Lesson 1

Software Development Methodologies

Wholeness

- A Software Development methodology is a process followed by a team in making a software product.
- Following a methodology is essential for successful execution of a software project because the process lays out a structured sequence of steps/activities that guide the team through each stage of development.
- *Science of consciousness: Life is structured in layers*

Methodologies

- There are many methodologies in existence. Notable ones: **Waterfall**, **RUP**, **Agile**, **DevOps** etc.
- For a successful project, the team must choose an appropriate methodology that will work best for the project.
- All have different strengths and weaknesses and serve different needs.

Success in software development

Late, over budget, and/or with less features and functions than defined in the initial scope

On-time, on-budget, and with all features and functions as defined in the initial scope

	2011	2012	2013	2014	2015
SUCCESSFUL	29%	27%	31%	28%	29%
CHALLENGED	49%	56%	50%	55%	52%
FAILED	22%	17%	19%	17%	19%

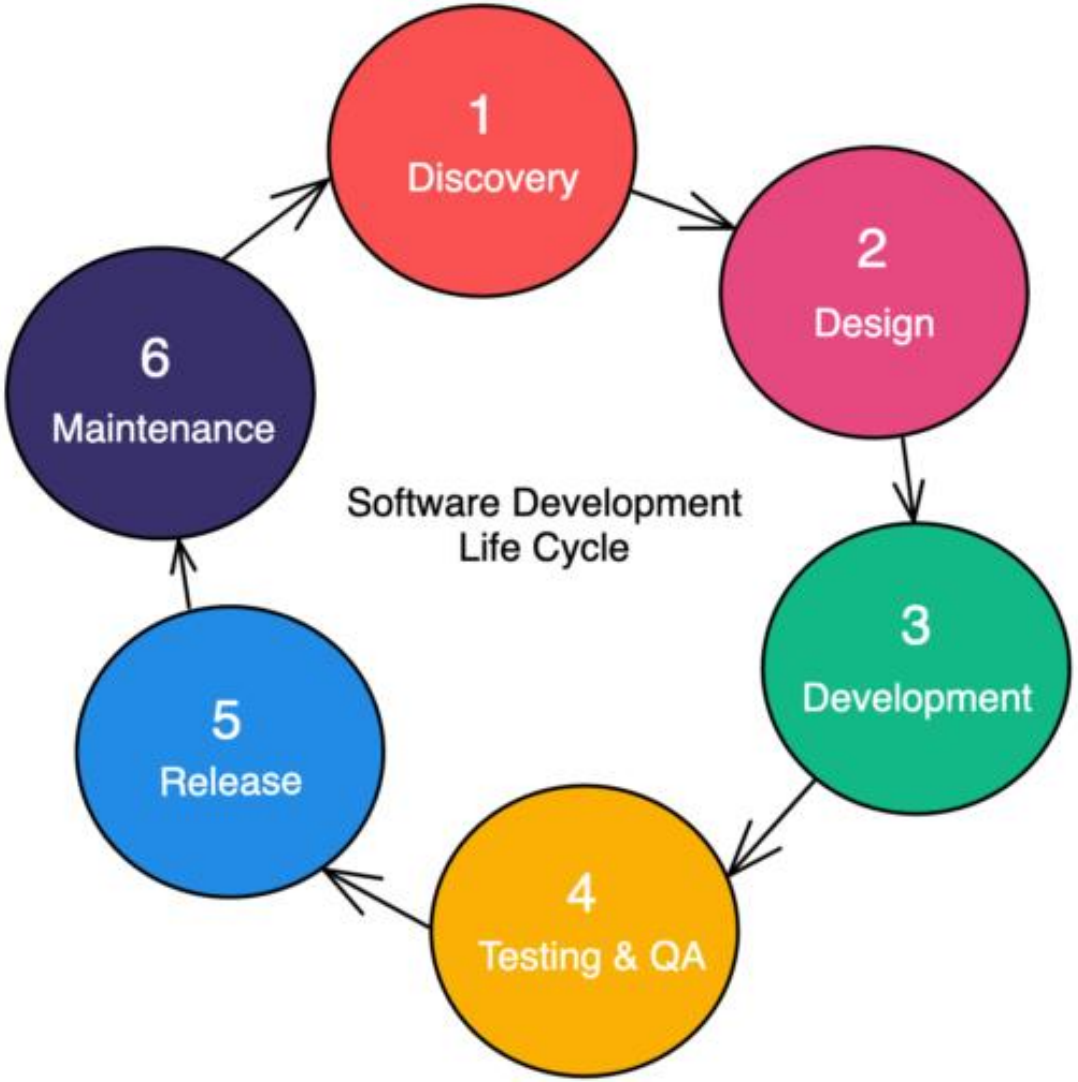
Cancelled prior to completion, or delivered but never used

Software methodology

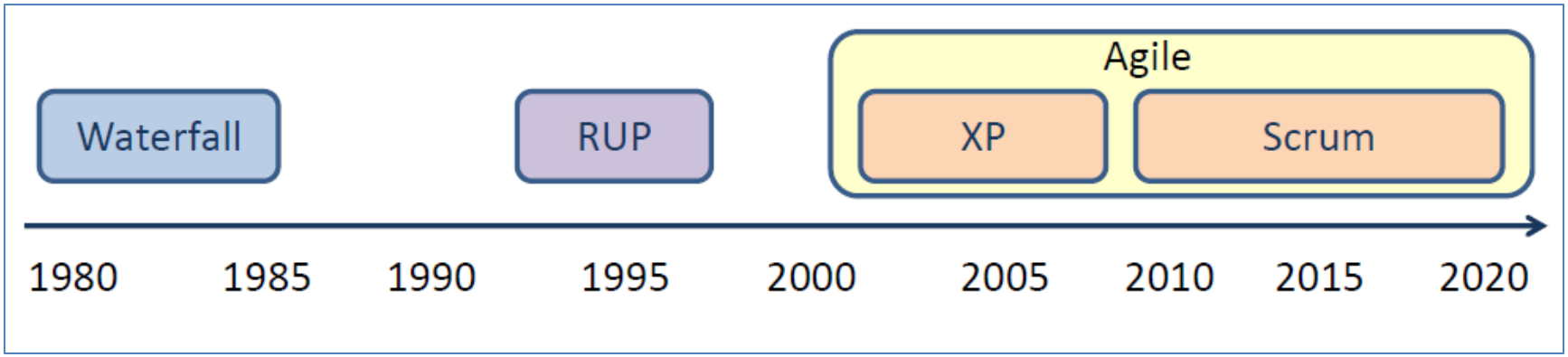
- Who
  - Roles of the people in the project
- What
  - What artefacts are created or used
- When
  - The order in which activities are done/performed
- How
  - What disciplines, activities, best practices etc.

Software Development Lifecycle

- Typically, a Software Development project will involve the following lifecycle of activities:

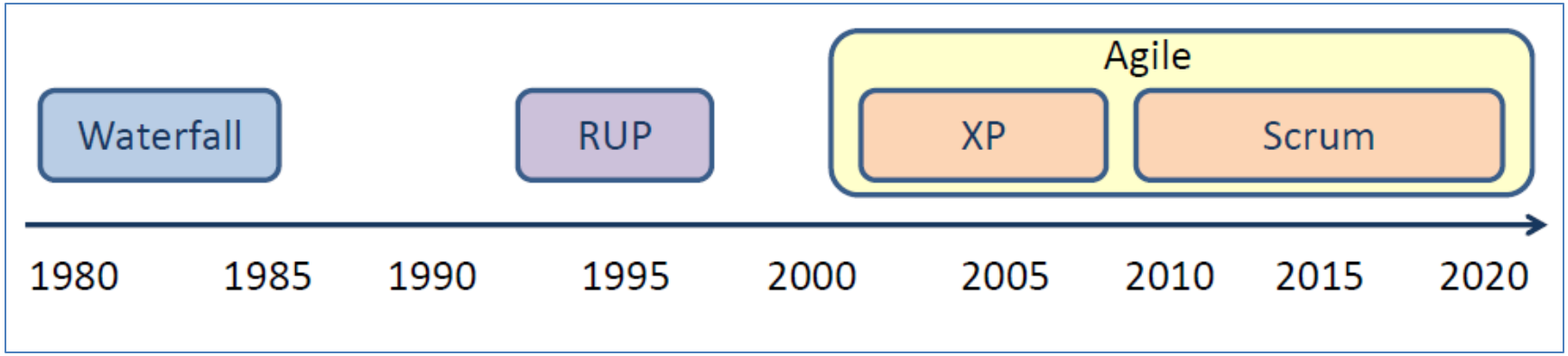


Evolution of the software development methodologies



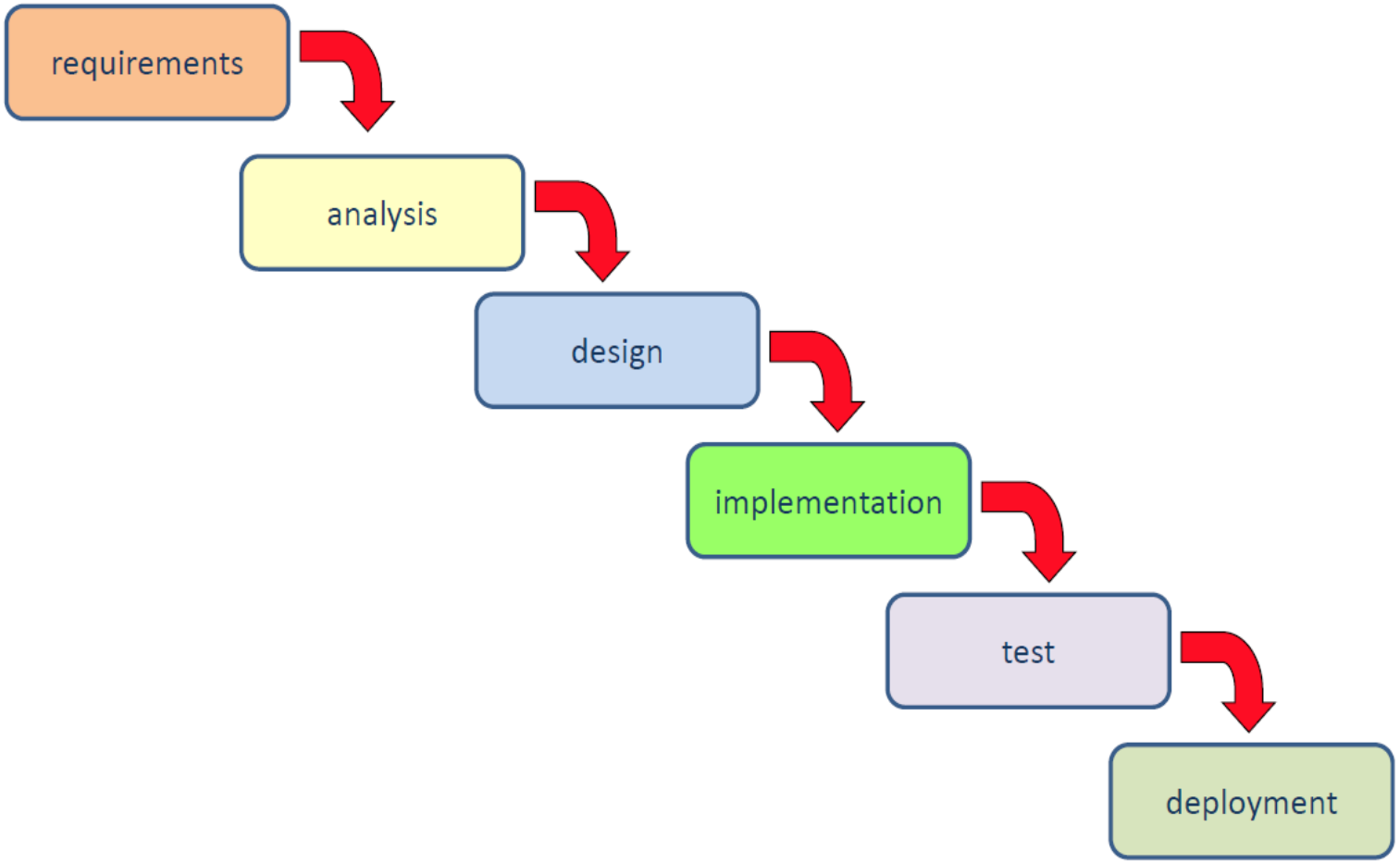
Waterfall methodology

Software development methodologies: Waterfall



- Linear
- Different roles
- Document driven
- Customer is outside the project
- Large projects(time, nr. of people)
- Req. statements

Waterfall methodology



Real-World Contexts

1. E-commerce Website	**Requirements**: Customers can browse products, add them to a cart, and check out.	**Analysis**: Define entities like Products, Users, and their relationships. Create use cases for adding items to a cart or processing payments.	**Design**: Define APIs for GET /products, POST /cart, and POST /checkout. Design database tables for Products, Users, and Orders.
2. Mobile Banking App	**Requirements**: Users can transfer money, check their balance, and view transaction history.	**Analysis**: Create a sequence diagram for transferring money (e.g., authentication -> select account -> confirm transaction).	**Design**: Design UI screens for transfer and balance checks. Create database schemas for Accounts and Transactions.
3. House Construction	**Requirements**: The house must have 3 bedrooms, 2 bathrooms, and a kitchen.	**Analysis**: Create a floor plan showing the layout of rooms and dimensions.	**Design**: Choose construction materials, plumbing and electrical layouts, and architectural drawings.

Req vs Analysis vs Design

Requirements define **what** the system must do, focusing on stakeholder needs.

Analysis determines **how** to meet these requirements conceptually and ensures feasibility.

Design specifies the **technical implementation** of these plans, creating detailed blueprints for development or construction.

Real-World Contexts

Phase	E-commerce Website	Mobile Banking App	House Construction
Implementation	Code shopping cart and APIs, integrate payment gateway.	Build UI, secure APIs, integrate with core systems.	Build foundation, install plumbing and wiring.
Testing	Test cart functionality, simulate user traffic.	Validate transactions, test app security.	Inspect structure, test plumbing and wiring.
Deployment	Deploy to AWS, monitor traffic, enable CI/CD.	Publish to app stores, monitor usage.	Clean up, hand over keys, ensure utility setup.

Core Roles

- Project Manager
- Analyst
- Developer
- Tester
- Architect

Project Manager

- Check project status
- Facilitate the team
- Create funding
- Acquire resources
- Communication with the business
- Planning
- Task distribution
- Solving problems
- Manage risks
- Check project progress
- Manage quality

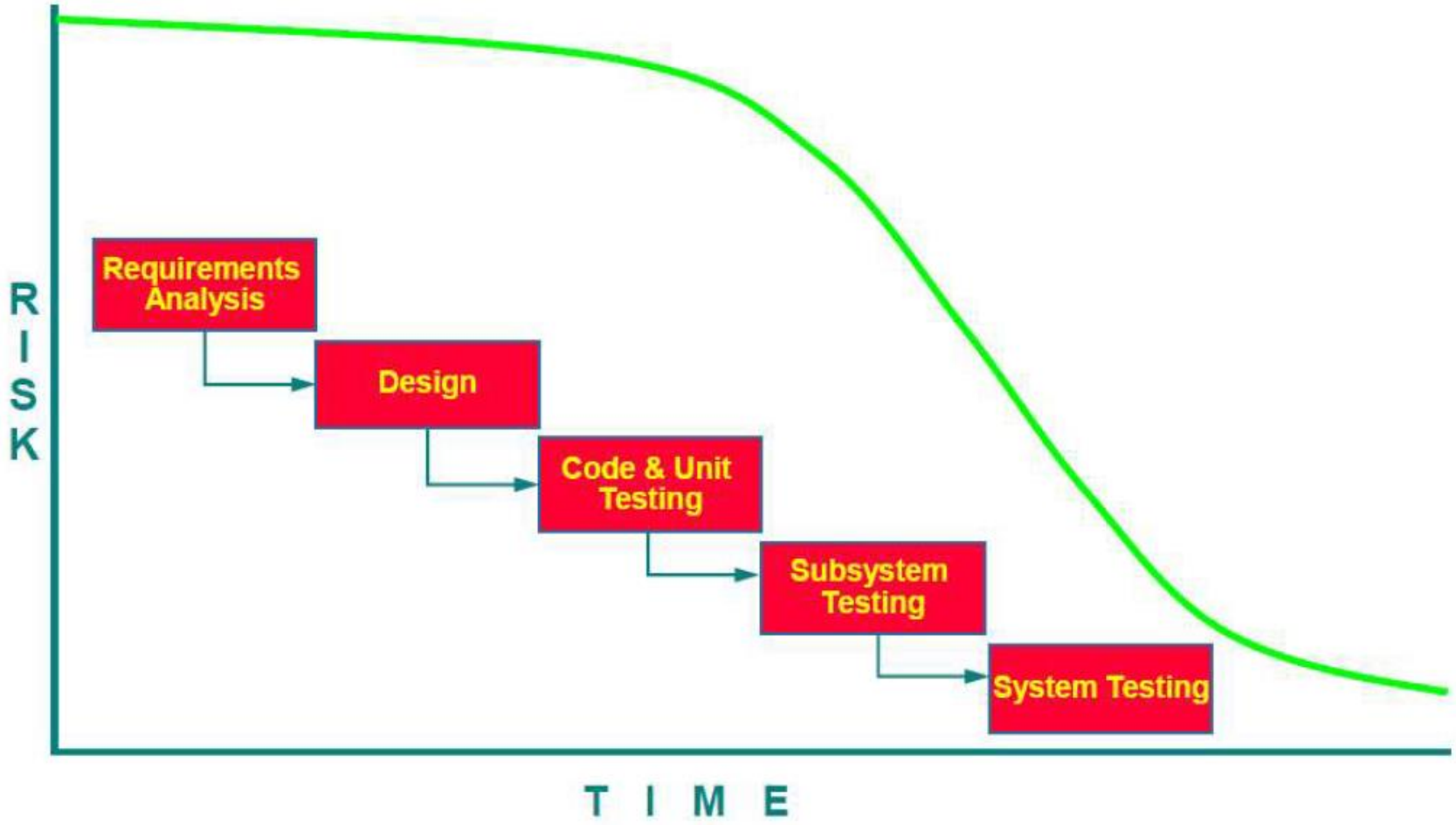
Core Artefacts

- Planning document
- Requirements document
- Architecture document
- Design document
- Code
- Test plan

Artefacts of each phase

Phase	Artifacts
Requirements	SRS, Use Case Diagrams, Stakeholder Requirements, Constraints Document
Analysis	Feasibility Report, DFD, Risk Analysis Document
Design	System Architecture, Database Schema, ERD, Component Diagrams, UI Mockups
Implementation	Source Code, Version Control Logs, Build Scripts, Code Documentation
Test	Test Cases, Test Plan, Bug Reports, Test Results Report
Deployment	Deployment Plan, Release Notes, Installation Scripts, Monitoring Configuration

Risk

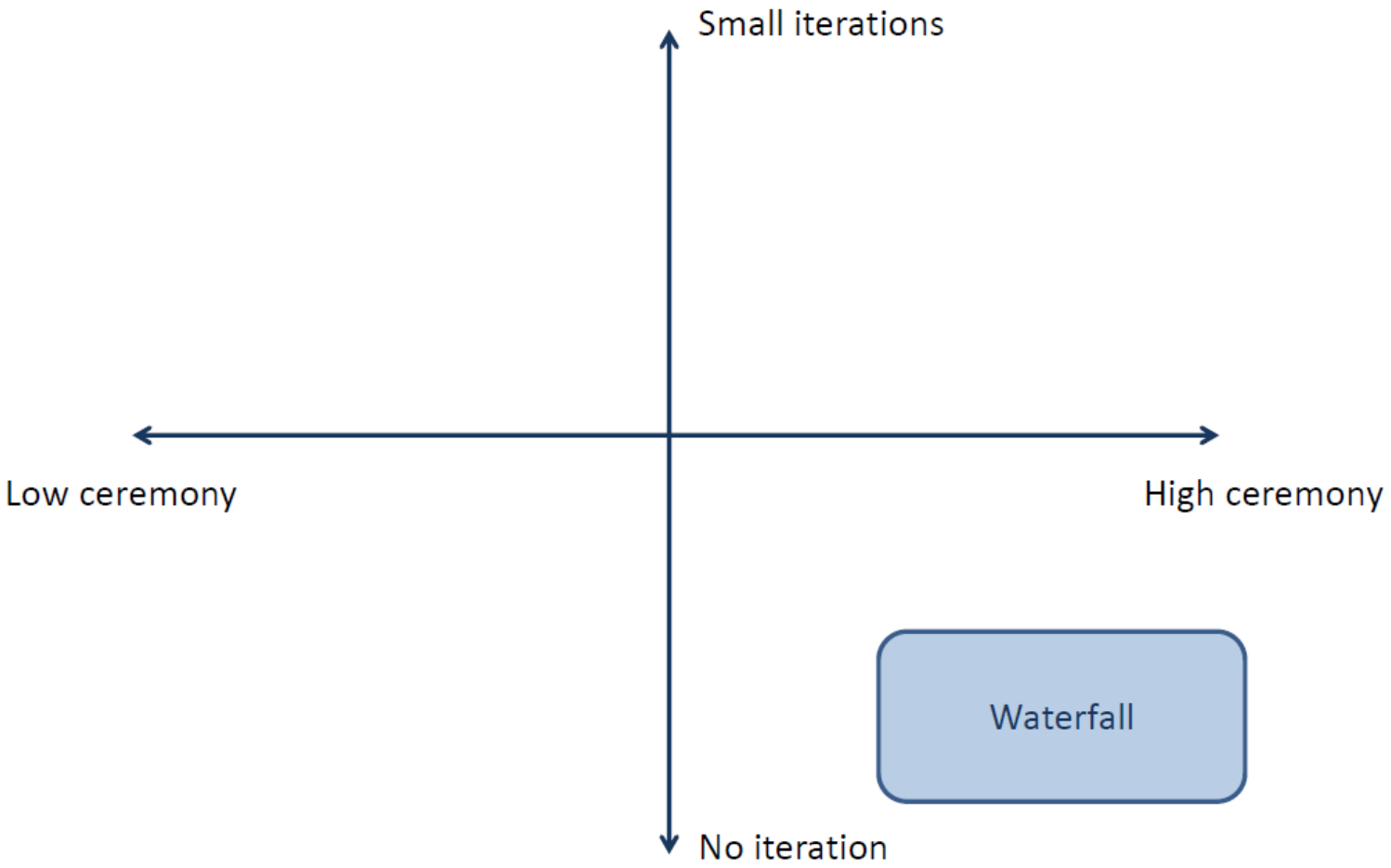

RISK
Requirements Analysis
Design
Code & Unit Testing
Subsystem Testing
System Testing
TIME

Characteristics of Waterfall

Document driven	The customer is involved only at the beginning of the project
Risks are found late in the project	Lot of different roles
Requirements are frozen	Software can be used only at the end of the project
Throw artefacts over the wall	Not much possibilities for reflection and improvement
Project status is not clear	No feedback
No possibilities to learn other disciplines	There is no time left for testing

- Waterfall is highly risky, inefficient and static
- It works for the Project Manager but not for the dev team

Software development methods

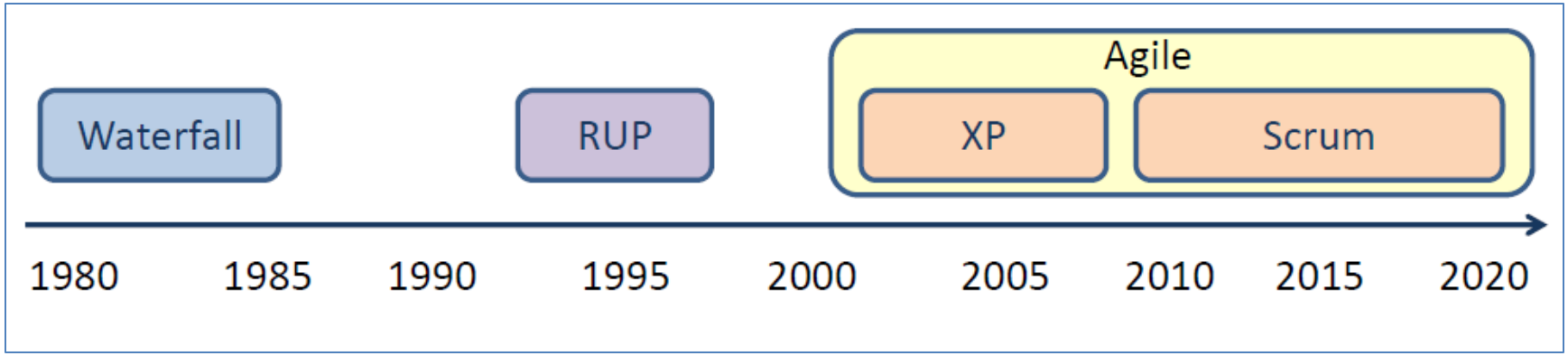


Sample Projects

- Restaurant Setup
- Car Manufacturing
- Build a shopping mall
- Movie Production
- School Management System
- Hospital Construction

Rational Unified Process (RUP) methodology

Software development methodologies: RUP



Linear | Iterations

Different roles

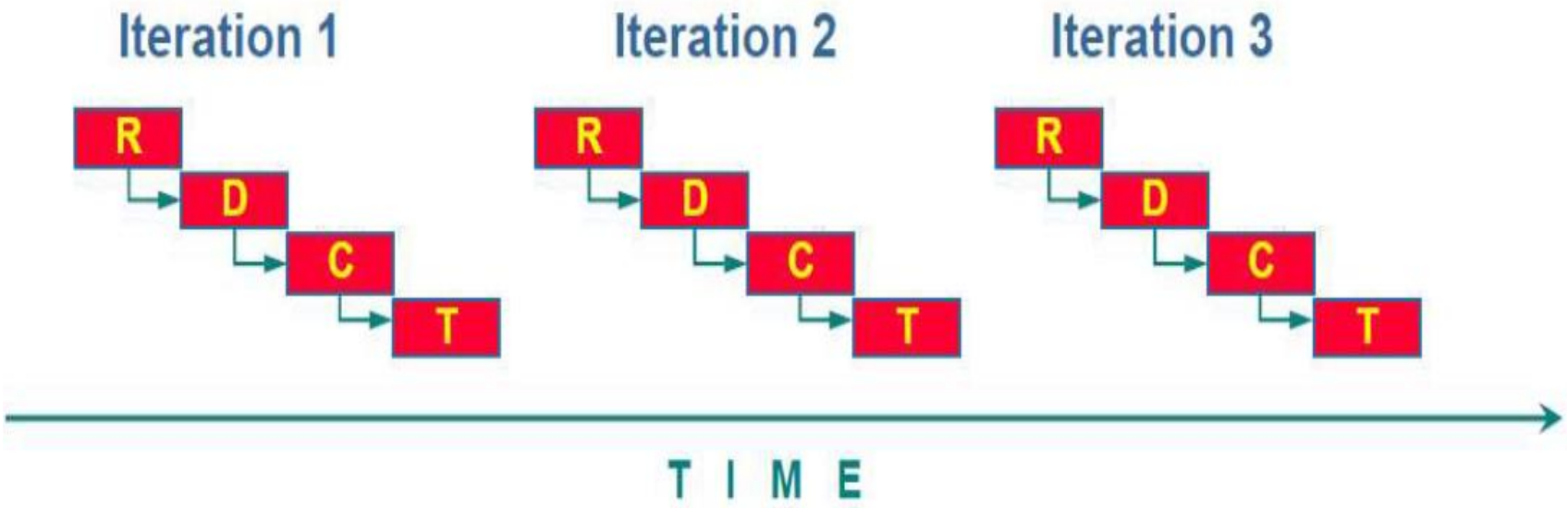
Document driven

Customer is outside the project

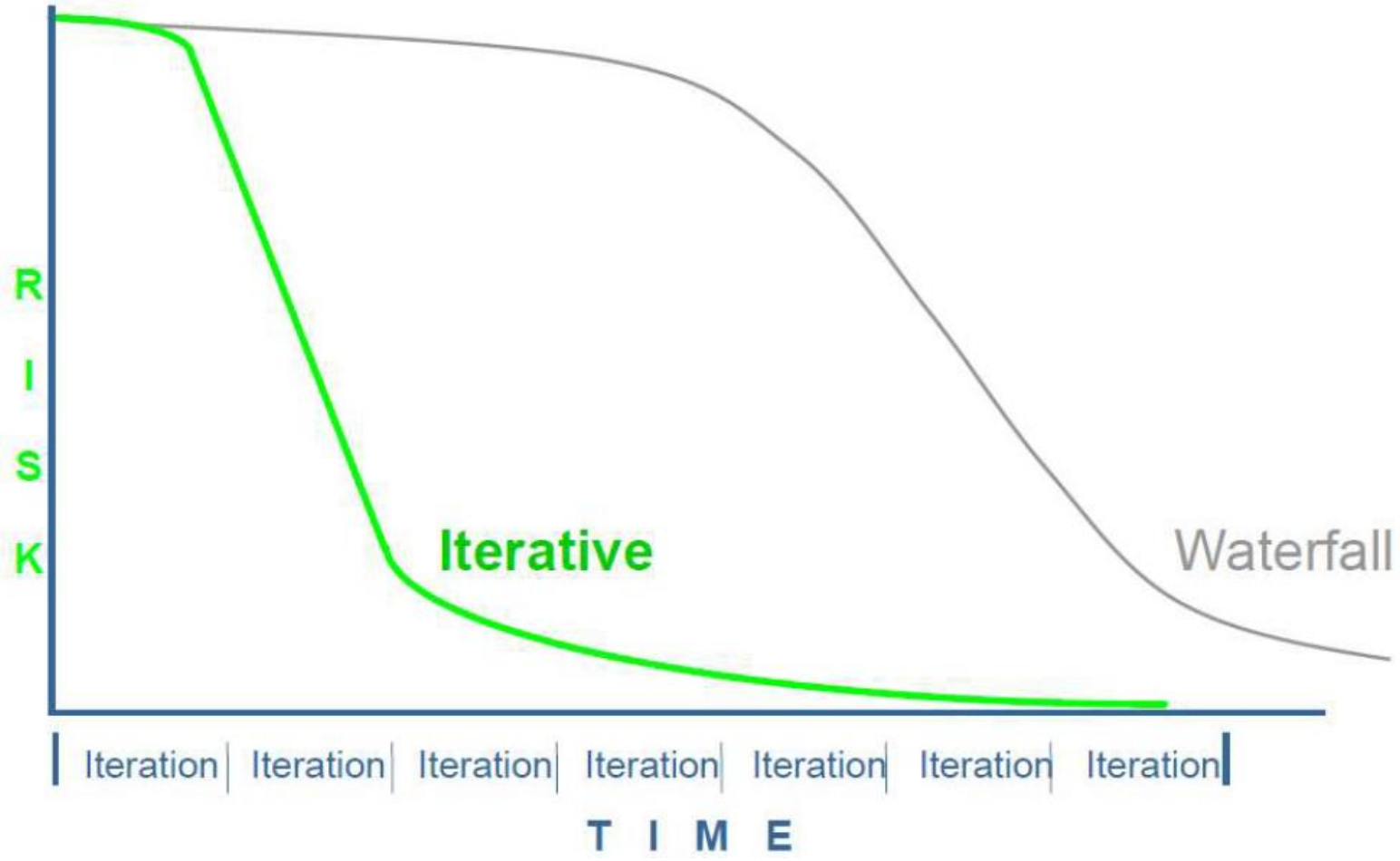
Large projects(time, nr. of people)

Req. statements | Use cases

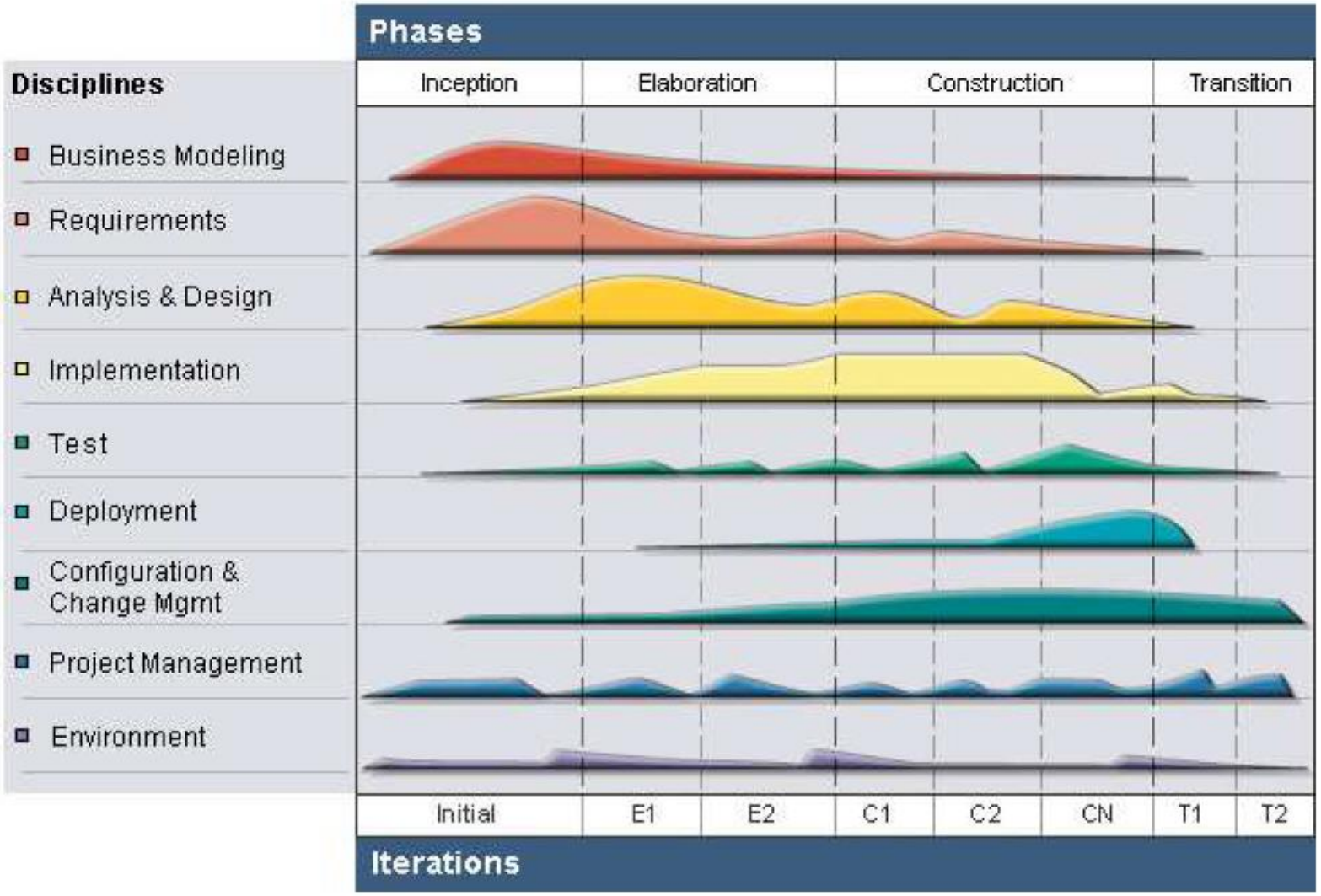
Iterations



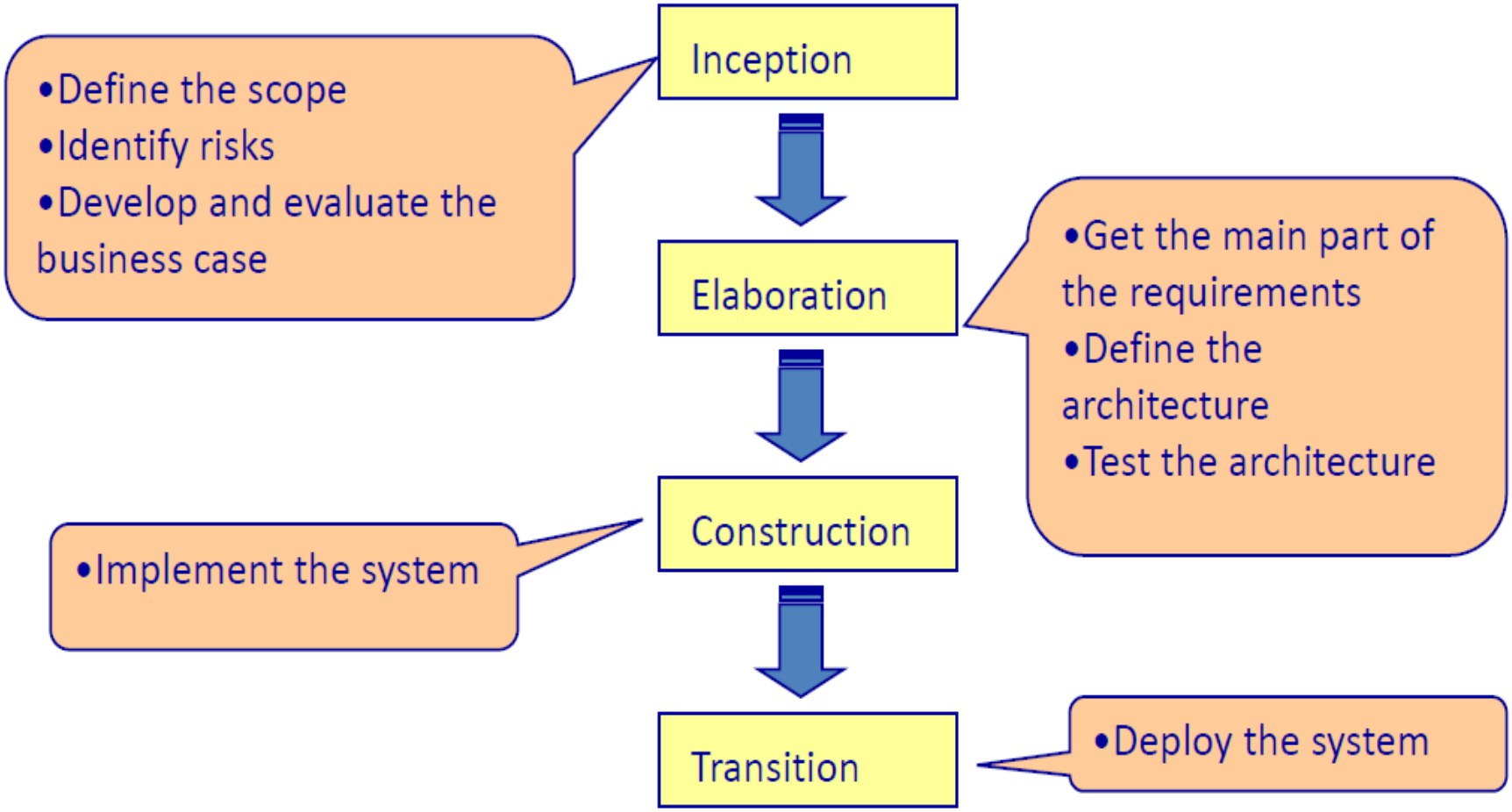
Risk


R
I
S
K
Iterative
Waterfall
Iteration
Iteration
Iteration
Iteration
Iteration
Iteration
Iteration
T I M E

Rational Unified Process (RUP)

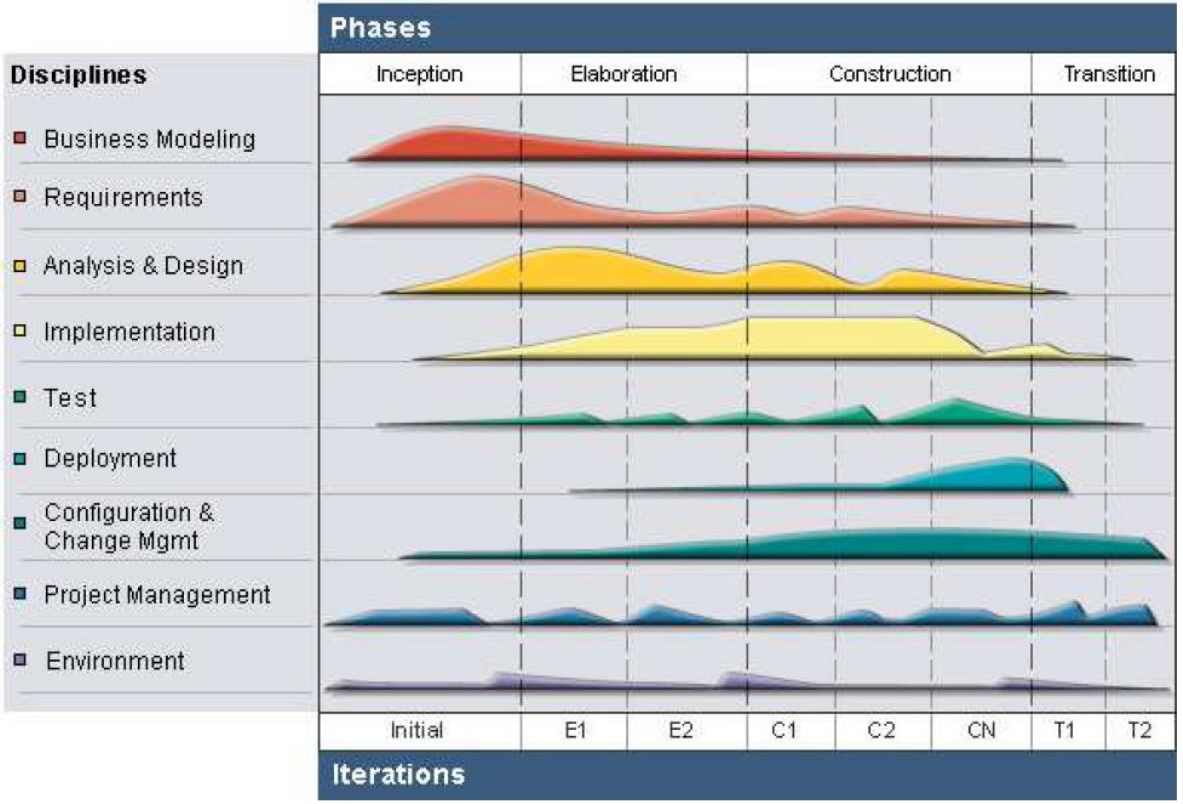


RUP Phases


Inception
•Define the scope
•Identify risks
•Develop and evaluate the business case
Elaboration
•Get the main part of the requirements
•Define the architecture
•Test the architecture
Construction
•Implement the system
Transition
•Deploy the system

RUP characteristics

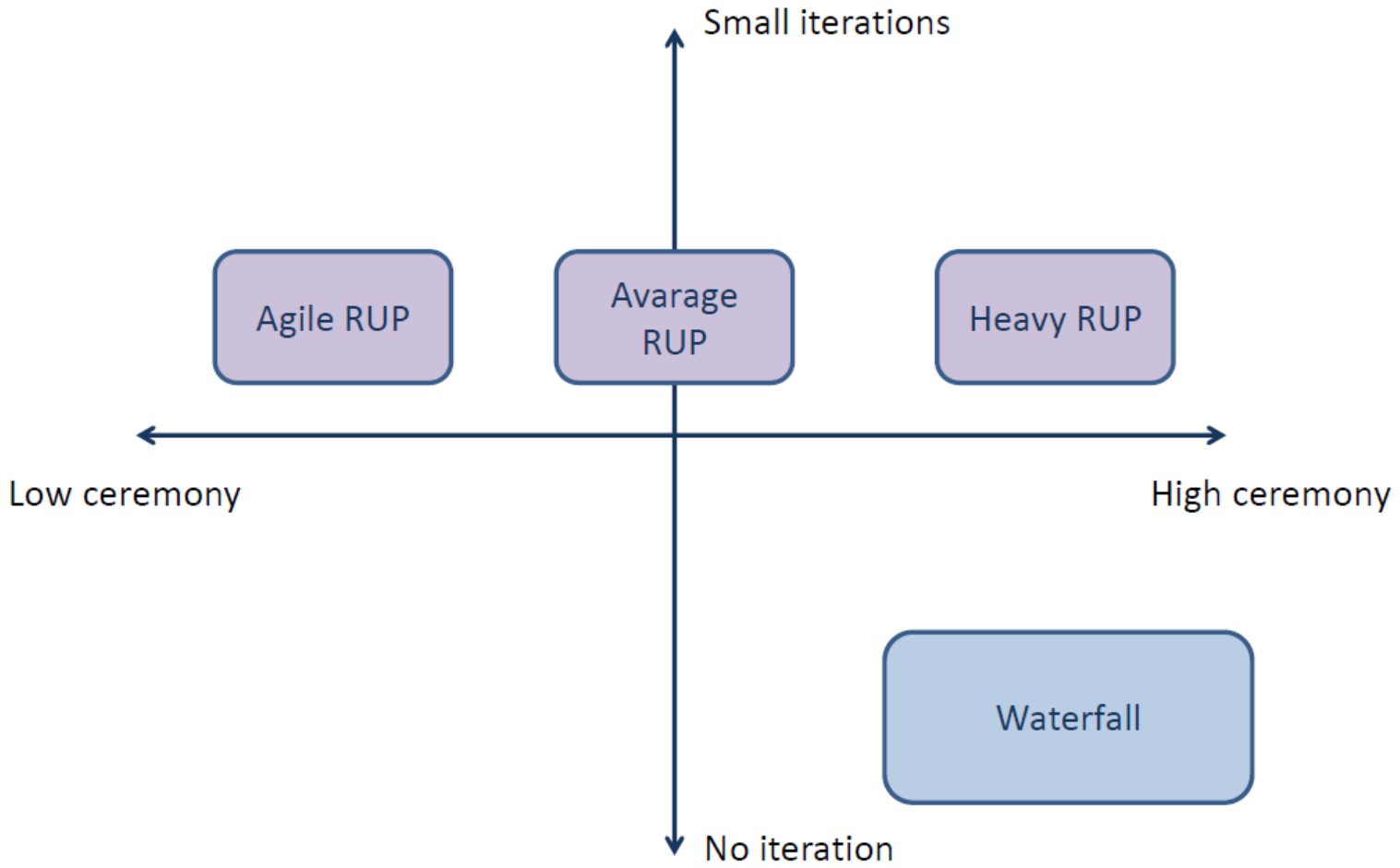
- Iterations
- Use-case driven
- Visual modeling: UML
- Architecture centric
- Test everything
- Manage changes



Roles

- Analysts
  - Business Architect
  - Business Designer
  - Business-Process Analyst
  - Requirements Specifier
  - Stakeholder
  - System Analyst
- Developers
  - Capsule Designer
  - Database Designer
  - Designer
  - Implementer
  - Integrator
  - Security Architect
  - Software Architect
  - User-Interface Designer
- General Roles
  - Review Coordinator
  - Reviewer
  - Stakeholder
  - Technical Reviewer
- Managers
  - Change Control Manager
  - Configuration Manager
  - Deployment Manager
  - Management Reviewer
  - Project Manager
  - System Administrator
  - Test Manager
- Production & Support
  - Course Developer
  - Graphic Artist
  - Process Engineer
  - System Administrator
  - Technical Writer
  - Tool Specialist
- Testers
  - Test Analyst
  - Test Designer
  - Test Manager
  - Tester

Software development methods



Unified Modeling Language (UML)

What is the UML?

- Unified Modeling Language
- A graphical language for drawing models
- Originally developed for modeling Object-Oriented software systems
- Contains 13 different types of diagrams

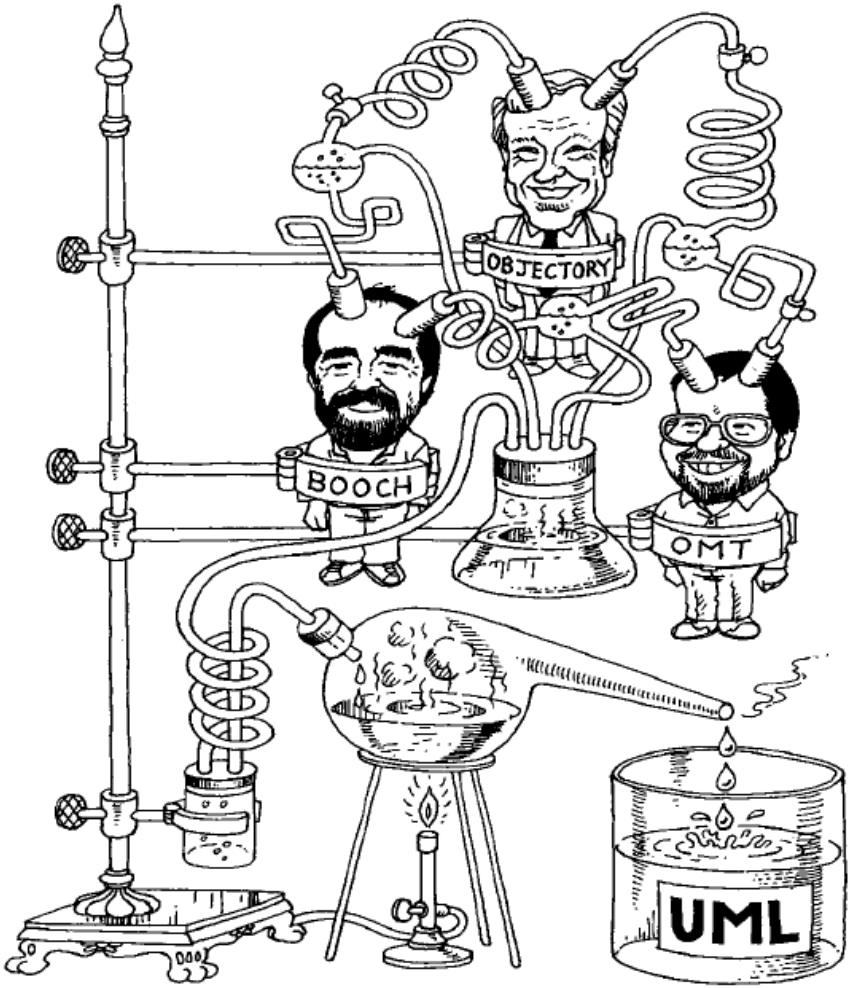
  – Note: The UML is NOT a software development method by itself

Model

- More complexity -> More Modeling
  - Higher level of abstraction
  - Allows for visualization
  - Vehicle of communication

Origin of the UML

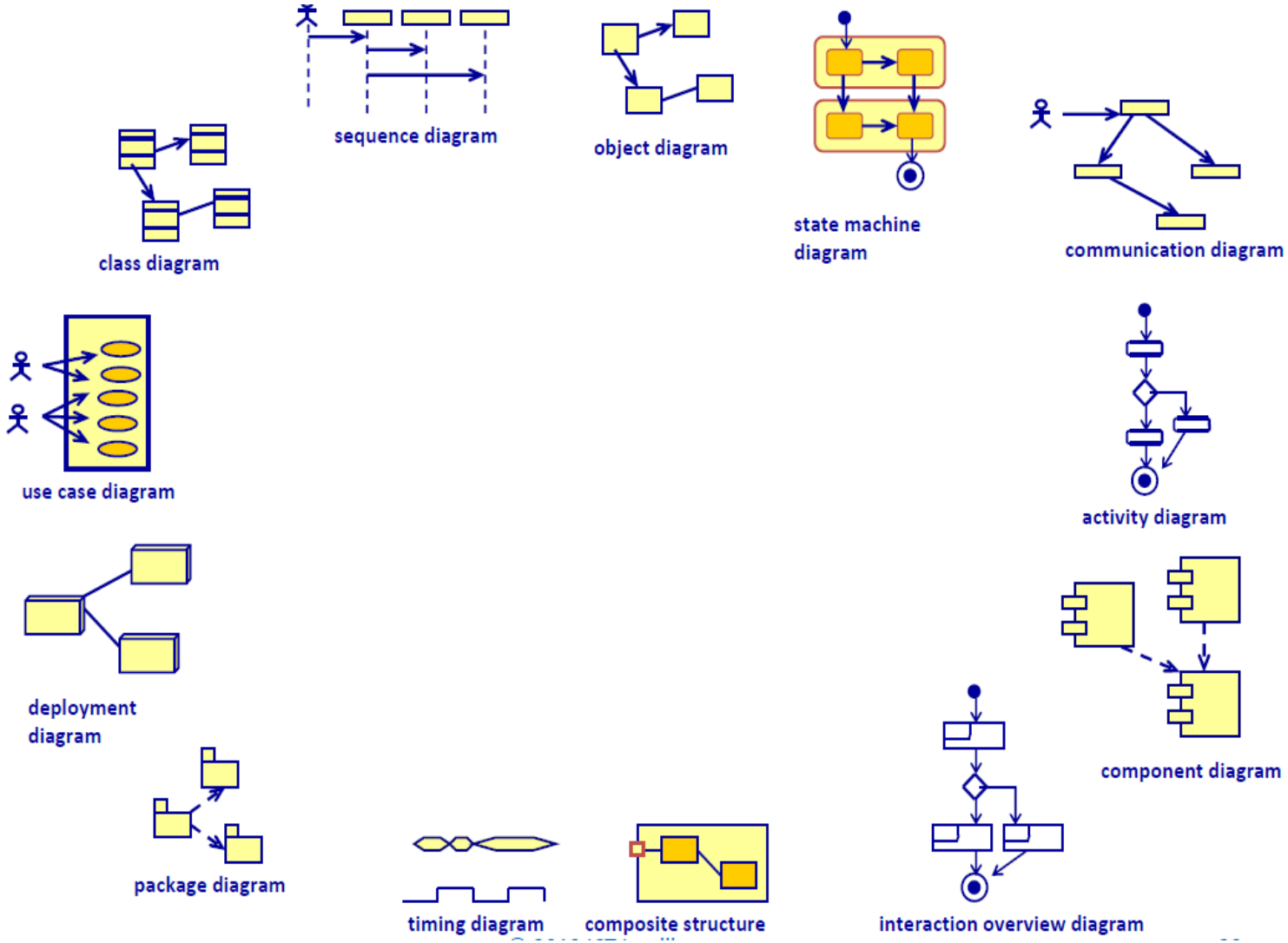
- Rational Software Corporation
  - Booch
  - Jacobsen
  - Rumbaugh
- Object Management Group (OMG)



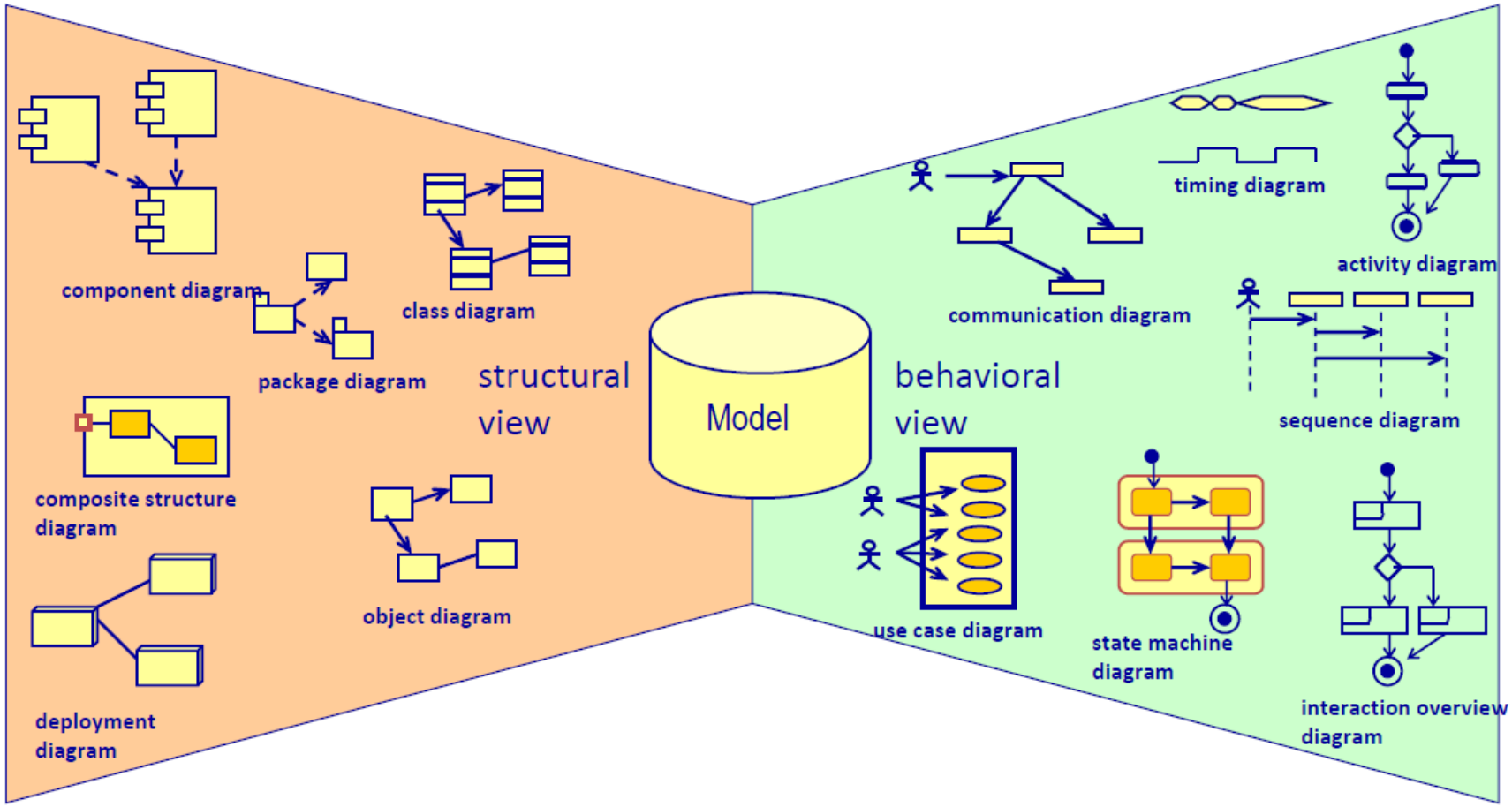
Why UML?

- UML is an ISO standard
- UML supports
  - Abstraction
  - Decomposition
  - Zoom-in and zoom-out functionality
  - Different views on the same model
- Tools support e.g. StarUML (free)

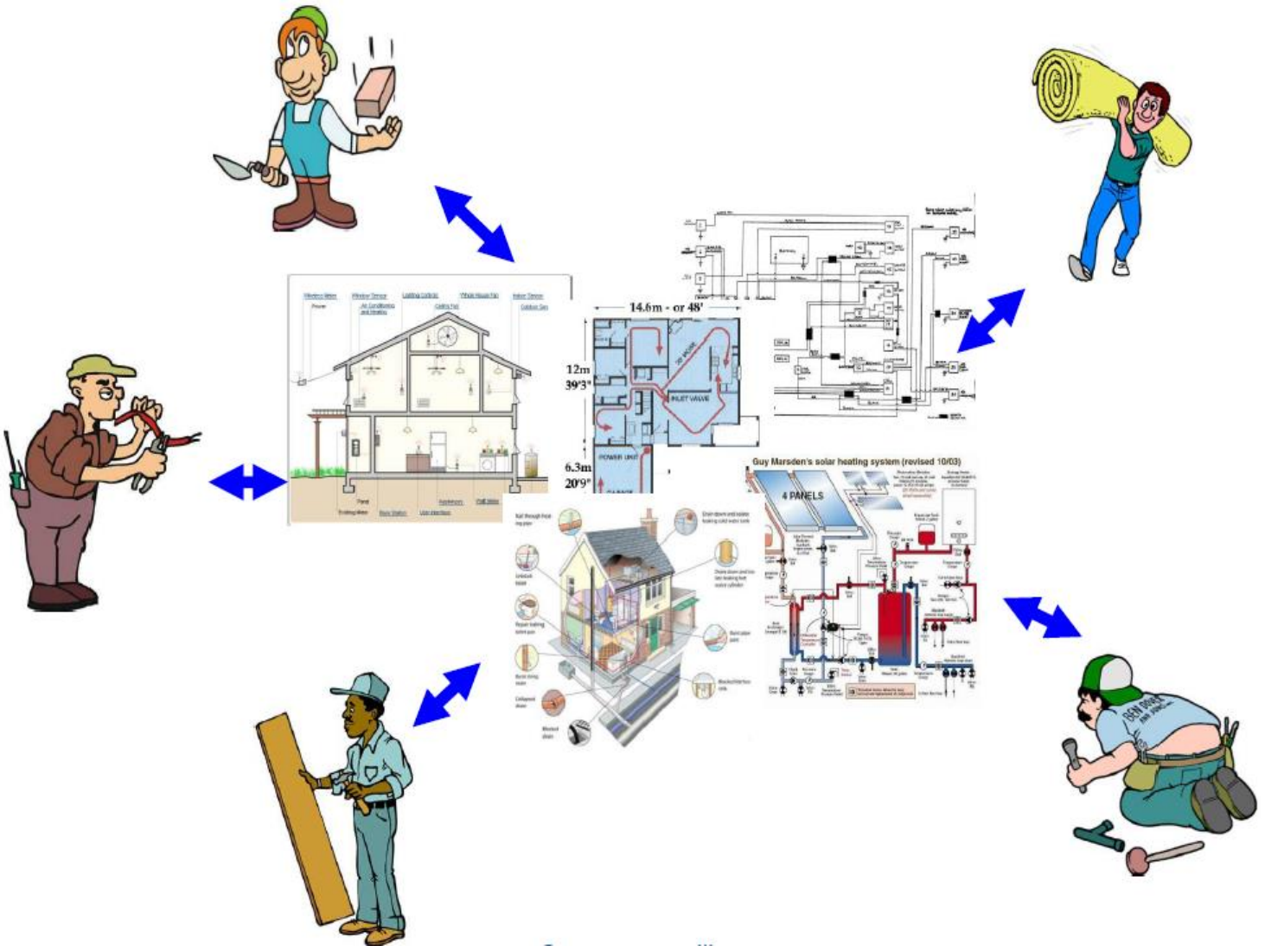
UML Diagrams



Diagrams and views



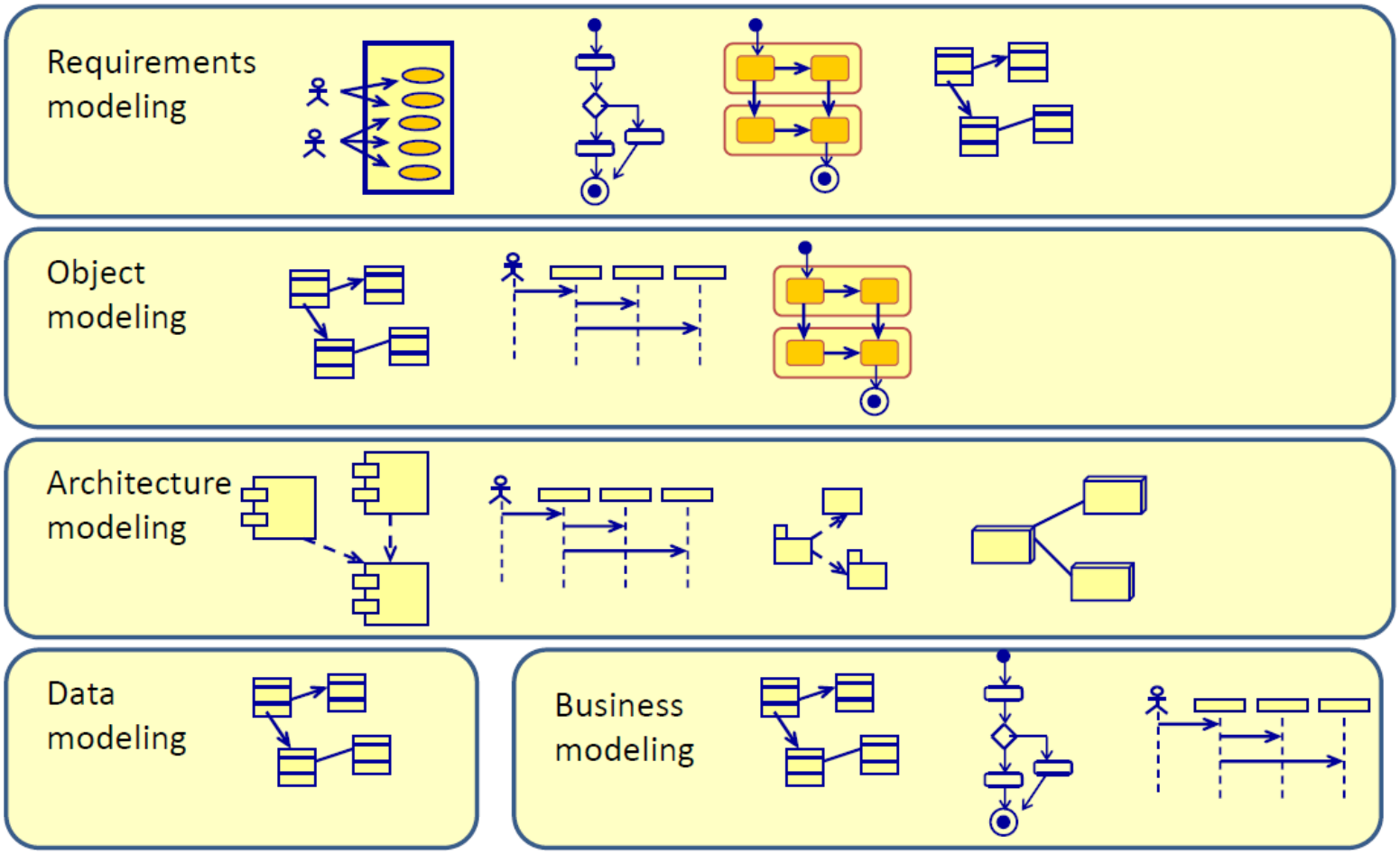
Model and views



Some popular UML tools

- StarUML
- Enterprise Architect
- Microsoft Visio
- https://drawio.com
- Lucid chart - https://www.lucidchart.com/pages/
- etc…

Applying the UML

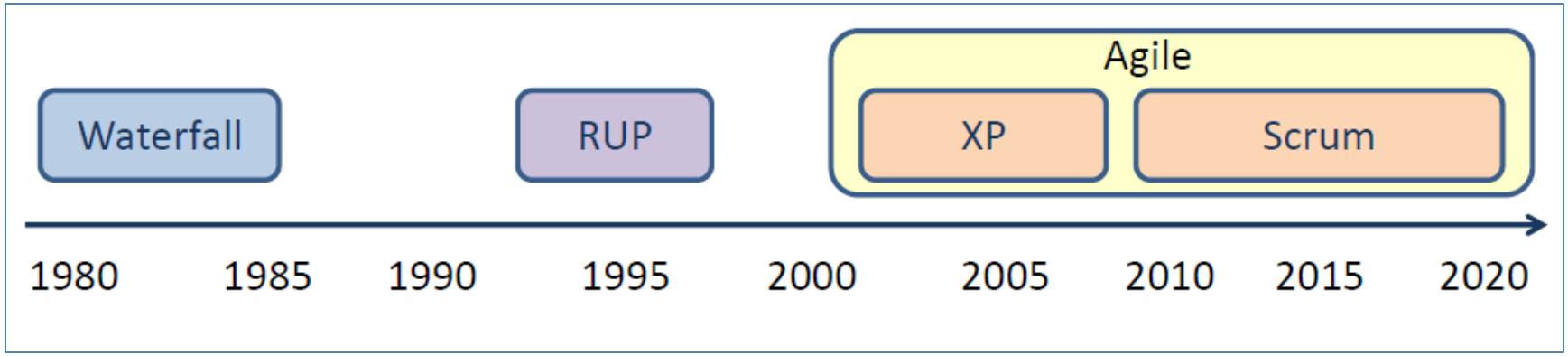


Agile Software Development methodology

What is the Agile methodology?

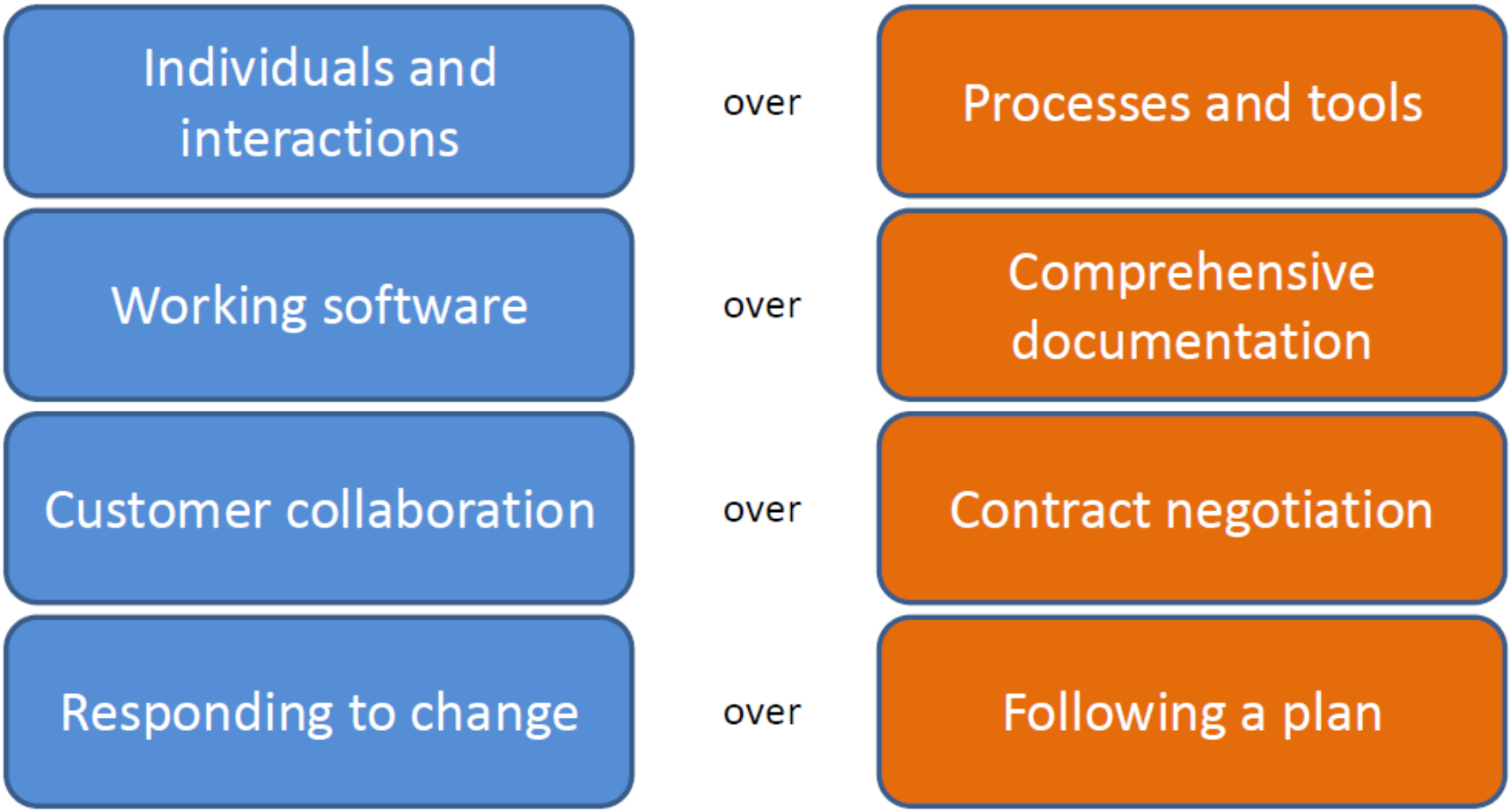
- Agile software development is a set of methods that results in fast and frequent delivery of value to the customer.
- It promotes well-planned, small iterations by highly collaborative, cross-functional team.
- Agile methods provide a better alternative to the linear/sequential development and long release cycles associated with the Waterfall approach.

Software development methodologies: Agile



Linear	Iterations	
Different roles		Cross-functional team
Document driven		Face-to-face
Customer is outside the project		Customer inside project
Large projects(time, nr. of people)		Small projects
Req. statements	Use cases	User stories

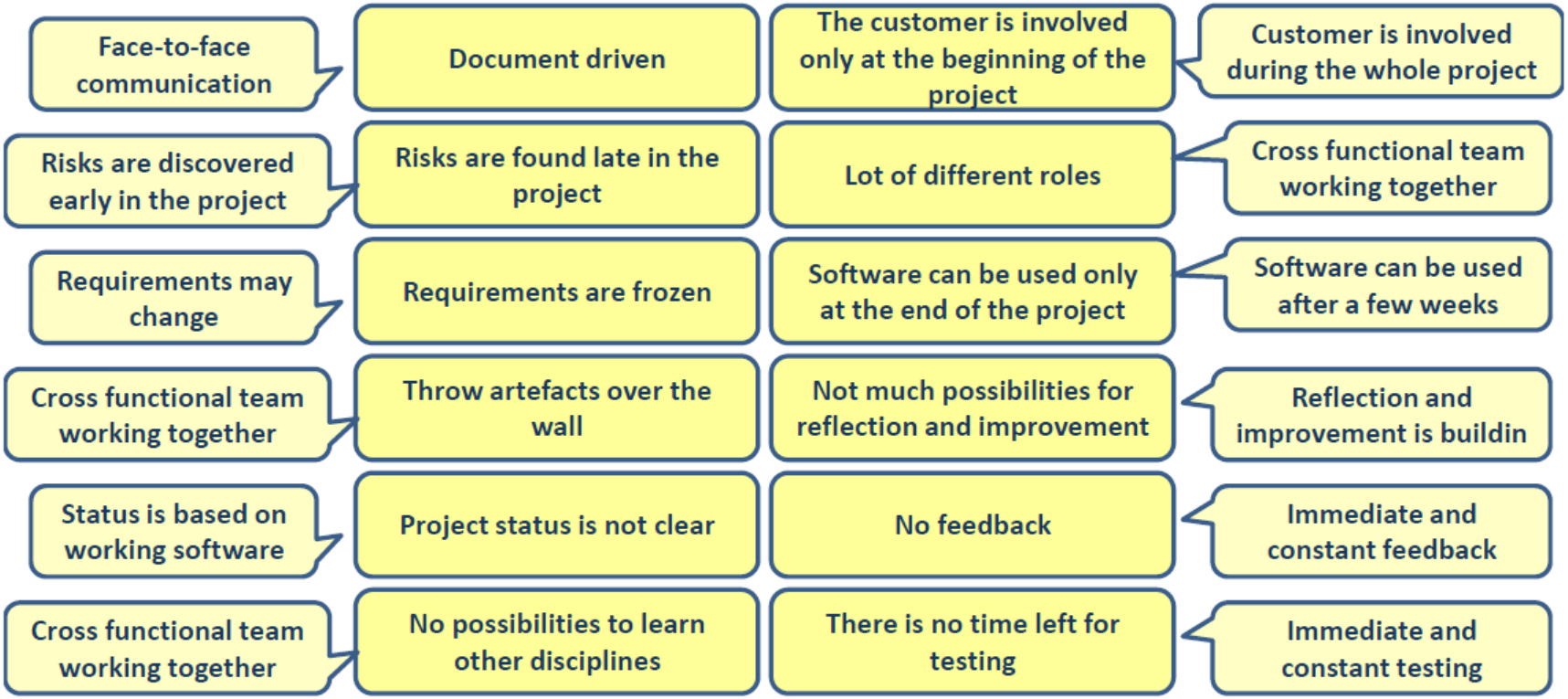
The Agile Manifesto



Agile principles

- Early and continuous delivery of valuable software.
- Welcome changing requirements.
- Business people and developers must work together daily.
- Give the team the environment and support they need, and trust them to get the job done.
- Prefer face-to-face conversation.
- Working software is the primary measure of progress.
- Continuous attention to technical excellence and good design
- Simplicity is essential.
- Self-organizing teams.

How is Agile different?

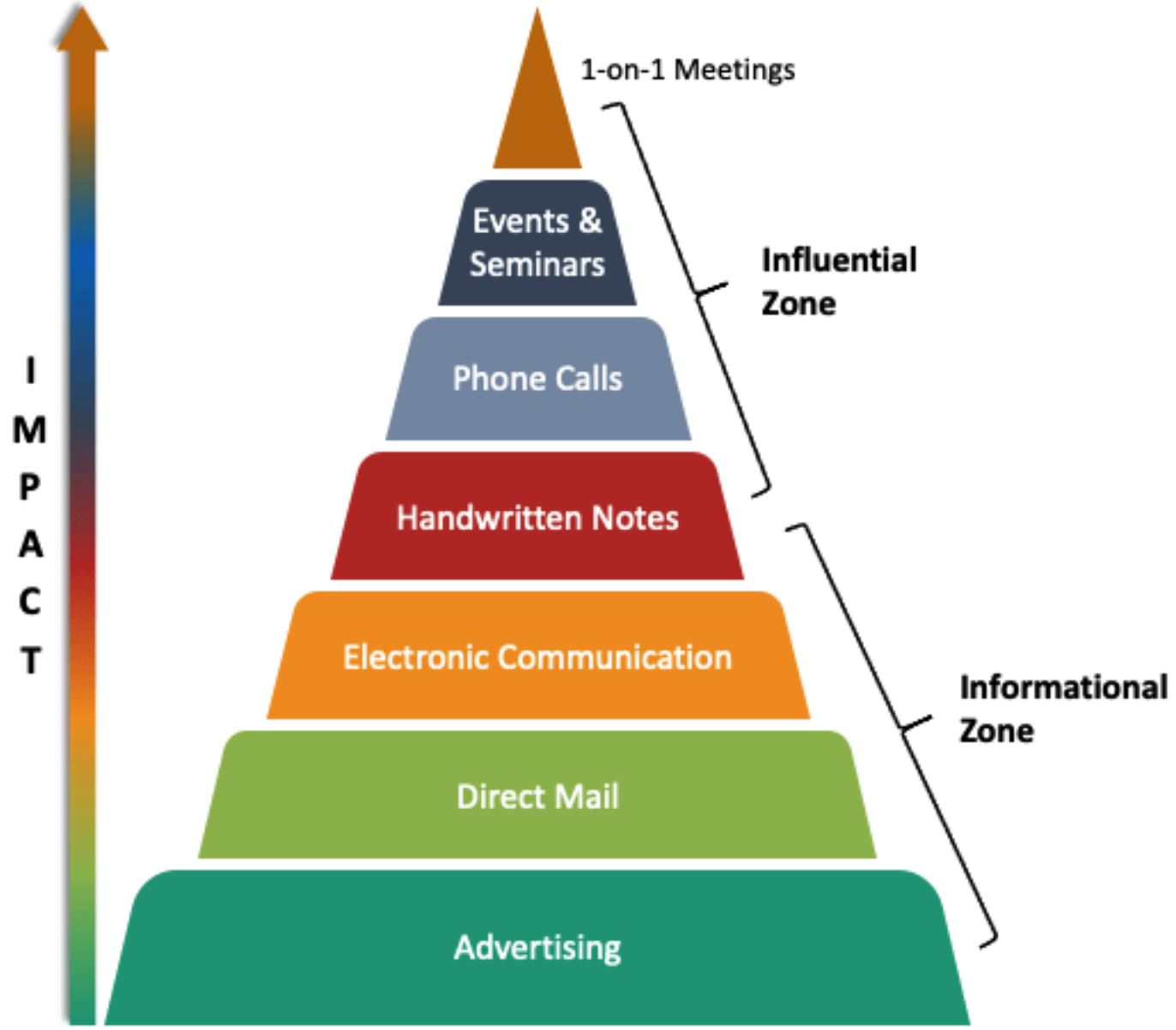


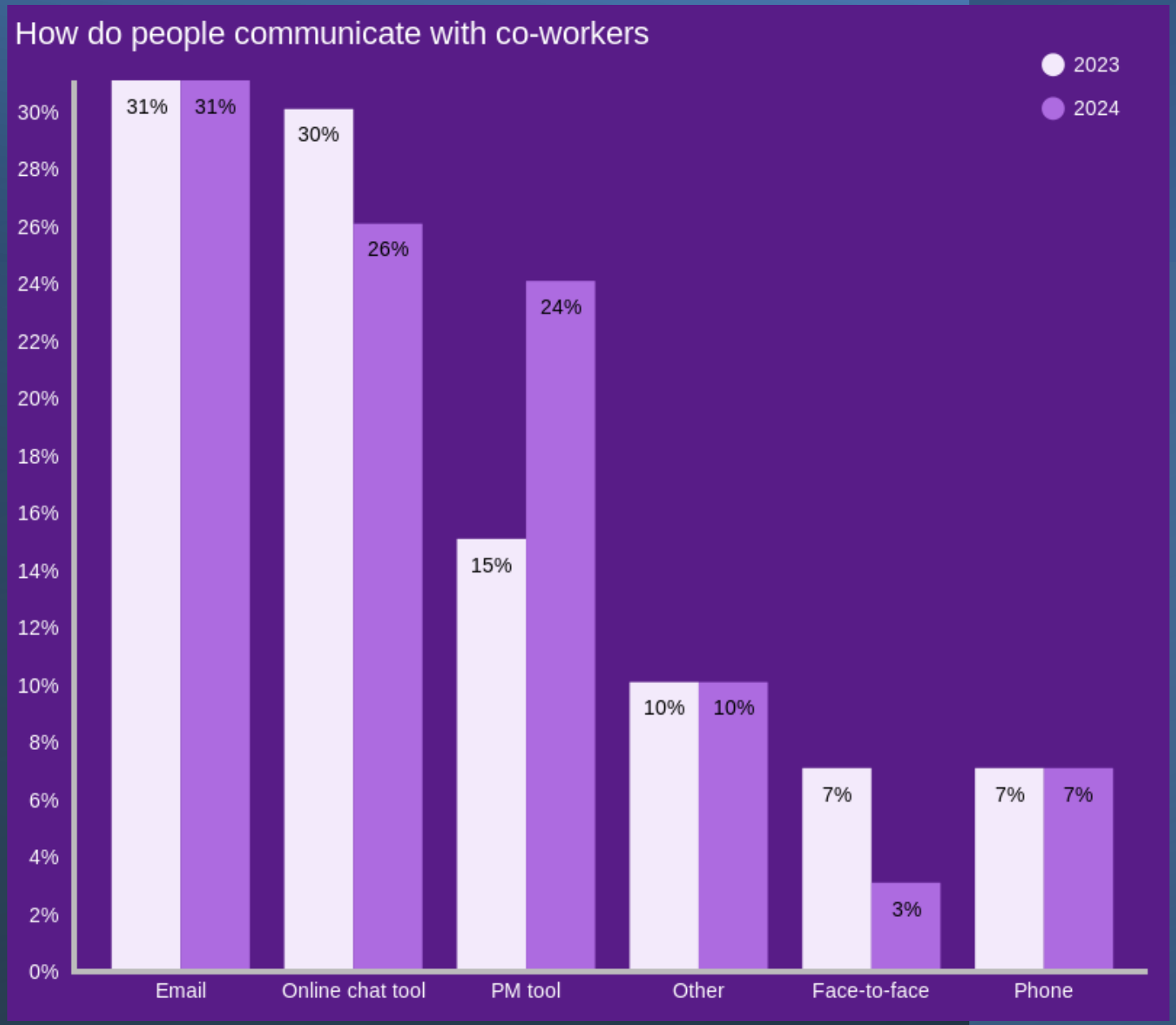
Lower risks, efficient and dynamic

Agile works good for the development team

COMMUNICATION PYRAMID

Enter your sub headline here

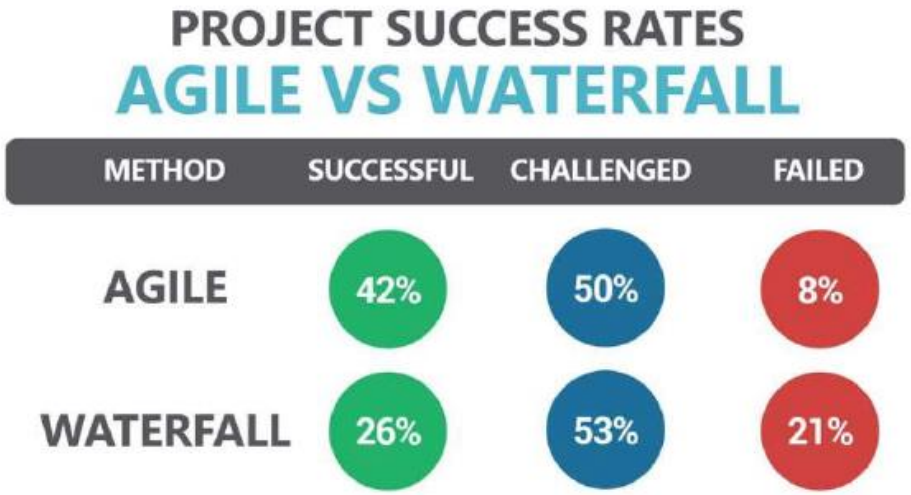



How do people communicate with co-workers
2023
2024
30%
28%
26%
24%
22%
20%
18%
16%
14%
12%
10%
8%
6%
4%
2%
0%
31%
31%
30%
26%
15%
24%
10%
10%
7%
3%
7%
7%
Email
Online chat tool
PM tool
Other
Face-to-face
Phone

Comparing Agile and Waterfall

The CHAOS Manifesto (2015)

The chaos manifesto 2017



Not a Silver bullet

External Links

- https://exoft.net/software-development-methodologies-pros-cons/
- https://www.ijert.org/research/a-comparison-between-two-software-engineering-processes-rup-and-waterfall-models-IJERTV2IS70559.pdf
- https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban?srsltid=AfmBOorvz6fY2-Z_tFhj7PduXyFI0udUeyo6qzgRxuBJ7nasyV7wYjSW
- https://www.atlassian.com/agile/project-management/waterfall-methodology