Scaling Agility Explored

Ran Nyman, Ari Tikka

XP2015 27.5.2015

GOSEI

About this session

About

- Root causes and basic assumptions behind Scaling Agile
- For you to evaluate LeSS and SAFe

How to analyze

- Coordination
- Organizational layers of control with the theory by William G. Ouchi
- Flow of work
- Batch size and Queues
- Corporate and business perspective

Speakers

Ran Nyman

CERTIFIED LeSS TRAINER

Scrum Alliance CST CERTIFIED

Ari Tikka

SCALED AGILE FRAMEWORK PROGRAM CONSULTANT

DIGILE | N4S

Why to Scale Agile?

Don't!

Adding more people just makes you slower.

-- one of the directors of SAGE program 1950's

-- Frederick Brooks at Mythical man-month, 1975

Still want to scale up?

Jersey Shore www.joiseyshowaa.com

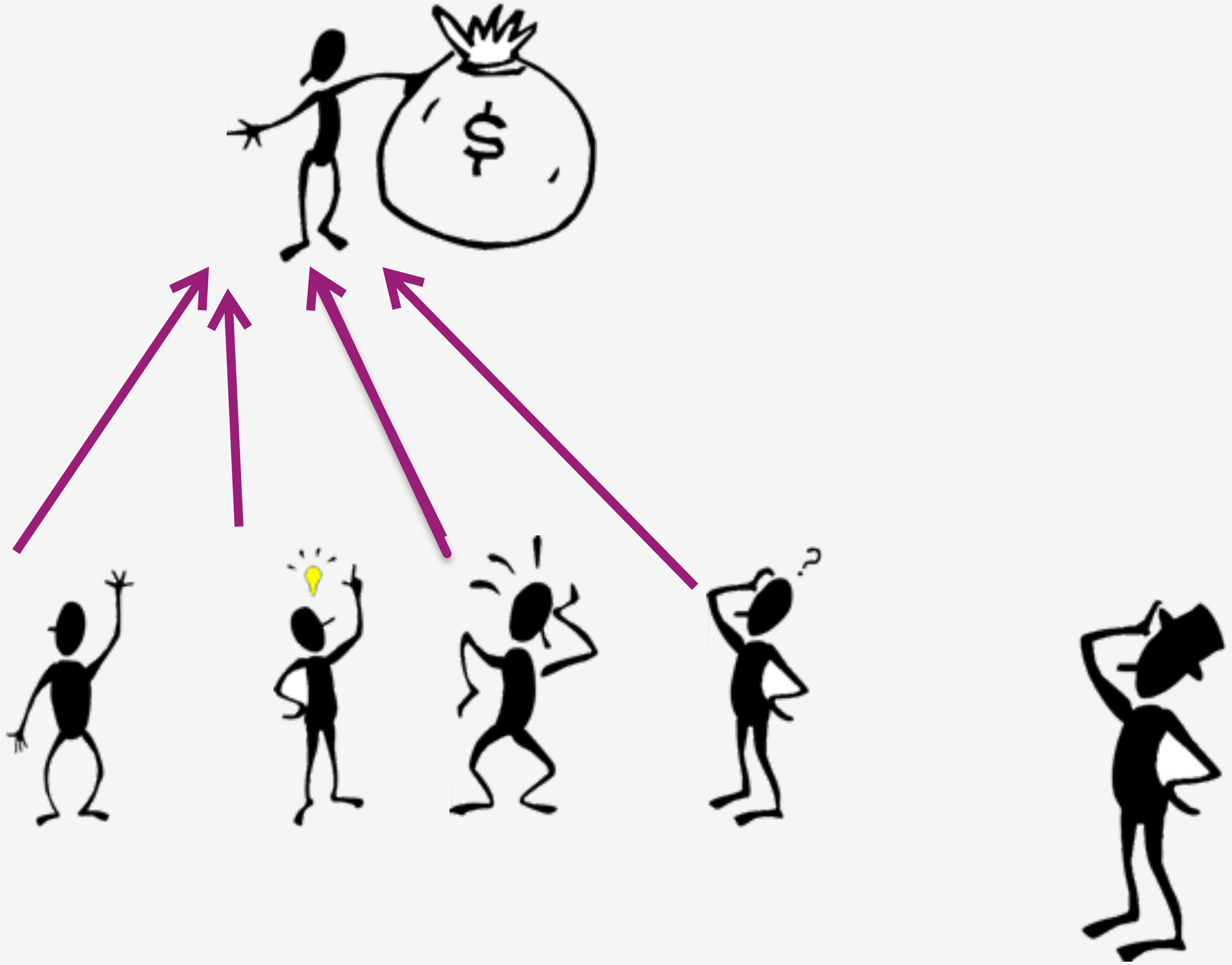
We have grown big, slow and wasteful.

We are creating complex big products. We don’t want to become slow and wasteful.

How do you end up slow and wasteful?

Common sense and fashionable solutions

In the beginning

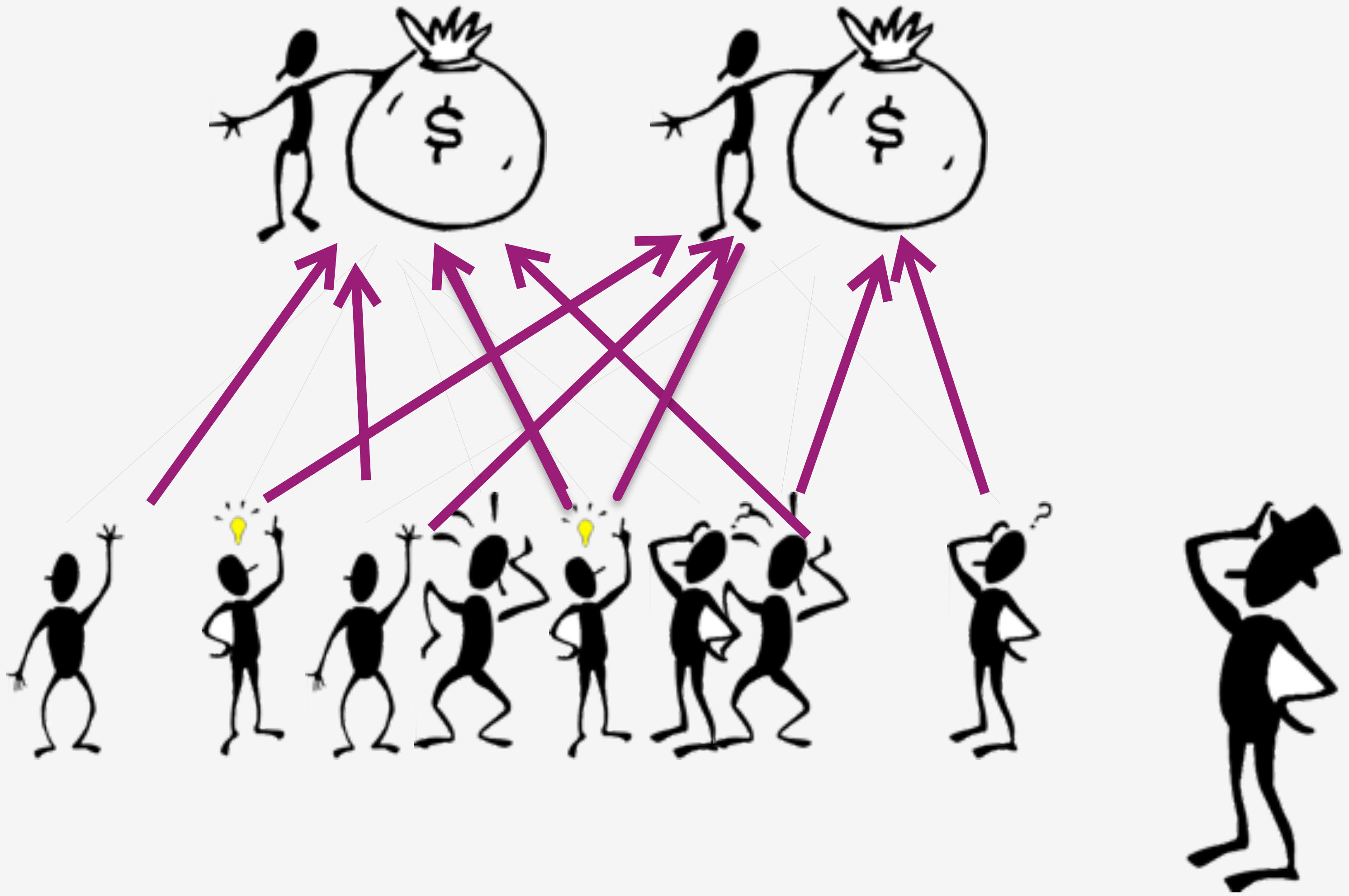


“Hey, We have business! And it is growing!”

“People just find their roles.”

“Specialists are irreplaceable. We need to optimize their individual performance.”

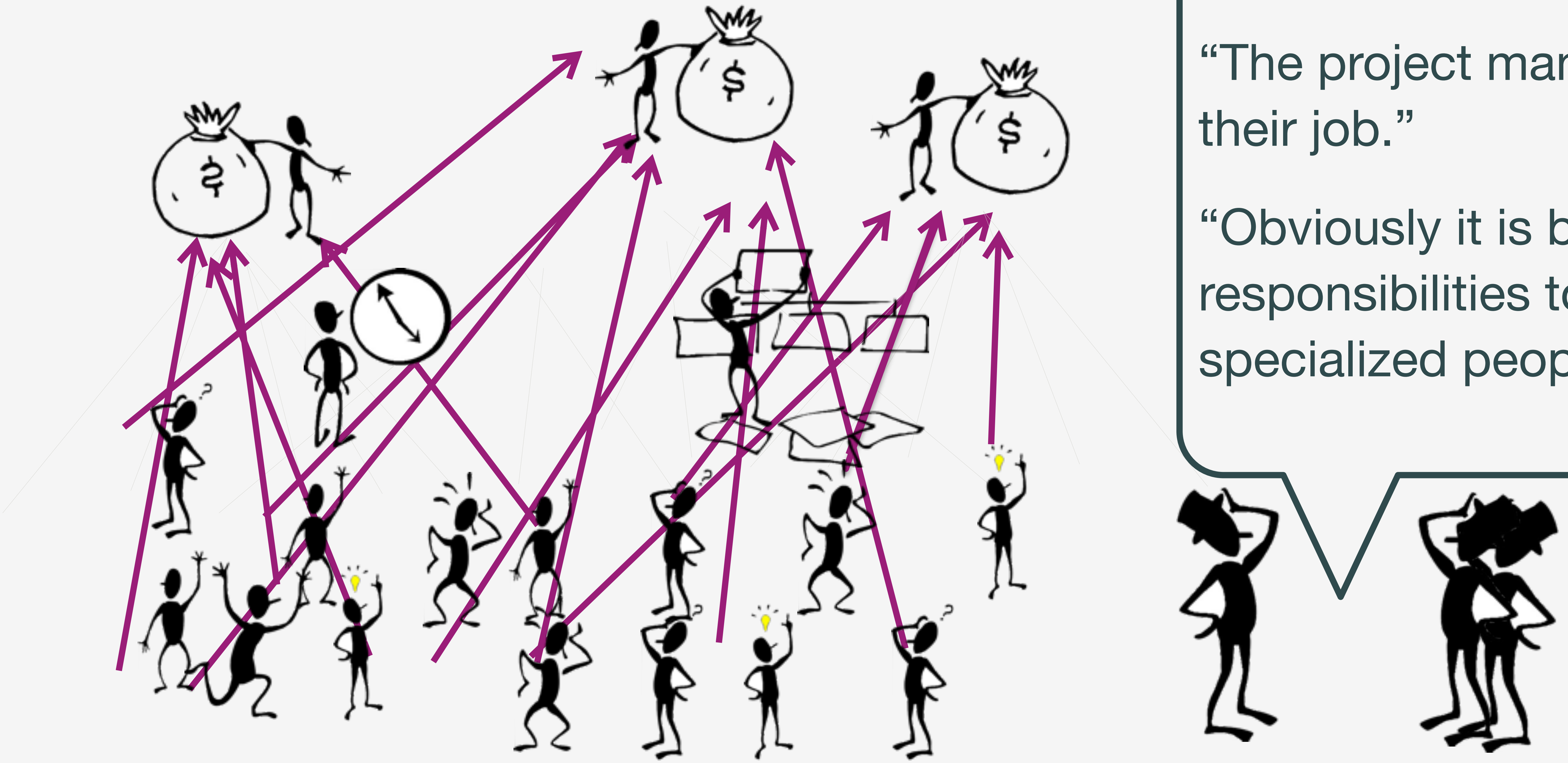
Growing the using common sense



“It starts to get messy. We need someone to look after things.”

“Lets hire a coordination specialist - the project manager.”

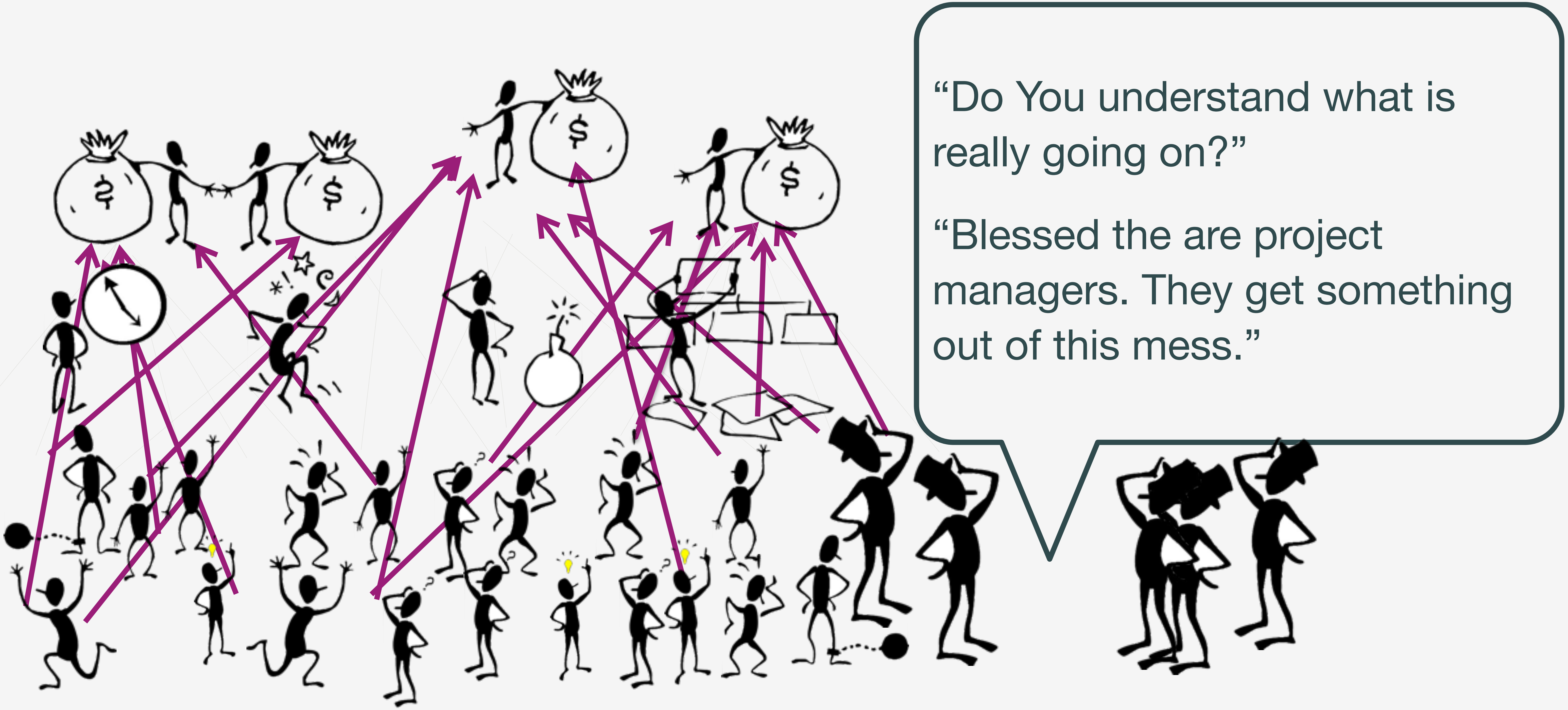
Growth continues



“The project managers really do their job.”

“Obviously it is best to give responsibilities to the specialized people.”

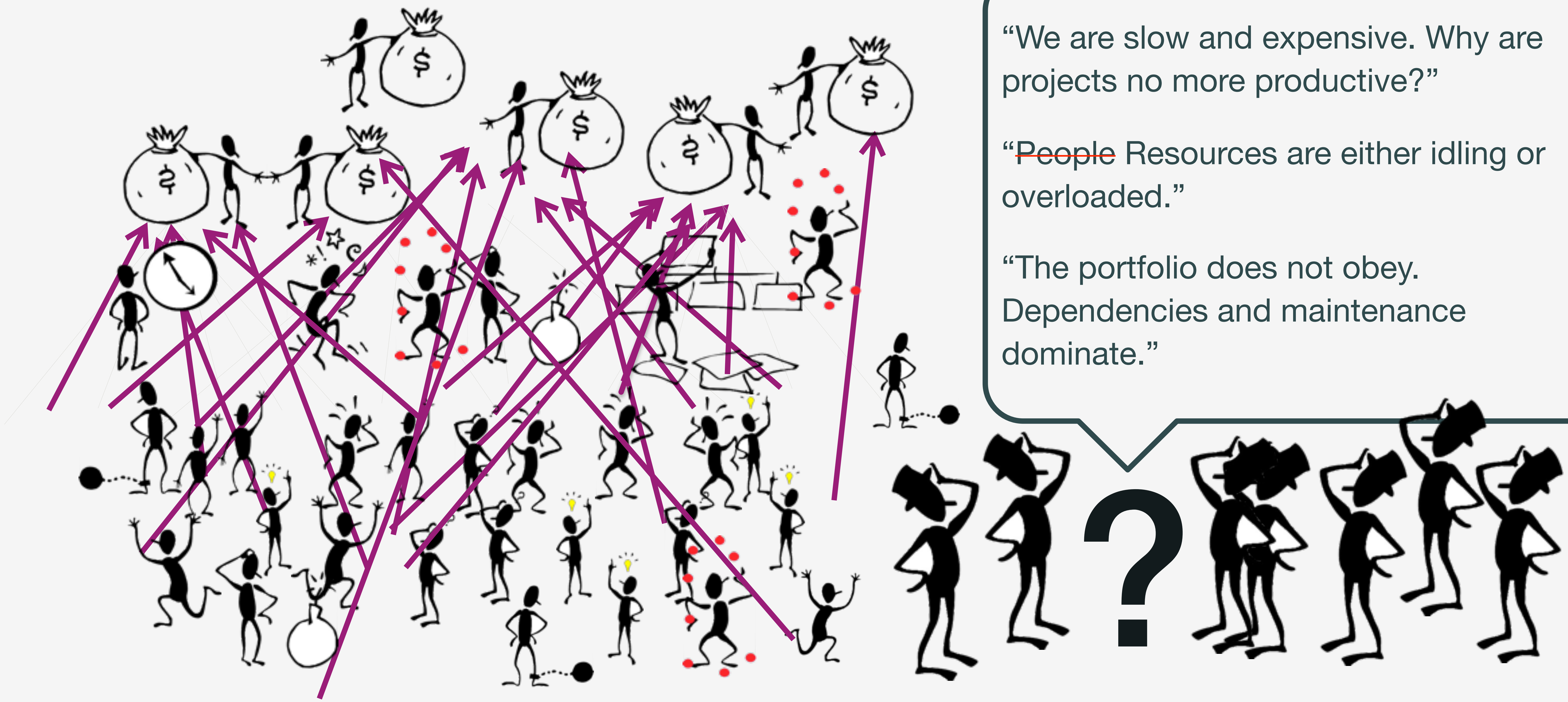
The coordinators become the heroes



“Do You understand what is really going on?”

“Blessed the are project managers. They get something out of this mess.”

But… too much to be coordinated

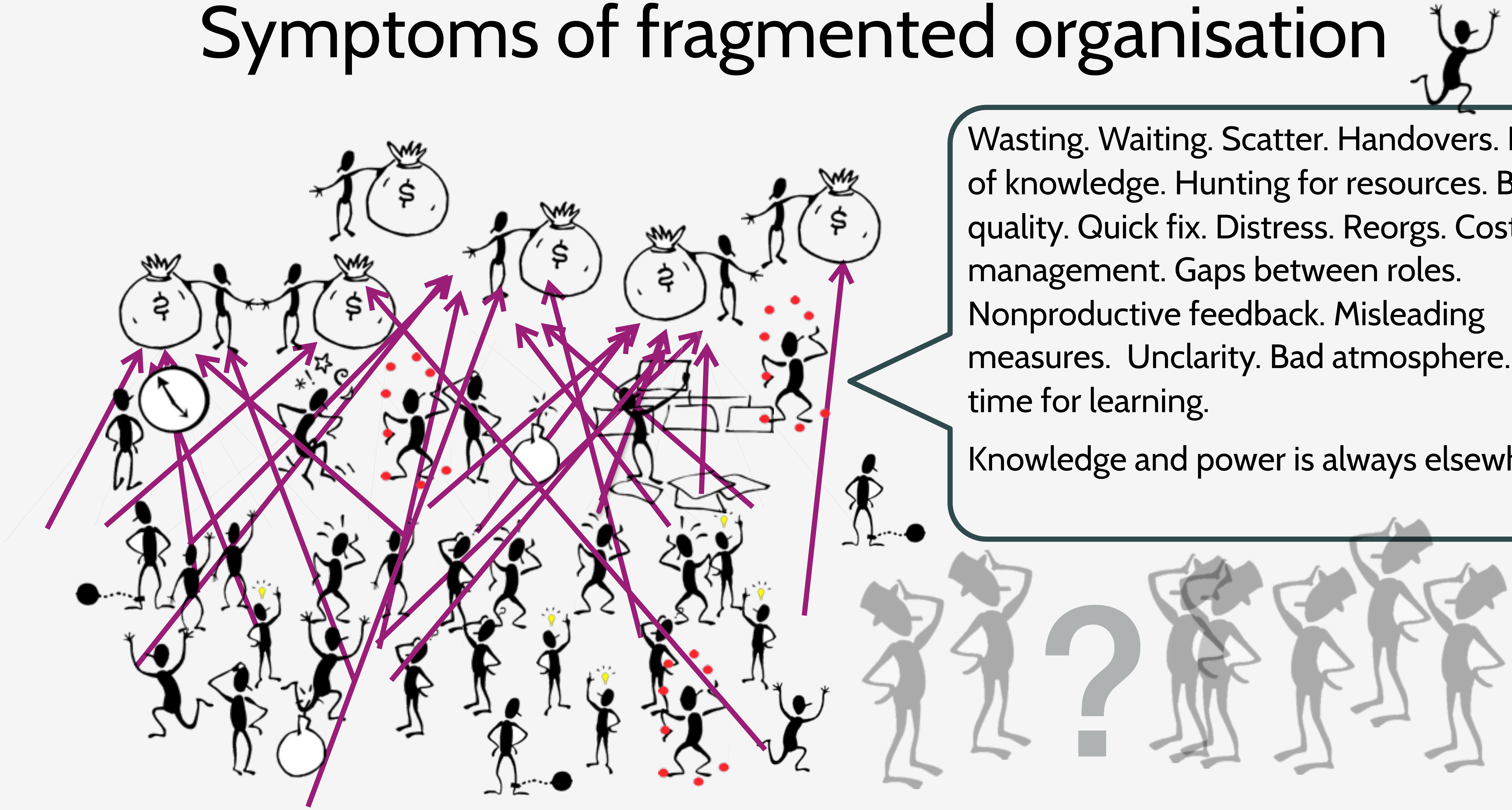


“We are slow and expensive. Why are projects no more productive?”

“~~People~~ Resources are either idling or overloaded.”

“The portfolio does not obey. Dependencies and maintenance dominate.”

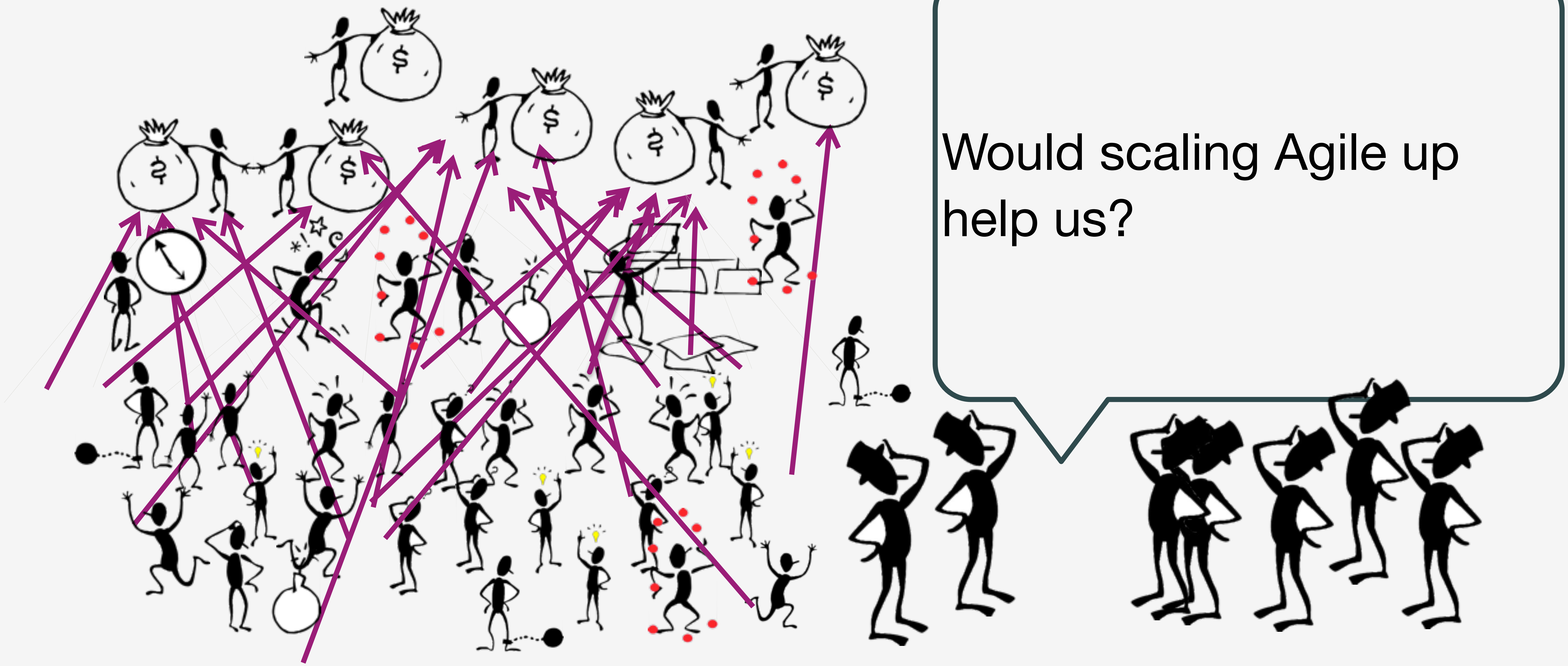
Symptoms of fragmented organisation



Wasting. Waiting. Scatter. Handovers. Loss of knowledge. Hunting for resources. Bad quality. Quick fix. Distress. Reorgs. Cost management. Gaps between roles. Nonproductive feedback. Misleading measures. Unclarity. Bad atmosphere. No time for learning.

Knowledge and power is always elsewhere!

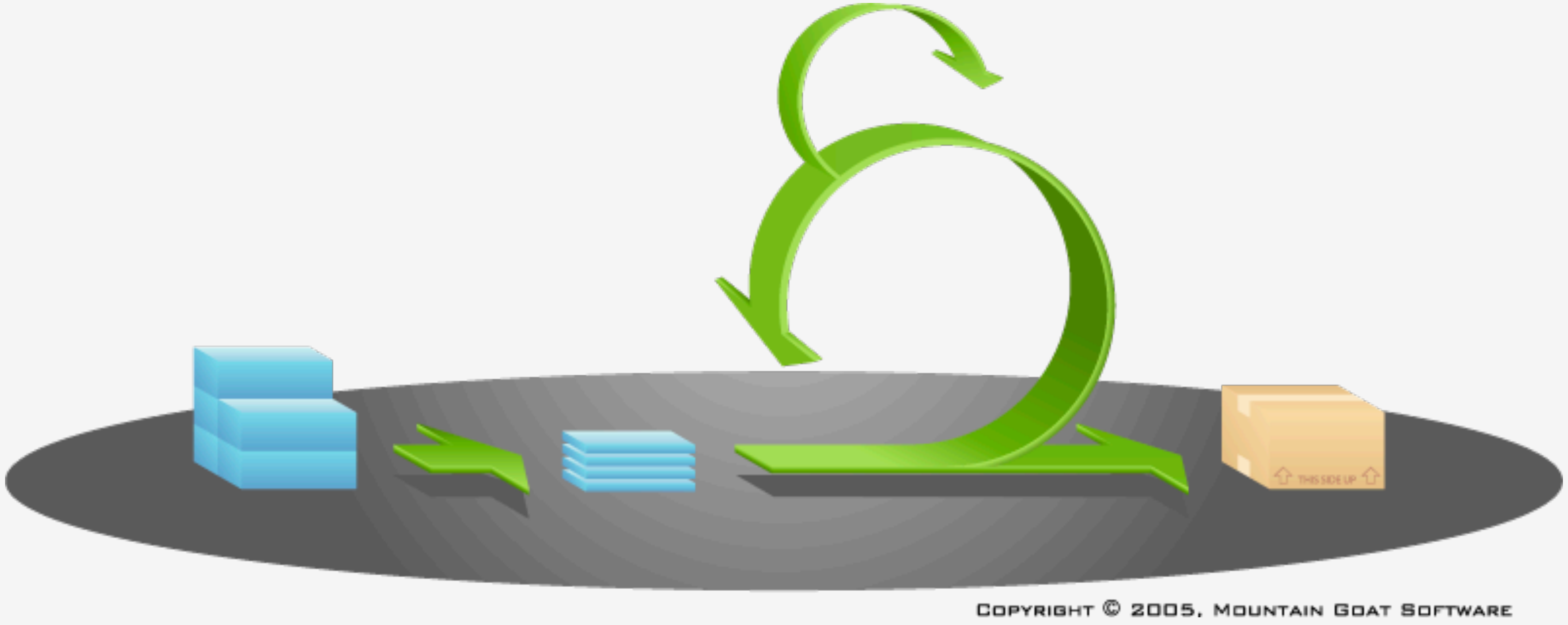
We are slow and wasteful!



Would scaling Agile up help us?

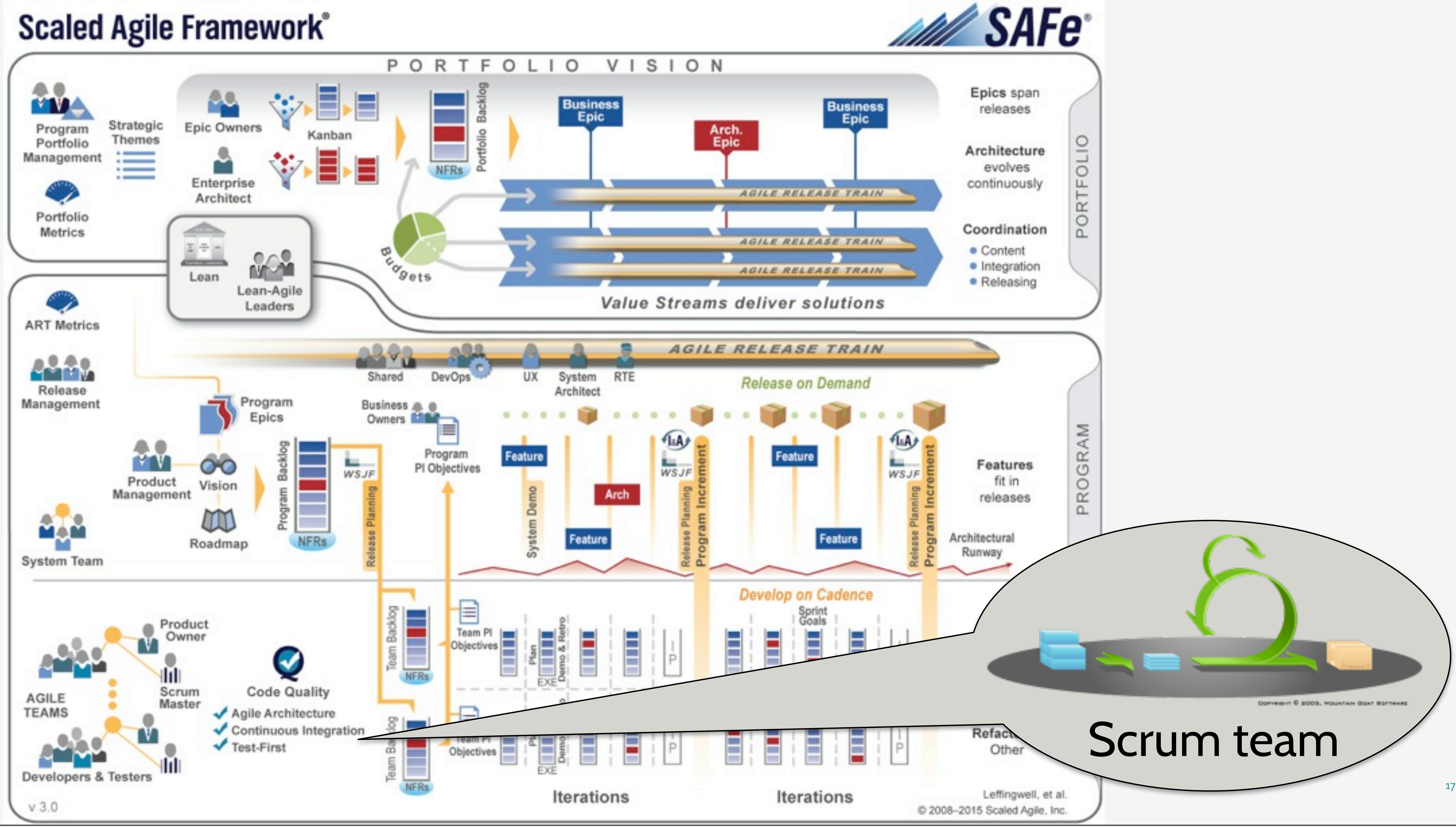
What to DO?

Scrum works for one team



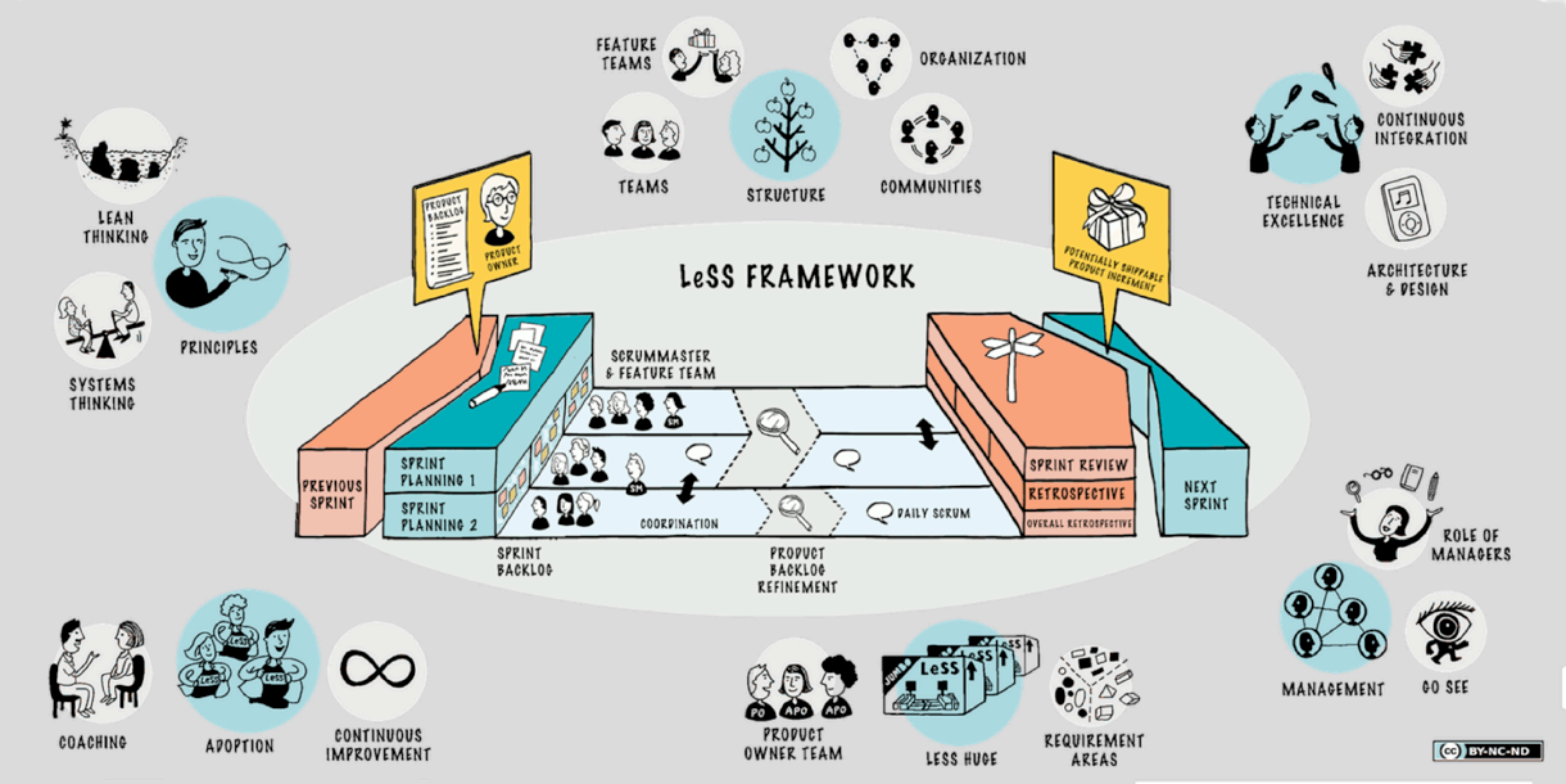
Copyright © 2005, Mountain Goat Software

Program Execution

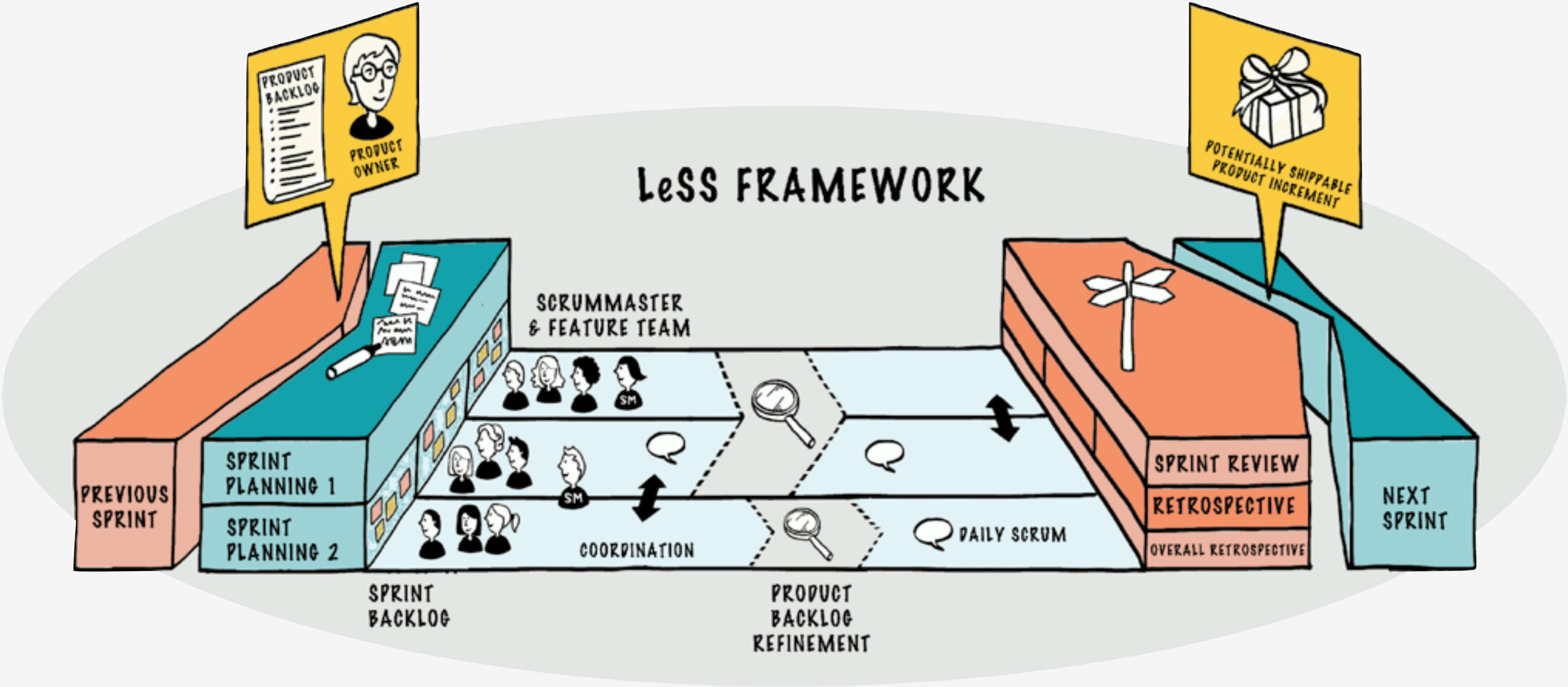


Scrum team

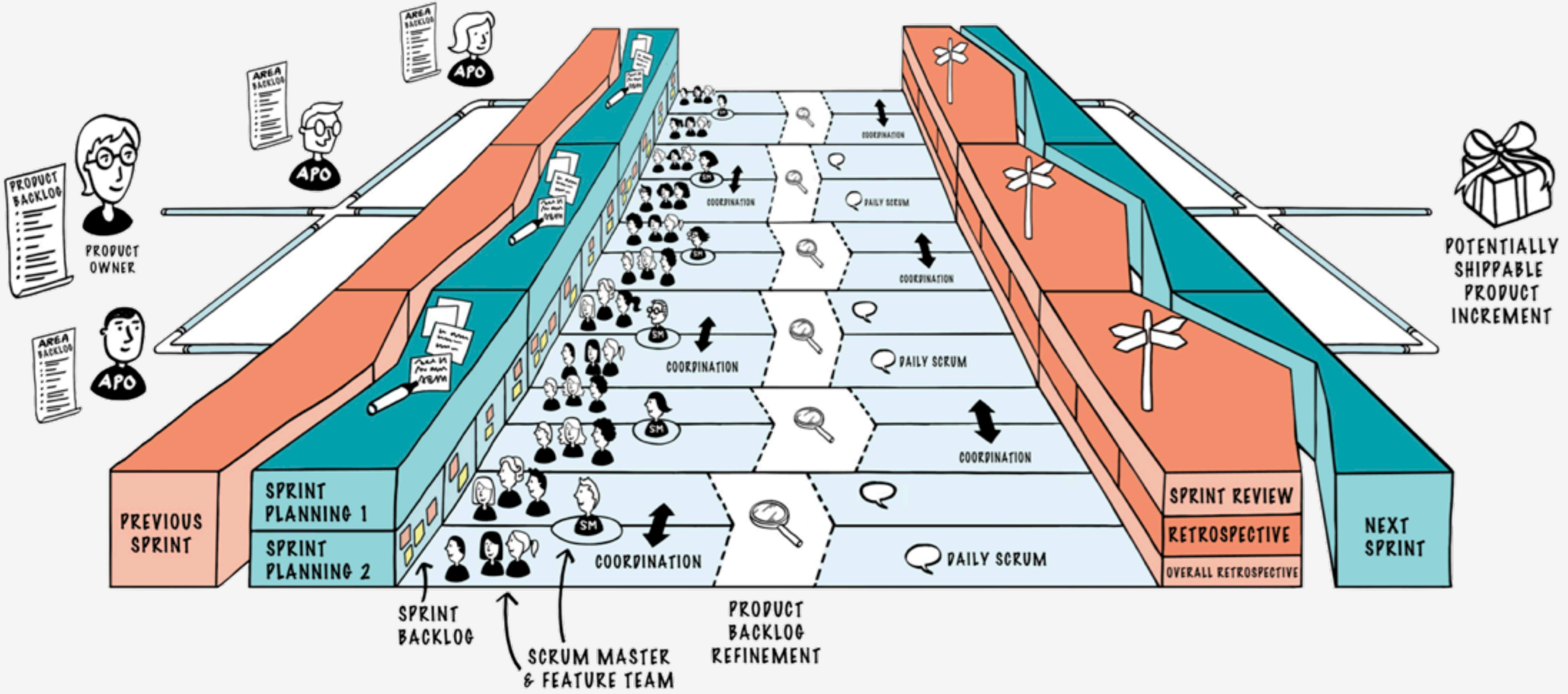
Customer-centric learning



LeSS Framework



LeSS Huge


PRODUCT BACKLOG
PRODUCT OWNER
AREA BACKLOG
APO
AREA BACKLOG
APO
AREA BACKLOG
APO
AREA BACKLOG
APO
PREVIOUS SPRINT
SPRINT PLANNING 1
SPRINT PLANNING 2
SM
COORDINATION
DAILY SCRUM
SPRINT REVIEW
RETROSPECTIVE
OVERALL RETROSPECTIVE
NEXT SPRINT
POTENTIALLY SHIPPABLE PRODUCT INCREMENT
SPRINT BACKLOG
SCRUM MASTER & FEATURE TEAM
PRODUCT BACKLOG REFINEMENT

Control Systems and Coordination

Control Systems in organizations by William G. Ouchi

Market system

Measure Input (€) and Output (€). Contractual between parties. Exact contract!

Bureaucratic system

Written rules and processes. E.g. Employment agreement and supervision.

Clan system

Informal value based rules that allow innovation and collaboration. Only this works for <u>unique, interdependent or ambiguous</u> task. E.g. SW Development

William G. Ouchi

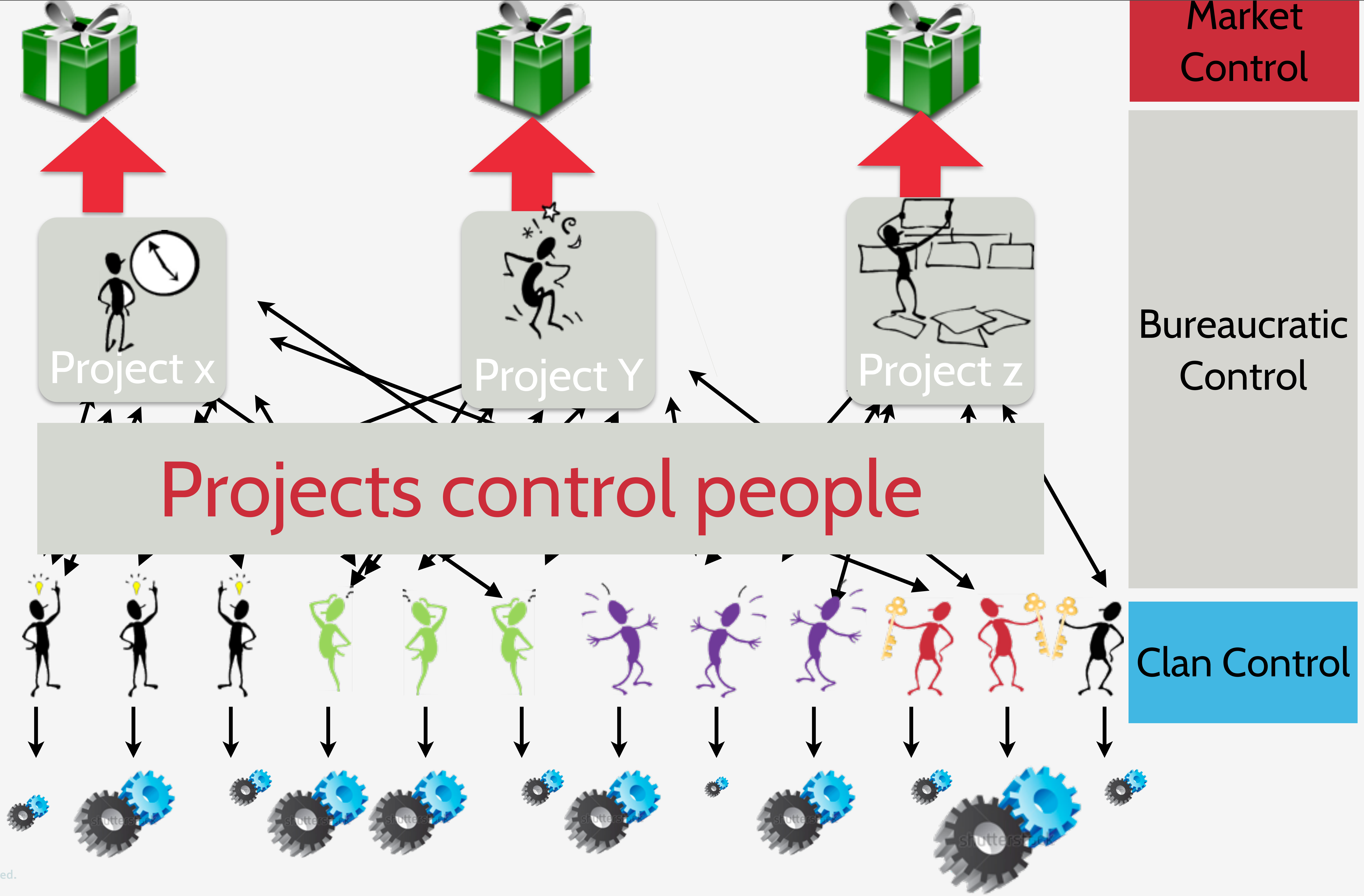
Inventor of management control mechanisms

Inventor of motivation Theory Z

- Addition to well know Theory X and Y

Influenced by Japanese management style

Projects Coordinate Peoples Time and Technology Dependencies



Market Control

Bureaucratic Control

Clan Control

Project x

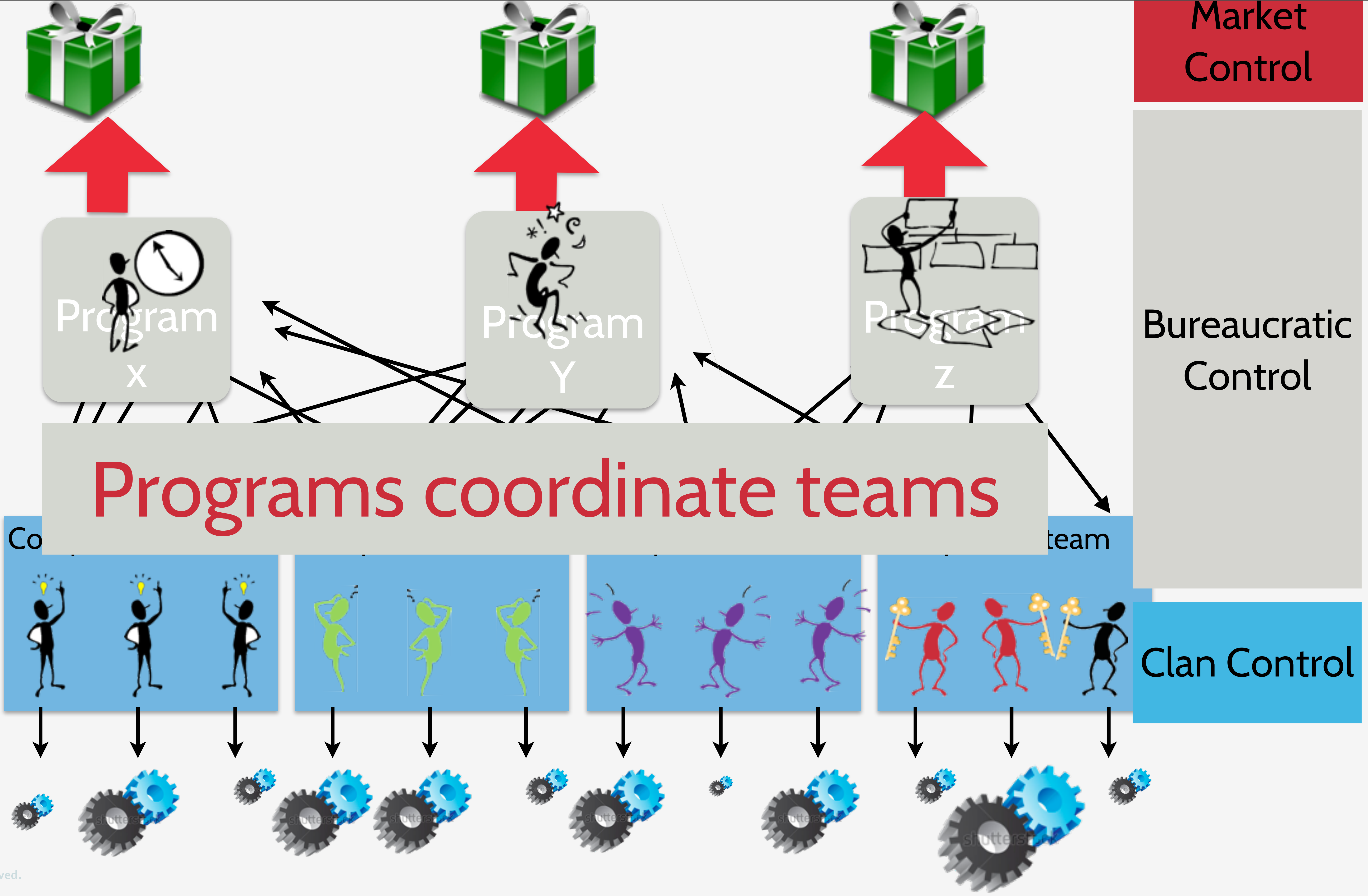
Project Y

Project z

Projects control people

Programs Coordinate Teams and Technology Dependencies

Programs coordinate teams

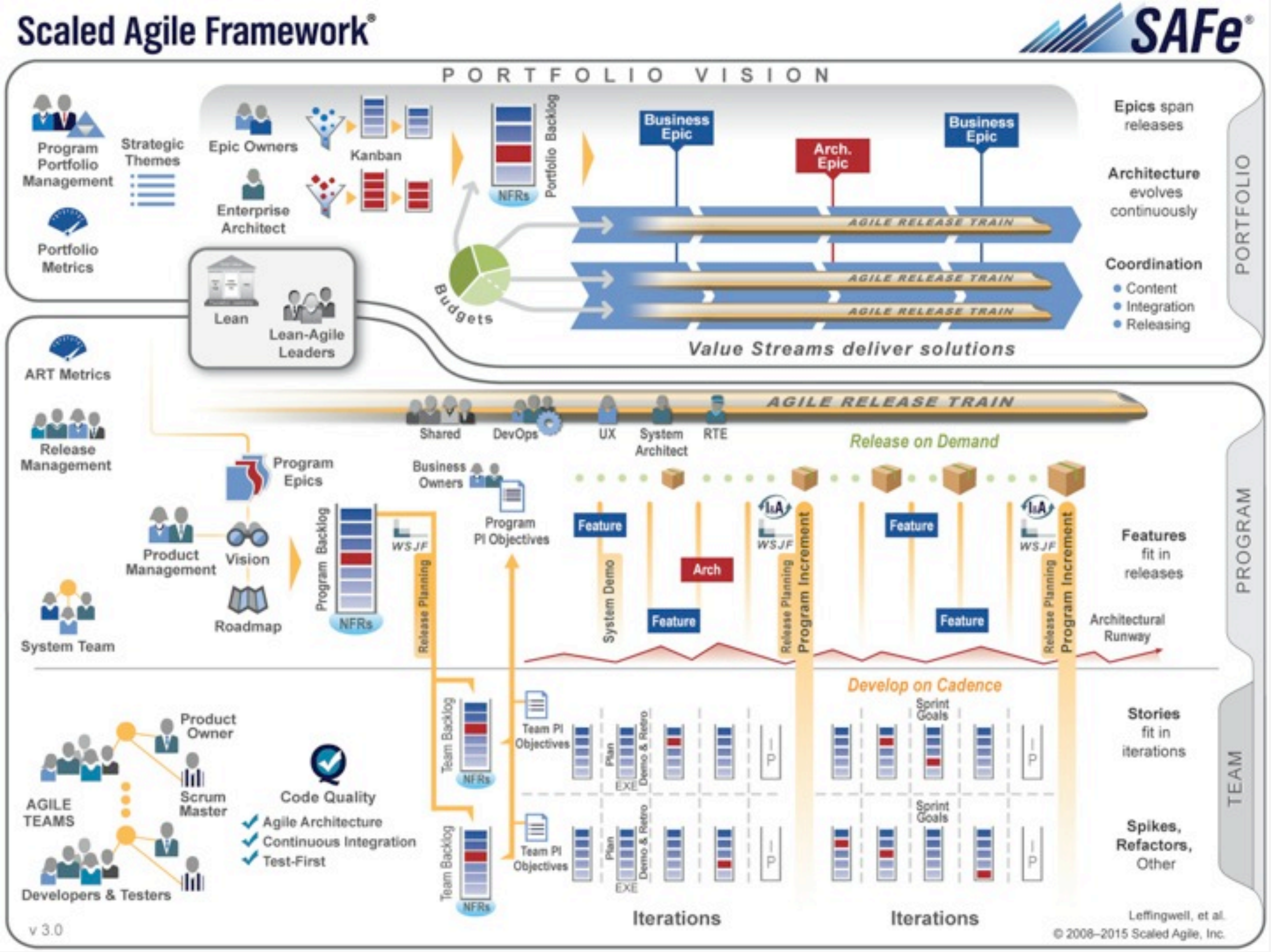


Control by SAFe

Market System (€)

Bureaucratic System (process, written rules,role descriptions)

Clan System (social rules)

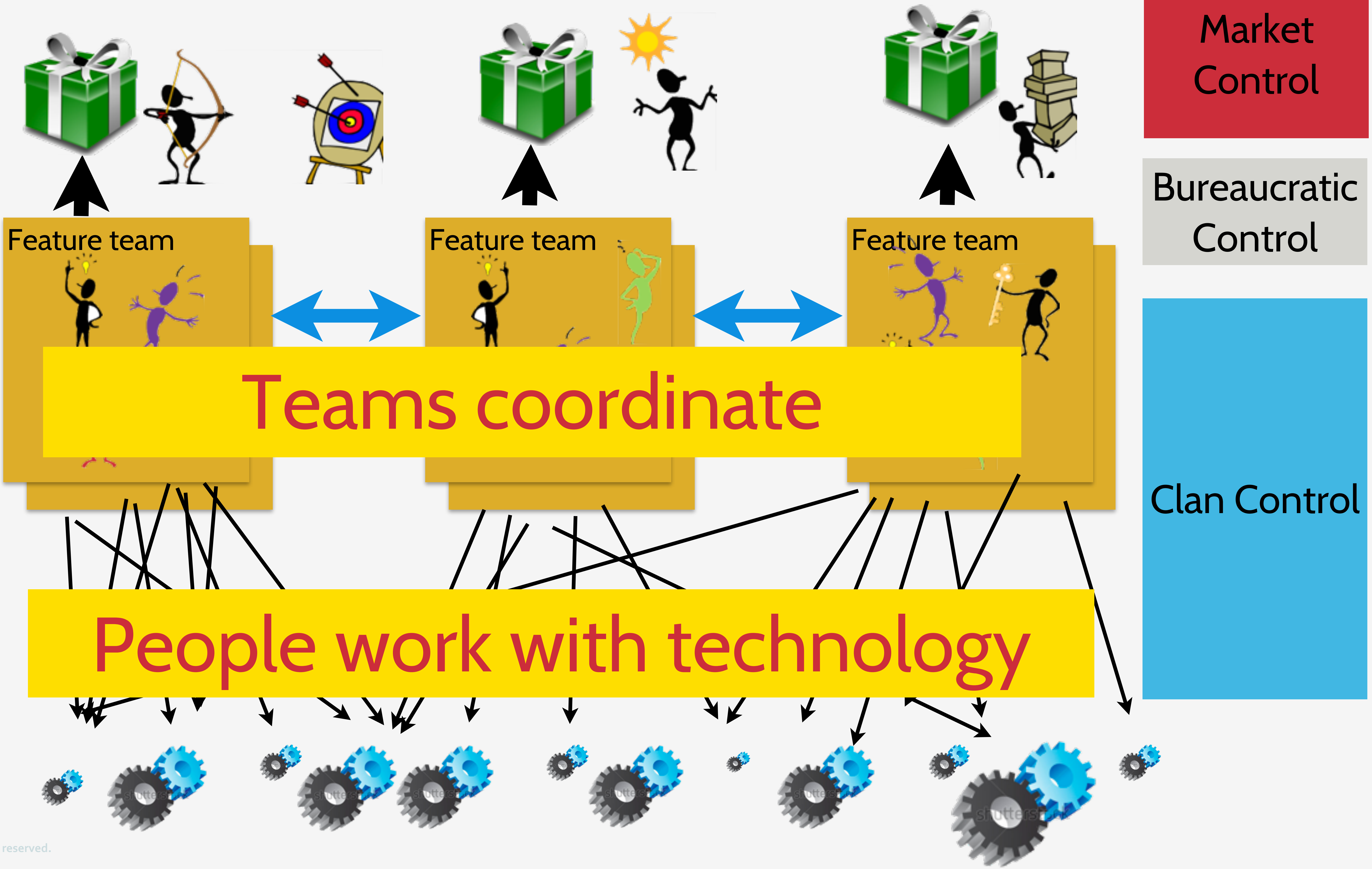


What changes	Risk or challenge
Release trains and 3-month cadence instead of parallel projects.	The parallel projects existed for a reason, which is not yet solved. Still need for substantial planning, because the underlying organizational design is unchanged.
From time based resource planning to team output estimation. Improved communication by all hands release planning meeting.	The amount of dependencies, and queuing them for solution remains a challenge and results in branching and late integration. Contract game remains for planning needs. Welcoming teams to middle management.
Training and consultation for Lean-Agile best practices.	Culture follows structure. Focus in coordination, not value adding work. Thinking and communicating in organizations differs from what is actually happening. *)

*) Mats Alvesson and André Spicer: "A Stupidity-Based Theory of Organizations", Journal of Management Studies 49:7, November 2012.
André SPICER: 2013 "Shooting the shit: the role of bullshit in organisations" M@n@gement, 16(5), 653-666, Cass Business School, CU London.

What is not changing	Risk or challenge
The numerous middle management roles are renamed. As before, scarce resources are moved from teams to management (e.g. UI design, Architects).	The change does not happen. The change is not useful.
Corporate layers of power and control legitimized to be Agile.	No real change. No business Agility developed. Business decides, programs execute. Contract game with business.
Little emphasis for structural change from functional to feature teams.	Technical capability and competence limit the effectiveness of the change.

Teams Coordinate Dependencies and Technology



Market Control

Bureaucratic Control

Clan Control

Feature team

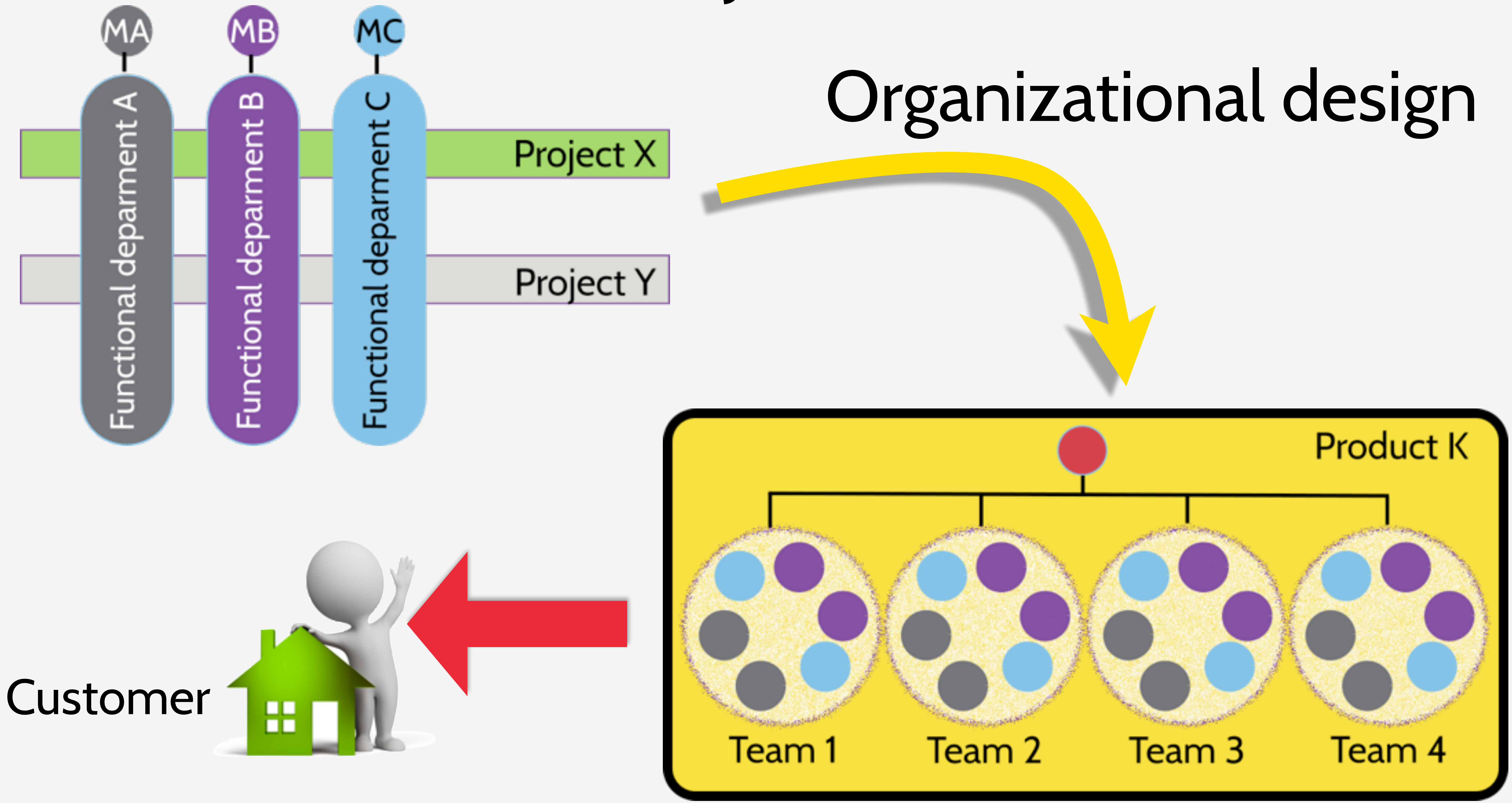
Feature team

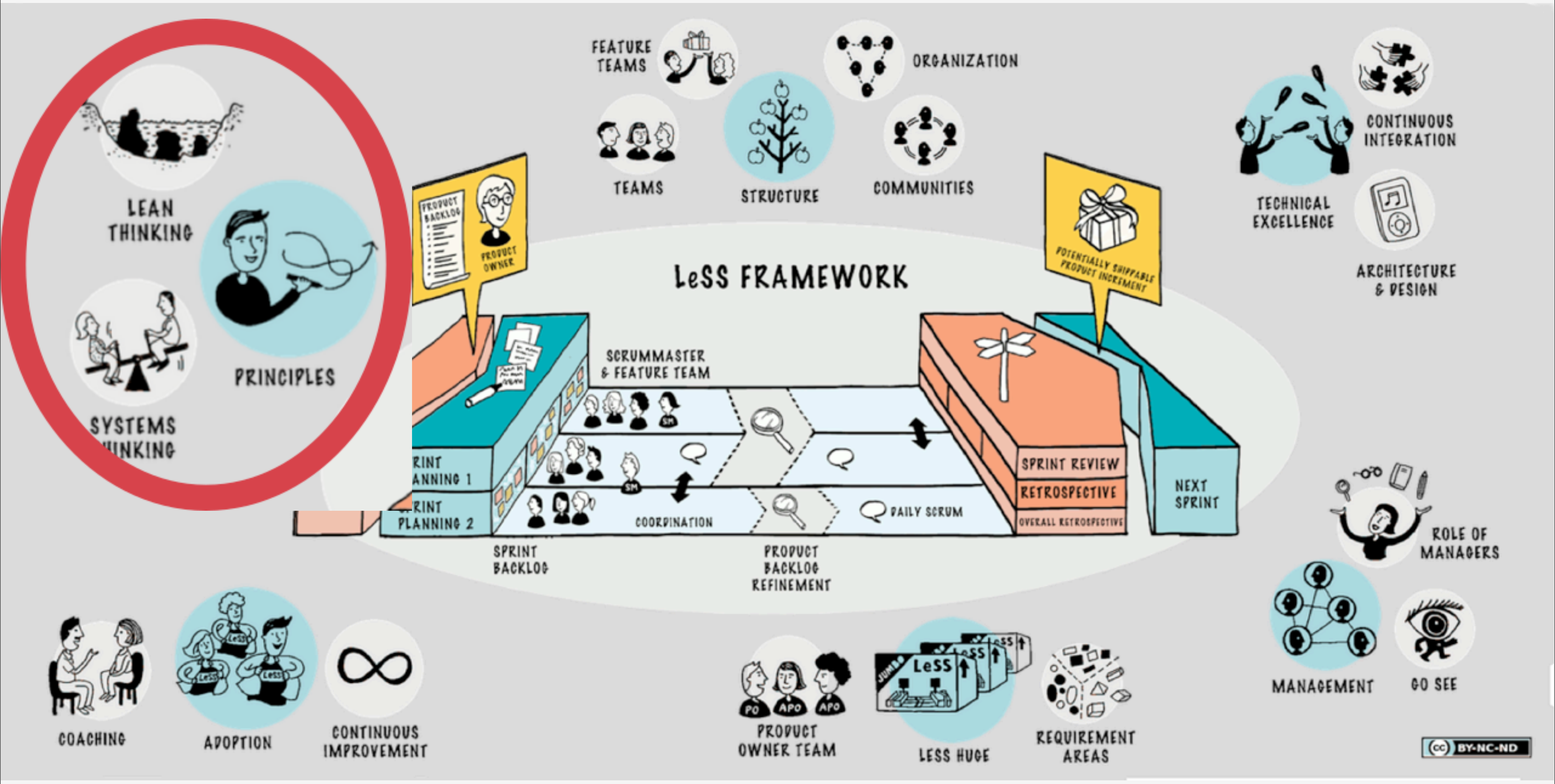
Feature team

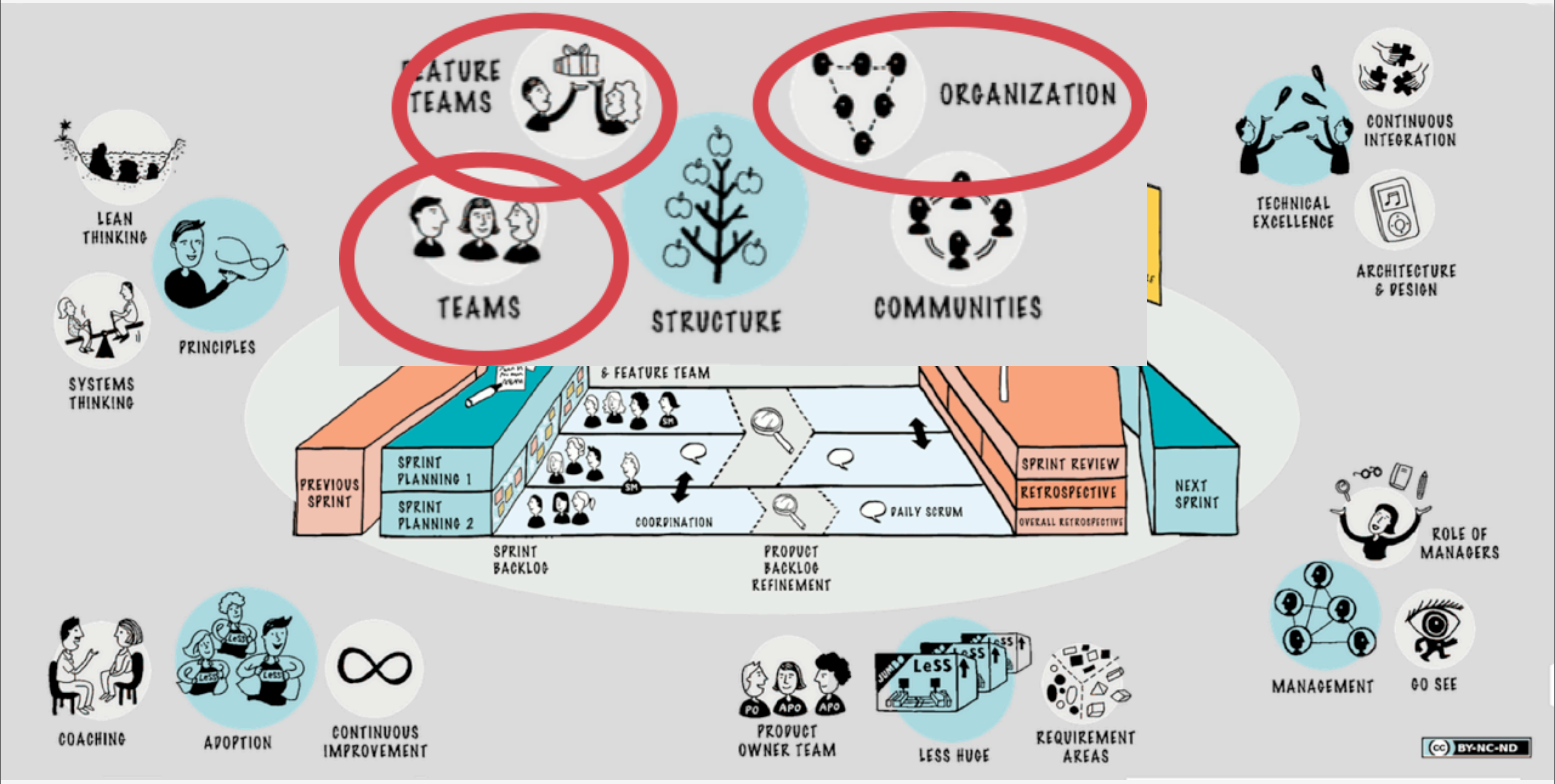
Teams coordinate

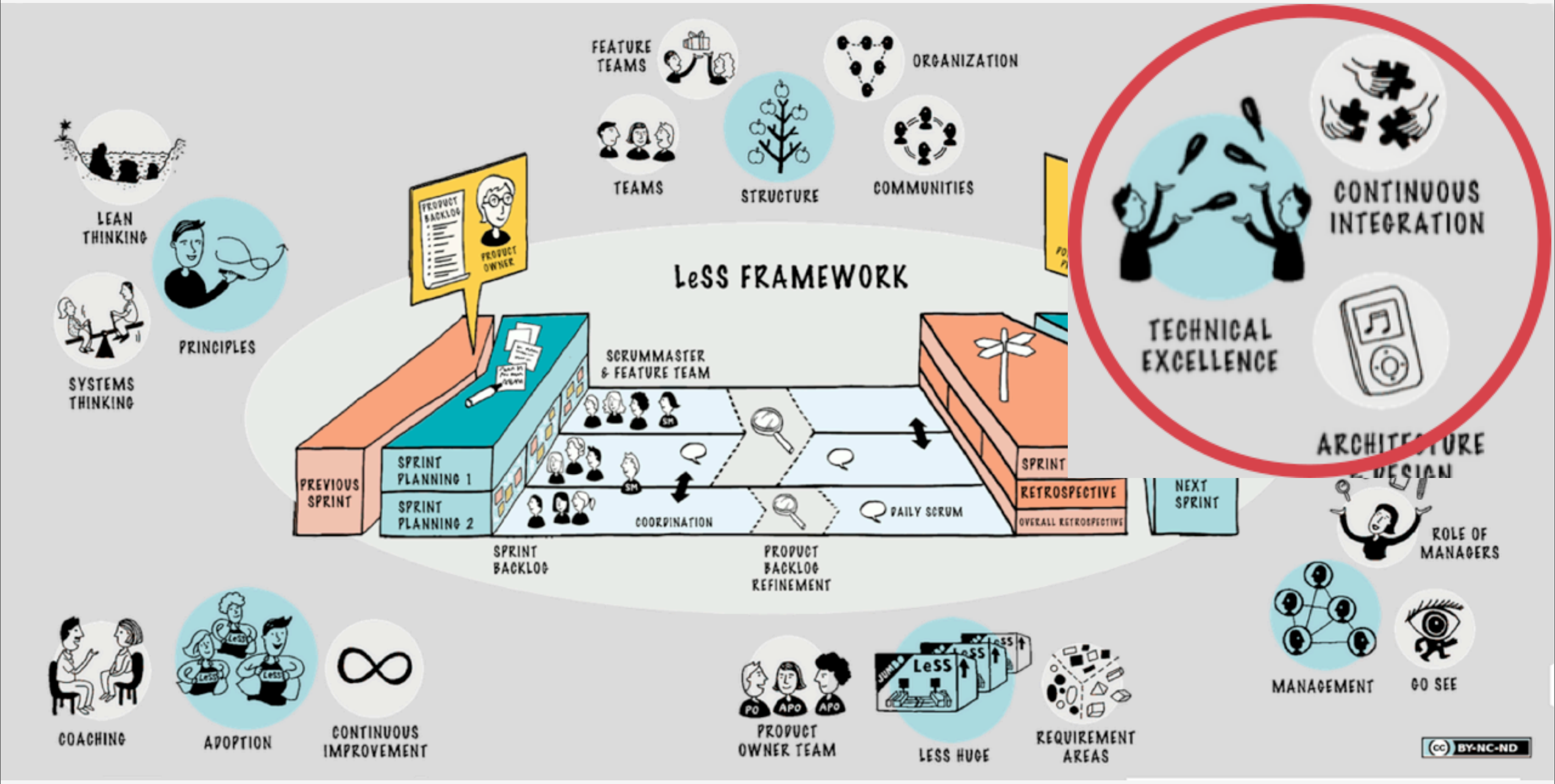
People work with technology

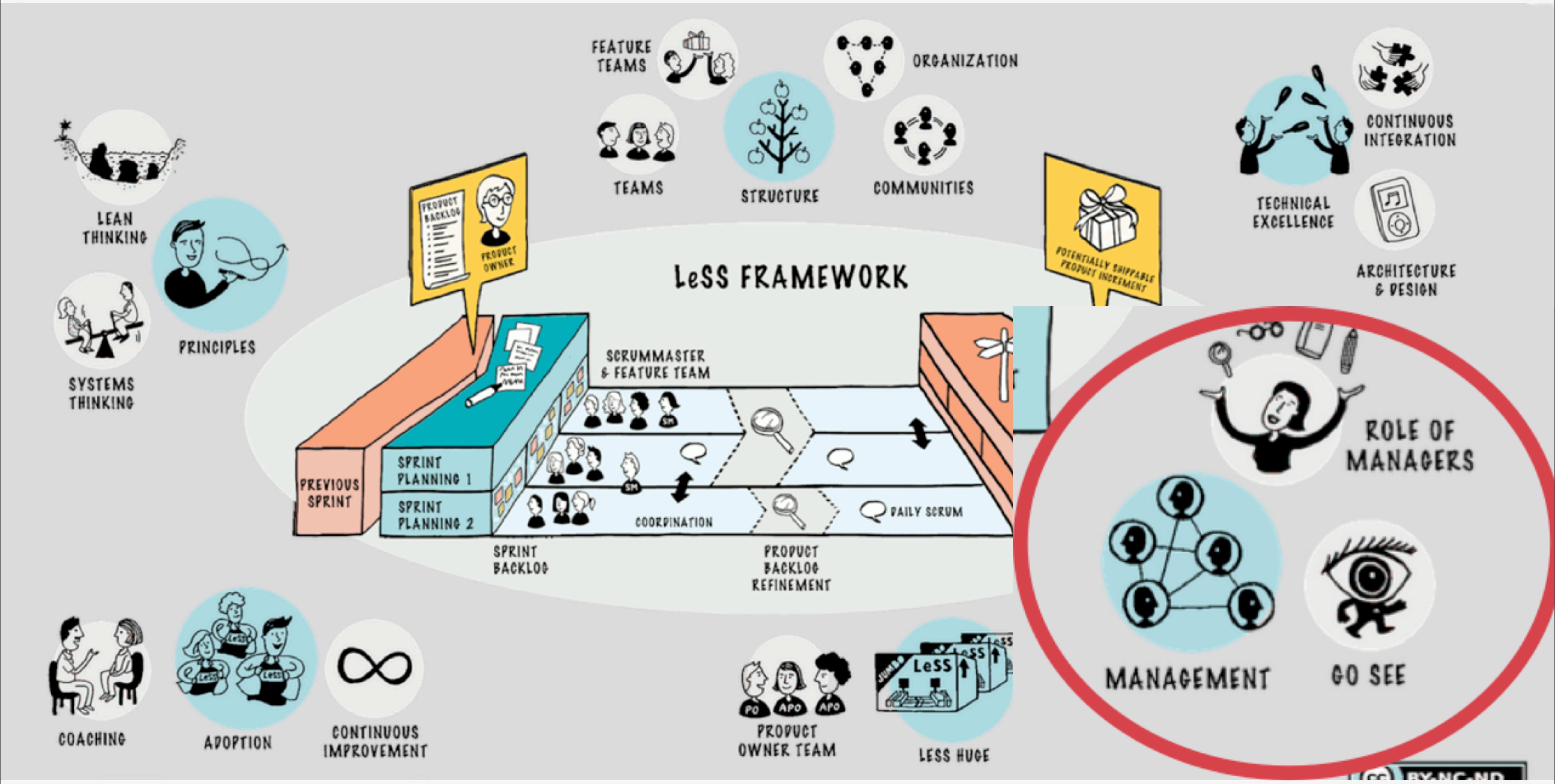
Focus from Projects to Customer




LEAN THINKING
PRINCIPLES
SYSTEMS THINKING
FEATURE TEAMS
TEAMS
STRUCTURE
ORGANIZATION
COMMUNITIES
CONTINUOUS INTEGRATION
TECHNICAL EXCELLENCE
ARCHITECTURE & DESIGN
PRODUCT BACKLOG
PRODUCT OWNER
LeSS FRAMEWORK
POTENTIALLY SHIPPABLE PRODUCT INCREMENT
SCRUMMASTER & FEATURE TEAM
SM
SM
RINT ANNING 1
RINT PLANNING 2
COORDINATION
DAILY SCRUM
SPRINT REVIEW
RETROSPECTIVE
OVERALL RETROSPECTIVE
NEXT SPRINT
SPRINT BACKLOG
PRODUCT BACKLOG REFINEMENT
ROLE OF MANAGERS
COACHING
ADOPTION
CONTINUOUS IMPROVEMENT
PO
APO
APO
PRODUCT OWNER TEAM
LeSS
LESS HUGE
REQUIREMENT AREAS
MANAGEMENT
GO SEE
BY-NC-ND


EATURE TEAMS
ORGANIZATION
LEAN THINKING
PRINCIPLES
SYSTEMS THINKING
TEAMS
STRUCTURE
COMMUNITIES
CONTINUOUS INTEGRATION
TECHNICAL EXCELLENCE
ARCHITECTURE & DESIGN
& FEATURE TEAM
PREVIOUS SPRINT
SPRINT PLANNING 1
SPRINT PLANNING 2
SM
SM
COORDINATION
DAILY SCRUM
SPRINT REVIEW
RETROSPECTIVE
OVERALL RETROSPECTIVE
NEXT SPRINT
SPRINT BACKLOG
PRODUCT BACKLOG REFINEMENT
ROLE OF MANAGERS
MANAGEMENT
GO SEE
COACHING
ADOPTION
LeSS
LeSS
LeSS
CONTINUOUS IMPROVEMENT
PO
APO
APO
PRODUCT OWNER TEAM
LeSS
LeSS
LeSS
LESS HUGE
REQUIREMENT AREAS
BY-NC-ND


LeSS FRAMEWORK
FEATURE TEAMS
ORGANIZATION
TEAMS
STRUCTURE
COMMUNITIES
LEAN THINKING
PRINCIPLES
SYSTEMS THINKING
PRODUCT BACKLOG
PRODUCT OWNER
SCRUMMASTER & FEATURE TEAM
SM
SM
PREVIOUS SPRINT
SPRINT PLANNING 1
SPRINT PLANNING 2
COORDINATION
DAILY SCRUM
SPRINT BACKLOG
PRODUCT BACKLOG REFINEMENT
SPRINT
RETROSPECTIVE
OVERALL RETROSPECTIVE
NEXT SPRINT
CONTINUOUS INTEGRATION
TECHNICAL EXCELLENCE
ARCHITECTURE
DESIGN
ROLE OF MANAGERS
MANAGEMENT
GO SEE
COACHING
LeSS
LeSS
LeSS
ADOPTION
CONTINUOUS IMPROVEMENT
PO
APO
APO
PRODUCT OWNER TEAM
JUMBO
LeSS
LeSS
LeSS
LESS HUGE
REQUIREMENT AREAS
cc BY-NC-ND


LeSS FRAMEWORK
FEATURE TEAMS
TEAMS
STRUCTURE
ORGANIZATION
COMMUNITIES
LEAN THINKING
PRINCIPLES
SYSTEMS THINKING
PRODUCT BACKLOG
PRODUCT OWNER
POTENTIALLY SHIPPABLE PRODUCT INCREMENT
TECHNICAL EXCELLENCE
CONTINUOUS INTEGRATION
ARCHITECTURE & DESIGN
SCRUMMASTER & FEATURE TEAM
SM
PREVIOUS SPRINT
SPRINT PLANNING 1
SPRINT PLANNING 2
COORDINATION
DAILY SCRUM
SPRINT BACKLOG
PRODUCT BACKLOG REFINEMENT
ROLE OF MANAGERS
MANAGEMENT
GO SEE
COACHING
ADOPTION
LeSS
CONTINUOUS IMPROVEMENT
PO
APO
APO
PRODUCT OWNER TEAM
JUMBO
LeSS
LESS HUGE
BY-NC-ND

Nooooooo!

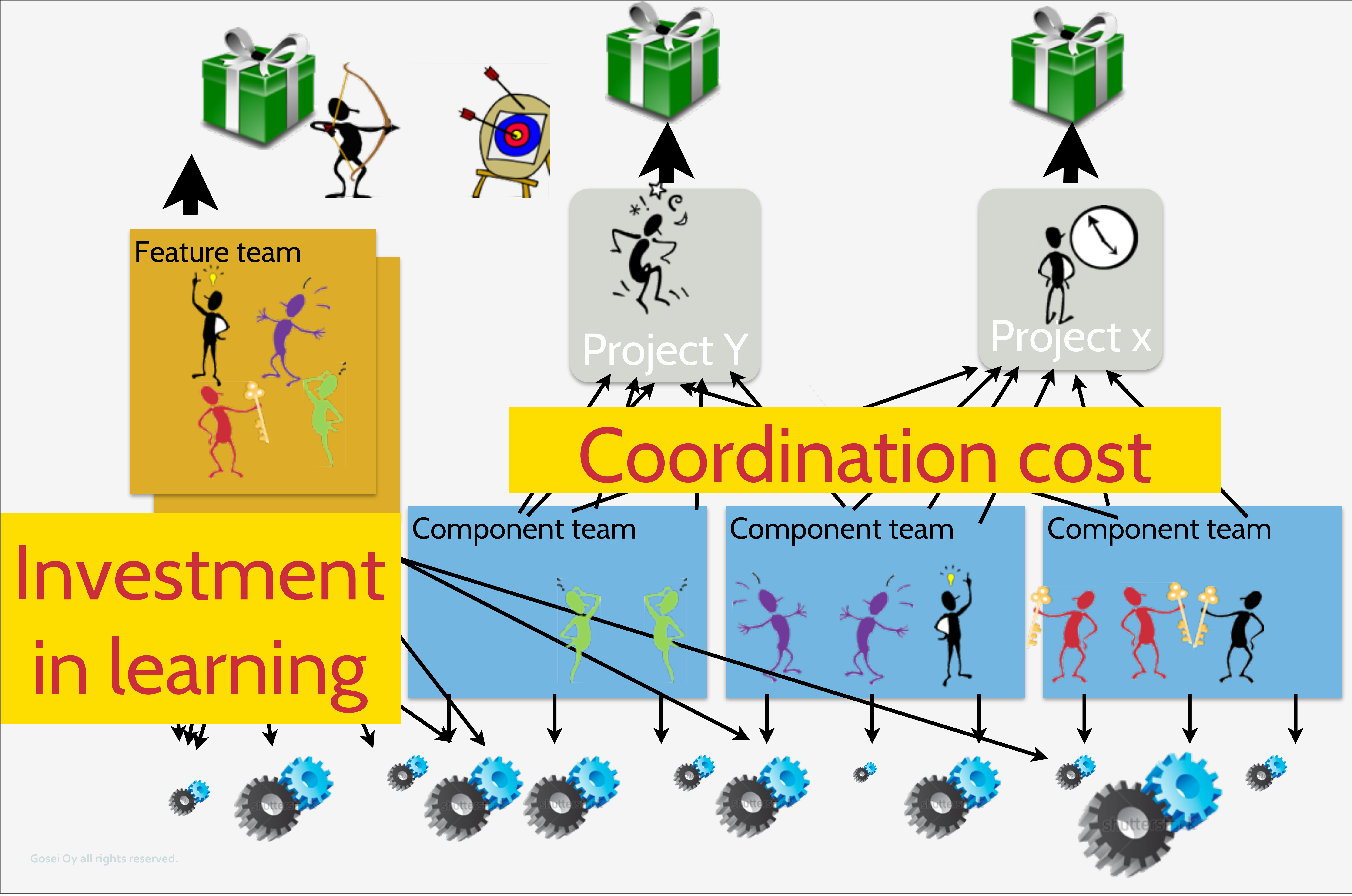
We can not change everything.

Your Fear is Just

Expecting big improvements without significant change is unreasonable!

Changing “everything” in a small independent part is the ONLY way to real change.

- Experiment and learn with limited risk
- Resources for enough support
- Moore’s chasm



Feature team

Project Y

Project x

Coordination cost

Component team

Component team

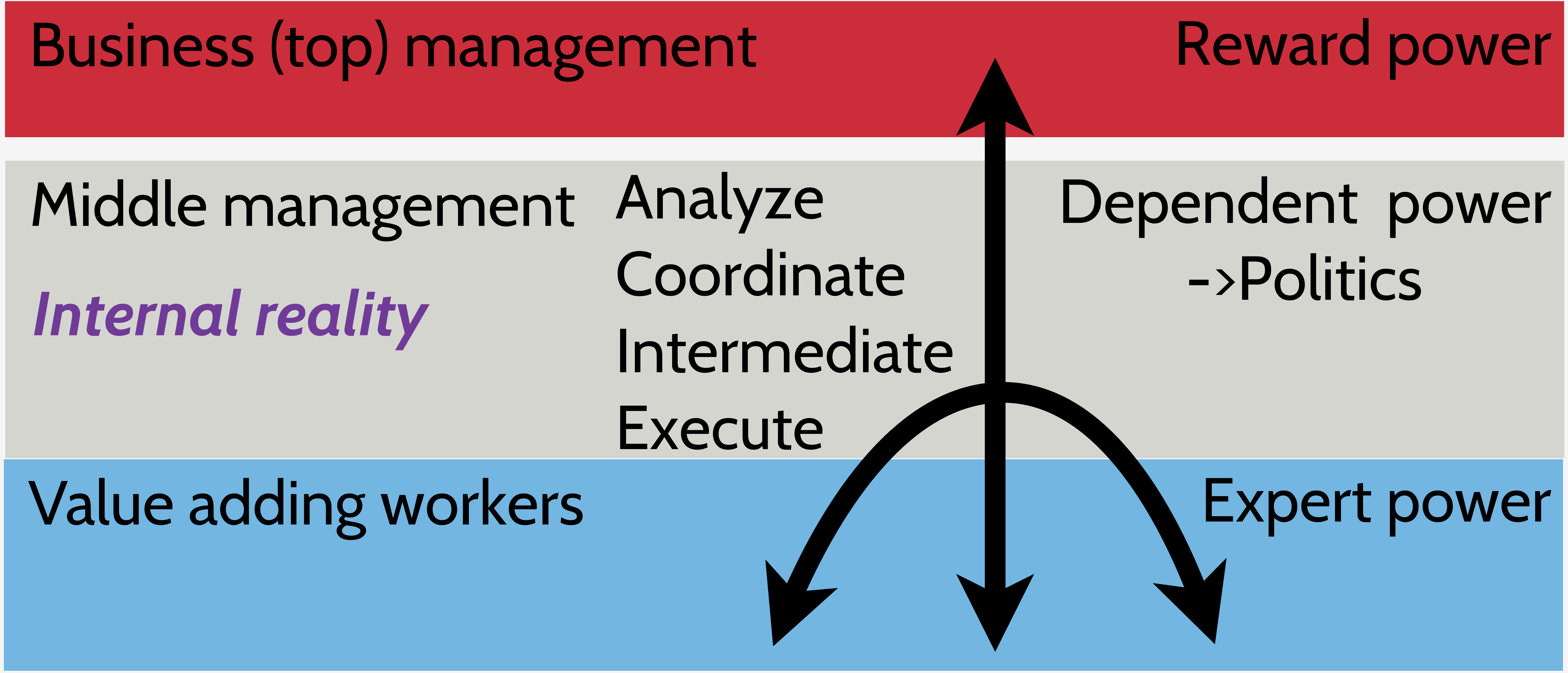
Component team

Investment in learning

Flow of work

Three Layers in (large) Organizations

Economical reality



Business (top) management

Reward power

Middle management

Analyze
Coordinate
Intermediate
Execute

Dependent power
->Politics

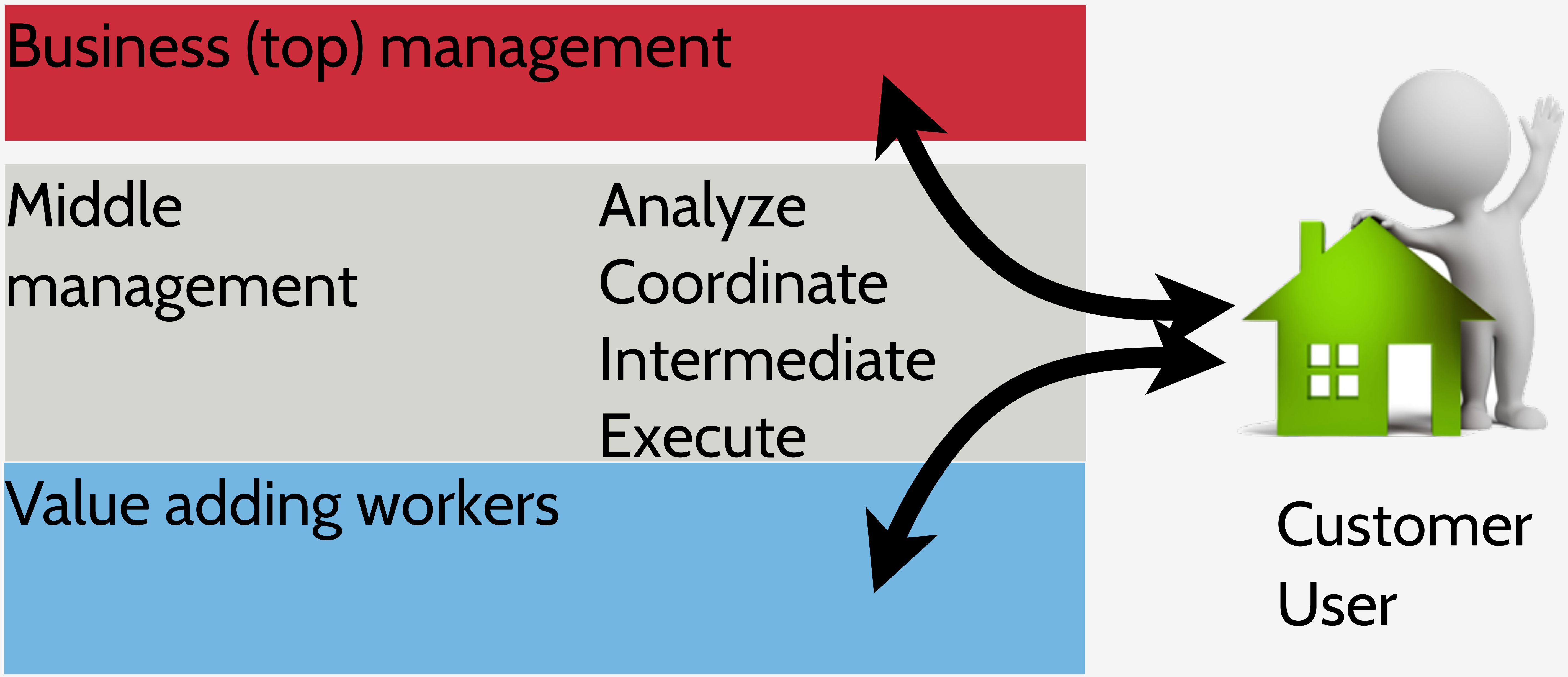
Internal reality

Value adding workers

Expert power

Technical reality

Who is missing?



Business (top) management

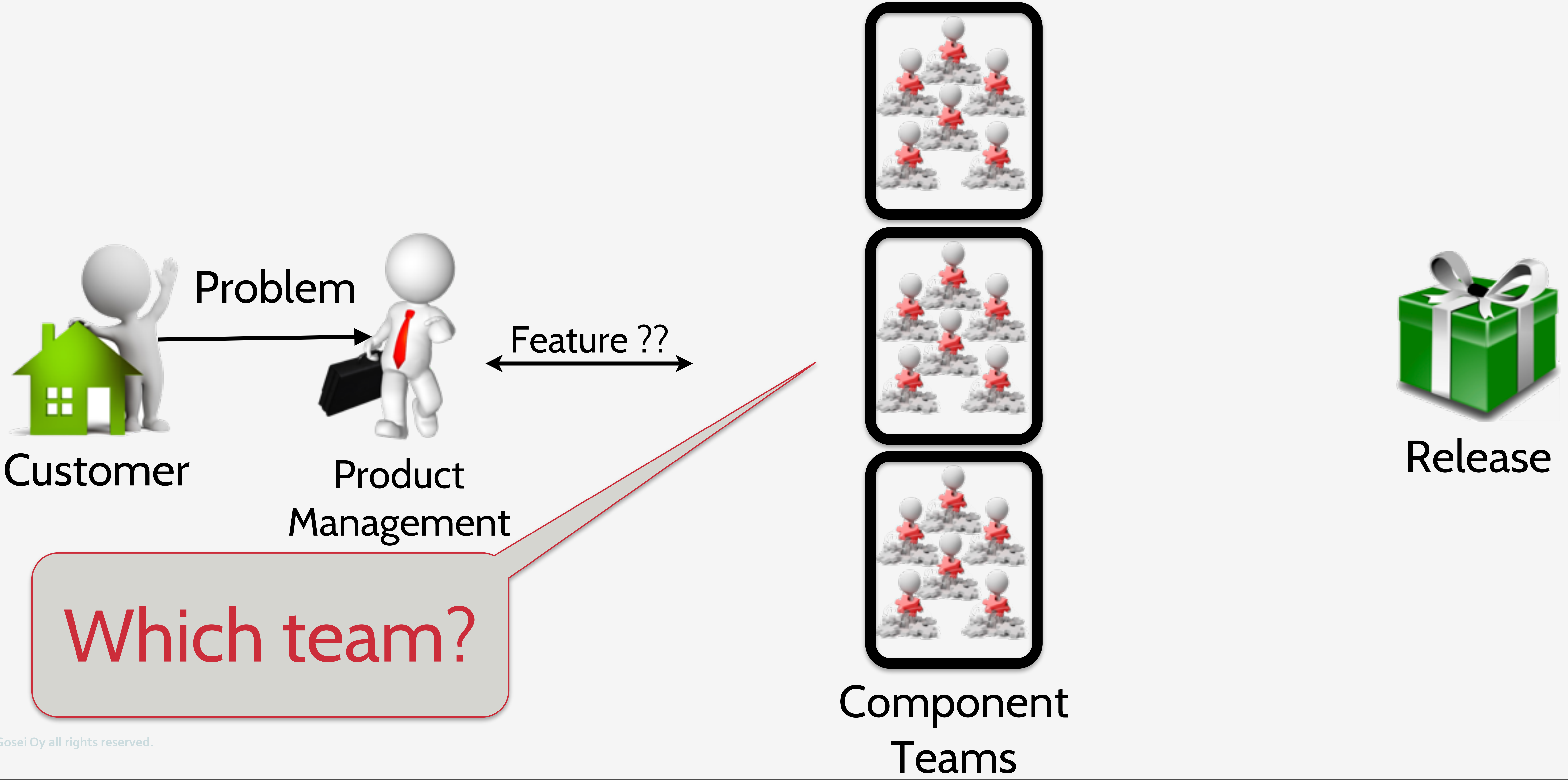
Middle management

Analyze
Coordinate
Intermediate
Execute

Value adding workers

Customer
User

Technology specailization leads to functional silos



Problem

Feature ??

Customer

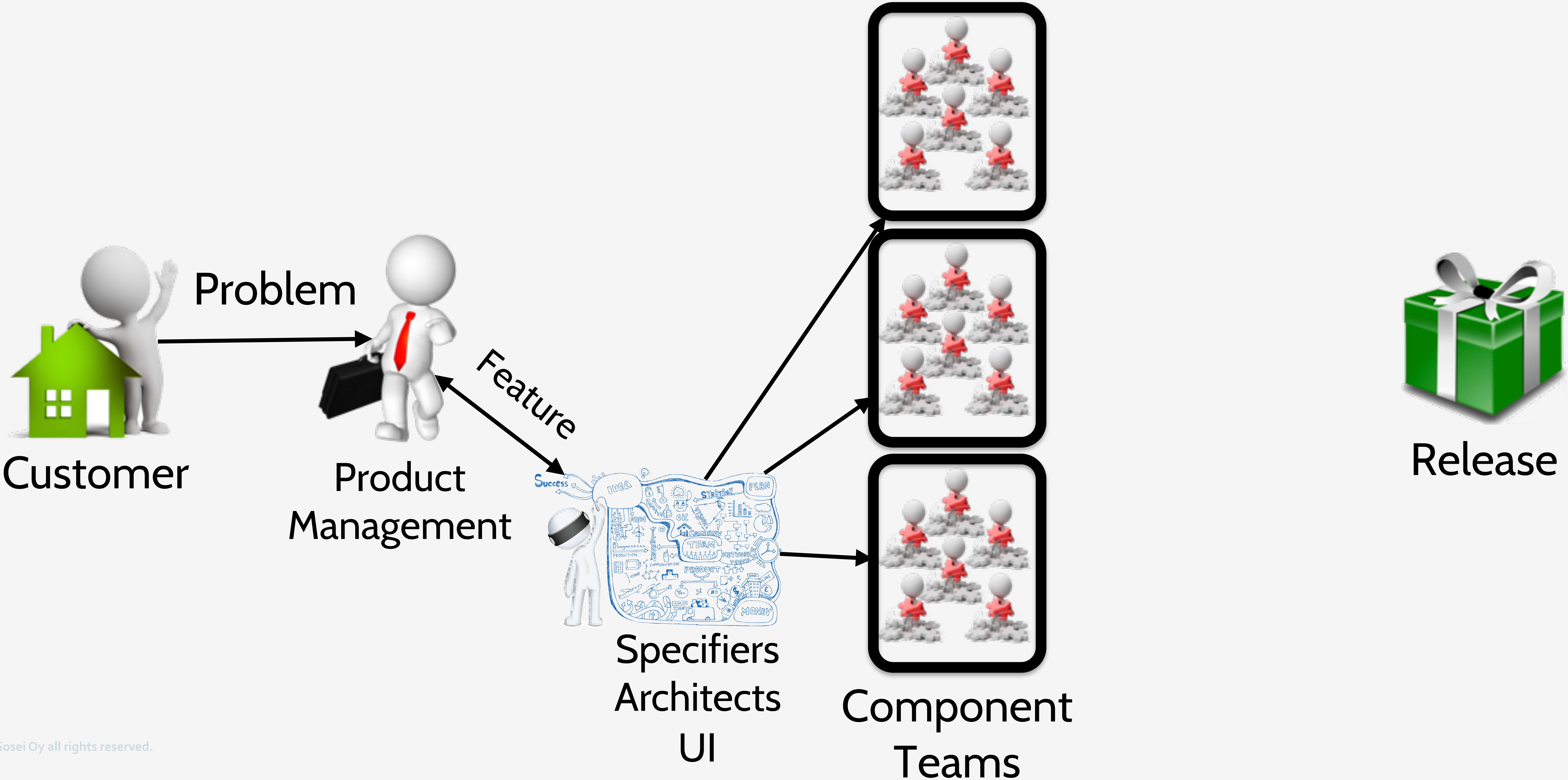
Product Management

Which team?

Component Teams

Release

Technology specailization leads to functional silos



Problem

Feature

Customer

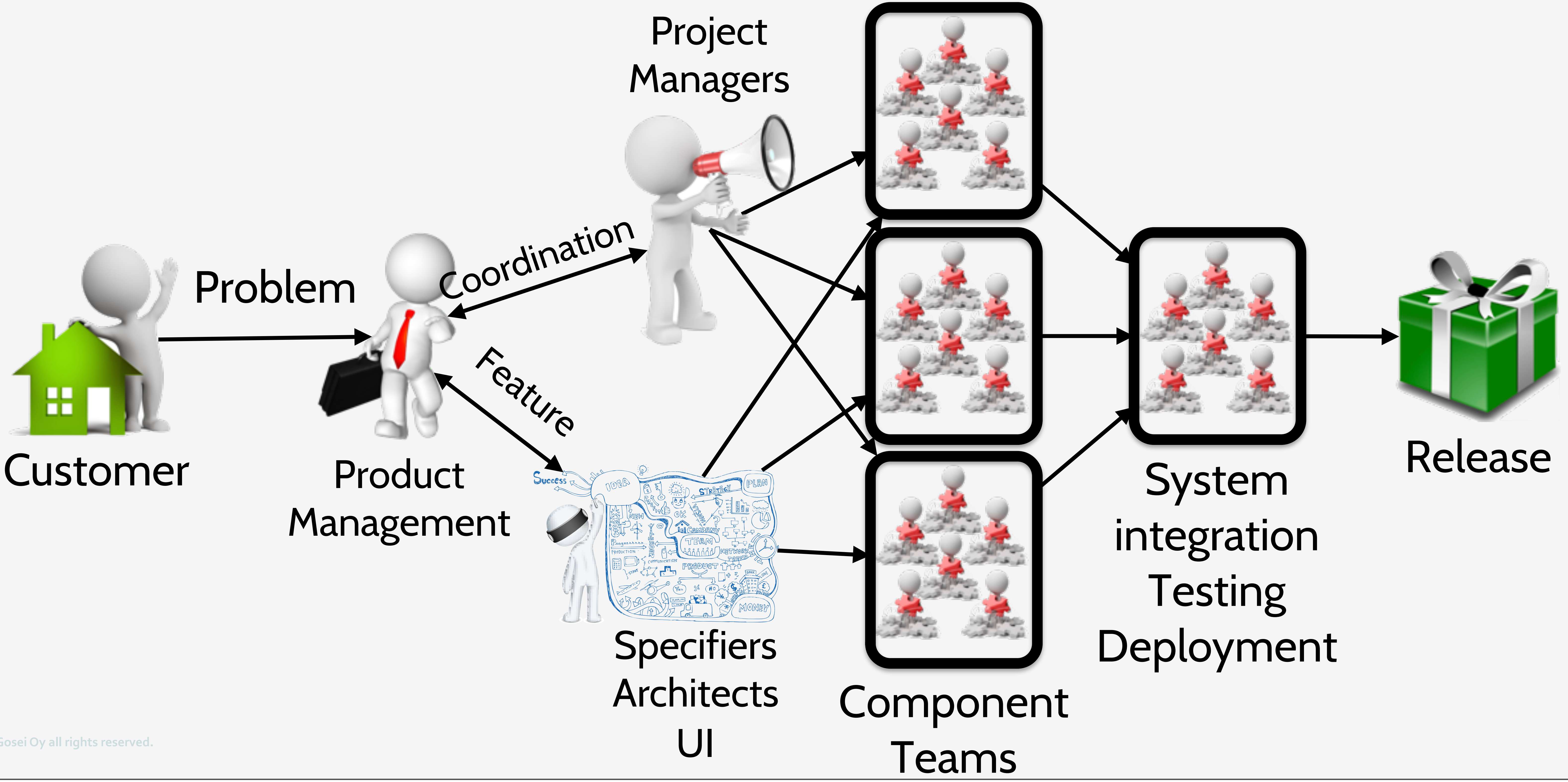
Product Management

Specifiers
Architects
UI

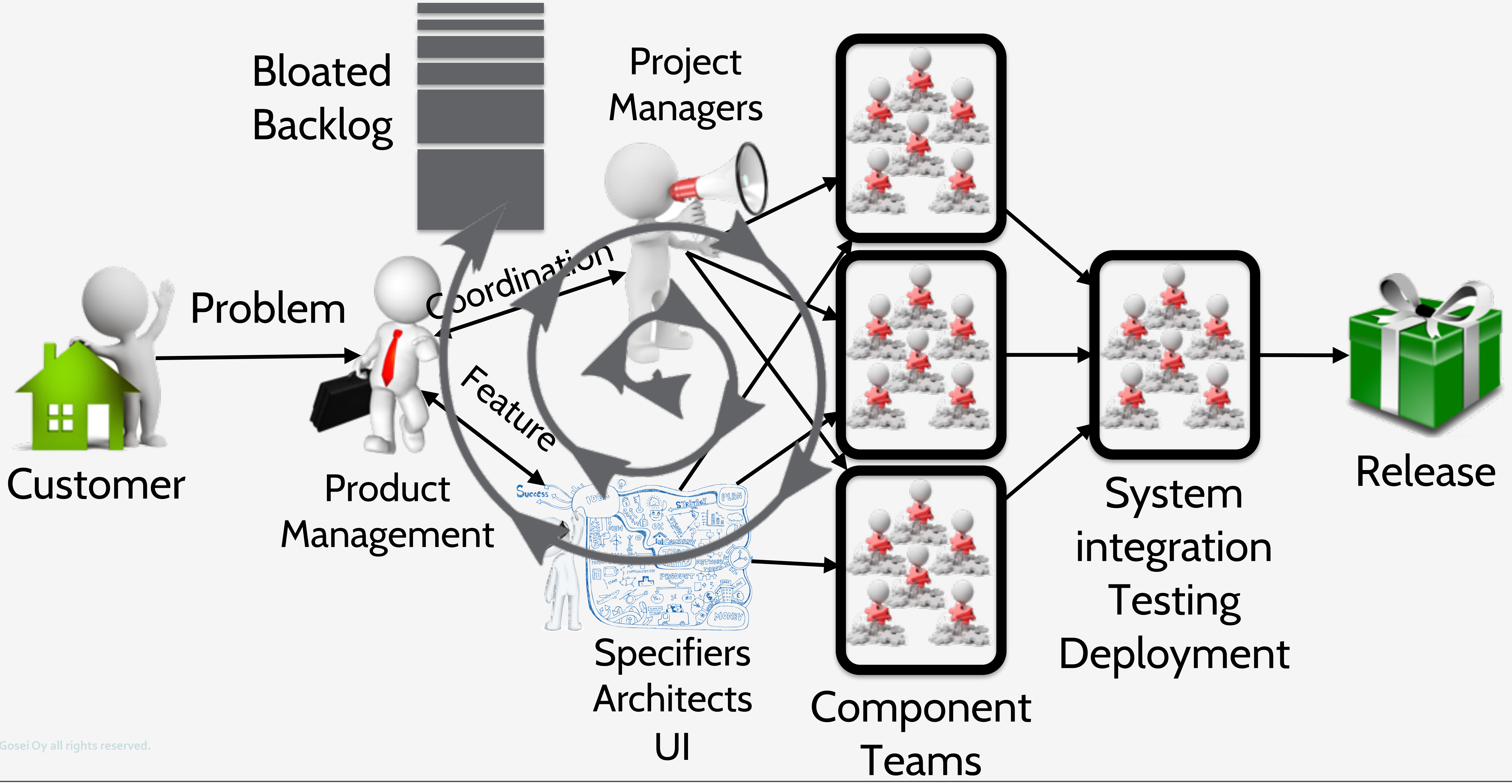
Component Teams

Release

Technology specailization leads to functional silos



Technology specailization leads to functional silos



Bloated Backlog

Project Managers

Problem

Coordination

Feature

Customer

Product Management

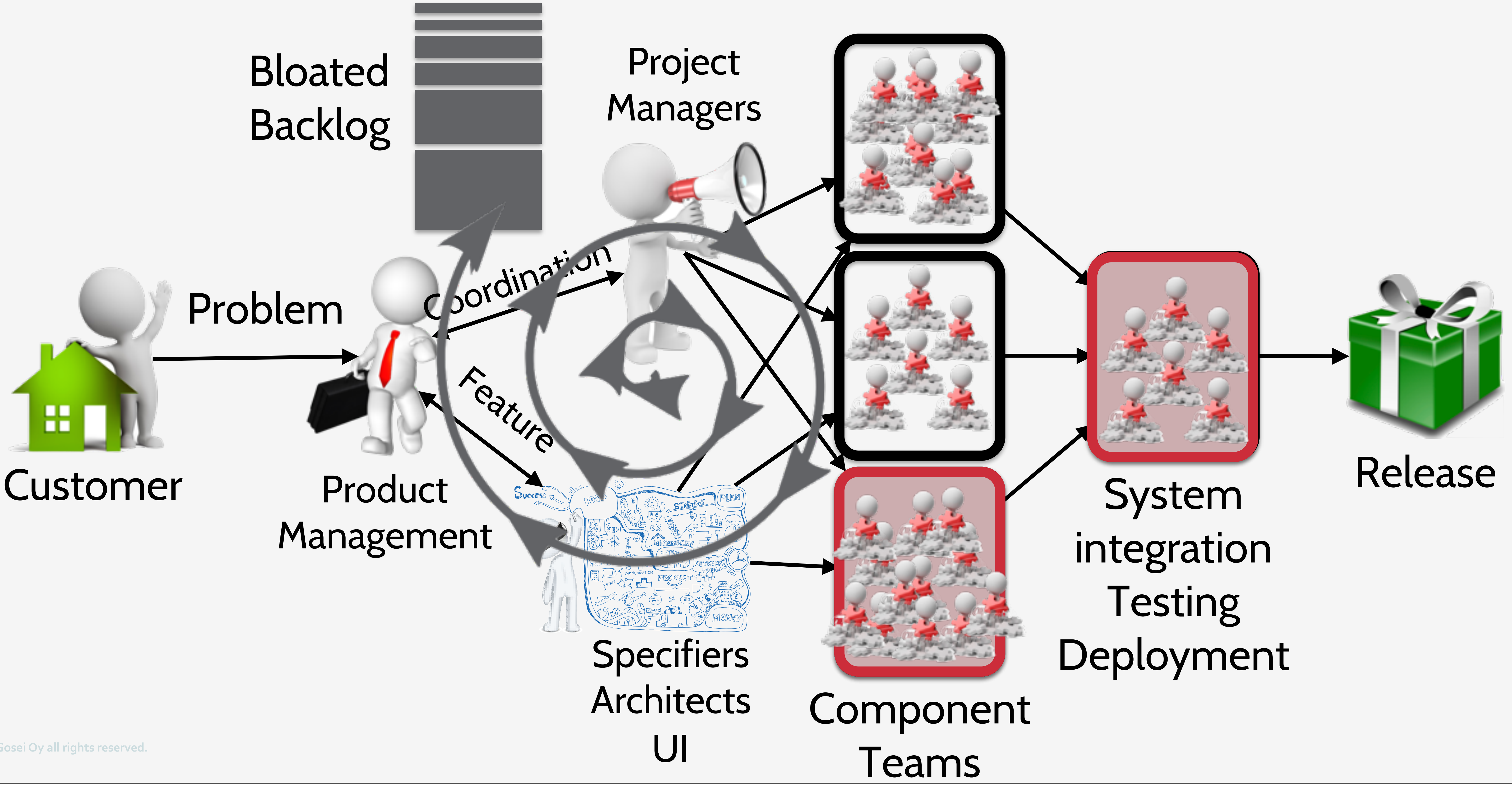
Specifiers Architects UI

Component Teams

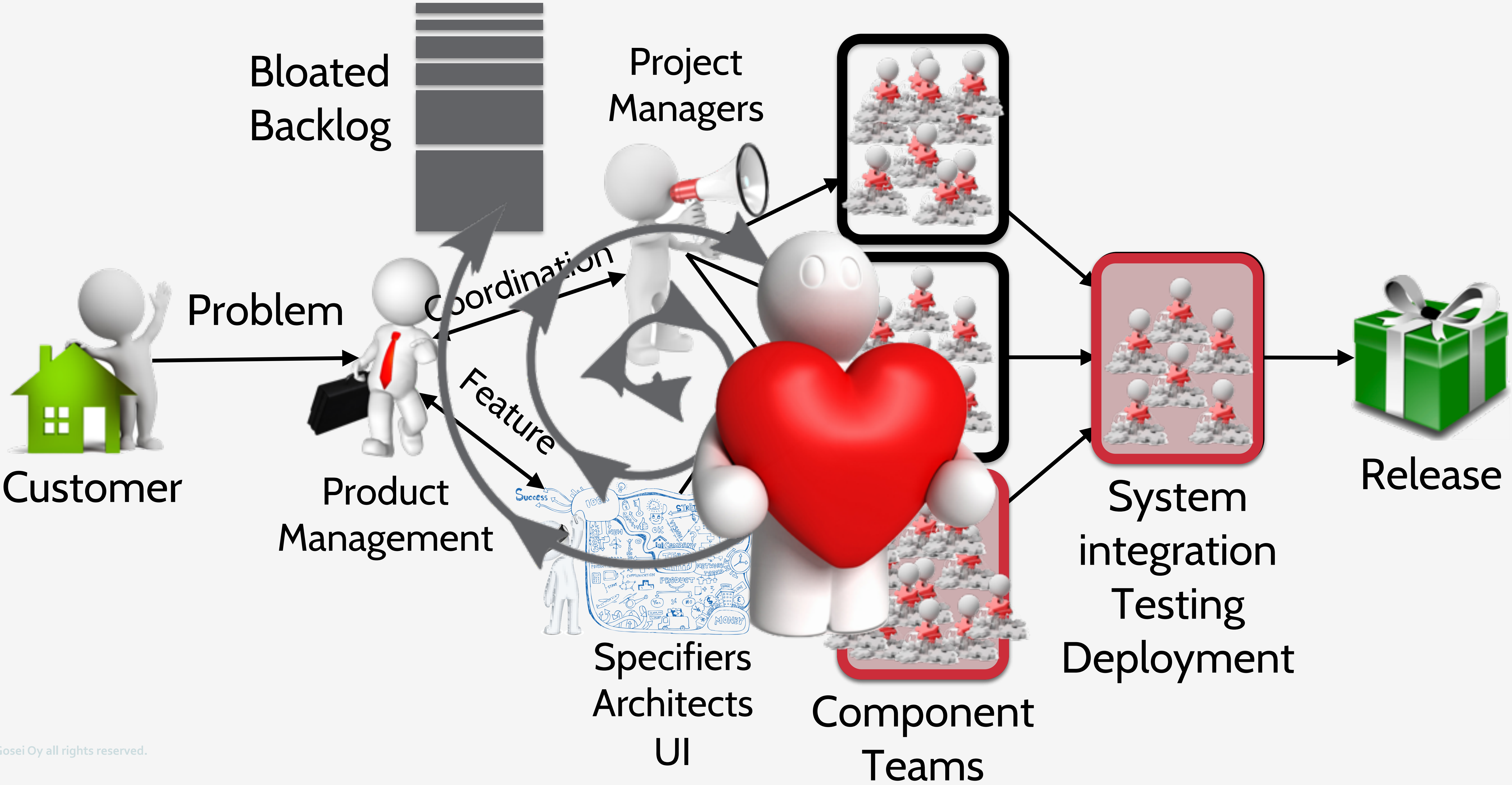
System integration Testing Deployment

Release

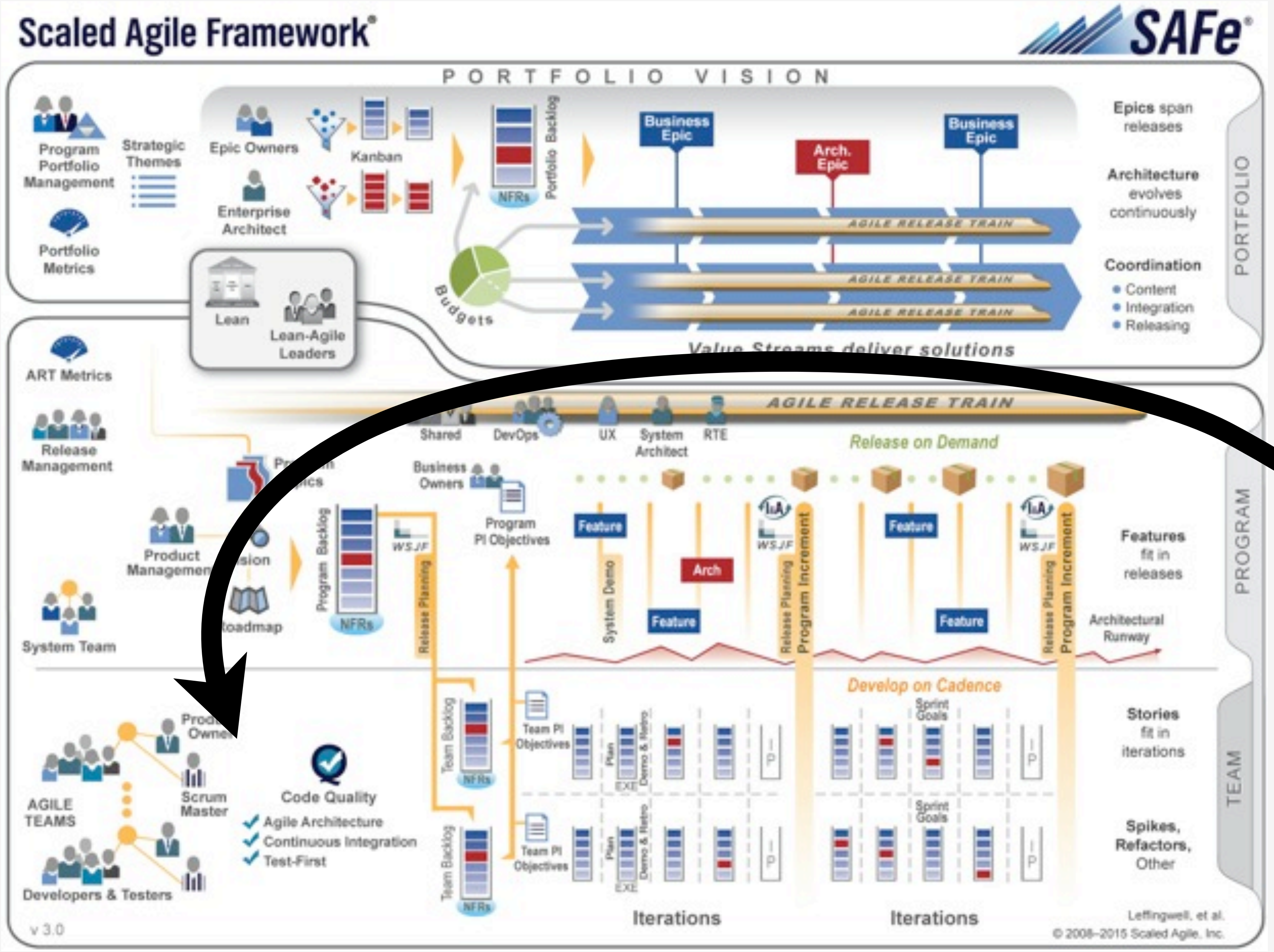
Technology specailization leads to functional silos



Technology specailization leads to functional silos

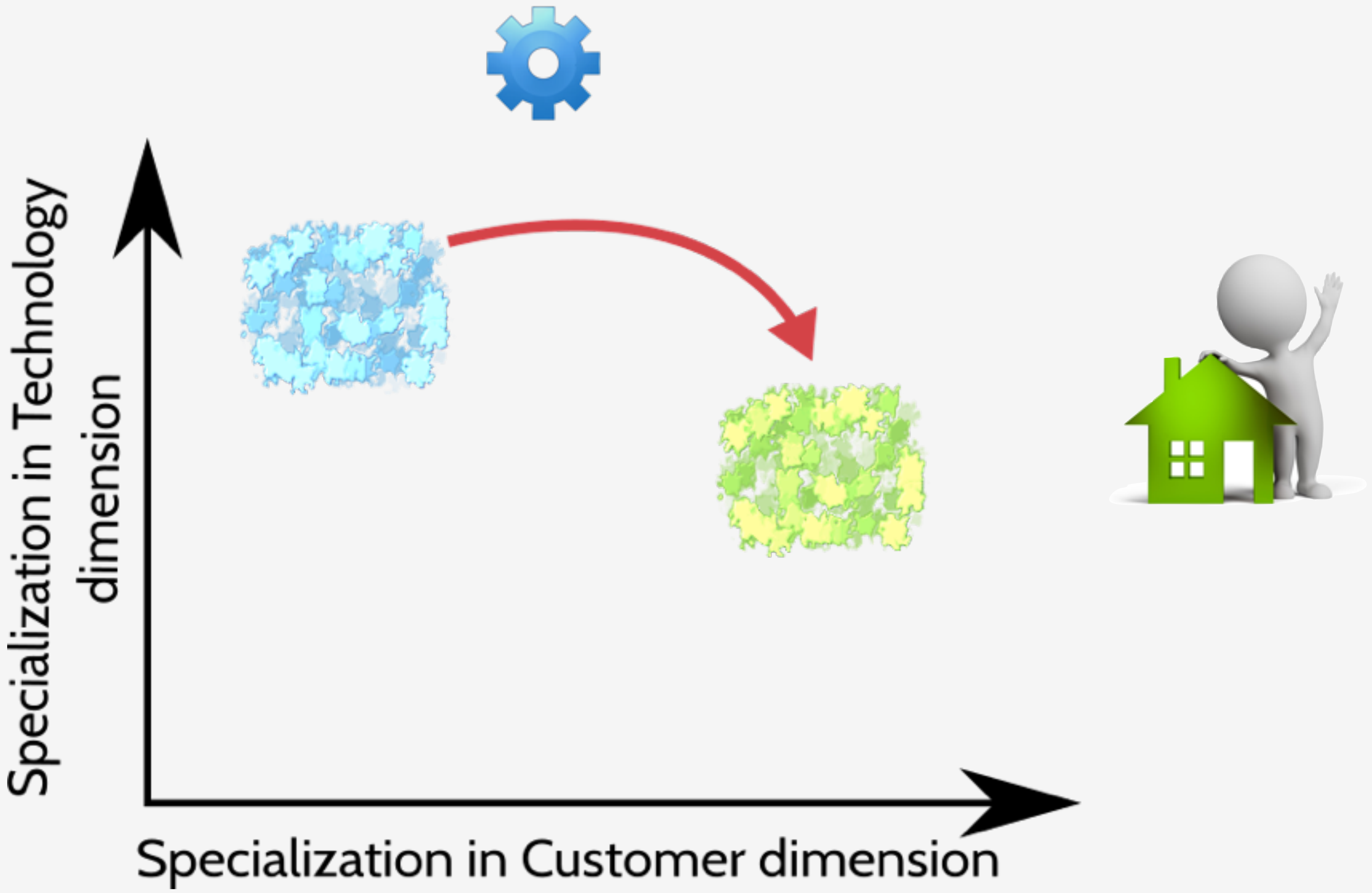


Through backlog and specialists

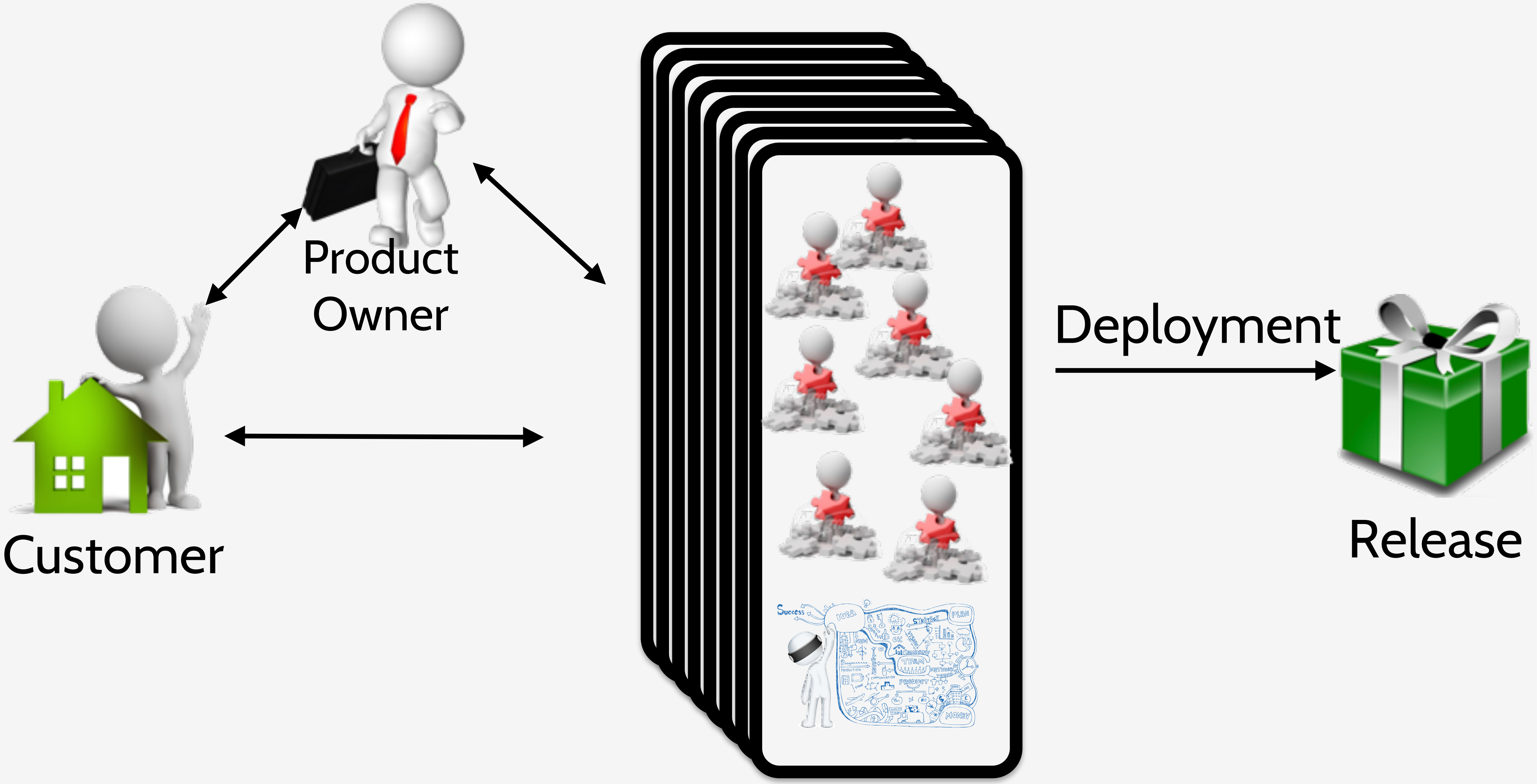


Customer
User

From Technical To Customer Specialization


Specialization in Technology dimension
Specialization in Customer dimension

Flow of work with Customer Specialization



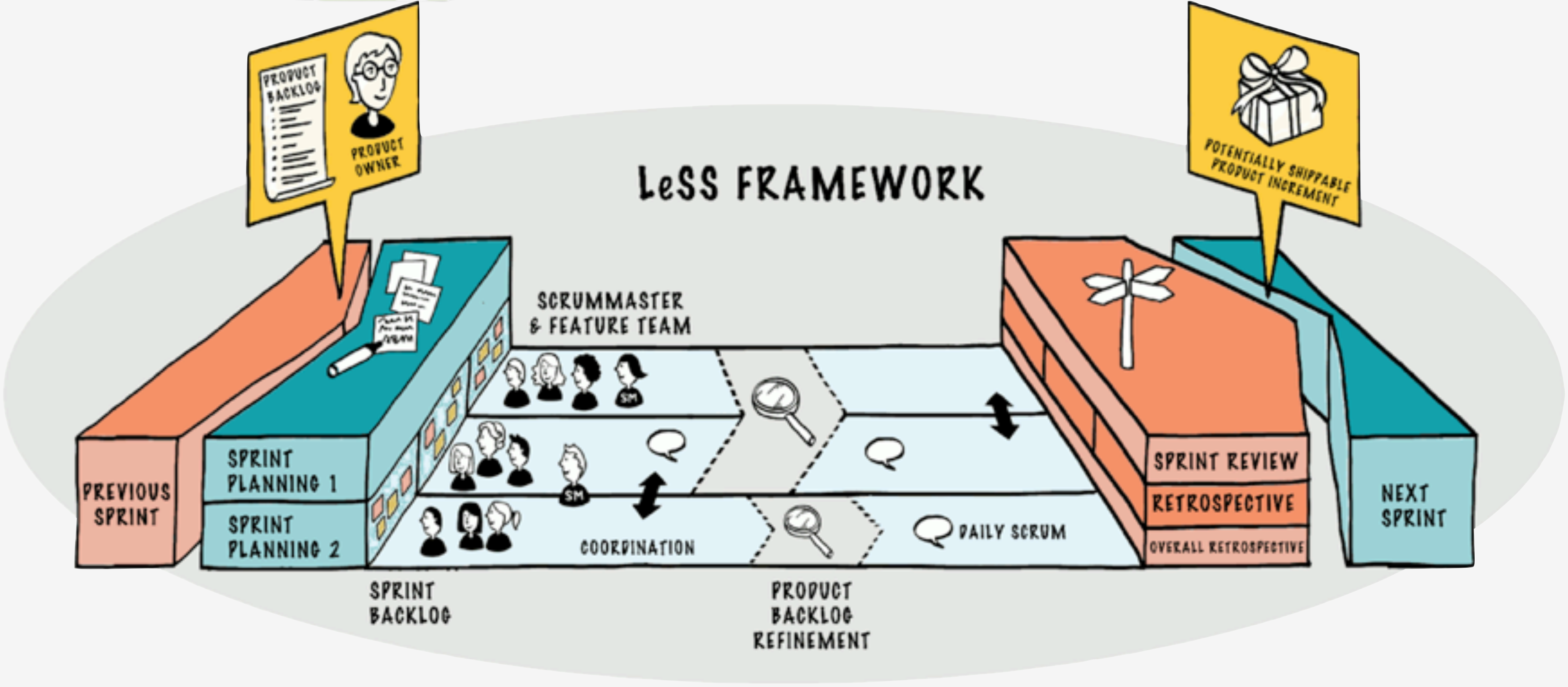
Product Owner

Customer

Deployment

Release

Flow of Work LeSS



Noooooo!

It is too simplistic.

We are so many!

Yes, it is simple and not easy

Technology, competence, identites and culture need to develop.

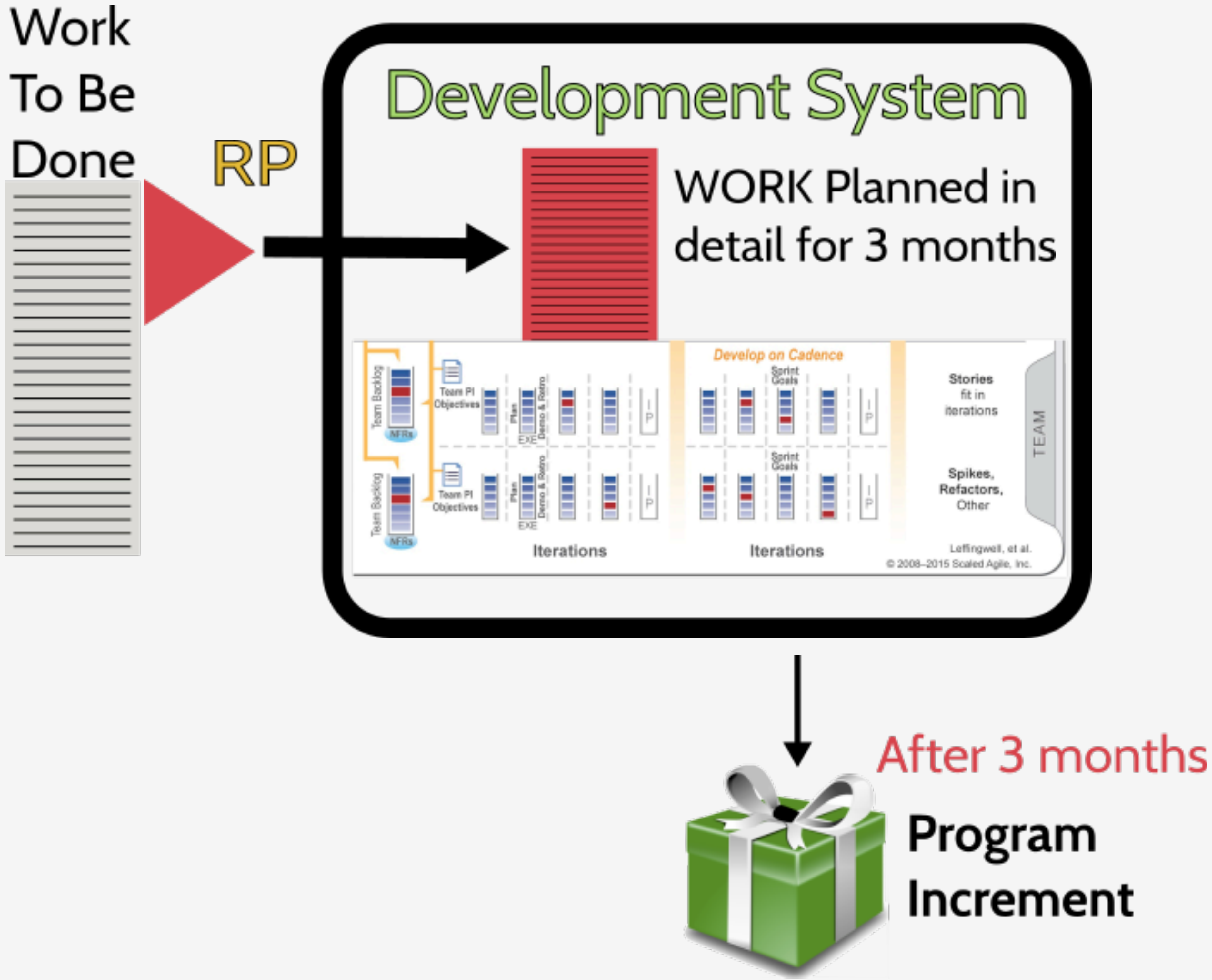
Learning causes anxiety. Only survival anxiety is greater. (E. Schein)

- Takes time, like any real change.
- There will be worry and resistance.

Leadership challenge

Batch size and Queues

SAFe Batch Size



Work To Be Done

RP

Development System

WORK Planned in detail for 3 months

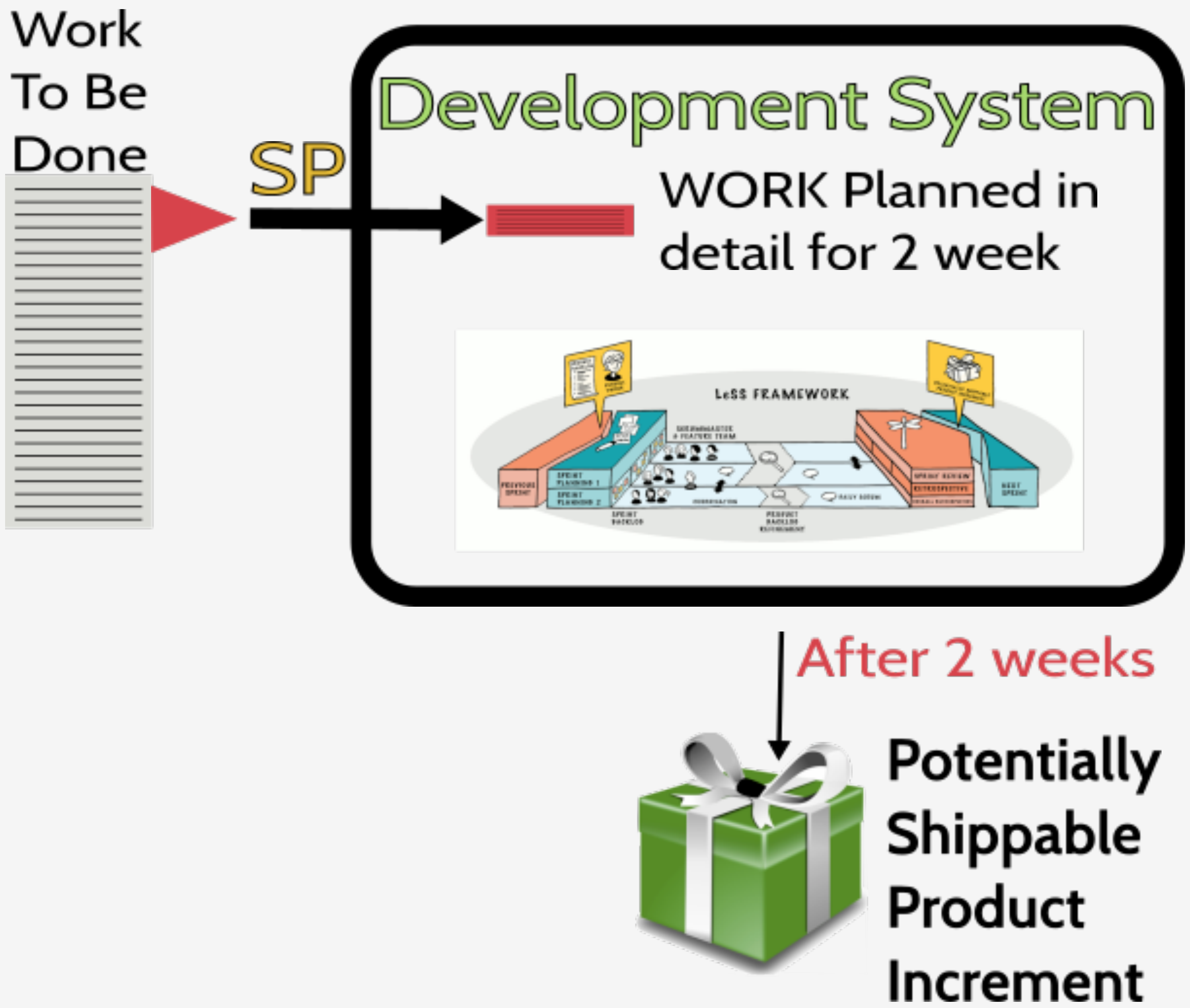
After 3 months

Program Increment

Planning cycle 3 months

Create big batch of work to reduce total cost

LeSS Batch Size



Work To Be Done

SP

Development System

WORK Planned in detail for 2 week

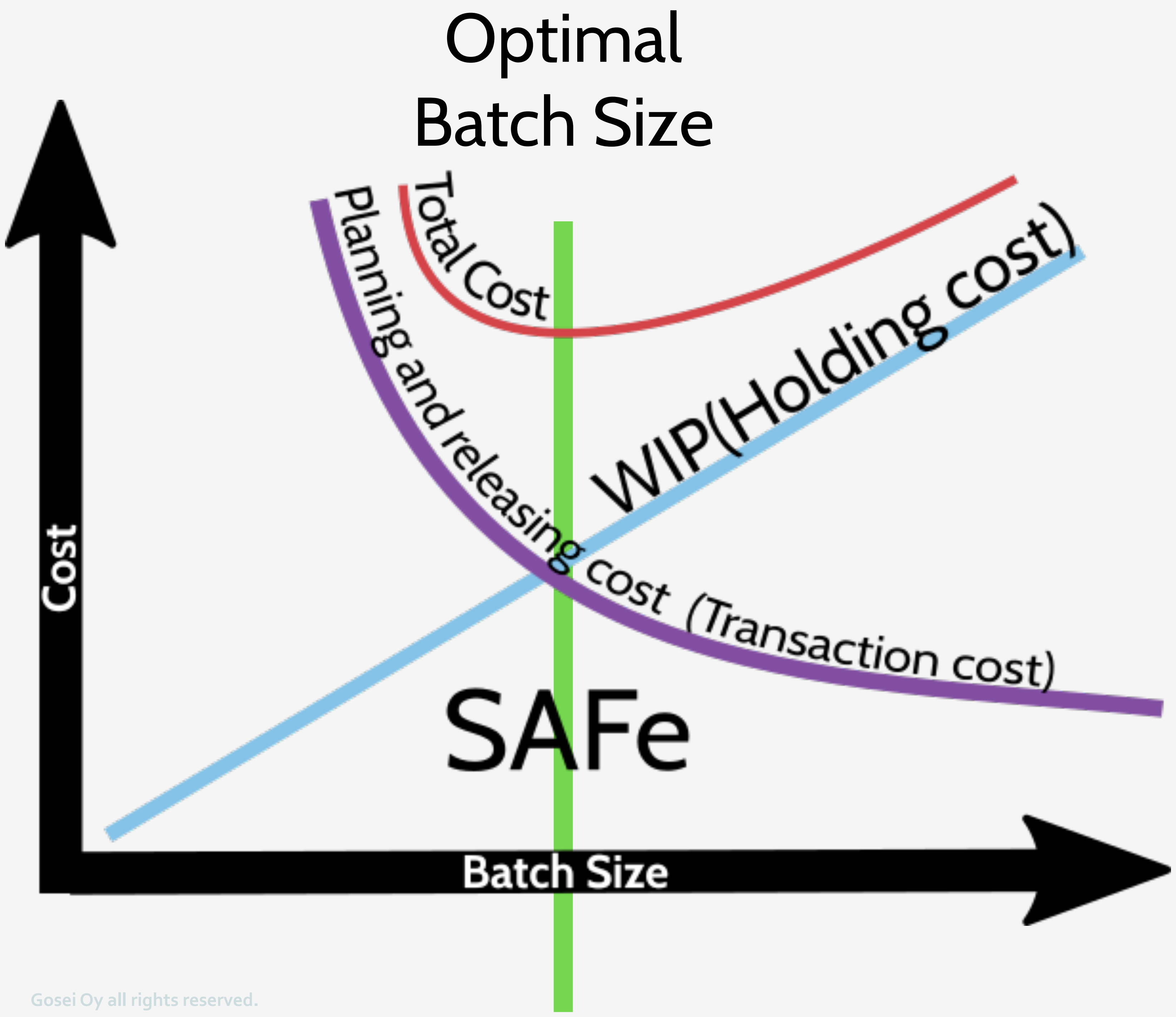
After 2 weeks

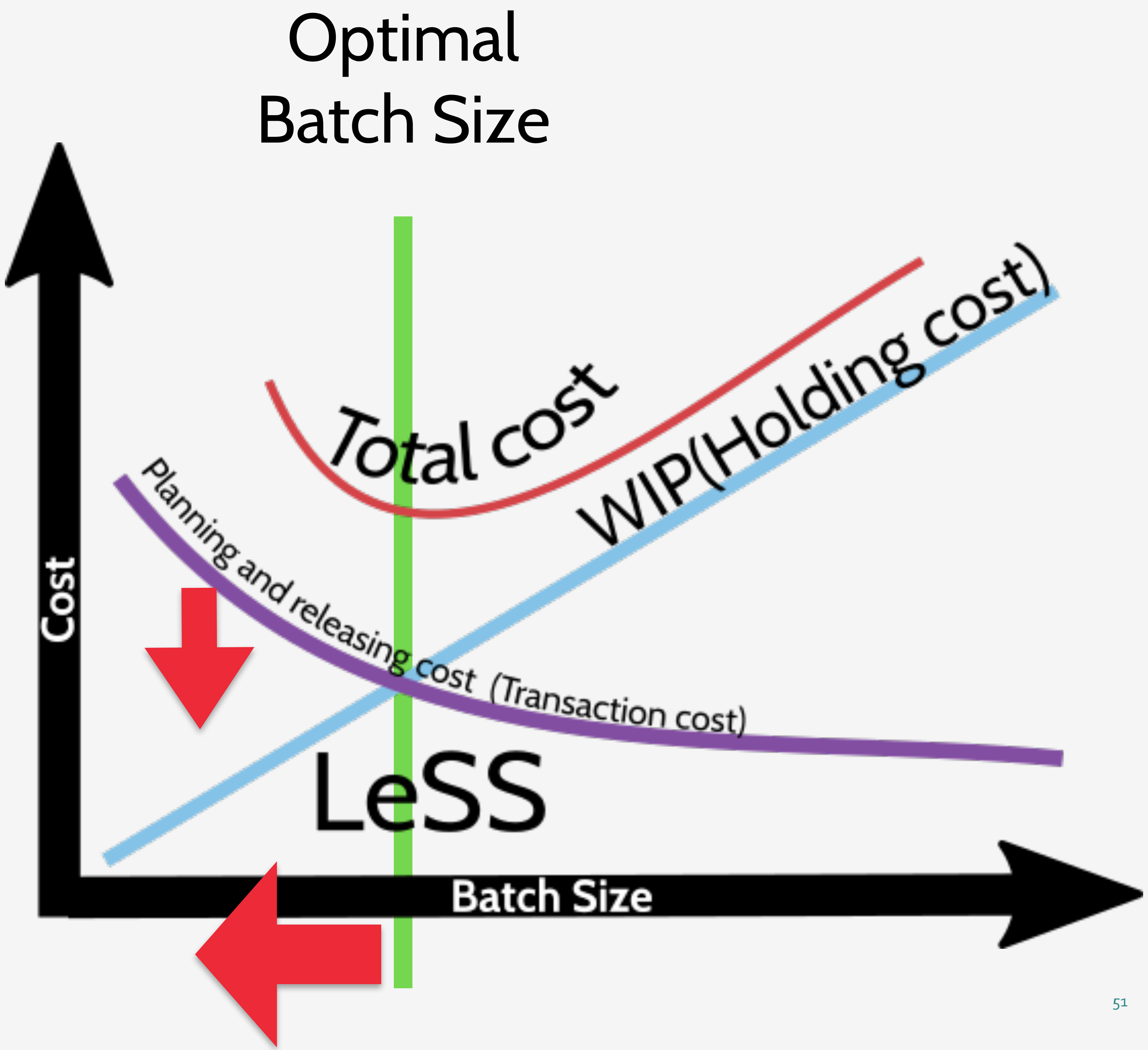
Potentially Shippable Product Increment

Planning 2 week increments

Create small batches of work that will enable fast feedback

Why is the Batch Size Problem?



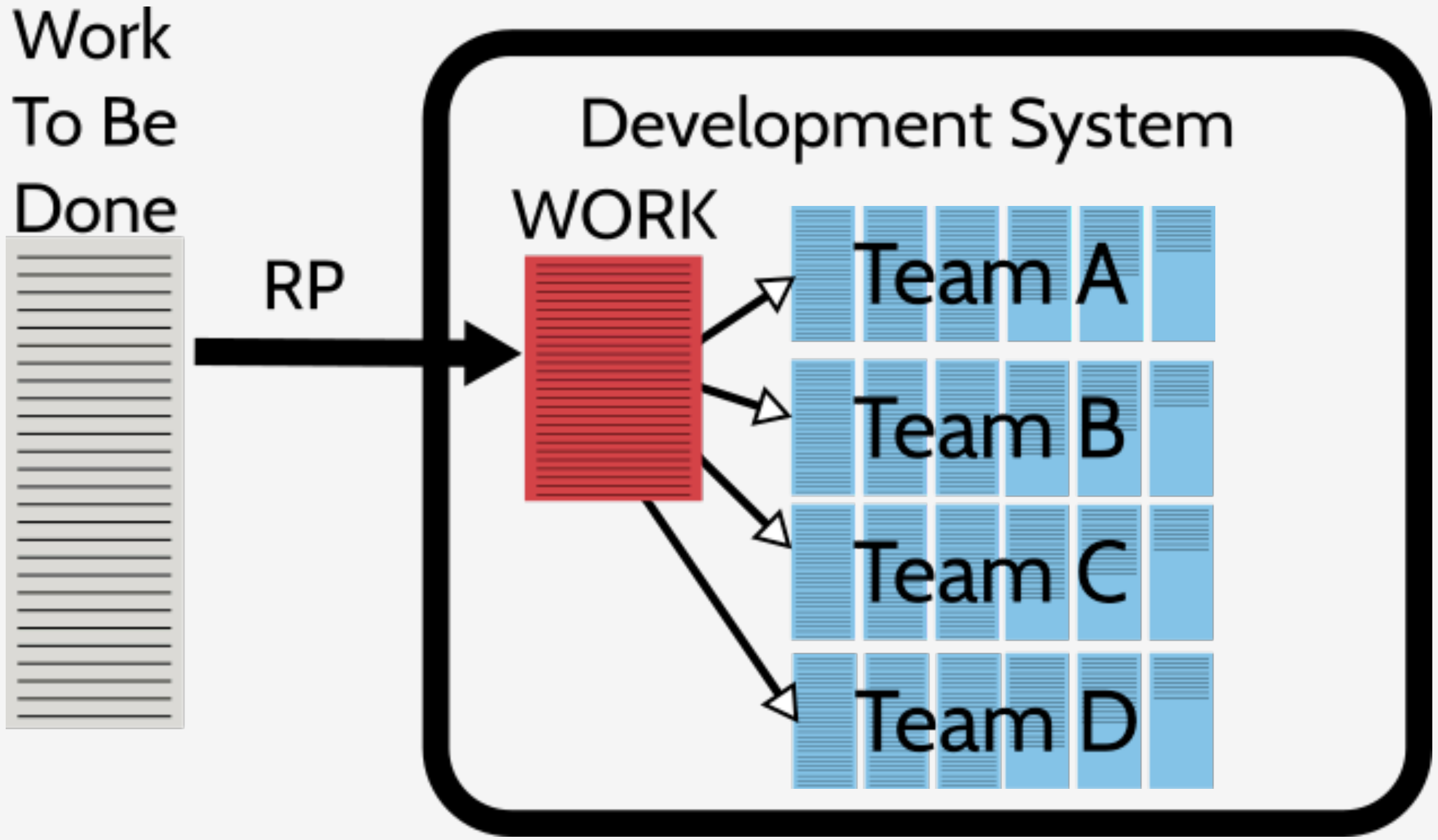


Product Development and Big Batches

“We have found out that reducing batch size improves most development projects significantly.”

- Six Myths of Product Development Stefan Thomke and Donald Reinertsen -

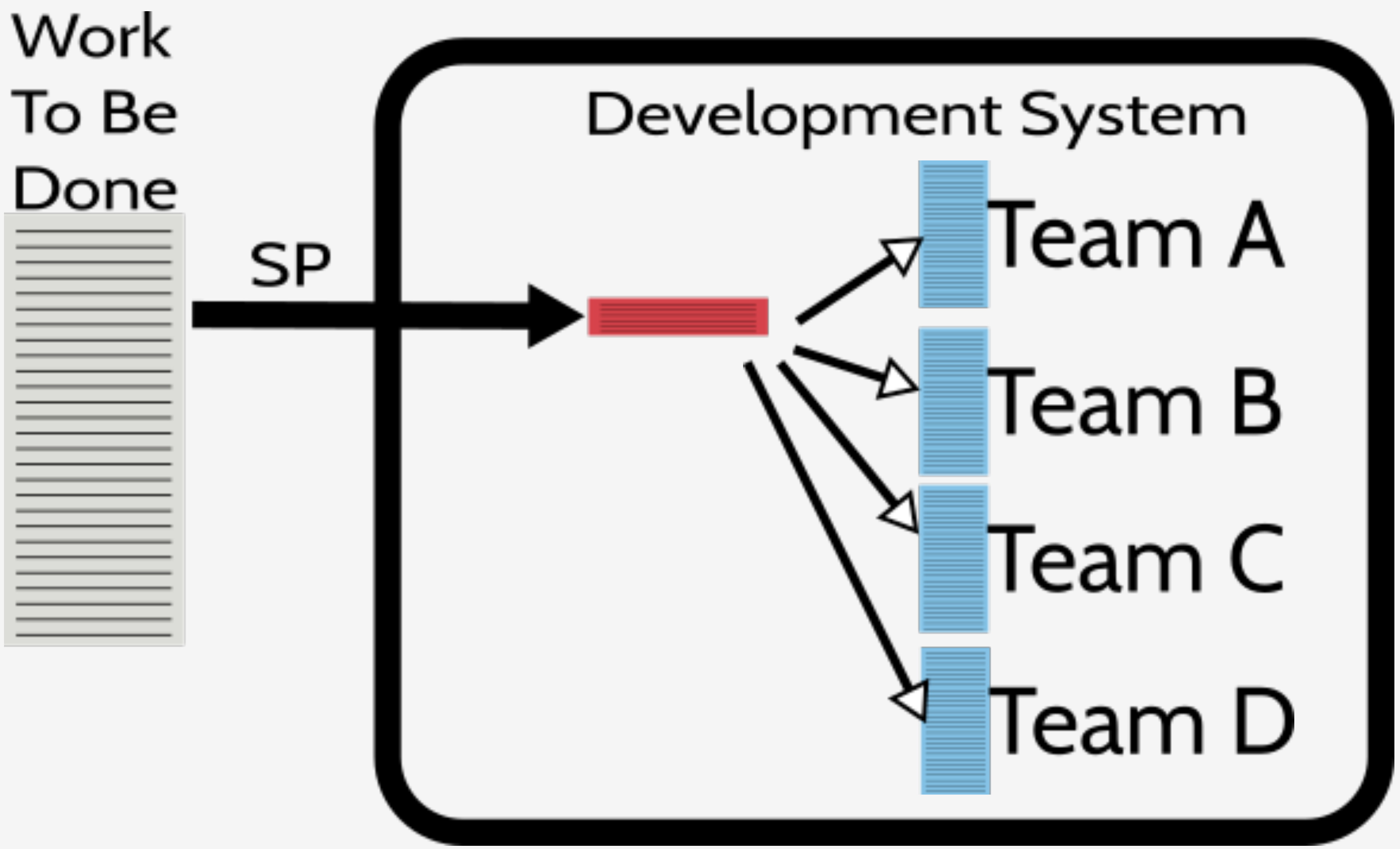
Queues SAFe



SAFe

- ○ Loads the system full of queues for a Program Increment
- ○ Optimizes resource utilization

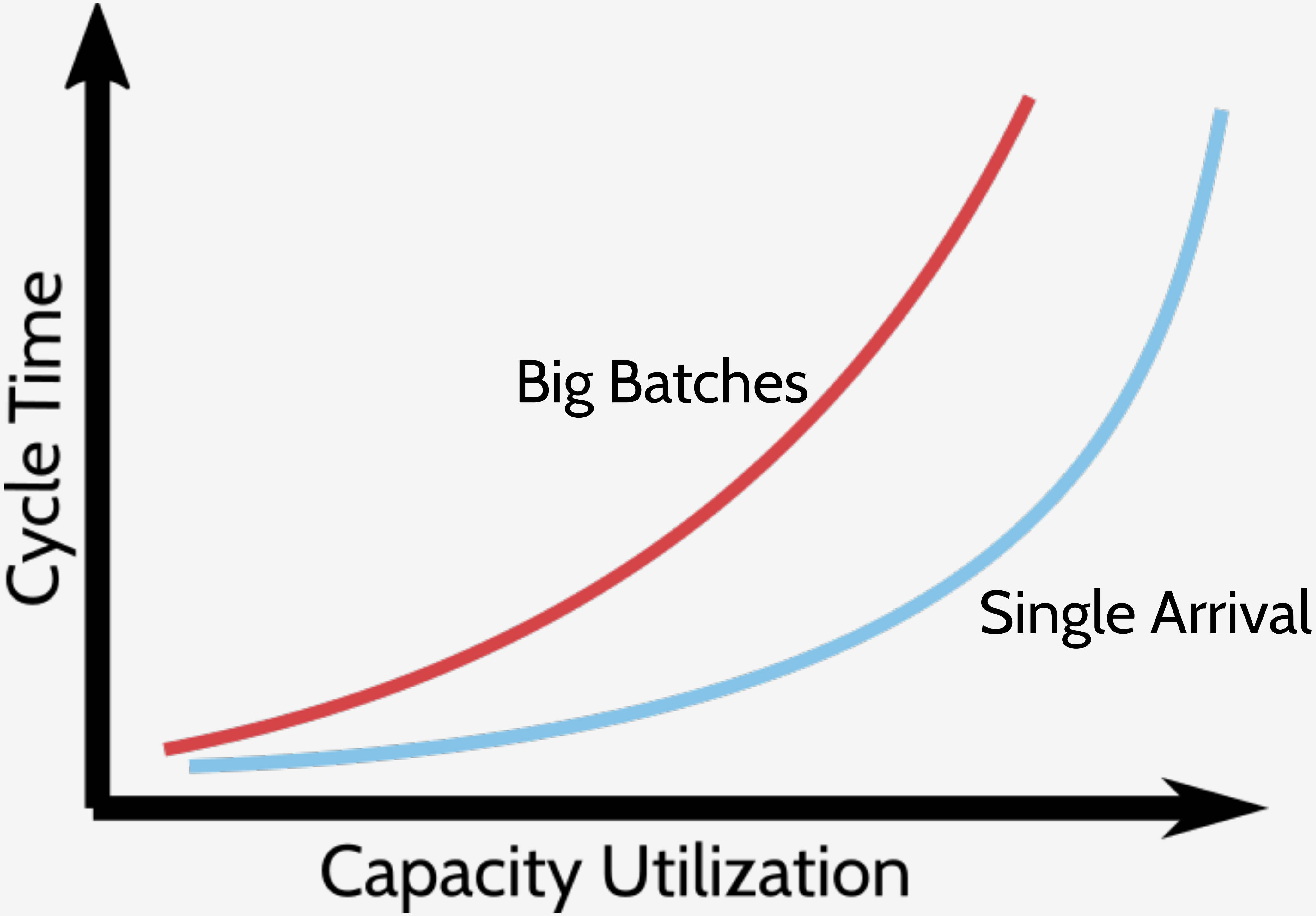
Queues LeSS



LeSS

- Tries to keep queues outside of system
- Optimizes outcome after each iteration

General Problem With Queues and Big Batches



Product Development and Queues

"Queues delay feedback, causing developers to follow unproductive paths longer. They make it hard for companies to adjust to evolving market needs and to detect weaknesses in their product before it's too late."

- Six Myths of Product Development Stefan Thomke and Donald Reinertsen -

Coordination Summary

Fundamental formula

Reach (length) of the plan = Utilization x Specialization

When coordinating the work to be done,

the more technology-specialized the organization is

and the more you want to optimize utilization

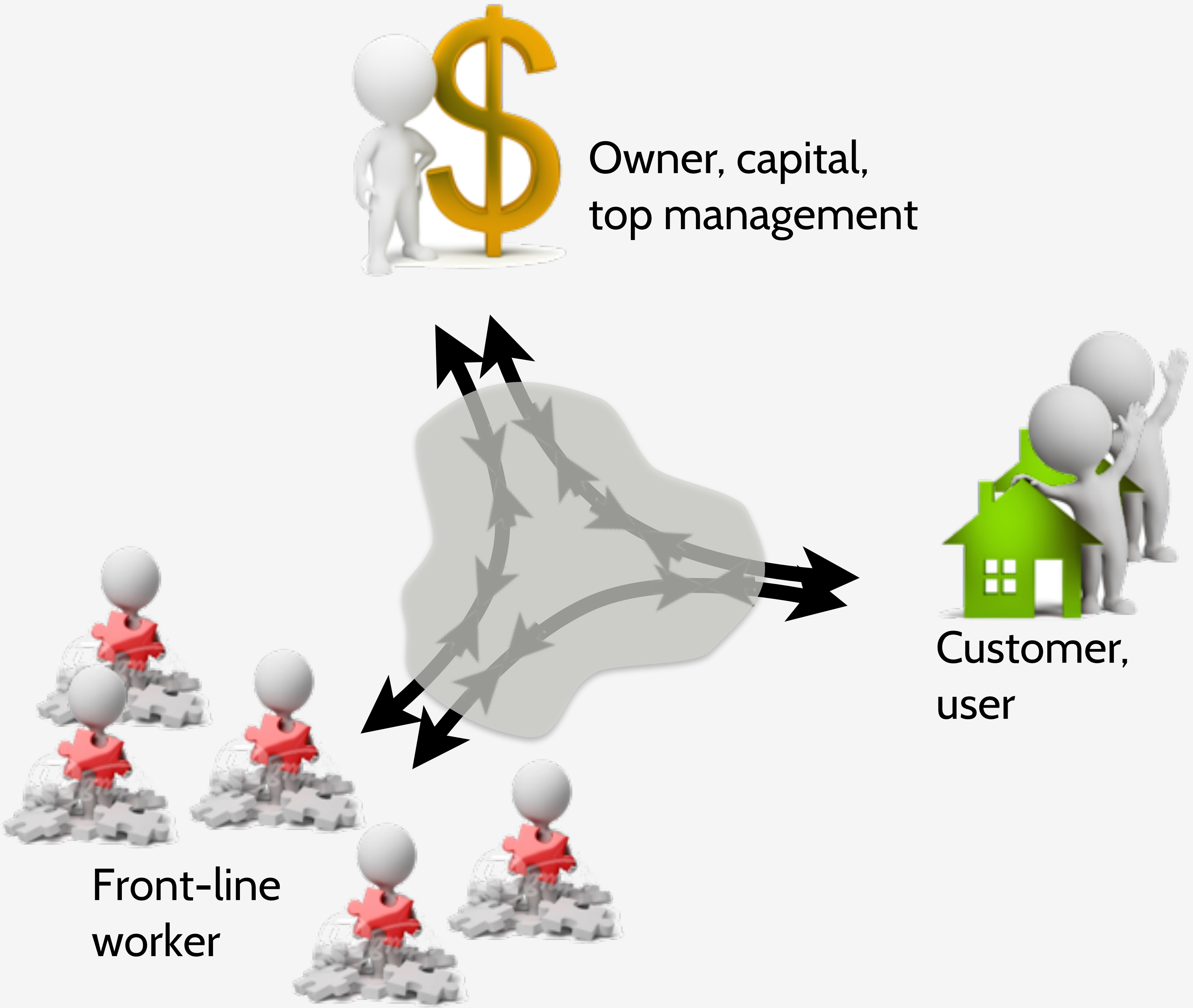
the further into the future you need to plan.

Coordination Approaches Compared

	SAFe	LeSS
Main control mechanism	Bureaucratic	Clan
Solving dependencies	Coordinate people	People work with technology
Batch size	Big and slow tasks for scarce resources (people). 3-month releases needed to plan.	Fast, small and parallel technical transactions. Sprint-long iterations.
Cost of dependencies	Coordination is seemingly necessary waste	Learning to work with technology is investment
Optimization	Resource optimization (coordination)	Outcome optimization
Customer contact	Intermediated	Direct
Organizational maturity	Possible with lower skill Learning for the role "Natural" development	Higher skill needed Learning what is needed Skilled evolution, leading learning
Requires stability in	Component organization functions in unchanging environment.	Long-living teams adapt fast to changes in environment.

Business view

Growth of the middle management



Owner, capital,
top management

Customer,
user

Front-line
worker

From Agile Manifesto:

Individuals and interaction

Business and developers work together daily

Face-to-face conversation

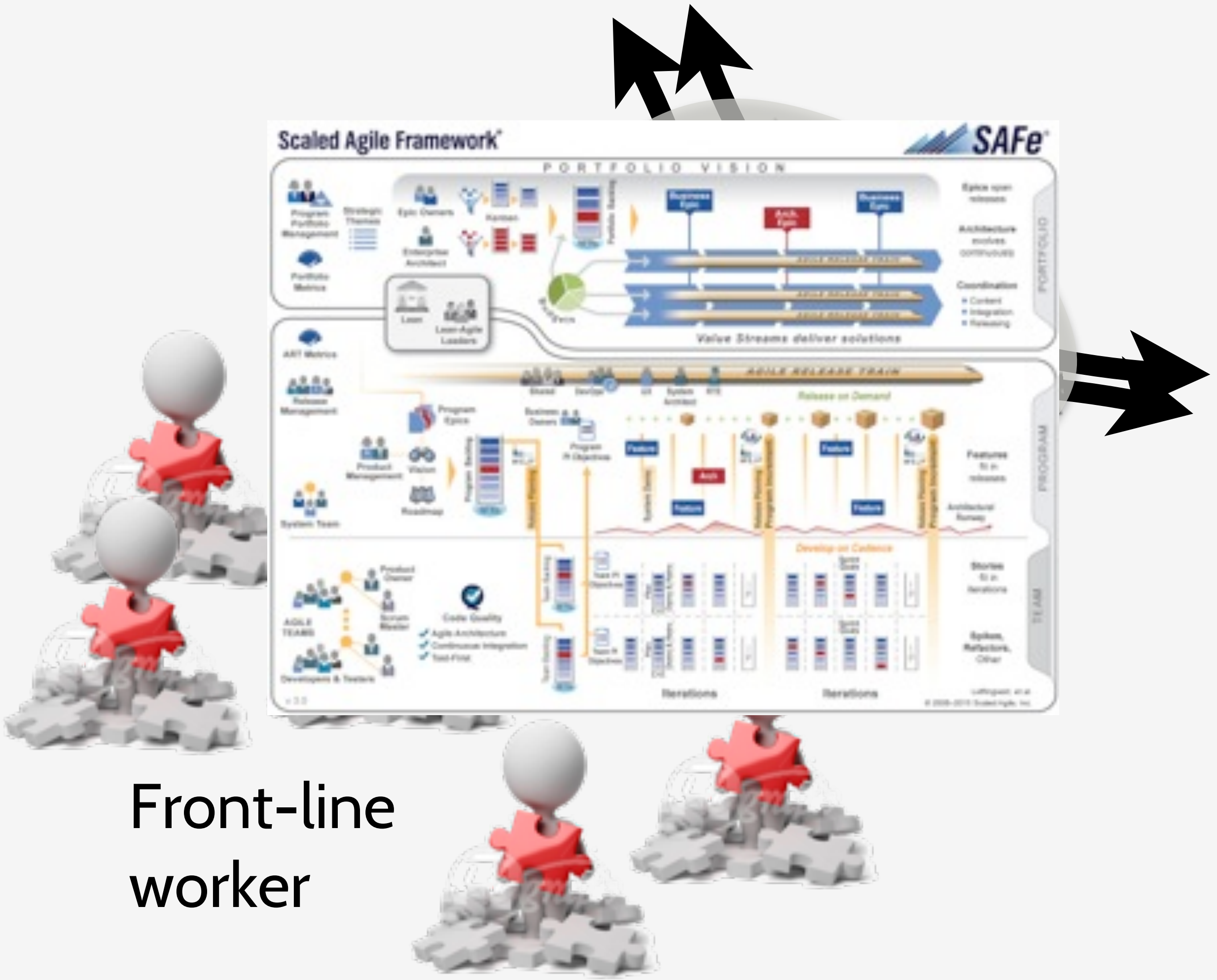
Simplicity

Self-organization

Learning from reality

1. New process and best practises by SAFe

Owner, capital,
top management



Front-line
worker

Customer,
user

From Agile Manifesto:

Individuals and interaction

Business and developers work together daily

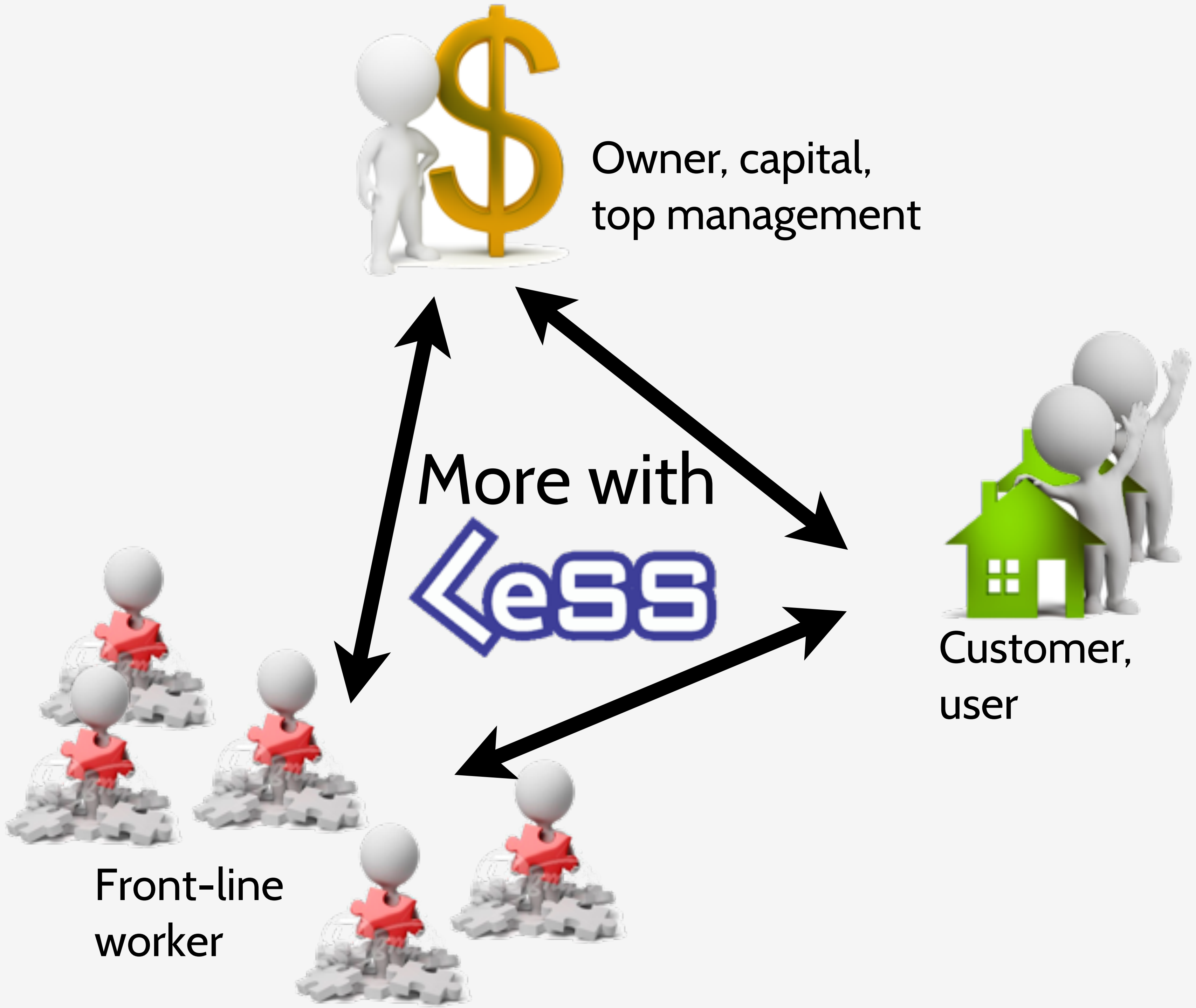
Face-to-face conversation

Simplicity

Self-organization

Learning from reality

2. Dis-intermediating by LeSS



From Agile Manifesto:

Individuals and interaction

Business and developers work together daily

Face-to-face conversation

Simplicity

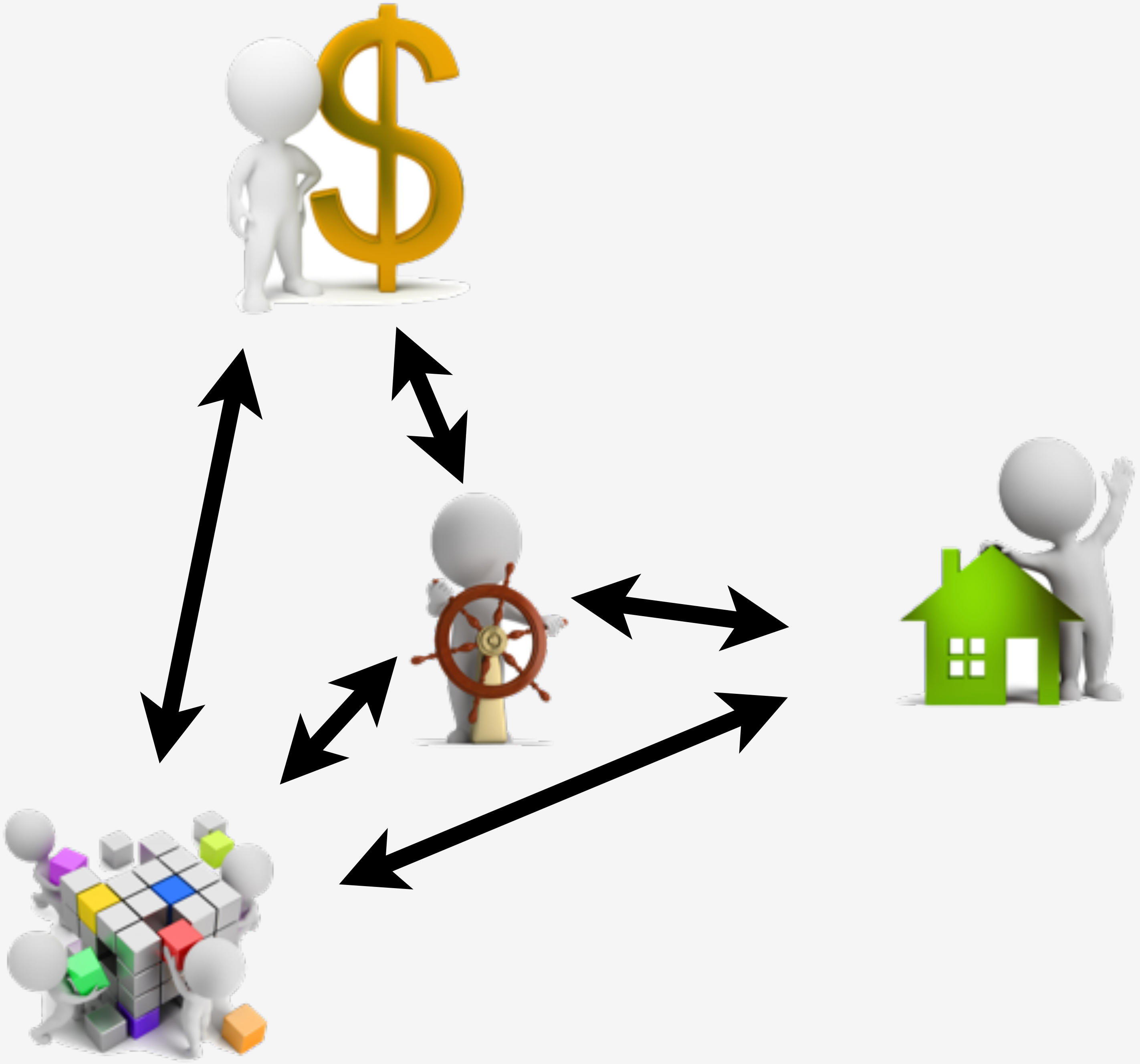
Self-organization

Learning from reality

Nooooooo!

It will break!

LeSS Organizational design



Find your product to enable direct customer interaction.

Build customer-oriented feature teams.

Learning away from coordination chaos. Decoupling in practise.

The Product Owner decides, customer interaction clarifies.

The line management grows the value of the organization.

Your key questions

What do you want?
What do you dare?

Questions

	SAFe	LeSS
Slogan	Program Execution	Customer-centric Learning
Framed problem	Internal efficiency	Optimal response to customer demand
Value proposition for “Scaling Agile”	Improved program execution Lean-Agile ways of working	More with less: Effective and agile value-adding work Minimal bureaucracy
Solution	Program process and best practises	Organizational design: principles, guides, rules and 600 experiments for inspect and adapt
Main control mechanism	Bureaucratic	Market, Clan
Real-time delivery	Detailed planned 3-month cycle.	Continuously improve real-time delivery
Adoption scope	Program level	One product first

Thank You