirements have actually been **achieved**.

User Requirements

- User requirements specify **what** the user expects the system to be able to perform.
- It describes tasks that the users perform as an integral part of the business' operations, such as:
  - "Determine available credit"; and "Look up account balances."
  - "Schedule a client appointment";
  - "Place a new customer order";
  - "Re-order inventory";

Functional Requirements

- A functional requirement (FR)
  - is a requirement defining functions of the system under development
  - Describes what the system should do
  - **Defines** how the system will support the user requirements.
  - A functional requirement **relates** directly to a ***process*** the system has to perform



Functional Requirements (FR)

- What inputs the system should accept
- What outputs the system should produce
- What data the system should store other systems might use
- What computations the system should perform
- The timing and synchronization of the above
- The user shall be able to search either all of the initial set of databases or select a subset from it.
- The system shall provide appropriate viewers for the user to read documents in the document store.
- Every order shall be allocated a unique identifier (ORDER_ID) which the user shall be able to copy to the account's permanent storage area.
- add a new entry for an appointment type, including patient name with up to 30 alpha characters and duration in the unit of minutes;
- modify the duration in an existing entry.

Non-Functional Requirements

- A non-functional requirement (NFR) is a requirement that is not functional. This requirement category defines some important behavioral properties or **characteristics** (quality attributes) that the system must have, such as **performance** and **scalability**.
- Non-functional requirements are important
  - If they are not met, the system is useless
  - Non-functional requirements may be very difficult to state precisely (especially at the beginning) and imprecise requirements may be difficult to verify
- They are sometimes called quality requirements, quality of service, or extra-functional requirements.
- Many will be discovered during conversations with users in the analysis phase.
- These requirements do not describe business processes or information, but they are very important in understanding what the final system should be like.
- Note: All requirements must be verifiable (by some test, inspection, audit etc.)

Non-Functional Requirements (NFR) Types (1)

Performance requirements reflecting and characterizing system properties such as : expected performance, usability, efficiency, reliability, maintainability, reusability, and security requirements)

- Response time, throughput
- Resource usage
- Reliability, availability
- Recovery from failure
- Allowances for maintainability, enhancement, and reusability

Design constraints: Categories constraining the environment and technology of the system.

- Platform (minimal requirements, OS, devices…)
- Technology to be used (language, DB, …)

- Commercial constraints: Categories constraining the project plan and development methods
  - Development process (methodology) to be used
  - Cost and delivery date
  - Often put in contract or project plan instead

Non-Functional Requirements (NFR) Types (2)

Important To Users

- Performance
- Security
- Usability
- Compatibility
- Accessibility
- Reliability
- Flexibility
- Interoperability

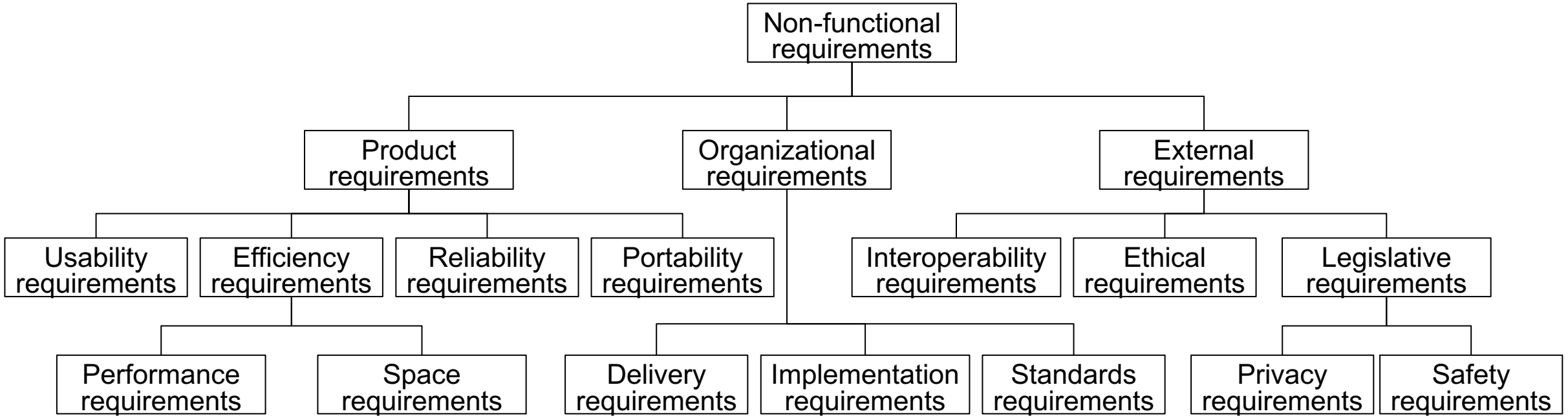
Important To Developers

- Maintainability
- Portability
- Reusability
- Testability

Non-Functional Requirements (NFR) Types (3)

- ***Usability***: the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system.
  - Relates to the user interface—number of nested levels in menus, color schemes …, online help, level of documentation …
- ***Dependability***: the property of a system such that reliance can justifiably be placed on the service it delivers. Includes reliability, robustness, and safety.
- ***Reliability***: the ability of a system to perform its required functions under stated conditions for a specified period of time.
  - Includes acceptable mean time to failure, the ability to detect specified faults or withstand specified security attacks
- ***Robustness***: the degree to which a system can function correctly in the presence of invalid inputs or stressful environment conditions.
- ***Safety***: A measure of the absence of catastrophic consequences to the environment.

NFR: Sommerville's Classification



Non-Functional Requirements (NFR) Types (4)

- ***Performance***: Quantifiable attributes of the system such as response time, throughput, availability, accuracy.
- ***Response time***: how quickly the system reacts to a user input.
- ***Throughput***: how much work the system can accomplish within a specified amount of time.
- ***Availability***: the degree to which a system is operational and accessible when required for use.
  - E.g., an availability of 0.998 means that in every 1000-time units, the system is likely available for 998 units.
- ***Accuracy***: a quantitative measure of the magnitude of error.
- ***Supportability***: Requirements concerned with the ease of changes to the system after deployment. Includes adaptability, maintainability.
- ***Adaptability***: the ability to change the system to deal with additional application domain concepts.
- ***Maintainability***: the ability to change the system to deal with new technology or to fix defects.

Examples of Non-Functional Requirements

- Product requirement
  - The product should be readily portable to the Linux operating system.
  - The product should be easy to use with only one hand
  - The product should be used with poor lighting conditions and the users will wear gloves
  - The product should identify an aircraft within 0.25 seconds
- Process requirement
  - The system development process and deliverable documents shall conform to the process and deliverables defined in XYZCoSPSTAN95.
- Security requirement
  - The system shall not disclose any personal information about customers apart from their name and reference number to the operators of the system.

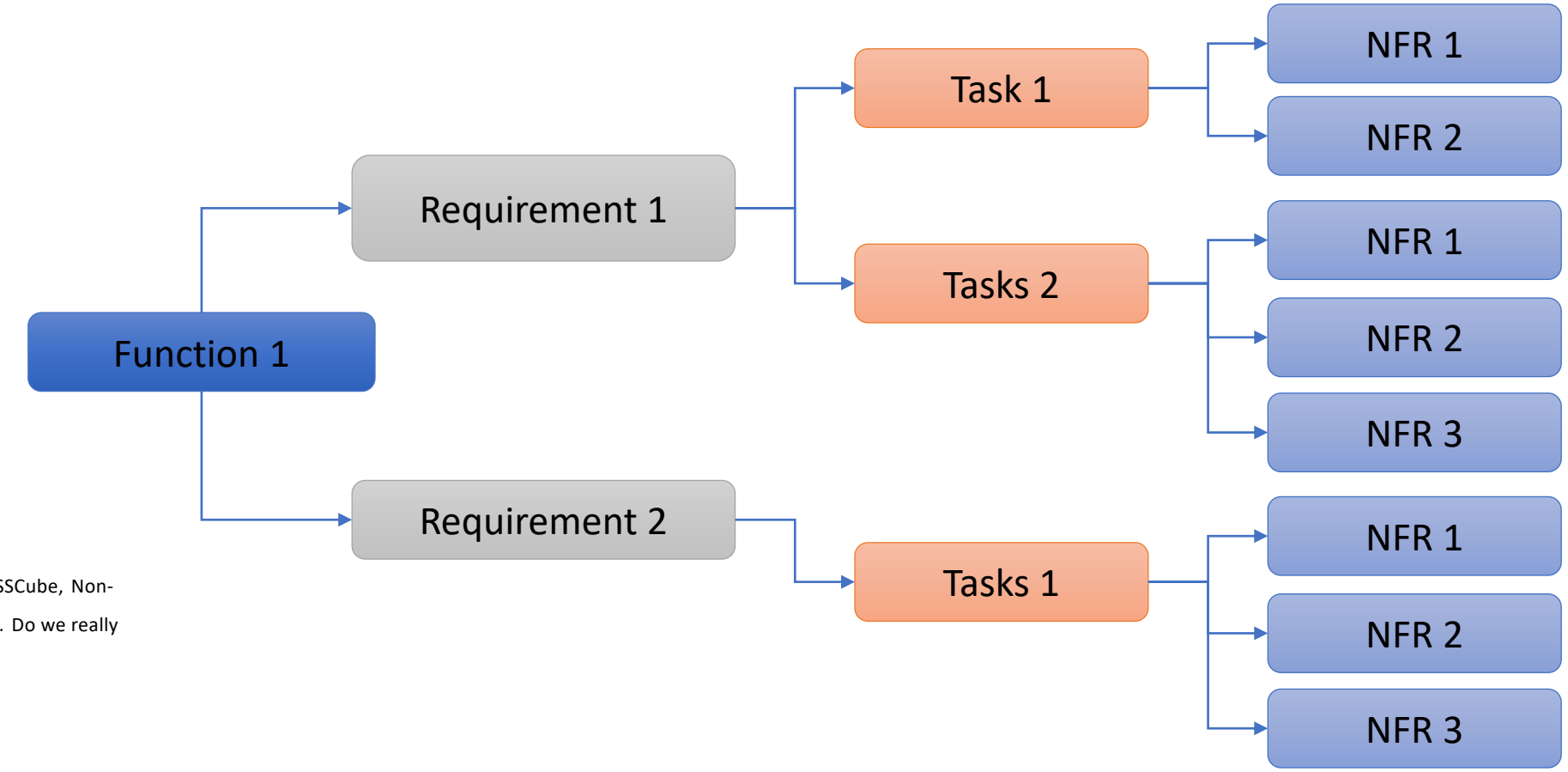
Quantitative Non-Functional Requirements

Property	Measure
Speed	Processed transactions/second User/Event response time Screen refresh time
Size	K Bytes Number of RAM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

Source: Gerald Kotonya and Ian Sommerville, Requirements Engineering – Processes and Techniques, Wiley, 1998

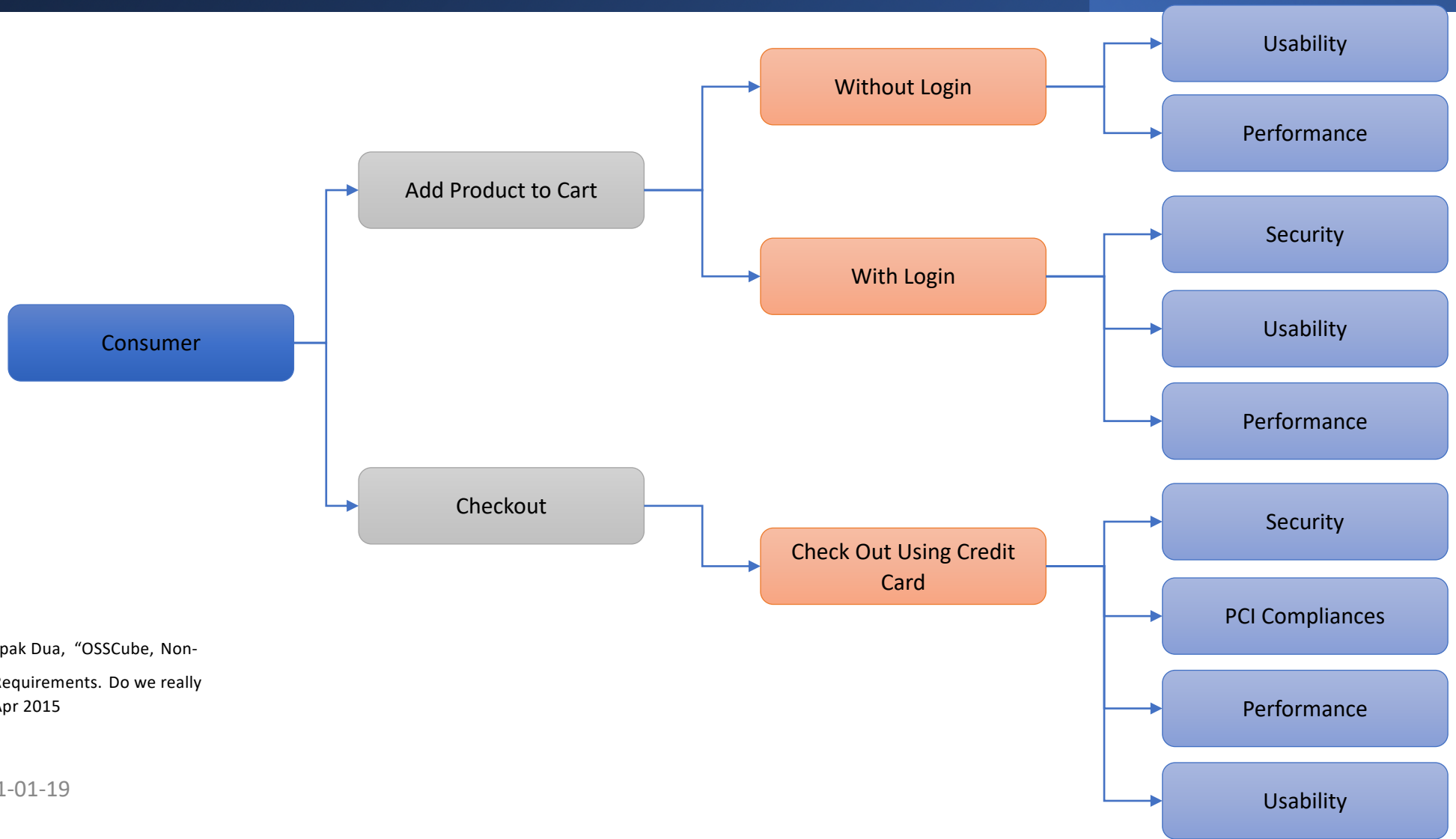
Identification of Non-Functional Requirements

- Non-Functional Requirements can be identified by breaking the requirement in Tasks and Sub Tasks



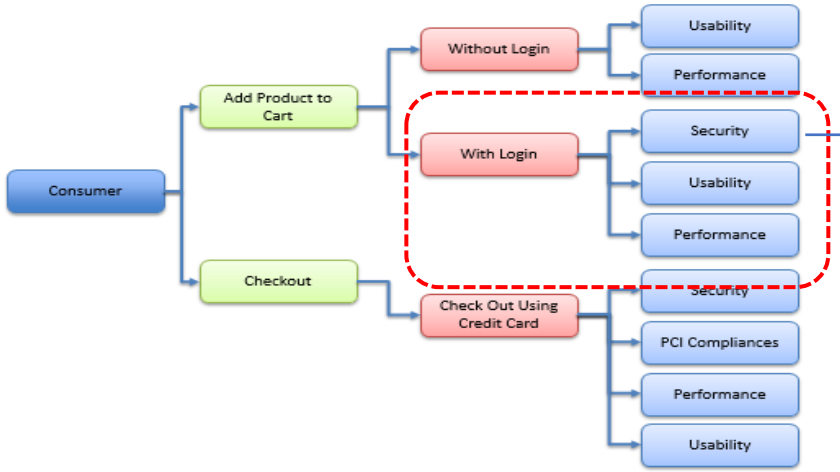
Source: Deepak Dua, "OSSCube, Non-Functional Requirements. Do we really care…?", 9 Apr 2015

Example: E-Commerce Website



Source: Deepak Dua, "OSSCube, Non-Functional Requirements. Do we really care…?", 9 Apr 2015

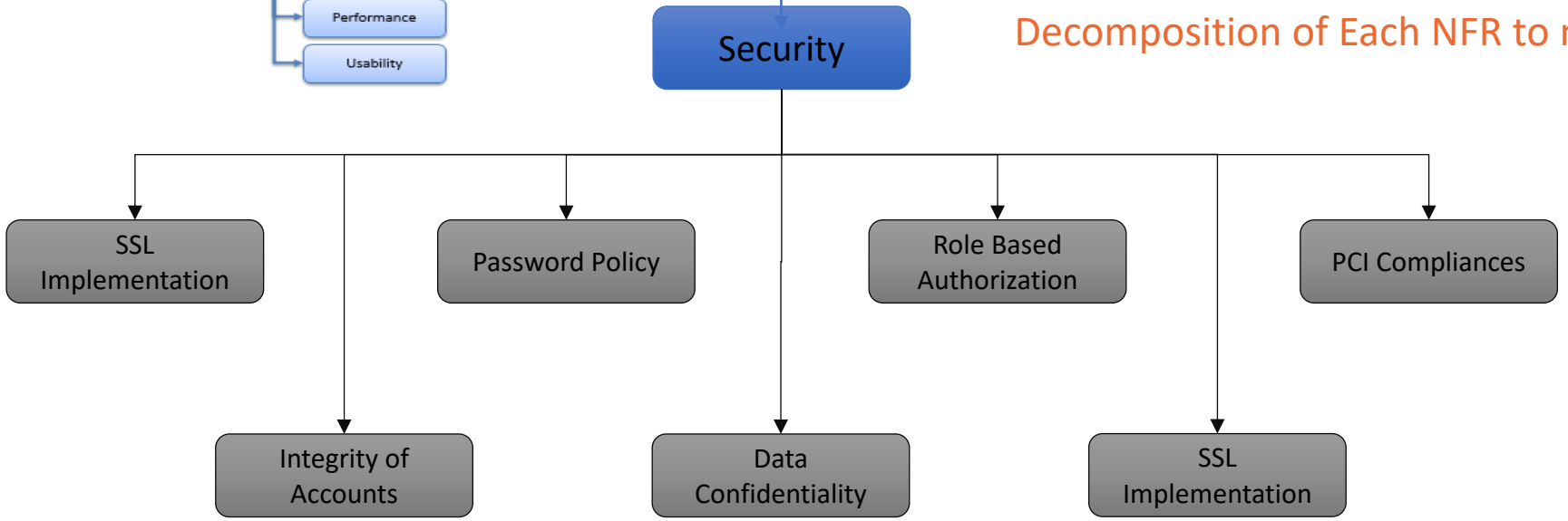
Identification of Non-Functional Requirements



- Identify NFR for each Task
- Establish the Goal
- Continue until you cannot decompose further

Here we are taking the example of Security

Decomposition of Each NFR to meet the Goal



Source: Deepak Dua, "OSSCube, Non-Functional Requirements. Do we really care…?", 9 Apr 2015

Functional vs. Non-functional

Functional Requirements	Non-Functional Requirements
• Product features	• Product property
• Describe the actions with which the user work is concerned	• Describe the experience of the user while doing the work
• A function that can be captured in use cases	• Non-functional requirements are **global constraints** on a software system that results in development costs, operational costs
• A behavior that can be analyzed by drawing sequence diagrams, state charts, etc.	• Often known as software qualities
• Can be traced to individual set of a program	• Usually, cannot be implemented in a single module of a program

Source of requirements

- **Discussions with all classes of stakeholders** in order to elicit their requirements based on their individual perspectives of the process.
- **Competitive analysis** of any competing systems already in the marketplace.
- **Policy and Procedure Manuals** for the business and process at hand so that all system interactions, regulatory constraints and so on can be identified.
- **The marketing and customer care departments** should not be overlooked. It is likely that they have data that can be extracted from customer surveys and questionnaires.
- For **legacy systems**, the following should also be considered: System Manuals, Specifications, Issue Logs, and Enhancement Requests.

Requirements Characteristics

- A complete Requirement Specifications must be:
  - Clear,
  - Correct,
  - Consistent,
  - Coherent,
  - Comprehensible,
  - Modifiable,
  - Verifiable,
  - Prioritized,
  - Unambiguous,
  - Traceable,
  - Credible source

Quality Example [Lethbridge]

Restaurant Advisor System:

"This system will allow people to choose a restaurant in a city. Users enter one or more of the following criteria, and then the system searches its database for suitable restaurants: food type, price range, neighbourhood, size, service type (fast food, cafeteria, buffet, full service), smoking arrangements (none allowed, separately ventilated section, non-separately-ventilated section). The user can also specify a desired day and time period, and the number of people in their party. The system will tap into the reservation database (of participating restaurants) and only display restaurants that have available space. After entering the criteria, the user clicks on "search" and the system displays a list of matching restaurants. For restaurants that participate in the automated reservation system, the user can click on "reserve" next to a selection in order to make a reservation.

Point out problems that you find in this
"short statement of functional requirements"

Restaurant Advisor System: quality deficiencies

- *Duplication* (ie. saying twice in two different ways) :
  - System searches for suitable / System displays matching.
- Food type, price range, neighbourhood, size are *inadequately defined*.
  - Are these taken from a fixed set of values, does a database contain free-form information? Will it be standardized (to make searches easier)?
- *Ambiguity* in “reservation database” and “automated reservation system”
  - Same thing or not ?
- *Unclear*: It appears that some listed restaurants are not in the reservation system/database.
  - What does the system do with restaurants that are not participating ? Are they omitted from the list?
- *Unspecified*: Can user select just one option or more than one option for “type of food”, or smoking arrangement.
- *Incomplete*: If user selects “reserve”, there must be some way for the system to record identifying information about the user so restaurant knows who made it. This is omitted.

Amazon System Requirements / 1

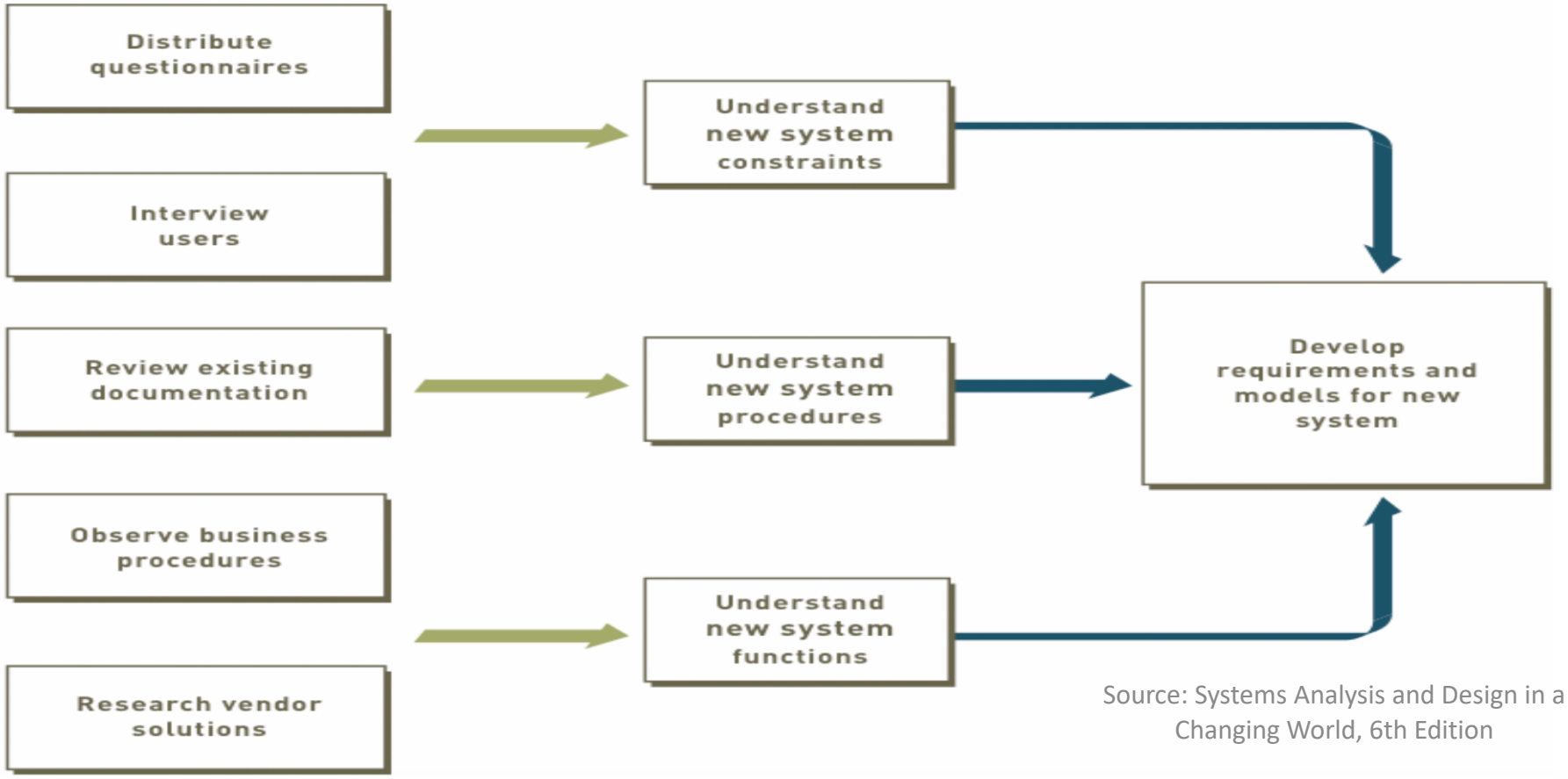
- The following list includes some of the functional and non-functional requirements of the Amazon.ca website.
- **Functional** Requirements include:
  - **Search** - enable user to find item(s) based on variety of item characteristics
  - **Browse** - enable user to look through items
  - **Shop** - enable user to select and purchase items
  - **Comment** - enable user to submit his/her comments on items and read other users' comments on items

Amazon System Requirements / 2

- Non-functional Requirements include:
  - **Operational** - the system should work on any web browser.
  - **Performance** - the system should be available 24/7.
  - **Security** - the system enables registered customers to review their own accounts.
  - **Cultural** - the system exists in versions tailored to global users,
    - e.g., French, Japanese, German, etc.

Modeling

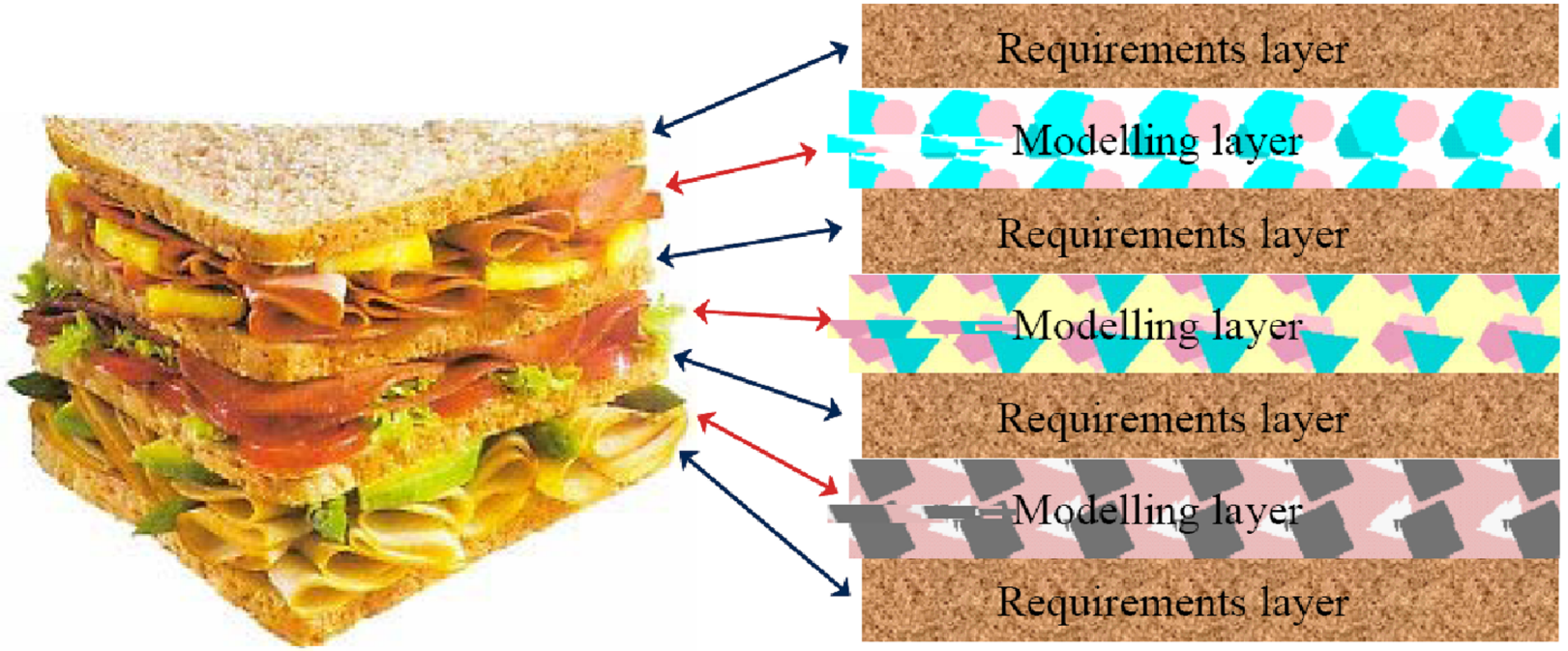
Gathering and Requirements Development



Source: Systems Analysis and Design in a Changing World, 6th Edition

Requirements and Modeling go together

- The systems engineering sandwich!



Source: http://www.telelogic.com/download/paper/SystemsEngineeringSandwich.pdf

Why combining RE with modeling ?

- For analysis – models help to understand the problem domain
- For documentation – models can be used for describing requirements (instead of solely using natural language)
- Requirements are always in a textual format, thus there is a chance of high ambiguity or miss understanding.
    - Graphical Modeling is a visual way of understanding requirements, unification of representations (i.e., stockholder with variant culture or language)
- We will come back to various modeling approaches later in this course.

Models and Modeling

- **Model** – a representation of some aspect of the system being built.
- Types of Models
  - **Textual** model– something written down, described
  - **Graphical** models– diagram, schematic
  - **Mathematical** models– formulas, statistics, algorithms

Two Approaches to System Development

- Describes the system as a set of processes
- Processes interact with data stores
- Processes accept inputs and produces output

- Views system as collection of interacting objects that work together to accomplish tasks
- Object send and respond to messages

Structured System Analysis

Environmental Model

- Defines the scope of the proposed system
- Defines the boundary and interaction between the system and the outside world
- Composed of *Statement of Purpose*, *Context Diagram*, and *Event List*

Behavioral Model

- Model of the internal behavior and data entities of the system
- Model the functional requirements
- Composed of *Data Dictionary*, *Data Flow Diagram*, *Entity Relationship Diagram*

Object-Oriented System Analysis using UML

- Use-cases
- Class Diagrams
- Sequence Diagrams
- Activity Diagrams
- State Diagrams
- Package Diagrams
- Etc.

Summary

- Definition and Importance of Requirements
- Requirement Engineering activities
- Types of Requirements
- Requirements Quality
- Business Examples
- Modeling Techniques

Next Class

- Requirements Inceptions and Elicitations