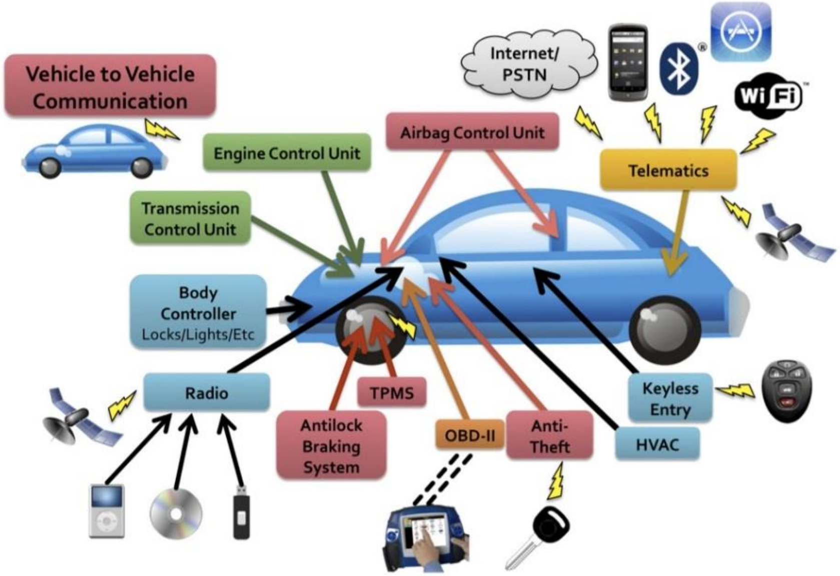
Engineering Software Intensive Systems

Software from a Systems Perspective, Development Processes, Requirements

Lecture 2

27 March 2025

Slides created by
Prof Amir Tomer
tomera@cs.technion.ac.il

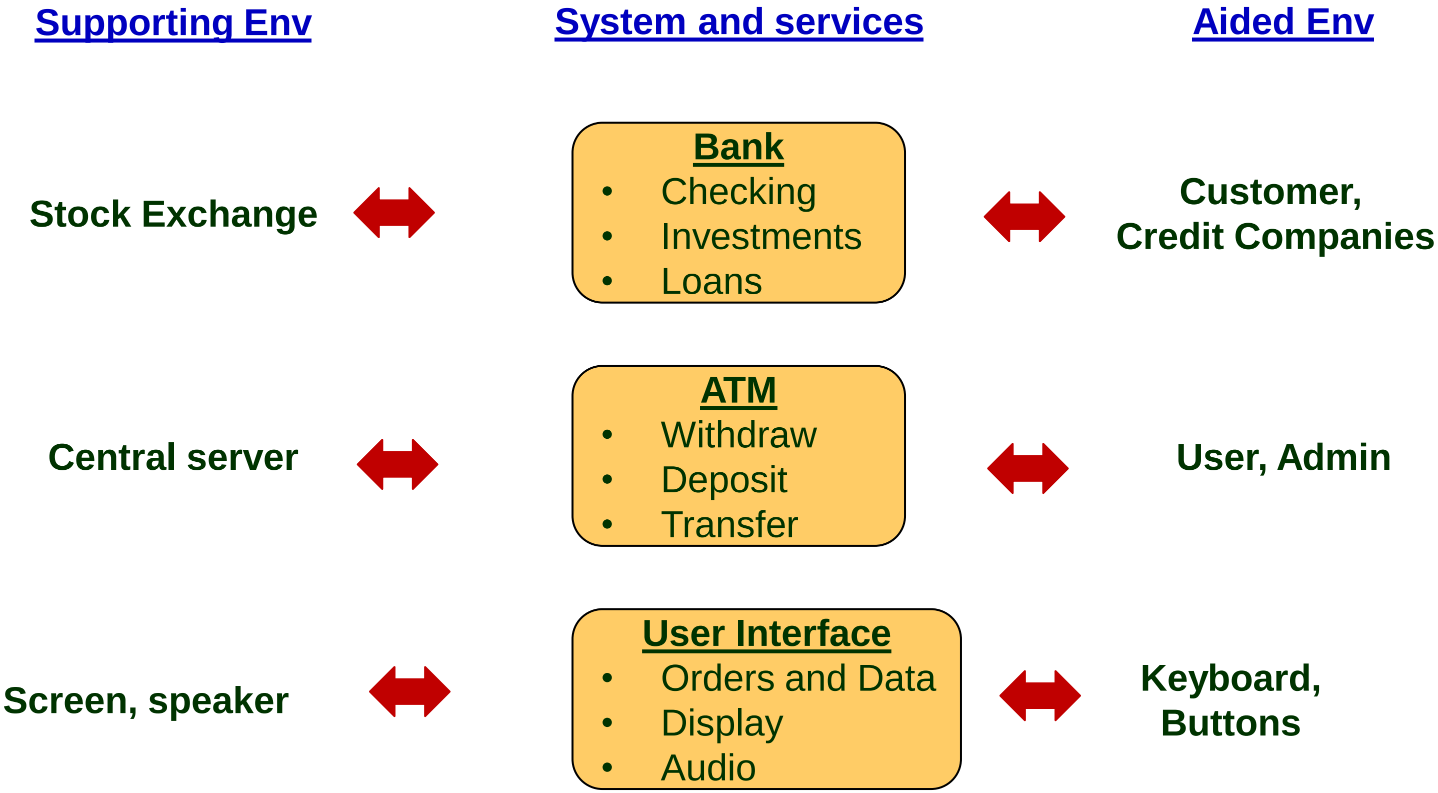


Topics for today

- Software from a Systems Perspective
- Development Process of a Software Intensive System
- Requirements and Stakeholders
- Elicitation and Management

A system in its environment

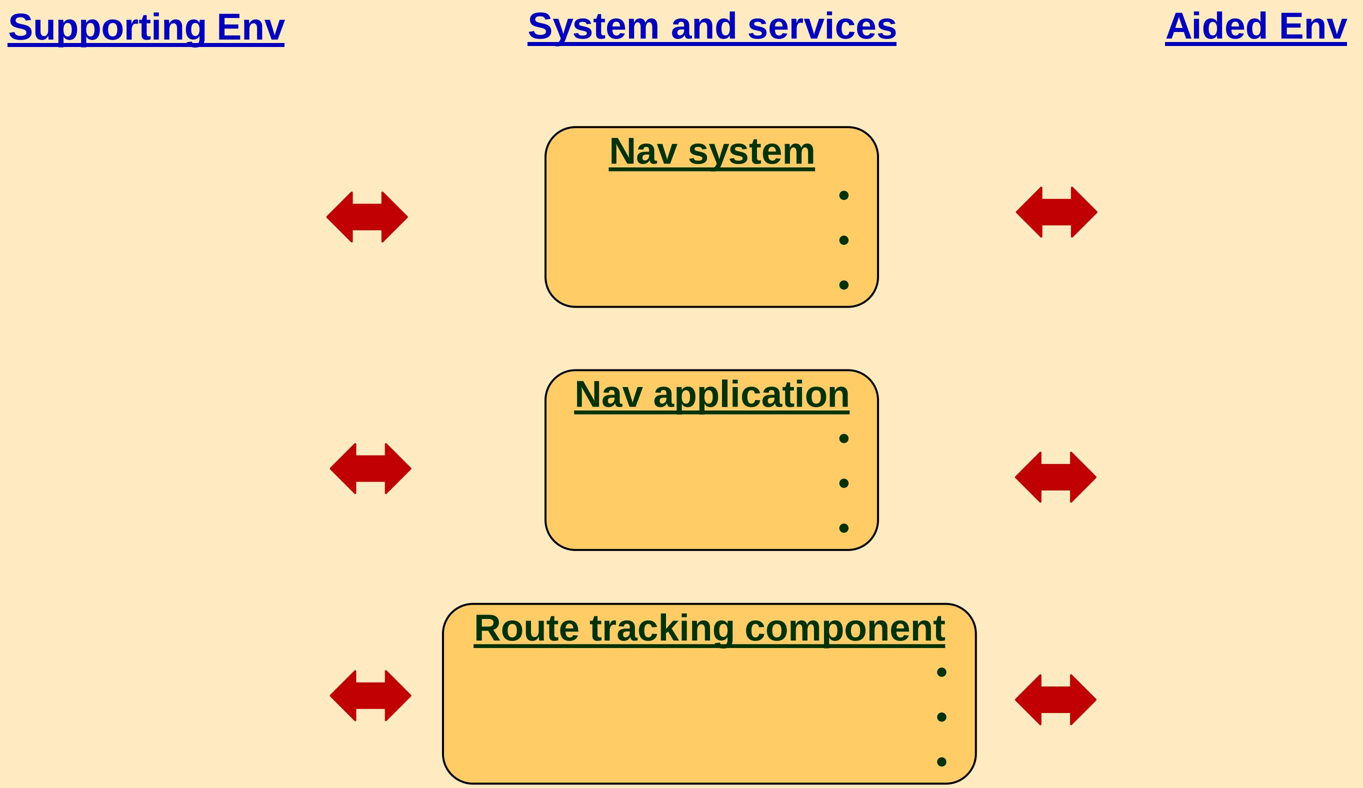
- The level (width) is the environment in which the system operates
  - Provides services to its environment (the aided environment)
  - Receives services from its environment (the supporting environment)



Supporting Env		System and services		Aided Env
Stock Exchange	⟺	**Bank** • Checking • Investments • Loans	⟺	Customer, Credit Companies
Central server	⟺	**ATM** • Withdraw • Deposit • Transfer	⟺	User, Admin
Screen, speaker	⟺	**User Interface** • Orders and Data • Display • Audio	⟺	Keyboard, Buttons

A system in its environment (exercise)

- For each of the systems related to **in-car navigation**, write the services, aided environment, and supporting environment



Supporting Env

System and services

Aided Env

Nav system
-
-
-

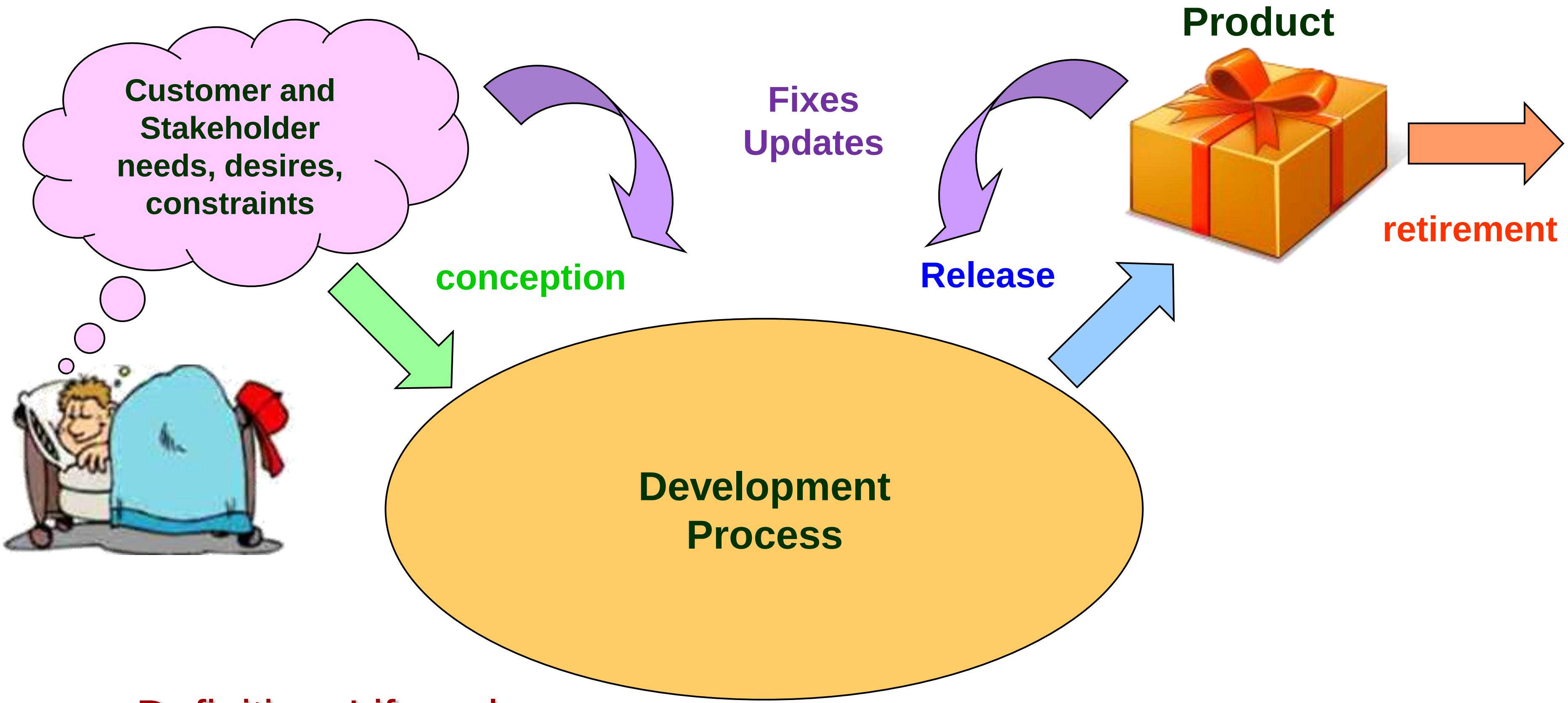
Nav application
-
-
-

Route tracking component
-
-
-

So Far

- Software from a Systems Perspective
- Development Process of a Software Intensive System
- Requirements and Stakeholders
- Elicitation and Management

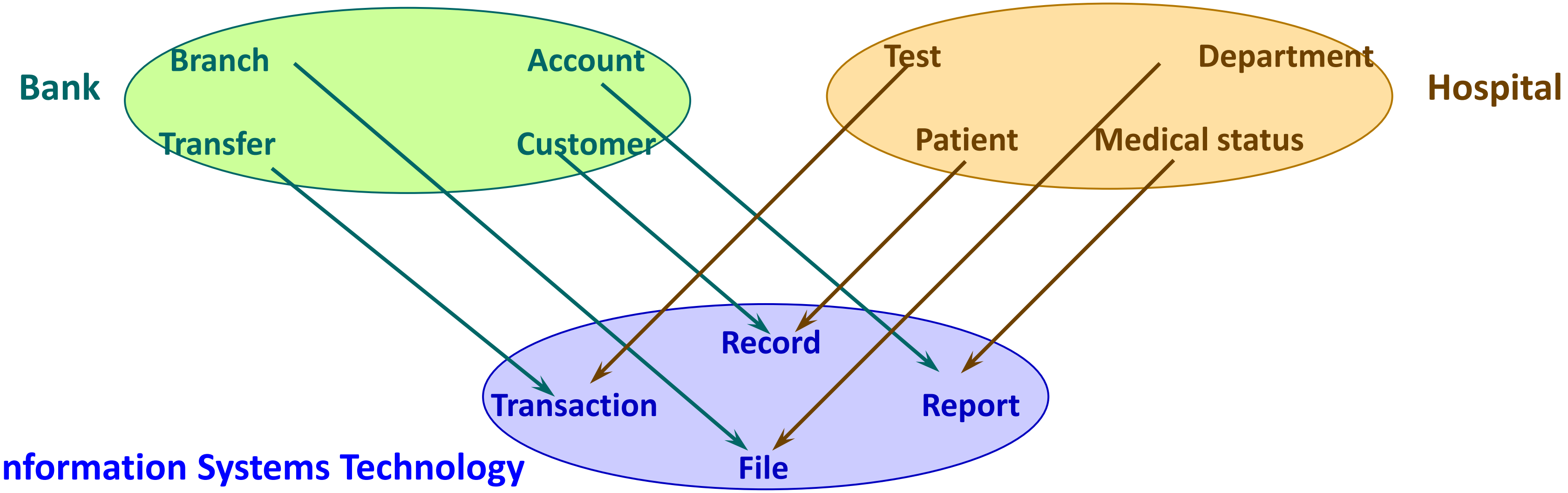
Software Lifecycle



- Definition: Lifecycle
    - Evolution of the system, product, service, project or other man-made entity from conception to retirement [ISO/IEC 15288]

What is a Development Process?

- A development process is meant to solve a problem or a need by mapping from the problem space to the solution space
  - Solution space: Technology, engineering results
  - Problem space varies from system to system
- Example: Information Systems Technology
  - Solution space: Data records, files, reports, transactions
  - Problem space 1: A bank = {branch, customer, account, transfers}
  - Problem space 2: A hospital = {department, test, patient, diagnosis}

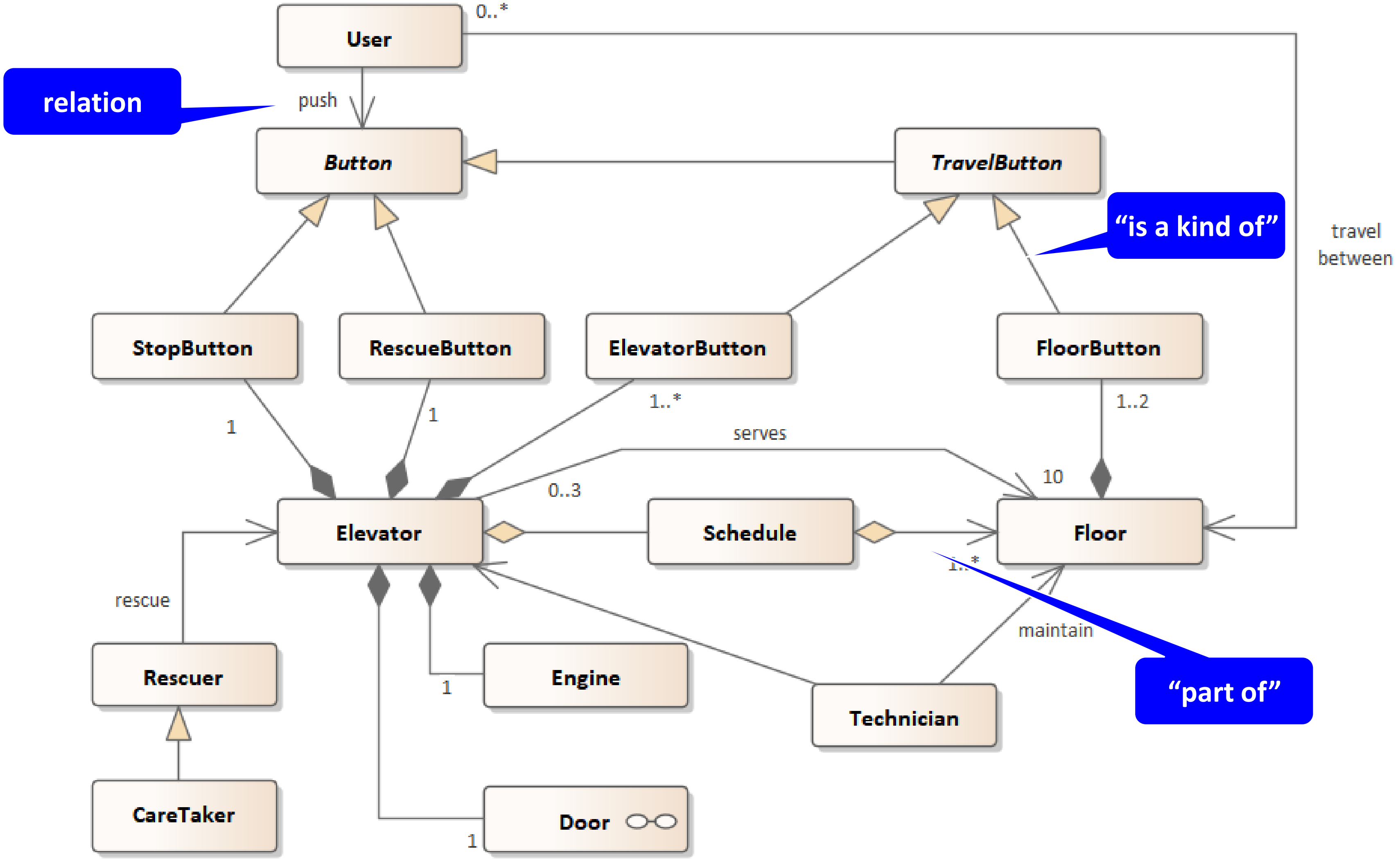


Information Systems Technology

Aside: Problem Domain Modelling

- How to give stakeholders a shared understanding of basic system concepts?
  - User
  - Button
  - Report
  - Input

Aside: Problem Domain Modelling



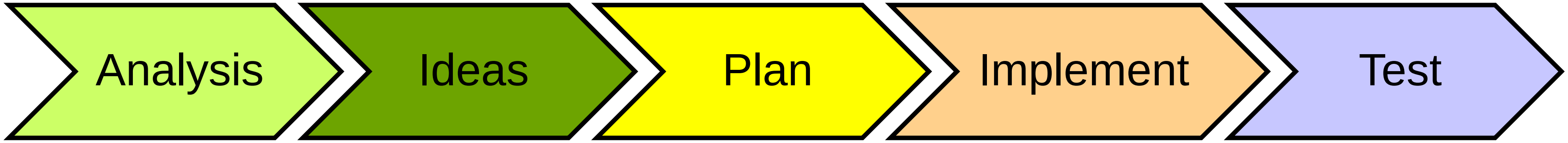
relation

“is a kind of”

“part of”

27 March 2025

Steps in the Development Process


Analysis

Ideas

Plan

Implement

Test

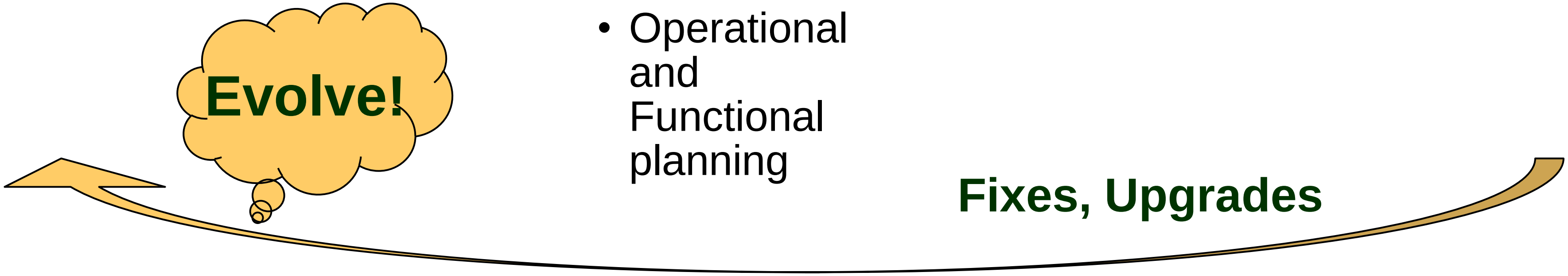
- Understand the problem or need

- Assemble the solution basics (system)
- Structural concept
- Operational concept

- Planning the solution
- System → Subsystem → Assemblies → Modules
- Structural planning
- Operational and Functional planning

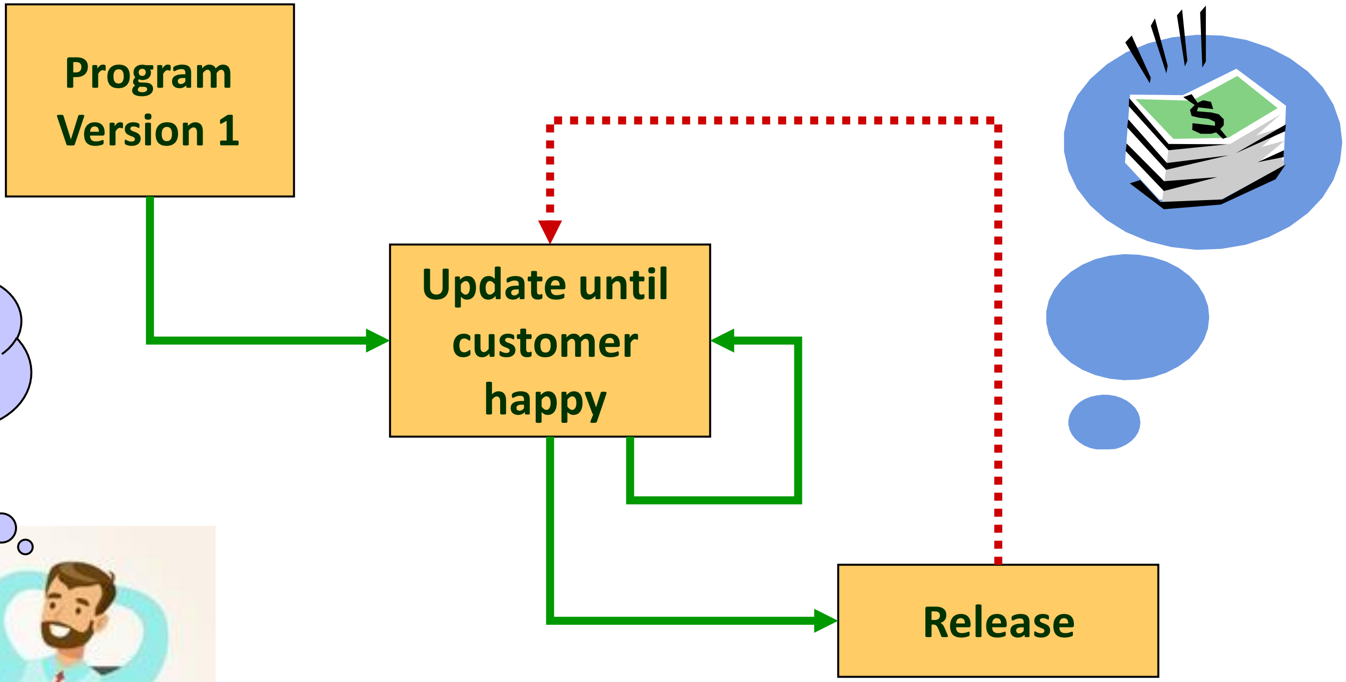
- Build ingredients
- Integration

- Validate against plans
- Validate against needs and problems


Evolve!

Fixes, Upgrades

Simple development process: Build & Fix


Program Version 1

Update until customer happy

Release

27 March 2025

Build & Fix

Uncontrolled evolution!

Process is not well defined

- Attempts to understand the problem by solving it via trial and error

Development without planning

- You know when you're going to start, no clue when you will finish

Uncontrolled process

- Requirements can change at any time

Build & Fix

Uncontrolled evolution!

Main risks

- Lots of reworking
- It’s never clear whether the final product will satisfy the customer

Why do people do it?

- There’s no time! Got to start right now and delive

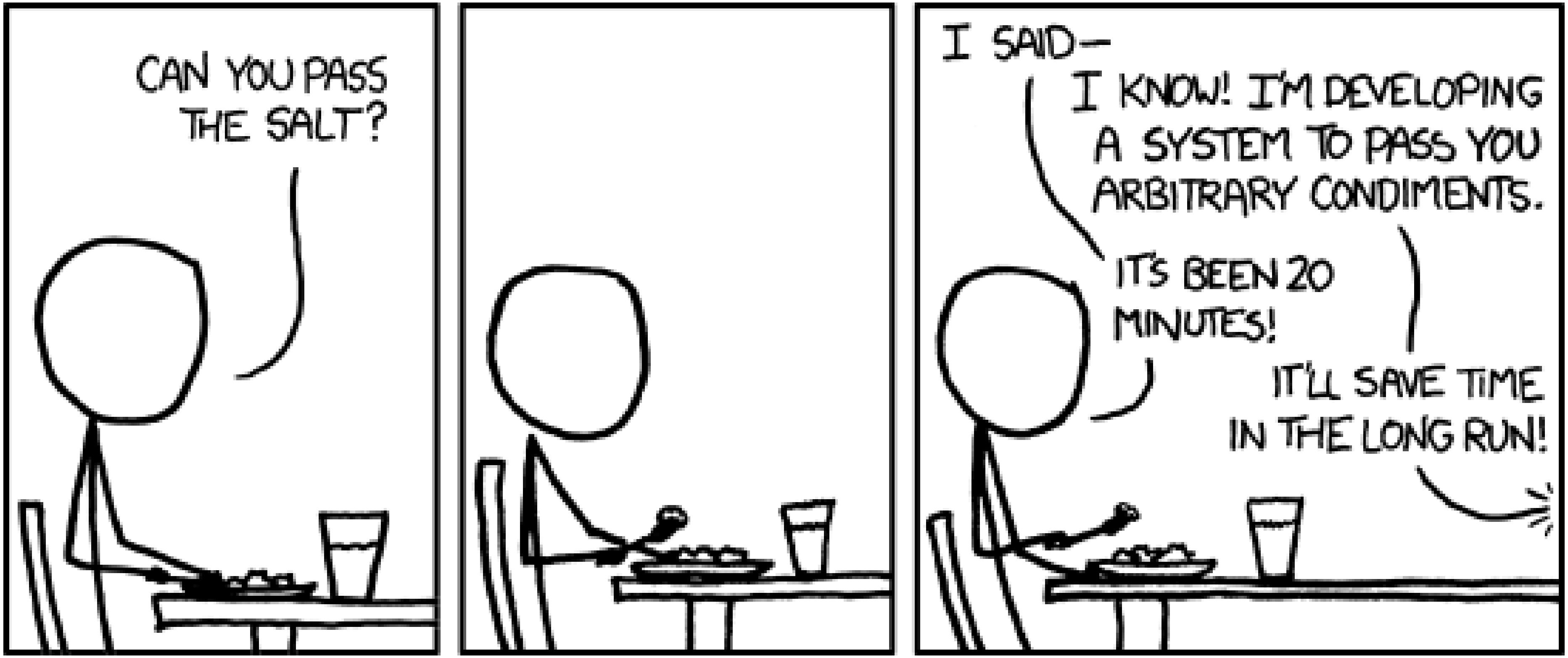
Main results

- The longer you’re at it, each change becomes more difficult
  - A technological house of cards
- Maintenance is impossible

If you don't have time to do it right, Where would you get the time to do it again???

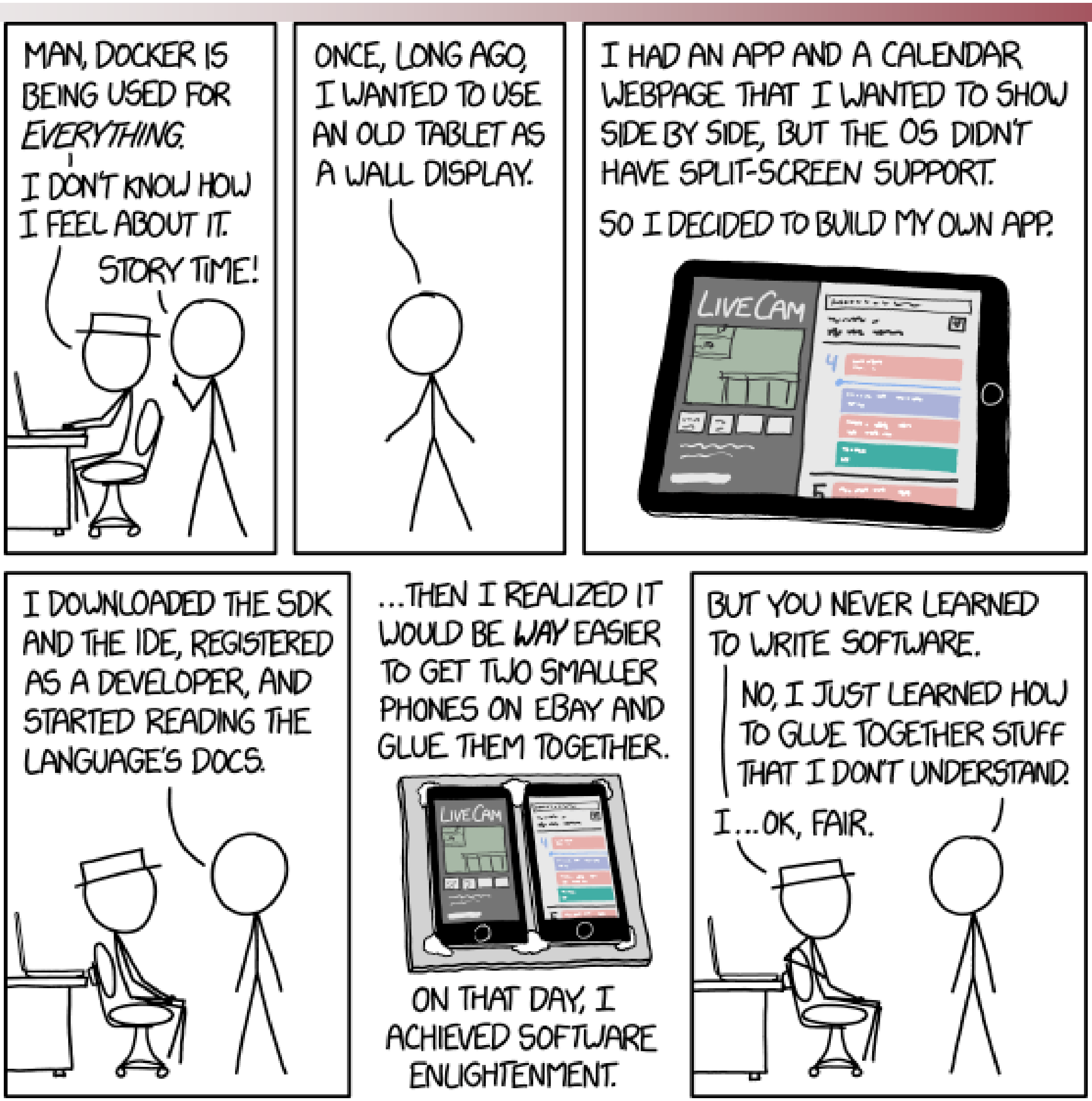
27 March 2025

https://xkcd.com/974/

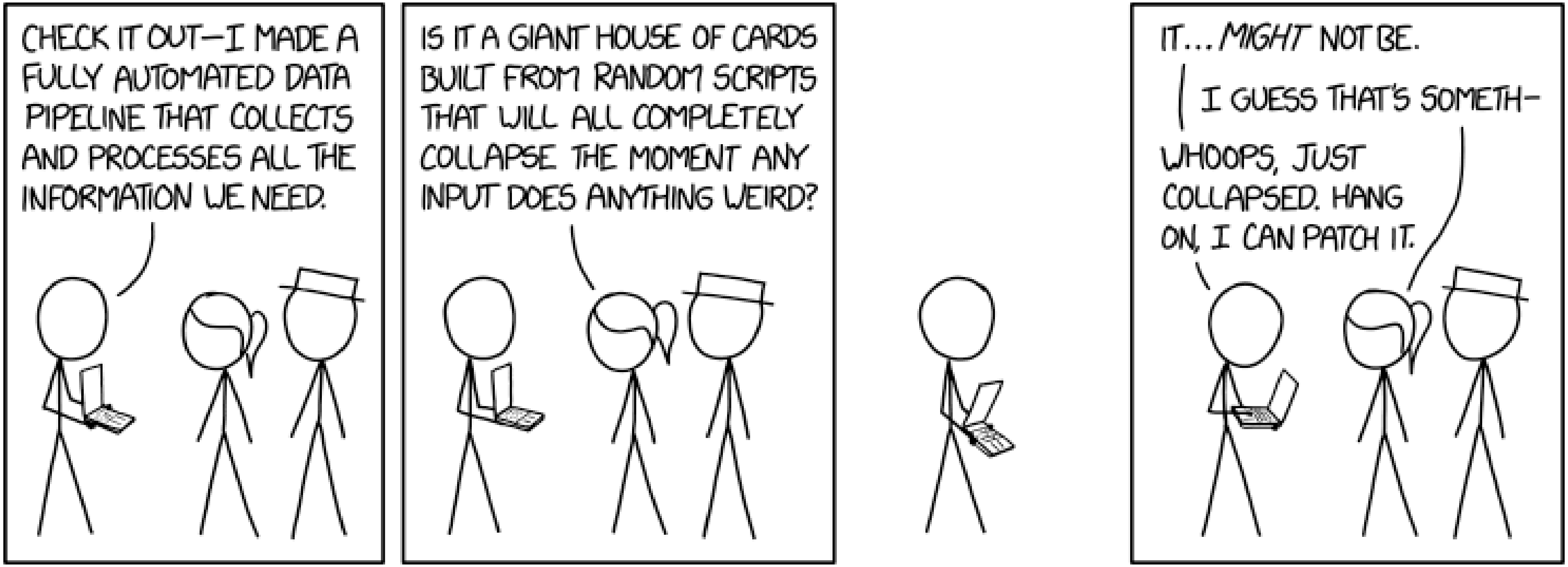

CAN YOU PASS THE SALT?
I SAID—
I KNOW! I'M DEVELOPING A SYSTEM TO PASS YOU ARBITRARY CONDIMENTS.
IT'S BEEN 20 MINUTES!
IT'LL SAVE TIME IN THE LONG RUN!

I find that when someone's taking time to do something right in the present, they're a perfectionist with no ability to prioritize, whereas when someone took time to do something right in the past, they're a master artisan of great foresight.

https://xkcd.com/1988/


MAN, DOCKER IS BEING USED FOR *EVERYTHING*.
I DON'T KNOW HOW I FEEL ABOUT IT.
STORY TIME!
ONCE, LONG AGO, I WANTED TO USE AN OLD TABLET AS A WALL DISPLAY.
I HAD AN APP AND A CALENDAR WEBPAGE THAT I WANTED TO SHOW SIDE BY SIDE, BUT THE OS DIDN'T HAVE SPLIT-SCREEN SUPPORT.
SO I DECIDED TO BUILD MY OWN APP.
LIVE CAM
I DOWNLOADED THE SDK AND THE IDE, REGISTERED AS A DEVELOPER, AND STARTED READING THE LANGUAGE'S DOCS.
...THEN I REALIZED IT WOULD BE *WAY* EASIER TO GET TWO SMALLER PHONES ON EBAY AND GLUE THEM TOGETHER.
LIVE CAM
ON THAT DAY, I ACHIEVED SOFTWARE ENLIGHTENMENT.
BUT YOU NEVER LEARNED TO WRITE SOFTWARE.
NO, I JUST LEARNED HOW TO GLUE TOGETHER STUFF THAT I DON'T UNDERSTAND.
I...OK, FAIR.

https://xkcd.com/2054/


CHECK IT OUT—I MADE A FULLY AUTOMATED DATA PIPELINE THAT COLLECTS AND PROCESSES ALL THE INFORMATION WE NEED.
IS IT A GIANT HOUSE OF CARDS BUILT FROM RANDOM SCRIPTS THAT WILL ALL COMPLETELY COLLAPSE THE MOMENT ANY INPUT DOES ANYTHING WEIRD?
IT... *MIGHT* NOT BE.
I GUESS THAT'S SOMETH–
WHOOPS, JUST COLLAPSED. HANG ON, I CAN PATCH IT.

27 March 2025

Building a house

Stakeholder Requirements:
Registration
Infrastructure (interfaces)

Implement, Integrate, Test

Customer needs, desires, wants
Living and Utilities
Appearance
Future potential
Budget and schedule constraints

Requirements and Architectural Specs
Walls, floors, roofs
Elevations
Water and power connections
Doors, stairs

Planning and Quantities
Construction plan
Infrastructure plans
(power, water, sewer)

Construction
Walls and floors
Electrical cabling, pipes
Finishings

Define and Detail

Building software

Stakeholder Requirements:
Standards
Interface requirements

Customer needs, desires, wants
Services, tasks
Performance
Future capabilities
Budget and schedule constraints

Requirements and Architectural Specs
Use cases
Logical architecture & interfaces
Data Entities
Operational concept

Planning and Quantities
Modules and physical interfaces
Databases and data structures
Algorithms
Protocols

Construction
Code, Compile
Link
Integrate

Implement, Integrate, Test

Define and Detail

Between building a house and building software

Speed of Evolution	Ease of change	Materials	Most importantly
• Houses change slowly • Software changes fast	• Hard to change ingredients or structure (“move this wall!”) • Easier to change software ingredients or structure	• Each house requires new materials • Software can be copied without additional materials	• Houses are static • It doesn’t work, it exists

A first model: Waterfall

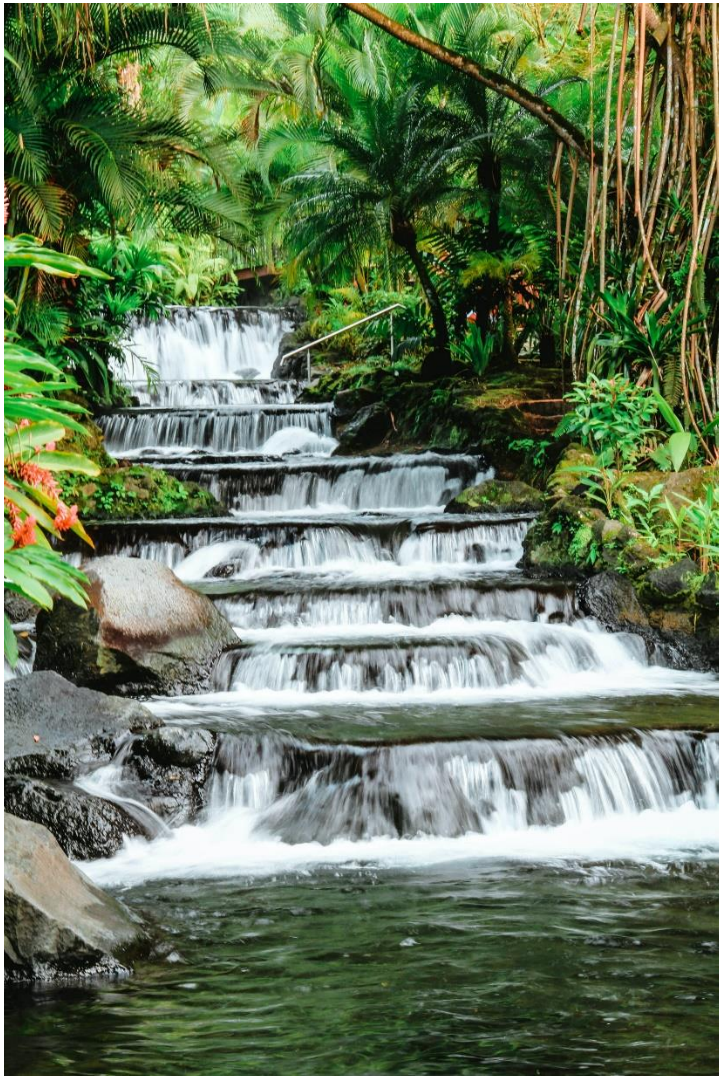
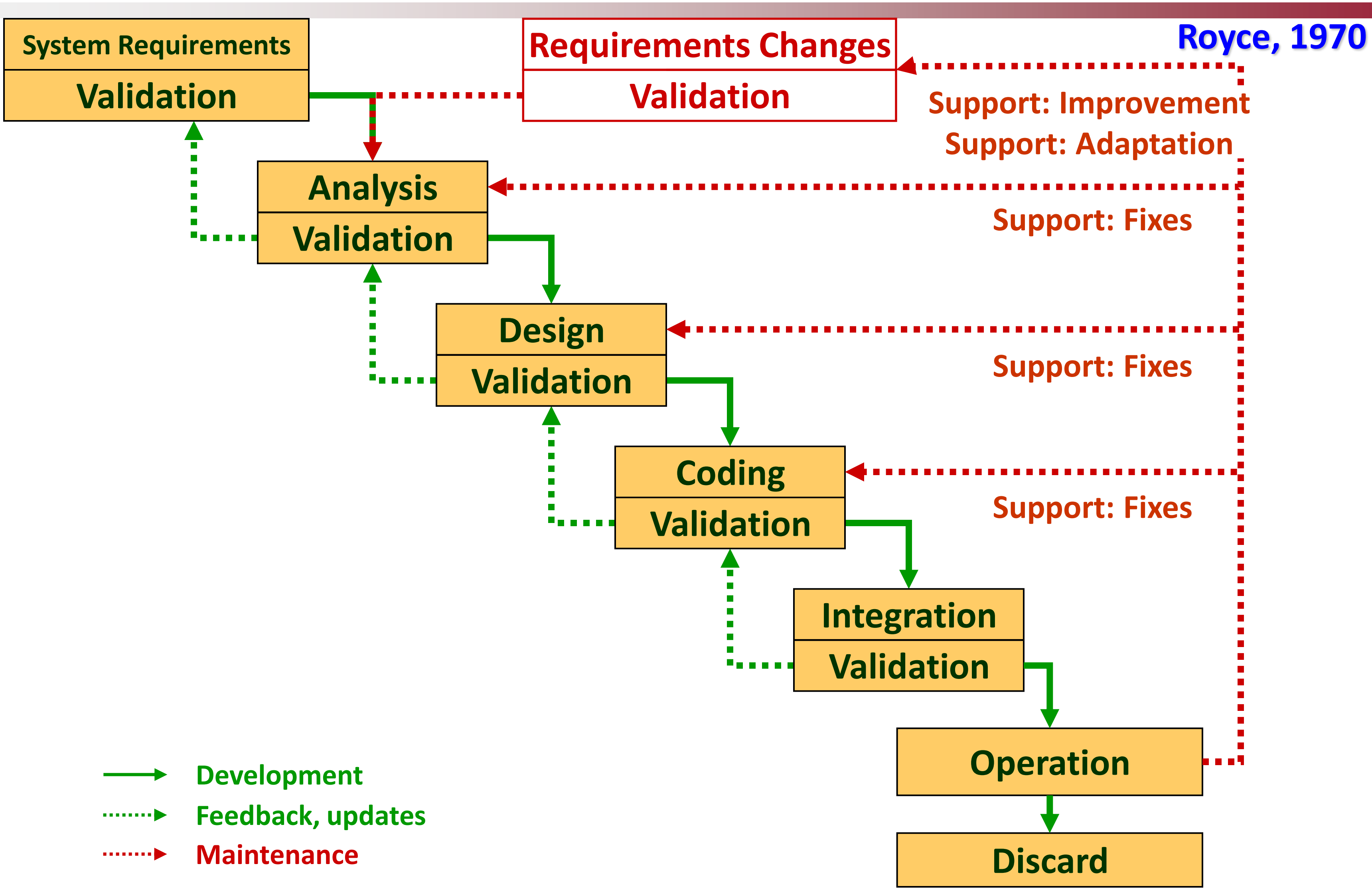


Photo by Ezgi Kimball on Unsplash

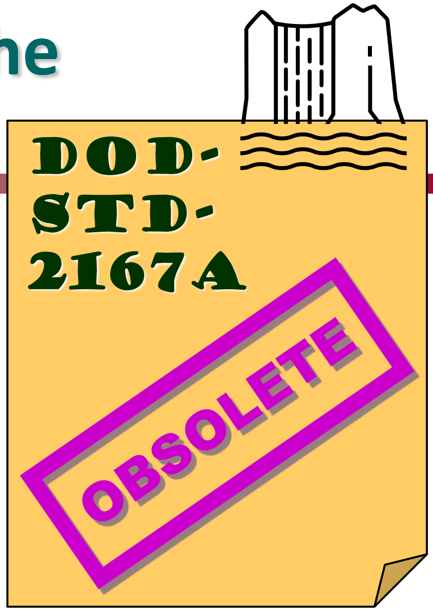
27 March 2025

Waterfall Model

Unplanned Evolution!



The (Short) life and (unsurprising) death of the Waterfall Model



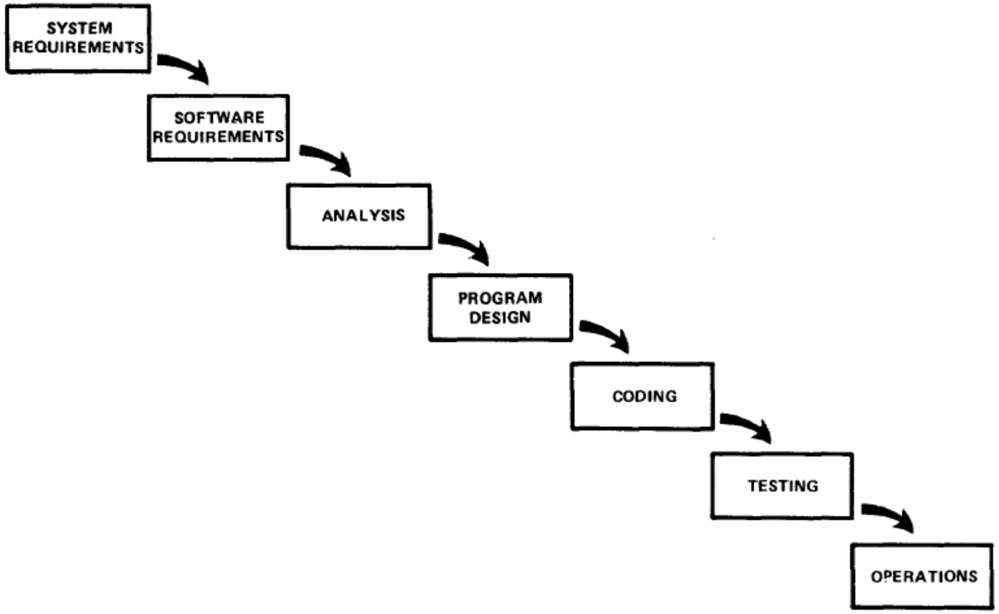


Figure 2. Implementation steps to develop a large computer program for delivery to a customer.

I believe in this concept, but the implementation described above is risky and invites failure. The problem is illustrated in Figure 4. The testing phase which occurs at the end of the development cycle is the first event for which timing, storage, input/output transfers, etc., are experienced as distinguished from analyzed. These phenomena are not precisely analyzable. They are not the solutions to the standard partial differential equations of mathematical physics for instance. Yet if these phenomena fail to satisfy the various external constraints, then invariably a major redesign is required. A simple octal patch or redo of some isolated code will not fix these kinds of difficulties. The required design changes are likely to be so disruptive that the software requirements upon which the design is based and which provides the rationale for everything are violated. Either the requirements must be modified, or a substantial change in the design is required. In effect the development process has returned to the origin and one can expect up to a lO0-percent overrun in schedule and/or costs.

Waterfall - Attributes

Every stage is made up of one activity

- Goal is to finish the whole effort in one phase
- No preparation for changes or additions
- There are response cycles between adjacent stages - Only if needed to fix problems that arise

Process is document-driven

- First phases produce only docs
- Advancing to the next phase depends on docs from previous phase
  - Leads to paralysis by analysis

Plusses

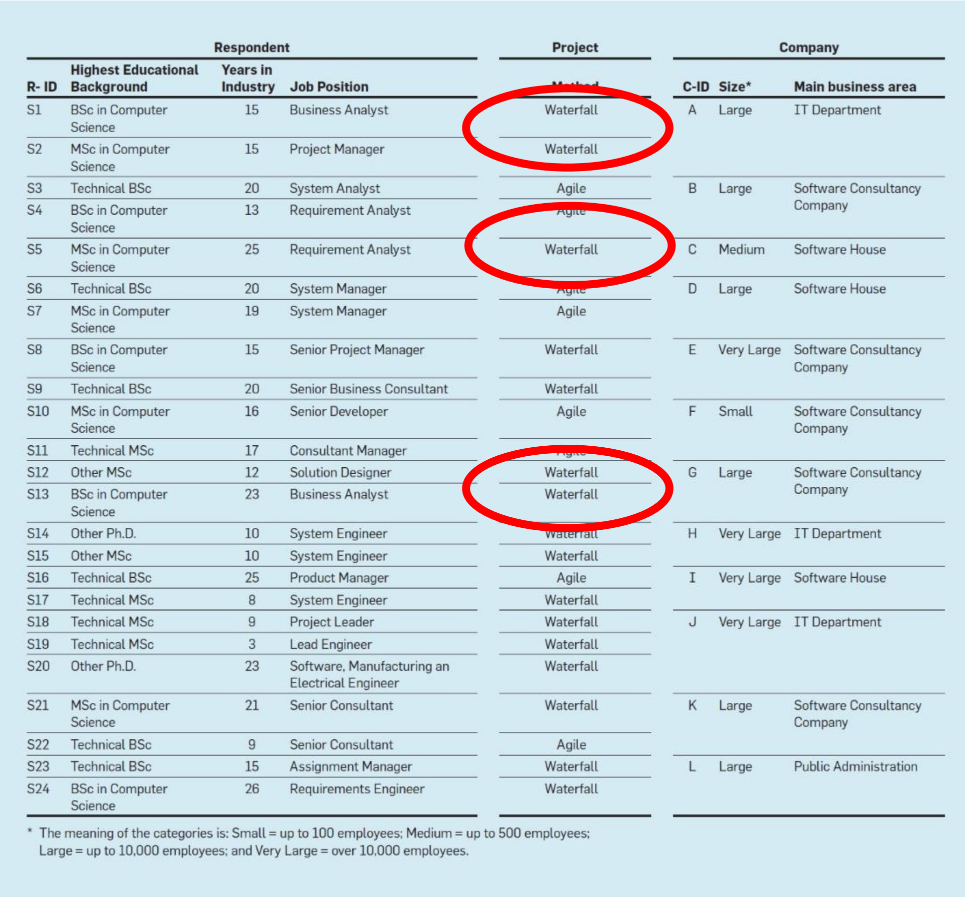
- Process is well documented
- Support is easier

Minuses

- Validation of analysis and planning can only be done on paper

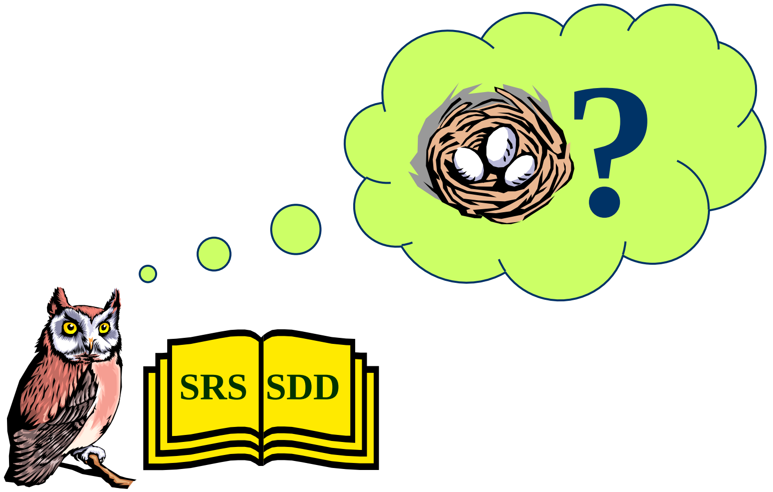
Waterfall - Attributes

- Works for small systems where change is not expected - Such as?

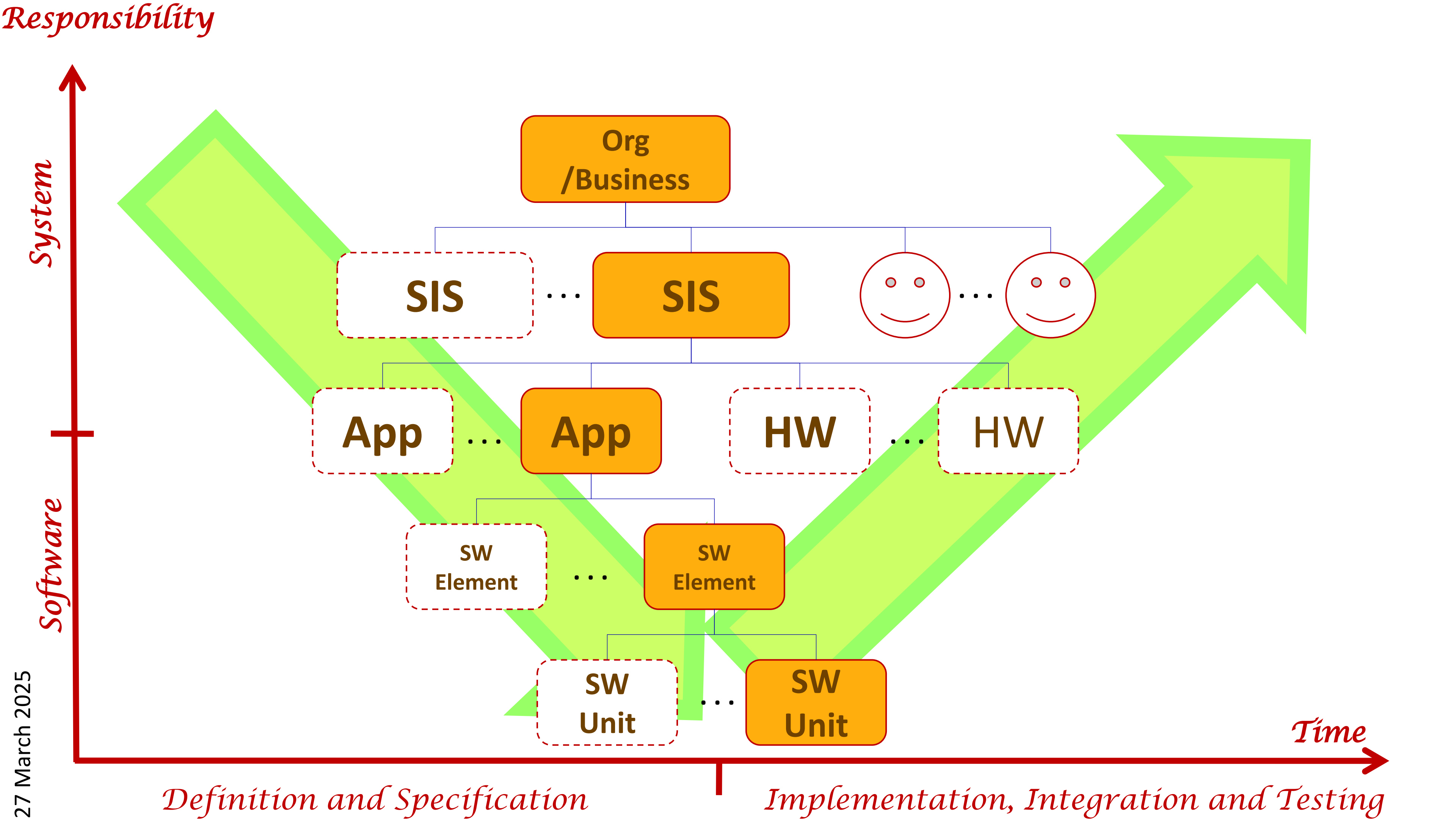


Respondent				Project	Company		
R- ID	Highest Educational Background	Years in Industry	Job Position	Method	C-ID	Size*	Main business area
S1	BSc in Computer Science	15	Business Analyst	Waterfall	A	Large	IT Department
S2	MSc in Computer Science	15	Project Manager	Waterfall			
S3	Technical BSc	20	System Analyst	Agile	B	Large	Software Consultancy Company
S4	BSc in Computer Science	13	Requirement Analyst	Agile			
S5	MSc in Computer Science	25	Requirement Analyst	Waterfall	C	Medium	Software House
S6	Technical BSc	20	System Manager	Agile	D	Large	Software House
S7	MSc in Computer Science	19	System Manager	Agile			
S8	BSc in Computer Science	15	Senior Project Manager	Waterfall	E	Very Large	Software Consultancy Company
S9	Technical BSc	20	Senior Business Consultant	Waterfall			
S10	MSc in Computer Science	16	Senior Developer	Agile	F	Small	Software Consultancy Company
S11	Technical MSc	17	Consultant Manager	Agile			
S12	Other MSc	12	Solution Designer	Waterfall	G	Large	Software Consultancy Company
S13	BSc in Computer Science	23	Business Analyst	Waterfall			
S14	Other Ph.D.	10	System Engineer	Waterfall	H	Very Large	IT Department
S15	Other MSc	10	System Engineer	Waterfall			
S16	Technical BSc	25	Product Manager	Agile	I	Very Large	Software House
S17	Technical MSc	8	System Engineer	Waterfall			
S18	Technical MSc	9	Project Leader	Waterfall	J	Very Large	IT Department
S19	Technical MSc	3	Lead Engineer	Waterfall			
S20	Other Ph.D.	23	Software, Manufacturing an Electrical Engineer	Waterfall			
S21	MSc in Computer Science	21	Senior Consultant	Waterfall	K	Large	Software Consultancy Company
S22	Technical BSc	9	Senior Consultant	Agile			
S23	Technical BSc	15	Assignment Manager	Waterfall	L	Large	Public Administration
S24	BSc in Computer Science	26	Requirements Engineer	Waterfall			

* The meaning of the categories is: Small = up to 100 employees; Medium = up to 500 employees; Large = up to 10,000 employees; and Very Large = over 10,000 employees.

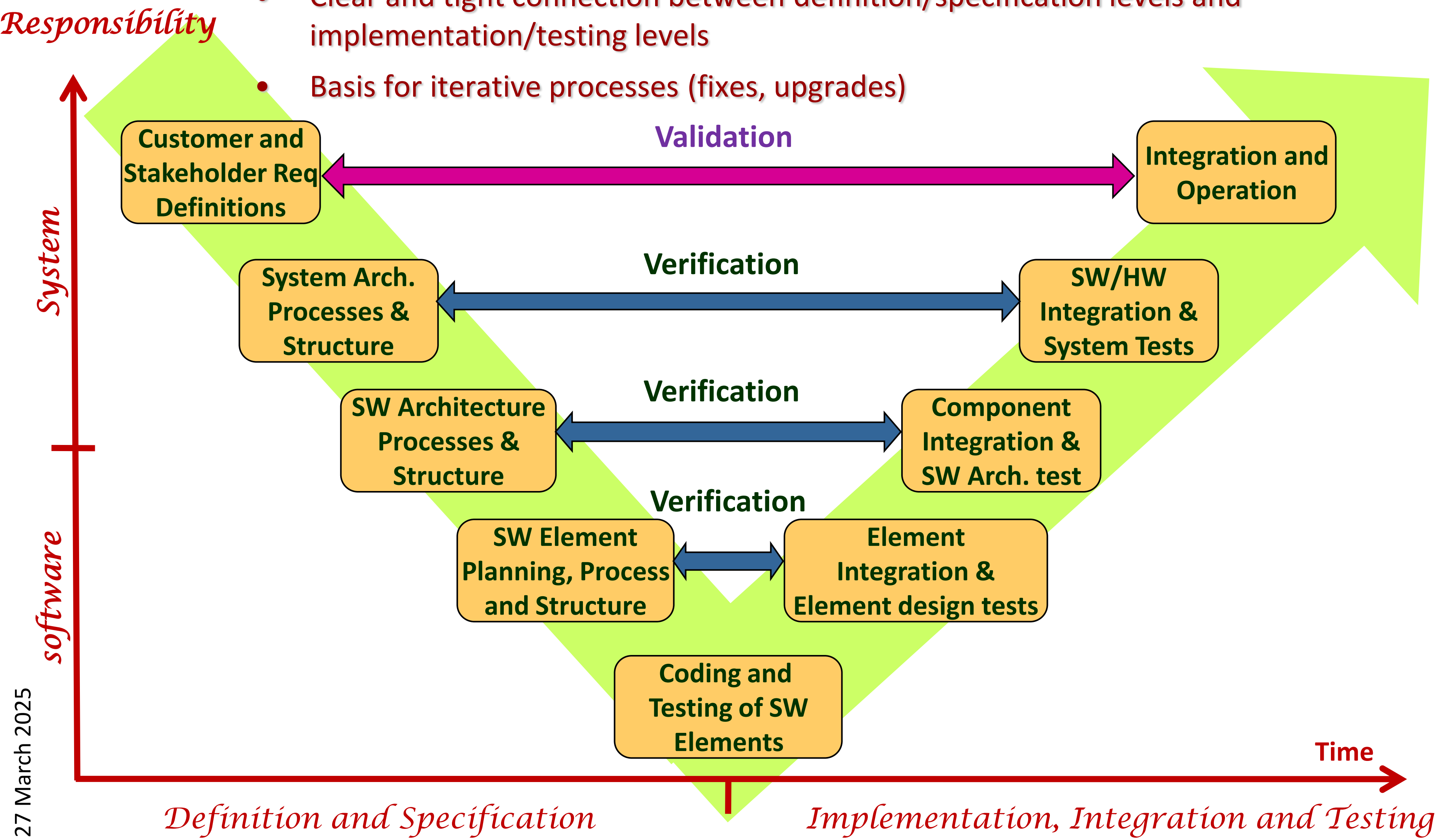


A framework for software intensive development



V Model – Generic framework for SIS development

- Clear and tight connection between definition/specification levels and implementation/testing levels
- Basis for iterative processes (fixes, upgrades)


Responsibility
System
software
Customer and Stakeholder Req Definitions
Validation
Integration and Operation
System Arch. Processes & Structure
Verification
SW/HW Integration & System Tests
SW Architecture Processes & Structure
Verification
Component Integration & SW Arch. test
Verification
SW Element Planning, Process and Structure
Element Integration & Element design tests
Coding and Testing of SW Elements
Time
Definition and Specification
Implementation, Integration and Testing

V Model – A simplified view

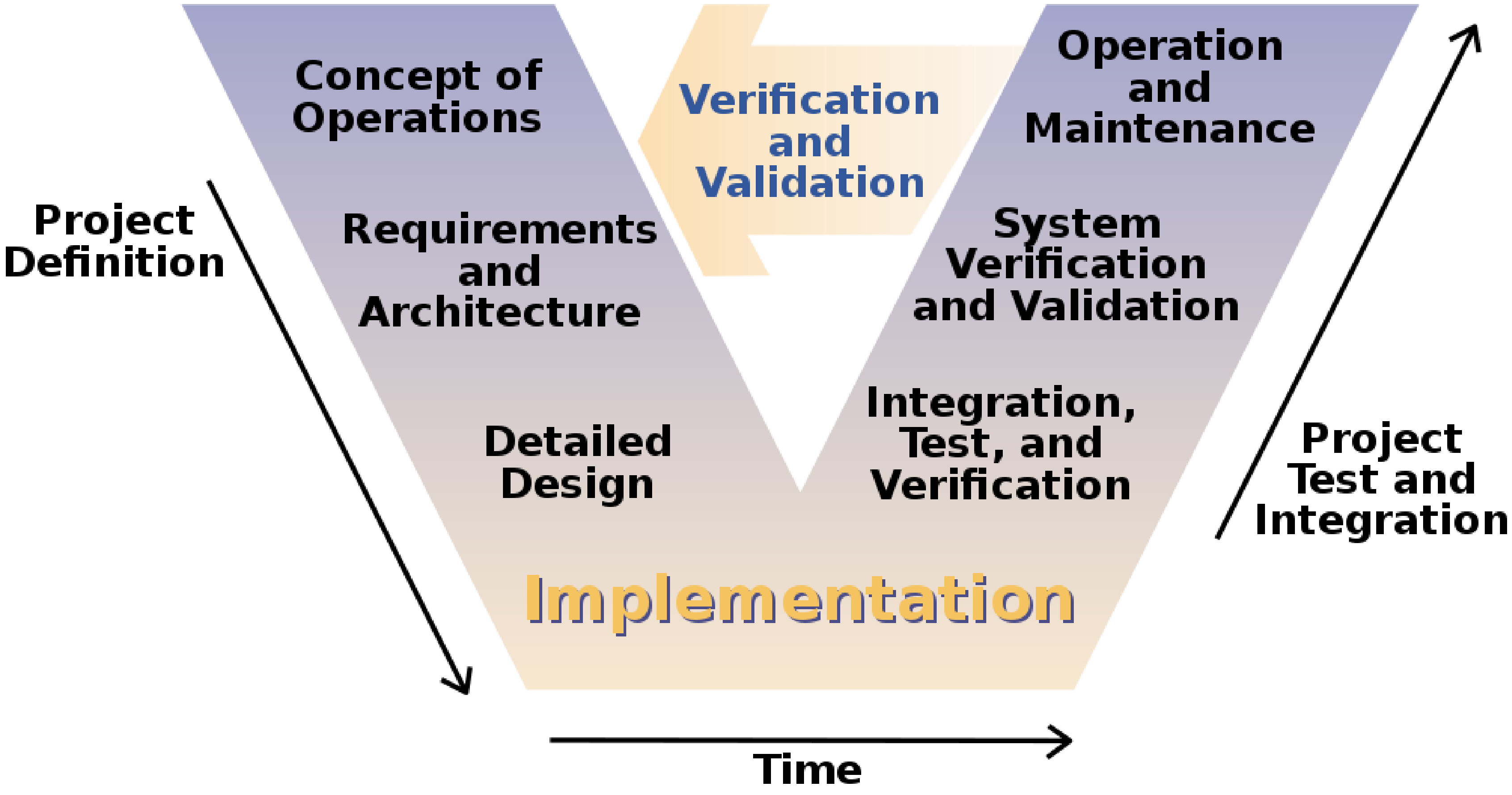

Concept of Operations
Verification and Validation
Operation and Maintenance
Project Definition
Requirements and Architecture
System Verification and Validation
Detailed Design
Integration, Test, and Verification
Project Test and Integration
Implementation
Time

Image source: By Leon Osborne, Jeffrey Brummond, Robert Hart, Mohsen (Moe) Zarean Ph.D., P.E, Steven Conger ; Redrawn by User:Slashme. – Image extracted from Clarus Concept of Operations. Publication No. FHWA-JPO-05-072, Federal Highway Administration (FHWA), 2005, Public Domain, https://commons.wikimedia.org/w/index.php?curid=10275054

Development Processes that support evolution

Incremental/Iterative

- Build the system in a fixed number of rounds
- Defined during an initial, partial round

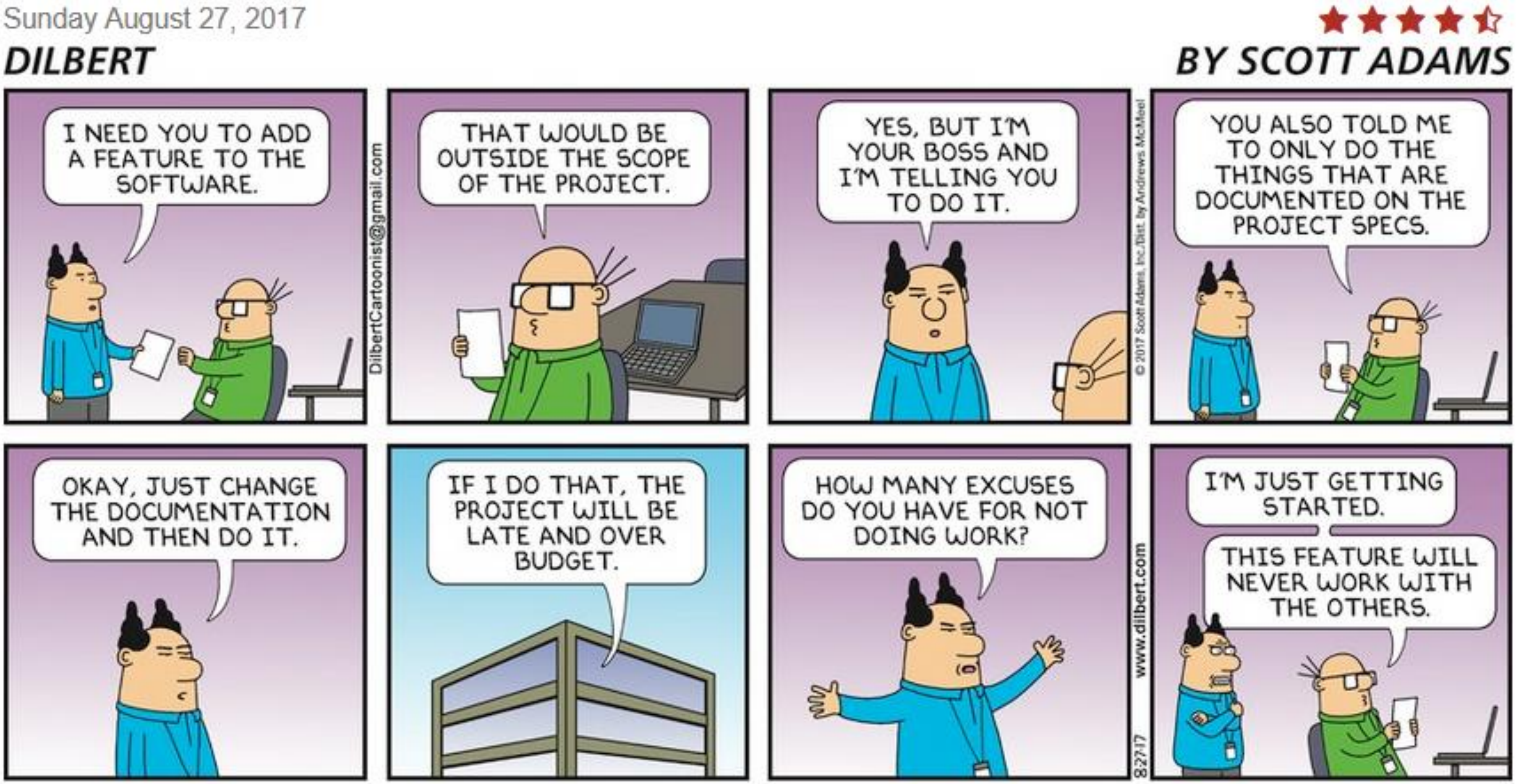
Agile development methods

- Short development rounds emphasizing implementation
- Update existing products as needed

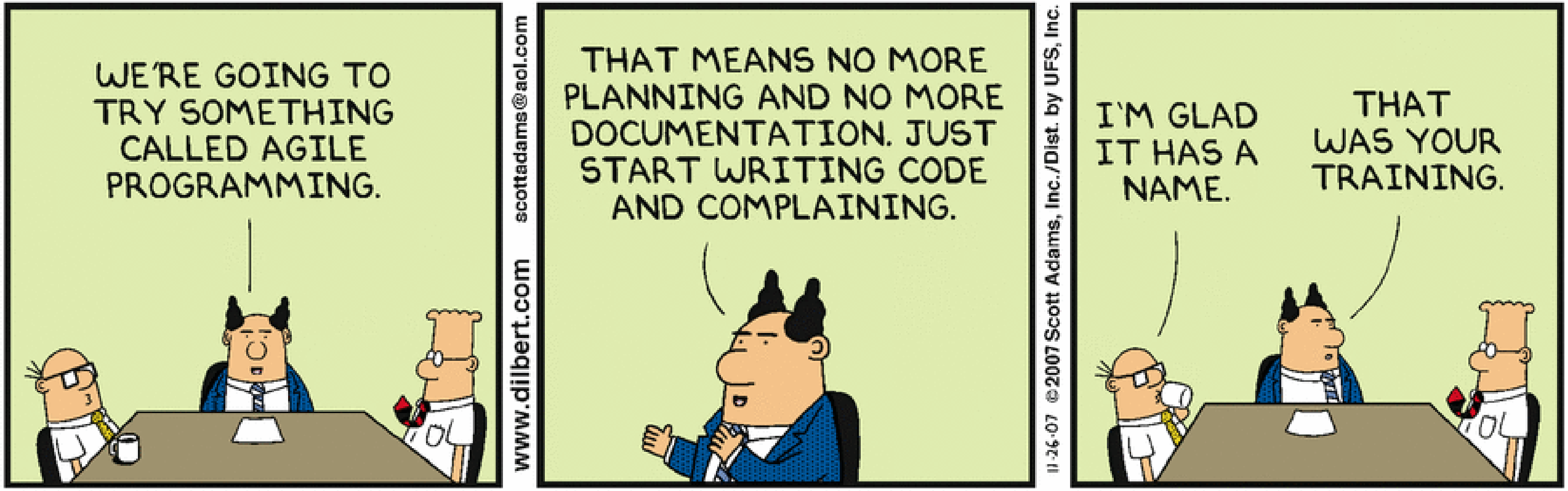
DevOps

- Integrate quick development and operation in the field
- Short bursts of activity, heavy automation

https://dilbert.com/strip/2017-08-27


Sunday August 27, 2017
DILBERT
BY SCOTT ADAMS
I NEED YOU TO ADD A FEATURE TO THE SOFTWARE.
DilbertCartoonist@gmail.com
THAT WOULD BE OUTSIDE THE SCOPE OF THE PROJECT.
YES, BUT I'M YOUR BOSS AND I'M TELLING YOU TO DO IT.
© 2017 Scott Adams, Inc./Dist. by Andrews McMeel
YOU ALSO TOLD ME TO ONLY DO THE THINGS THAT ARE DOCUMENTED ON THE PROJECT SPECS.
OKAY, JUST CHANGE THE DOCUMENTATION AND THEN DO IT.
IF I DO THAT, THE PROJECT WILL BE LATE AND OVER BUDGET.
HOW MANY EXCUSES DO YOU HAVE FOR NOT DOING WORK?
www.dilbert.com
8-27-17
I'M JUST GETTING STARTED.
THIS FEATURE WILL NEVER WORK WITH THE OTHERS.

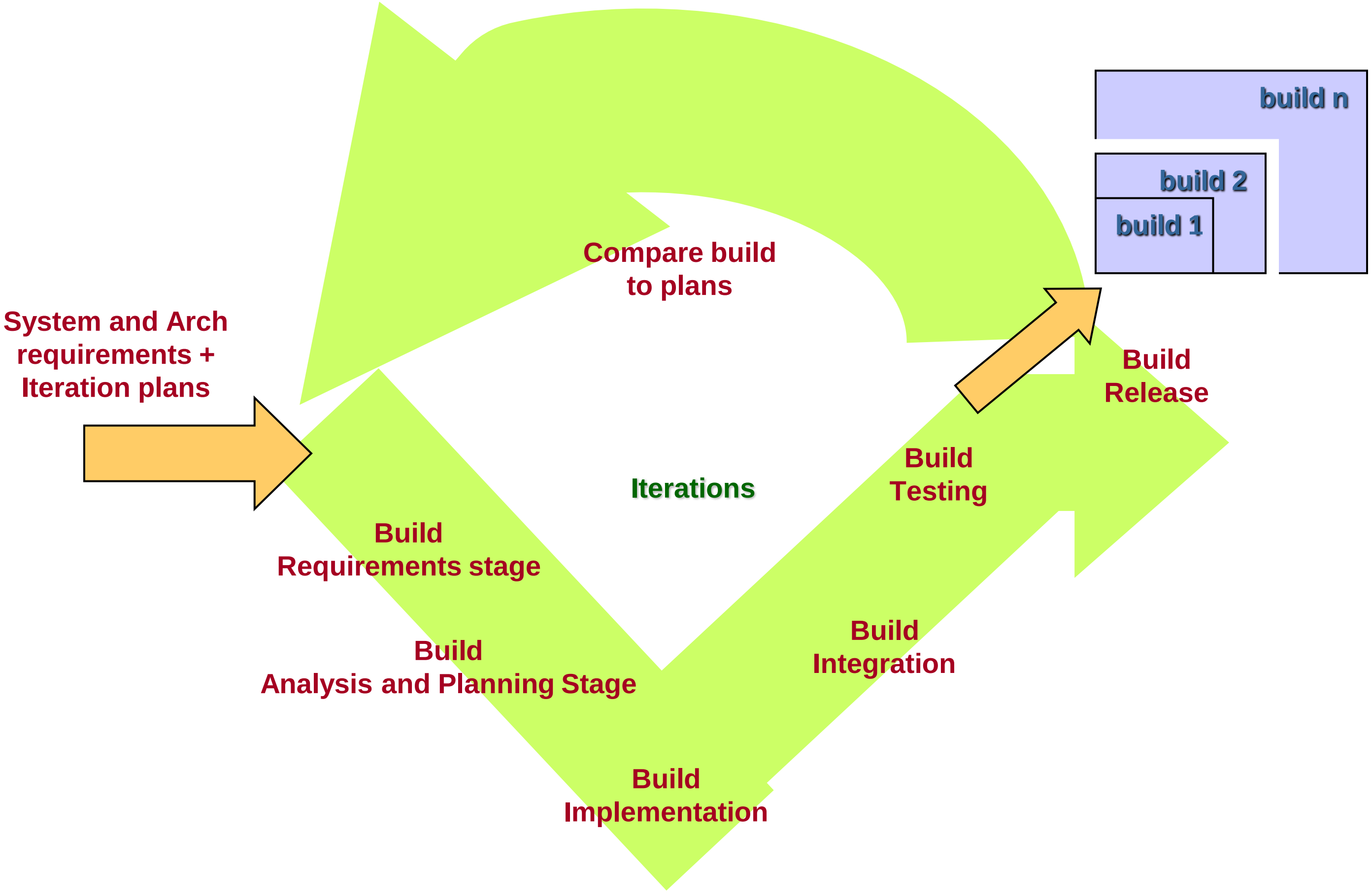
https://dilbert.com/strip/2007-11-26


WE'RE GOING TO TRY SOMETHING CALLED AGILE PROGRAMMING.
scottadams@aol.com
www.dilbert.com
THAT MEANS NO MORE PLANNING AND NO MORE DOCUMENTATION. JUST START WRITING CODE AND COMPLAINING.
11-26-07 ©2007 Scott Adams, Inc./Dist. by UFS, Inc.
I'M GLAD IT HAS A NAME.
THAT WAS YOUR TRAINING.

Incremental/Iterative Development Concept

- Divide the system into “pieces”
  - Plan the iterations and closely manage them
  - Intermediate builds – operational – that are tested and work!
  - Risks track iterations
    - Each iteration’s product removes some primary risk
- No clear definition of terms
  - One option:
    - The tool is incremental
    - The development process is iterative.

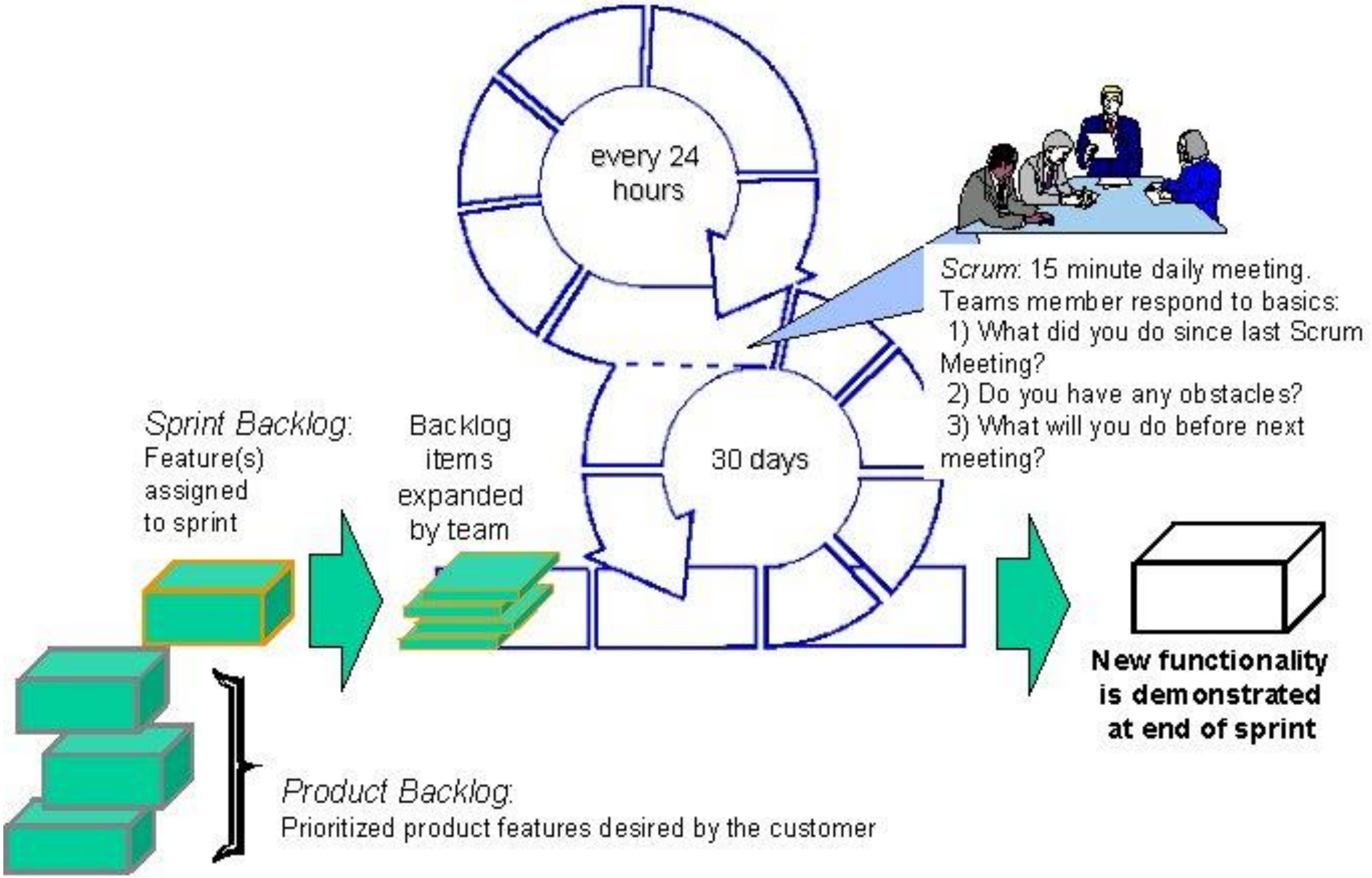
Incremental/Iterative Development Process



Agile Development Concept

- Previous models were “Plan Driven”
- Claim: Software can’t be developed in fixed plans because
  - It’s hard to see the whole picture
  - Design and Specification changes all the time
  - Software is flexible, so it can handle change
  - The lack of a “production” phase in software development means you can do quick iterations and delivery
- Therefore…
  - We need a quick, flexible development process that can support development under the above conditions

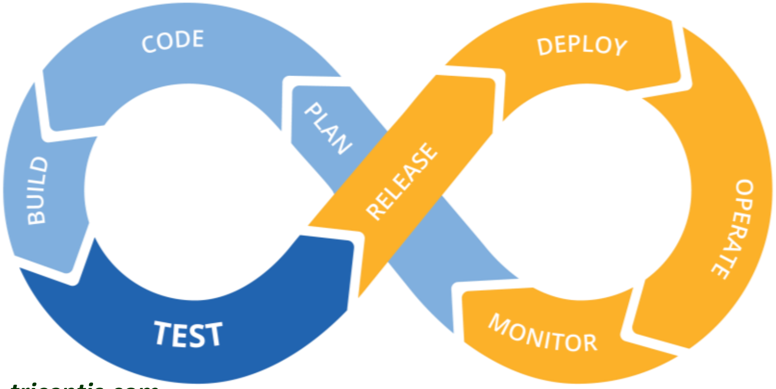
Agile Development: Scrum methodology


every 24 hours
30 days
Sprint Backlog:
Feature(s) assigned to sprint
Backlog items expanded by team
New functionality is demonstrated at end of sprint
Product Backlog:
Prioritized product features desired by the customer

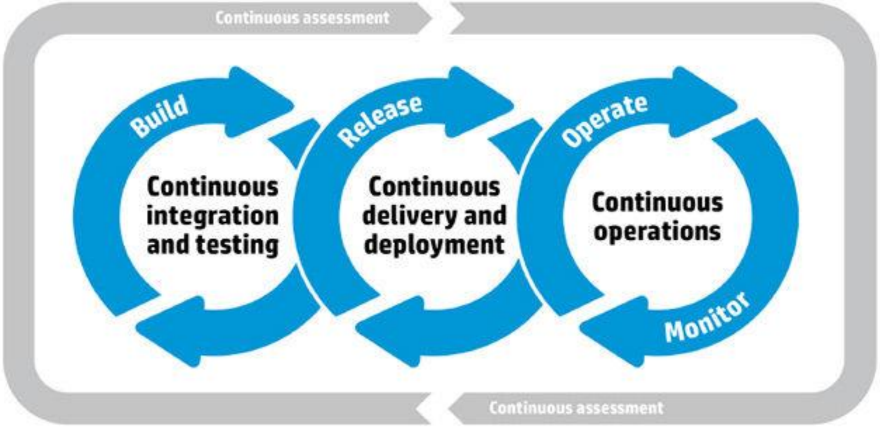
Scrum: 15 minute daily meeting.
Teams member respond to basics:
1) What did you do since last Scrum Meeting?
2) Do you have any obstacles?
3) What will you do before next meeting?

DevOps: Development Operations (Infinite Development)

- Concept: Regular releases based on information and actual usage

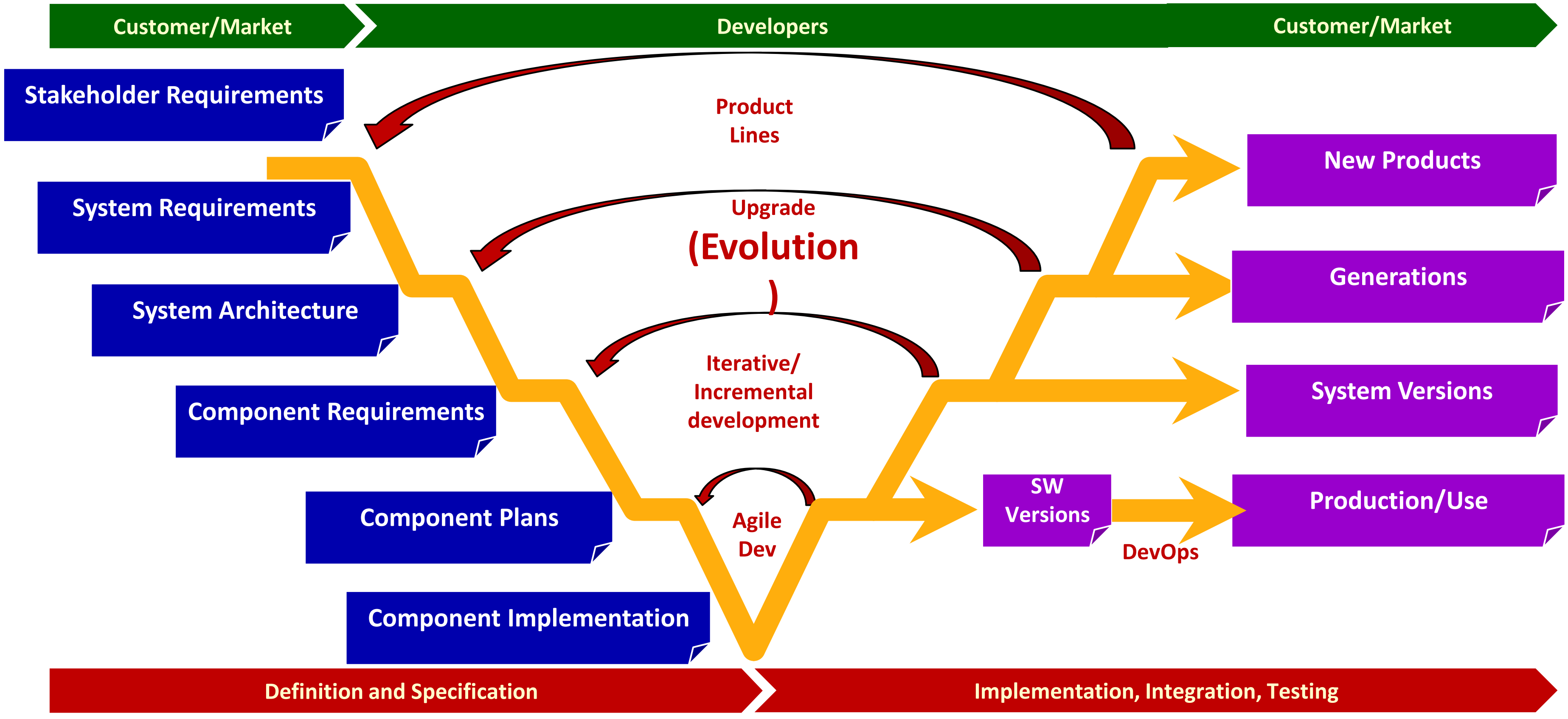


Picture Source: http://cdn.tricentis.com



Picture Source: https://www.linkedin.com/pulse/what-really-devops-does-benone-bitencourt

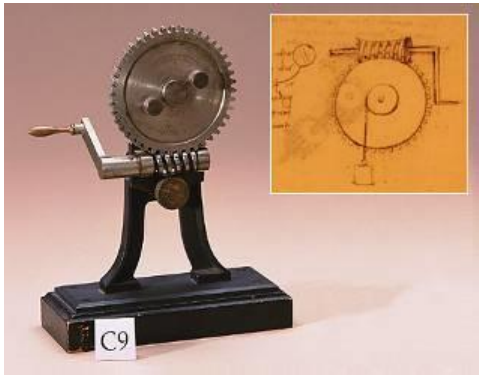
Complete Lifecycle of Software Intensive System



Model Based Development

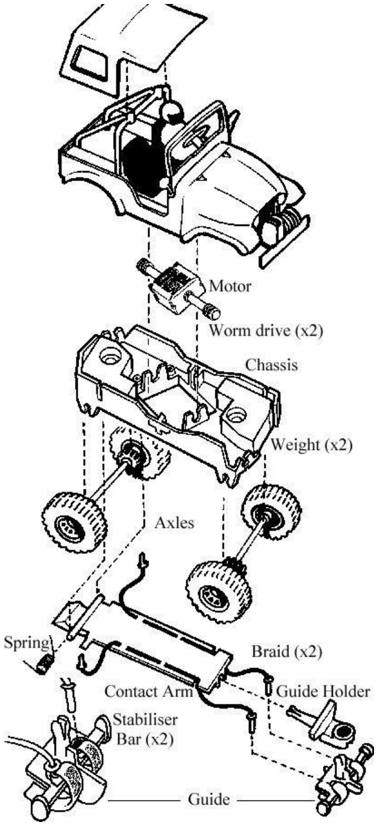
- Modeling
  - A means to capture ideas, relationships, decisions and requirements in a well-defined notation that can be applied to many different domains

    [Pilone, D., *UML 2.0 in a Nutshell*, O’REILLY®, 2005
- Models give a simplified and abstract idea of complex entities
  - Models focus on primary elements without getting into details
  - Models require some translation to the real world
  - Models have different degrees of freedom in interpretation



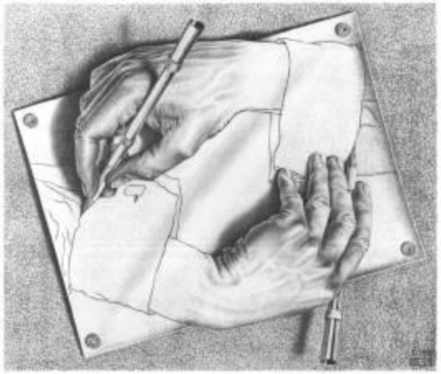
Using Models: Two directions

- Forward Modeling: Model before implement
  - Sketches of new ideas
  - Brainstorming before solutions
  - Offer solution alternatives
  - Drive development plans
- Reverse Modeling: Model after implementation
  - Document a system as built
  - Explain a system to others
  - Support production/maintenance/upgrading of systems
  - Use as an input for forward modeling for future systems

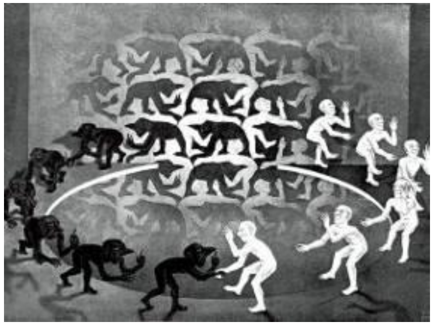


Static and Dynamic Models

- Static/Structural Models: Model that describes entities and connections between them
  - Organizational chart
  - Mechanical sketch
  - Molecular structure
  - Table/Relation in a DB



- Dynamic/Behavioral/Operational Model: Models that describe the process, flow, or state transitions
  - Flow chart/Algorithm
  - Graph of function over time
  - Automaton
  - Animation/Simulation

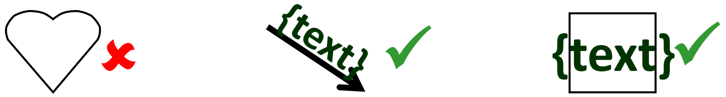


- Models can be graphical, textual, or a combination

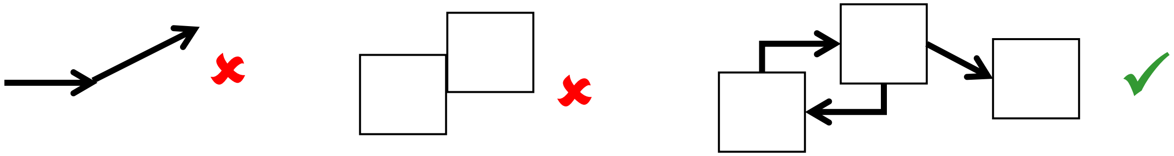
27 March 2025

Modeling in a graphical language

- Set of legal symbols (alphabet)



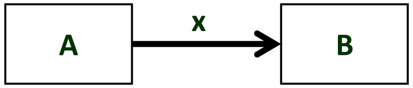
- Legal combinations (grammar)



- Meaning (semantics)

✗ Process A sends data x to Process B

✓ Computer A is connected to Computer B via interface x



?

- Expressiveness

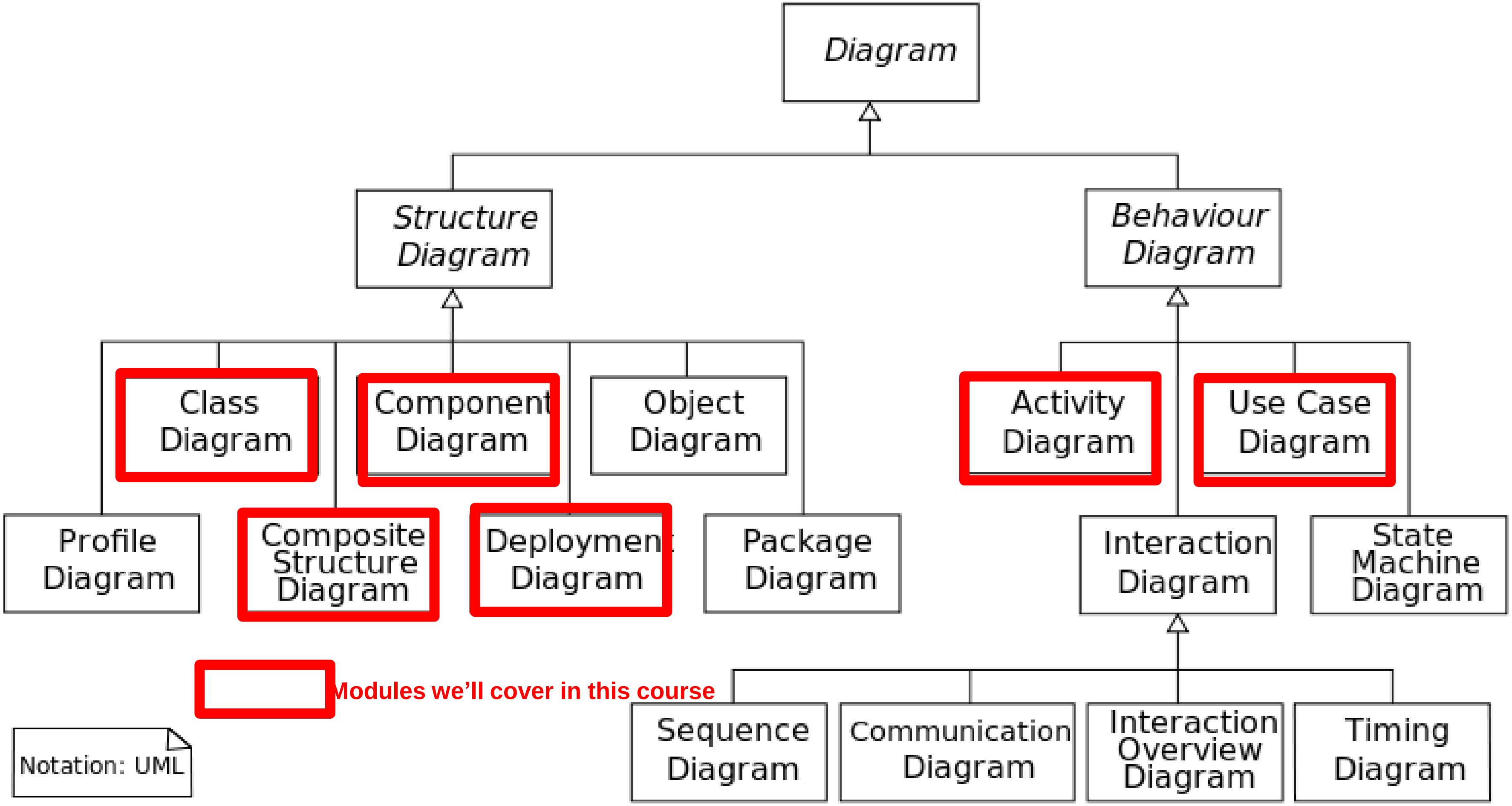
✗ Computer A is connected to at least 1 and no more than 10 Computer B's at once

✓ Computer A is connected to Computer B via interface x

UML = Unified Modeling Language

- Graphical language for Object Oriented Analysis and Design
  - Current version UML 2.5.1 (Dec 2017)
- Includes a collection of modeling tools for different aspects of software
- Evolved by merging sets of existing methods
  - Grady Booch, 1991-1996
  - James Rumbaugh (OMT), 1992-1996
  - Ivar Jacobson (Objectory), 1992-1997
- Computer Aided Software Engineering (CASE) tools from Rational (IBM)
- Object Management Group (OMG) adopted it in 1997 as an ad hoc standard
  - Represents advice for about 800 companies and organizations.

UML Model Families



Methodology: Connects Language and Process
MBSE = Model Based System/Software Engineering

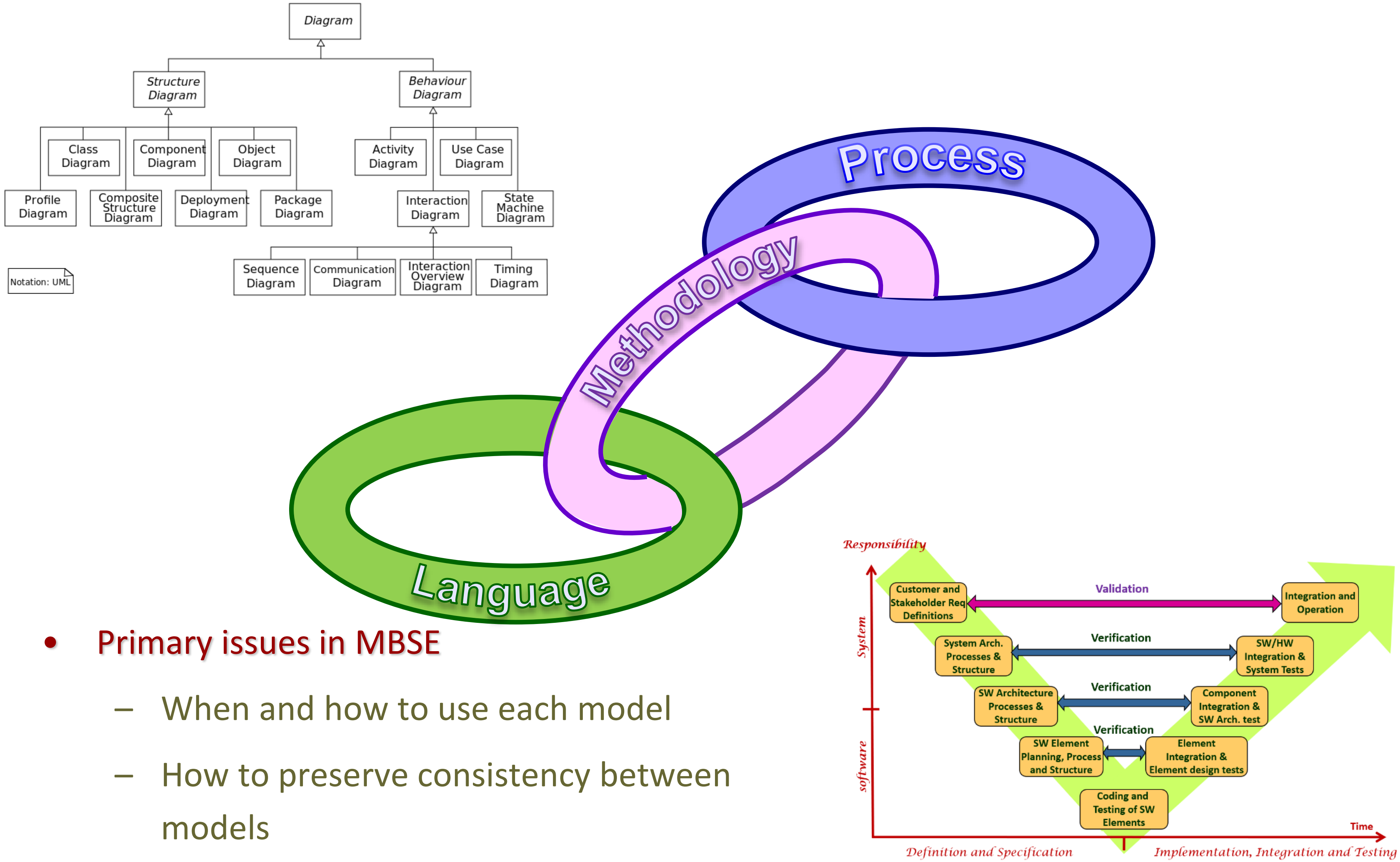
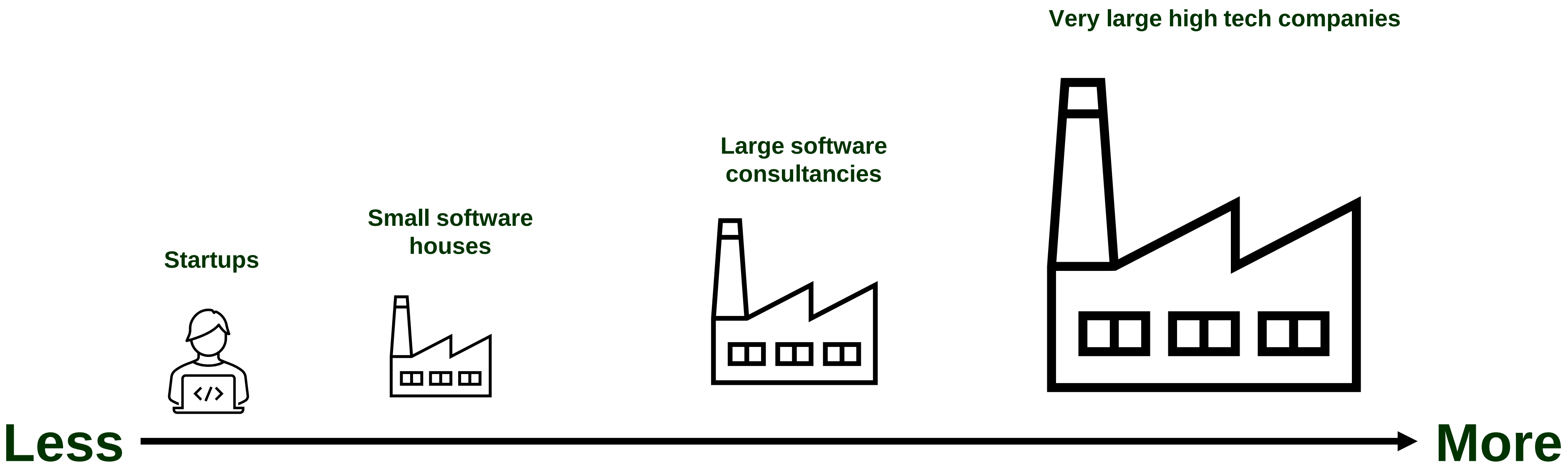


Diagram
Structure Diagram
Behaviour Diagram
Class Diagram
Component Diagram
Object Diagram
Profile Diagram
Composite Structure Diagram
Deployment Diagram
Package Diagram
Activity Diagram
Use Case Diagram
Interaction Diagram
State Machine Diagram
Sequence Diagram
Communication Diagram
Interaction Overview Diagram
Timing Diagram
Notation: UML
Process
Methodology
Language
Responsibility
System
software
Customer and Stakeholder Req Definitions
Validation
Integration and Operation
System Arch. Processes & Structure
Verification
SW/HW Integration & System Tests
SW Architecture Processes & Structure
Verification
Component Integration & SW Arch. test
Verification
SW Element Planning, Process and Structure
Element Integration & Element design tests
Coding and Testing of SW Elements
Time
Definition and Specification
Implementation, Integration and Testing

- Primary issues in MBSE
  - When and how to use each model
  - How to preserve consistency between models

Who uses UML and modelling?

“Industry”



Very large high tech companies

Large software consultancies

Small software houses

Startups

Less

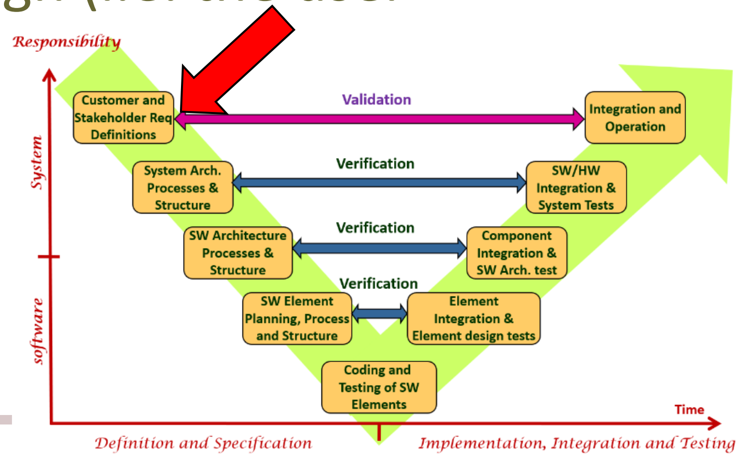
More

So Far

- Software from a Systems Perspective
- Development Process of a Software Intensive System
- Requirements and Stakeholders
- Elicitation and Management

Eliciting requirements from customers and stakeholders

- Goal of the stage
  - Understand the problem/need + the solution concept
  - Collect, concretize, categorize requirements
    - Reflect the parts of the problem/need and the solution concept
    - Are the basis for managing the development
- Input
  - Technical design, operational design (i.e. the user stories and stakeholder stories)
- Result
  - Categorized requirements table


Responsibility
System
software
Customer and Stakeholder Req Definitions
Validation
Integration and Operation
System Arch. Processes & Structure
Verification
SW/HW Integration & System Tests
SW Architecture Processes & Structure
Verification
Component Integration & SW Arch. test
SW Element Planning, Process and Structure
Verification
Element Integration & Element design tests
Coding and Testing of SW Elements
Time
Definition and Specification
Implementation, Integration and Testing

I am thirsty!

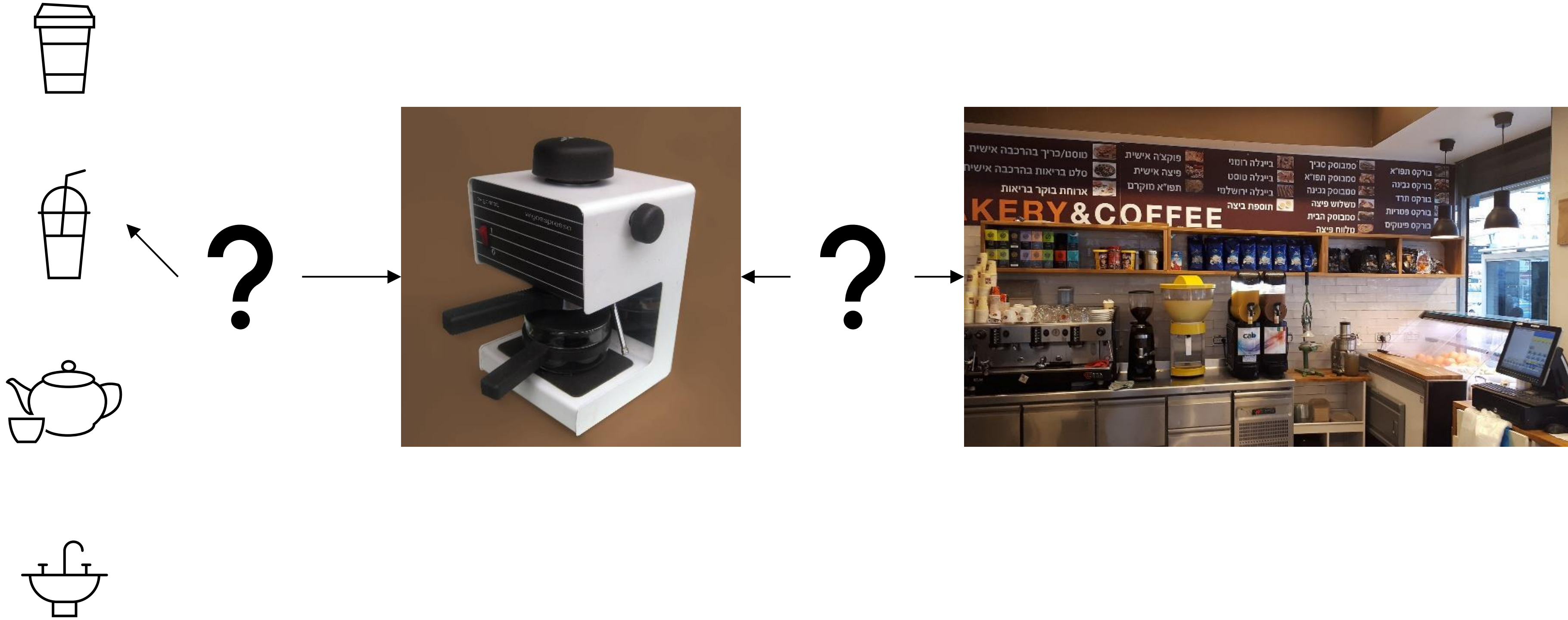


Image sources: Acosta.eu, CC BY-SA 3.0,
via Wikimedia Commons, Rakoon, CC0, via Wikimedia Commons

Discussion Question

What are “Requirements”

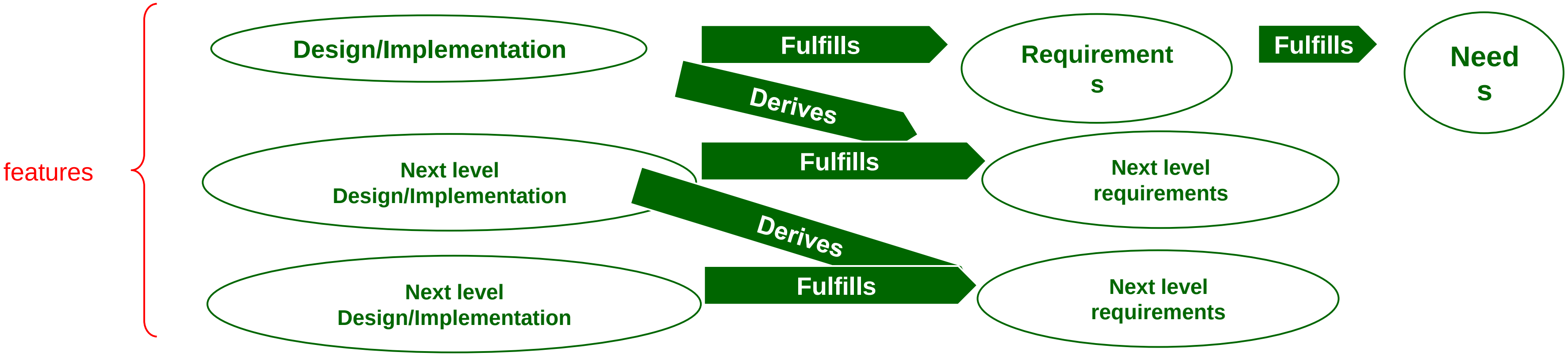
- Give examples of requirements for:
  - A car
  - A software engineering student
  - Birthday present

Who decides/chooses the requirements?

What’s the difference between requirement and features?

Definitions

- A requirement
  - A property or a capability of the product that is required to address a problem/need
    - Needed to solve a problem or achieve some goal (for users)
    - To meet a contract, standard, or applicable document (for stakeholders)
- Design/Implementation
  - A technical solution through which you can satisfy one or more requirements
  - Design of each level becomes the foundation for requirements at the next level



Stakeholders – Those who require the requirements

- Definition: Stakeholder: An entity that is affected by the system’s development or are in someway responsible for its development
- Common stakeholders in SIS:

Users

- Use the system for their own needs

Customers/Clients

- Ordered the system

Market Analysts

- Represent customers and users
- Analyze their needs and wants

Regulators and Professional Bodies

- Define standards or rules that affect the system

Engineers

- Represent technical interests
- Affect system design, acceptable technologies, reuse, off-the-shelf components

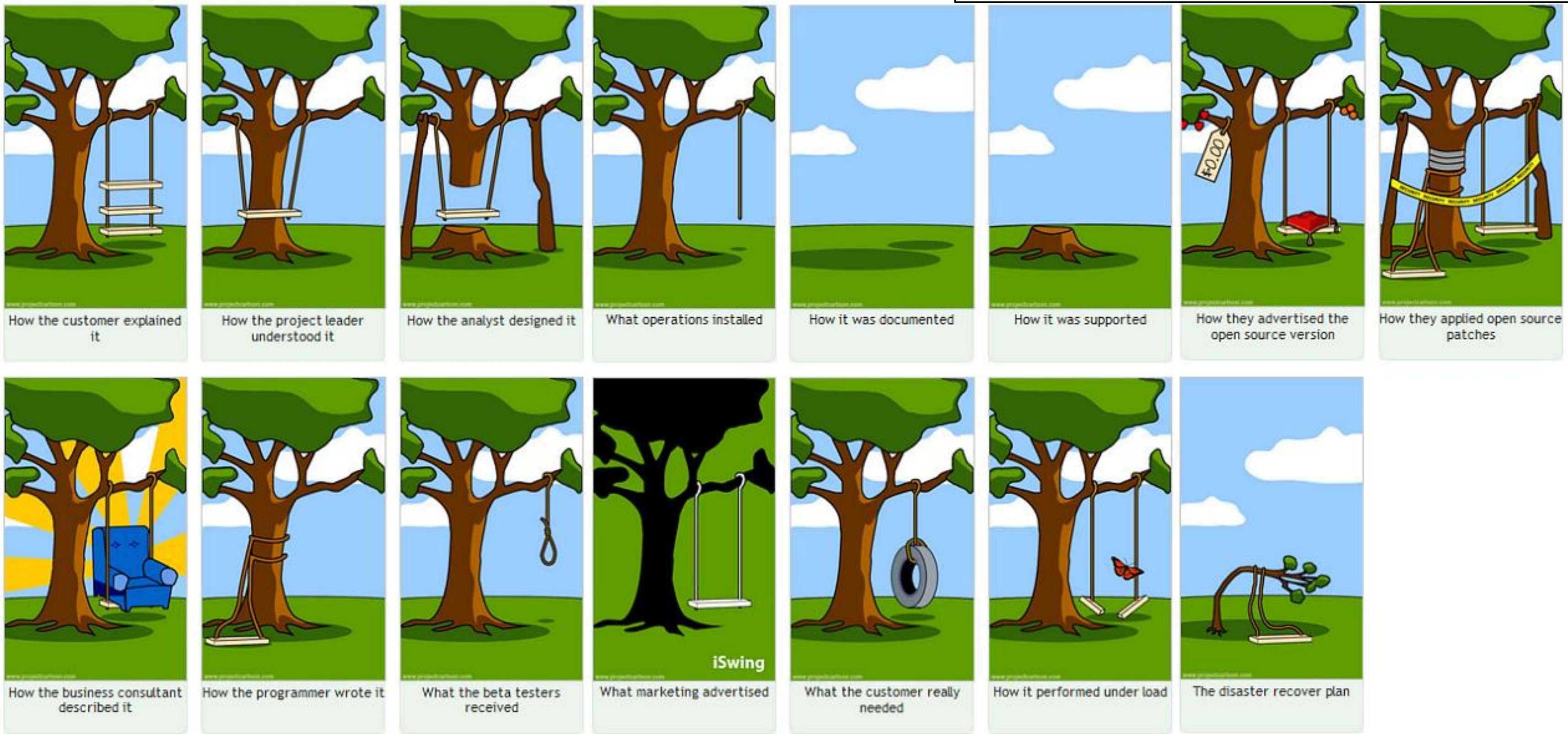
Managers

- Represent organizational interests
- Affect requirements such as human resources, other projects, business goals, budget, timing

Everyone sees things differently

Product development from an IT failures perspective

Image source: https://i.pinimg.com/originals/81/69/66/816966c6145a936ff046c8e9c9c63986.png



How the customer explained it

How the project leader understood it

How the analyst designed it

What operations installed

How it was documented

How it was supported

How they advertised the open source version

How they applied open source patches

How the business consultant described it

How the programmer wrote it

What the beta testers received

What marketing advertised

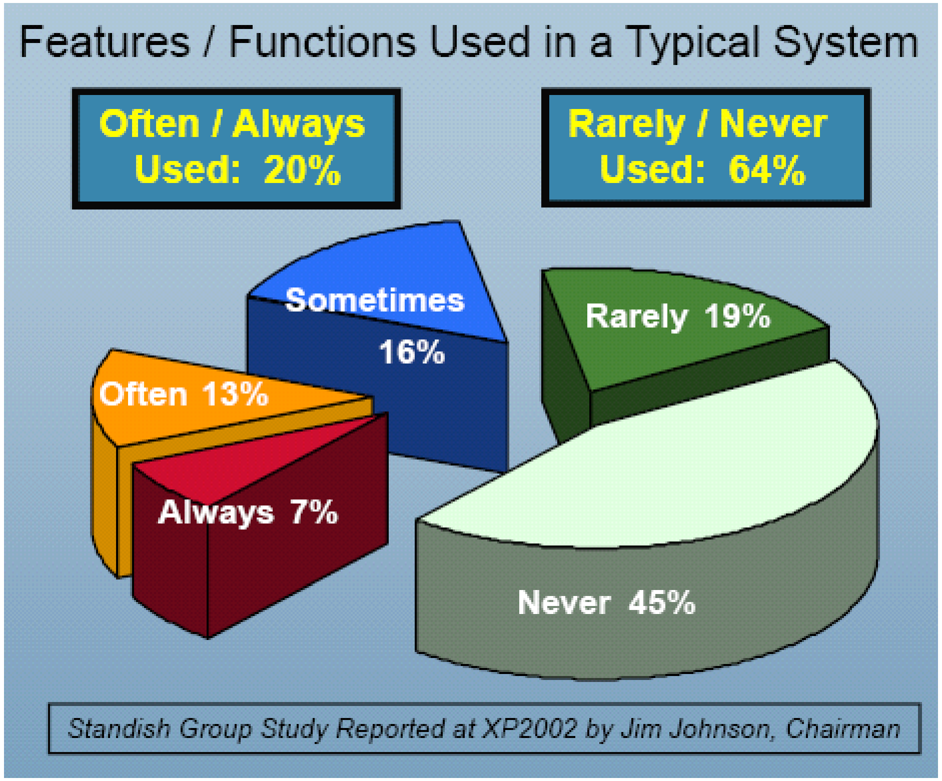
What the customer really needed

How it performed under load

The disaster recover plan

27 March 2025

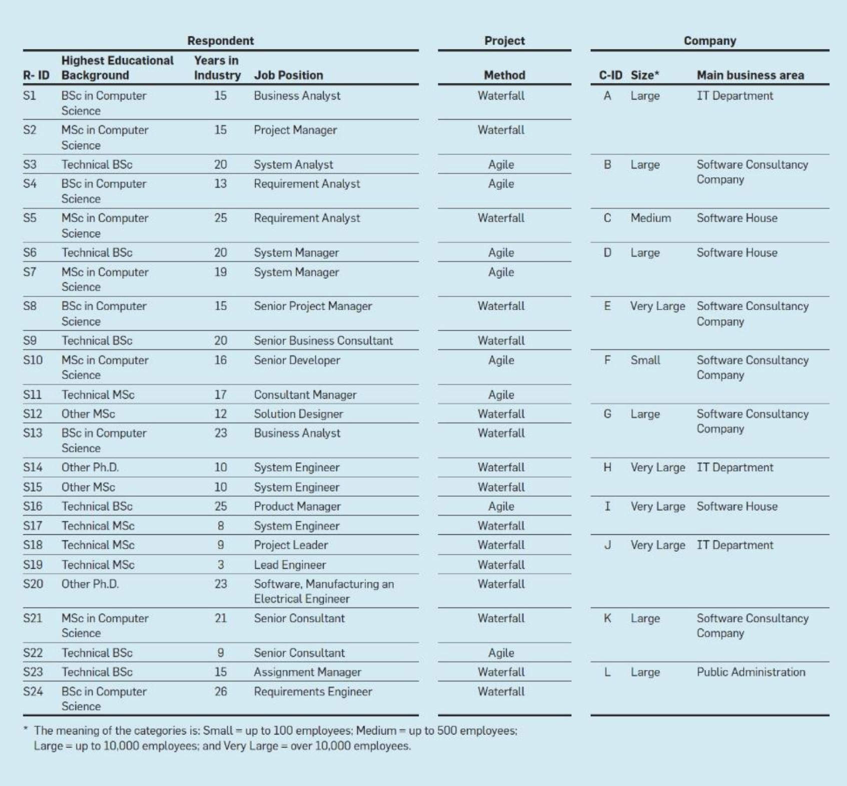
What requirements are needed?



Standish Group Study Reported at XP2002 by Jim Johnson, Chairman

- Give an example of a feature that almost never used.

Aside: Requirements Engineering



Respondent				Project	Company		
R- ID	Highest Educational Background	Years in Industry	Job Position	Method	C-ID	Size*	Main business area
S1	BSc in Computer Science	15	Business Analyst	Waterfall	A	Large	IT Department
S2	MSc in Computer Science	15	Project Manager	Waterfall			
S3	Technical BSc	20	System Analyst	Agile	B	Large	Software Consultancy Company
S4	BSc in Computer Science	13	Requirement Analyst	Agile			
S5	MSc in Computer Science	25	Requirement Analyst	Waterfall	C	Medium	Software House
S6	Technical BSc	20	System Manager	Agile	D	Large	Software House
S7	MSc in Computer Science	19	System Manager	Agile			
S8	BSc in Computer Science	15	Senior Project Manager	Waterfall	E	Very Large	Software Consultancy Company
S9	Technical BSc	20	Senior Business Consultant	Waterfall			
S10	MSc in Computer Science	16	Senior Developer	Agile	F	Small	Software Consultancy Company
S11	Technical MSc	17	Consultant Manager	Agile			
S12	Other MSc	12	Solution Designer	Waterfall	G	Large	Software Consultancy Company
S13	BSc in Computer Science	23	Business Analyst	Waterfall			
S14	Other Ph.D.	10	System Engineer	Waterfall	H	Very Large	IT Department
S15	Other MSc	10	System Engineer	Waterfall			
S16	Technical BSc	25	Product Manager	Agile	I	Very Large	Software House
S17	Technical MSc	8	System Engineer	Waterfall			
S18	Technical MSc	9	Project Leader	Waterfall	J	Very Large	IT Department
S19	Technical MSc	3	Lead Engineer	Waterfall			
S20	Other Ph.D.	23	Software, Manufacturing an Electrical Engineer	Waterfall			
S21	MSc in Computer Science	21	Senior Consultant	Waterfall	K	Large	Software Consultancy Company
S22	Technical BSc	9	Senior Consultant	Agile			
S23	Technical BSc	15	Assignment Manager	Waterfall	L	Large	Public Administration
S24	BSc in Computer Science	26	Requirements Engineer	Waterfall			

* The meaning of the categories is: Small = up to 100 employees; Medium = up to 500 employees; Large = up to 10,000 employees; and Very Large = over 10,000 employees.

SYSTEMS, ENGINEERING, PROCESS

REQUIREMENTS ENGINEERING

- The latest techniques from practising requirements engineers
- A flexible process for a variety of system development contexts
- Explains the important new concept of rich traceability

M. Elizabeth C. Hull, Ken Jackson and A. Jeremy J. Dick

Springer

PRACTITIONER SERIES

https://link.springer.com/book/10.1007/978-1-4471-3730-6

Xavier Franch, Cristina Palomares, and Tony Gorschek. 2021. On the requirements engineer role. *Commun. ACM* 64, 6 (June 2021), 69–75. DOI:https://doi.org/10.1145/3418292

So Far

- Software from a Systems Perspective
- Development Process of a Software Intensive System
- Requirements and Stakeholders
- Elicitation and Management

Specification: Customer and Stakeholder Story

- Textual description of an entity from which we can derive requirements
- Technical specification
  - The technical details that refer to the structure and main ingredients of the entity
- Operational specification
  - Description of how the entity behaves and how it serves the goals of the stakeholders

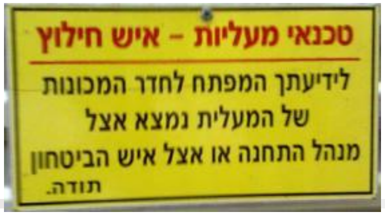
Technical Specification: Elevator System

An elevator system in an office building has the following elements:

- 3 elevators serving 10 floors
  - Every car has an independent controller + control software
- Every car has a panel for users with the following elements
  - Floor buttons with number 0, 1, 2, 3 …, 9
  - An emergency stop button
  - An alarm/rescue button

Technical Specification: Elevator System

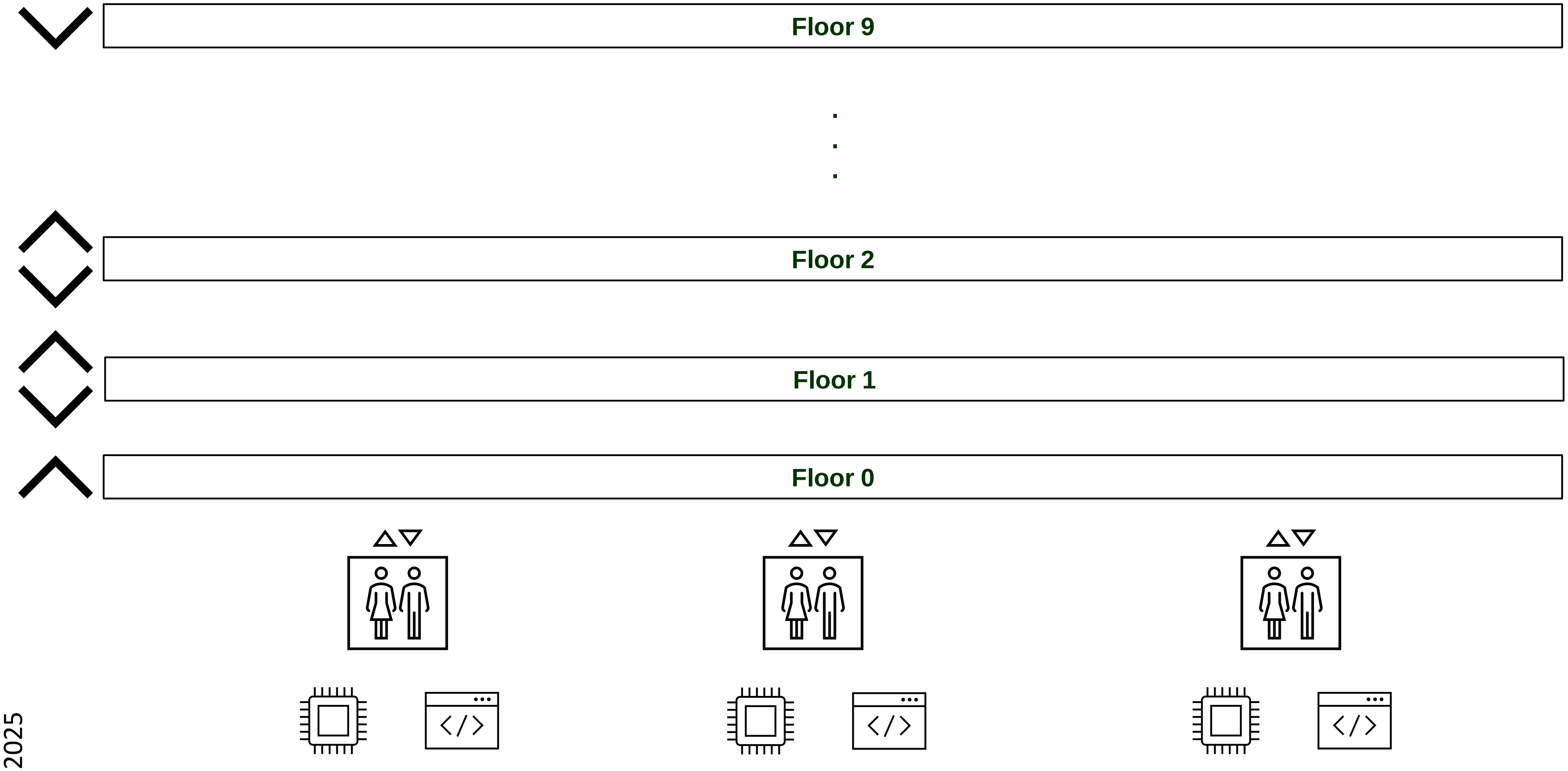
- Floors have up and down buttons
  - Top floor has only down. Bottom floor has only up
- Machine room with:
  - Computer + software to control and monitor all elevators
  - Emergency operator panel for rescue services from stuck elevators
    - Rescuer is a maintenance engineer in the building and is part of the service
  - Technician operator panel for testing and repairs
    - Technician is not part of the building and is an external service



טכנאי מעליות – איש חילוץ

לידיעתך המפתח לחדר המכונות
של המעלית נמצא אצל
מנהל התחנה או אצל איש הביטחון
תודה.

Elevators



Floor 9

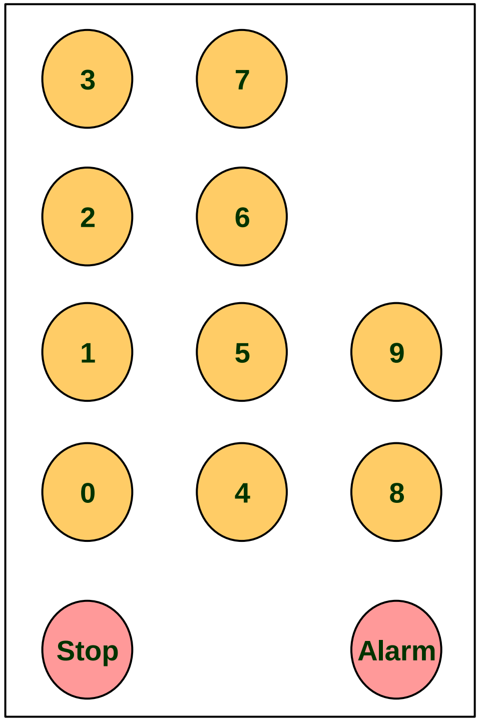
.
.
.

Floor 2

Floor 1

Floor 0

Elevator Panel



Operational Specification: Elevator System

A passenger who is on a particular floor and wants to call an elevator presses the appropriate button for the direction of travel (up or down). If the button was not already lit, the button lights up after being pressed. An elevator car traveling in the requested direction will arrive at the floor within one minute at most. When the car arrives, the door opens and the light on the button turns off.

A passenger who is in an elevator car and wants to travel to a particular floor presses on the button associated with the desired floor. If the button wasn't already lit, the button lights up and a new stop request for the floor is registered. The door closes. After a short pause, the car continues moving. It stops at every floor for which there is a stop request. When the car stops at a floor, the door opens and the button for the floor turns off.

A passenger can stop the elevator car when its moving by pressing on the emergency stop button. In that case, the elevator stops immediately and all registered stop requests are erased. Afterwards, the elevator can resume operation by pressing a button for any floor.

Operational Specification: Elevator System

If the elevator gets stuck while in operation, passengers can call for help using the emergency rescue button. The rescuer (the building maintenance engineer) will access the emergency panel in the machine room and perform operations to move the elevator car to the bottom floor and open the door.

The maintenance engineer is responsible to start up the elevators at the beginning of the day and to shut it down at the end of the day. The technician comes once every 6 months and performs a complete check of the system and fixes problems using the technician panel in the machine room.

The elevator system must meet all applicable safety rules.

The system must be accessible to the handicapped.

Discussion Questions

1. Give example requirements that can be derived from the technical and operational stories

2. Can you discern general requirements from requirements at the feature level?

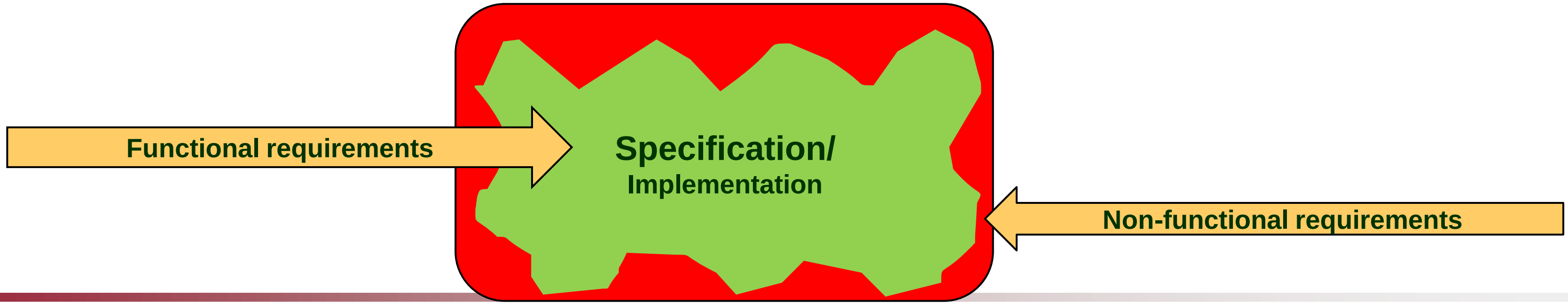
Functional and Non-Functional Requirements

Functional Requirements

- Define the contents of the solution
- Are clearly and specifically addressed in the solution (design/implementation)

Non-Functional Requirements

- Define properties and constraints on the way the solution is created/implemented
- Are addressed when the solution meets the required properties and constraints



Categorizing requirements for SIS:
Functional Requirements: Specify what the system must do

Operational Requirement (OR)

- Refers to an operation, interaction, data processing step, or system behavior
- Operations, scenarios, response to events
- Functions, services, algorithms
- Timing, order of operations

Data Requirement (DR)

- Refers to data elements – but don’t specify what the system does with it
- Data about the system
- Data the system processes
- Input/output data

Operational requirements: What the system must do
Data requirements: What the system must know

Examples of Functional Requirements

Operational

- “rider …presses the appropriate button”
- “the button lights up after being pressed”
- “If the technician finds a bug, he tries to fix it and perform the test again”

Data

- Floors have up and down buttons. Top floor has only down. Bottom floor has only up

Note

- Operational requirements include data requirements within them (i.e. the data they need work)
  - E.g. turning on/off the light implies the light has two states – on and off
- Software generally “knows” details about the physical aspects of the system (especially parts it controls).

Non-Functional Requirements: Quality of the solution

Specify additional aspects of the solution that must be met while meeting the functional requirements

Performance Requirement (PR)

- Parameters that measure the speed of actions
- Response time, data size, processor utilization

Quality Attributes (QA)

General aspects of the solution

- Reliability: Works without errors for a certain amount of time
- Availability: Continuous service, fast recovery from errors
- Safety: Protects users and the environment from the system
- Security: Protects the system from users
- Testability: Ability to test and verify the systems actions (also after the fact)
- Maintainability: Ability to easily change and repair the product
- Usability: Effectiveness and efficiency that the system gives users in performing their tasks and reaching their goals

For QA, ensure the requirement is measurable and verifiable!

Non-Functional Requirements: Constraints

Conditions and limitations that affect the choice of the solution

Hardware Constraint (HC)

- Hardware that must be used for the system
  - Parts, interfaces, architecture
- Based on the customer's technical specification or a higher-level system design

Implementation Constraint (IC)

- Specific implementation choice that must be used
  - Algorithm, data structure, data base, particular operational behavior, reuse of data, use of a technology

Non-Functional Requirements: Constraints

Conditions and limitations that affect the choice of the solution

Management Constraint (MC)

- Management conditions that must be followed by the implementation
  - Budget, schedule
  - Availability of resources
  - Following standards

Examples of Non-Functional Requirements

Performance

- An elevator car traveling in the requested direction will arrive at the floor within one minute at most.

Testability

- The technician comes once every 6 months and performs a complete check of the system

Maintainability

- The technician …fixes problems

Usability

- The system must be accessible to the handicapped

Hardware Constraint

- Every car has an independent controller + control software

Implementation Constraint

- A rider can stop the elevator car when its moving by pressing on the emergency stop button.

Management Constraint

The elevator system must meet all applicable safety rules.

Conclusion

- Software from a Systems Perspective
- Development Process of a Software Intensive System
- Requirements and Stakeholders
- Elicitation and Management