Lesson 1
Software methodologies

Rene de Jong
rene@ictintelligence.nl

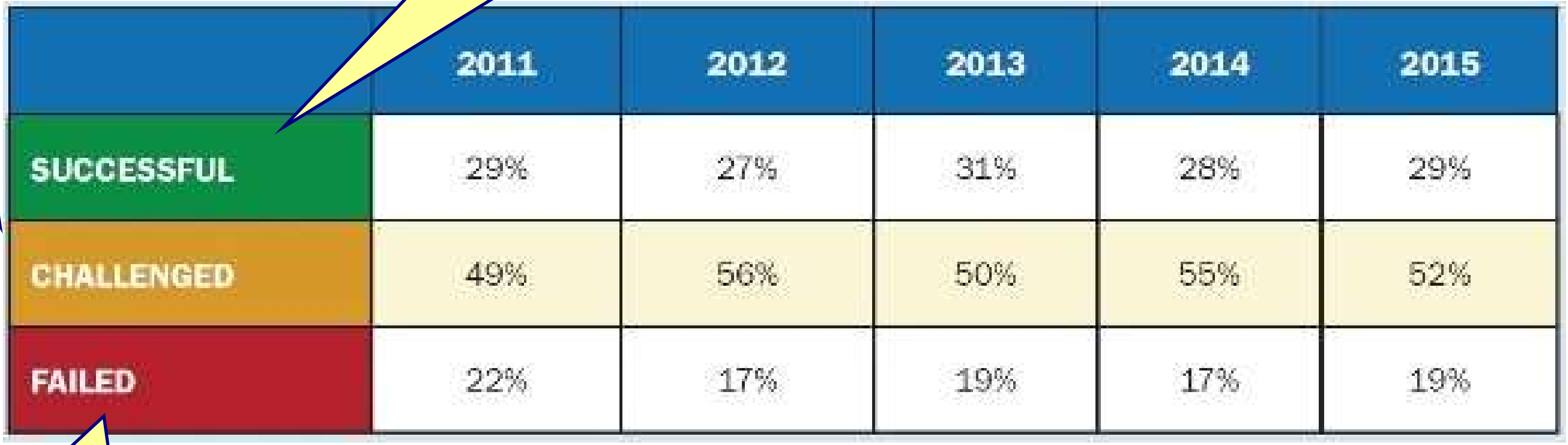
Software methodology

- Who
  - Roles of the people in the project
- What
  - What artefacts are used or created
- When
- How
  - Disciplines, activities, best practices

Success in software development

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

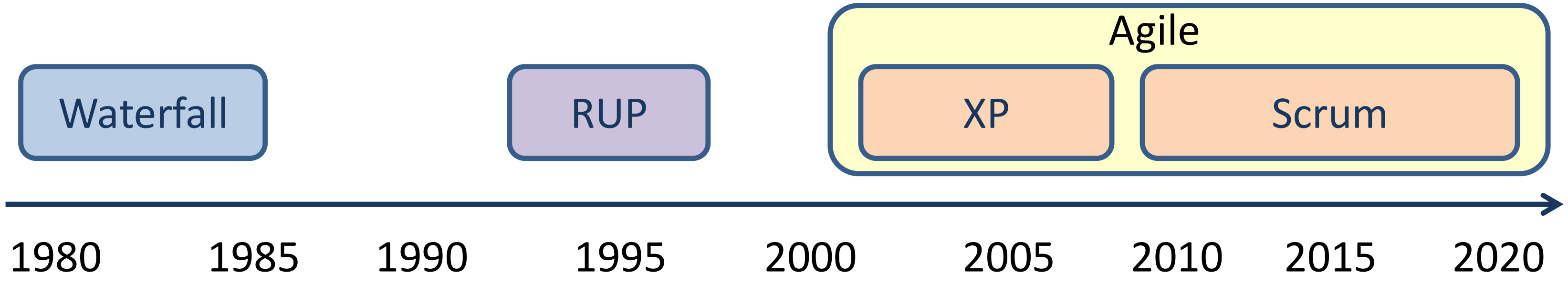
On-time, on-budget, and with all features and functions as defined in the initial scope



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

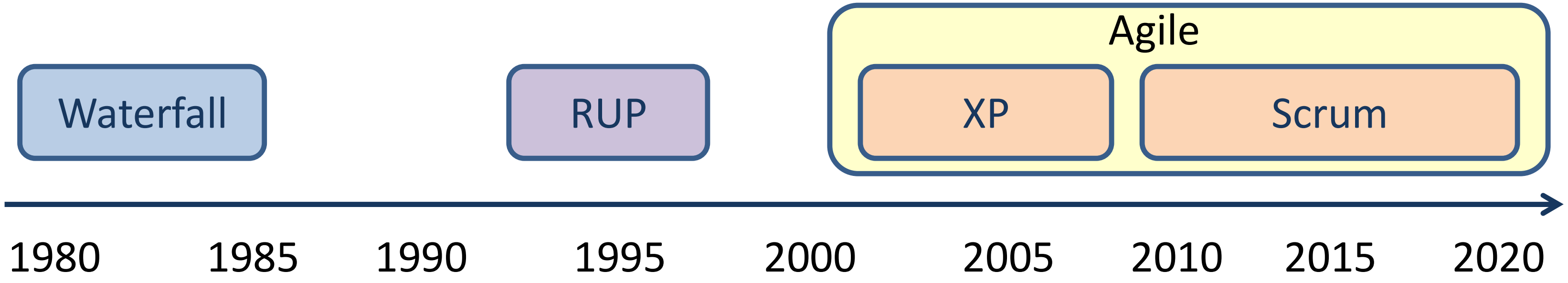
Cancelled prior to completion, or delivered but never used

Software development methods



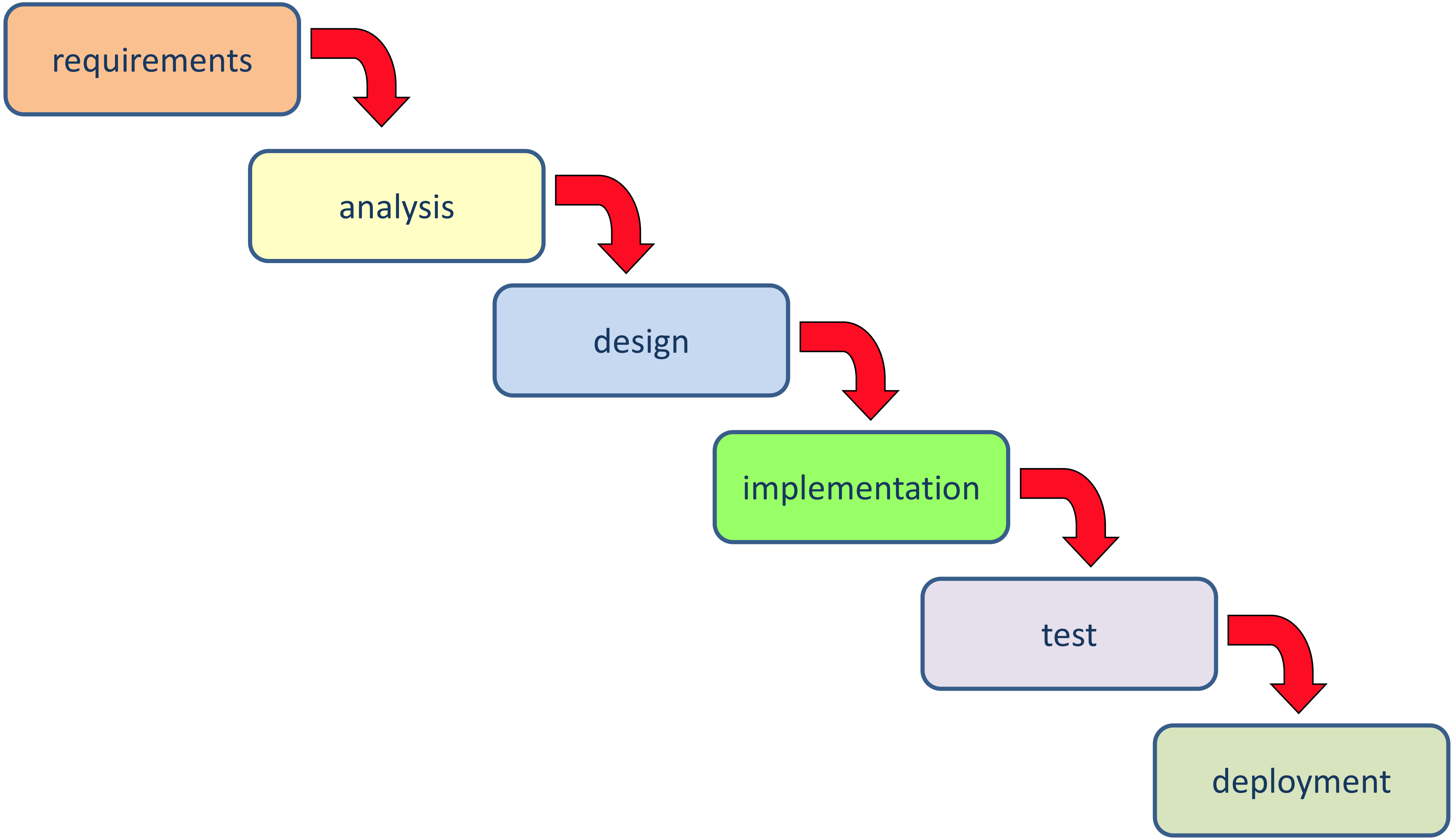
WATERFALL

Software development methods



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

Waterfall


requirements
analysis
design
implementation
test
deployment

Core Roles

- Project manager
- Analyst
- Developer
- Tester
- Architect

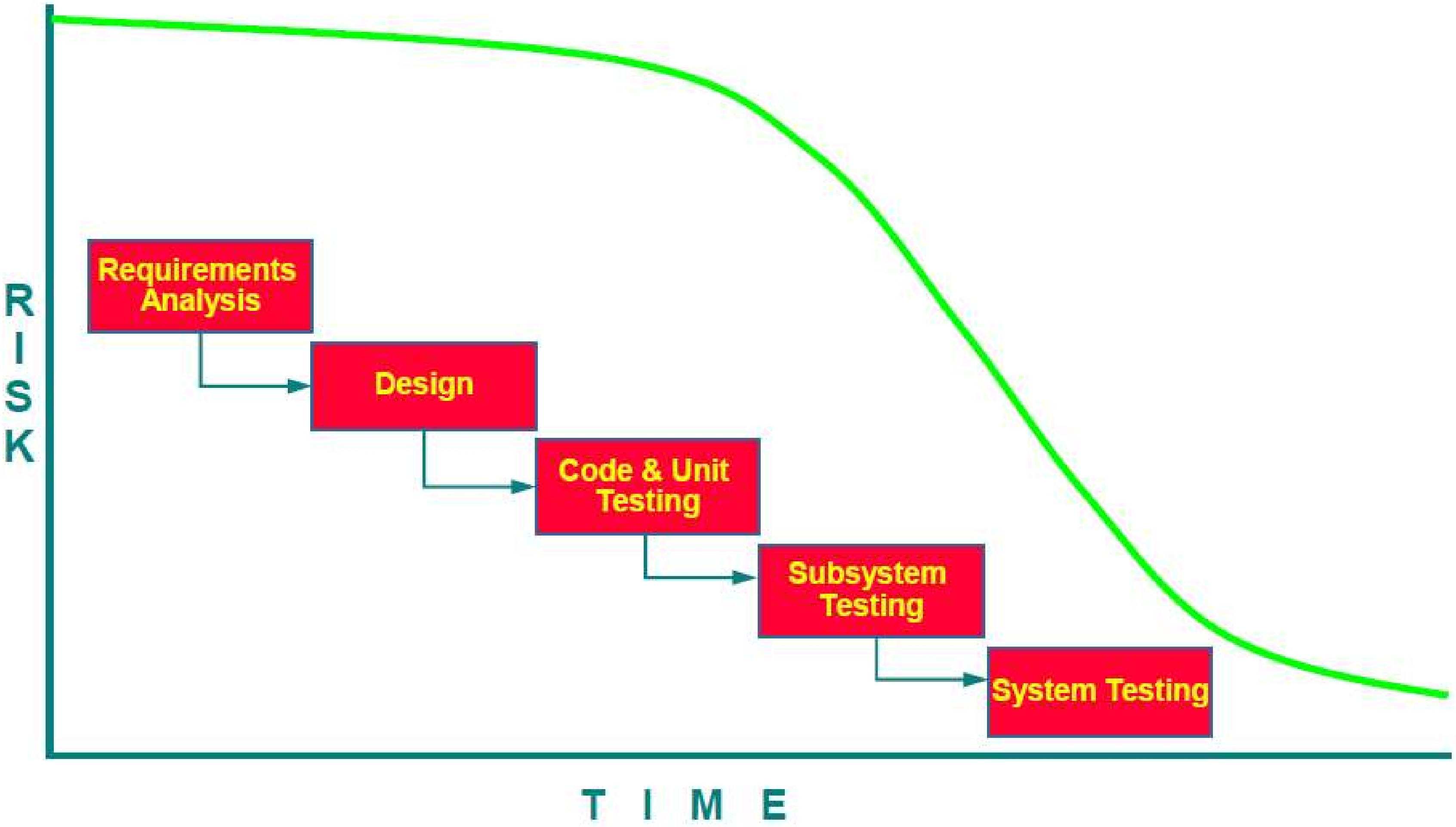
Project Manager

- 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 artifacts

- Requirements doc.
- Architecture doc.
- Design doc
- Code
- Test doc
- Planning doc

Risk


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

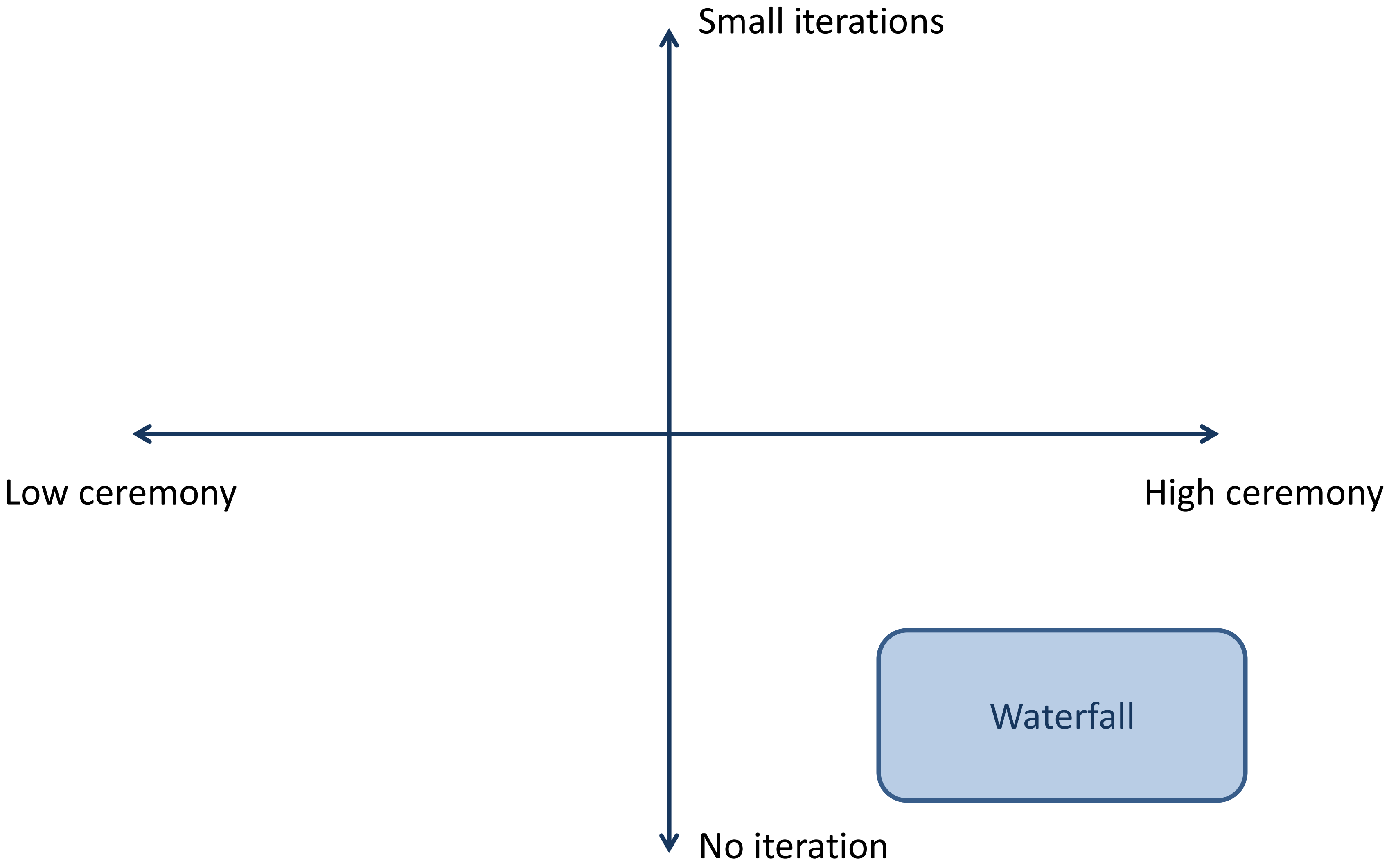
Characteristics of waterfall

- 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

High risks, inefficient and static

Waterfall works good for the project manager, but not for the development team

Software development methods



Main point

Software engineering

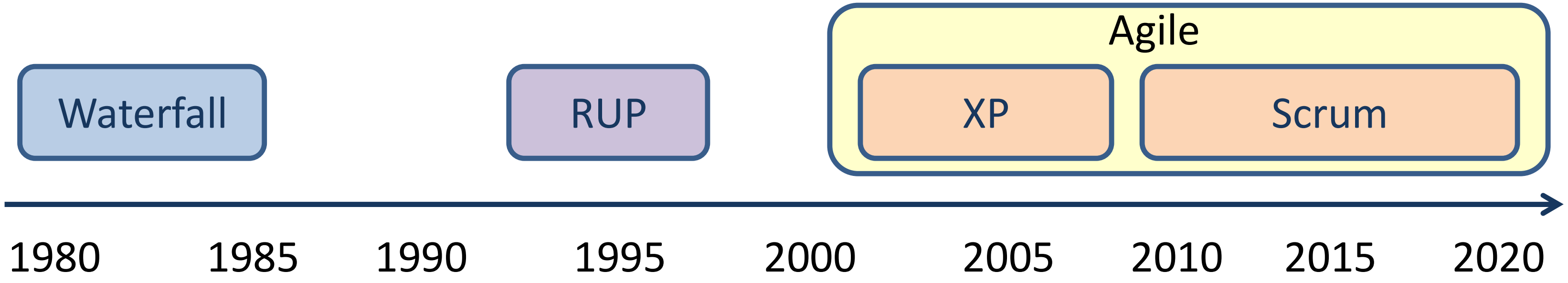
The waterfall way of software development is a very inefficient way to develop software.

SCI

Daily use of TM increases effectivity in ones live by increasing creativity, happiness and intelligence.

RATIONAL UNIFIED PROCESS (RUP)

Software development methods



Agile

Waterfall

RUP

XP

Scrum

1980 1985 1990 1995 2000 2005 2010 2015 2020

Linear

Iterations

Different roles

Document driven

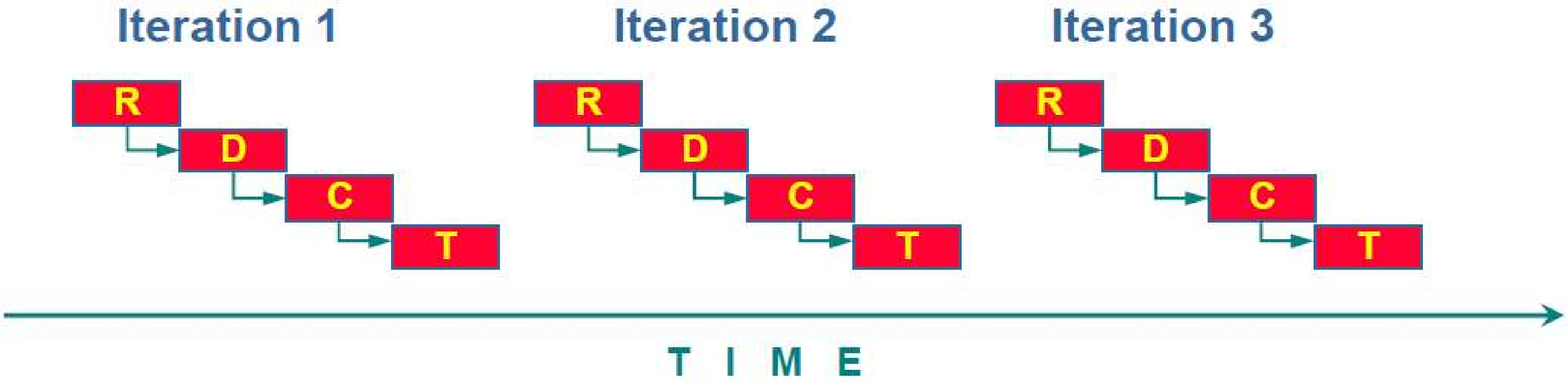
Customer is outside the project

Large projects(time, nr. of people)

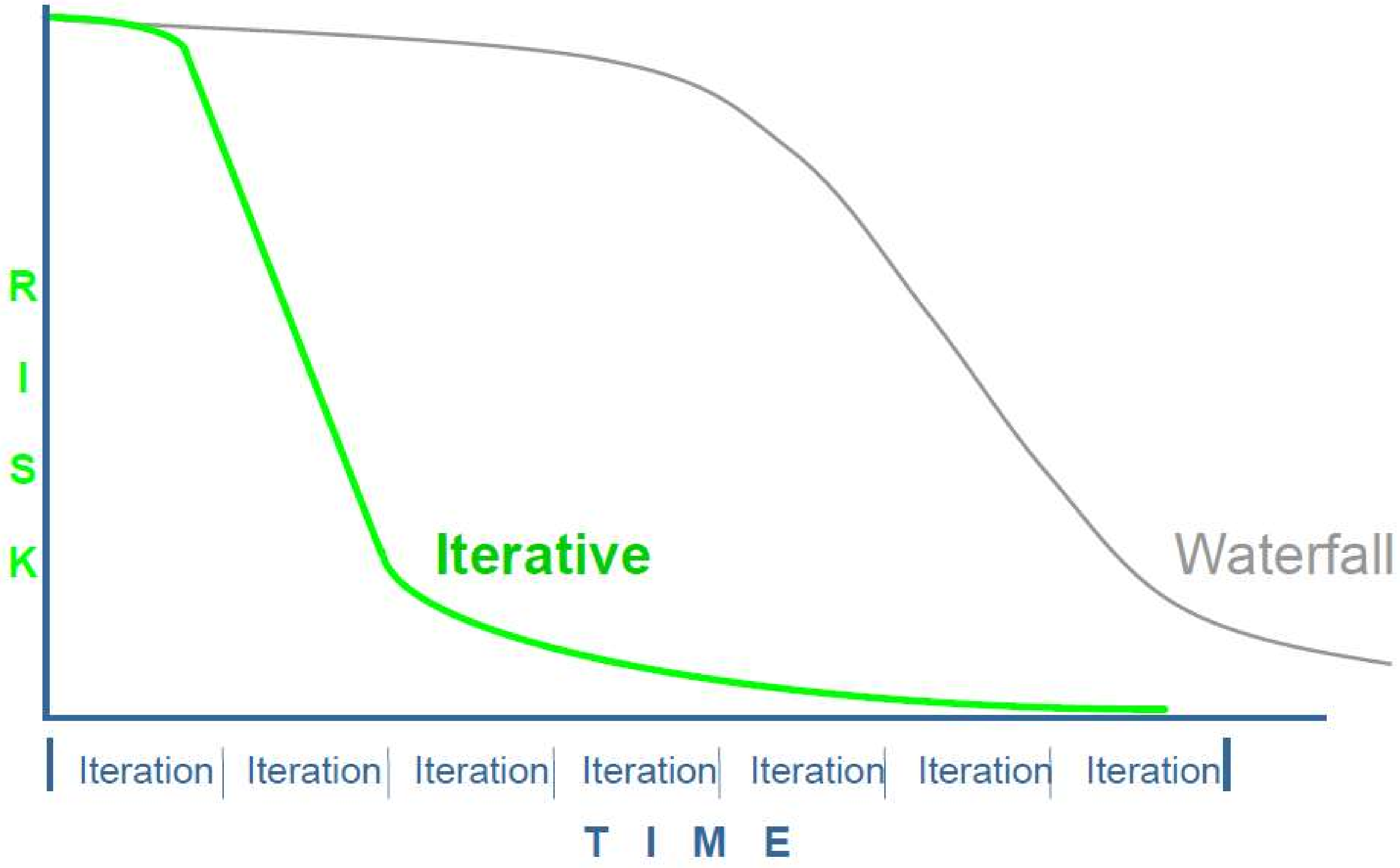
Req. statements

Use cases

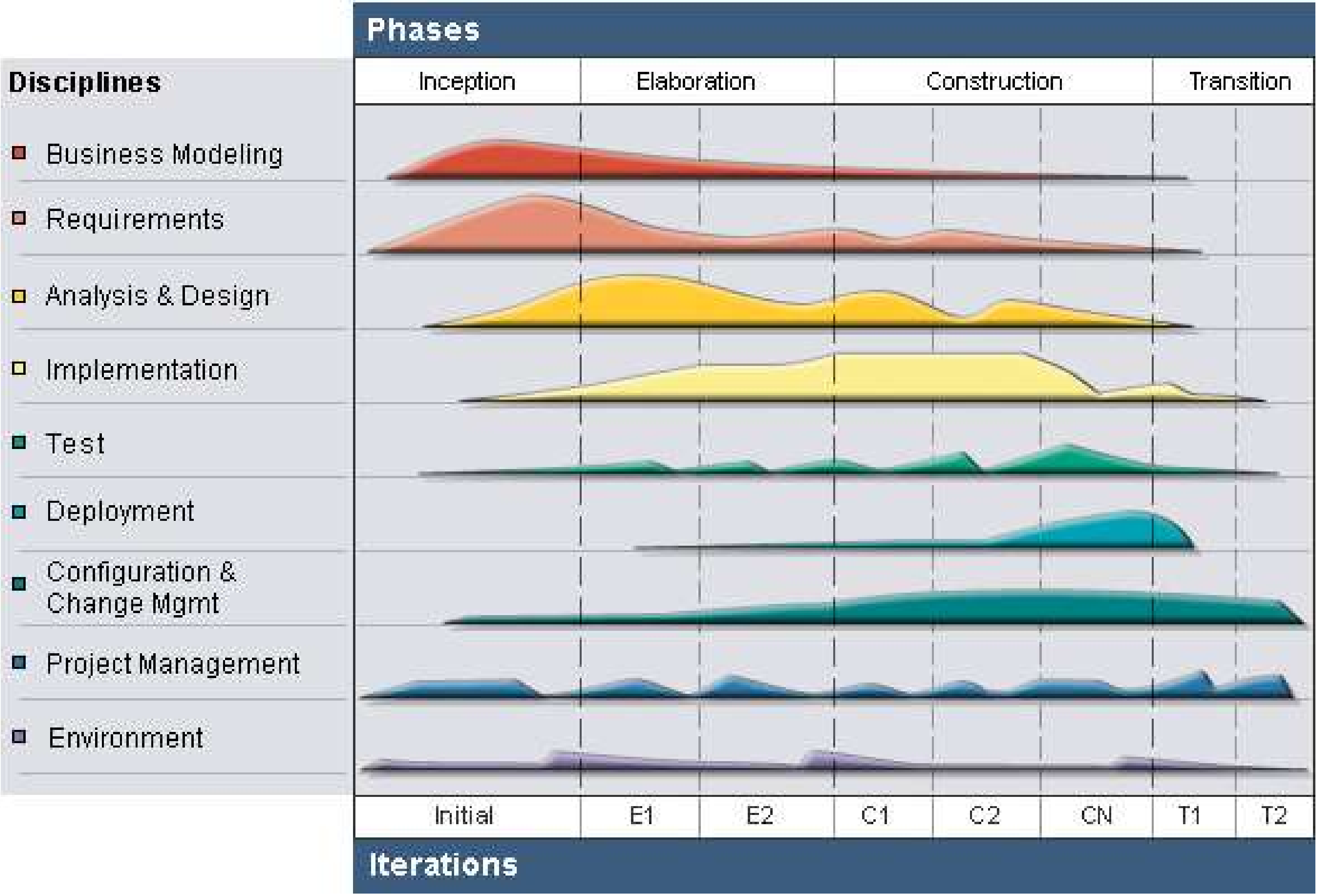
Iterations



Risk

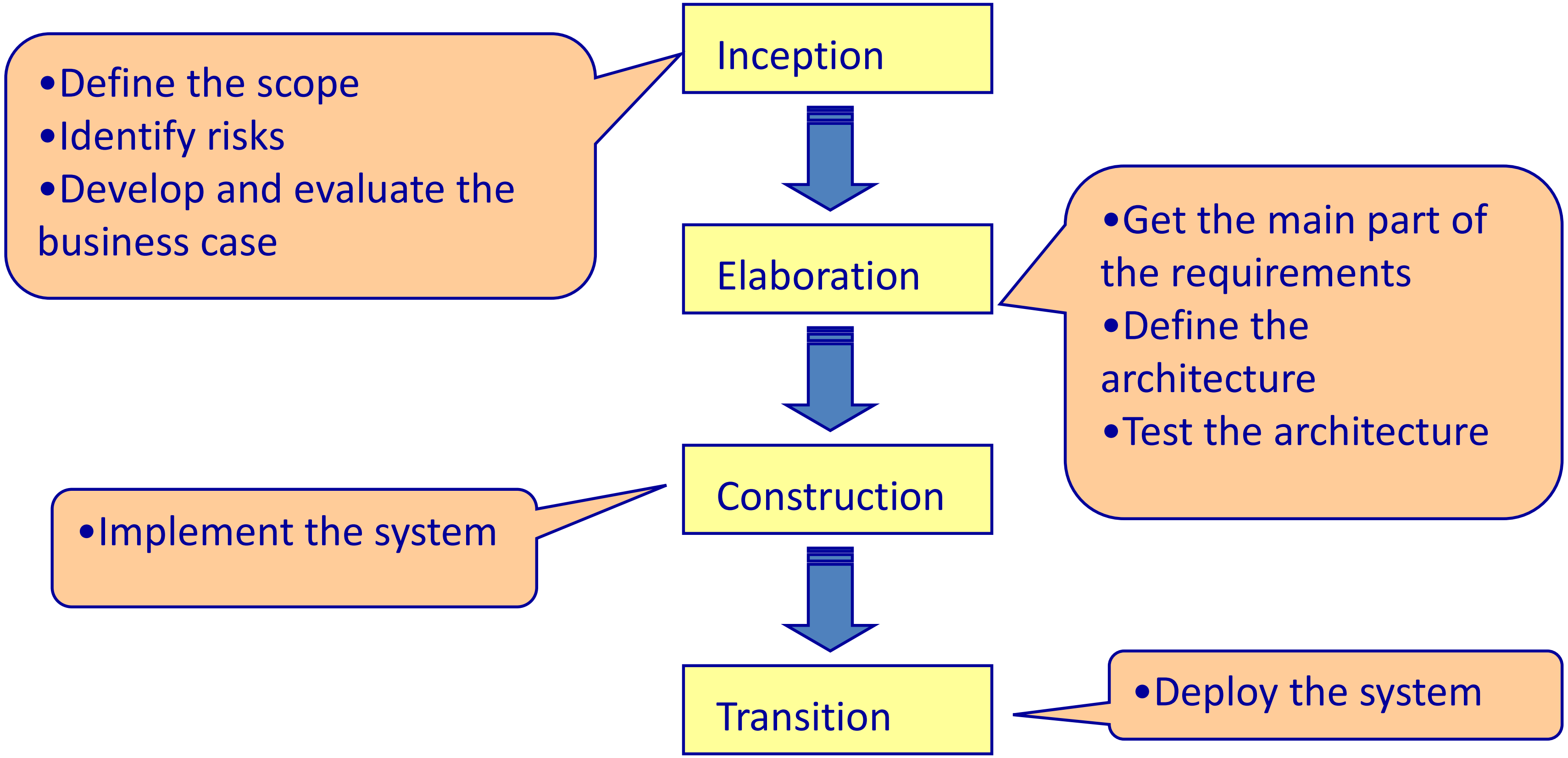

R I S K
Iterative
Waterfall
Iteration
Iteration
Iteration
Iteration
Iteration
Iteration
Iteration
T I M E

RUP



Phases
Inception
Elaboration
Construction
Transition
Disciplines
Business Modeling
Requirements
Analysis & Design
Implementation
Test
Deployment
Configuration & Change Mgmt
Project Management
Environment
Initial
E1
E2
C1
C2
CN
T1
T2
Iterations

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

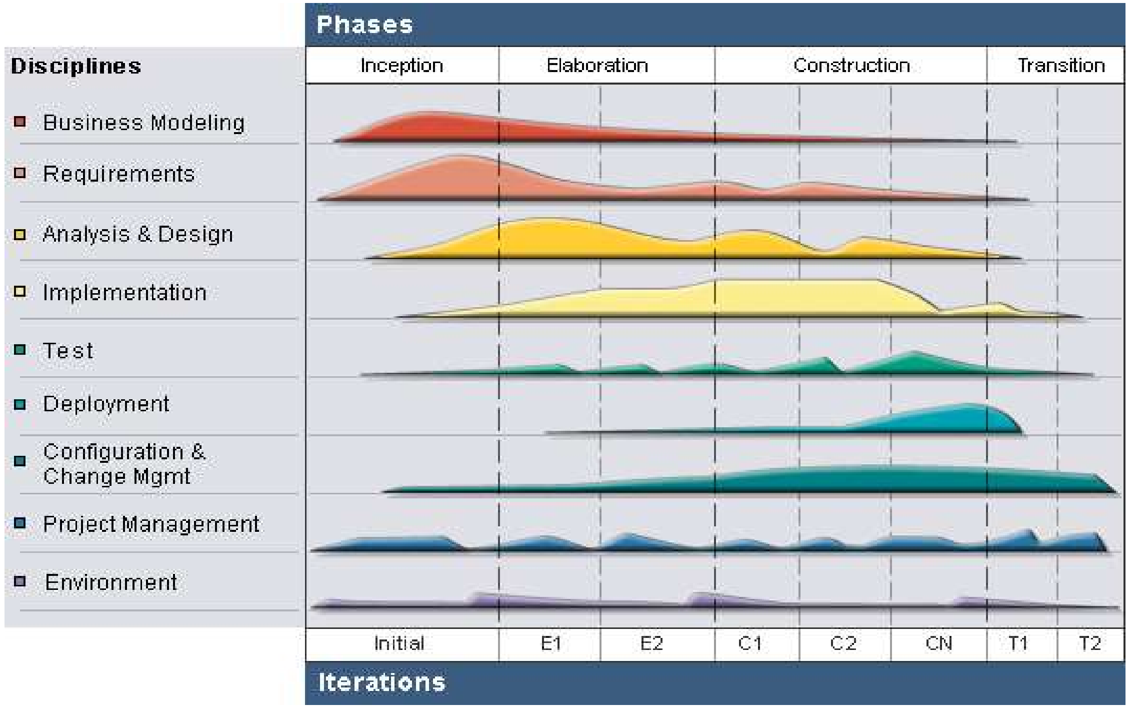
- Implement the system

Transition

- Deploy the system

RUP best practices

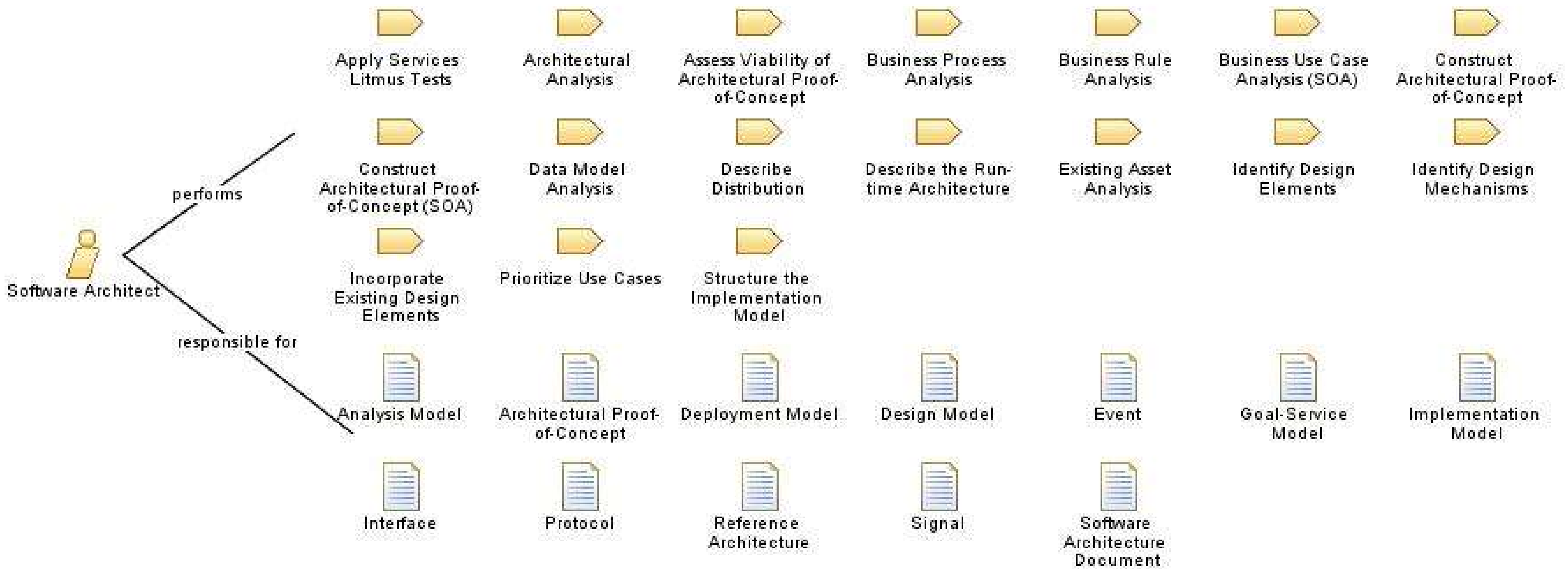
- Iterations
- Use case driven
- Visual modeling: UML
- Architecture centric
- Test everything
- Manage changes



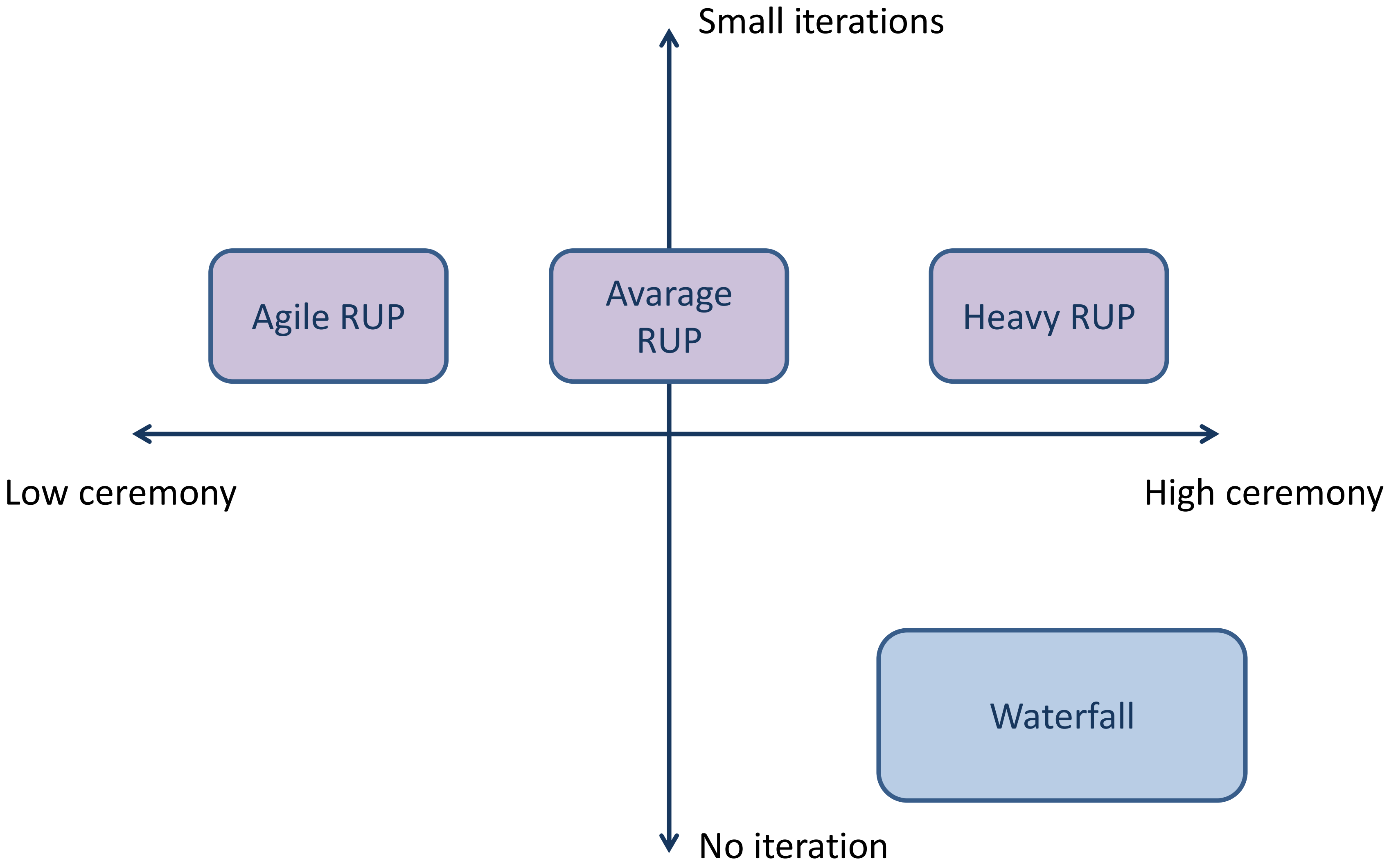
Roles

- 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

Roles, activities and artefacts


Software Architect
performs
responsible for
Apply Services Litmus Tests
Architectural Analysis
Assess Viability of Architectural Proof-of-Concept
Business Process Analysis
Business Rule Analysis
Business Use Case Analysis (SOA)
Construct Architectural Proof-of-Concept
Construct Architectural Proof-of-Concept (SOA)
Data Model Analysis
Describe Distribution
Describe the Run-time Architecture
Existing Asset Analysis
Identify Design Elements
Identify Design Mechanisms
Incorporate Existing Design Elements
Prioritize Use Cases
Structure the Implementation Model
Analysis Model
Architectural Proof-of-Concept
Deployment Model
Design Model
Event
Goal-Service Model
Implementation Model
Interface
Protocol
Reference Architecture
Signal
Software Architecture Document

Software development methods



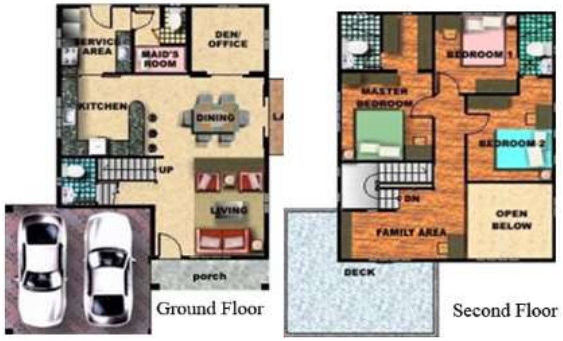
UML

What is UML?

- Unified Modeling Language
- A language for drawing models
- Originally developed for modeling object-oriented software systems
- Contains 13 different diagrams

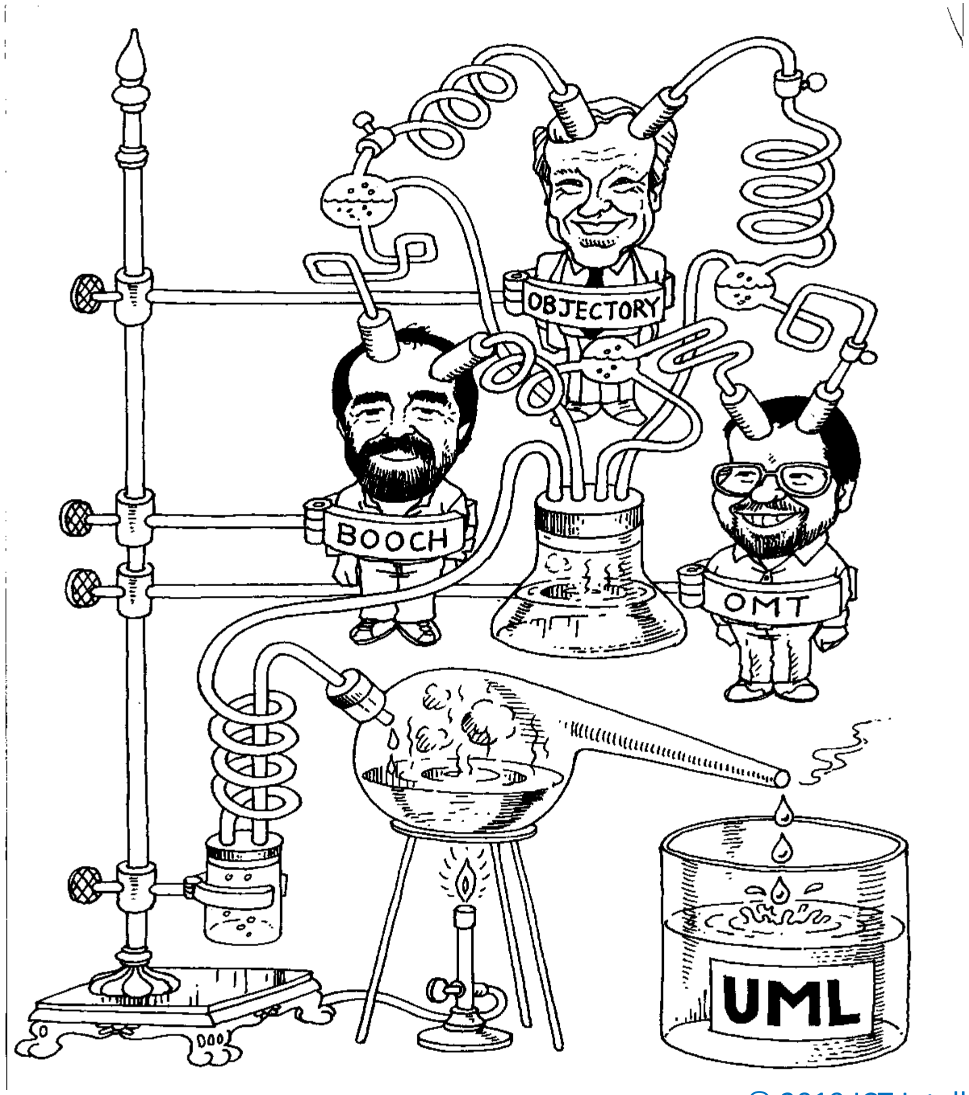
- UML is not a development method by itself

Model



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

UML

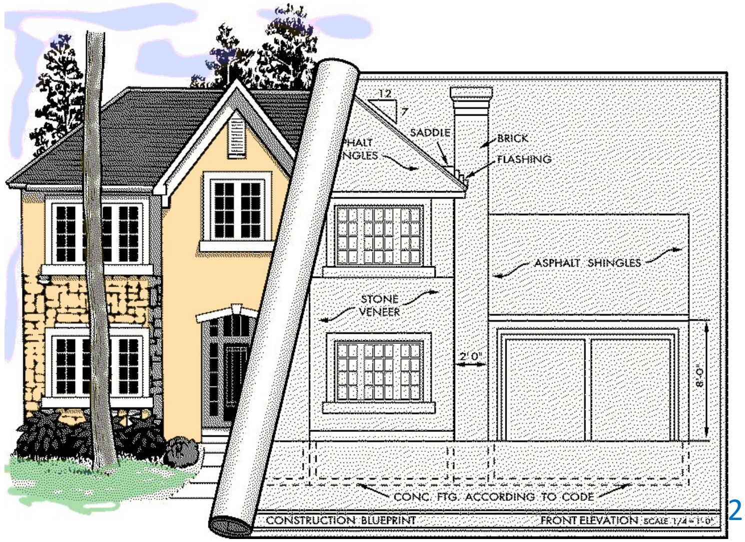


- Rational
  - Booch
  - Jacobsen
  - Rumbaugh

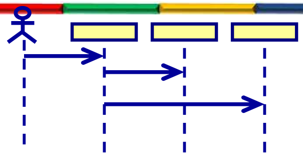
- Object Management Group (OMG)

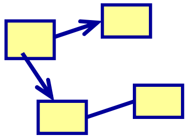
Why UML?

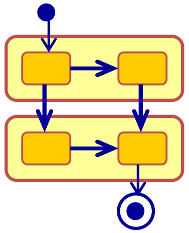
- UML is a standard
- UML supports
  - Abstraction
  - Decomposition
  - Zoom-in and zoom-out functionality
  - Different views on the same model
- Tool support

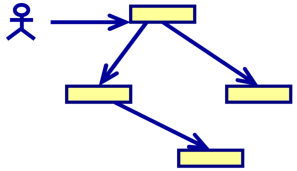


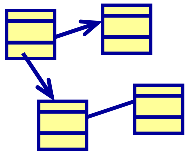
UML Diagrams

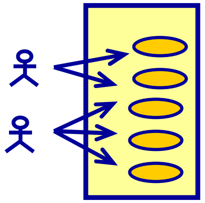

sequence diagram

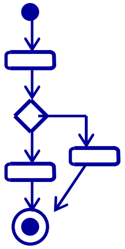

object diagram

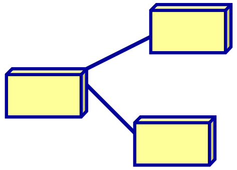

state machine
diagram

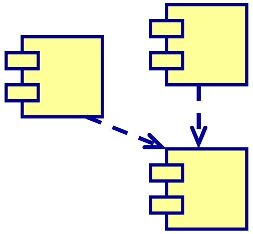

communication diagram

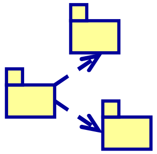

class diagram

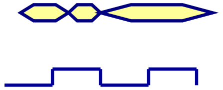

use case diagram

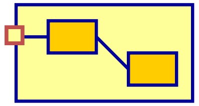

activity diagram

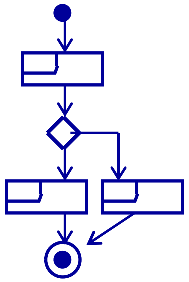

deployment
diagram


component diagram

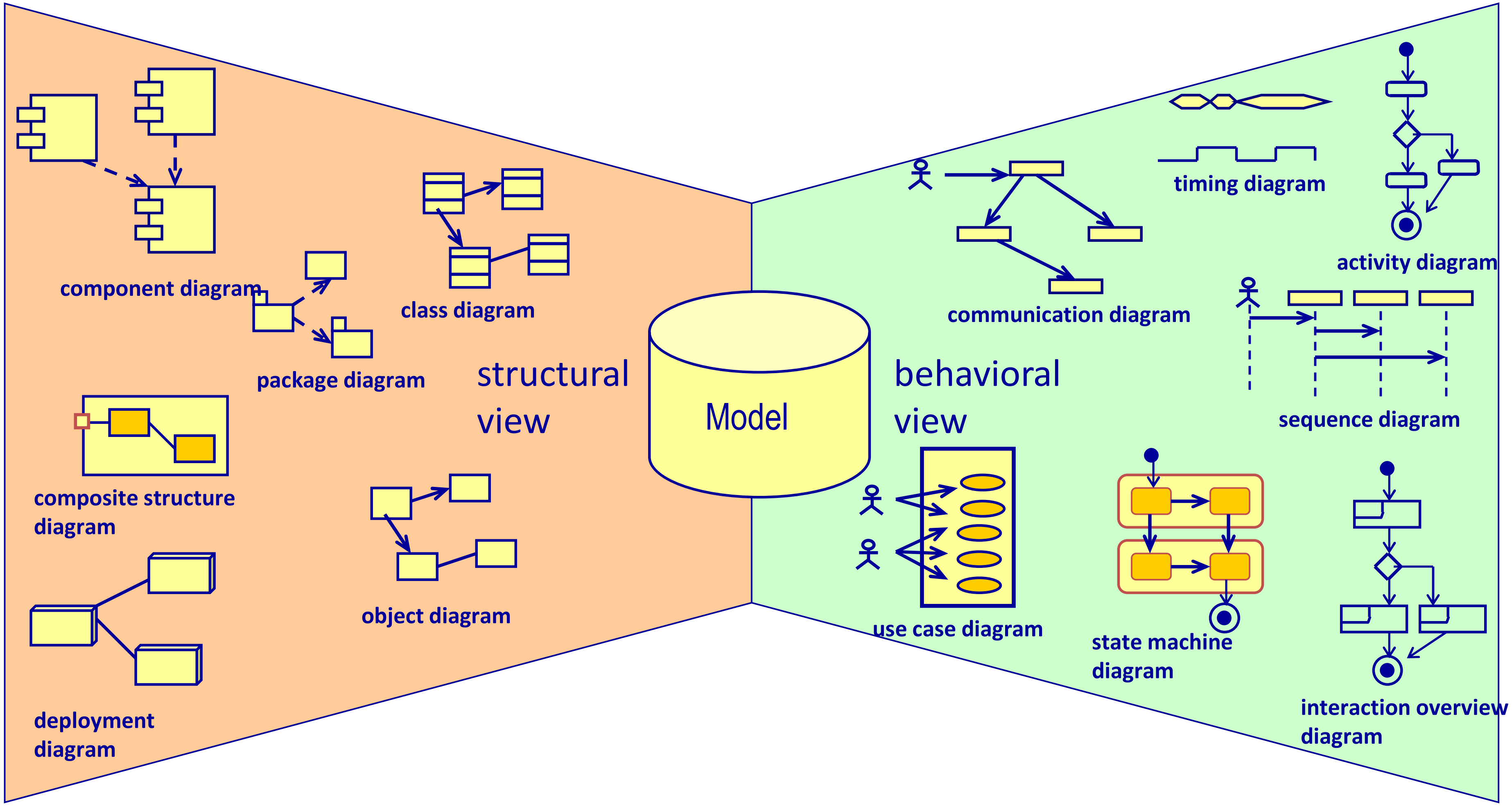

package diagram


timing diagram

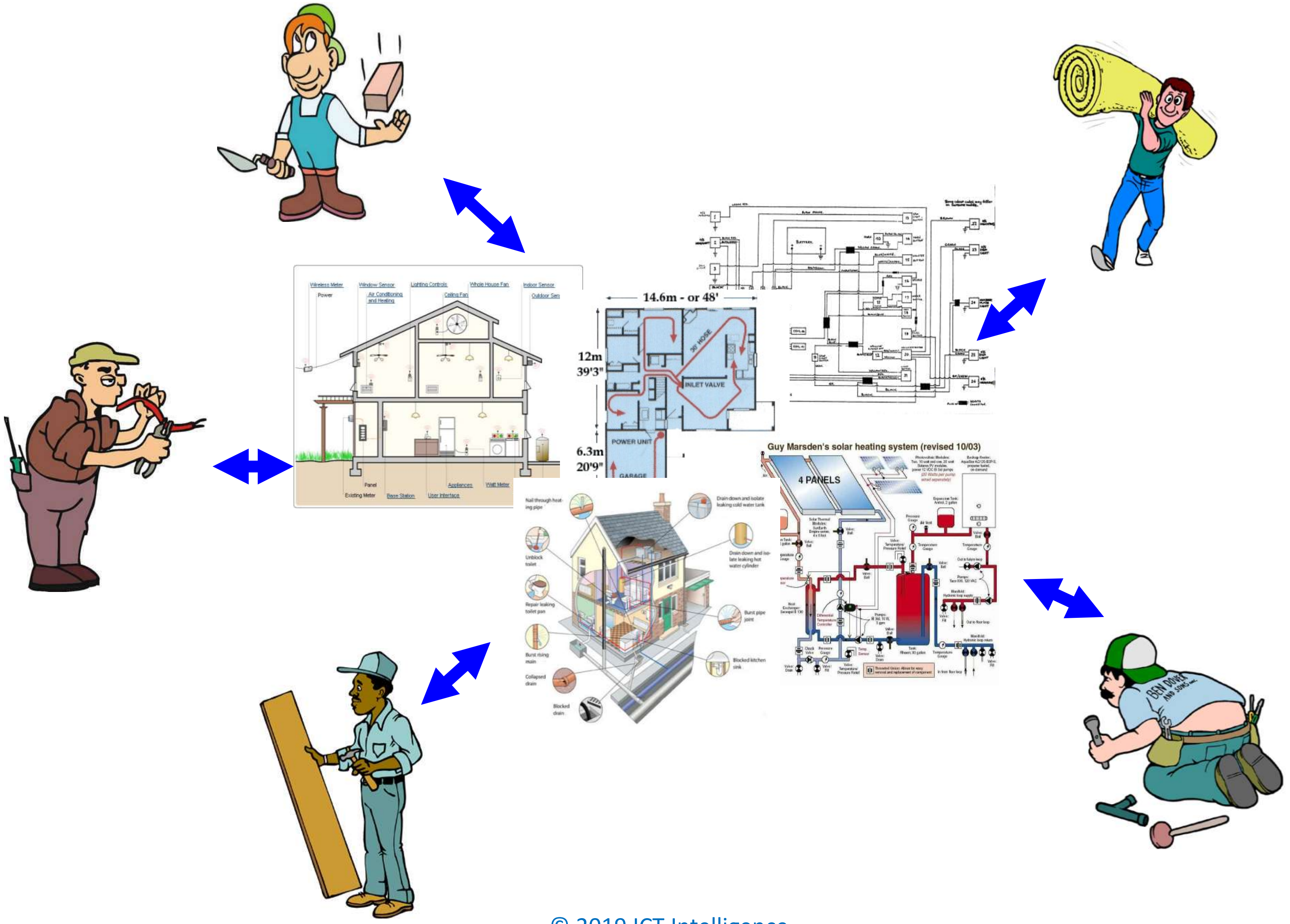

composite structure
diagram


interaction overview diagram

Diagrams and views

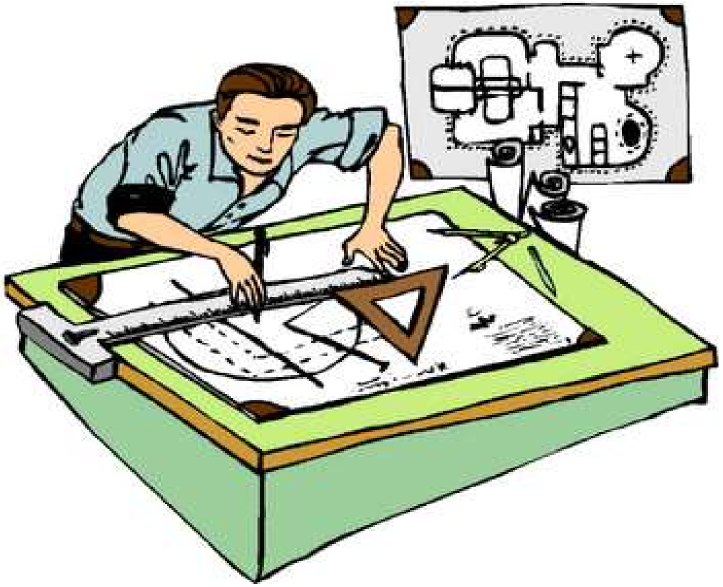


Model and views

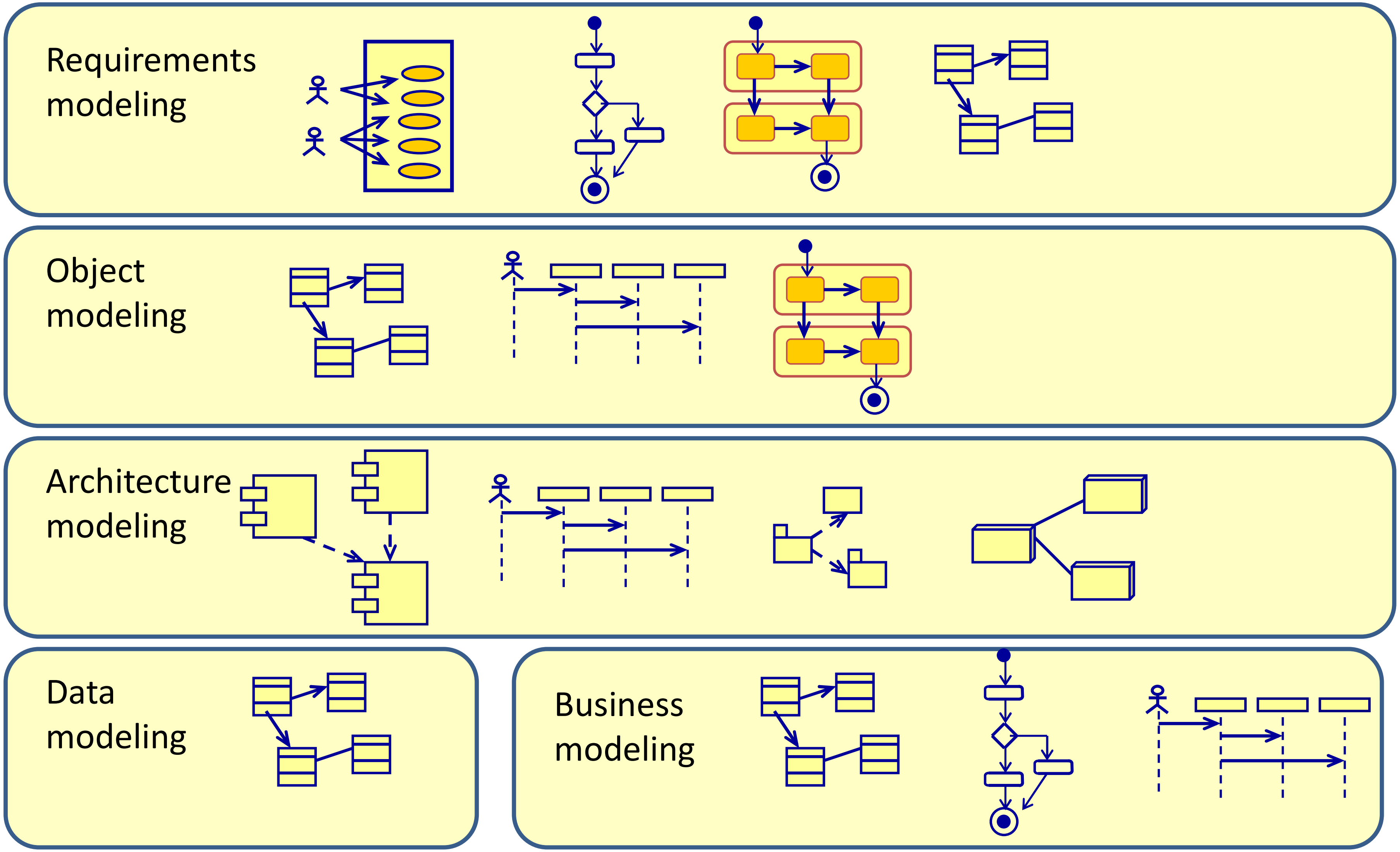


UML tools

- Rational System Modeler
- Enterprise Architect
- MagicDraw
- Visio
- StarUML
- ArgoUML
- Poseidon
- …



Applying UML



Requirements modeling

Object modeling

Architecture modeling

Data modeling

Business modeling

Main point

Software engineering

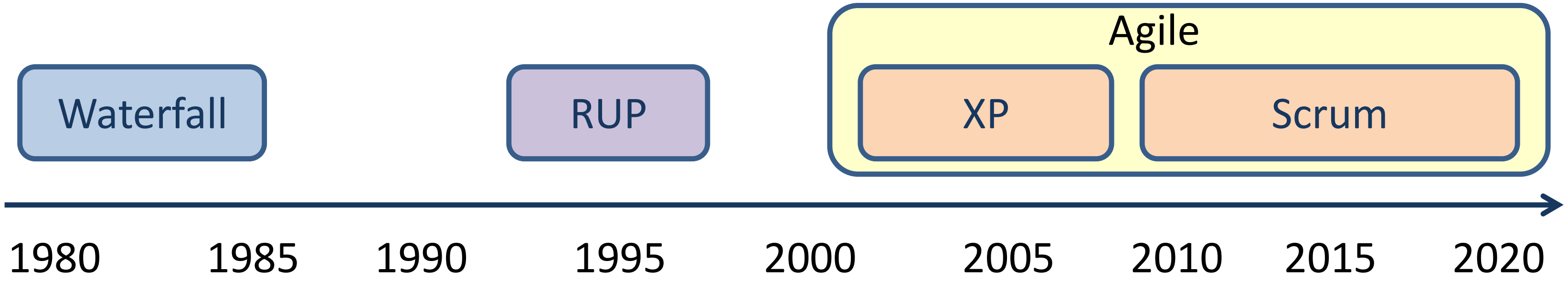
RUP introduces iterations, UML and use cases to software development, but still inherits bad practices from waterfall development

SCI

Contacting the source of all intelligence leads to spontaneous right actions

AGILE

Software development methods



Waterfall

RUP

Agile

XP

Scrum

1980 1985 1990 1995 2000 2005 2010 2015 2020

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 Manifest

Individuals and interactions	over	Processes and tools
Working software	over	Comprehensive documentation
Customer collaboration	over	Contract negotiation
Responding to change	over	Following a plan

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.**

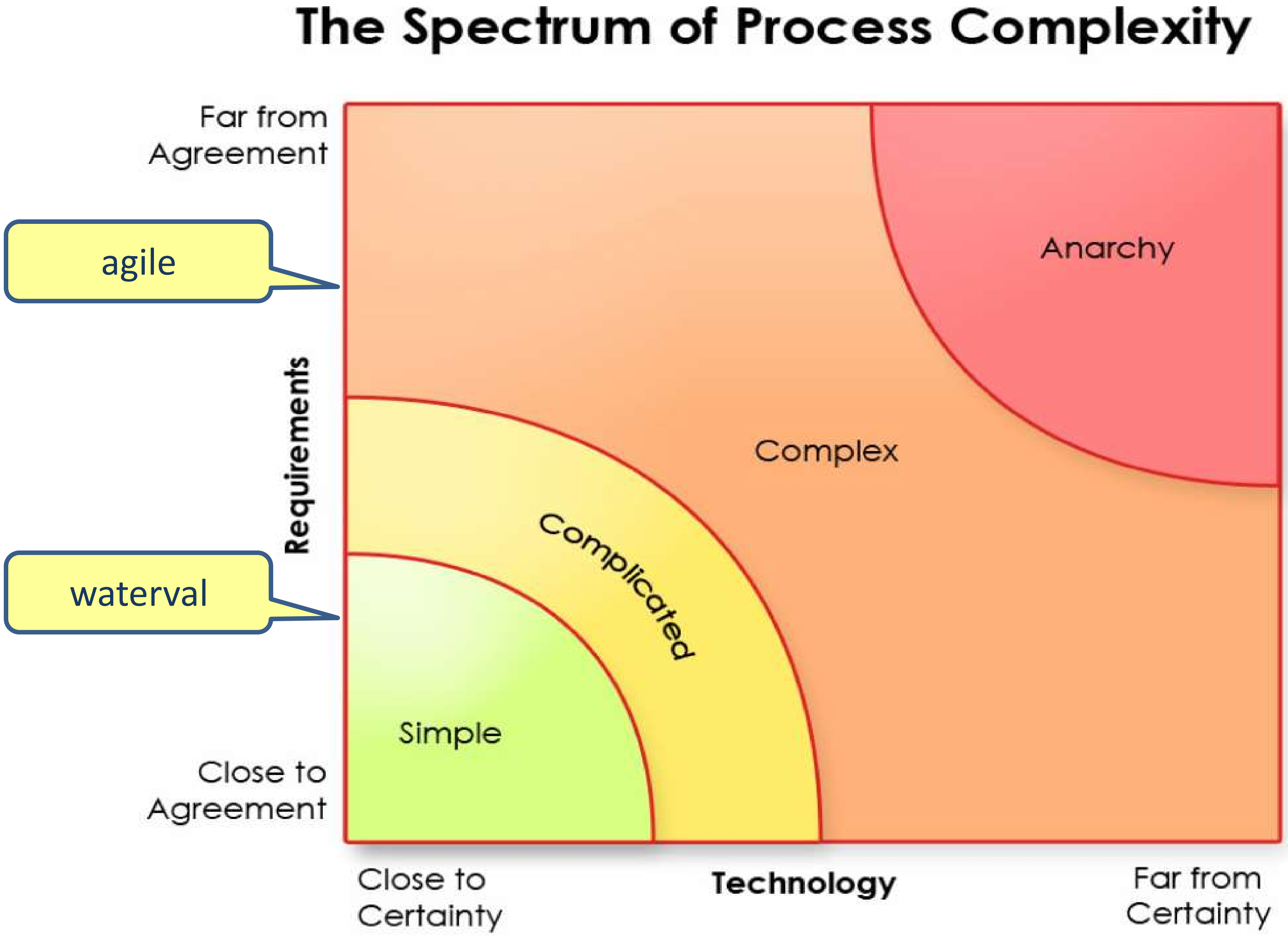
How is agile different?

Face-to-face communication	Document driven	The customer is involved only at the beginning of the project	Customer is involved during the whole project
Risks are discovered early in the project	Risks are found late in the project	Lot of different roles	Cross functional team working together
Requirements may change	Requirements are frozen	Software can be used only at the end of the project	Software can be used after a few weeks
Cross functional team working together	Throw artefacts over the wall	Not much possibilities for reflection and improvement	Reflection and improvement is buildin
Status is based on working software	Project status is not clear	No feedback	Immediate and constant feedback
Cross functional team working together	No possibilities to learn other disciplines	There is no time left for testing	Immediate and constant testing

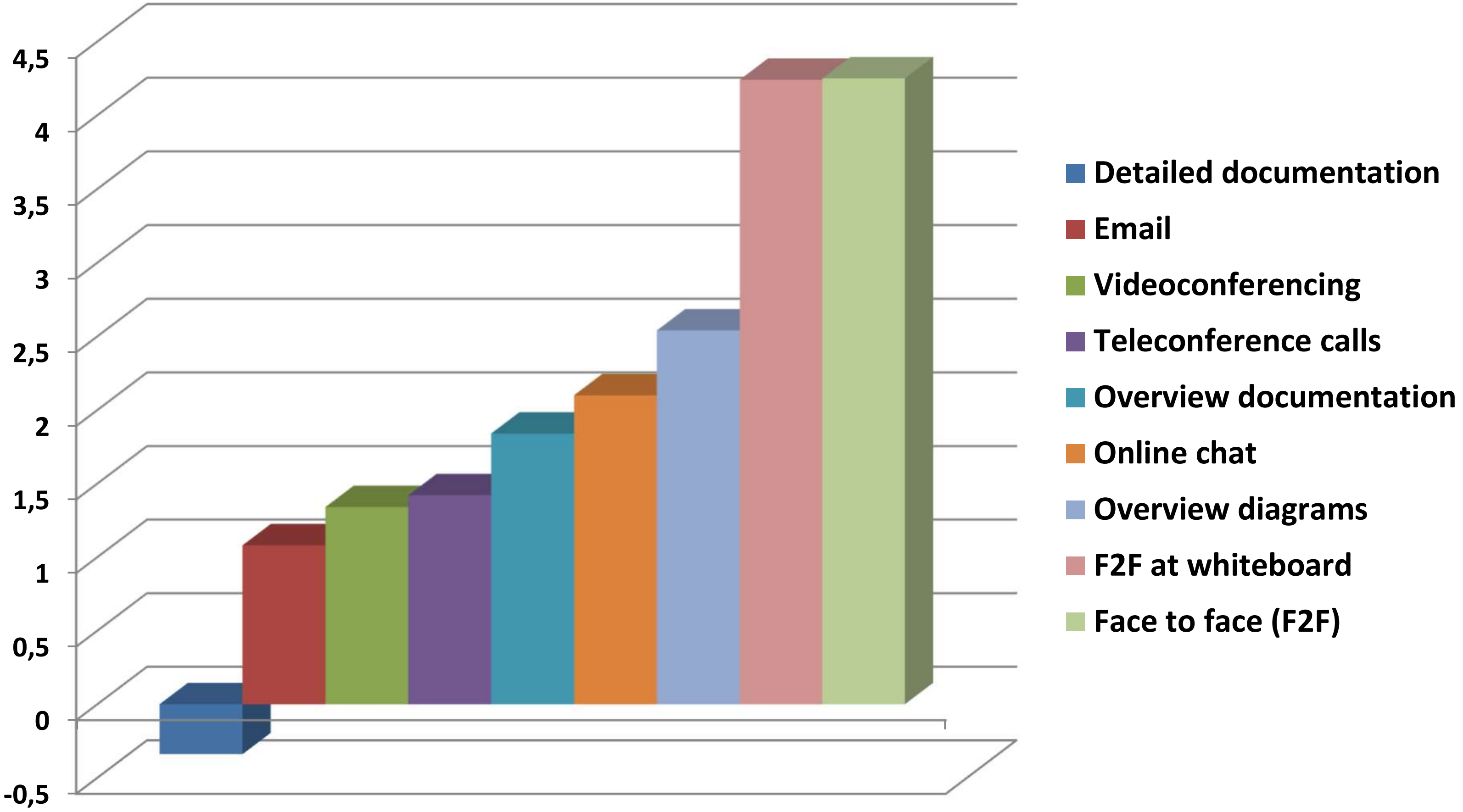
Lower risks, efficient and dynamic

Agile works good for the development team

When to use agile?



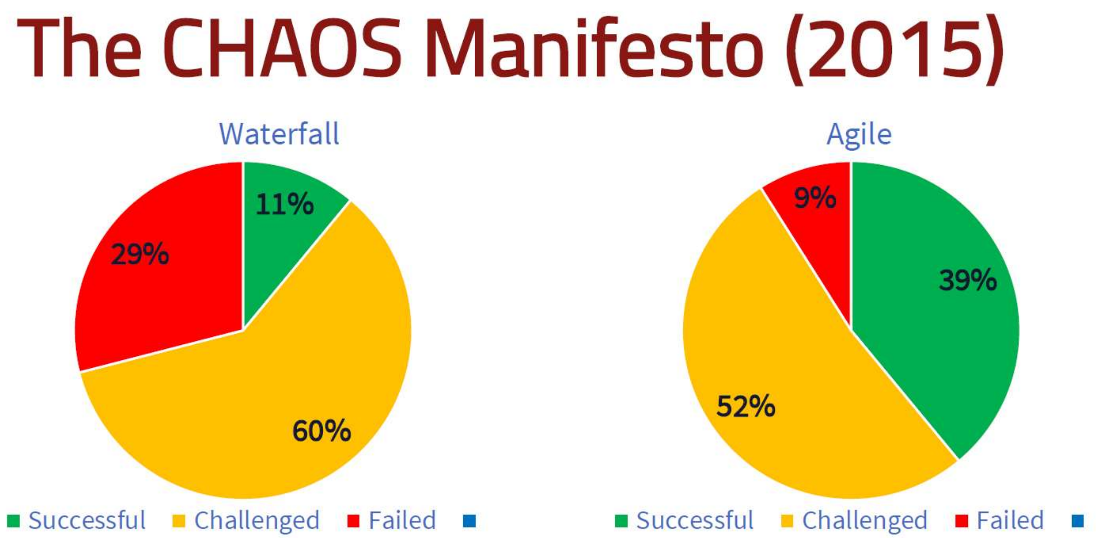
Effectiveness of communication



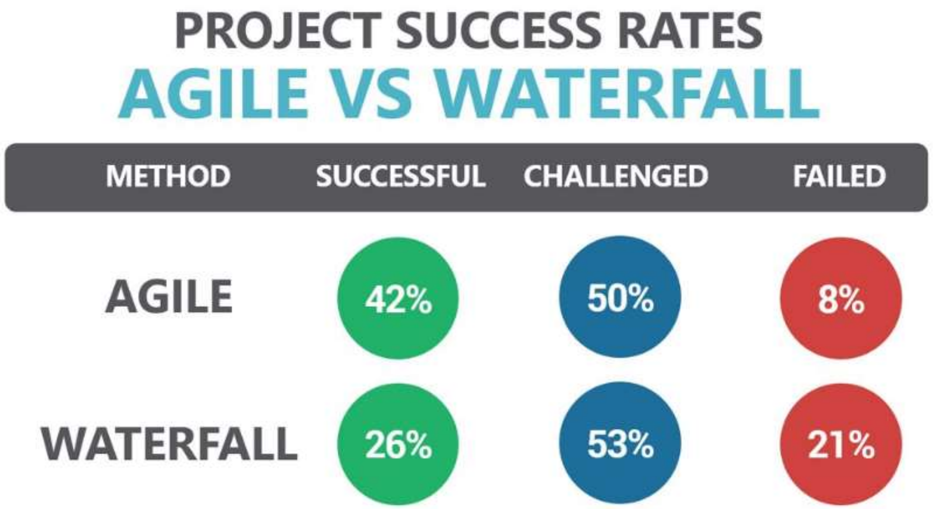
No solver bullet

SCRUM

Comparing waterfall and agile



The chaos manifesto 2017


PROJECT SUCCESS RATES

AGILE VS WATERFALL

METHOD	SUCCESSFUL	CHALLENGED	FAILED
AGILE	42%	50%	8%
WATERFALL	26%	53%	21%

SCRUM OVERVIEW

What is Scrum?

- A framework for project management
- Easy to learn
- Difficult to apply

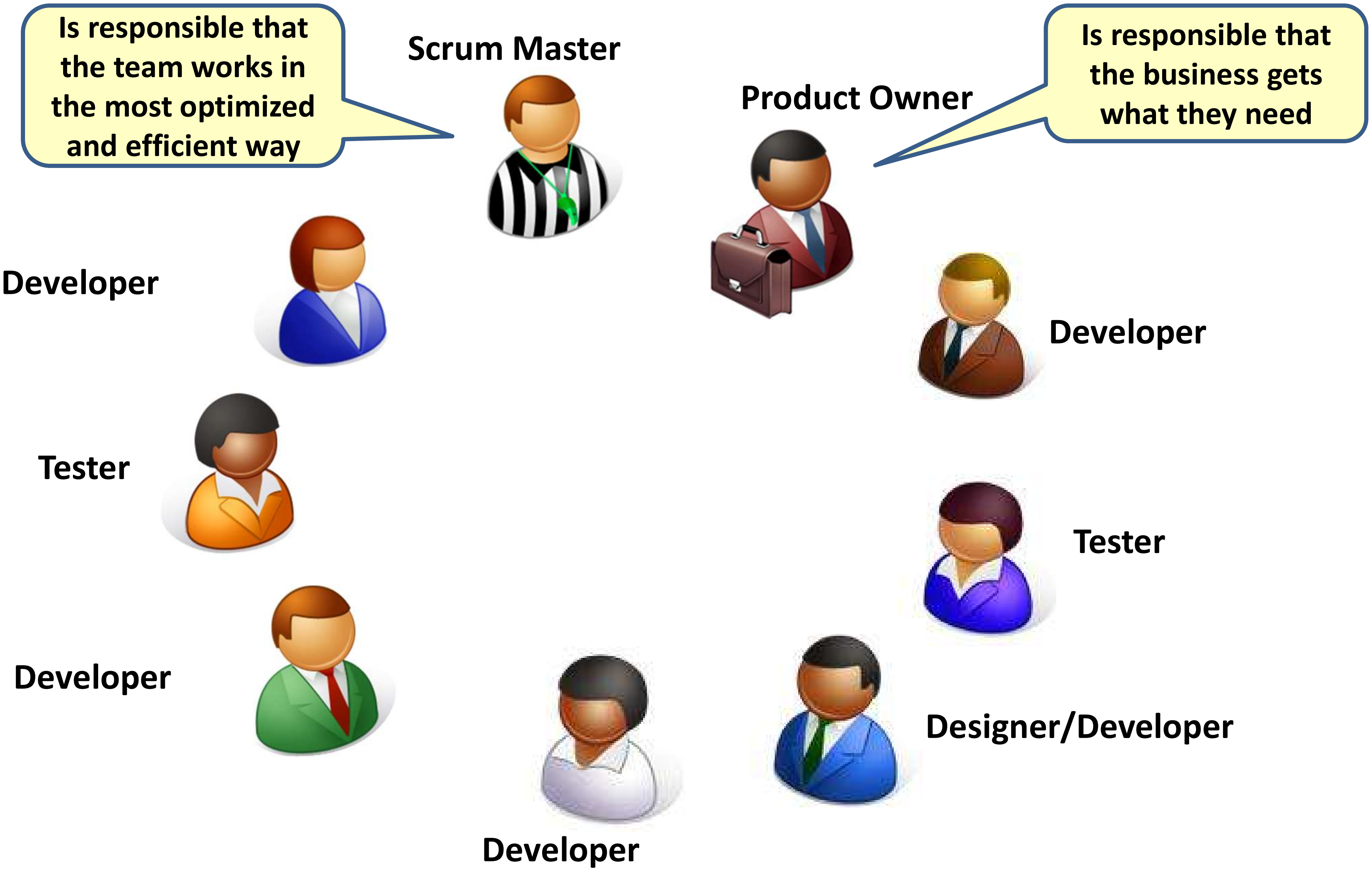
Scrum Framework

- Roles
  - Product owner
  - ScrumMaster
  - Team

- Ceremonies
  - Sprint planning
  - Sprint review
  - Sprint retrospective
  - Daily scrum meeting
  - Product backlog refinement

- Artifacts
  - Product backlog
  - Sprint backlog
  - Burndown charts
  - Definition of done

Scrum team



Is responsible that the team works in the most optimized and efficient way

Scrum Master

Product Owner

Is responsible that the business gets what they need

Developer

Developer

Tester

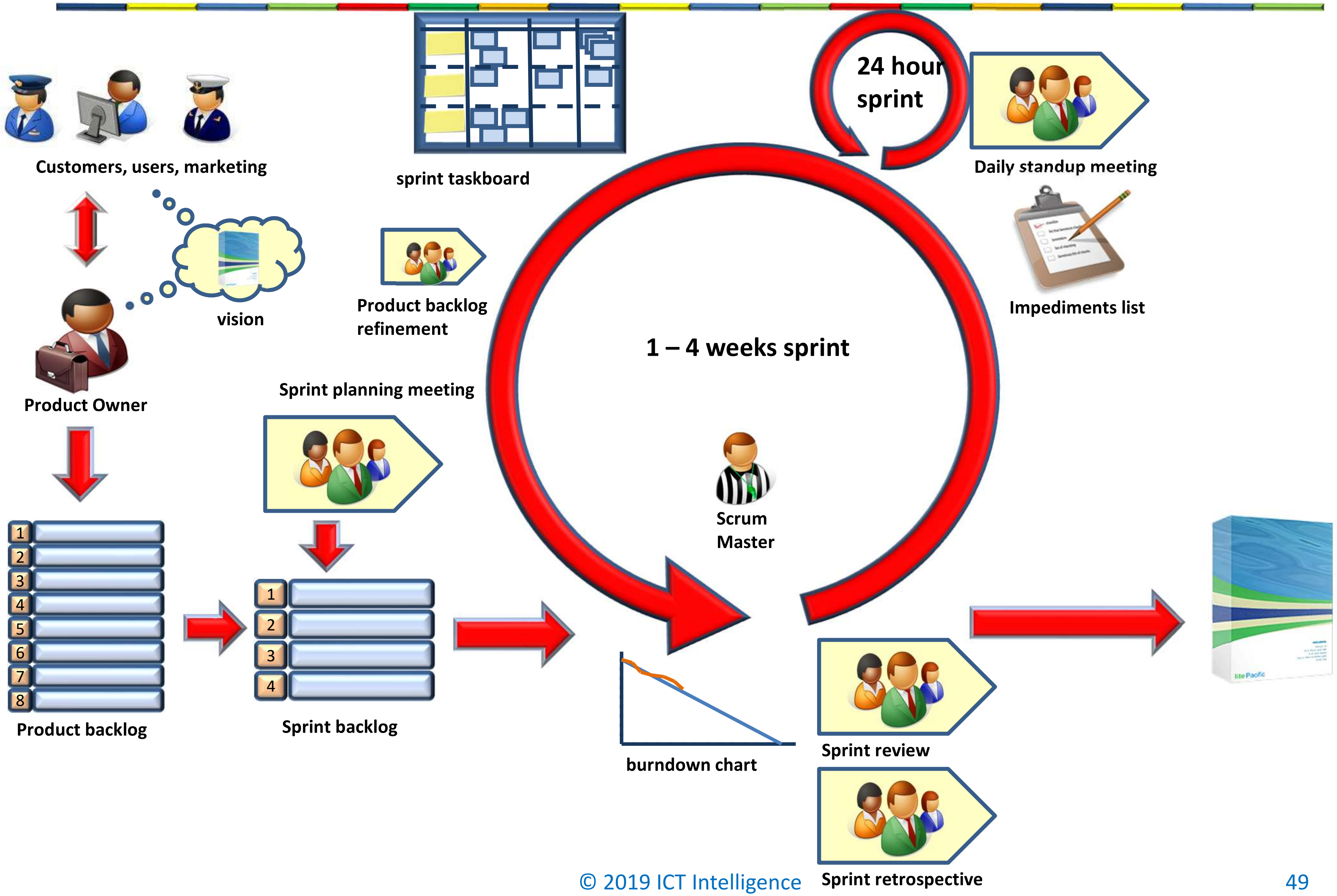
Tester

Developer

Designer/Developer

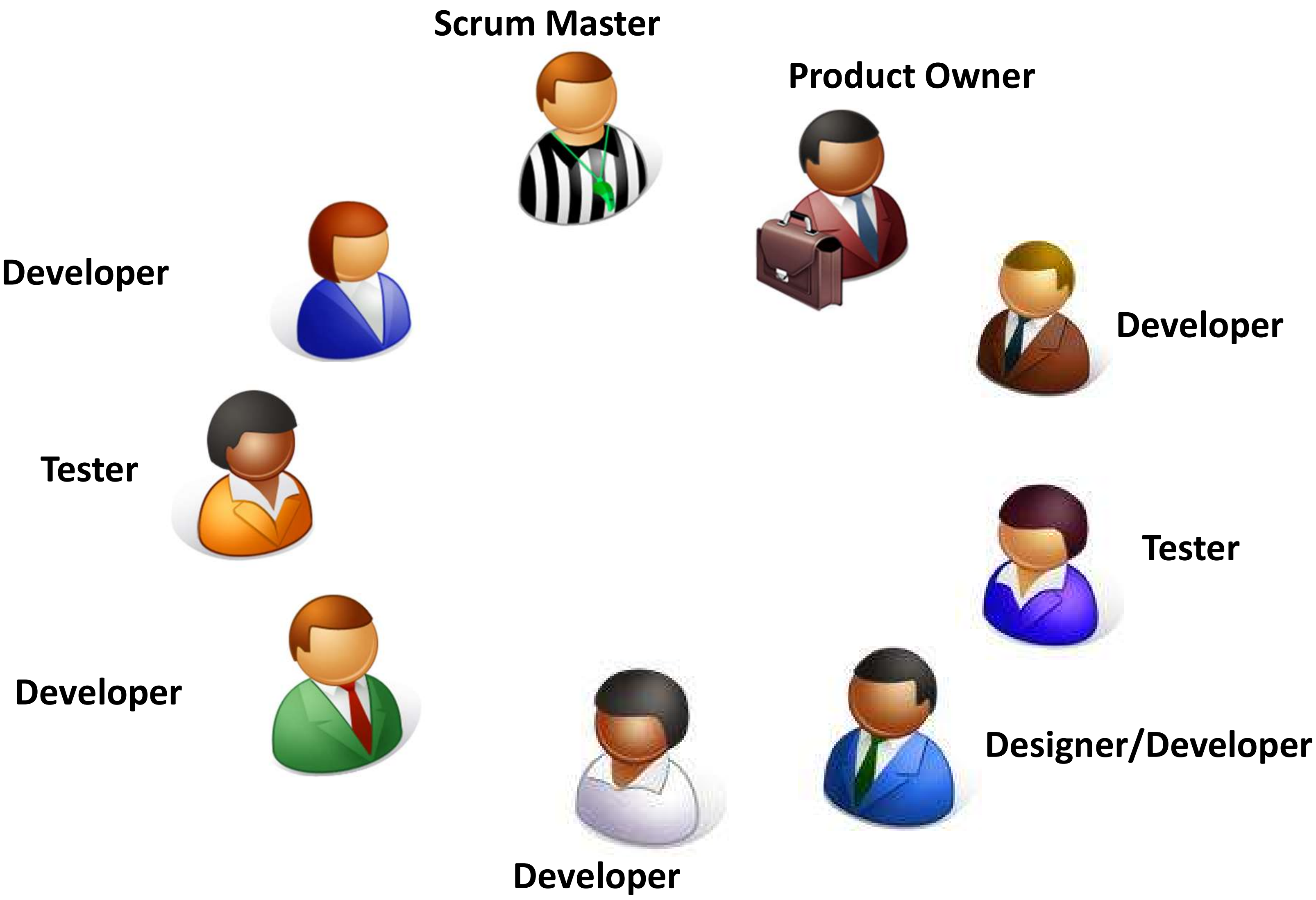
Developer

Scrum in action



SCRUM TEAM

Cross functional team



Scrum Master

Product Owner

Developer

Developer

Tester

Tester

Developer

Designer/Developer

Developer

Why a cross functional team?

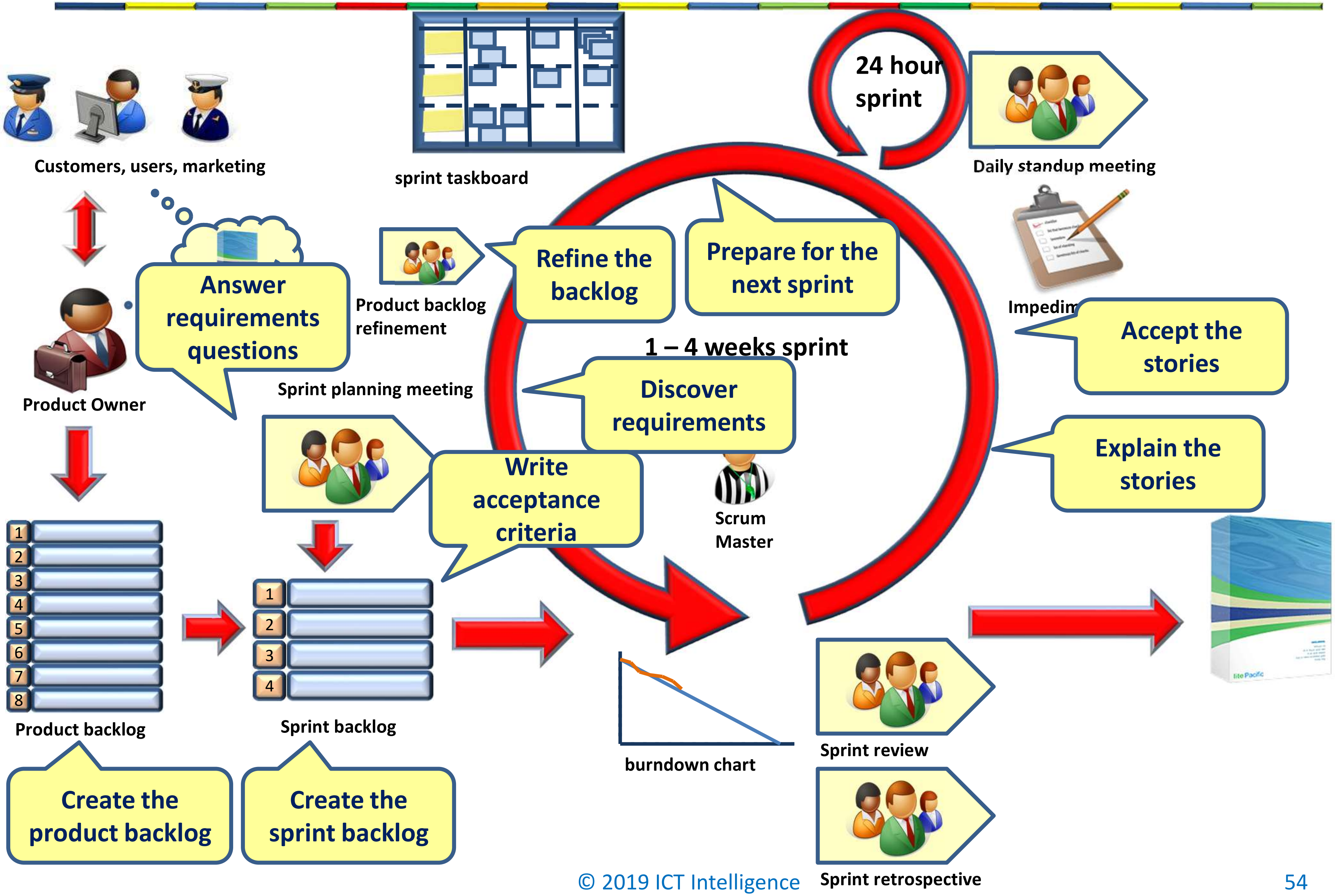
- Not everyone knows everything
- We get different perspectives
- Improved innovation
- Increased speed
  - Team has all the skills necessary to achieve it’s goal.

The product owner

Product Owner

- Is responsible that the business gets what they need:
  - Discovers and defines the product features (requirements)
  - Changes these features and priority in every iteration
  - Communicates these features to the team
  - Accepts the result created by the team

Tasks of the product owner

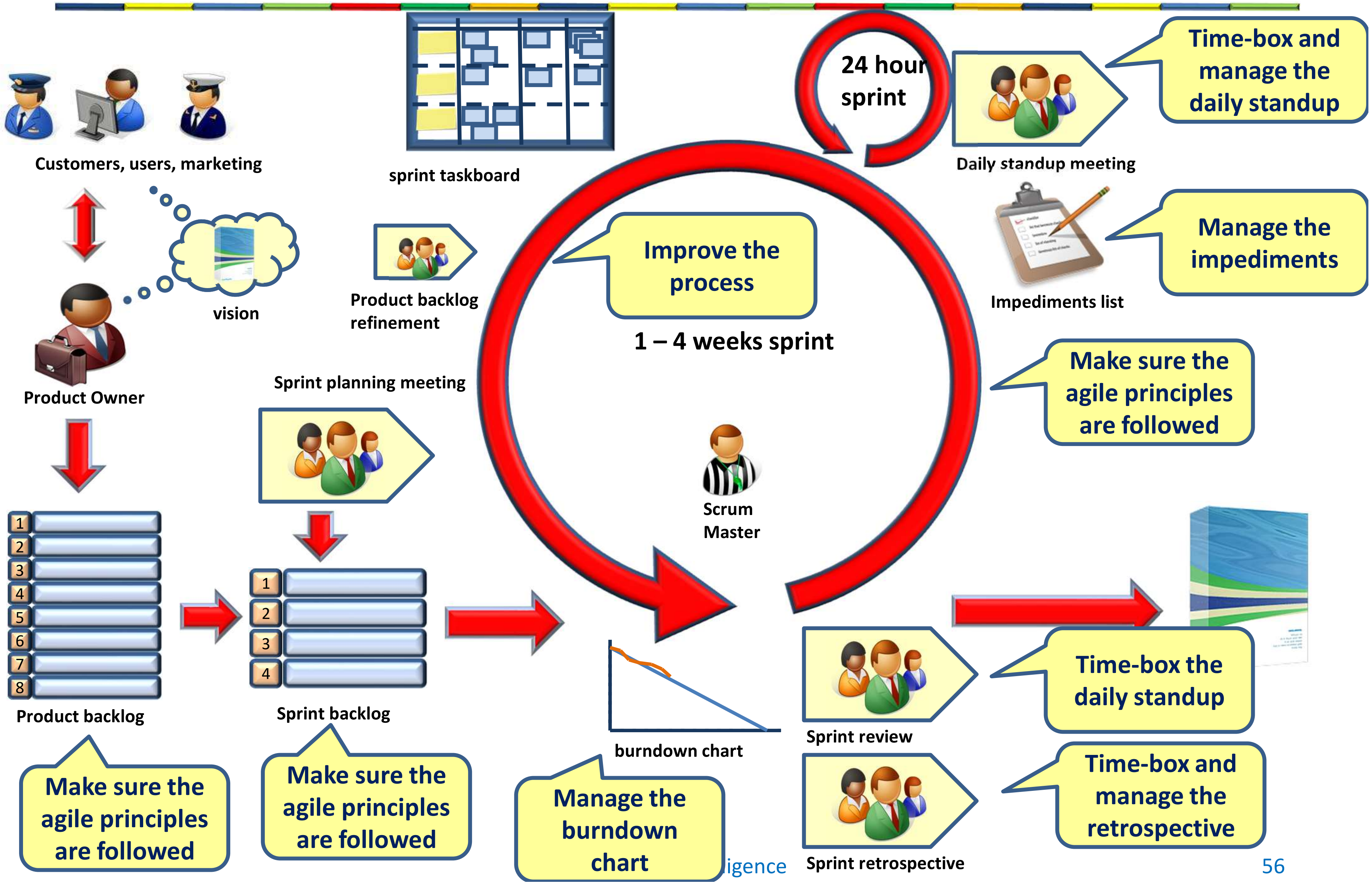


De scrum master

Scrum Master

- Is responsible that the team works in the most optimized and efficient way:
  - Improves the productivity of the team
    - Protects the team from disruptions
    - Facilitates
  - Creates the team
    - Like a sport coach
  - Makes sure the agile principles are understood and correctly implemented
  - Removes impediments
  - Coaches the whole team
- Is not a project manager

Tasks of the scrum master



Customers, users, marketing

vision

Product Owner

sprint taskboard

Product backlog refinement

Sprint planning meeting

Product backlog

Sprint backlog

24 hour sprint

Daily standup meeting

Time-box and manage the daily standup

Impediments list

Manage the impediments

Improve the process

1 – 4 weeks sprint

Make sure the agile principles are followed

Scrum Master

burndown chart

Sprint review

Time-box the daily standup

Sprint retrospective

Time-box and manage the retrospective

Make sure the agile principles are followed

Make sure the agile principles are followed

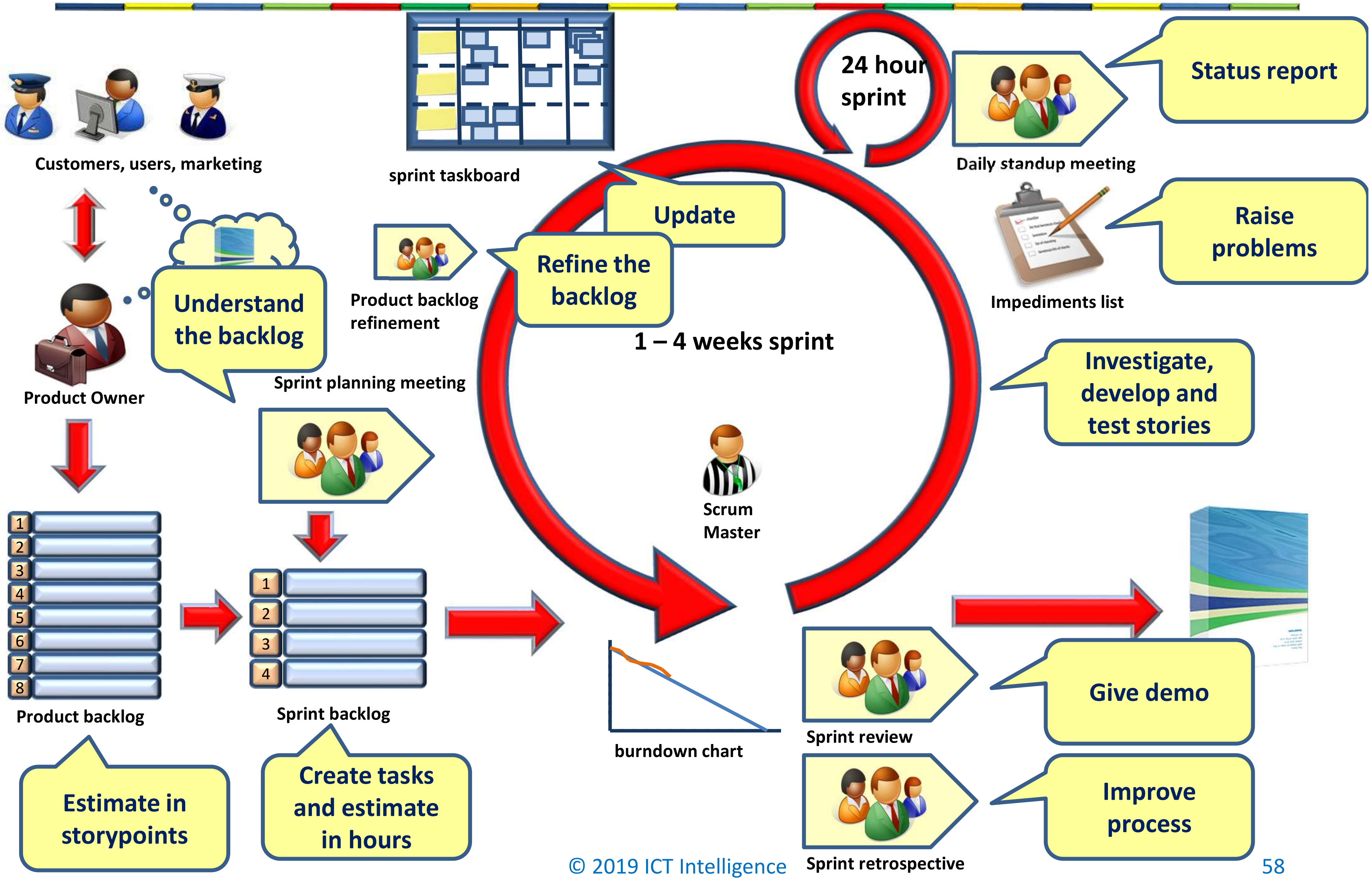
Manage the burndown chart

The team

Scrum Team

- The people that implement the product:
  - Cross-functional
  - Self-organizing
  - Self managing
  - Team members are full time team members(ideally)
  - Team works at the same location (ideally)
  - Committed
  - Empowered

Tasks of the team

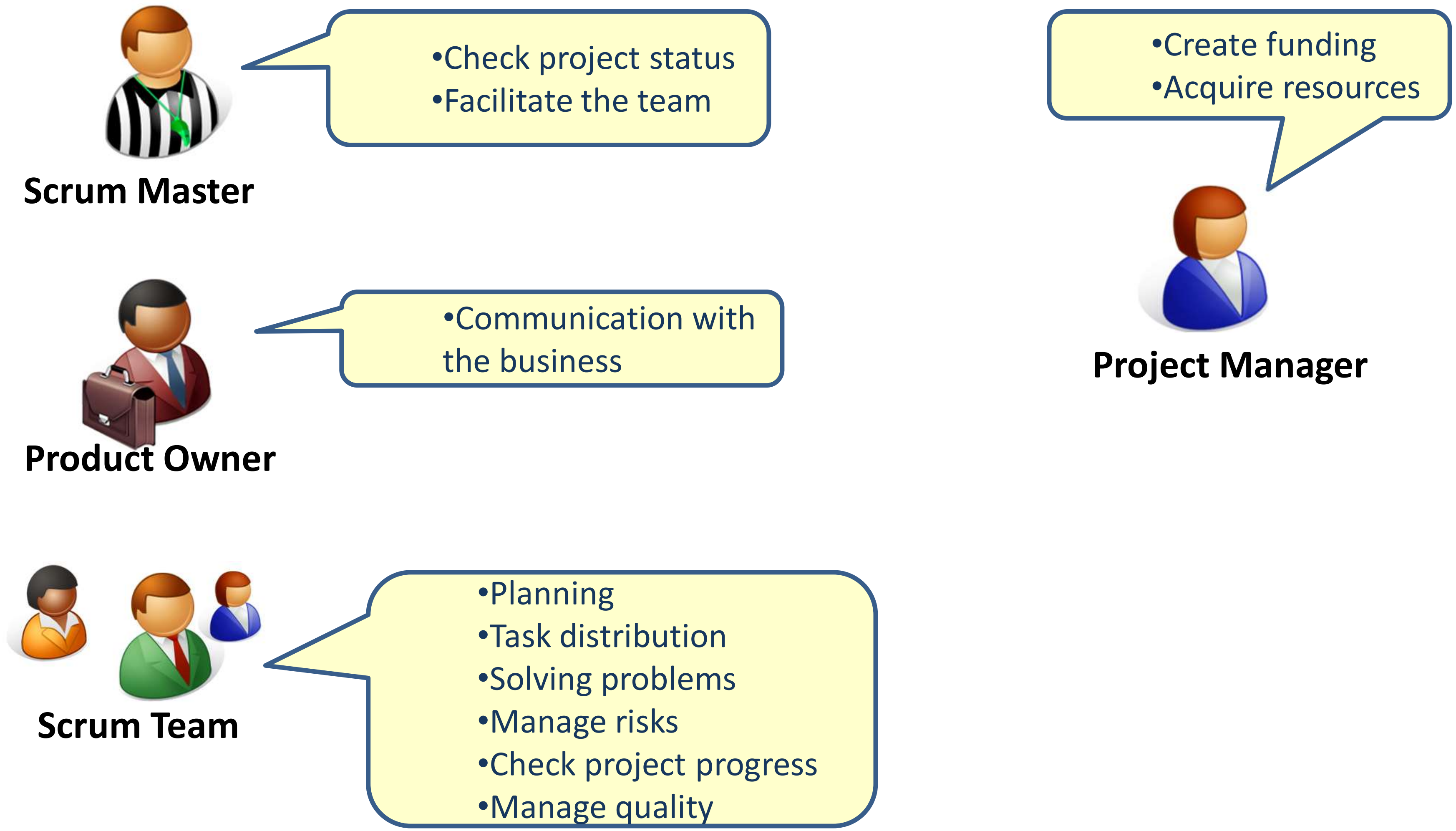


Self managing team

- The team decides(not the manager)
- The team is responsible(not the manager)
- The team improves continuously

HEY PIG, I WAS THINKIN' WE SHOULD OPEN A RESTAURANT.
I DON'T KNOW. WHAT WOULD WE CALL IT?
Ham -n- Eggs
HOW ABOUT "HAM -N- EGGS."
NO THANKS. I'D BE COMMITTED, BUT YOU'D ONLY BE INVOLVED!
By Clark & Vizdos
© 2006 implementingscrum.com

Project management tasks in Scrum



Scrum Master

- Check project status
- Facilitate the team

Product Owner

- Communication with the business

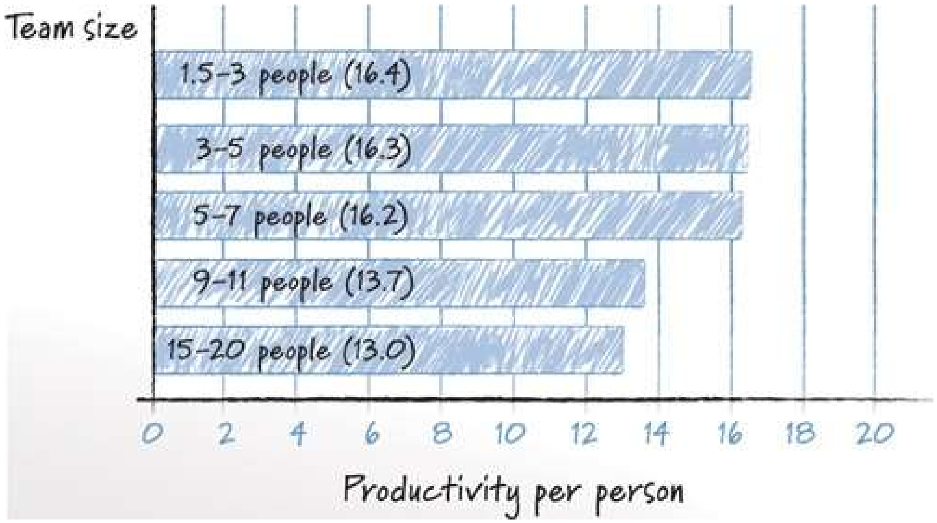
Scrum Team

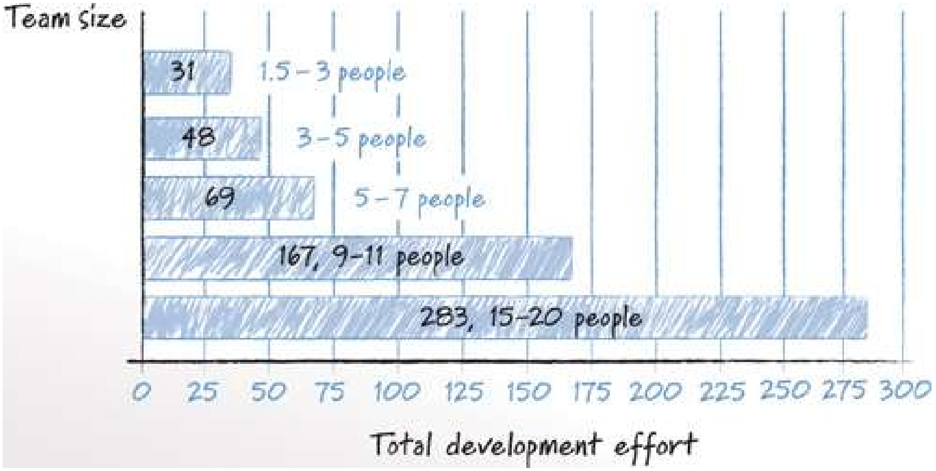
- Planning
- Task distribution
- Solving problems
- Manage risks
- Check project progress
- Manage quality

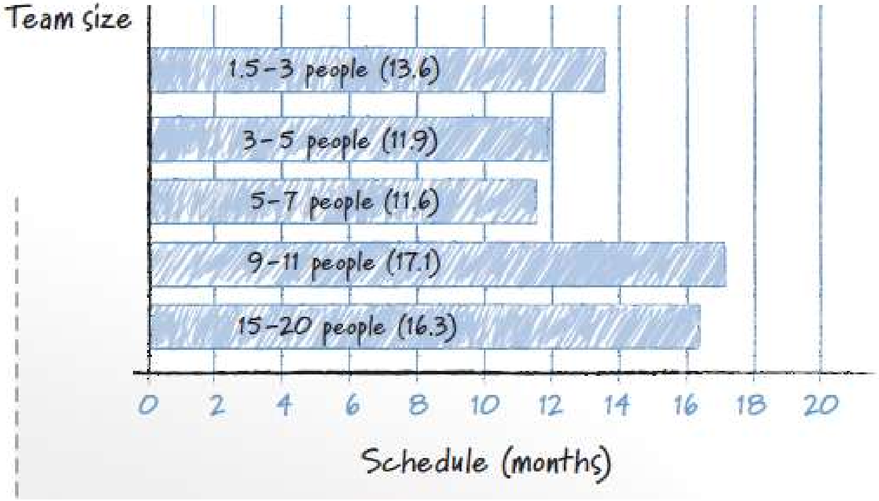
Project Manager

- Create funding
- Acquire resources

Size of the team (5-9 people)







Main point

Software engineering

Scrum is currently the most efficient software development methodology

SCI

With TM, one gets access to the unlimited potential of nature

AGILE/SCRUM CHALLENGES

Agile/Scrum challenges

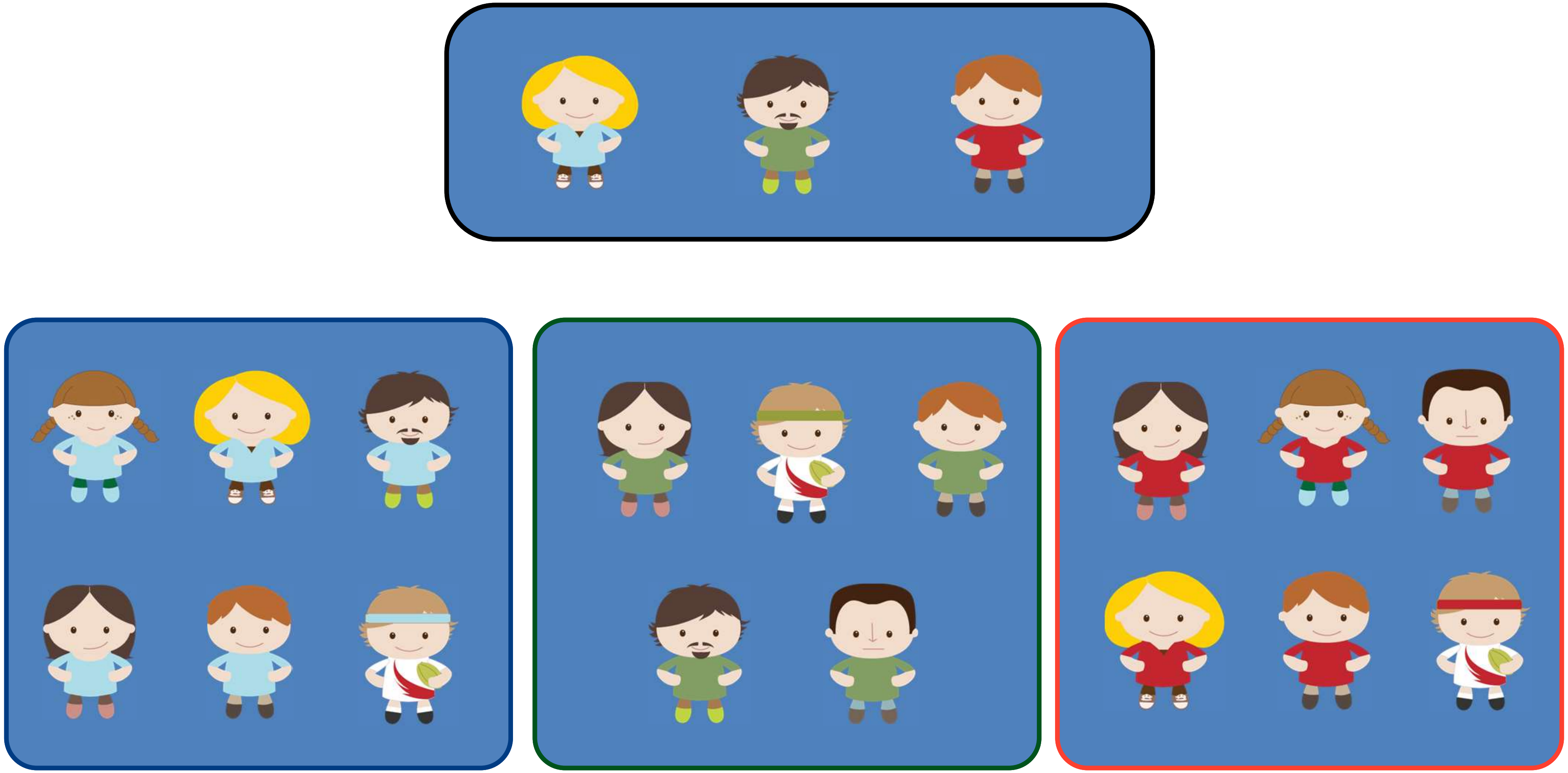
- The organizational culture needs to change
- Scaling scrum
- Scrum but
- Scrum does not solve all your problems
- The Product Owner role is difficult
- Scrum does not always fit

The organizational culture needs to change

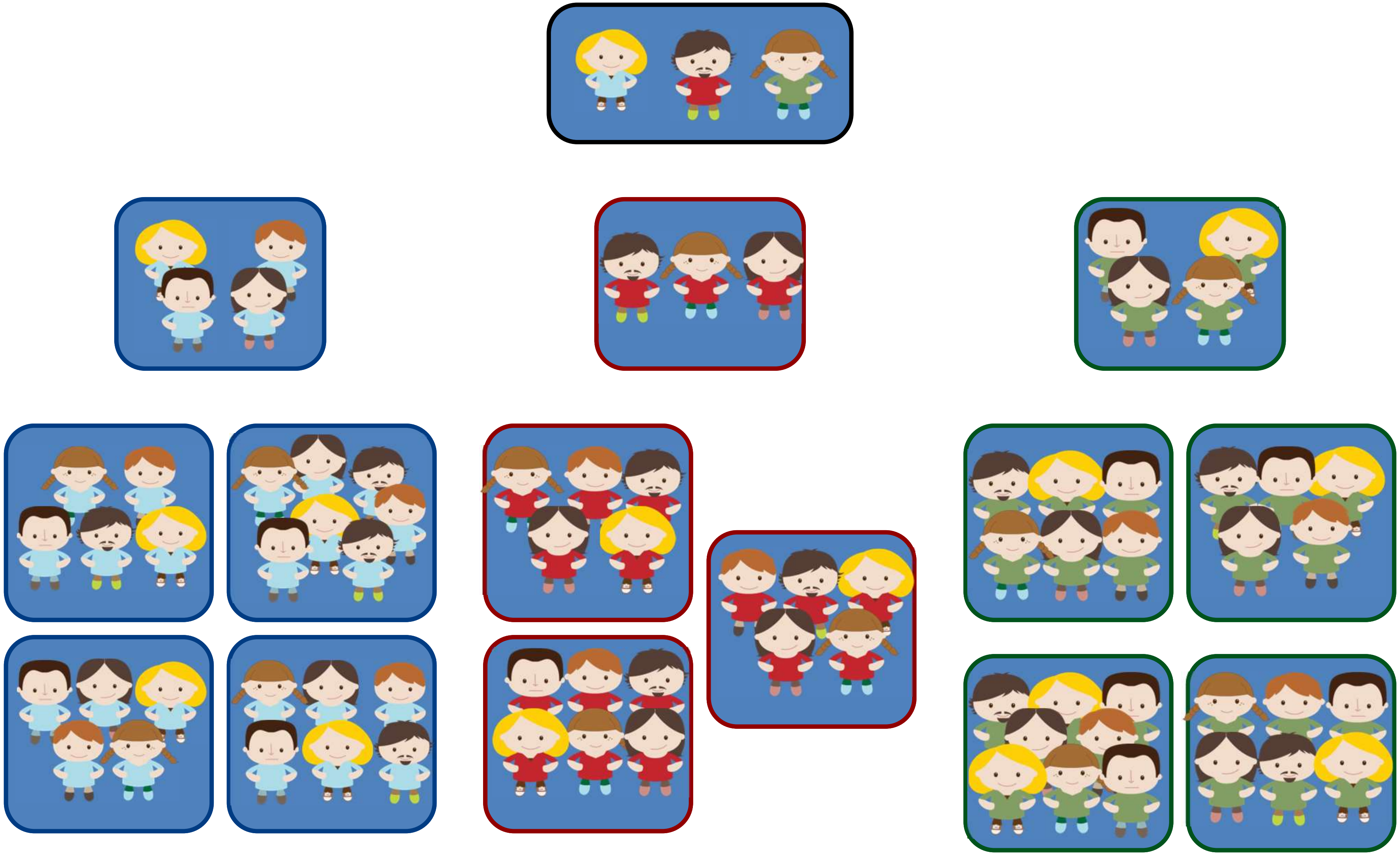
- Change of roles and responsibilities
  - There is no project leader anymore
  - The product owner is part of the development team
- The whole organization needs to become agile
  - Focus on priorities instead of fixed plans

Scaling Scrum

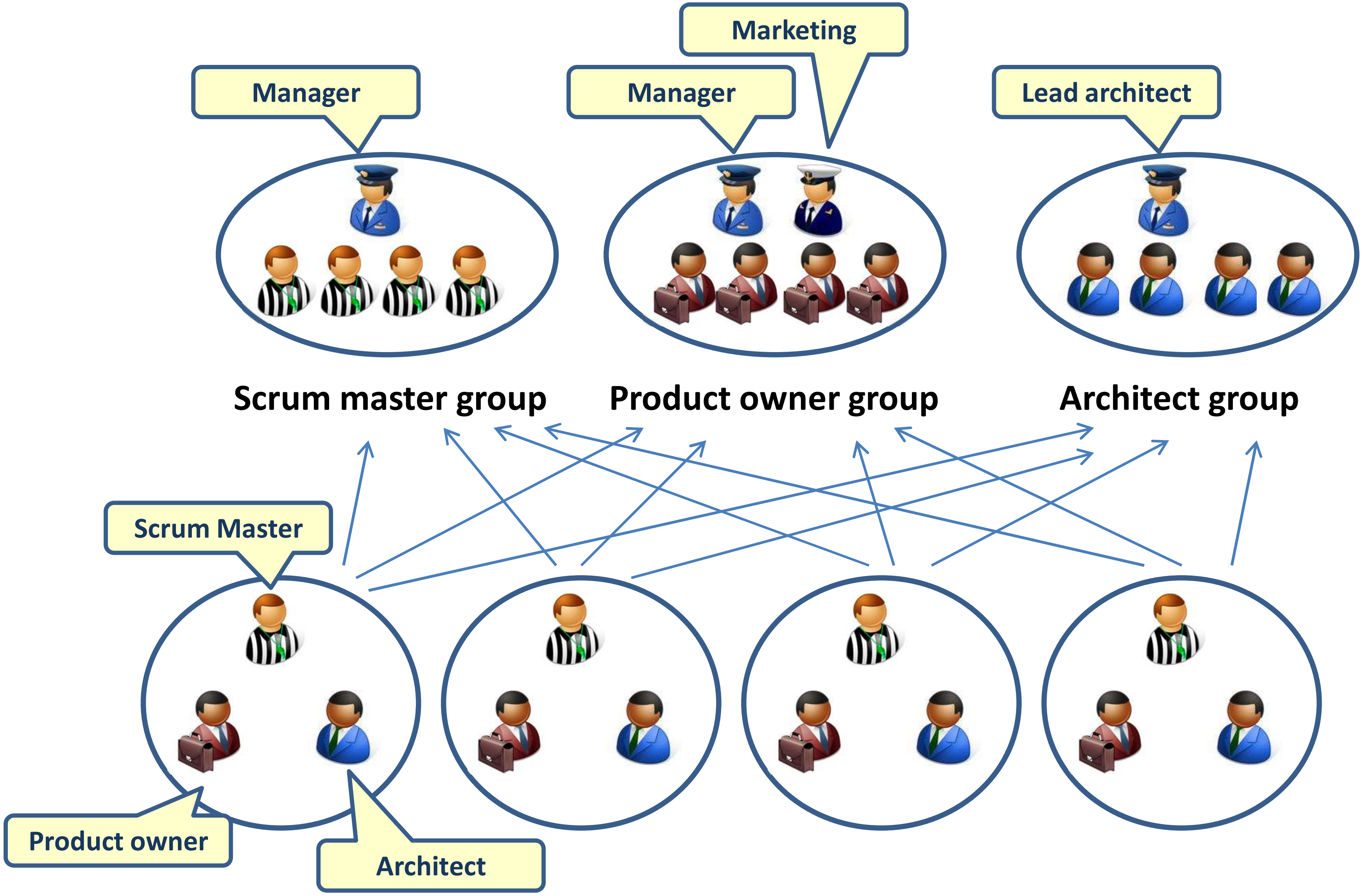
- Scrum of Scrums



Scrum of scrums of scrums



Scaling scrum



Scrum but

- Scrum but
  - We use Scrum, but we have these unique circumstances so we have had to modify Scrum so it works here
  - Modify Scrum without understanding the basic agile principles
- Solution(s):
  - start applying scrum from the book so you learn the basic agile principles
  - Then modify Scrum based on the basic agile principles
  - Maybe you organization can/must change

Scrum does not solve all your problems

- Scrum is not a silver bullet
- Scrum is not complete
  - We need a preparation phase
  - We need agile engineering best practices
    - Continuous integration
    - Automatic testing
    - Refactoring
  - We need a complete estimation and plan upfront so we can apply for budget
  - We need requirements best practices

The Product Owner role is difficult

- Requirements is the hardest part of software development
- You need customer involvement
- You need a capable PO
  - Who is allowed to make decisions
- The PO needs to discover requirements together with the business and at the same time answer questions from the Scrum team.
- Solution(s):
  - Create PO teams

Scrum does not always fit

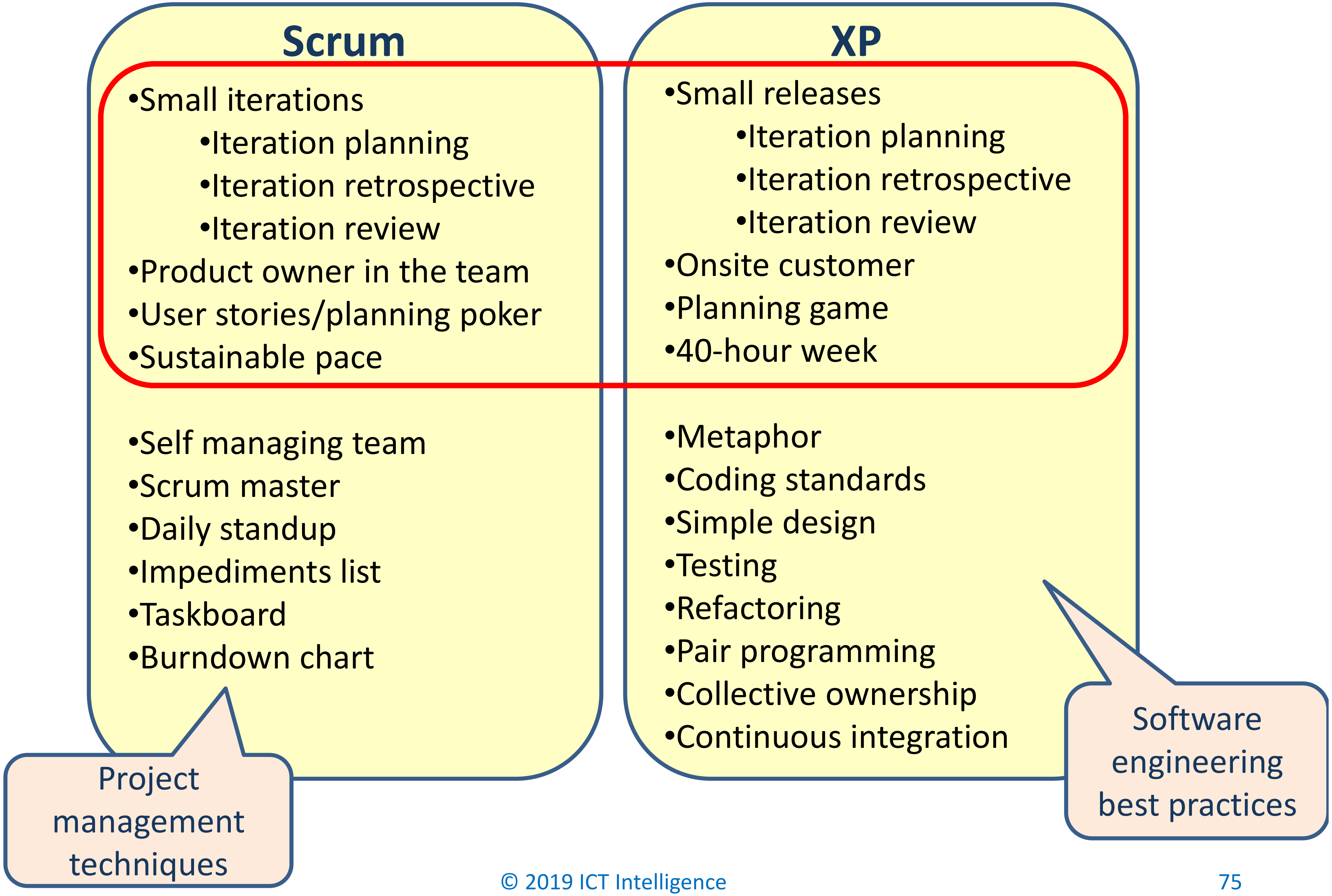
- You need a team of 5-9 people
  - Who work on the same product
    - Writing new functionality or changing existing functionality
  - At the same location
- Software maintenance often does not work that way
  - Most agile principles can still be adopted

OTHER AGILE SOFTWARE DEVELOPMENT METHODS

eXtreme Programming (XP)

- Small releases
- Onsite customer
- Planning game
- Simple design
- Testing
- Refactoring
- Pair programming
- Collective ownership
- Continuous integration
- Metaphor
- Coding standards
- 40-hour week

XP and Scrum



Scrum

- Small iterations
  - Iteration planning
  - Iteration retrospective
  - Iteration review
- Product owner in the team
- User stories/planning poker
- Sustainable pace

- Self managing team
- Scrum master
- Daily standup
- Impediments list
- Taskboard
- Burndown chart

XP

- Small releases
  - Iteration planning
  - Iteration retrospective
  - Iteration review
- Onsite customer
- Planning game
- 40-hour week

- Metaphor
- Coding standards
- Simple design
- Testing
- Refactoring
- Pair programming
- Collective ownership
- Continuous integration

Project management techniques

Software engineering best practices

Lean

- Provide perfect value to the customer through a perfect value creation process that has the least amount of waste
- History
  - Ford
    - Process thinking (flow)
    - You can choose any color as long as it is black

  - Toyota
    - Toyota Production System (TPS)
- Not only for production systems.

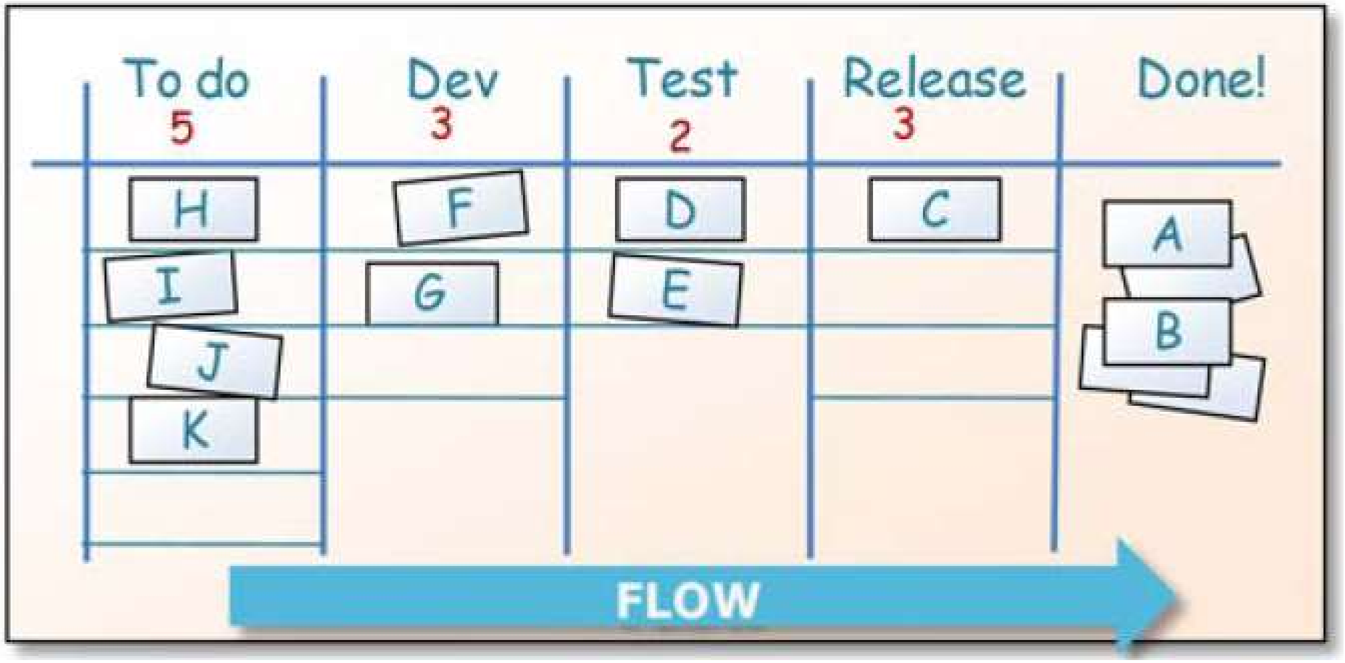
The Seven Principles of Lean Thinking

- Eliminate Waste
  - Eliminate anything that doesn’t add value (as perceived by the customer) to the product
- Amplify Learning
  - Use short feedback loops
- Decide as late as possible
- Deliver as fast as possible
- Empower the team
  - Self managing and self organizing
- Build integrity in
  - High quality software accepted by the business
- See the whole

Implementing Lean

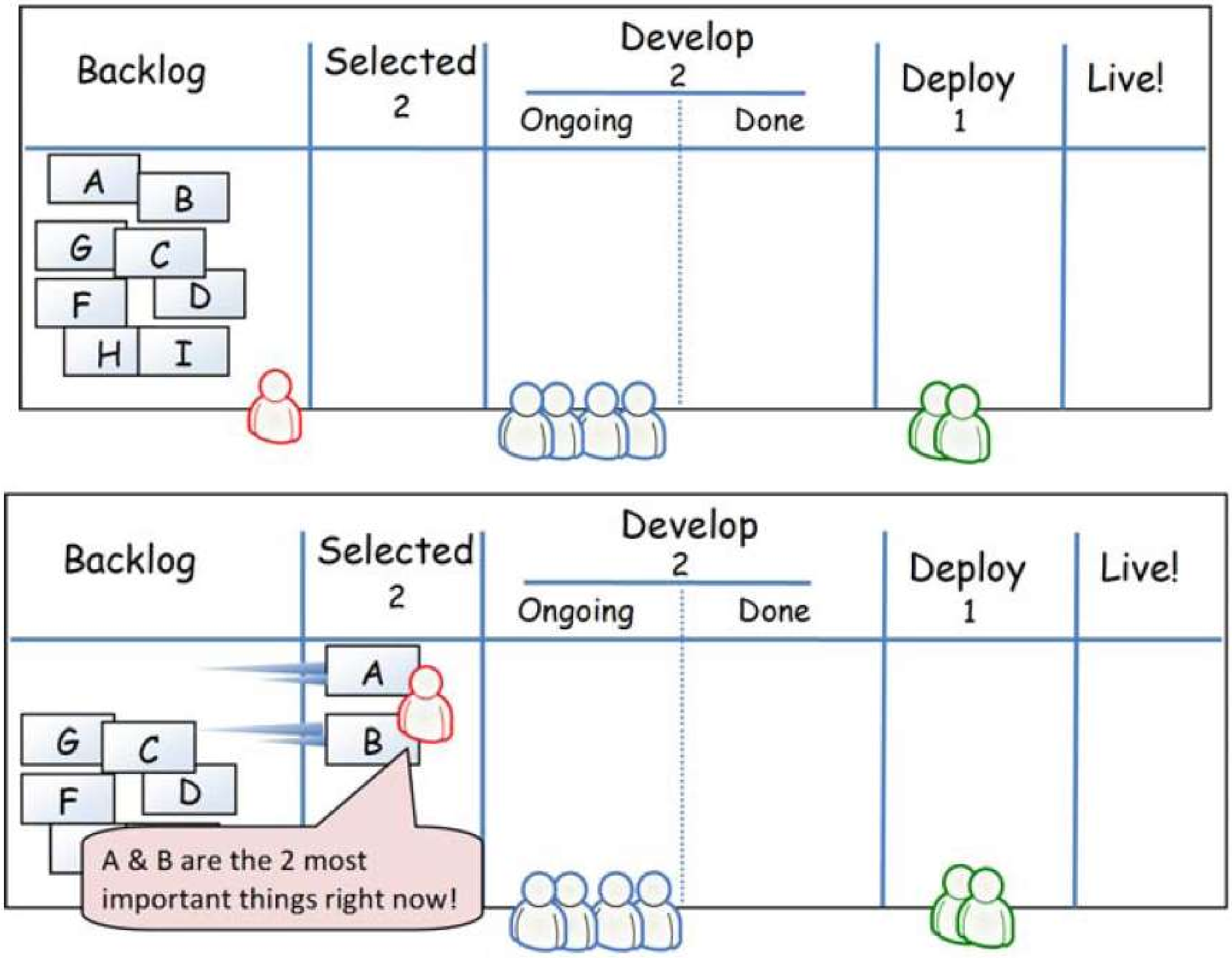
Lean principle	Software development discipline
Eliminate waste	Dedicated Product Owner
	Agile documentation
	Agile communication
	Customer acceptance testing
Amplify learning	Small iterations
	Early feedback
	Continuous integration
Decide as late as possible	Agile requirements/user stories
	Small iterations
	Agile planning
Deliver as fast as possible	Small iterations
Empower the team	Self-managing team
Build integrity in	Coding standards
	Simple design
	Testing
	Refactoring
	Pair programming
	Collective ownership
	Continuous integration
See the whole	Agile estimation
	Burndown chart
	The whole organization needs to become agile

Kanban

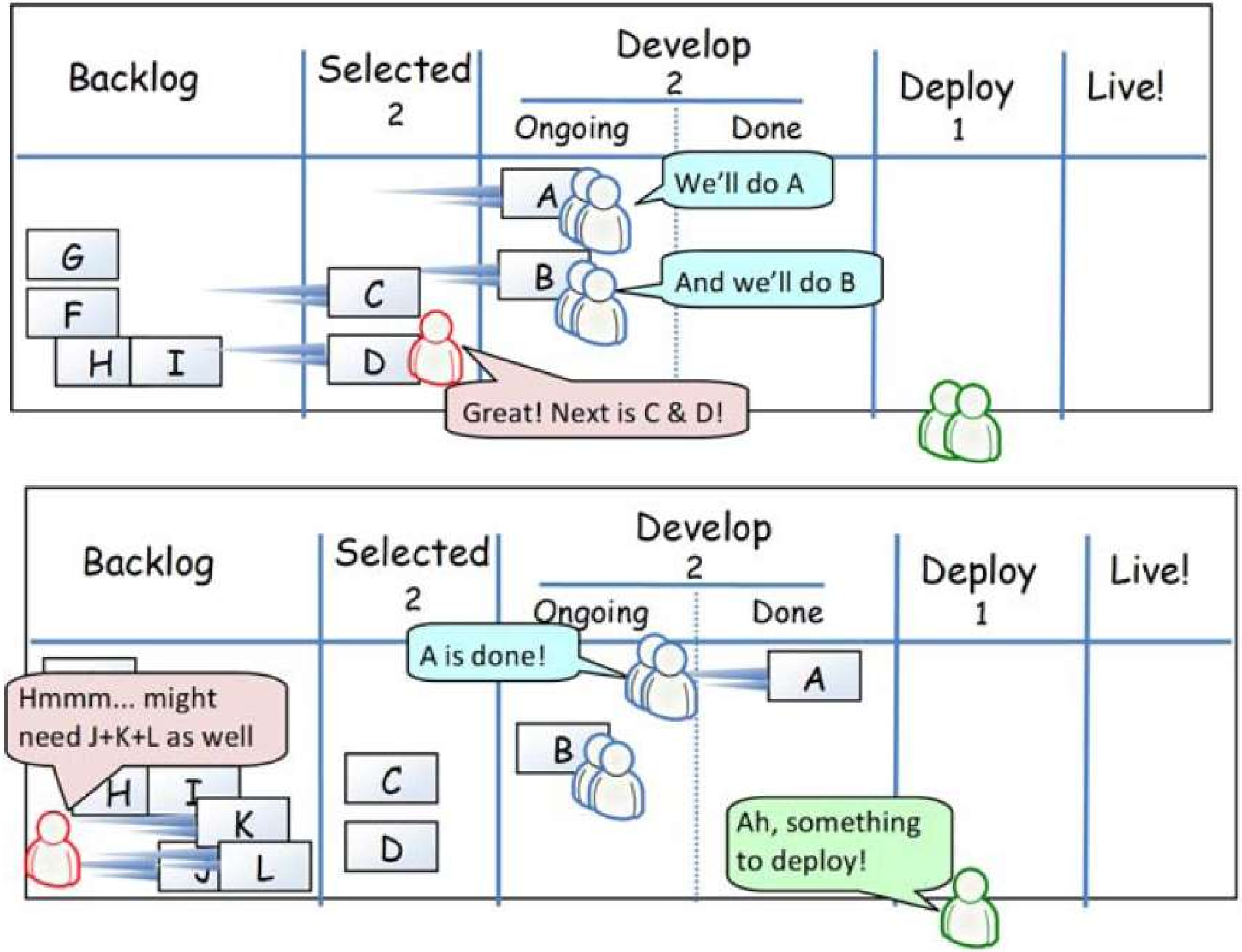


- Scrum without fixed iterations(sprints)

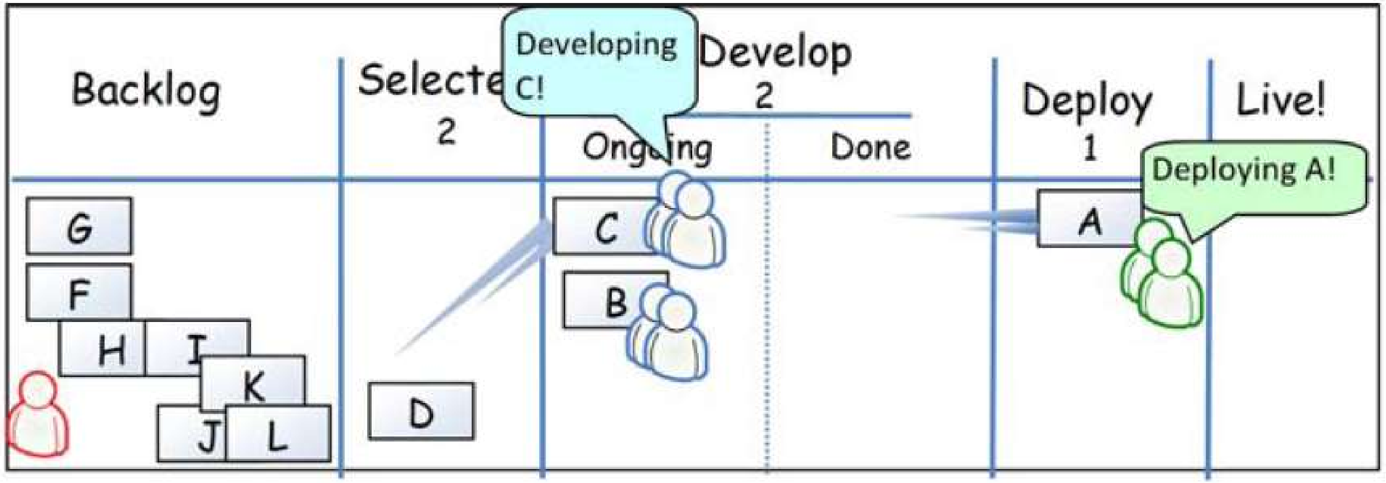
Kanban example

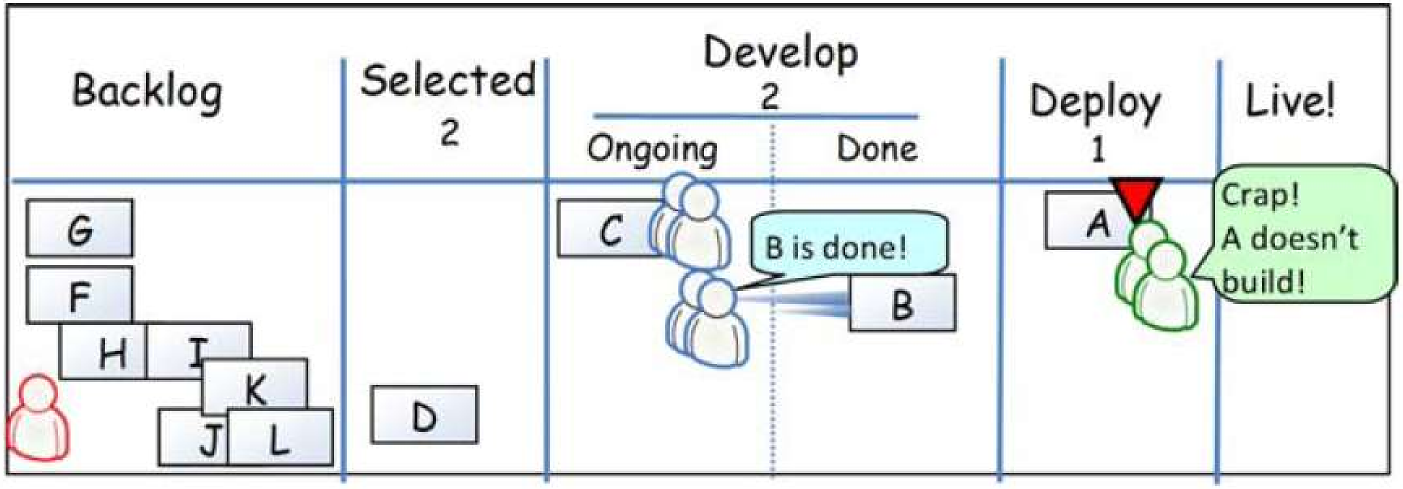


Kanban example

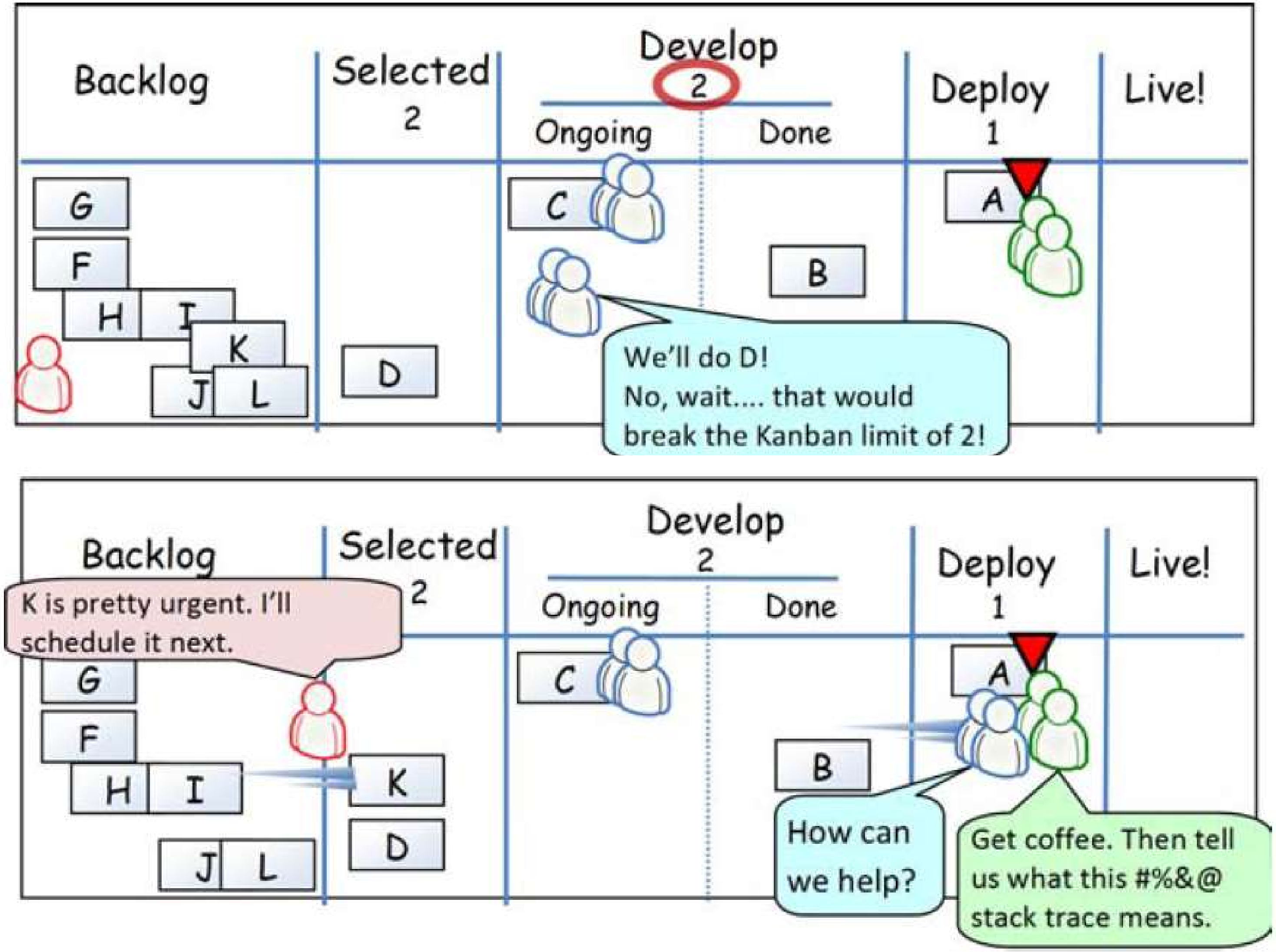


Kanban example

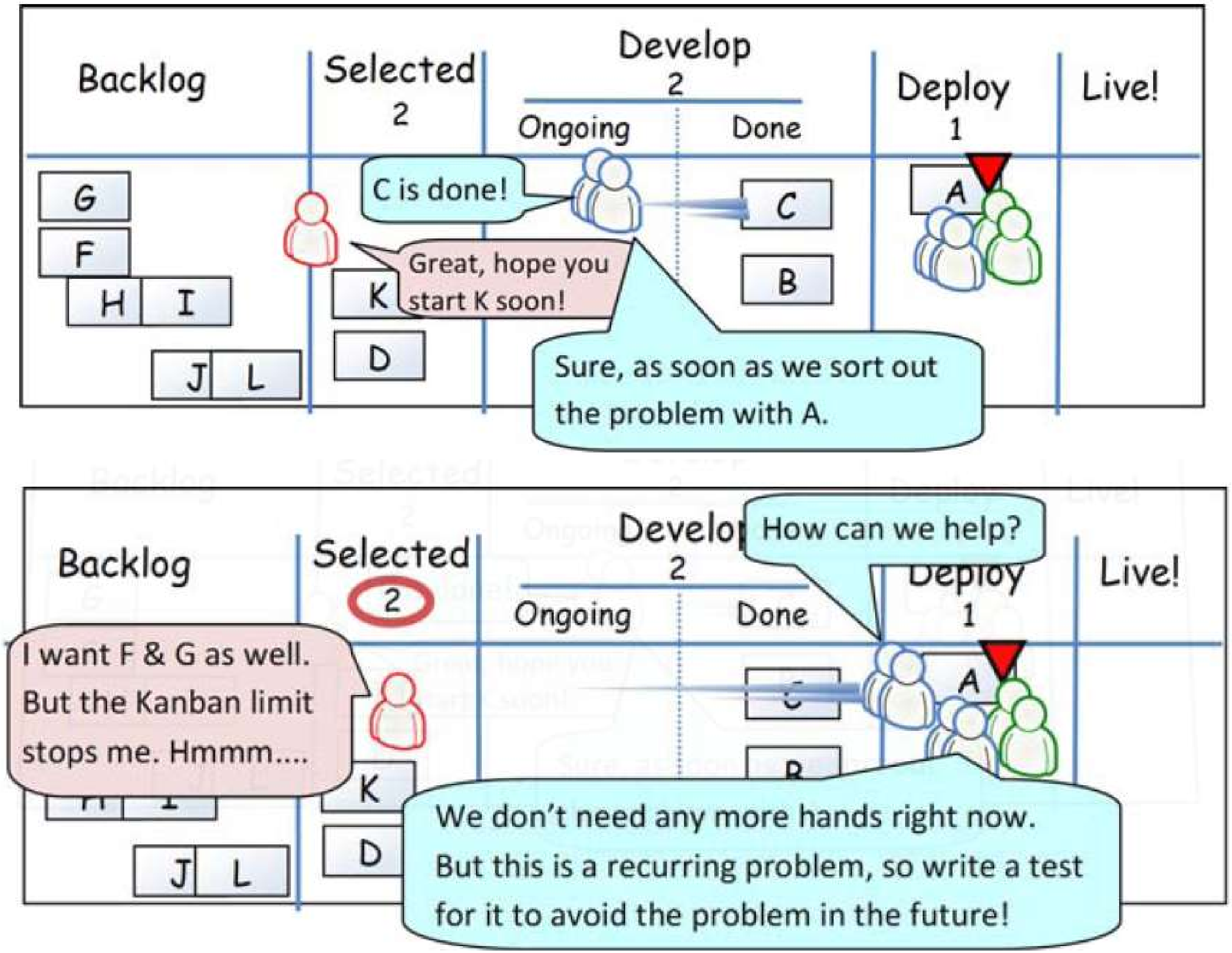




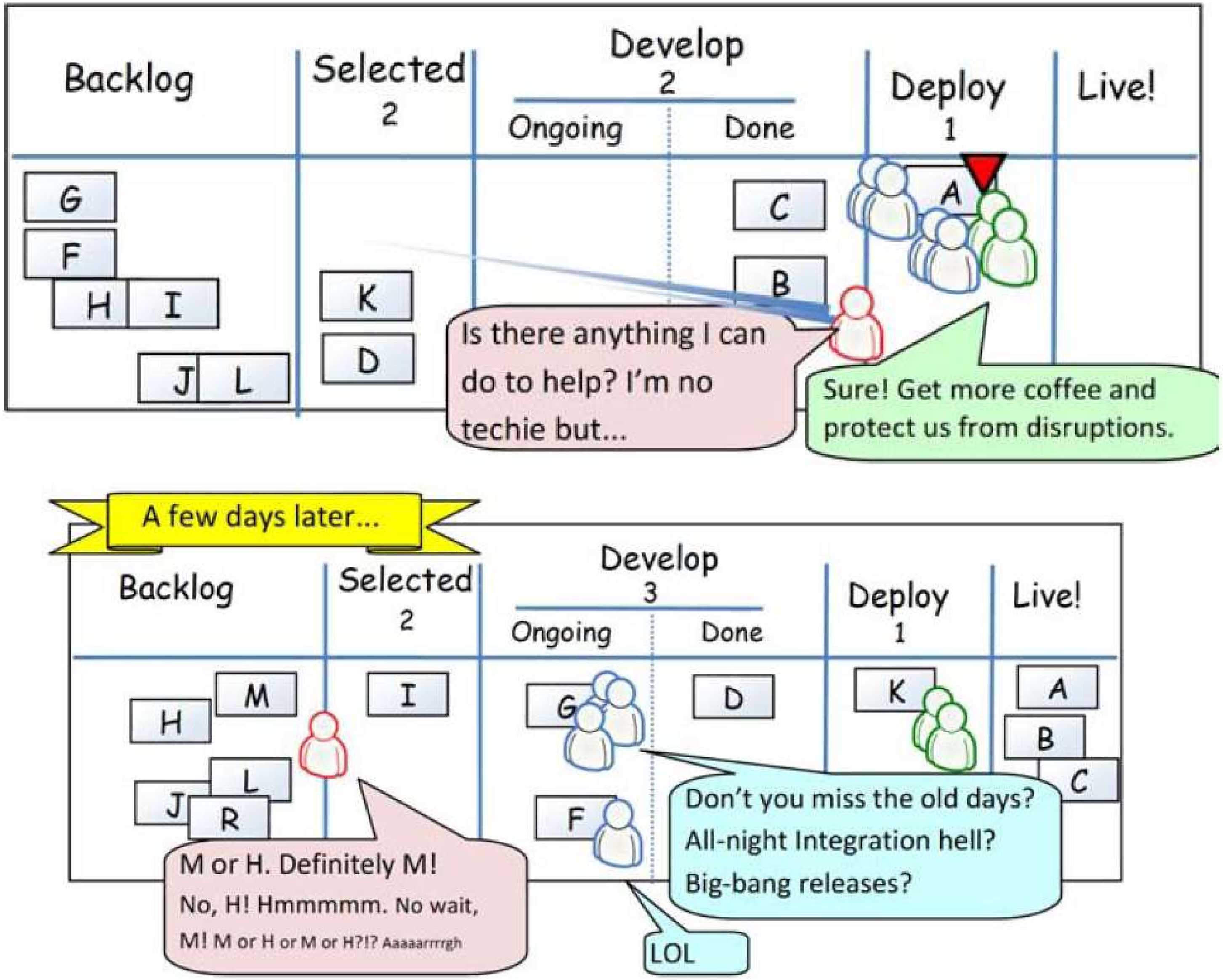
Kanban example



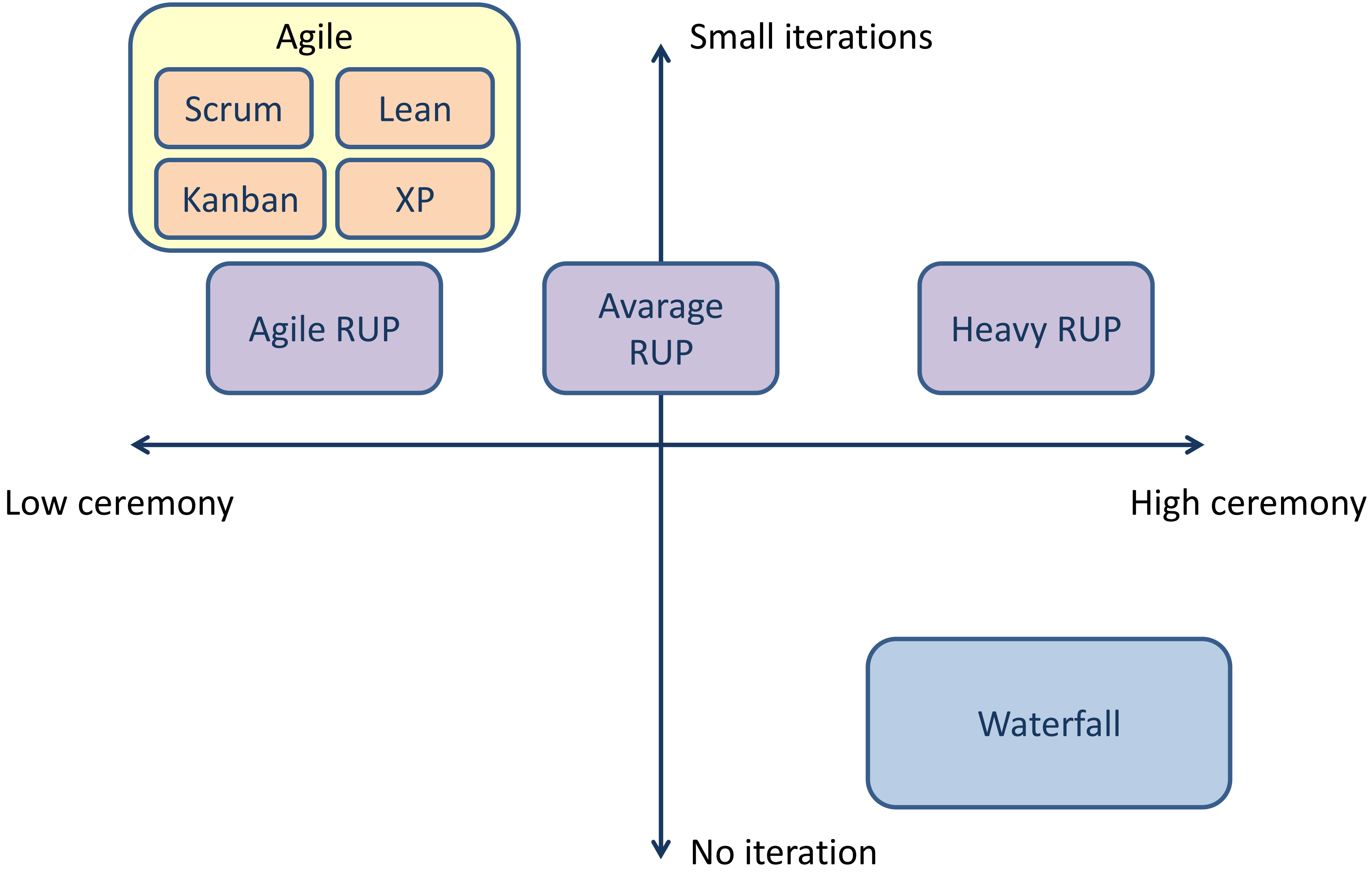
Kanban example


Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
G
F
H
I
J
L
K
D
C
B
A
C is done!
Great, hope you start K soon!
Sure, as soon as we sort out the problem with A.
Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
How can we help?
I want F & G as well. But the Kanban limit stops me. Hmmm....
K
D
J
L
C
B
A
We don't need any more hands right now. But this is a recurring problem, so write a test for it to avoid the problem in the future!

Kanban example


Backlog
Selected 2
Develop 2
Ongoing
Done
Deploy 1
Live!
G
F
H
I
J
L
K
D
C
B
A
Is there anything I can do to help? I'm no techie but...
Sure! Get more coffee and protect us from disruptions.
A few days later...
Backlog
Selected 2
Develop 3
Ongoing
Done
Deploy 1
Live!
M
H
L
J
R
I
G
F
D
K
A
B
C
M or H. Definitely M! No, H! Hmmmmm. No wait, M! M or H or M or H?!? Aaaaarrrrgh
Don't you miss the old days? All-night Integration hell? Big-bang releases?
LOL

Software development methods



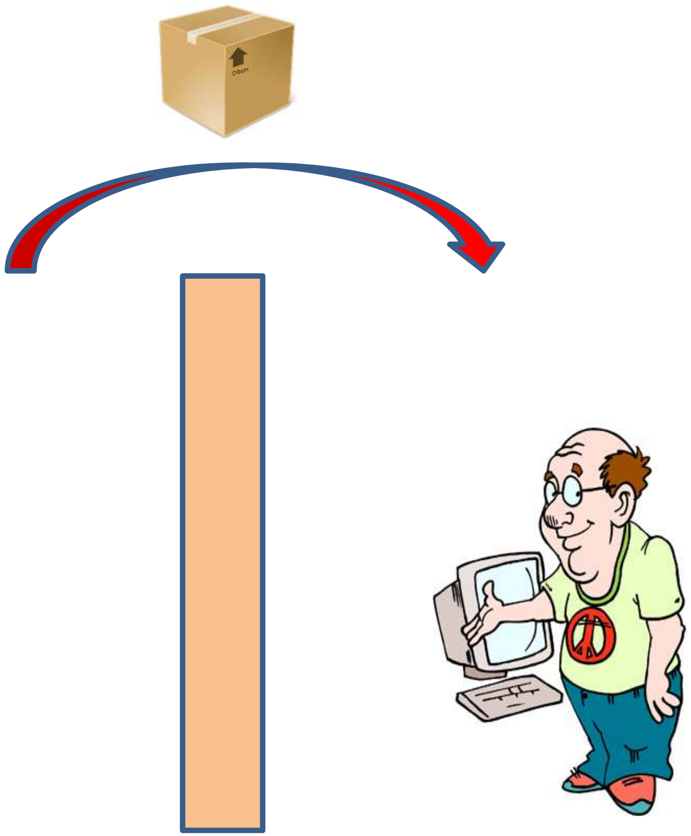
DEVOPS

Agile software development: Scrum

- Close collaboration
- Better communication
- Short delivery cycles
- Short feedback loops

Product owner (business)
and developers in one team

Application



Operations

Why DevOps?

Business

I want

1. The software to change to adept to the changing world as fast as possible
2. The existing IT services to remain stable and not disrupted from the changes

- New releases
- New features
- New platforms
- New architecture
- Functional requirements

Development

I provide change

Both have their own

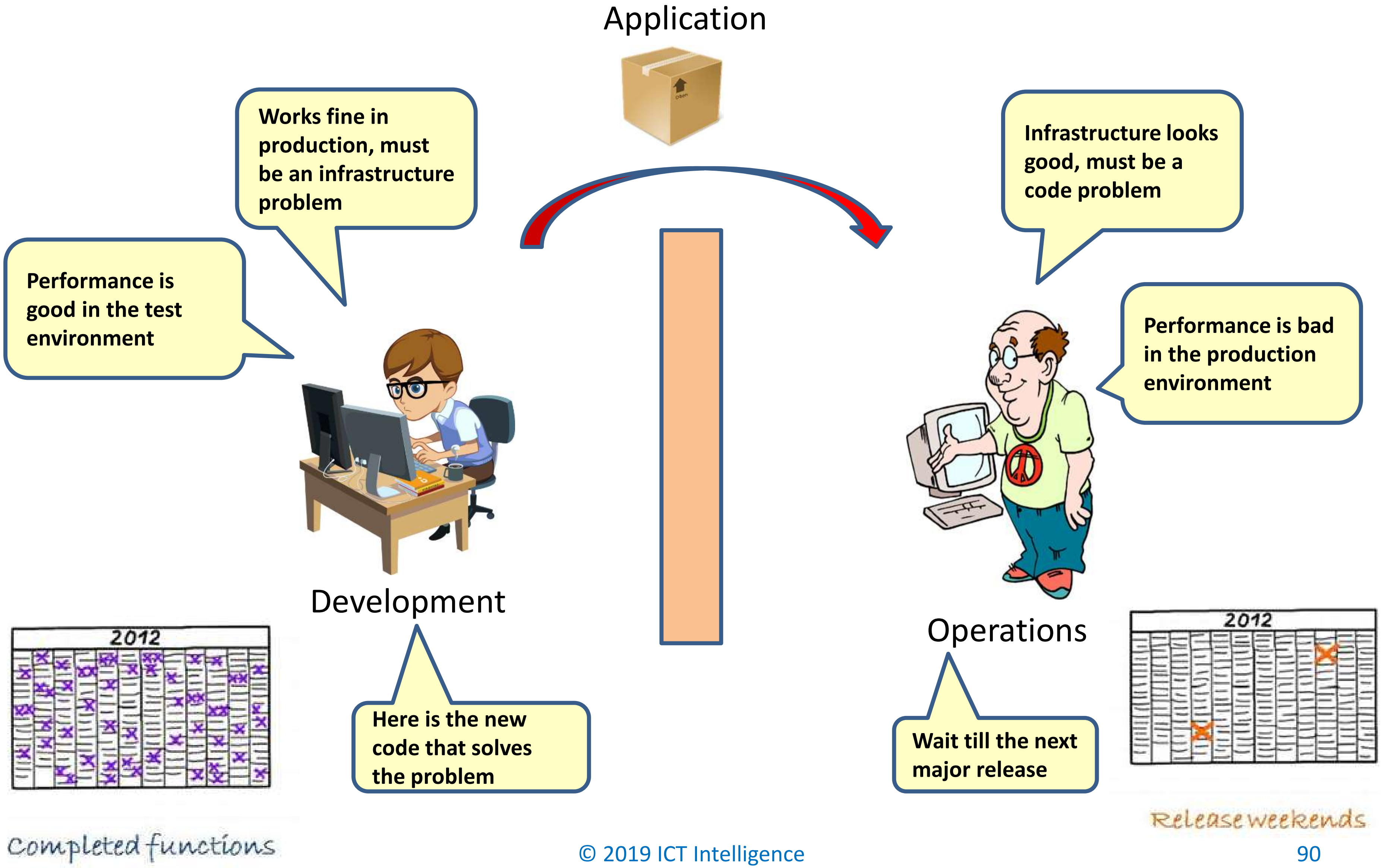
- Goals
- Tools
- Process
- Schedules

- Stable platforms
- No downtime
- Scalable platform
- Non-functional requirements

Operations

I provide stability

Why DevOps?



Application

Performance is good in the test environment

Works fine in production, must be an infrastructure problem

Development

Here is the new code that solves the problem

Infrastructure looks good, must be a code problem

Performance is bad in the production environment

Operations

Wait till the next major release

DevOps

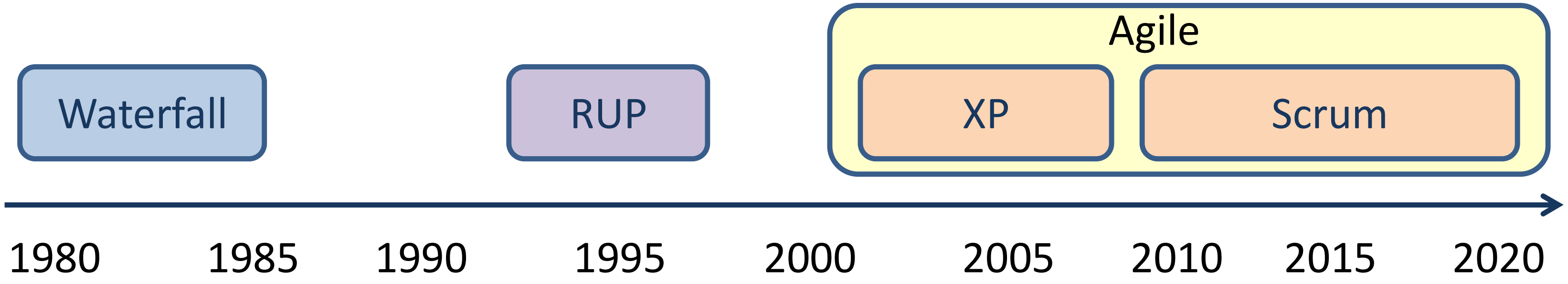
- Close collaboration between developers and operations
- Streamlines the delivery process of software from business requirements to production
- Better communication
- Identical development and production environment
- Shared tools
  - Automate everything
  - Monitor everything

Product owner (business) and developers in one team

Operations

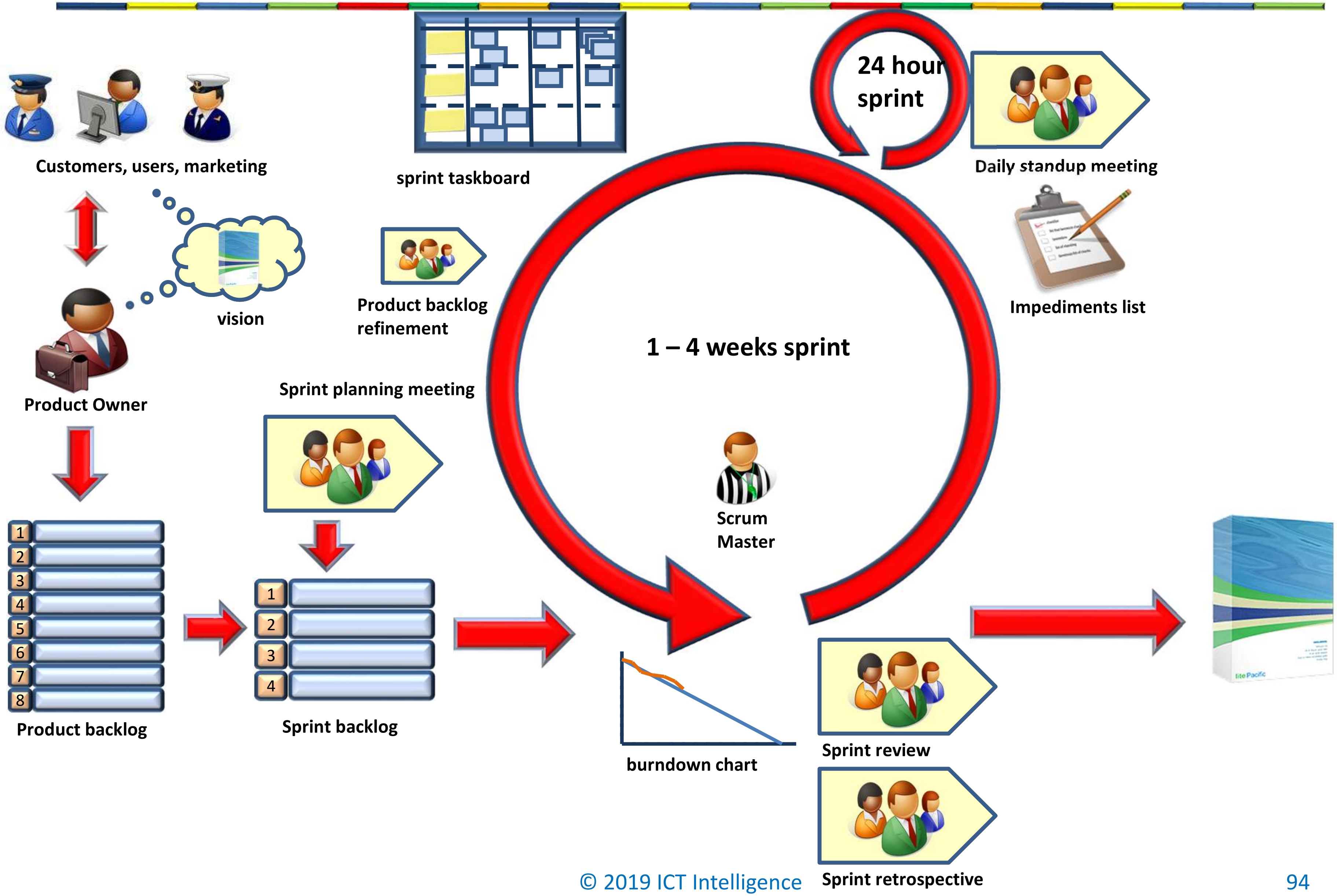
SUMMARY

Software development methods



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

Scrum framework



Advantages of agile

- Increased productivity
- Improved project visibility
- Higher software quality
- Higher customer satisfactio
- Less risks
- Faster time-to-market
- Better alignment between IT & business
- More Enjoyable

Advantages of agile

- Increased productivity
  - Motivation
    - People are most productive when they manage themselves
    - People take their own commitment more serious than other people commitments
  - Performance
    - Teams and people are most productive when they aren’t interrupted
    - Teams improve most when they solve their own problems
    - Face-to-face communication is the most productive way for teams to work together
    - Developers can start developing before they know all requirements
    - Less waste
- Improved project visibility
- Higher software quality
- Higher customer satisfaction
- Less risks
- Faster time-to-market
- Better alignment between IT & business
- More Enjoyable

Advantages of agile

- Increased productivity
- Improved project visibility
  - Project status is clear and visible
  - Project problems are clear and visible
  - Information radiators
- Higher software quality
- Higher customer satisfaction
- Less risks
- Faster time-to-market
- Better alignment between IT & business
- More Enjoyable

Advantages of agile

- Increased productivity
- Improved project visibility
- Higher software quality
  - Product is tested throughout the project life cycle
  - Frequent review ensures the customer gets the expected result
  - Continuous attention to technical excellence and good design.
- Higher customer satisfaction
- Less risks
- Faster time-to-market
- Better alignment between IT & business
- More Enjoyable

Advantages of agile

- Increased productivity
- Improved project visibility
- Higher software quality
- Higher customer satisfaction
  - Frequent review ensures the customer gets the expected result
  - Changes are possible
    - Change the features as you learn
  - Potentially shippable product delivery every 1-4 weeks
  - Customer sees the product grow
    - Progress is always clear
    - Trust
  - Higher probability that the product will be marketed
- Less risks
- Faster time-to-market
- Better alignment between IT & business
- More Enjoyable

Advantages of agile

- Increased productivity
- Improved project visibility
- Higher software quality
- Higher customer satisfaction
- Less risks
  - High risks are tackled first
  - Fast feedback
  - Everything is visible
- Faster time-to-market
- Better alignment between IT & business
- More Enjoyable

Advantages of agile

- Increased productivity
- Improved project visibility
- Higher software quality
- Higher customer satisfaction
- Less risks
- Faster time-to-market
    - Possibility of deliverable software after every sprint
    - Faster feedback from the business
- Better alignment between IT & business
- More Enjoyable

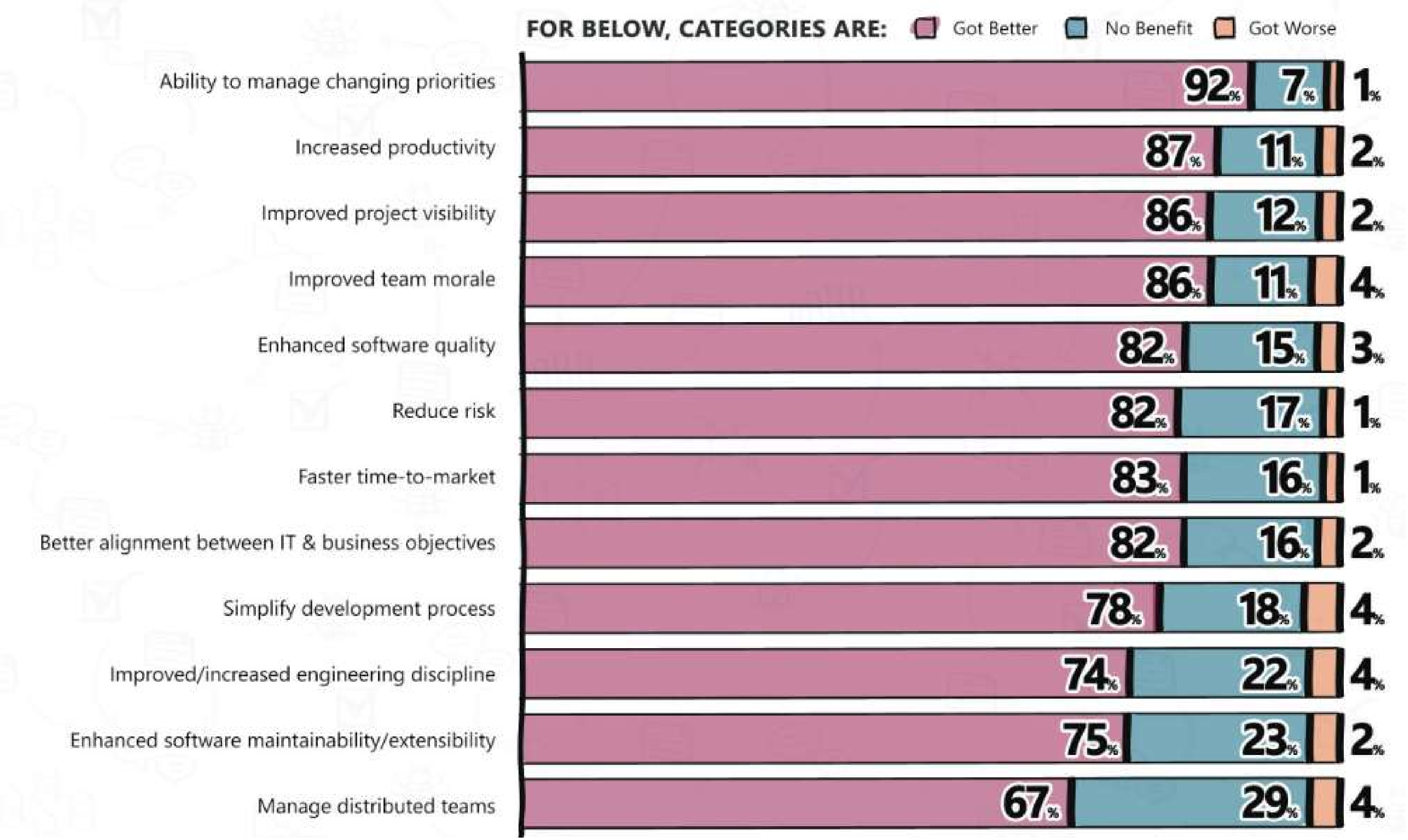
Advantages of agile

- Increased productivity
- Improved project visibility
- Higher software quality
- Higher customer satisfaction
- Less risks
- Faster time-to-market
- Better alignment between IT & business
  - The business and IT work together
  - The business (PO) is in control
- More Enjoyable

Advantages of agile

- Increased productivity
- Improved project visibility
- Higher software quality
- Higher customer satisfaction
- Less risks
- Faster time-to-market
- Better alignment between IT & business
- More Enjoyable
  - Team members learn different disciplines
  - Constant improvement of process and skills
  - Team is in control
  - No more
    - long meetings
    - big documents
    - lengthy status reports
    - long project plans

Benefits obtained from agile



Successful with Scrum

- Business and IT focus on priorities
  - Instead of contract negotiation and deadlines,
- Business and IT focus on communication
  - Instead of large documents and tools
- Develop an agile mindset
  - Instead of following the scrum activities
- Learn,adapt and improve
  - Use small feedback loops

Connecting the parts of knowledge with the wholeness of knowledge

1. Scrum is a dynamic process where the team members are central.
2. In software projects, choreography (between the team members) works better then orchestration (by the project leader)

---

3. **Transcendental consciousness** is the natural experience of pure consciousness, the home of all the laws of nature.
4. **Wholeness moving within itself:** In unity consciousness, one appreciates and enjoys the underlying blissful nature of life even in all the abstract expressions of pure consciousness.