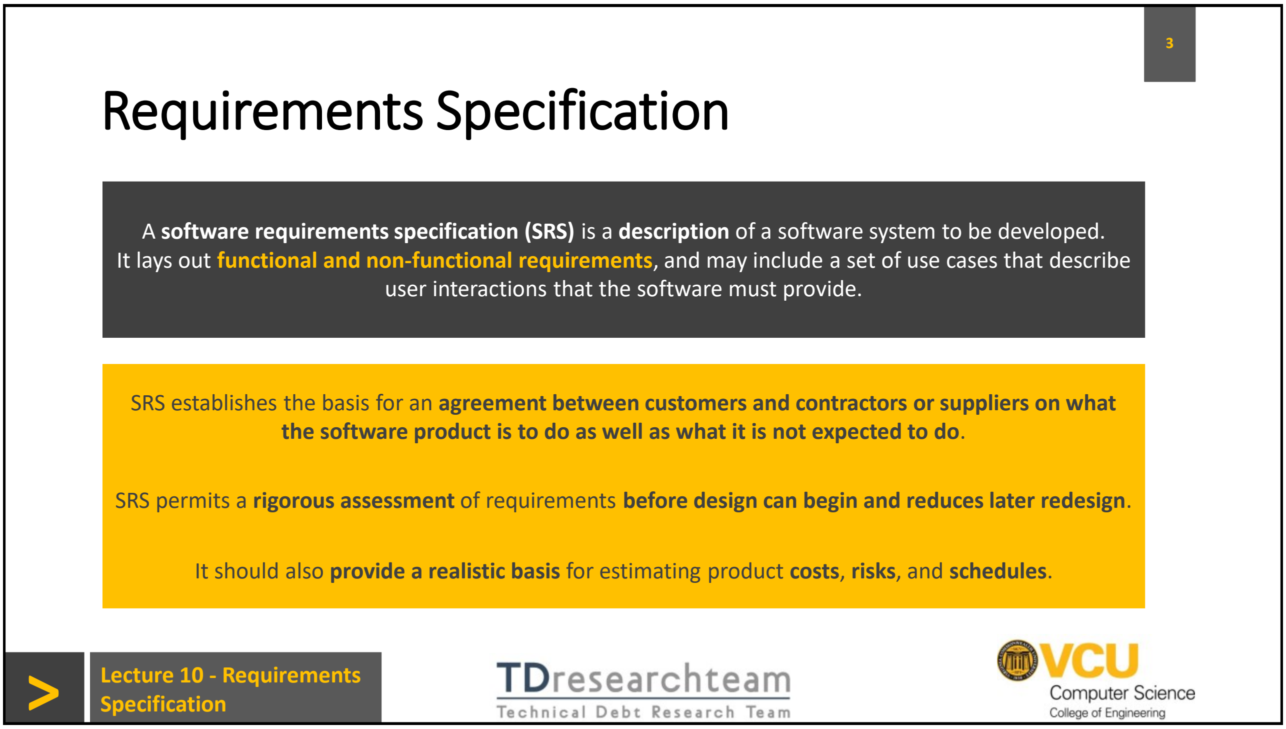
Requirements Specification

Dr. Rodrigo Spínola

Agenda

- Types of requirements
  - Functional
  - Non-functional
  - Domain
- Requirements Specification
  - Use cases



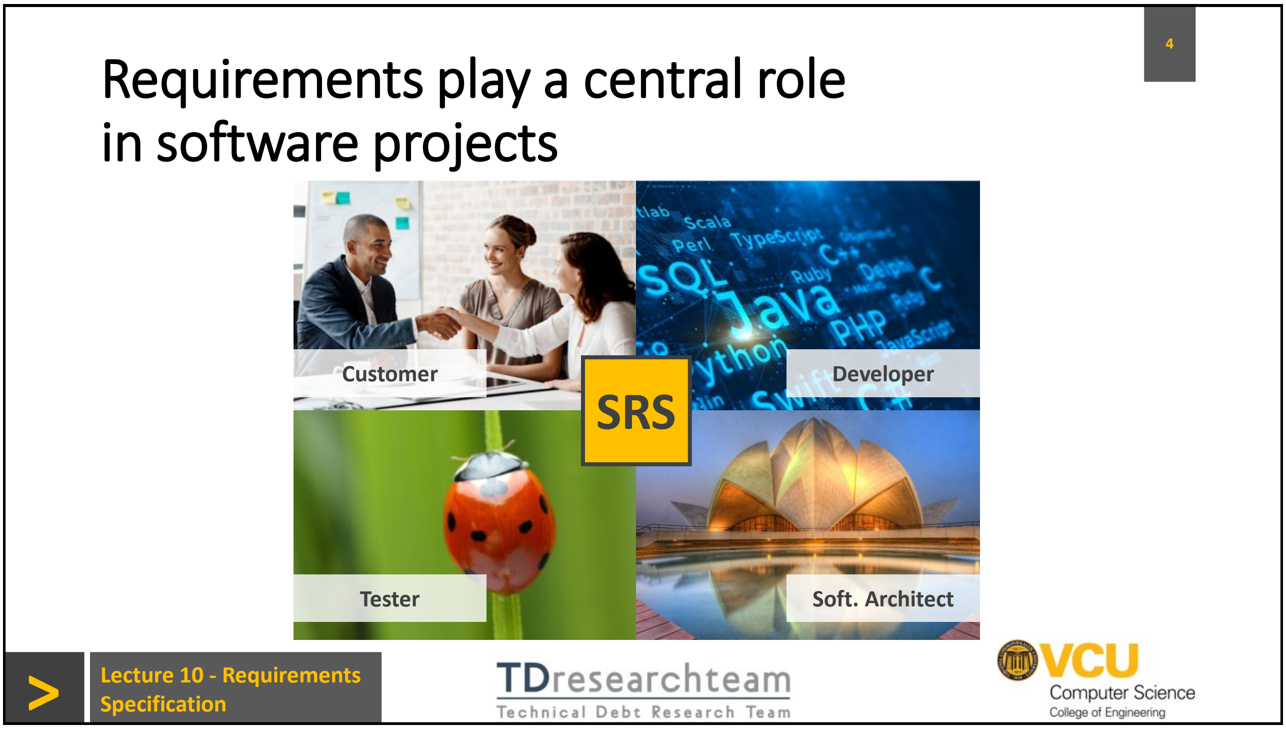
Requirements Specification

A **software requirements specification (SRS)** is a **description** of a software system to be developed. It lays out **functional and non-functional requirements**, and may include a set of use cases that describe user interactions that the software must provide.

SRS 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**.

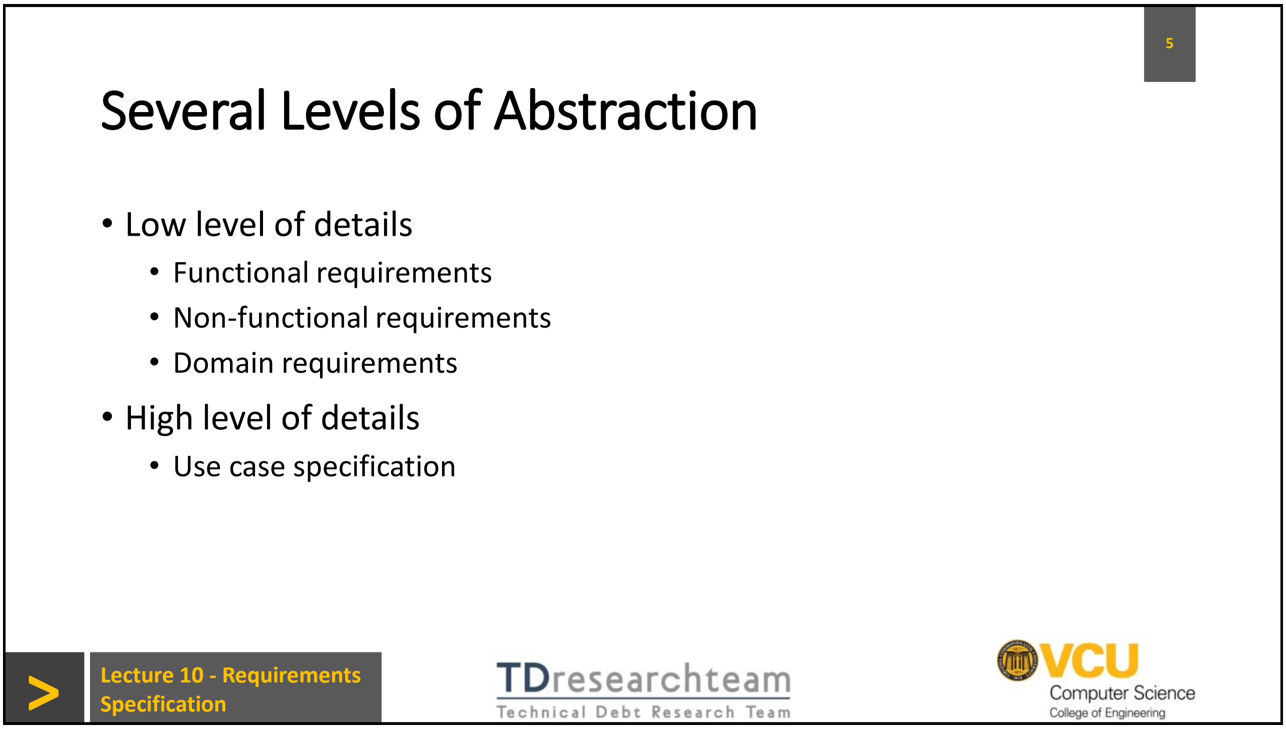
SRS permits a **rigorous assessment** of requirements **before design can begin and reduces later redesign**.

It should also **provide a realistic basis** for estimating product **costs**, **risks**, and **schedules**.



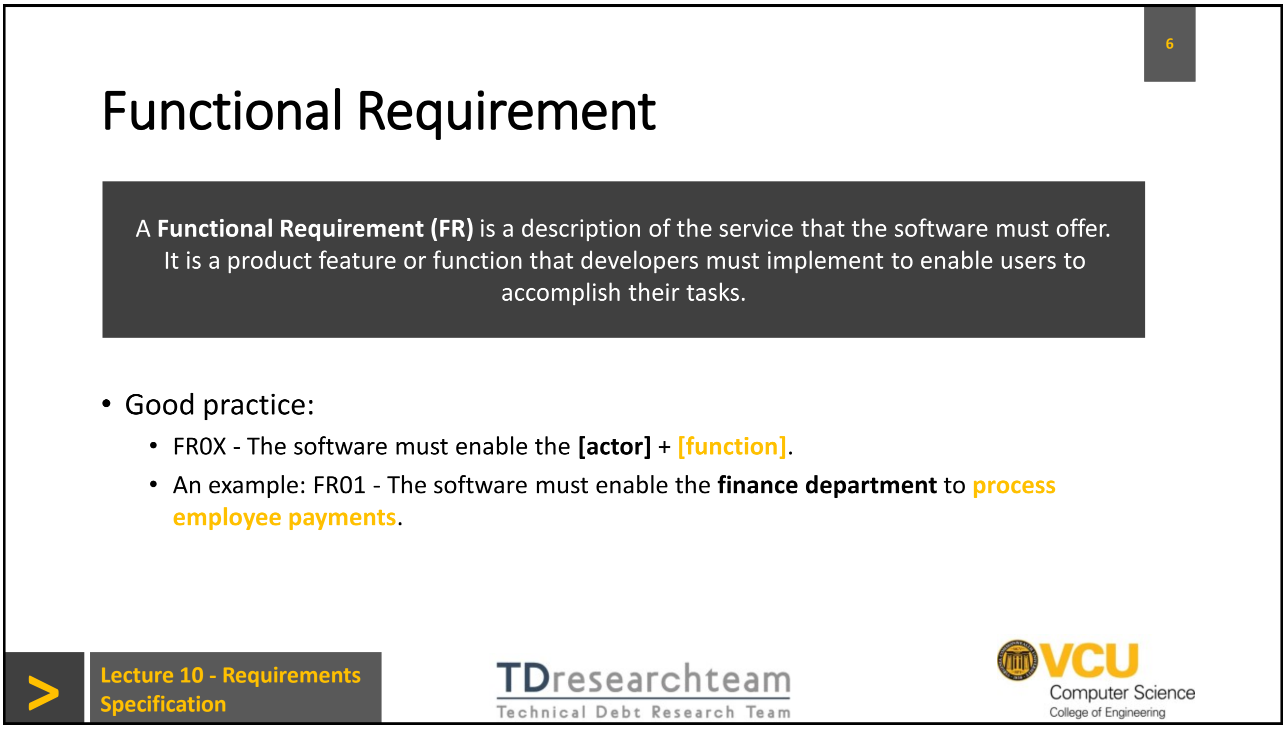
Requirements play a central role in software projects

Customer
Developer
SRS
Tester
Soft. Architect



Several Levels of Abstraction

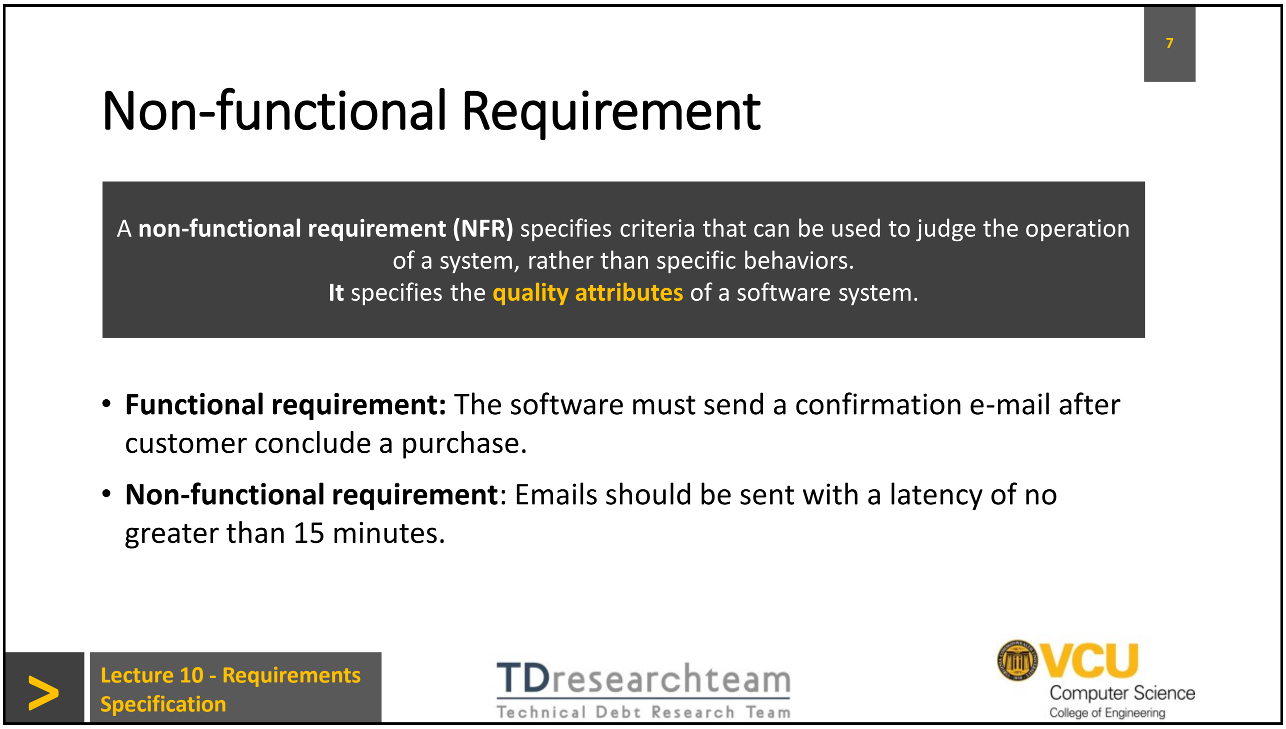
- Low level of details
  - Functional requirements
  - Non-functional requirements
  - Domain requirements
- High level of details
  - Use case specification



Functional Requirement

A **Functional Requirement (FR)** is a description of the service that the software must offer. It is a product feature or function that developers must implement to enable users to accomplish their tasks.

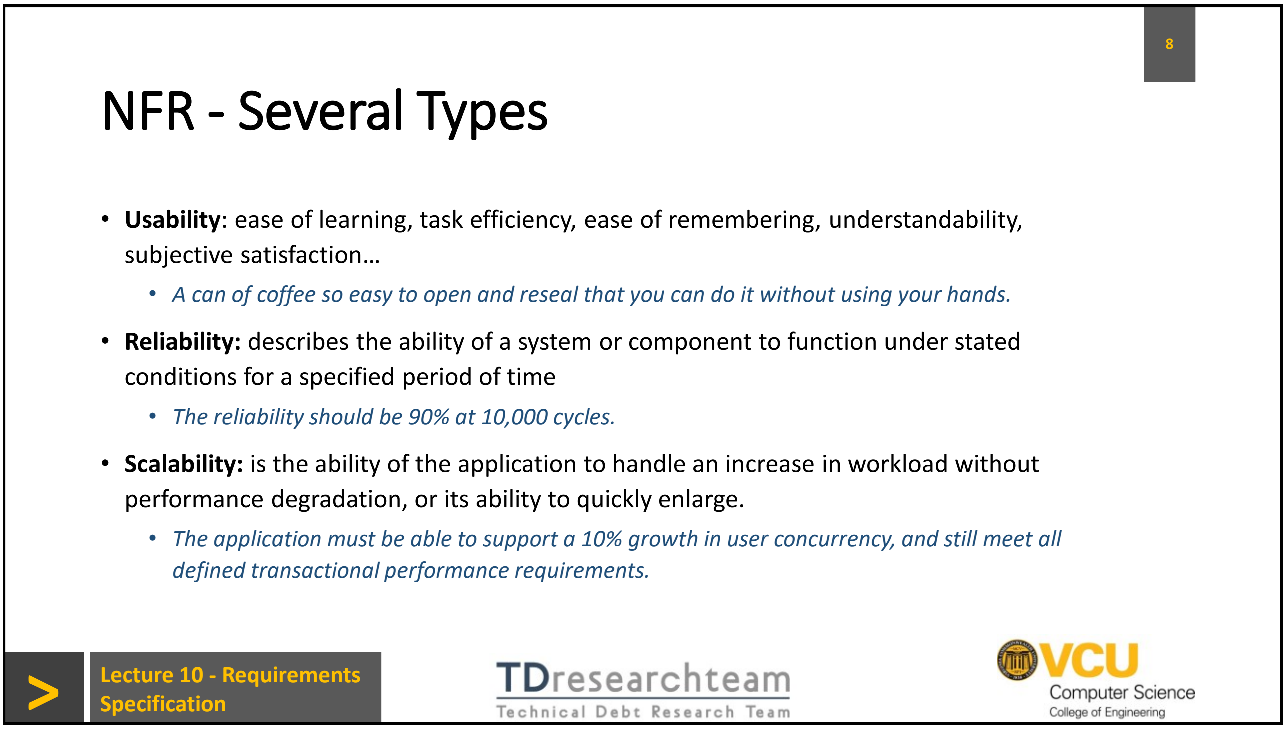
- Good practice:
  - FR0X - The software must enable the **[actor]** + **[function]**.
  - An example: FR01 - The software must enable the **finance department** to **process employee payments**.



Non-functional Requirement

A **non-functional requirement (NFR)** specifies criteria that can be used to judge the operation of a system, rather than specific behaviors.
It specifies the **quality attributes** of a software system.

- **Functional requirement:** The software must send a confirmation e-mail after customer conclude a purchase.
- **Non-functional requirement**: Emails should be sent with a latency of no greater than 15 minutes.



NFR - Several Types

- **Usability**: ease of learning, task efficiency, ease of remembering, understandability, subjective satisfaction…
  - *A can of coffee so easy to open and reseal that you can do it without using your hands.*
- **Reliability:** describes the ability of a system or component to function under stated conditions for a specified period of time
  - *The reliability should be 90% at 10,000 cycles.*
- **Scalability:** is the ability of the application to handle an increase in workload without performance degradation, or its ability to quickly enlarge.
  - *The application must be able to support a 10% growth in user concurrency, and still meet all defined transactional performance requirements.*

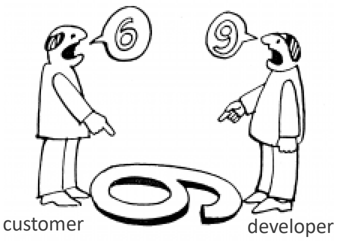
NFR - Several Types

- **Performance:** it is a measure of an application, component or service’s ability to respond within a defined period of time
  - *A batch process must complete its processing within 60 minutes.*
- **Portability:** how easily the software system can be transferred from its current hardware or software environment to another environment.
  - *A game running on Windows 11 is said to be portable if the same game can be run on Windows 10 without any change in the behavior of the game*
- **Security**
  - *The application does not store hard coded sensitive information.*

NFR Must be Measurable!

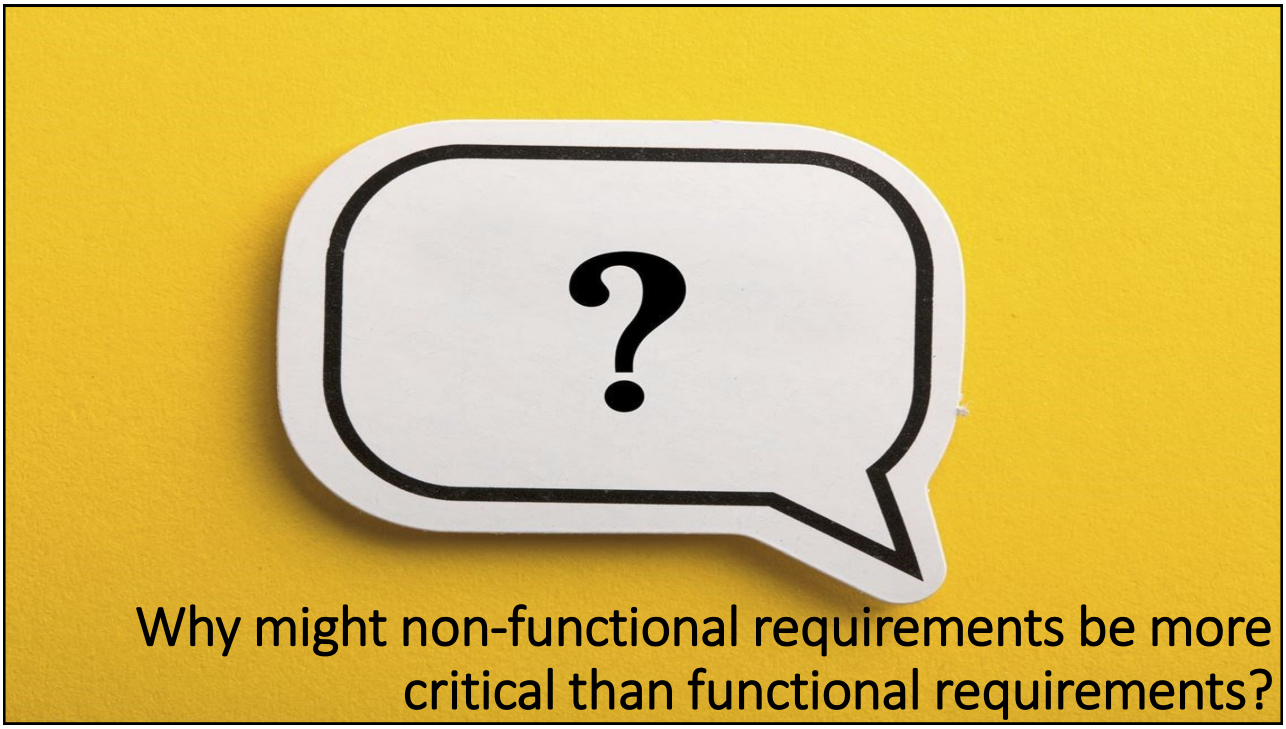
Avoid terms like friendly interface, small files, quick performance, very safe...

You can not measure them!!!

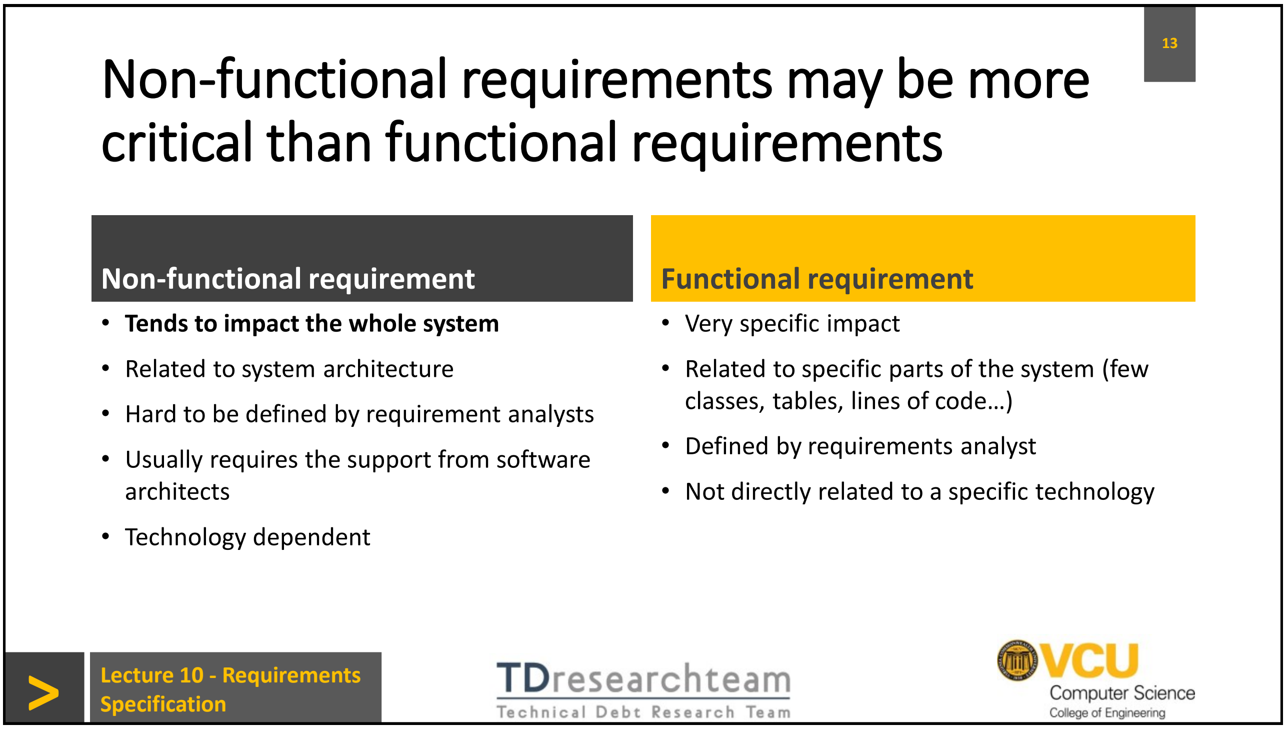


You can not depend on different points of view when evaluating the system.

Extra credit question



Why might non-functional requirements be more critical than functional requirements?



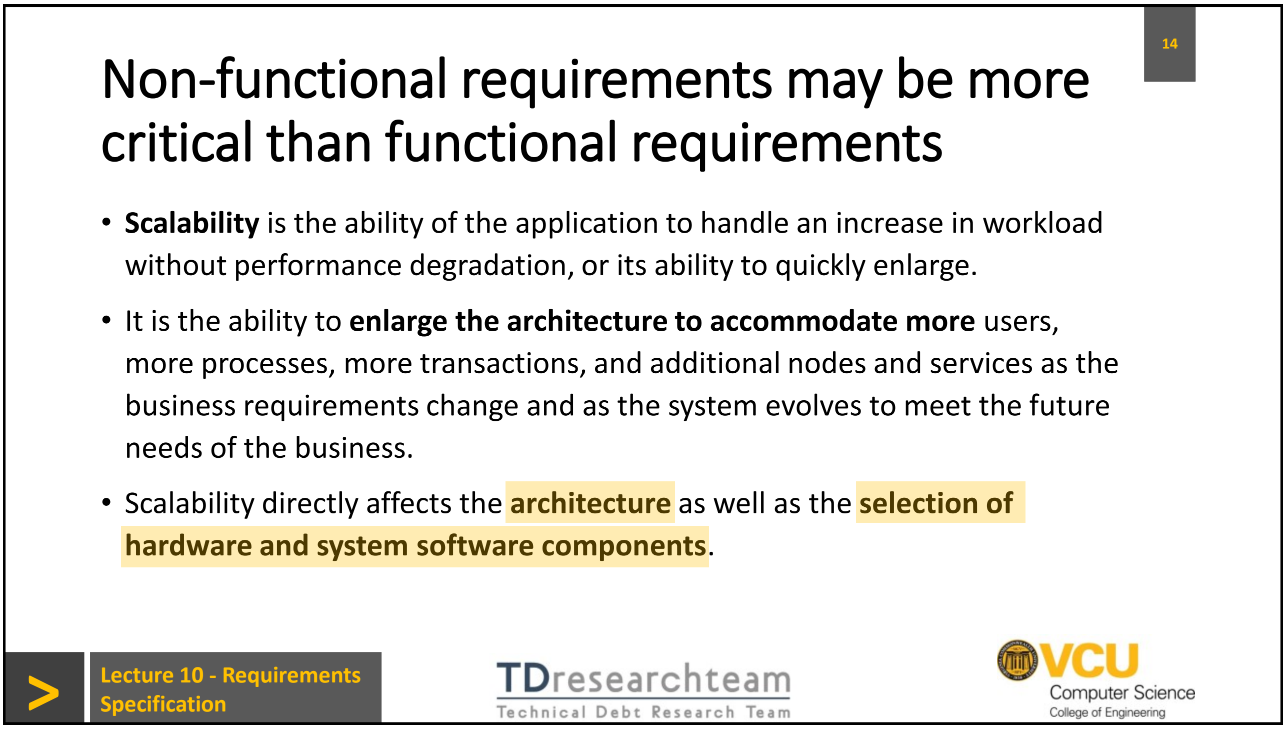
Non-functional requirements may be more critical than functional requirements

Non-functional requirement

- **Tends to impact the whole system**
- Related to system architecture
- Hard to be defined by requirement analysts
- Usually requires the support from software architects
- Technology dependent

Functional requirement

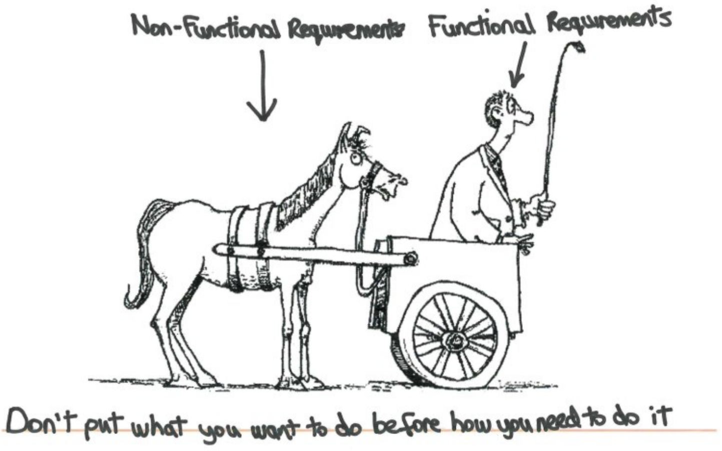
- Very specific impact
- Related to specific parts of the system (few classes, tables, lines of code…)
- Defined by requirements analyst
- Not directly related to a specific technology



Non-functional requirements may be more critical than functional requirements

- **Scalability** is the ability of the application to handle an increase in workload without performance degradation, or its ability to quickly enlarge.
- It is the ability to **enlarge the architecture to accommodate more** users, more processes, more transactions, and additional nodes and services as the business requirements change and as the system evolves to meet the future needs of the business.
- Scalability directly affects the **architecture** as well as the **selection of hardware and system software components**.

Functional Requirement vs Non-functional Requirement


Non-Functional Requirements
Functional Requirements
Don't put what you want to do before how you need to do it

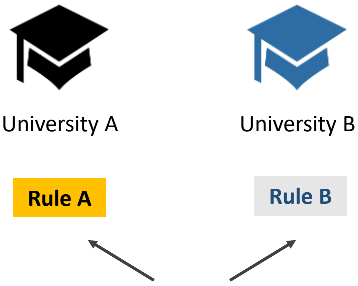
Domain Requirement

- Usually, they complement the specification of functional requirements
- For example, the domain requirement below is included in the requirements specification for an automated train protection system. This system automatically stops a train if it goes through a red signal. This requirement states how the train deceleration is computed by the system:
  - *The deceleration of the train shall be computed as:*

    D (train) = D (control) + D (gradient)
  - *where*
    - *D (gradient) is 9.81ms2 * compensated gradient/alpha and*
    - *the values of 9.81ms2 /alpha are known for different types of train.*

Domain Requirement - Another Example

Both have a FR related to student approval in courses



University A

University B

Rule A

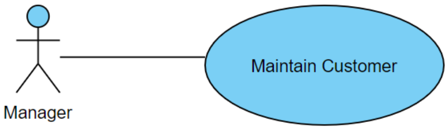
Rule B

But the rules can be different from each other

We define those rules through domain requirements (or business rules)

Use Case Specification

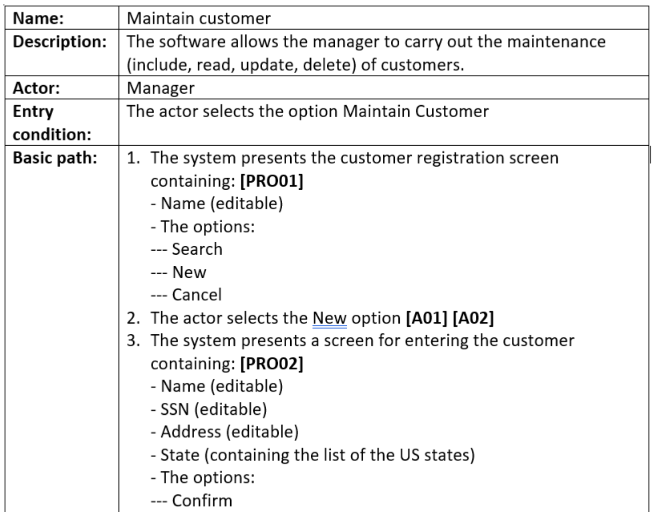
A **Use Case Specification** is a textual description of the functionality provided by the system. It captures **actor-system interaction**. That is, it specifies how a user interacts with a system and how the system responds to the user actions.



Sample outline of a use-case specification

Use-case name	States the use-case name. Typically, the name expresses the objective or observable result of the use case, such as "Withdraw Cash" in the case of an automatic teller machine
Brief description	Describes the role and purpose of the use case
Basic path	Describes the ideal, primary behavior of the system
Alternative paths	Describes exceptions or deviations from the basic flow, such as how the system behaves when the actor enters an incorrect user ID and the user authentication fails
Business Rules	Describes specific domain-related behaviors of the system
Prototypes	Presents prototypes related to the steps of the flow of events

An Example



Name:	Maintain customer
Description:	The software allows the manager to carry out the maintenance (include, read, update, delete) of customers.
Actor:	Manager
Entry condition:	The actor selects the option Maintain Customer
Basic path:	1. The system presents the customer registration screen containing: **[PRO01]** - Name (editable) - The options: --- Search --- New --- Cancel 2. The actor selects the New option **[A01] [A02]** 3. The system presents a screen for entering the customer containing: **[PRO02]** - Name (editable) - SSN (editable) - Address (editable) - State (containing the list of the US states) - The options: --- Confirm

Summary

- Types of requirements
  - Functional
  - Non-functional
  - Domain
- Requirements Specification
  - Use cases

TDresearchteam
Technical Debt Research Team

VCU
Computer Science
College of Engineering

Class is over, questions?

Requirements Specification

Dr. Rodrigo Spínola

> Lecture 10 - Requirements Specification

TDresearchteam
Technical Debt Research Team

VCU
Computer Science
College of Engineering