Software Design & Architecture

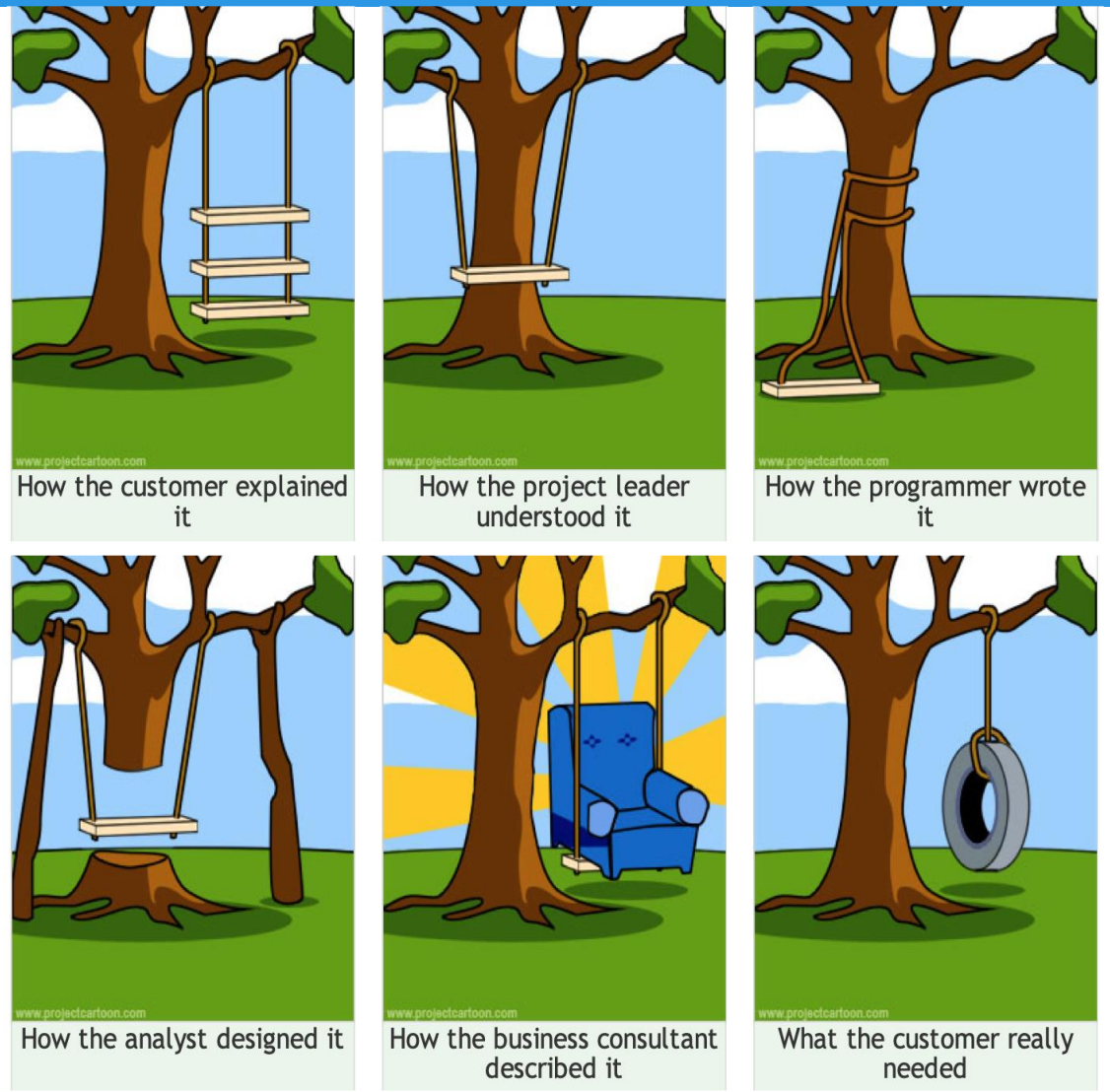
Non-Functional Requirements

Pengyu Nie

Acknowledgements: slides adapted from previous versions by Mei Nagappan and Shane McIntosh, which are adapted from previous versions by Zhen Ming Jiang, Ahmed E. Hassan, Reid Holmes. Some images taken from https://www.grt.ca/.

Agenda

- Stakeholders
  - customer-side
  - developer-side
- Non-functional requirements
  - vs. functional requirements
  - types of NFRs
  - good and bad NFRs
  - conflicts and tradeoffs



Source: http://projectcartoon.com

Who is a system stakeholder?

- Definition: a person with an interest or concern in something
- Broadly speaking
  - The customers who use the system

  - Software developers who build the system

Stakeholder in the Architecture

- “A stakeholder in the architecture of a system is an individual, team, organization, or classes thereof, having an interest in the realization of the system.”

  -- Rozanski & Woods

- Customers using the software; software developers; …other people?

Running Example

- ION light rail's ticket vending machine
- Who have an interest in it?

Customer-side Stakeholders

- The customers buying tickets
  - paying by cash
  - paying by credit card
  - buying one time ticket
  - buying a refillable card
- The Region of Waterloo
  - that paid for the system

Developer-side Stakeholders

- Software developers
- User interface designers
- Testing (Quality Assurance) engineers
- Release engineers
- Operators
- Maintenance engineers
- Managers

And more...

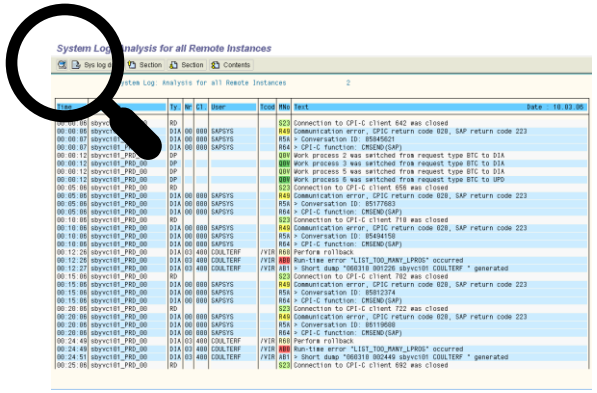
- The person who installs the system in the stations
- The person who inspects the systems
- The person who collects money from the machines
- Network engineers who keep the machine connected
- The electricians who set up power for it
- Customer service operators who deal with complaints
- City officials who liaison with the company to manage fixes and upgrades

TL;DR

- Systems are more complex than you initially think
- There are more stakeholders than one can initially think
- While their stake in the system varies, every one of them has a say in how some part of the system is designed and built

Why do we need to care about stakeholders?

- Different stakeholders have different concerns that may not be compatible with each other
- It is impossible to talk to all stakeholders;
  at least, consider who they might be and what their concerns may be



Inspectors who audit the machines

has sufficient data for inspection/debugging?
leaks private user data (e.g., credit card numbers)?
…

Exercise: Where is the line?

- ION light rail's ticket vending machine
- Discuss whether or not the following candidates should be stakeholders

Security staff at the ION stations where vending machines reside
Drivers of ION vehicles
The spouses of the developers of the ION fare vending machine
The CEO of the software firm that was hired to develop the ION fare vending system
The firm hired to lay the rails for the ION vehicles
A sight-impaired ION rider

Non-Functional Requirements

- Functional requirements (aka features)
  - what the system is supposed to do (“the system shall do X”)
- Non-functional requirements
  - what the system is supposed to be (“the system shall be X”)
  - constraints under which the system should work
  - e.g., efficiency, scalability, security, usability

FR vs NFR

- Products are sold based on their functional requirements
  - e.g., cell phone, car, tent
- However, non-functional requirements play a critical role in perception
  - “This program keeps crashing”
  - “It’s too slow”
  - “It doesn’t work with …”
- Non-functional requirements are differentiators for similar products
  - e.g., BlackBerry, iPhone, Android

How to write requirements?

- Ask stakeholders / consider the questions they may ask
- Management: are we on schedule?
- Developers: who is responsible for implementing what?
- Sales: can we claim it can do this task?
- QA: what teams do we talk to about defects?
- DevOps: where should this component be deployed?
- Support: which QA team signed off on this?
- Maintenance: how can we add this feature?

Types of NFRs

Efficiency	Privacy	Availability	Complexity
Usability	Security	Reliability	Readability
Portability	Survivability	Robustness	Heterogeneity
Accessibility	Safety	Fault-tolerance	
		Scalability	
		Evovability	

NFRs Related to User Experience

- Efficiency: Ability to meet performance requirements
  - e.g., completes operation Y in X seconds
- Usability: How intuitive the user interface of the system is
  - e.g., easily navigate through different features, including X Y Z
- Portability: Ability to execute on multiple platforms while retaining their functional and non-functional properties
  - e.g., runs on Windows, MacOS, and Linux
- Accessibility: The degree to which a product, device, service, or environment is available to as many people as possible
  - e.g., is compatible with screen reader and uses alt text for all images

NFRs Related to User Safety

- Privacy: How a system protects the private information of the user
  - e.g., end-to-end encryption for private messages
- Security: How well the system protects the user from an external attack
  - e.g., multi-factor authentication to protect user accounts from unauthorized access
- Survivability: Ability to resist, recover, and adapt to threats (attacks, failures, accidents, etc.)
  - e.g., redundancy in infrastructure to ensure continuous operation during cyber-attacks or natural disasters
- Safety: Ability to avoid failures that will cause loss of life, injury, or loss to property
  - e.g., collision detection in autonomous driving system

NFRs Related to Operation

normal use

when things go wrong

- Availability: The probability the system is available at a particular instant in time
  - e.g., up time of XX%
- Reliability: The probability that a system will perform within its design limits without failure over time
  - e.g., XX% of success rate
- Robustness: Ability to respond adequately to unanticipated runtime conditions
  - e.g., does not crash given the invalid user input of XX
- Fault-tolenrance: Ability to respond gracefully to failures at runtime (from environment, components, connectors, component-connector mismatches, etc.)
  - e.g., can continue operate with X number of nodes fail

NFRs Related to Operation (cont.)

- Scalability: Capability of a system to be adapted to meet new size/scope requirements
  - e.g., can automatically scale resources to support at most X users
- Evolvability: Ability to react on change, satisfy new requirements, and add support for new environments
  - e.g., features can be easily added/modified under the modular architecture

NFRs Related to Development

- Complexity: The size of a system, the volume of constituent elements, their internal structure, and their interdependencies
  - e.g., consists of X modules, each with around Y lines of code
- Readability: How well the system is comprehensible to a new developer
  - e.g., every function should have no more than X lines and have comments
- Heterogeneity: Ability to be composed of, or execute within, disparate parts
  - e.g., consists of X modules implemented in programming languages A B C correspondingly

Evaluating NFRs

- Think about NFRs concretely
  - how can they be measured?
  - use specific numbers/cases/items

	Good	Bad
Efficiency	The system shall issue new tickets in under 10 seconds	The system shall issue tickets quickly
Usability	The system shall enable a user with no training to buy tickets in four clicks or less	The system shall be user-friendly

Exercise: good and bad NFRs

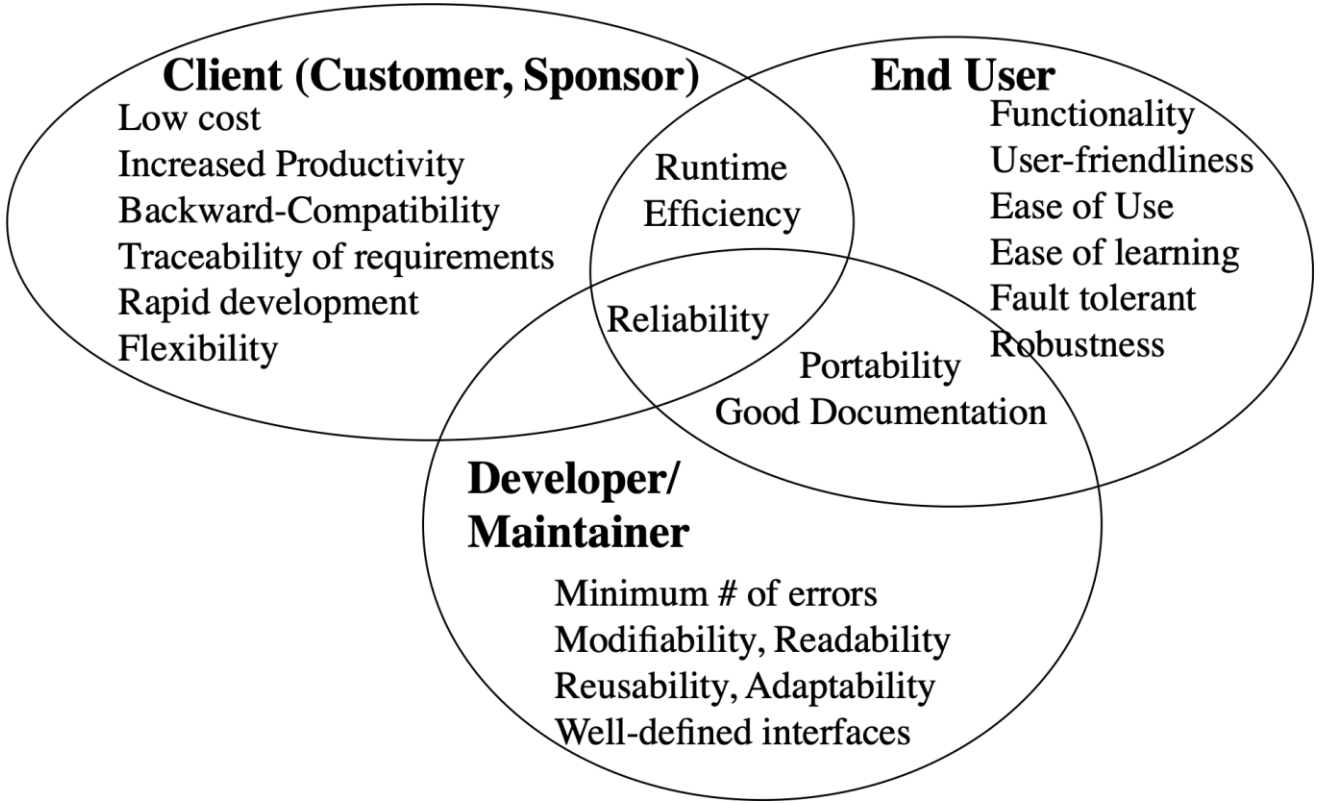
- ION light rail's ticket vending machine
- Discuss and come up with examples of NFRs
- Focus on the following types of NFRs:

Accessibility	Robustness
Privacy	Scalability

- Your response should include
  - NFR type:
  - Good example:
  - The bad version of it:

Stakeholder Conflicts

- Each stakeholder will have their own opinion about what NFRs matter most



Typical Tradeoffs

- Functionality vs. Usability
- Efficiency vs. Portability
- Reusability vs. Cost

Agenda (Recap)

- Stakeholders
  - customer-side
  - developer-side
- Non-functional requirements
  - vs. functional requirements
  - types of NFRs
  - good and bad NFRs
  - conflicts and tradeoffs

⇨ *Think about stakeholders of your project*

⇨ *Think about NFRs concretely*

these will be a part of P2: project proposal