SOFTWARE ENGINEERING

CO3001

CHAPTER 3 – REQUIREMENTS ENGINEERING

Anh Nguyen-Duc
Tho Quan Thanh

Adapted from https://iansommerville.com/software-engineering-book/slides/

TOPICS COVERED

- ✓ Course's project
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Requirements specification
- ✓ Requirement in Agile
- ✓ Requirements validation
- ✓ Requirements change

TOPICS COVERED

- ✓ Course's project
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Requirements specification
- ✓ Requirement in Agile
- ✓ Requirements validation
- ✓ Requirements change

REQUIREMENTS

EFFECTS OF INADEQUATE REQUIREMENTS DEVELOPMENT – AIRBUS

- ✓ **Requirement**: *Reverse thrust may only be used when the airplane is landed.*
- ✓ **Translation**: *Reverse thrust may only be used while the wheels are rotating.*
- ✓ **Implementation**: *Reverse thrust may only be used while the wheels are rotating fast enough.*

- **Situation**: Rainstorm – aquaplaning
- **Result**: Crash due to overshooting the runway!
- **Problem**: Erroneous in the requirement phase

OTHER SOFTWARE RELATED ACCIDENTS & INCIDENTS

- **Accidents & incidents**
  - Alarm flooding, **power distribution** failure, BP Grangemouth Scotland, 29th May - 10th June 2000
  - failure in a data bus + faulty logic in the software => engine power loss, **Airbus** A340-642, 2005
  - failed accelerometer + software bug => Faulty airspeed metering, **Boeing** 777-200, 2005
  - Software bug => shutdown of radio communication between **ATC and aircraft**, 2004. disrupted 800 flights
  - Safety-related software flaws => recall of 200,000 **pacemakers** in 1990-2000
  - **radiotherapy machines** attacked by computer **viruses**, 2005
  - Buggy software incorporate computer + connection between corporate and control systems networks => shutdown of the **nuclear power plant**, USA 2008
- Many software failures are actually "bugs" in the **detailed requirement specifications,** i.e., poor understanding of the very detailed requirements

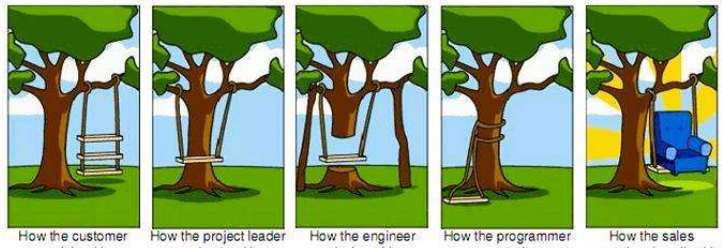
Korean Air 747 in Guam, 200 deaths (1997): incorrect configuration of "minimum altitude" warning system

The Ariane 5 accident – 1

- Single root cause failure!
- **The "bug": attitude deviation** stored as 2-byte integer (max value 65,535) instead of 4-byte (max value 4,294,967,295)
- SW module was reused from Ariane 4
- Insufficient V&V of detailed requirements: larger attitude deviation tolerated in Ariane 5 than in Ariane 4
- Ariane 5 production cost 10 years and $7 billion; luckily, no victims because it was unmanned.

65,535 = 00000000 00000000 11111111 11111111

65,536 = 00000000 00000001 00000000 00000000



WHAT IS A REQUIREMENT?

Requirement engineering = establishing the ***services*** that the customer requires from a system and the ***constraints*** under which it operates and is developed.

- ✓ Requirement = the descriptions of
  - the system services
  - and constraints
- ✓ It may range
  - from a high-level abstract statement
  - to a detailed mathematical functional specification.
- ✓ May serve a dual function
  - The basis for a bid for a contract - must be open to interpretation;
  - The basis for the contract itself - must be in detail;

MENTCARE SYSTEM

- ✓ Mentcare (Medical practice management system): manages the day-to-day operations of a clinic, such as appointment scheduling, billing and other administrative tasks.

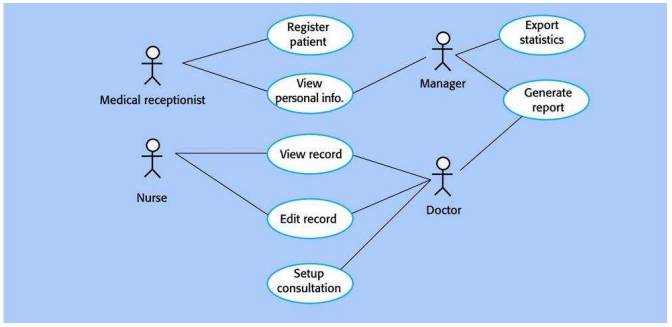
	S	Cat #	Catalog Description	Item #	Description	Qty / Purchased	Unit	Qty	Unit	Released / Due	Par Max	Current Stock	Par Min	Par Critical	
2		202407387	AMLODIPINE BESYLATE 5MG (NORVASC) TAB U/D; 100/BOX	51079045120	AMLODIPINE BESYLATE 5 MG TAB	2.0000 0	100 TA	200.0000	TA	0 2		27 / 27			0
3		202407388	AMLODIPINE BESYLATE 10MG (NORVASC) TAB U/D; 100/BOX	00054010220	AMLODIPINE BESYLATE 10 MG TAB	1.0000 0	100 TA	100.0000	TA	0 1		274 / 275			0
4		200805780	AMOXICILLIN CAP 500MG U/D; 100/BOX	00093310993	AMOXICILLIN 500 MG CAPSULE	2.0000 0	100 CA	200.0000	CA	0 2		186 / 188			0
5		202801738	ASPIRIN EC TAB 81MG U/D 25X30; 750/BOX	63739027201	ASPIRIN EC 81 MG TABLET	1.0000 0	750 TE	750.0000	TE	0 1		172 / 173			0
8		201202229P	BENZTROPINE	00904105661	BENZTROPINE MES	1.0000	100	100.0000	TA	0		104 / 104			0

TOPICS COVERED

- ✓ Course's project – UWC 2.0
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Requirements specification
- ✓ Requirements validation
- ✓ Requirements change

MENTCARE SYSTEM

- ✓ Mentcare (Medical practice management system): manages the day-to-day operations of a clinic, such as appointment scheduling, billing and other administrative tasks.



FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

- ✓ Functional requirements
  - Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
  - May state what the system should not do.
- ✓ Non-functional requirements
  - Constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.
  - Often apply to the system as a whole rather than individual features or services.
  - Domain requirements
    - Constraints on the system from the domain of operation

FUNCTIONAL REQUIREMENTS

- ✓ Describe functionality or system services.
  - Functional user requirements may be high-level statements of what the system should do.
  - Functional system requirements should describe the system services in detail.

MENTCARE SYSTEM: FUNCTIONAL REQUIREMENTS

- ✓ A user shall be able to search the appointments lists for all clinics.
- ✓ The system shall generate each day, for each clinic, a list of patients who are expected to attend appointments that day.
- ✓ Each staff member using the system shall be uniquely identified by his or her 8-digit employee number.

NON-FUNCTIONAL REQUIREMENTS

- ✓ Define system properties and constraints
  - Properties: reliability, response time and storage requirements.
  - Constraints: I/O device capability, system representations, etc.
- ✓ Non-functional requirements may be more critical than functional requirements.
  - If these are not met, the system may be useless.

NON-FUNCTIONAL REQUIREMENTS IMPLEMENTATION

- ✓ Non-functional requirements may affect the overall architecture of a system
  - rather than the individual components
  - cross-cutting concern
- ✓ A single non-functional requirement
  - may generate a number of related functional requirements
  - and may also generate requirements that restrict existing requirements.

NON-FUNCTIONAL CLASSIFICATIONS

- ✓ Product requirements
  - Requirements which specify that the delivered product must behave in a particular way e.g. execution speed, reliability, etc.
- ✓ Organisational requirements
  - Requirements which are a consequence of organisational policies and procedures e.g. process standards used, implementation requirements, etc.
- ✓ External requirements
  - Requirements which arise from factors which are external to the system and its development process e.g. interoperability requirements, legislative requirements, etc.

EXAMPLES OF NONFUNCTIONAL REQUIREMENTS IN THE MENTCARE SYSTEM

- ✓ Product requirement
  - The Mentcare system shall be available to all clinics during normal working hours (Mon–Fri, 0830–17.30). Downtime within normal working hours shall not exceed five seconds in any one day.
- ✓ Organizational requirement
  - Users of the Mentcare system shall authenticate themselves using their health authority identity card.
- ✓ External requirement
  - The system shall implement patient privacy provisions as set out in HStan-03-2006-priv.

TYPES OF REQUIREMENT

- ✓ Requirements definition
  - A statement in natural language plus diagrams of the services the system provides and its operational constraints. Written for customers
- ✓ Requirements specification
  - A structured document setting out detailed descriptions of the system services. Written as a contract between client and contractor
- ✓ Software specification
  - A detailed software description which can serve as a basis for a design or implementation. Written for developers

GOALS VS. REQUIREMENTS

- ✓ Goal
  - A general intention of the user such as ease of use.
  - usually high-level and describe the desired end result of a project
  - Often less volatile
- ✓ Requirements are often
  - Concrete
  - Measureable
  - Testable

GOALS VS. REQUIREMENTS (CONT.)

Goal:
The system should be easy to use by medical staff.

Non-functional requirement:
Medical staff shall be able to use all the system functions after four hours of training.

SYSTEM STAKEHOLDERS

- ✓ Any person or organization who is affected by the system in some way and so who has a legitimate interest
- ✓ Stakeholder types
  - End users
  - System managers
  - System owners
  - External stakeholders

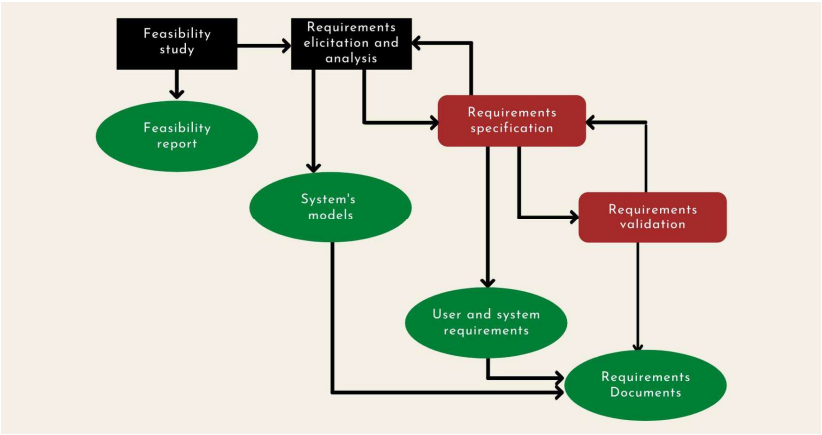
STAKEHOLDERS IN THE MENTCARE SYSTEM

Stakeholder	Why? - Role
Patients	whose information is recorded in the system
Doctors	responsible for assessing and treating patients
Nurses	coordinate the consultations with doctors and administer some treatments
Medical receptionists	manage patients' appointments
IT staff	responsible for installing and maintaining the system
Medical ethics manager	ensure that the system meets current ethical guidelines for patient care
Health care managers	obtain management information from the system
Medical records staff	responsible for ensuring that system information can be maintained and preserved, and that record keeping procedures have been properly implemented.

TOPICS COVERED

- ✓ Course's project – UWC 2.0
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Requirements specification
- ✓ Requirements validation
- ✓ Requirements change

A TYPICAL VIEW OF THE PROCESS

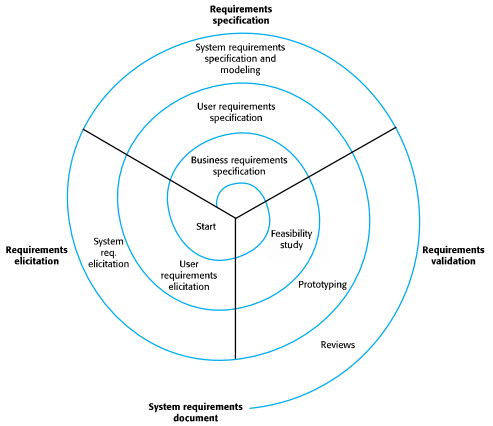


REQUIREMENTS ENGINEERING PROCESSES

- ✓ Processes to "generate" all requirements
- ✓ Generic activities common to all processes
  - Requirements elicitation;
  - Requirements analysis;
  - Requirements validation;
  - Requirements management.
- ✓ In practice, RE is an iterative activity

"The client kept changing the requirements on a daily basis, so we decided to freeze them until the next release."

A SPIRAL VIEW OF THE PROCESS



TOPICS COVERED

- ✓ Course's project – UWC 2.0
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Requirements specification
- ✓ Requirements validation
- ✓ Requirements change

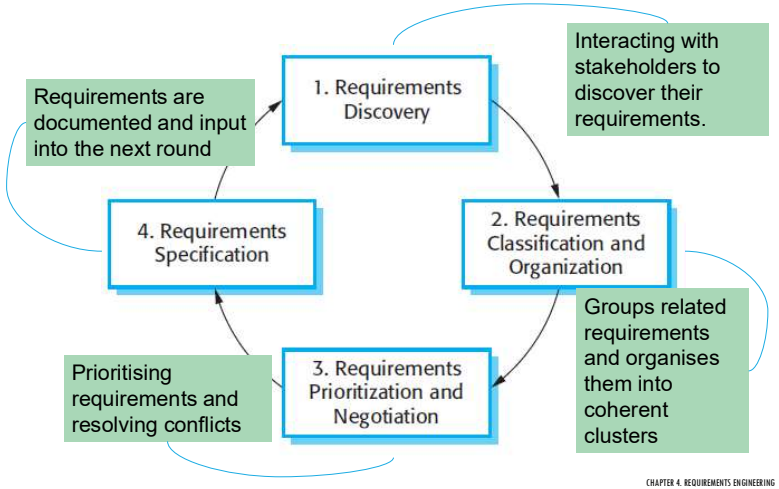
REQUIREMENTS ELICITATION AND ANALYSIS

- ✓ ~ requirements elicitation or requirements discovery.
- ✓ Work with customers to find out:
  - the application domain, the services and the operational constraints (system performance, hardware constraints, etc.).
- ✓ May involve
  - end-users, managers, engineers involved in maintenance, domain experts, trade unions, etc. These are called stakeholders.

PROBLEMS OF REQUIREMENTS ELICITATION

- ✓ Stakeholders don't know what they really want.
- ✓ Stakeholders express requirements in their own terms.
- ✓ Different stakeholders may have conflicting requirements.
- ✓ Organisational and political factors may influence the system requirements.
- ✓ The requirements change during the analysis process.
  - New stakeholders may emerge and the business environment change.

THE REQUIREMENTS ELICITATION AND ANALYSIS PROCESS



REQUIREMENTS DISCOVERY

- ✓ To gather information about the required and existing systems and distil the user and system requirements from this information.
- ✓ Main concerns:
  - Stakeholders
  - Discovery techniques/approaches/…

DISCOVERY TECHNIQUE - INTERVIEWING

- ✓ Part of most RE processes.
- ✓ Types of interview
  - Closed vs Open => mixed?
- ✓ Be effective
  - Be open-minded, avoid pre-conceived ideas about the requirements and are willing to listen to stakeholders.
  - Prompt the interviewee to get discussions going using a springboard question, a requirements proposal, or by working together on a prototype system.

DISCOVERY TECHNIQUE - ETHNOGRAPHY

- ✓ Observational technique
  - used to understand operational processes and help derive support requirements for these processes
- ✓ How
  - A social scientist spends a considerable time observing and analysing how people actually work.
  - People do not have to explain or articulate their work.
  - Social and organisational factors of importance may be observed.

REQUIREMENT ELICITATION TECHNIQUES

	Interviews	Workshops	Focus groups	Observations	Questionnaires	System interface analysis	User interface analysis	Document analysis
Mass-market software	x		x		x			
Internal corporate software	x	x	x	x		x		x
Replacing existing system	x	x		x		x	x	x
Enhancing existing system	x	x				x	x	x
New application	x	x				x		
Packaged software implementation	x	x		x		x		x
Embedded systems	x	x				x		x
Geographically distributed stakeholders	x	x			x			

Suggested elicitation techniques by project characteristics

TOPICS COVERED

- ✓ Course's project – UWC 2.0
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Requirements specification
- ✓ Requirements validation
- ✓ Requirements change

SOFTWARE ENGINEERING

CO3001

CHAPTER 4 – REQUIREMENTS ENGINEERING

Anh Nguyen-Duc
Tho Quan Thanh

Adapted from https://iansommerville.com/software-engineering-book/slides/

EXERCISE: 15 MINS (+ 10 MINS BREAK)

- ✓ Watch the clip: https://www.youtube.com/watch?v=J9p47AO0Xco
- ✓ Imagine that each employee will install a new Messenger App as a part of the UWC 2.0 system.

Task:

- Identify the stakeholders of the app
- Specify 03 functional requirements, at least one functional requirement extracted from the clip. Write them in a natural language approach
- Specify 02 non-functional requirements you can think of! Make sure they are testable!
- Document your answers:

https://docs.google.com/spreadsheets/d/1JAkNwIClopI4gjk_0cP9fJq-54TS1zc004gb3Q5mahI/edit?usp=sharing

TOPICS COVERED

- ✓ Course's project – UWC 2.0
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Quality of requirements
- ✓ Requirements specification
- ✓ Requirement in Agile
- ✓ Requirements validation
- ✓ Requirements change

EXAMPLE OF NIGHTMARE REQUIREMENTS

- ✓ The system shall perform at the maximum rating at all times except that in emergencies it shall be capable of providing up to 125% rating unless the emergency condition continues for more than 15 minutes in which case the rating shall be reduced to 105% but in the event that only 95% can be achieved then the system shall activate a reduced rating exception and shall maintain the rating within 10% of the stated values for a minimum of 30 minutes.

More than one requirement in a paragraph
Rambling: like a novel
Vague terms: emergencies

USE CASES

Notation	Description
Graphical notations	Graphical models, supplemented by text annotations (best for functional requirements); UML use case and sequence diagrams are commonly used.

- ✓ Use-cases are a kind of scenario
  - identify the actors in an interaction and which describe the interaction itself
  - Included in the UML
- ✓ A set of use cases should describe all possible interactions with the system.
- ✓ UML sequence diagrams may be used to add detail to use-cases
  - show the sequence of event processing in the system

ANOTHER EXAMPLE OF NIGHTMARE REQUIREMENTS

- ✓ The system shall provide general word processing facilities which shall be easy to use by untrained staff and shall run on a thin Ethernet Local Area Network wired into the overhead ducting with integrated interface cards housed in each system together with additional memory if that should be necessary.

More than one requirement in a paragraph
Rambling: like a novel
Let-out clauses: if that should be necessary
Vague words: be easy to use, additional memory

COMPLETENESS

- ✓ Definition of the responses of the software to all realizable classes of input data in all realizable classes of situations (IEEE Std).
- ✓ All possible situations must be covered

  "If X then….", "If Y then…." Must also consider what will happen "If neither X nor Y…"

COMPLETENESS (CONT')

- ✓ Automatic door opener

If the door is closed and a person is detected, then send signal Open_Door.

If no person is detected after 10 sec., send signal Close_Door.

- ✓ What are missing?

UNAMBIGUOUS (CONT')

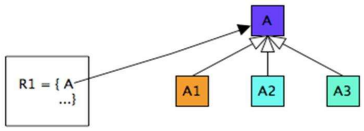
- ✓ Requirement of autonomous cruise control (ACC) system is:

The maximum speed of a vehicle in a busy road shall be 10 mph.

Ambiguity: What type of vehicle do you mean?
What is busy road?

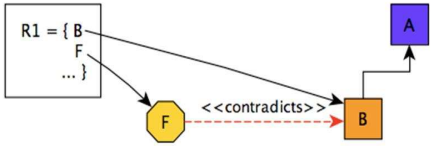
UNAMBIGUOUS

- ✓ A software requirement is unambiguous, if and only if, every requirement stated therein has only one interpretation (IEEE Std).
  - Ambiguity: Requirement with terms or statements that can be interpreted in different ways.



CONSISTENT

- ✓ A software requirement is internally consistent if, and only if, no subset of individual requirements described in is conflict.



CONSISTENT (CONT')

- Two requirements of the PROFIBUS network are in conflict
  - The PROFIBUS shall have a short reaction time of 60ms
  - PROFIBUS used in hazardous area shall have a low power dissipation of 3.6W

Conflict: Fast PROFIBUS has to have a high power

CORRECT

✓ Possible incorrect requirements
- Forward referencing
- Opacity
- Noise
- Etc.

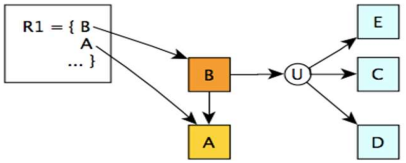
MAIN TYPES OF CONFLICTS

✓ Three main types of conflicts in software requirements
- Specific characteristics of real-world objects
- The logical or temporal conflict between two actions
- Different terms for describing the same real-world object

- Consistency is a challenge since we, at least in the general case, need a complete overview of all requirements that are related to the same event, function, or parameter.

FORWARD REFERENCING

✓ Requirement items that make use of problem world domain features that are not yet defined.



E, C, and D need to be mapped to a requirement item

FORWARD REFERENCING (CONT')

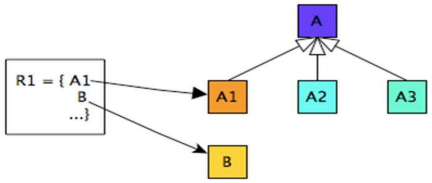
✓ A requirement of autonomous cruise control (ACC) system is:

ACC system shall maintain the preset speed of an ego-vehicle if there is no forward vehicle.

- Missing inference
  - Who sets the preset speed?
  - What is the value of the preset speed?

OPACITY

✓ Requirement items for which rational or dependencies are invisible.



Multiple unrelated concept mapping. A is not related to B

OPACITY (CONT')

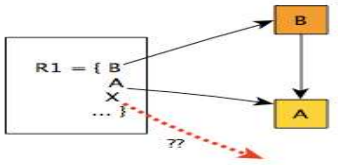
✓ Requirement:

Each time the freight train (Freight trains are trains used for carrying goods) doors are closed, the passengers must all seated.

- Opacity
  - There is no visible relationship between the freight trains and passengers.

NOISE

✓ Requirement items that yield no information on problem world features.



X refers to a concept undefined in the domain

NOISE (CONT')

- ✓ Requirement: The train system shall guarantee safe transportation of all passengers on their residence.

- Noise
  - A residence is an unknown concept within the train domain
  - The train can only transport passengers to the train station and not to their residence

TOPICS COVERED

- ✓ Course's project – UWC 2.0
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Quality of requirements
- ✓ Requirements specification
- ✓ Requirement in Agile
- ✓ Requirements validation
- ✓ Requirements change

REQUIREMENTS SPECIFICATION

REQUIREMENTS SPECIFICATION

- ✓ The process of writing down the user and system requirements in a requirements document.

- ✓ Notes:
  - User requirements have to be understandable by end-users and customers who do not have a technical background.
  - System requirements are more detailed requirements and may include more technical information.
  - The requirements may be part of a contract for the system development

WAYS OF WRITING A SYSTEM REQUIREMENTS SPECIFICATION

Notation	Description
Natural language	Sentences in natural language. Each sentence should express one requirement.
Structured natural language	Natural language statements on a standard form or template
Design description languages	Uses a language like a programming language, but with more abstract features
Graphical notations	Graphical models, supplemented by text annotations (best for functional requirements); UML use case and sequence diagrams are commonly used.
Mathematical specifications	Based on mathematical concepts such as finite-state machines or sets; Can reduce the ambiguity but hard to understand (and hard to check manually)

WAYS OF WRITING A SYSTEM REQUIREMENTS SPECIFICATION

Notation	Description
Natural language	Sentences in natural language. Each sentence should express one requirement.

Waste collection is often designated to an organization that provides professional waste management services. A typical waste collection process involves (1) back officers, who operate a central system to create calendar, coordinate front collectors and janitors ,(2) collectors, who drive different types of vehicles and (3) janitors who manually collect garbage from Major Collecting Points (MCPs). Calendar and tasks were assigned among teams of janitors and coordinated by back officers. These assignments are often arranged in a weekly basic. Back officers also plan which vehicles to use and their routes. This planning activity happens every month. Everyday, the back officers sent messages with information about collecting route and time to collectors and janitors. Janitors use trollers (see Figure 1b) to collect garbage in their assigned areas and deliver to the MCPs. Collectors will pick up garbage from all janitors at an MCP. One collector drives only one vehicle during his working shift. The collector will drive through several MCPs with a predetermined route by back officers.

WAYS OF WRITING A SYSTEM REQUIREMENTS SPECIFICATION

Notation	Description
Natural language	Sentences in natural language. Each sentence should express one requirement.

- ✓ Used for writing requirements because it is expressive, intuitive and universal.
  - The requirements can be understood by users and customers.
- ✓ Problems
  - Lack of clarity: Precision is difficult without making the document difficult to read.
  - Requirements confusion: Functional and non-functional requirements tend to be mixed-up.
  - Requirements amalgamation: Several different requirements may be expressed together.

STRUCTURED SPECIFICATIONS

Notation	Description
Structured natural language	Natural language statements on a standard form or template

- ✓ Writing on a standard form or template:
  - Name
  - Inputs, outputs
  - The information needed for the computation
  - Action
  - Pre and post conditions (if appropriate)
  - The side effects (if any)
- ✓ Writing in a boiler plate template

A S… PUMP

Insulin Pump/Control Software/SRS/3.3.2	
Function	Compute insulin dose: safe sugar level
Description	Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units.
Inputs	Current sugar reading (r2); the previous two readings (r0 and r1).
Source	Current sugar reading from sensor. Other readings from memory.
Outputs	CompDose—the dose in insulin to be delivered
Destination	Main control loop.
Action	CompDose is zero if the sugar level is stable or falling or if the level is increasing but the rate of increase is decreasing. If the level is increasing and the rate of increase is increasing, then CompDose is computed by dividing the difference between the current sugar level and the previous level by 4 and rounding the result. If the result, is rounded to zero then CompDose is set to the minimum dose that can be delivered.
Requirements	Two previous readings so that the rate of change of sugar level can be computed
Pre-condition	The insulin reservoir contains at least the maximum allowed single dose of insulin.
Post-condition	r0 is replaced by r1 then r1 is replaced by r2.
Side effects	None

CHARACTERISTICS OF GOOD SOFTWARE REQUIREMENTS SPECIFICATION*

- ➢ Complete
- ➢ Unambiguous
- ➢ Consistent
- ➢ Correct
- ✓ Verifiable
- ✓ Traceable
- ✓ Ranked for importance and/or stability
- ✓ Modifiable

* IEEE Std 830-1998

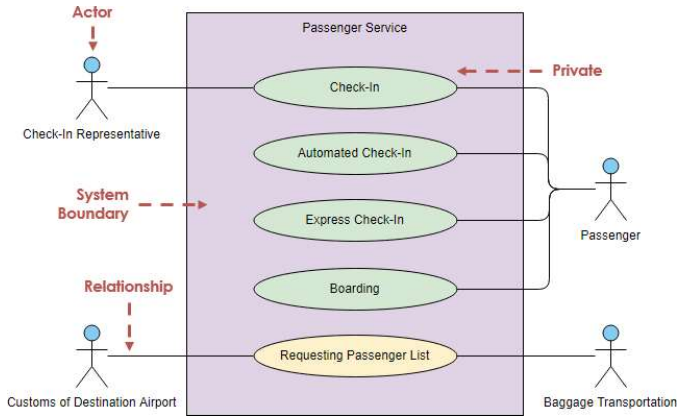
TABULAR SPECIFICATION

- ✓ Particularly useful when you have to define a number of possible alternative courses of action.
- ✓ Example:

Condition	Action
Sugar level falling (r2 < r1)	CompDose = 0
Sugar level stable (r2 = r1)	CompDose = 0
Sugar level increasing and rate of increase decreasing ((r2 – r1) < (r1 – r0))	CompDose = 0
Sugar level increasing and rate of increase stable or increasing ((r2 – r1) ≥ (r1 – r0))	CompDose = round ((r2 – r1)/4) **If** rounded result = 0 **then** CompDose = MinimumDose

USE CASE DIAGRAM

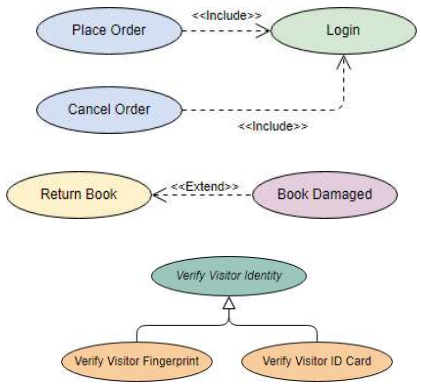
- Actors
- Use cases
- Association
- System boundary boxes



Partial use case diagram for the Chemical Tracking System (CTS)

USE CASE DIAGRAM

- Association types



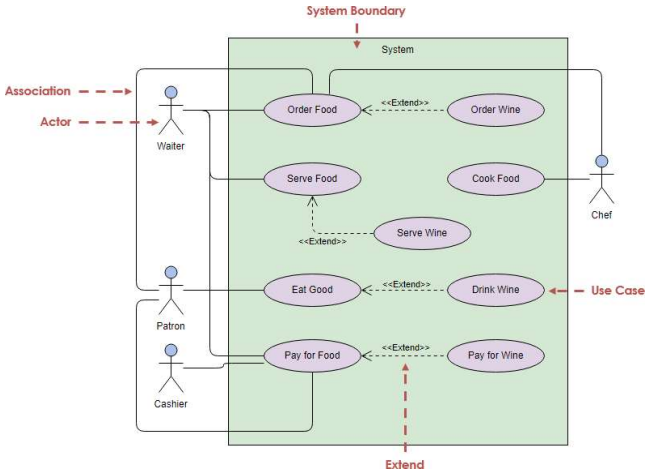
Partial use case diagram for the Chemical Tracking System (CTS)

USE CASE TEMPLATE AND EXAMPLE

Use-case ID Use-case name	U4. Overview MCP.
Use-case overview	To provide information about Major Collecting Points (MCPs) and their current capacity.
Actors	Back officers.
Preconditions	1. The system is running. 2. Database is connected to MCPs. 3. Internet connection is available.
Trigger	Users click the "MCPs' overview" button.
Steps	1. Retrieve all MCPs' information and capacity. 2. Display all information on the screen of users' devices. 3. Update MCPs' capacity every 15 minutes, then retrieve new capacities from the database and overwrite the old capacities with the new ones.
Post conditions	Required information are displayed on the screen of users' devices and are updated every 15 minutes.
Exception flow	None

USE CASE DIAGRAM

- Association types

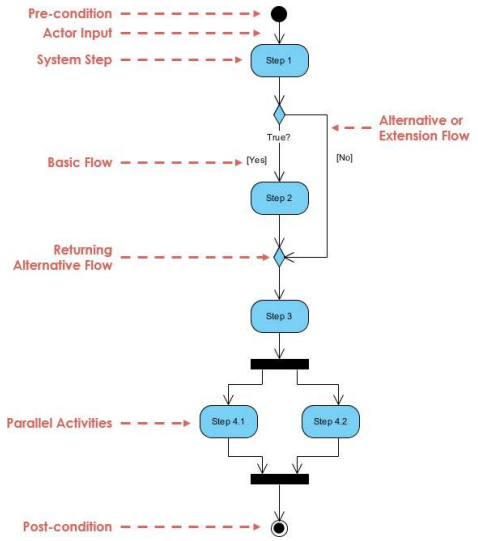


Partial use case diagram for the Chemical Tracking System (CTS)

- ✓ Draw a use-case diagram for the whole UWC 2.0 system (not only the Task management module)
- ✓ https://docs.google.com/presentation/d/1NNqqt_LuHjvWQGs5ifE6s88UPA77fG6cmTzmY0olULU/edit#slide=id.p

ACTIVITY DIAGRAM

1. Identify candidate use cases, through the examination of business workflows
2. Identify pre- and post-conditions (the context) for use cases
3. Model workflows between/within use cases
4. Model complex workflows in operations on objects
5. Model in detail complex activities in a high level activity Diagram

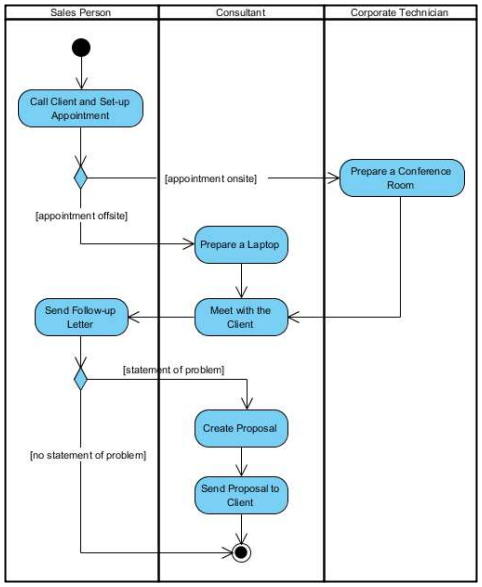


SEQUENCE DIAGRAM

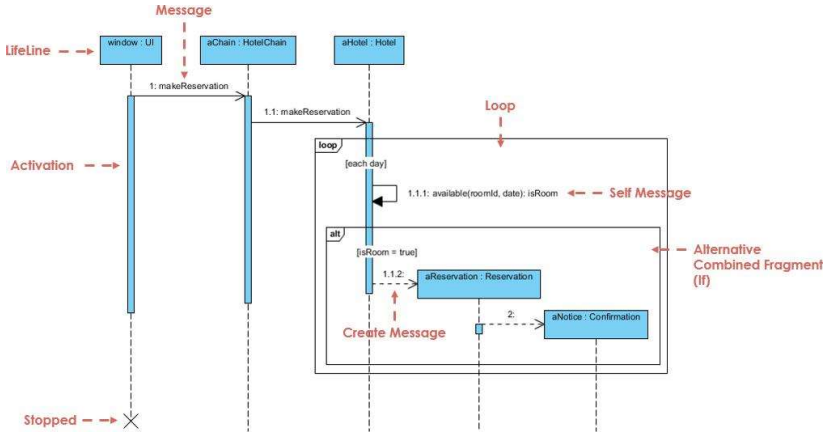
- ✓ an interaction diagram that details how operations are carried out -- what messages are sent and when
- ✓ Sequence diagrams are organized according to time.

ACTIVITY DIAGRAM

- ✓ https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-activity-diagram/



SEQUENCE DIAGRAM



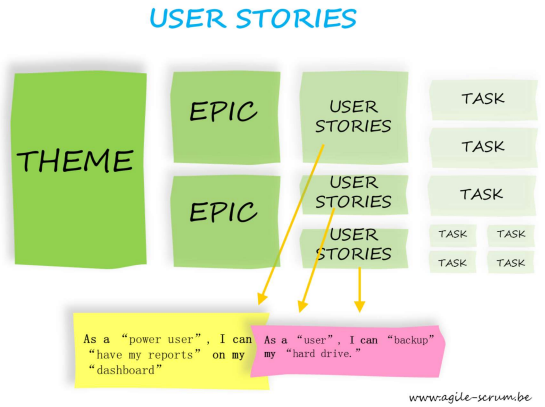
TOPICS COVERED

- ✓ Course's project – UWC 2.0
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Quality of requirements
- ✓ Requirements specification
- ✓ Requirement in Agile
- ✓ Requirements validation
- ✓ Requirements change

AGILE METHODS AND REQUIREMENTS

- ✓ Many Agile methods argue that producing detailed system requirements is a waste of time as requirements change so quickly.
- ✓ The requirements document is therefore always out of date.
- ✓ Agile methods usually use incremental requirements engineering and may express requirements as 'user stories'
- ✓ This is practical for business systems but problematic for systems that require pre-delivery analysis (e.g. critical systems) or systems developed by several teams.

AGILE METHODS AND REQUIREMENTS – USER STORY



USER STORY

What is a user story?

A user story is a short, simple description of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system. User stories typically follow a simple template:

As a < type of user >, I want < some goal > so that < some reason >.

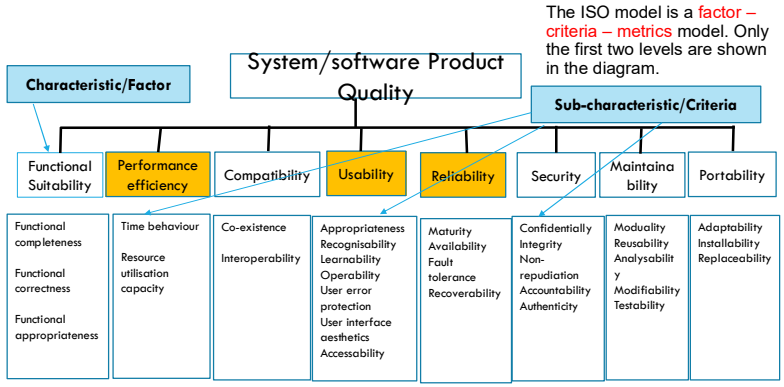
User stories were historically written on index cards or sticky notes, stored in a shoe box, and arranged on walls or tables to facilitate planning and discussion. Nowadays, they might just as easily be stored in a Jira issue.

User stories are designed to strongly shift the focus from writing about features to discussing them. In fact, these discussions are more important than whatever text is written.

AGILE METHODS AND REQUIREMENTS – USER STORY

- ✓ As a medical record stuff, I wish to view current medication from primary care general practice
- ✓ As a medical receptionist, I want to search appointments by patients' first name, their telephone number, their date of birth or their prescribed medicine
- ✓ As a healthcare manager, I want to see the monthly report including number of prescribed medicines ordered by their categories, by weeks, and by cost

ISO/IEC 25010 PRODUCT QUALITY MODEL



MORE ABOUT NON-FUNCTIONAL REQUIREMENTS (NFR)

- ✓ Many definitions can be found of NFR
- ✓ A NFR is basically an attribute of or a constraint on a system [1]
  - Attributes
    - **-ilities**: understandability, reliability, portability, flexibility, availability, maintainability, scalability, …
    - **-ities**: security, simplicity, clarity, ubiquity, integrity, safety, modularity, …
    - **-ness**: user-friendliness, robustness, timeliness, responsiveness, …
  - Constraint
    - **Physical**
    - **Environmental**
    - **Legal**
    - **Cultural**
    - …

TOPICS COVERED

- ✓ Course's project – UWC 2.0
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Quality of requirements
- ✓ Requirements specification
- ✓ Requirement in Agile
- ✓ Requirements validation
- ✓ Requirements change

REQUIREMENTS VALIDATION

- ✓ Requirements error can be costly !

Cost to Fix Requirements Error (in Ratios)			
Development Discipline/Phase	**Composite of Studies (NASA)***	**NASA Software#**	**Davis' Composite+**
Requirements	1 (baseline)	1 (baseline)	1 (baseline)
Design	5x	8x	2.5x – 5x
Code	10x	16x	5x – 10x
Test	50.5x	21x	Unit Test: 10x – 20x Acceptance Test: 25x – 50x
Post-Deployment	n/a	29x	100x – 200x

* NASA reviewed McGibbon (2003), Pavlina (2003), Cigital (2003), Rothman (2002), Hoffman (2001), Rothman (2000), and Boehm (1981) and assessed the median of their data, presented here.
# NASA analyzed its own software projects to assess the relevance of reviewed studies.
# Leffingwell & Widrig reviewed GTE, TRW, IBM, and Davis (2003), who himself reviewed several studies. The results of Davis' work are presented here.

REQUIREMENTS CHECKING

- ✓ Validity.
  - Does the system provide the functions which best support the customer's needs?
- ✓ Consistency.
  - Are there any requirements conflicts?
- ✓ Completeness.
  - Are all functions required by the customer included?
- ✓ Realism.
  - Can the requirements be implemented given available budget and technology
- ✓ Verifiability.
  - Can the requirements be checked?

REQUIREMENTS VALIDATION TECHNIQUES

- ✓ Requirements reviews
  - Systematic manual analysis of the requirements.
- ✓ Prototyping
  - Using an executable model of the system to check requirements.
- ✓ Test-case generation
  - Developing tests for requirements to check testability.

TOPICS COVERED

- ✓ Course's project
- ✓ Functional and non-functional requirements
- ✓ Requirements engineering processes
- ✓ Requirements elicitation
- ✓ Quality of requirements
- ✓ Requirements specification
- ✓ Requirement in Agile
- ✓ Requirements validation
- ✓ Requirements change

CHANGING REQUIREMENTS

- ✓ The business and technical environment of the system always changes after installation.
- ✓ The people who pay for a system and the users of that system are rarely the same people.
- ✓ Large systems usually have a diverse user community, with many users having different requirements and priorities that may be conflicting or contradictory.

REQUIREMENTS MANAGEMENT

- ✓ Requirements management is the process of managing changing requirements during the requirements engineering process and system development.
- ✓ New requirements emerge as a system is being developed and after it has gone into use.
- ✓ You need to keep track of individual requirements and maintain links between dependent requirements so that you can assess the impact of requirements changes. You need to establish a formal process for making change proposals and linking these to system requirements.

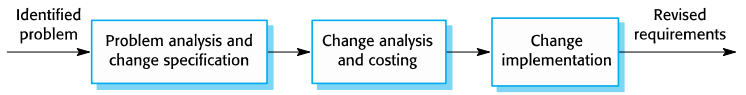
REQUIREMENTS MANAGEMENT PLANNING

- ✓ Establishes the level of requirements management detail that is required.

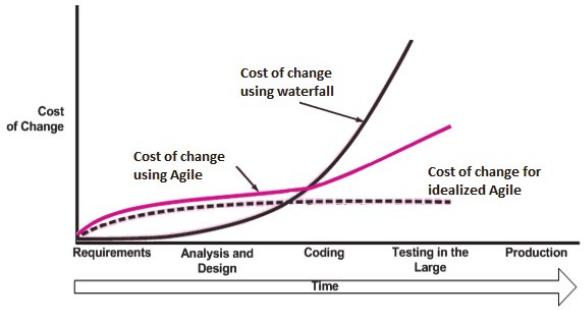
- ✓ Requirements management decisions:
  - Requirements identification
  - A change management process
  - Traceability policies
  - Tool support

REQUIREMENTS CHANGE MANAGEMENT

- ✓ Deciding if a requirements change should be accepted
  - Problem analysis and change specification
  - Change analysis and costing
  - Change implementation



RESPONSE TO CHANGE: AGILE APPROACH!



SUMMARY

- ✓ Requirements: what the system should do and constraints on its operation and implementation.
- ✓ Functional requirements = the services
- ✓ Non-functional requirements = constraints (development & use)
  - apply to the system as a whole.
- ✓ The software requirements document (i.e. SRS) is an agreed statement of the system requirements.
- ✓ The RE process is an iterative process
  - requirements elicitation, specification and validation.

SUMMARY (CONT.)

- ✓ Requirements elicitation and analysis = iterative process
  - requirements discovery, classification and organization, negotiation and requirements documentation.
- ✓ Techniques for requirements elicitation
  - interviews, scenarios, use-cases and ethnography, etc.
- ✓ Requirements validation = checking the requirements
  - for validity, consistency, completeness, realism and verifiability.
- ✓ Business, organizational and technical changes inevitably
  - => changes to the requirements for a software system.
- ✓ Requirements management = managing and controlling the requirement changes.