Introduction to Scrum

Version 1.1

August 2, 2018

Why Scrum is needed

Changes in business environments

In the age when anything released would sell, the structure was simple: it was a matter of **how much can be invested and how much can be made.**
A few bugs were not an issue, as the product would still sell.

Predictable	→	Difficult to predict
Competed on price	→	Competing on added value
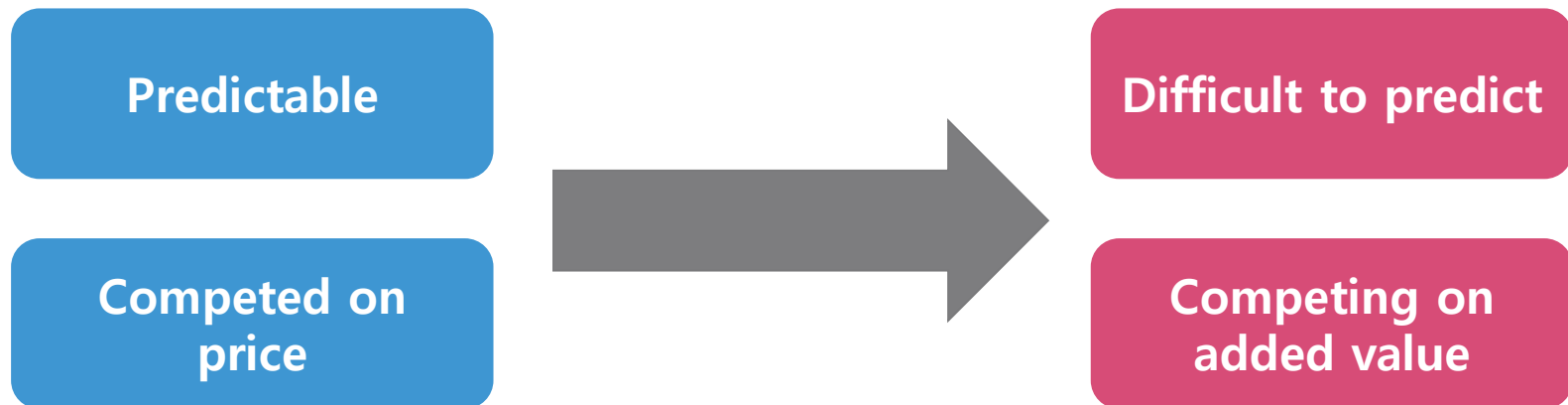
We are now dealing with an **uncertain market**, where we cannot be sure what will sell. No matter how much we invest, we cannot be sure that we will get a return on that investment. Business models are becoming increasingly short-lived.

We need to keep up with rapid changes in the market.

Definition of Scrum

Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.

Characteristics

- ✓ Lightweight
- ✓ Easy to understand
- ✓ Difficult to master

Only **19** roles and rules.
PMBOK V6 has **49** processes.

Scrum is experience-based.
Team members learn by making judgments based on practical experience and existing knowledge.

The rules for Scrum can be found in the Scrum Guide at the URL below.

https://www.scrumguides.org/scrum-guide.html

Agile, Scrum and Waterfall

Agile

Being *agile* means navigating a situation quickly and skillfully.

It is not just a practice but a **way of being**.

> " **Don't do agile, be agile** "

It is important to see agile development not as a purpose in itself but as the manner in which you approach your development.

The core principles can be summed up as **the Agile Manifesto** and **the principles behind it**.

Manifesto for Agile Software Development

In **2001**, 17 prominent figures in what was then known as lightweight software development came together to integrate their areas of focus and document the result.

The six who are best known in Japan are:

- Kent Beck (creator of XP and JUnit)
- Martin Fowler (writer of PoEAA, coined the term "microservice")
- Dave Thomas (author of books such as The Pragmatic Programmer. "Uncle Bob".)
- Robert C. Martin (Author of Clean Code, Clean Coder, etc.)
- Ken Schwaber (creator of Scrum)
- Jeff Sutherland (creator of Scrum)
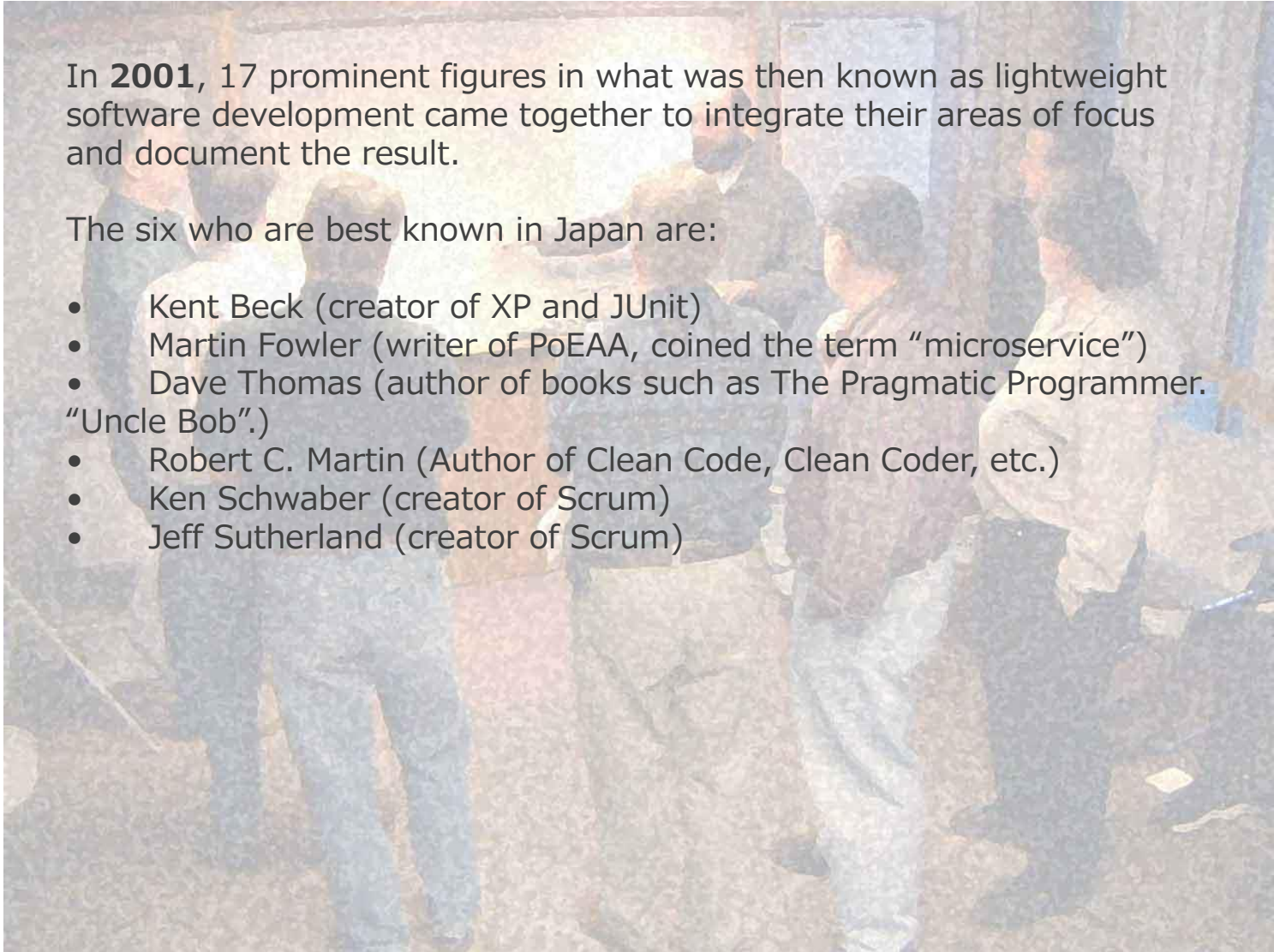
Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over **processes and tools**
Working software over **comprehensive documentation**
Customer collaboration over **contract negotiation**
Responding to change over **following a plan**

That is, **while there is value in the items on the right**, **we value the items on the left more.**

Kent Beck
Mike Beedle
Arie van
Bennekum
Alistair Cockburn
Ward Cunningham
Martin Fowler

James
Grenning
Jim Highsmith
Andrew Hunt
Ron Jeffries
Jon Kern
Brian Marick

Robert C.
Martin
Steve Mellor
Ken Schwaber
Jeff Sutherland
Dave Thomas

© 2001, the above authors
This declaration may be freely copied in any form,
but only in its entirety through this notice.

http://agilemanifesto.org

Principles behind the Agile Manifesto

We follow these principles:

Our highest priority is to **satisfy the customer**
through early and continuous delivery of **valuable software.**

Welcome changing requirements, even late in development.
Agile processes harness change **for the customer's competitive advantage**.

Deliver working software frequently, from a couple of weeks to a couple of months, **with a preference to the shorter timescale**.

Business people and developers must work together daily throughout the project.

Build projects around **motivated individuals**.
Give them the environment and support they need
and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is **face-to-face conversation.**

Principles behind the Agile Manifesto (continued)

Working software is the primary measure of **progress**.

Agile processes promote sustainable development.
The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Continuous attention to technical excellence and good design enhances agility.

**Simplicity— the art of maximizing the amount of work not done—is essential.**

The best architectures, requirements, and designs emerge from self-organizing teams.
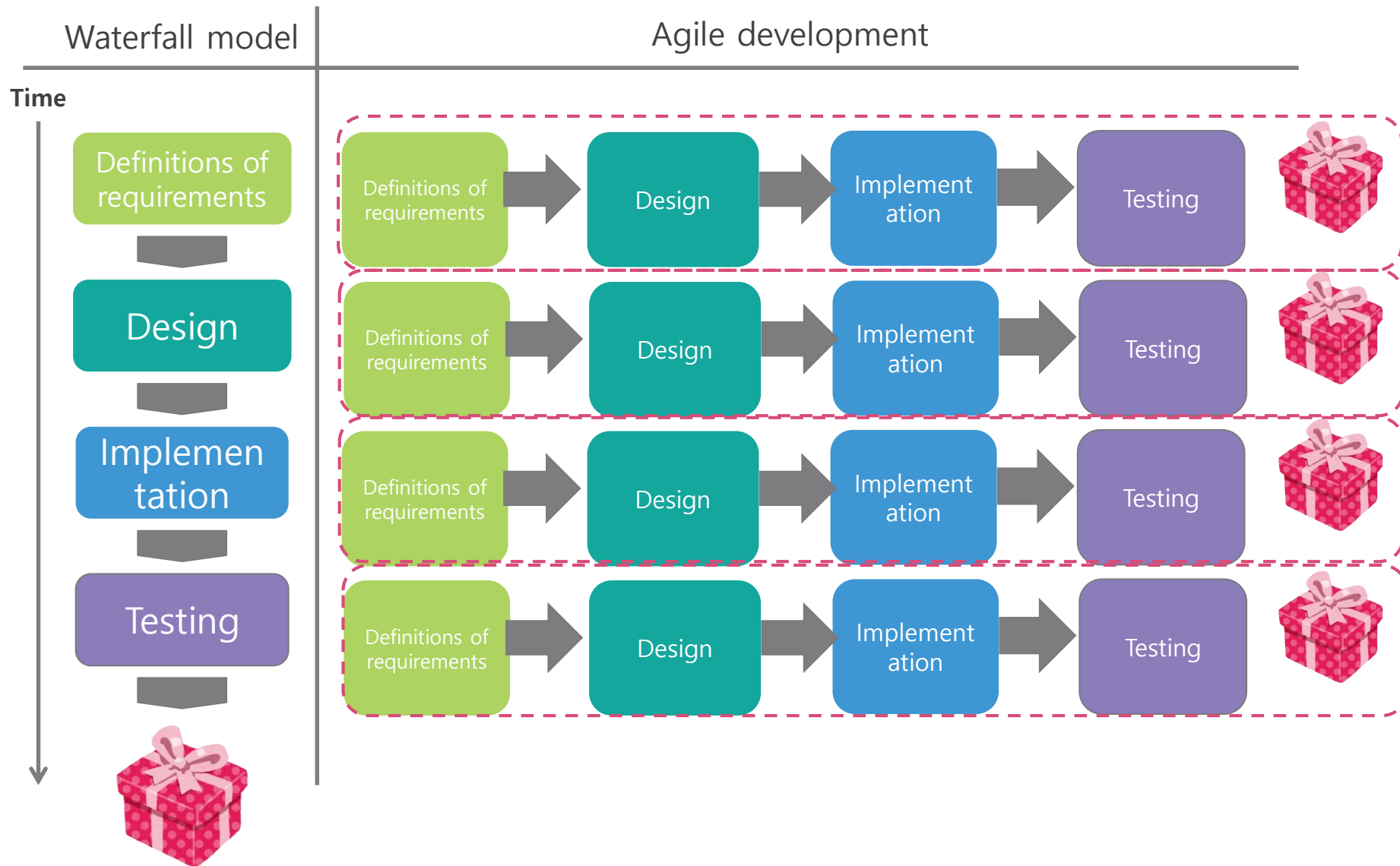
At regular intervals, the team reflects on how to become more effective,
then **tunes and adjusts its behavior** accordingly.

http://agilemanifesto.org/principles.html

10

The waterfall process

The familiar V-Model development process
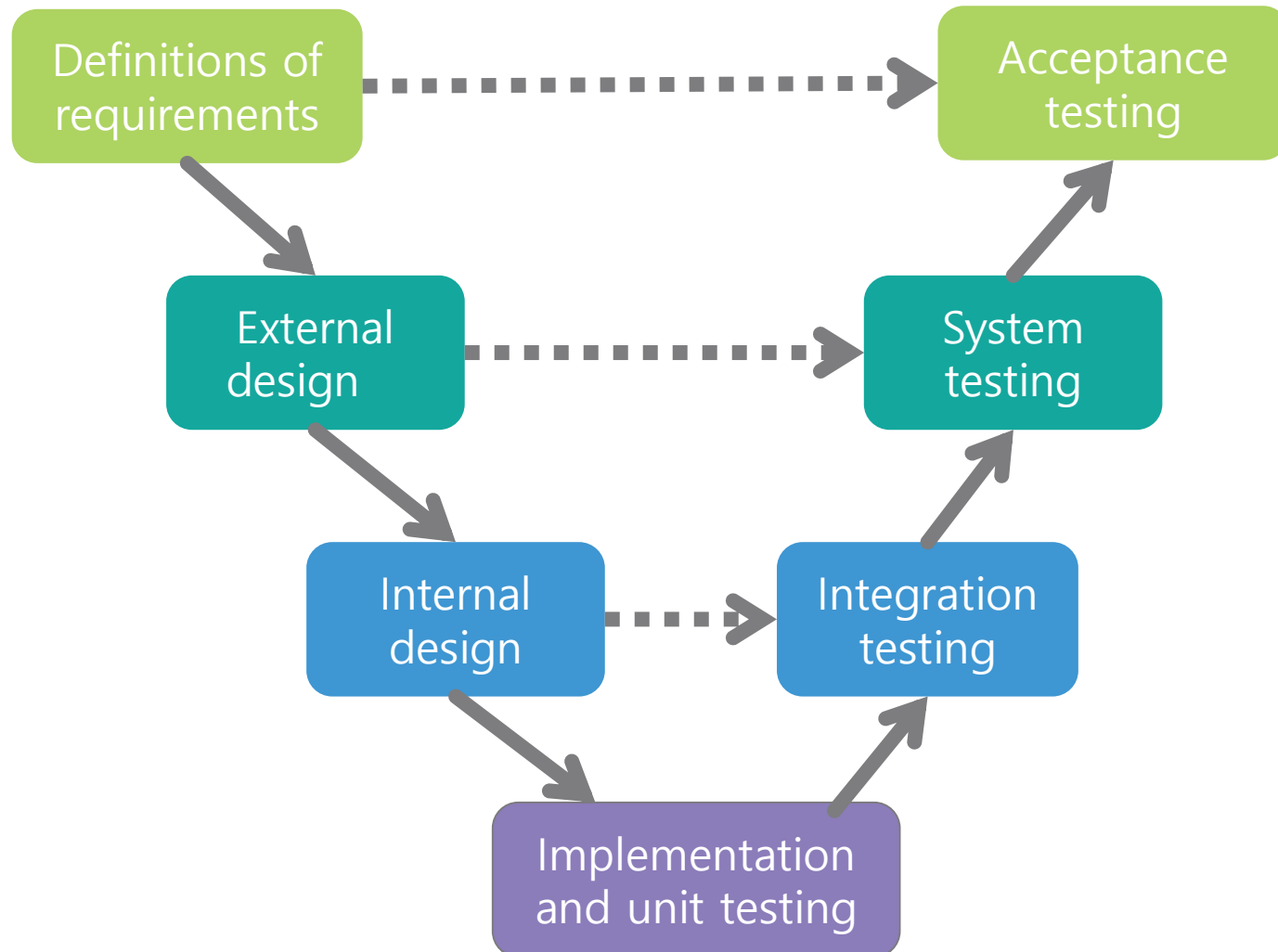
Differences in process between agile development and waterfall model

Waterfall model	Agile development

Time

Waterfall model:
- Definitions of requirements
- Design
- Implementation
- Testing

Agile development:

Definitions of requirements	Design	Implementation	Testing
Definitions of requirements	Design	Implementation	Testing
Definitions of requirements	Design	Implementation	Testing
Definitions of requirements	Design	Implementation	Testing

Differences in process between agile development and waterfall model

Process	Waterfall model	Agile development
Measurement of success	Whether processes were executed according to plan. Measurement by QCD.	Adapting to change. Customer satisfaction. Greater competitive advantage.
Management	Directive orders Top-down	Servant leadership Flat
Plan	Man-hours estimated by scope.	Man-hours estimated by schedule.
Requirements and design	Requirements identified first. All requirements designed.	Requirements received continuously. Designed at the necessary timing.
Implementation	All functions developed simultaneously.	Functions developed in order of priority.
Testing and QA	Done at end.	Testing done continuously from beginning.

Source: Scaling Software Agility
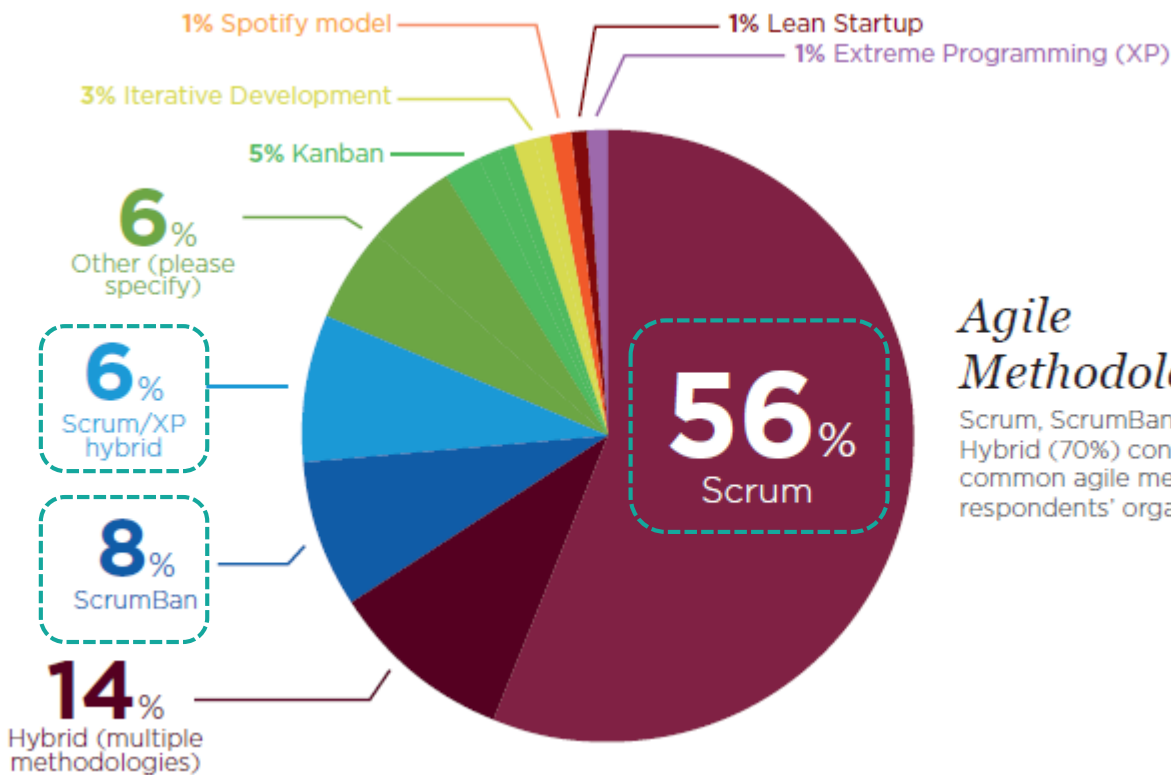
Relationship between agile and Scrum

Scrum is an element of agile.
In addition, Agile includes Kanban, which follows the flow of Lean, and XP.
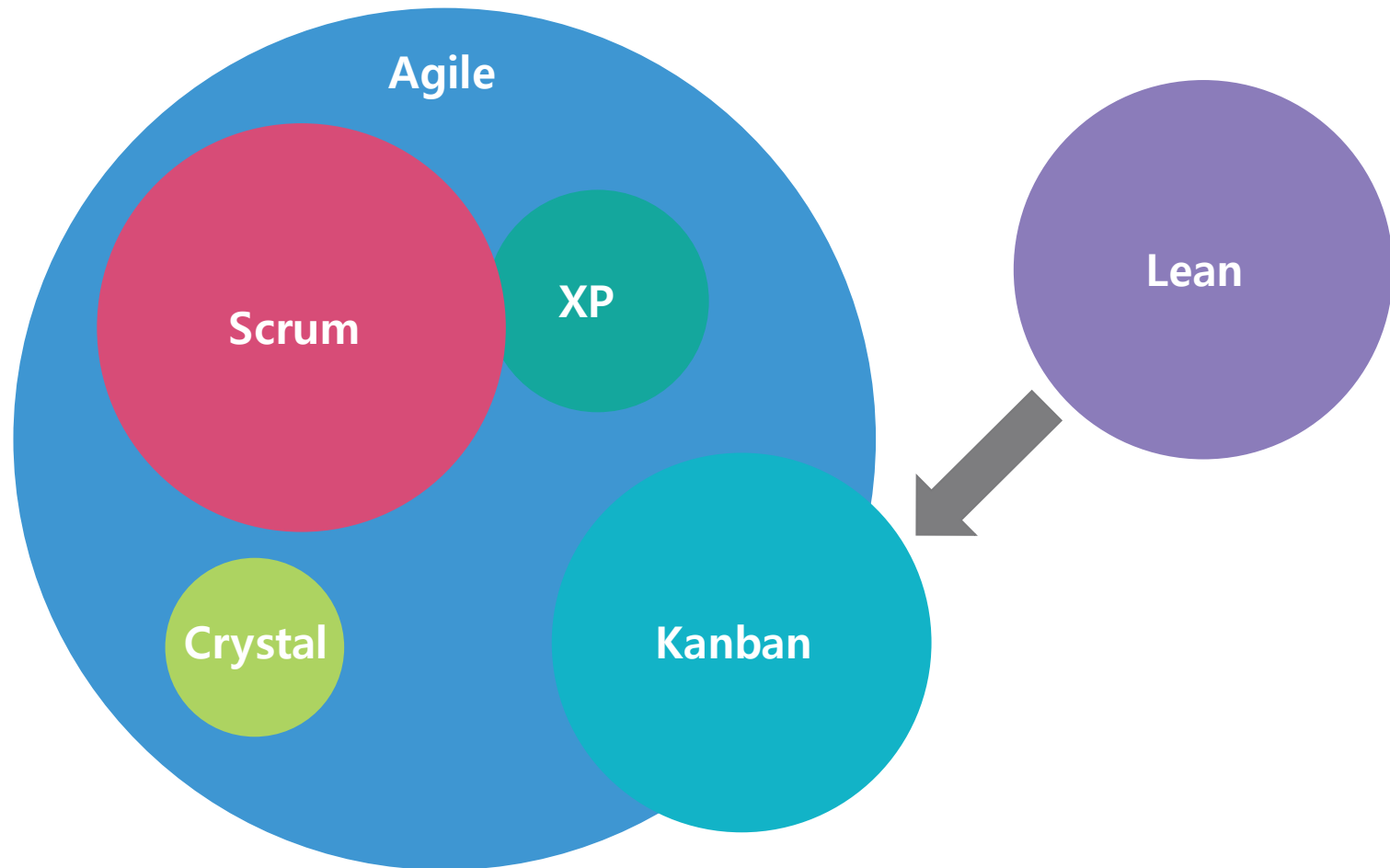
Adoption rate of Scrum in agile

Scrum has a 70% adoption rate including hybrids

VersionOne 12th Annual State of Agile Report
http://stateofagile.versionone.com/

What is Scrum?

Origin of Scrum

Scrum itself was announced by Jeff Sutherland and Ken Schwaber in **1995**, but it first appeared in The New New Product Development Game, a research paper published by **Ikujiro Nonaka** and **Hirotaka Takeuchi** in 1986.
This report likened the approach to development that had emerged in Japan to a **scrum** in rugby.

Scrum is actually older than agile.
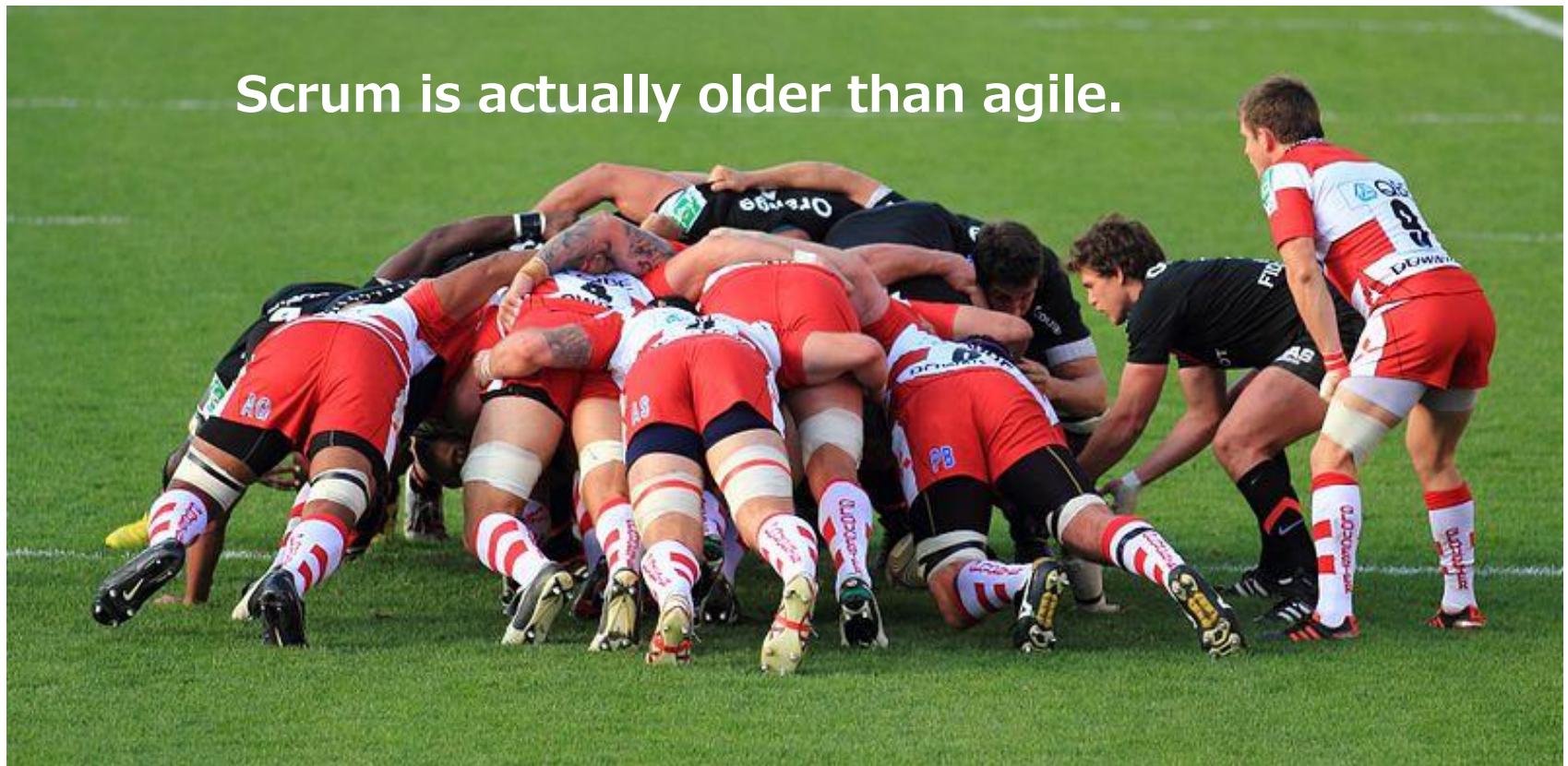
Luke Burgess introduces the ball into the scrum.
https://ja.wikipedia.org/wiki/%E3%83%95%E3%82%A1%E3%82%A4%E3%83%AB:ST_vs_Gloucester_-_Match_-_23.JPG

Definition of Scrum(reiterated)

Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.

Characteristics

✓ Lightweight
✓ Easy to understand

⟶ Only **19** roles and rules.
PMBOK V6 has **49** processes.

✓ Difficult to master

⟶ Scrum is experience-based.
Team members learn by making judgments based on practical experience and existing knowledge.

The rules for Scrum can be found in the Scrum Guide at the URL below.

https://www.scrumguides.org/scrum-guide.html

Projects that are unsuitable for Scrum

- Projects that are simple

- Projects in which a predetermined work is created

- Projects whose team life is short

  - Techniques and processes will not be developed if the period is shorter than 3 months

Requirement

Indeterminate

Chaos

Territory suited to Scrum

Simple

Fixed

Techniques

Refined   Uncertain
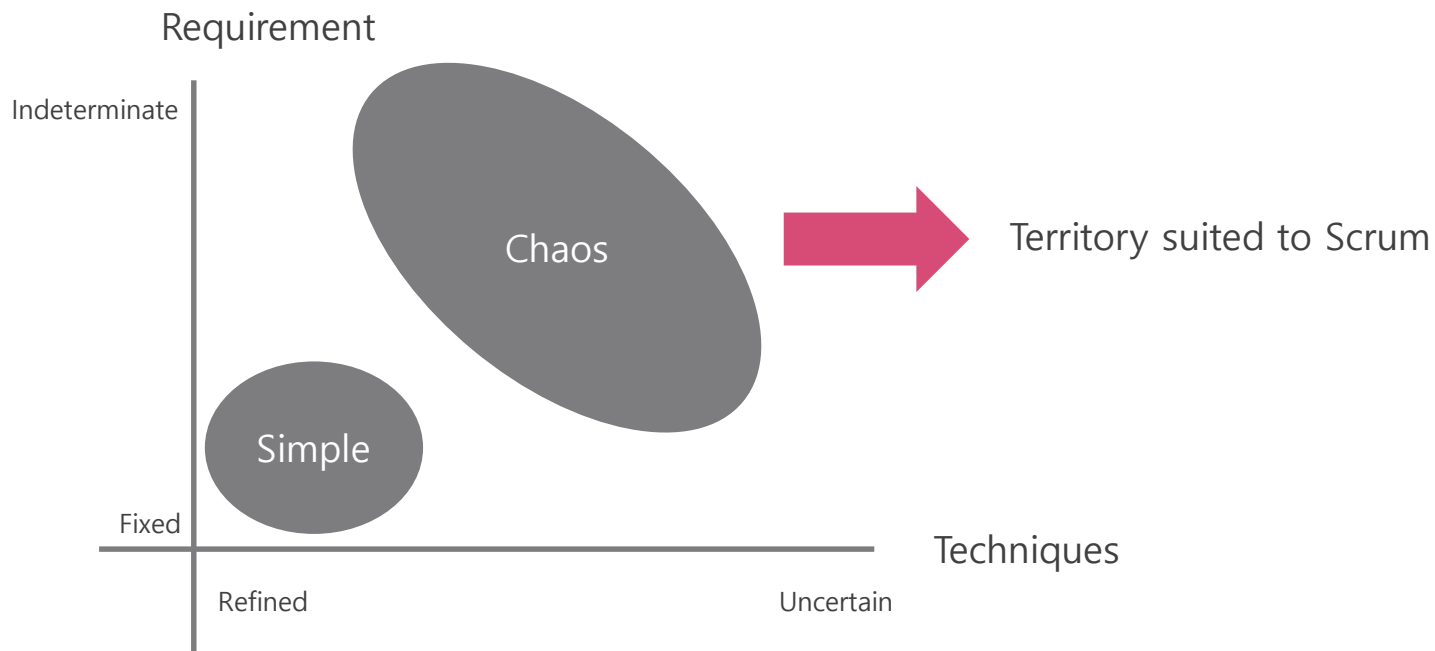
Theory of Scrum

Scrum is rooted in experience-based (empirical) process control.

Making and acting on a judgment builds **experience**

Actual experience

Knowledge

Judgment and Action

Using **repetitive, gradual** methods to **optimize predictability** and **manage risks**
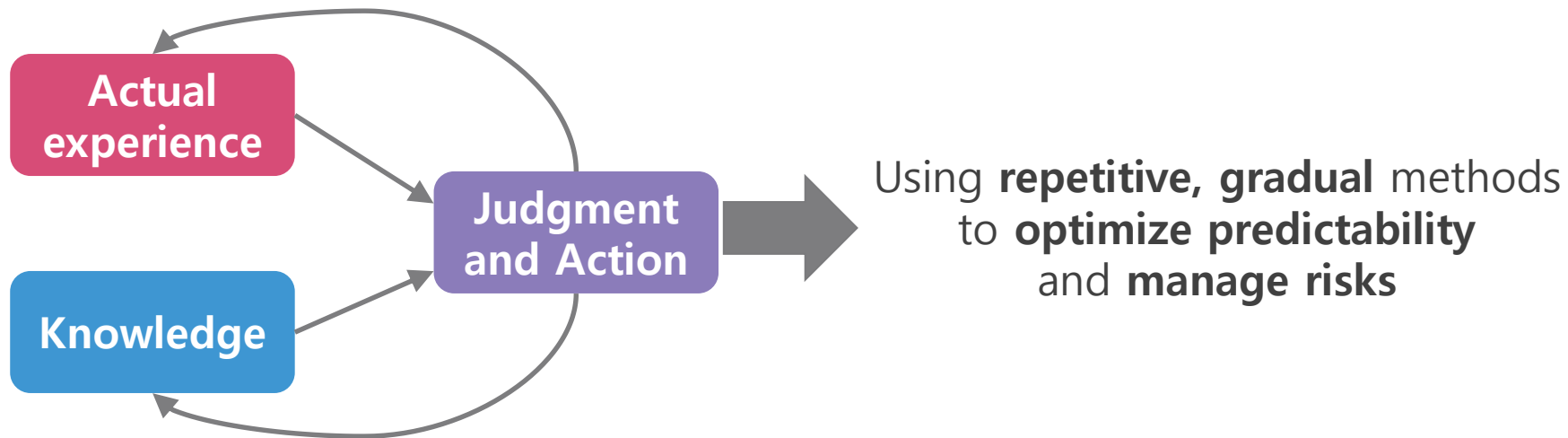
Making and acting on judgment leads to new **knowledge**

The three pillars of Scrum

The following three elements are needed to achieve experience-based process control.

Transparency
    Visible for those responsible for the results.
    Standardizing so that everyone who views the content is **on the same page**.

Inspect
    Frequent inspection of created content and progress to **detect changes.**
    With that said, inspections must not be so frequent that they impede work.

Adapt
    If a process is insufficient, adapt its components.
    Adapt processes as early as possible to prevent deviation.

One of these applies to every event, work and role in a scrum.

The Five Standards for Value in Scrum

To achieve transparency, inspection and adaptation, these five standards for value need to be adopted.

Commitment
Each **individual** must be **committed to achieving** the **goal** of the scrum team. The focus must not be on finishing at all costs.

Courage
Members of a scrum team must **have the courage to do what is right.** They must tackle difficult issues.

Focus
All members of the scrum team must **focus** on the **work in each sprint** and the **goal of the project**.

Openness
The scrum team and stakeholders must agree to be **open** about their **work**, **issues and progress**.

Respect
Members of the scrum team must **respect each other** as **skilled, independent individuals.**

The 19 rules of Scrum

- ✓ Transparency
- ✓ Inspection
- ✓ Adaptation

Product backlog refinement (5〜10%)

Sprint(1-4weeks)

Daily scrum

Daily scrum

Sprint planning: Part 1

Sprint backlog
Definition of "done"

Deliverable product

Sprint planning: Part 2

Daily scrum

Sprint review

Sprint retrospective

Development team(7±2)

Fault list

Product owner

Daily scrum

Daily scrum

Product backlog

Daily scrum

Scrum master

Cancellation of sprint
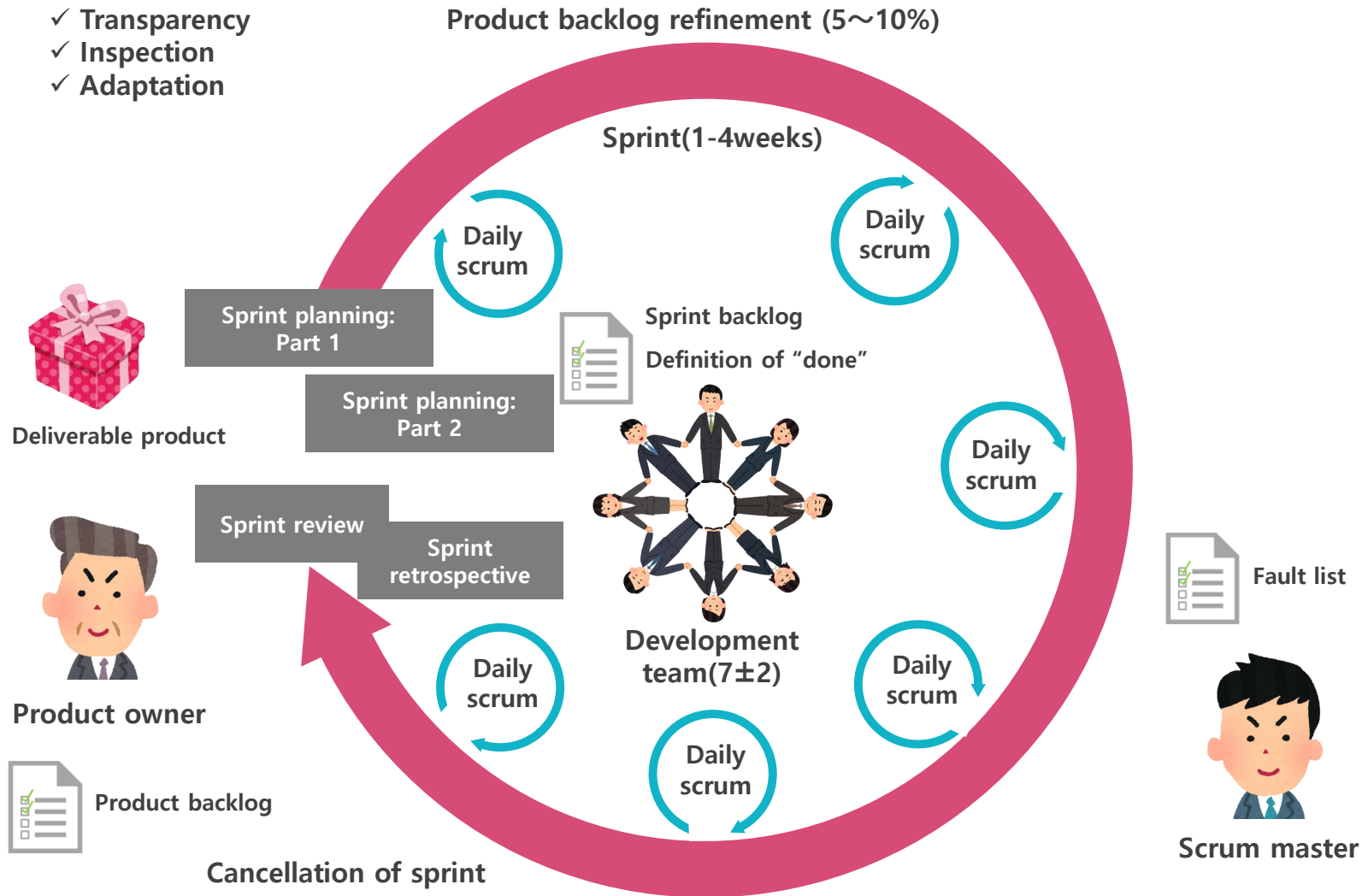
The three roles in Scrum

The three roles in Scrum

✓ **Transparency**
✓ **Inspection**
✓ **Adaptation**

Product backlog refinement (5〜10%)

Sprint(1-4weeks)

Daily scrum

Daily scrum

Sprint planning: Part 1

Sprint backlog

Definition of "done"

Deliverable product

Sprint planning: Part 2

Daily scrum

Sprint review

Sprint retrospective

Development team(7±2)

Fault list

Product owner

Daily scrum

Daily scrum

Scrum master

Product backlog

Daily scrum

Cancellation of sprint
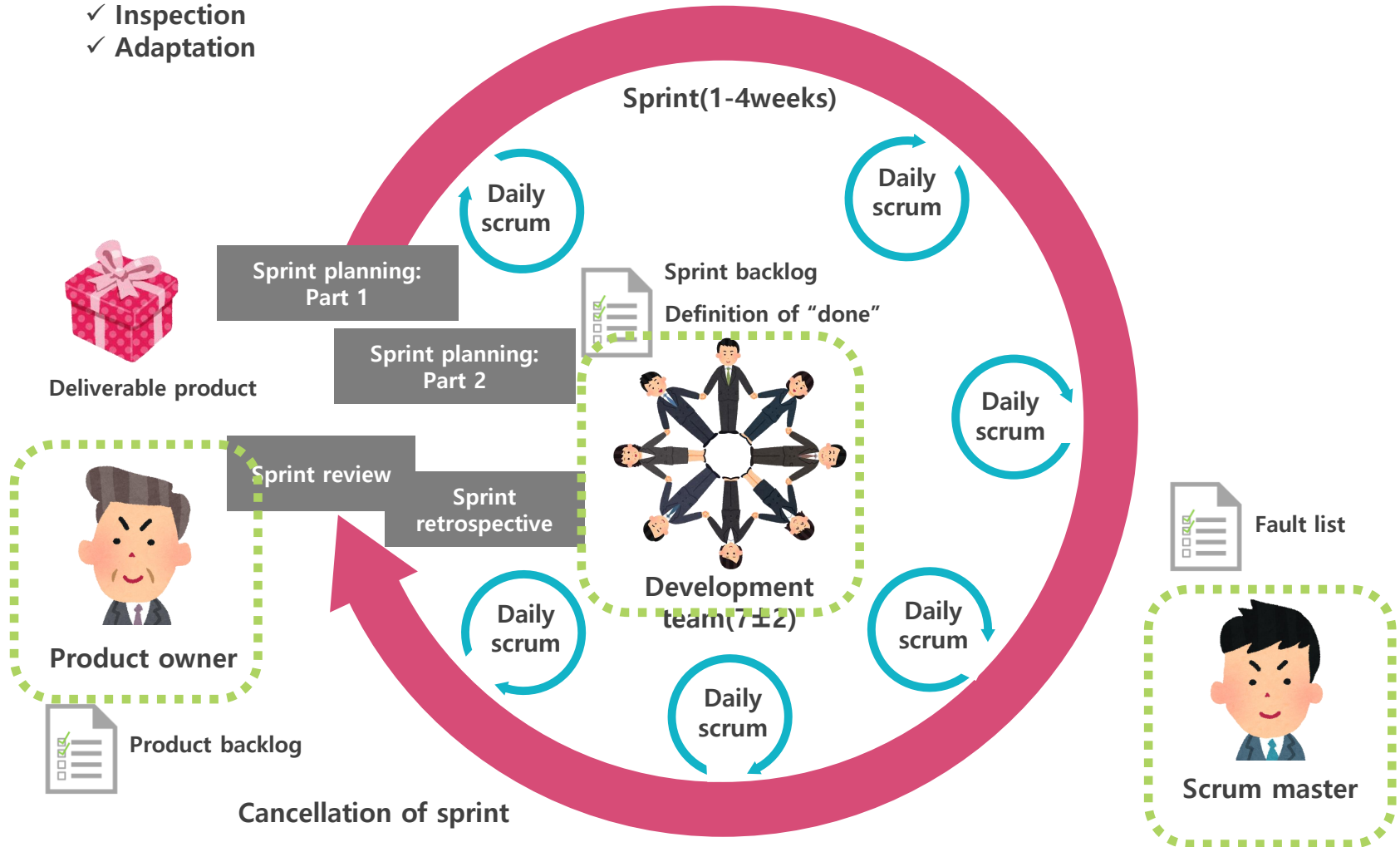
Scrum team

A scrum team comprises a product owner, development team and scrum master.
Scrum teams are **self-organized** with **cross-sectional functions**.
The **best practices** to achieve the goal of the project are **decided autonomously by the team.**

Scrum team

Responsible for explaining

Requirements

Stakeholders

Product owner

Work requests and explanations

Deliverables and questions

Development team (7±2)

Support

Support

Scrum master
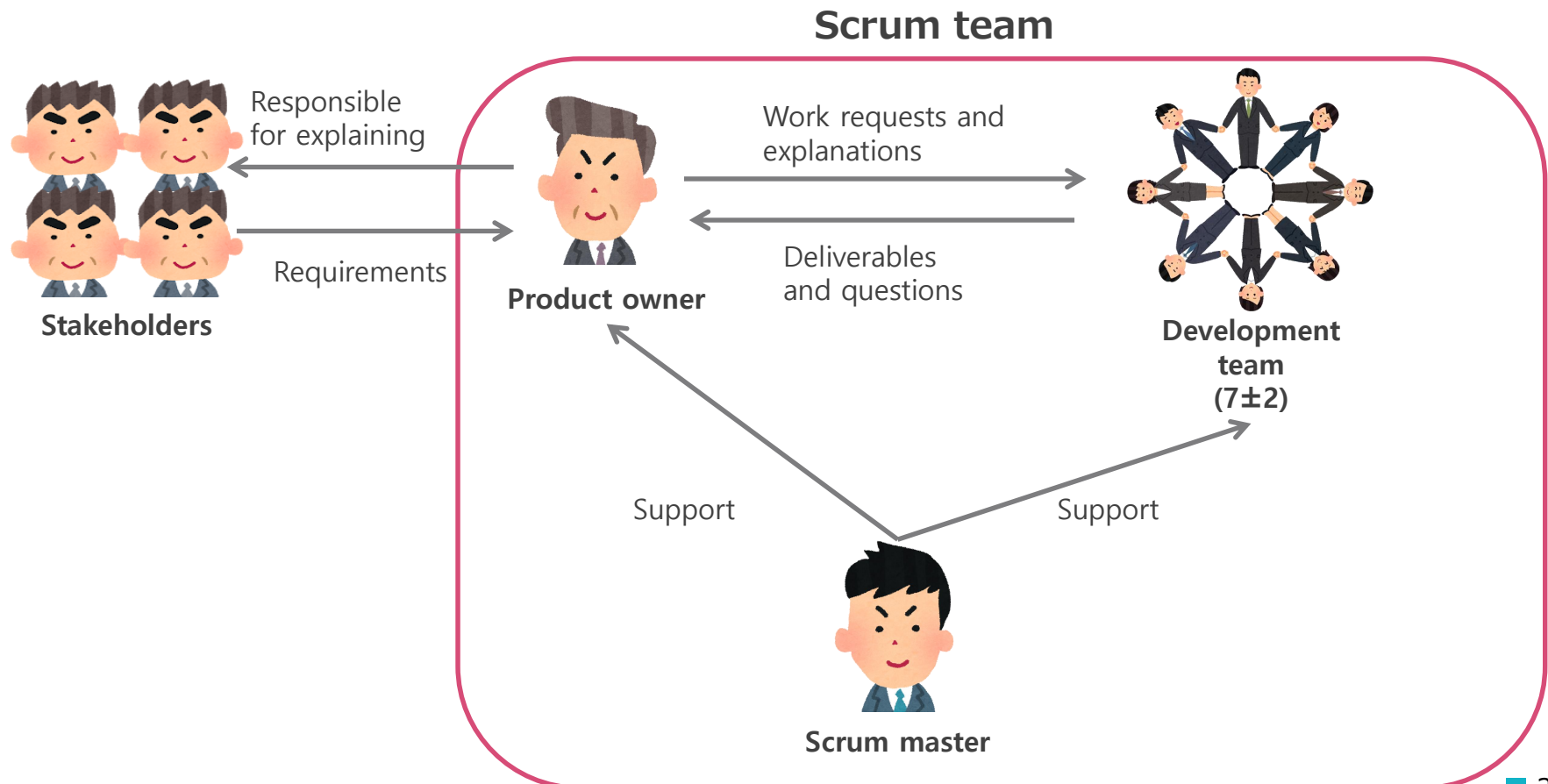
Product owner (PO)

Required skills

- Cost perception
- Consistency
- Negotiation skill
- Strategic thinking
- Ability to explain
- Decisiveness

Role

- **Responsible for maximizing the value of the development team's work and the product**
- Decides the release date and content to be released
- **Sole person responsible for management of product backlog**
- Decides priority order of product backlog (there can be a committee, but the PO is responsible for the decision)
- Work requests to development team (only the PO can request work by the development team)
- Acceptance and rejection of work results

Job

- Clearly indicates product backlog items (PBIs)
- Rearranges PBIs to achieve goal and mission
- Optimizes the value of the development team's work
- Makes PBIs visible, transparent and explicit to all team members and indicates the work that the scrum team needs to do next
- Ensures that the development team understands the PBIs at the necessary level

Development team

Required skills

- Developemnt skill
- **Autonomous organization**
- Ability to estimate
- **Cross-sectional work** skills (the collective skills of the members must be sufficient to create the product)

Role

- Creates something that can be released
- Manages the members' work **(considers and decides on best practices)**
- Responsible for **working to improve productivity**

Job

- Fulfilling promises made during sprint planning
- Consistently considering and carrying out actions to improve productivity
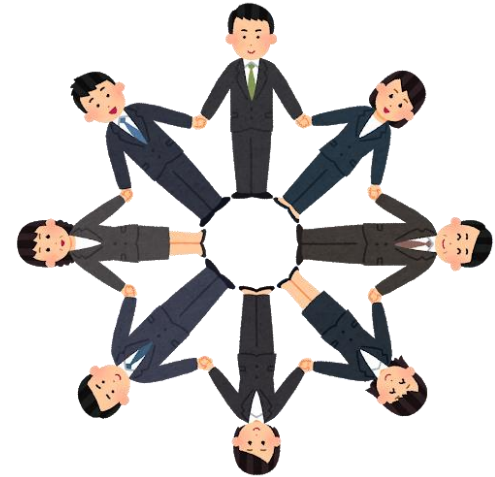
Scrum master

Required skills

- Servant leadership
- Teaching
- Facilitation
- Coaching

- Understanding of processes other than Scrum
- Understanding of group psychology
- Expressing facts (figures)

Role

- **Responsible for understanding and formation of scrums**
- Supports the product owner
- Supports the development team

Job

- Explains scrums in a way that can be understood
- Considers effective ways to manage product backlog
- Facilitates events (as necessary)
- Eliminates factors that impede the development team's progress

Scrum team anti-pattern:
Scrum master in multiple roles

Product owner

Development team(7±2)

Scrum master and member of development team
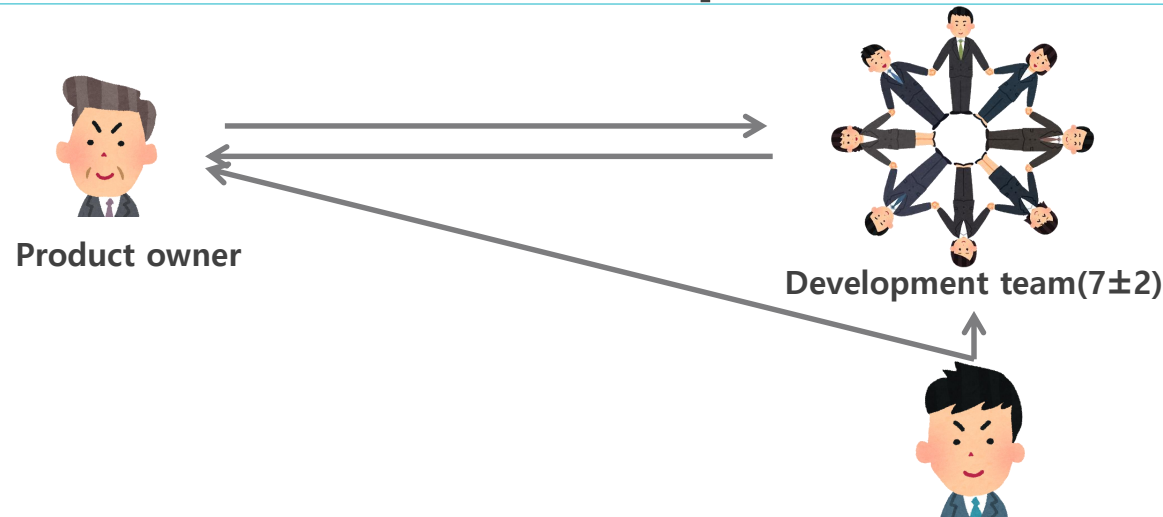
Issues

As a scrum master, sometimes this individual is strict with the development team.But in many cases, the rest of the team is unsure of whether the remarks as a scrum master or member of the development team.

This is particularly problematic in cases where the scrum master has worked as a team leader for conventional development projects, which often results in the individual gaining outstanding technical skills. In cases like this, the individual has to balance their role as a scrum master with work such as architectural judgments, design and develop, which places a large burden on them.

Solution

If the individual is an experienced scrum master, they should train other members of the team so that they can act as scrum masters or develop technical skills that can be relied on, and then step down from the role of scrum master.
If the individual is inexperienced, they should choose a scrum master from the development team and focus on development in their own work.

Scrum team anti-pattern: PO committee

Product owner committee

Development team(7±2)

Scrum master
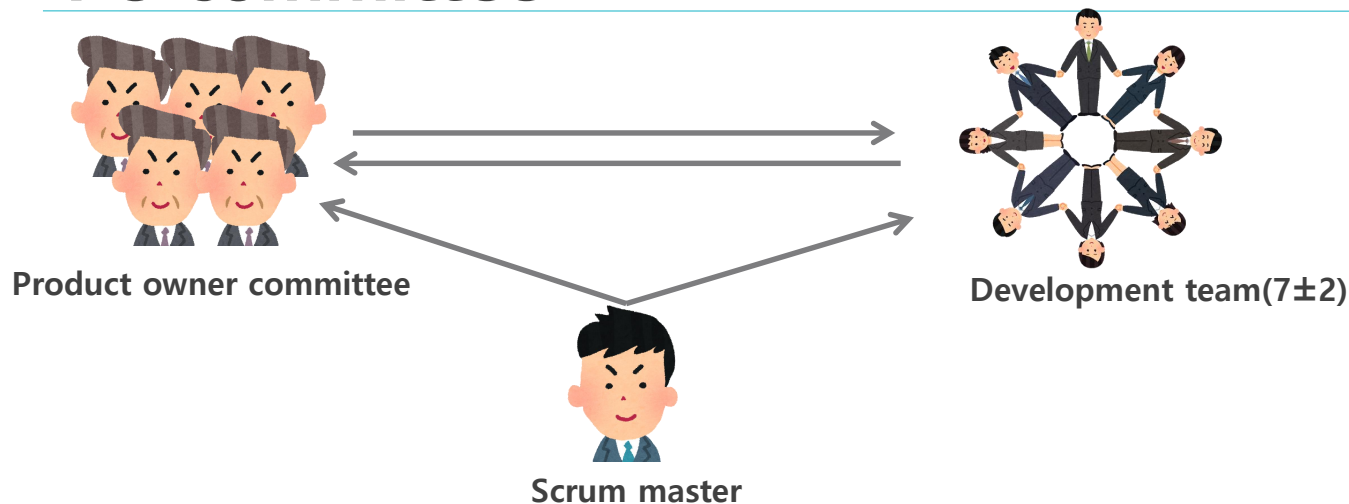
Issues

As there are multiple product owners, they will not agree on everything, which will confuse the development team.
Clashes may occur in the product owner committee, which will delay judgments.
The development team will not know who to contact with questions about specifications.

Solution

While there can be a committee, there needs to be only one product owner making decisions.
The scrum master should block work requests from parties other than the product owner.

Scrum team anti-pattern:
Stakeholders who want to be too involved

Stakeholders

Development team(7±2)

Issue

If stakeholders get actively involved, they will give the development team advice and information, but this will not be consistent with what the product owner says. They will also directly ask the development team to do work such as investigation work that will be applied to the product backlog.

This will lead to an increasing amount of work and information that the product owner is not aware of, leading to decreases in the velocity of the product and causing tasks that should be prioritized to be put off.

Solution

The scrum master should block direct requests from stakeholders and give explanations that ensure that the product owner's decisions are heard.
The product owner should collect information from stakeholders and provide them with explanations.

The three items produced by Scrum

The three items produced by Scrum

✓ **Transparency**
✓ **Inspection**
✓ **Adaptation**

Product backlog refinement (5～10%)

Sprint(1-4weeks)

Daily scrum

Daily scrum

Daily scrum

Daily scrum

Daily scrum

Daily scrum

Daily scrum

Sprint planning: Part 1

Sprint planning: Part 2

Sprint review

Sprint retrospective

Deliverable product

Sprint backlog

Definition of "done"

Development team(7±2)

Product owner

Product backlog

Cancellation of sprint

Fault list

Scrum master
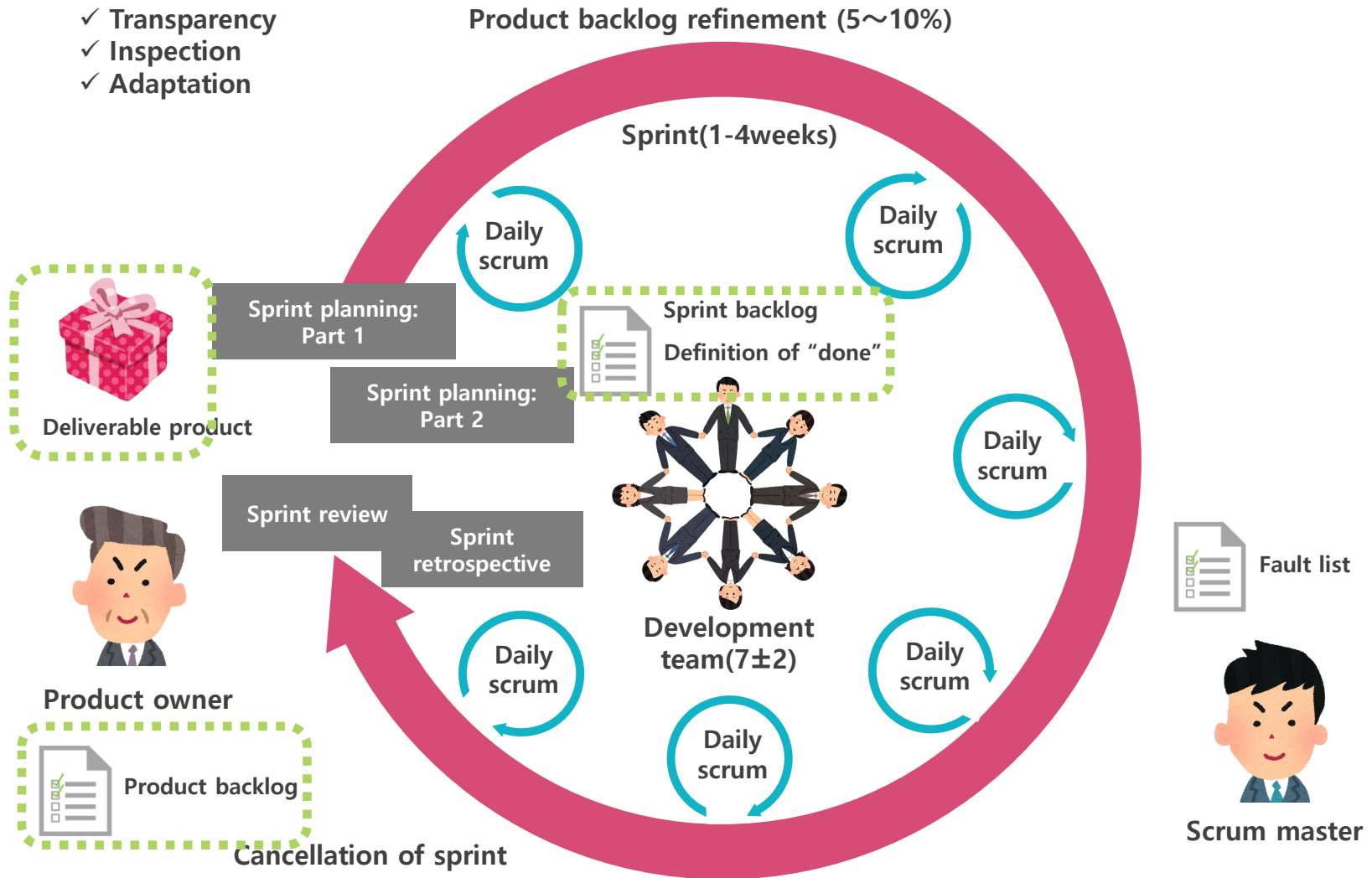
Product backlog

A list of everything required for the product, **in order of priority**.
The only source of information on changes that need to be made in the project.
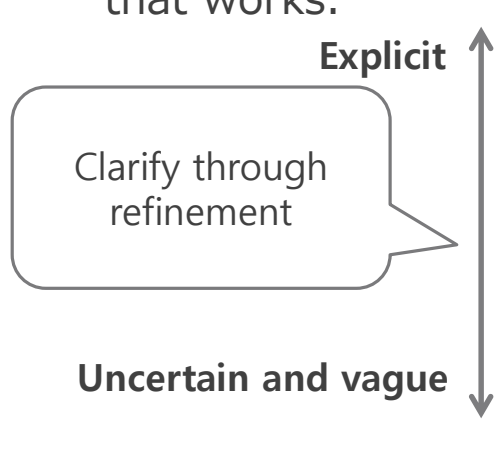Each element in the product backlog is called a product backlog item (PBI).
The product backlog is constantly changing according to factors such as the business requirements, market situation and technical changes.

The process of making changes to the details, estimates and order of the PBIs is called **product backlog refinement.**
The timing of refinements is decided by the scrum team.
Backlog refinement tends to account for 10% or less of the development team's work.

Estimates are **relative**. The values can be a Fibonacci sequence, S/M/L or anything that works.

Explicit

Uncertain and vague

Clarify through refinement

Priority	Story	Estimate
1	Complete XX as A.	2
2	Make YY referenceable in list form as B.	3
3	Improve performance of process C	5
…	…	…
100	Create report as D	8

Cone of uncertainty

This expresses the transition of variance in estimates as the project progresses.

Project
schedule

At the beginning, the maximum estimate is 16 times the minimum.

- **Uncertainty decreases with each sprint**
- **If a mistake occurs, simply make adjustments early**

4.0x	
2x	
1.5x	
1.25x	
1.0x	
0.8x	
0.67x	
0.5x	
0.25x	

Initial concept↵ Approved product definitions↵ Completion of requirements↵ Completion of user interface design↵ Completion of detailed design↵ Completion of software

Time

Confirmation of progress toward goal

Resources such as a release burndown chart can be used to track progress and confirm that the desired content will be released at the desired time. Confirming this during sprint reviews makes it possible to provide the product owner with materials for scope adjustment.
If you know the number of points required for release and the timing that is required, it is possible to tell whether a delay is occurring from one sprint.
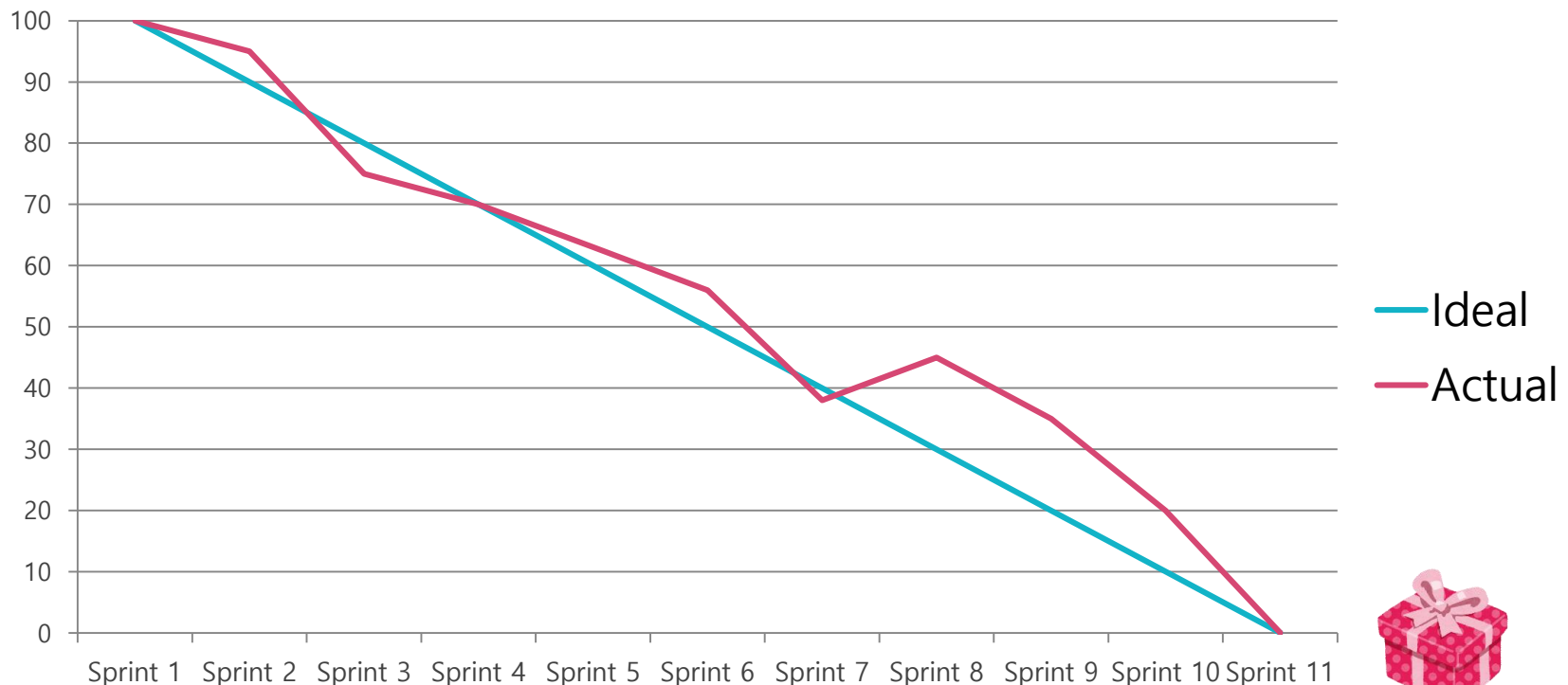
Sprint backlogs

Sprint backlogs are plans used to ensure that the product backlog items selected for a sprint can be submitted as deliverable products.
They are **lists of the work that the development team must do to achieve the sprint goals.**
They are written in the level of detail that is needed, and may be changed during the sprint.
Only the development team can change the sprint backlog.

Estimates indicate the **ideal time**.
Tasks should be broken into small units, as this enables more accurate estimation.
At a maximum, tasks should be small enough to be completed within the time allocated for development work in one day.

Story	Task	Estimate
Complete XX as A	UI coding	3.0h
	Data model design, changes, entity creation	3.0h
	Action coding	2.0h
Make YY referenceable in list form as B.	UI coding	4.0h
	Action coding	2.0h

Confirmation of progress of sprints

The work remaining in the sprint backlog is tracked to confirm progress.
The burndown chart is often used to check progress in daily scrums.

Legend:
- Ideal
- Actual

X-axis: Wed, Thu, Fri, Mon, Tue, Wed, Thu, Fri, Mon, Tue
Y-axis: 0, 5, 10, 15, 20, 25, 30
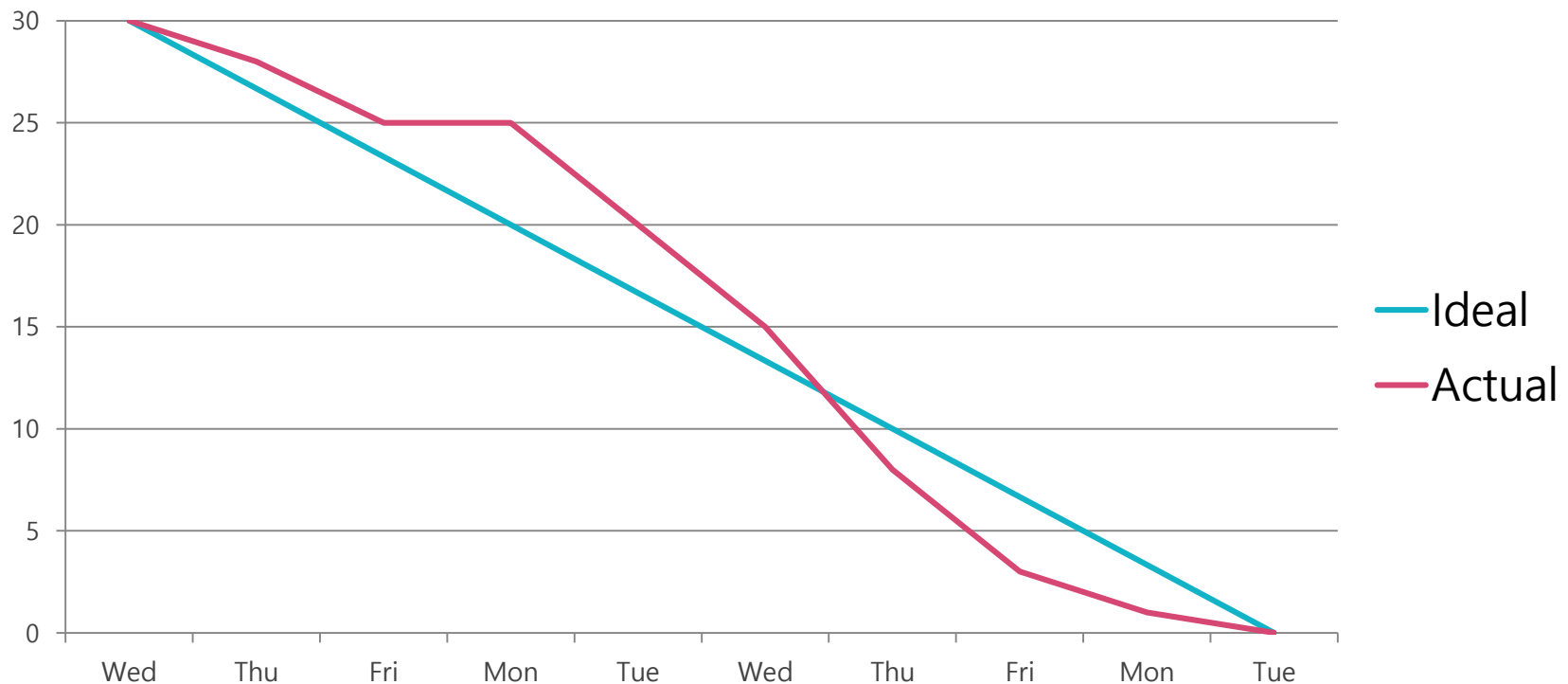
Deliverable products

This term encompasses the products created so far and the product backlog items that have been completed in the current sprint.
Deliverable products must be completed by the end of the sprint.
Products that meet the **definition of "done"** set by the scrum team are deliverable products.

Parts alone cannot be delivered. **Usable items, no matter how small**, need to be created.

Transparency of created items

Definition of "done"

The definition of "done" indicates the point at which a product is deliverable. This is used to evaluate whether work is completed. This needs to be done before releasing products.

Done

All sprints are completed

- Development
- UT
- 80% coverage or greater
- IT
- Regression testing
- Document creation
- • Static analysis

Undone

Not all sprints have been executed

- Scenario testing
- Load testing
- Security testing
- Release

It is important to get undone tasks done

The five events in Scrum

The five events in Scrum

- ✓ Transparency
- ✓ Inspection
- ✓ Adaptation

Product backlog refinement (5～10%)

Sprint(1-4weeks)

Daily scrum

Daily scrum

Sprint planning: Part 1

Sprint backlog

Definition of "done"

Daily scrum

Sprint planning: Part 2

Deliverable product

Sprint review

Sprint retrospective

Development team(7±2)

Daily scrum

Fault list

Product owner

Daily scrum

Daily scrum

Product backlog

Daily scrum

Scrum master

Cancellation of sprint
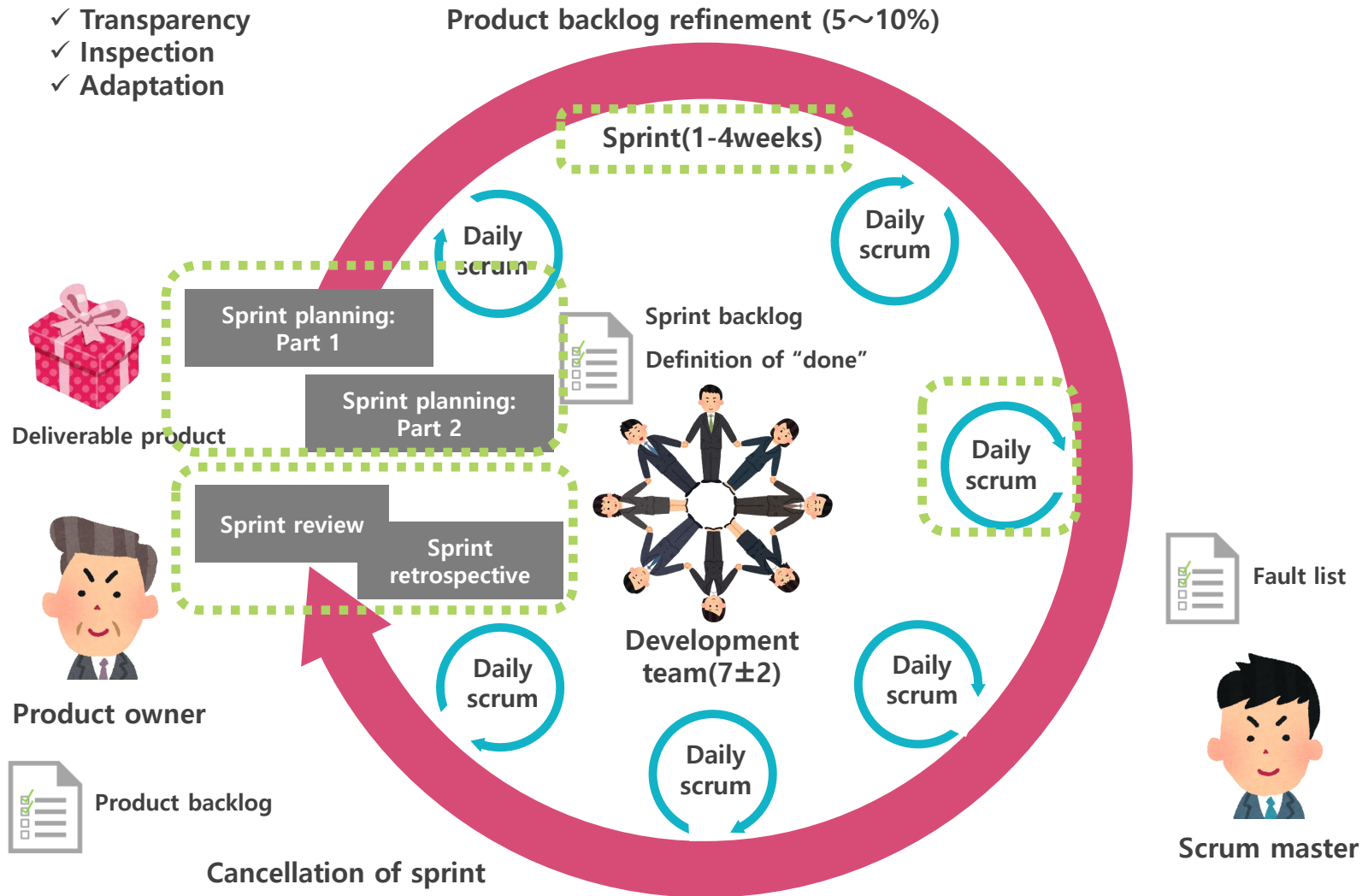
Sprint planning

Time box:　　4 hours every 2 weeks

Sprint planning: Part 1
This answers the question **"What can be delivered as an additional component of the deliverable product that will be created in the sprint?"**
The input consists of the product backlog, latest products and the predicted capacity and previous record of the development team.
This is used to determine the number of product backlog items to be selected. **The development team is responsible for deciding on the number of items.**

After predicting the product backlog that can be delivered during the sprint, **sprint goals are set**.
Sprint goals are **objectives for the sprint** **that need to be achieved by carrying out the tasks in the product backlog.**

Priority	Story	Estimate
1	Complete XX as A	2
2	Make YY referenceable in list form as B.	3
3	Improve performance of process C	5
...	...	...
100	Create report as D	8

Select PBIs to be executed during the sprint

Sprint planning

Sprint planning: Part 2

This part answers the question **"How will we complete the work that is needed in order to deliver this additional component of the deliverable product?"**
The product backlog items selected for execution in part 1 are placed in the sprint backlog.

This is when tasks for the first few days of the sprint are **broken down** to a workable size. The tasks may also be broken down during the sprint period as necessary.

The product owner helps to clarify the product backlog and make tradeoffs.
If there is too much or too little work after breaking down the tasks, the product owner is consulted to make adjustments.

The development team must be able to explain how the sprint goals will be achieved.

Story
Complete XX as A
Make YY referenceable in list form as B.

Break down into specific tasks

Task	Estimate
UI coding	3.0h
Data model design, changes, entity creation	3.0h
Action coding	2.0h
UI coding	4.0h
Action coding	2.0h

Sprints

Time box: Maximum 1 month

A continuous period of development work with a time box no larger than one month to create a deliverable product.
Sprints comprise sprint planning, daily scrums, development work, sprint reviews and sprint retrospectives.

Note

- Changes that will adversely affect the sprint goals cannot be made
- Quality targets must not be lowered
- Negotiation between the product owner and development team may become necessary as studying progresses and the scope is clarified.

Cancellation of sprints

Sprints can be canceled before the end of the time box, but **only the product owner has the authority to cancel a sprint.**
If a sprint is canceled, the completed items from the product backlog are reviewed.

Uncompleted backlog items are returned to the product backlog after a re-estimation.

Daily scrums

**Time box:**   15 minutes per day

During daily scrums, the work performed after the previous daily scrum is inspected and a plan is made for work to be performed by the next daily scrum.

This is a time to **inspect the progress** of work from the sprint backlog.
Daily scrums need to be held **at the same time and in the same place every day**.

Each member of the development team explains the following points during daily scrums.

- **What they did the previous day** to enable the development team to **achieve the sprint goals**
- **What they will do on this day** to enable the development team to **achieve the sprint goals**
- **Issues** that they or the development team faced that **affected their ability to achieve the sprint goals**

If a detailed point is raised during a daily scrum, all or some of the development team meets after the daily scrum to discuss the matter in detail and then adapt and re-plan.
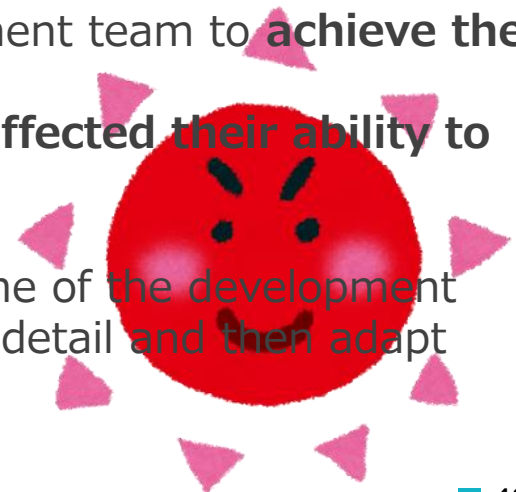
Sprint reviews

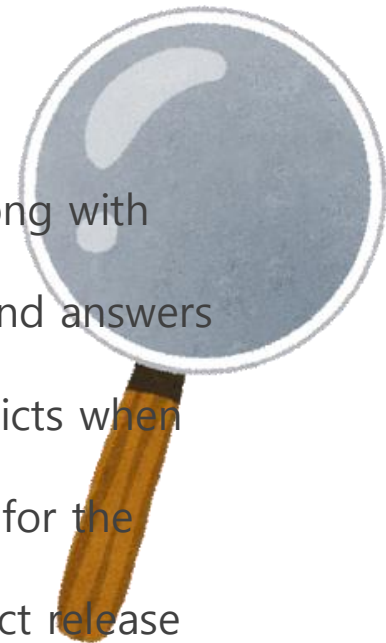**Time box:** 2 hours every 2 weeks

At the end of the sprint, the addition to the deliverable product is inspected and the product backlog is adapted if necessary.
During sprint reviews, the scrum team and other relevant personnel review the results of the sprint.

The purpose of sprint reviews is not to confirm progress but to elicit feedback and further cooperation.

Sprint reviews include the following:
- ✓ The product owner invites participants
- ✓ The **product owner** explains completed and uncompleted PBIs
- ✓ The development team discusses strong points from the sprint, along with issues that they faced and how they solved them
- ✓ The development team demonstrates what they have completed and answers questions
- ✓ The product owner confirms the current product backlog and predicts when it will be completed
- ✓ The whole group discusses what to do next and uses this as input for the planning of the next sprint
- ✓ The schedule, budget, performance and market for the next product release is reviewed

Sprint retrospectives

Time box: 1.5 hours every 2 weeks

These are an opportunity for the **scrum team** to conduct an inspection and create a plan for improvements to be made in the next sprint.
Sprint retrospectives are held after the sprint review, before the planning of the next sprint.

- The sprint is inspected from the perspectives of people, relationships, processes and tools
- Strong points and points requiring improvement in future are identified and organized
- Improvement measures to be applied in the next sprint are identified
- An improvement plan for the scrum team's work is created
- The definition of "done" is adjusted as appropriate

No specific method is required, but KPT is often used.
Information on practical usage of KPT is provided in the Practical Guide to Project Facilitation: Retrospective Guide[1].

Information on other methods is provided in Agile Retrospectives: Making Good Teams Great[2].

[1]: http://objectclub.jp/download/files/pf/RetrospectiveMeetingGuide.pdf (only Japanese)

[2]: http://shop.ohmsha.co.jp/shopdetail/000000001770/
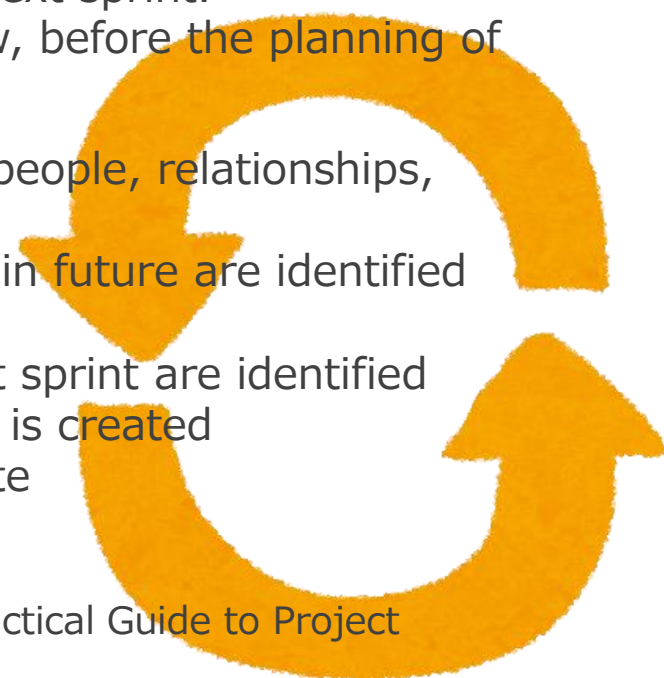
Meetings and Attendees

Guideline on meeting attendees	Sprint planning: Part 1	Sprint planning: Part 2	Daily scrums	Sprint reviews	Sprint retrospectives
Product owner Responsible for maximizing the value of the development team's work and the product. Responsible for management of product backlog.	A	A[1]	C[2]	A	C[4]
Scrum master Responsible for understanding and formation of scrums.	A	A	C[3]	A	A
Development team Develops the product. Responsible for fulfilling promises made during sprint planning.	A	A	A	A	A
Stakeholders Interested parties such as product users, investors and/or managers	D	B	D	D	B

A:Must attend
B:Cannot attend
C:May attend depending on the context
D:Can attend but does not have speaking rights

Meetings and Attendees (continued)

[1] Involvement of product owner in sprint planning: Part 2
The product owner does not need to attend sprint planning: Part 2, as this meeting is to discuss practical methods and make a plan to achieve them.
With that said, the product owner must answer questions when needed.
The development team also needs to provide the product owner with an explanation of how they will achieve the sprint goals.

[2] Involvement of product owner in daily scrums
Useful in cases such as those where obstacles for achieving the sprint goals occur frequently and support from the product owner is needed.

[3] Involvement of scrum master in daily scrums
The scrum master asks the development team to hold daily scrums, but the development team itself is responsible for holding them.
At the beginning of a team's work, the scrum master may also need to ensure that time boxes are met and rules are followed.
The scrum master does not need to be involved if the development team can run daily scrums well by themselves.

[4] Involvement of product owner in sprint retrospectives
Retrospectives are held without the PO if there are issues related to the inspection and improvement of processes that cannot be described freely if the PO is present.

Scrum and quality

Quality issues in waterfall

Quality is guaranteed by phase gates, but it is not possible to tell whether the desired product was created until acceptance testing.

The major risks are at the end.

Definitions of requirements	Good
Design	Good
Implementation	Good
Integration testing	Normal
Acceptance testing	Bad
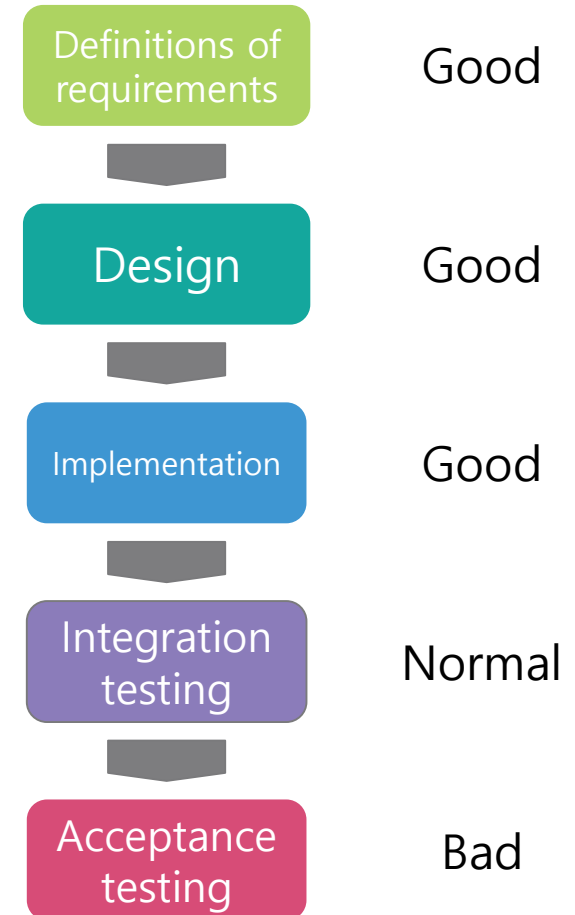
Quality in Scrum

The degree to which the product meets business requirements is confirmed during sprint reviews

Time

Definition of requirements	→	Design	→	Implementation	→	Testing	🎁	Bad
Definition of requirement	→	Design	→	Implementation	→	Testing	🎁	Normal
Definition of requirement	→	Design	→	Implementation	→	Testing	🎁	Good
Definition of requirement	→	Design	→	Implementation	→	Testing	🎁	Good

The Kano model

Customer satisfaction

Satisfied

"I like it"

Attractive Quality

"I can tolerate it"

Physical performance

Inferior performance

Superior performance

One-dimensional Quality (performance)

Must-be Quality (basic quality)

"I expect it"

"I dislike it"

Dissatisfied

The Kano model

Customer satisfaction

Satisfied

"I like it"

Scrum makes it easy to increase attractive and unified quality

Unique functions

"I can tolerate it"

Physical performance

Inferior performance

Superior performance

Throughput, response time, operation rate, etc.

"I expect it"

Whether the function works correctly/as expected

Must-have quality is the same as waterfall

"I dislike it"

Dissatisfied

QCD + S

- Quality : Fixed
- Cost : Fixed
- Delivery : Fixed
- Scope : **Adjustable**

Automation of testing

Testing is done more frequently than with waterfall, as development is done repeatedly. Automation is essential for purposes such as regression testing of functions added in the previous sprint.
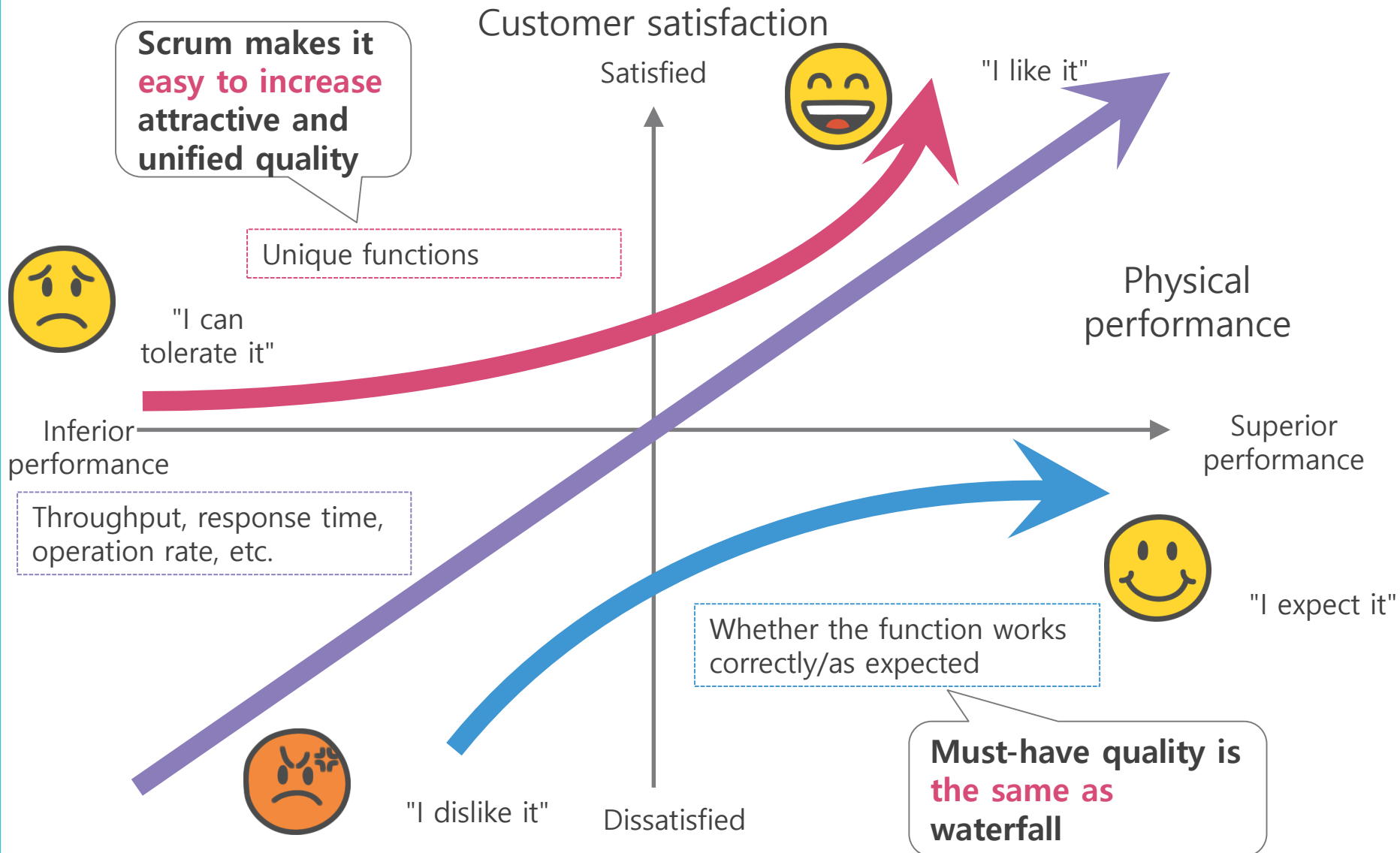
Time

| Definition of requirements | Design | Implementation | Testing |

| Definition of requirement | Design | Implementation | Testing |

| Definition of requirement | Design | Implementation | Testing |

| Definition of requirement | Design | Implementation | Testing |

Points such as the types of tests to be automated are decided by the scrum team, as these points are dependent on context.

Continuous integration and delivery

As with testing, automating this is recommended as frequent build and deployment work is required.

Metrics of Scrum

Velocity

Velocity is the total of completed story points for one sprint.
It indicates the **productivity** of the development team.
This is used as a reference when deciding the capacity during sprint planning.

Sprint 1	Sprint 2	Sprint 3	Sprint 4
3	2	5	5
1	3	1	5
5		1	

Completed stories

9	5	7	10
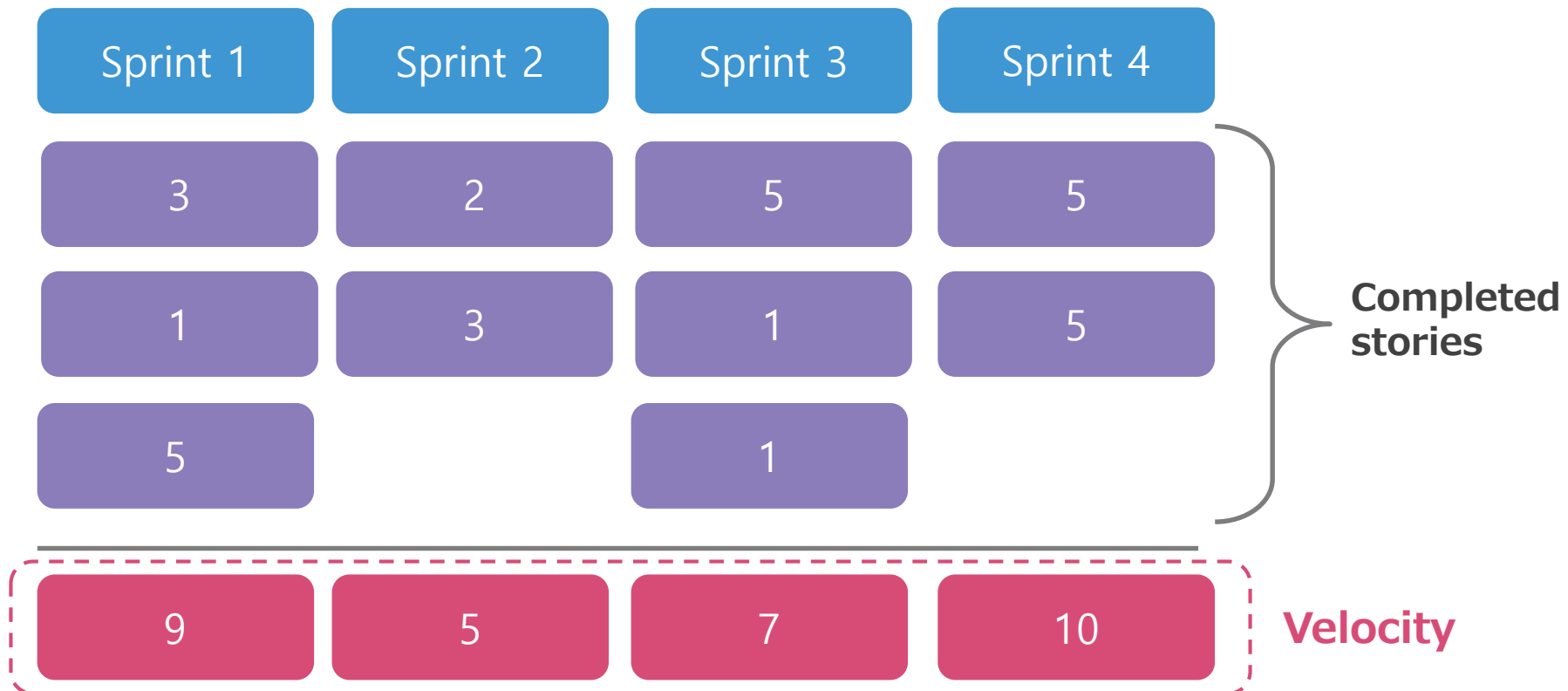
Velocity

Using velocity

Determining whether a release is behind schedule

Release points ≦ velocity (expected or performance so far) x remaining sprints

Example: 100 ≦ 10 points × 10 sprints → on schedule
100 ≧ 9 points × 10 sprints → behind schedule

Calculating capacity

Capacity = average effective velocity for 5 sprints

The effective velocity is calculated by removing
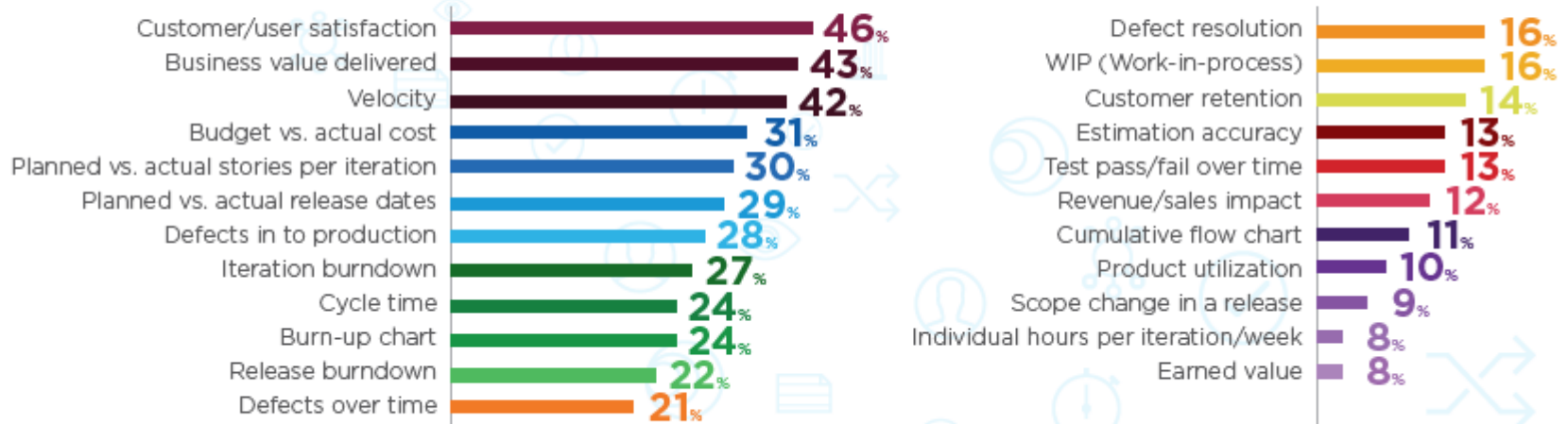the first velocity and the maximum and
minimum velocity so far.

If five sprints have not been completed,
the most recent velocity is used for the effective velocity.
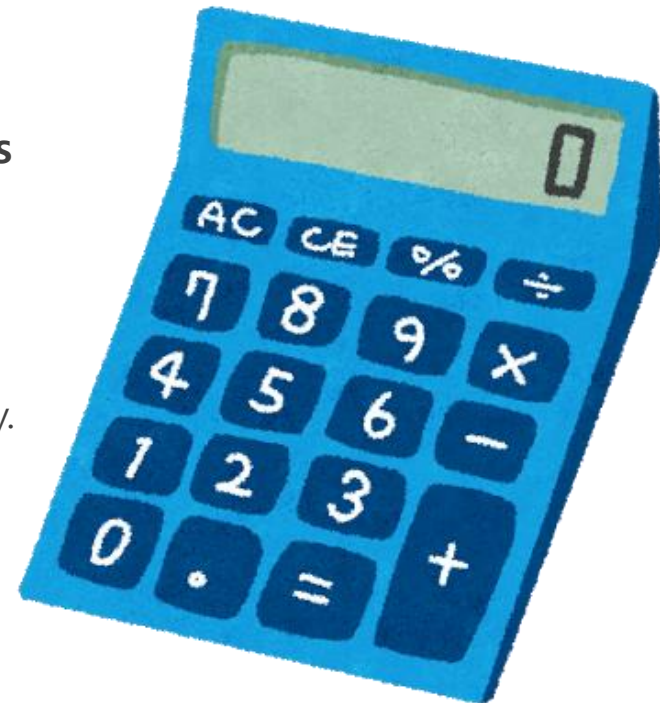
How success is measured in agile projects

How Success Is Measured...
with Agile Projects

Business value increased as a cited measure of agile project success from 23% in 2016 to 43% in 2017. Customer/user satisfaction increased from 28% in 2016 to 46% in 2017 while velocity had been the number one measure of an agile project's success decreased from 67% in 2016 to 42% in 2017. Iteration burndown also went down from 2016 (51%) to 2017 (27%).

Measure	%
Customer/user satisfaction	46%
Business value delivered	43%
Velocity	42%
Budget vs. actual cost	31%
Planned vs. actual stories per iteration	30%
Planned vs. actual release dates	29%
Defects in to production	28%
Iteration burndown	27%
Cycle time	24%
Burn-up chart	24%
Release burndown	22%
Defects over time	21%
Defect resolution	16%
WIP (Work-in-process)	16%
Customer retention	14%
Estimation accuracy	13%
Test pass/fail over time	13%
Revenue/sales impact	12%
Cumulative flow chart	11%
Product utilization	10%
Scope change in a release	9%
Individual hours per iteration/week	8%
Earned value	8%

*Respondents were able to make multiple selections.

VersionOne 12th Annual State of Agile Report
http://stateofagile.versionone.com/

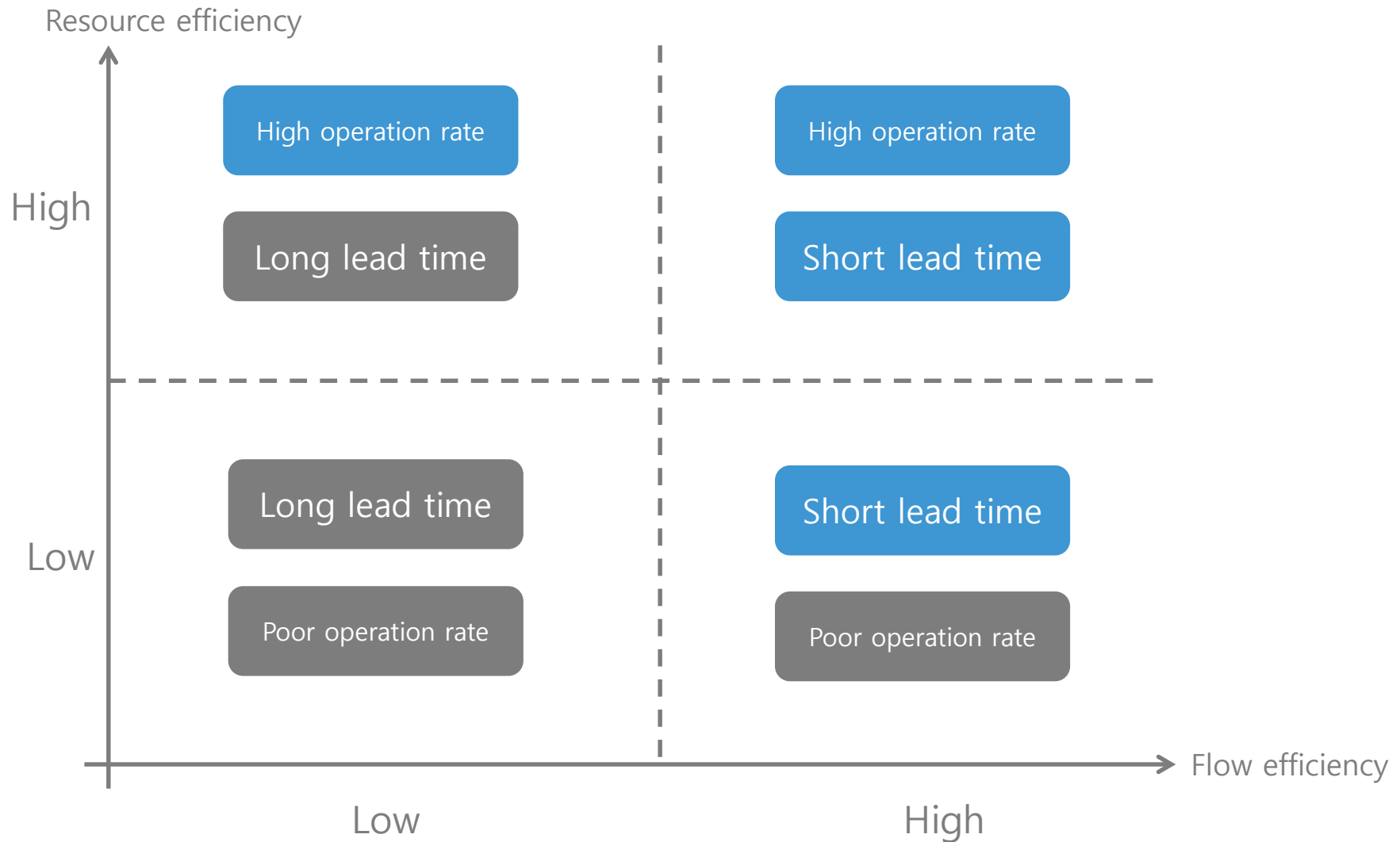# Common misunderstandings about Scrum and agile

Scrum is more productive

When making a predetermined item, waterfall is more productive. Scrum is best suited for complex situations such as those where rapid changes occur and the correct answer can only be determined by gauging the reaction of the market and users. Scrum enables a shorter lead time and faster market release.
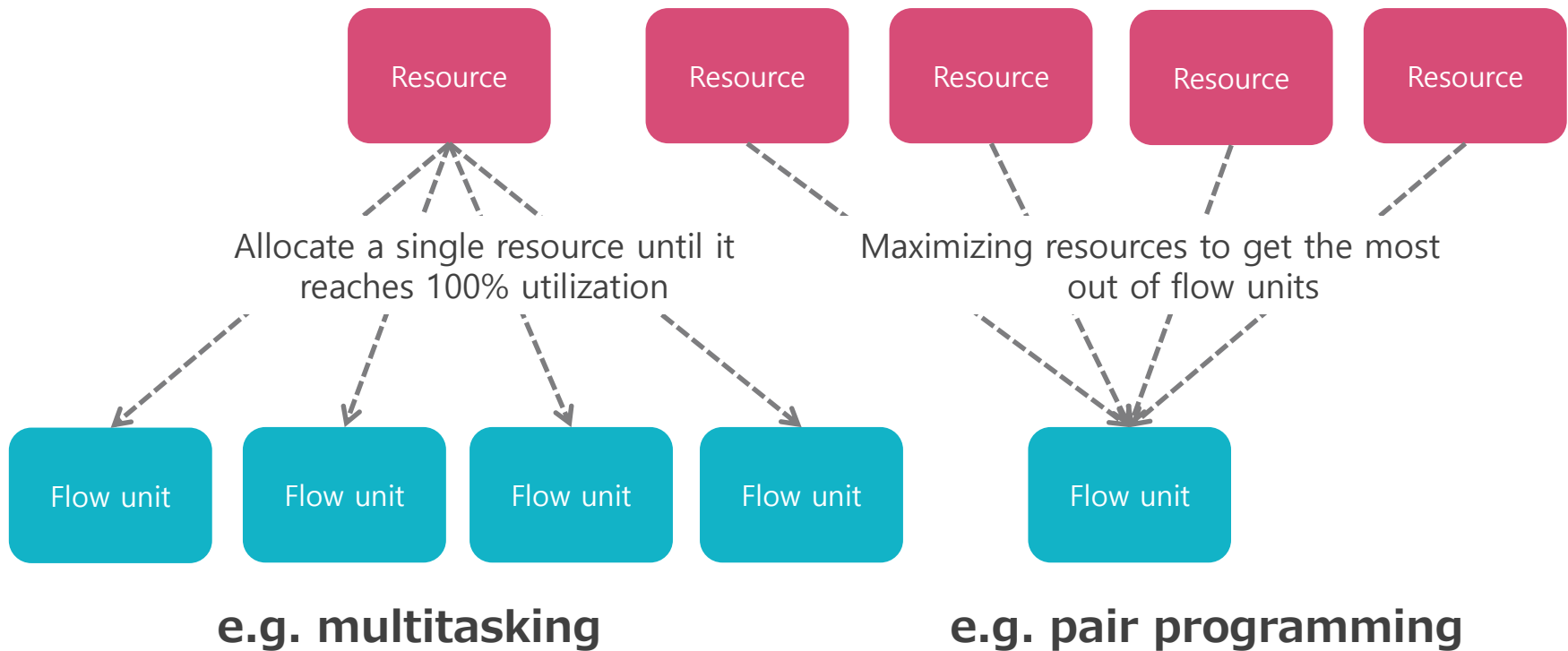
Lead time until market release = P + (S + Q + W)

Element	Definition	System development
Setup time	The preparation time.	The time required for planning and design
Process time	Processing and operation time	The time required for implementation and delivery
Queue time	The wait time caused by resource restrictions	Time during which a task is not handled as all team members are working
Wait time	The wait time for dependent tasks	Time spent waiting for dependent modules and delivery of items to be tested

Resource efficiency and flow efficiency
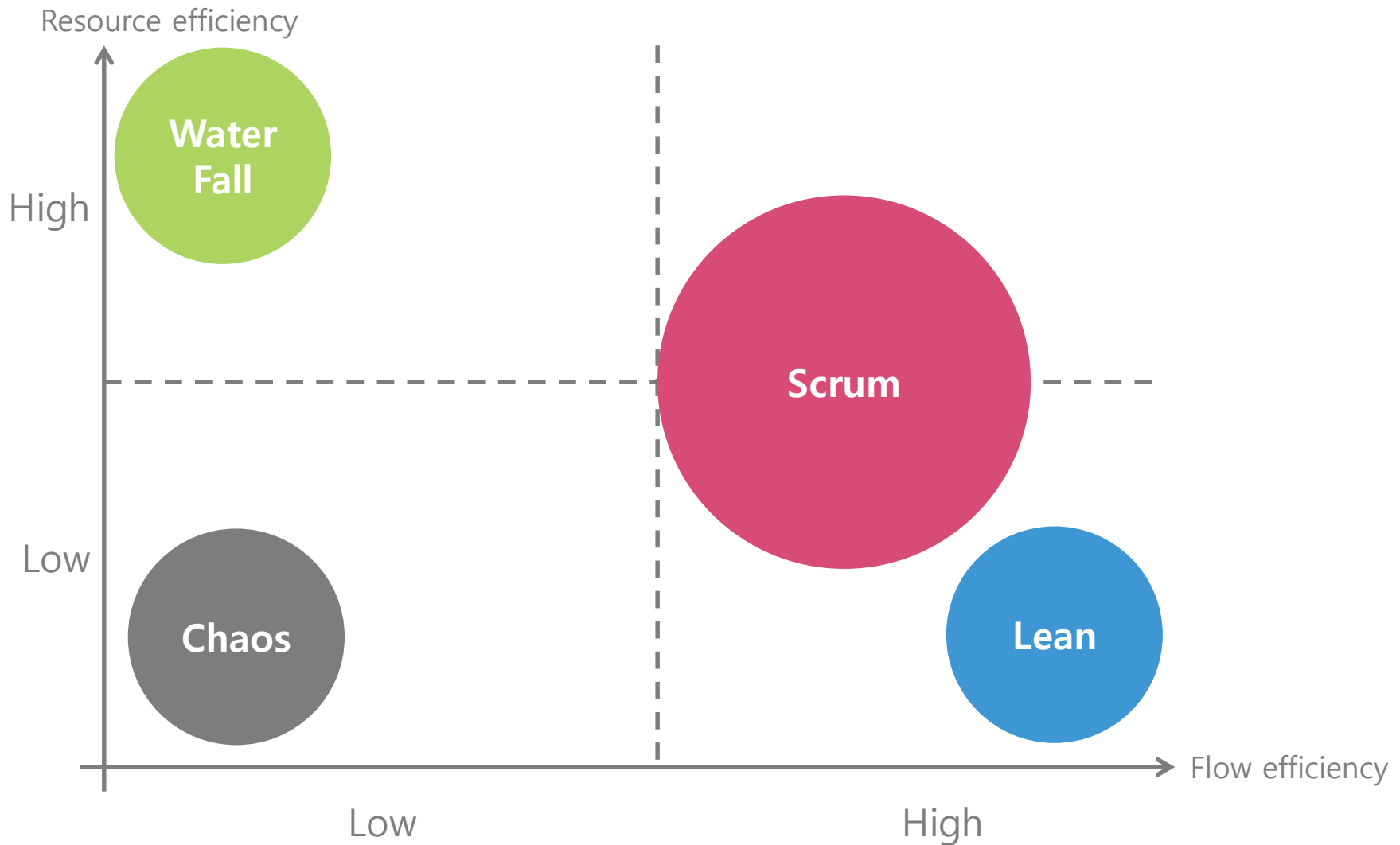

Resource efficiency

High

High operation rate

Long lead time

High operation rate

Short lead time

Low

Long lead time

Poor operation rate

Short lead time

Poor operation rate

Low   High

Flow efficiency

Resource efficiency and flow efficiency

| Resource | Resource | Resource | Resource | Resource |

Allocate a single resource until it reaches 100% utilization

Maximizing resources to get the most out of flow units

| Flow unit | Flow unit | Flow unit | Flow unit |

| Flow unit |

e.g. multitasking

e.g. pair programming

Resource efficiency and flow efficiency

Scrum is not suited to large-scale projects
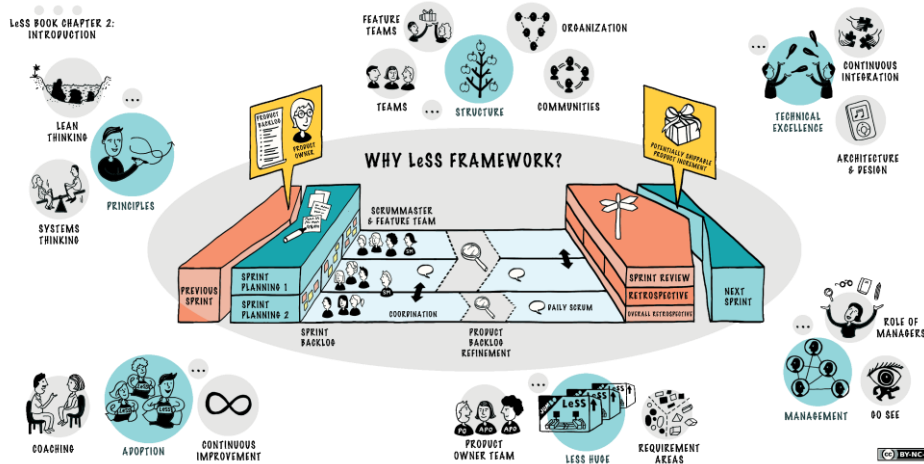
https://less.works/less/homepage?preferred_lang=en

Suited to development in teams of 2-8
One scrum master can manage 1-3 teams
One product backlog and one product owner per product
Same sprint cycle for each team

The LeSS Huge framework can be used for teams with more than 8 members

This is used for agile frameworks at a company level.
Retrospectives are held in 8- to 12-week cycles.
There are 5-15 teams (50-125 people) for each release.

http://www.scaledagileframework.com/
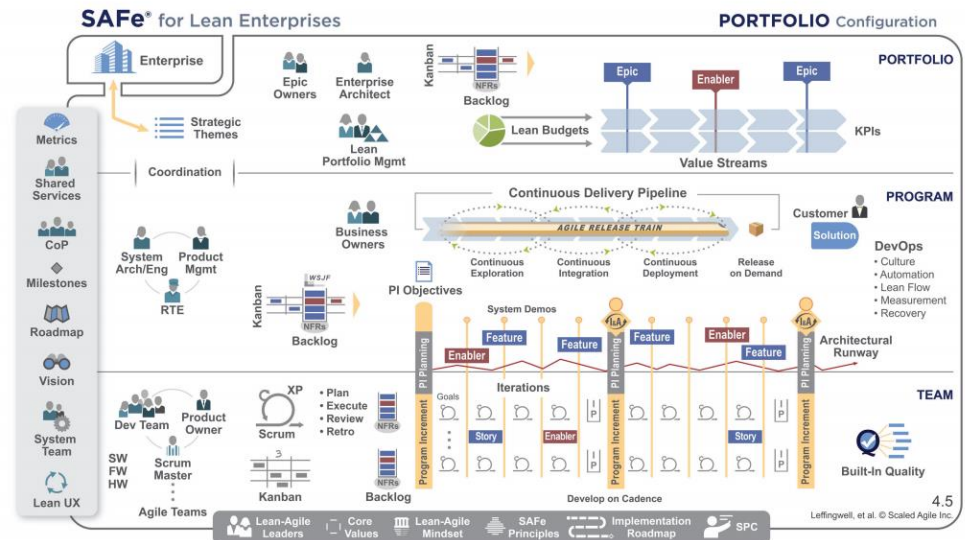
Documents are not created in Scrum

Scrum values working software over comprehensive documentation, although the necessary documents are, of course, created.

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Scrum produces lower quality than waterfall

The test types and test cases are the same, regardless of the development process, so there is no difference in the basic quality.
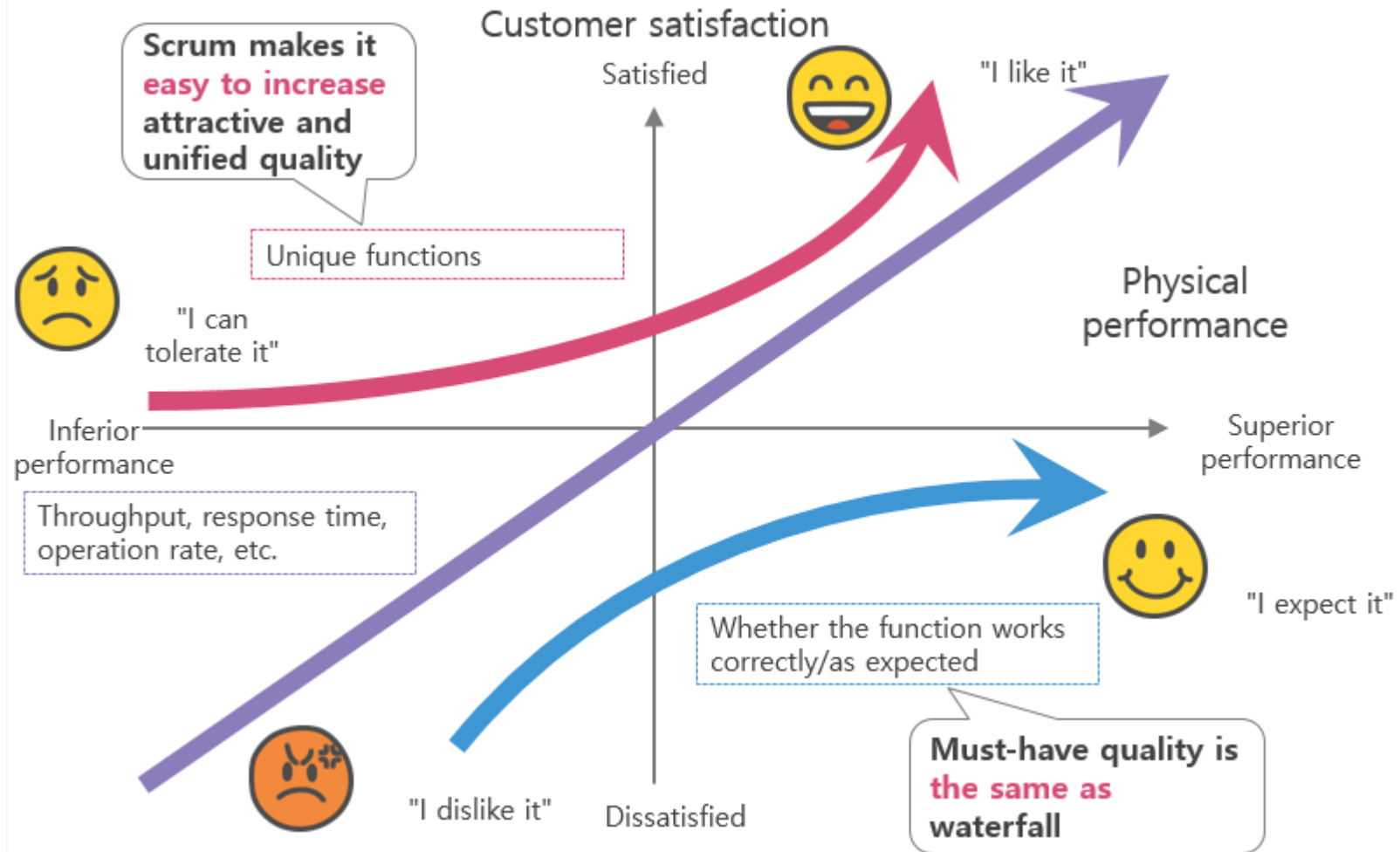
Appendix

Reasons for adopting agile

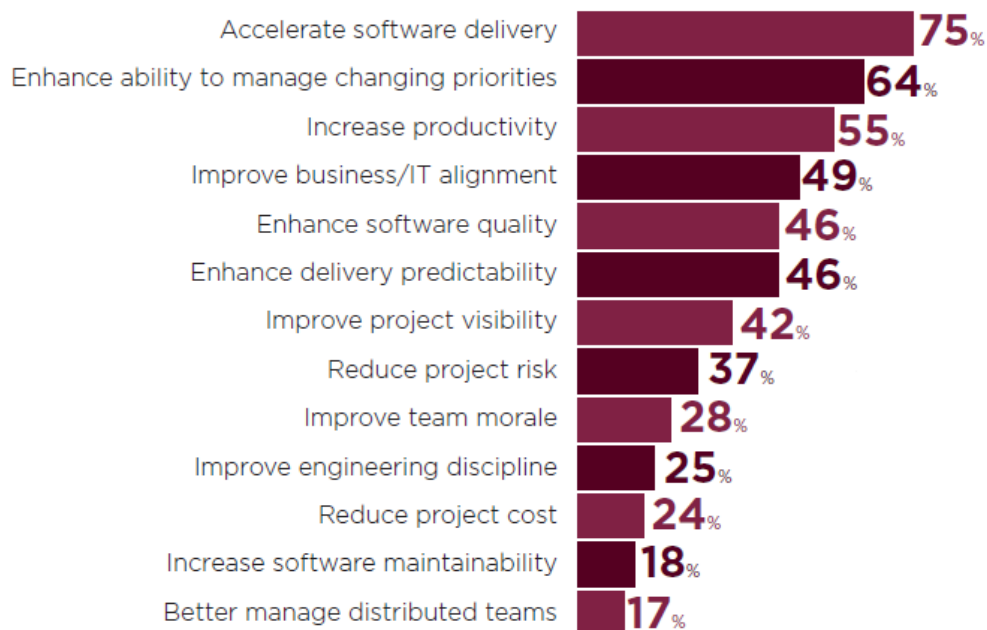## Reasons for Adopting Agile

The reasons stated for adopting agile follow a similar ranking as in the previous year though we did see the biggest change in responses in accelerate software delivery (75% compared to 69% last year), enhancing delivery predictability (46% compared to 30% last year), improving IT/Business alignment (49% compared to 42% last year), and reducing project cost (24% compared to 18% last year).

To focus more on accelerating software delivery.

Reason	Percent
Accelerate software delivery	75%
Enhance ability to manage changing priorities	64%
Increase productivity	55%
Improve business/IT alignment	49%
Enhance software quality	46%
Enhance delivery predictability	46%
Improve project visibility	42%
Reduce project risk	37%
Improve team morale	28%
Improve engineering discipline	25%
Reduce project cost	24%
Increase software maintainability	18%
Better manage distributed teams	17%

*Respondents were able to make multiple selections.

VersionOne 12th Annual State of Agile Report
http://stateofagile.versionone.com/

Benefits of adopting agile

Benefits of Adopting Agile

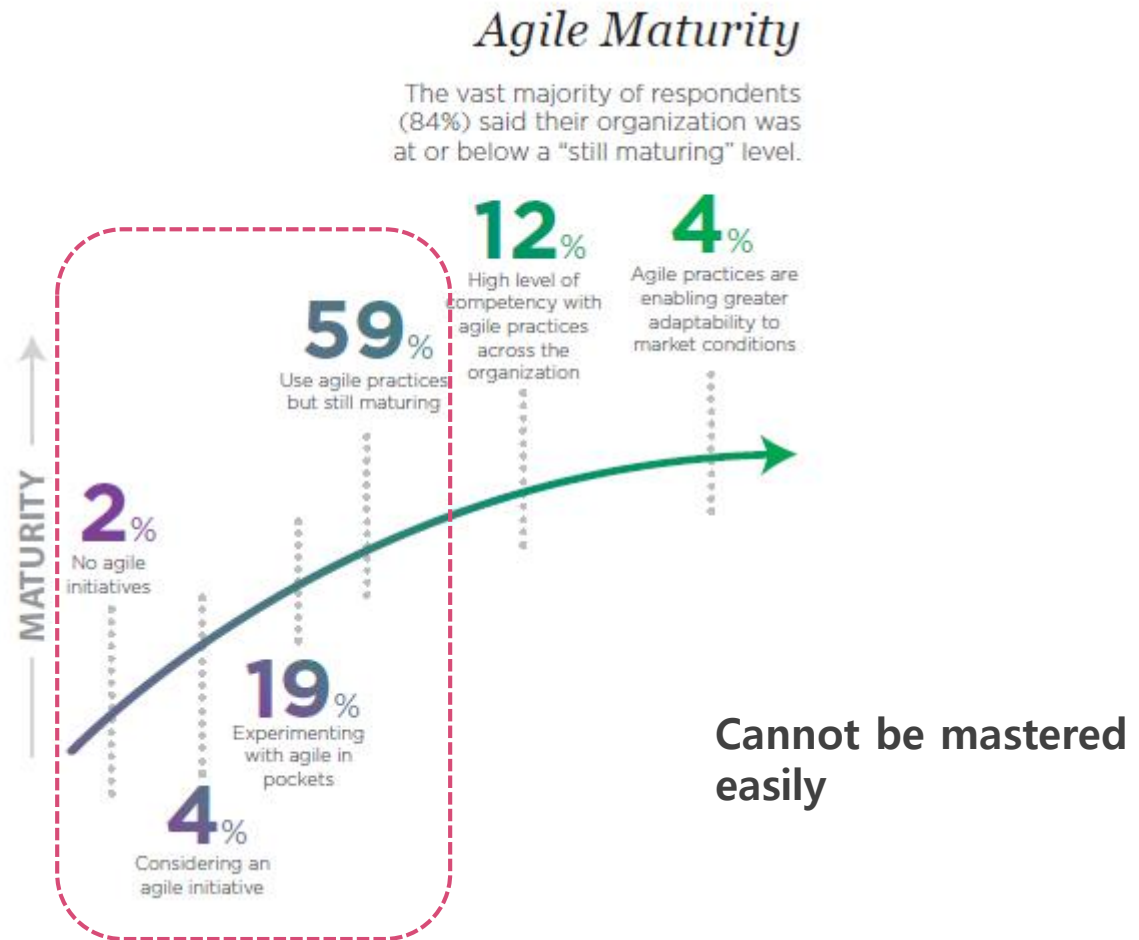By implementing agile, respondents cited seeing improvements in the following areas:

Area	Percentage
Ability to manage changing priorities	71%
Project visibility	66%
Business/IT alignment	65%
Delivery speed/time to market	62%
Increased team productivity	61%
Team morale	61%
Project predictability	49%
Software quality	47%
Project risk reduction	47%
Engineering discipline	43%
Managing distributed teams	40%
Software maintainability	33%
Project cost reduction	22%

*Respondents were able to make multiple selections.

An unexpectedly small number of respondents cited project cost reduction.

VersionOne 12th Annual State of Agile Report
http://stateofagile.versionone.com/

75

Agile maturity

Agile Maturity

The vast majority of respondents (84%) said their organization was at or below a "still maturing" level.

12% High level of competency with agile practices across the organization

4% Agile practices are enabling greater adaptability to market conditions

59% Use agile practices but still maturing

2% No agile initiatives

19% Experimenting with agile in pockets

4% Considering an agile initiative

MATURITY

Cannot be mastered easily

# Maturity model of configuration and release management

Practice	Build management and continuous integration	Environment and deployment	Release management and compliance	Testing	Data management	Configuration management
Level 3 – Optimization: The focus is on improvement of processes	There is a venue for regular team discussion, which covers points such as integration issues, automation-based solutions, prompt feedback and better visualization.	All environments are well managed. Provisioning is fully automated. Virtualization is used appropriately.	The system operation team and delivery team work together to manage risks and reduce cycle times.	Changes to the full-scale environment are rarely canceled. Any issues are immediately detected and fixed.	Feedback about points such as the database performance and the deployment process itself is received on each release.	Change management policies are constantly verified to confirm that efficient teamwork and prompt deployment are taking place. Checks are also carried out to confirm the auditability of the change management process.
Level 2 – Quantitative management: Processes are measurable and controlled	Build metrics are collected and visualized, and work is conducted accordingly. Builds are not left broken.	Procedures for integrated deployment management and cancellation of releases and re-releases are tested.	The environment is actively managed by monitoring the health of the application. Cycle times are monitored.	Quality metrics and their trends are tracked. Non-functional requirements are defined and measured.	Database updates and rollbacks are tested for each deployment. Database performance is monitored and optimized.	Developers check in on the main line at least once per day. Branches are only used for release work.
Level 1 – Consistent: Automated processes are applied to the entire life cycle of the application	Automated build and test cycles are executed each time a change is committed. Dependency relationships are managed. Scripts and tools are reused.	Software deployment is fully automated and can be completed by simply clicking a button. The same procedure is deployed for all environments.	A process for managing and approving changes is defined and followed. Protocols are followed	Testing such as unit testing and acceptance testing is automated. Acceptance tests are written by the tester. Testing is incorporated into the development process.	Changes to the database are performed automatically as part of the deployment process.	Libraries and dependency relationships are managed. Usage policies for version management systems are defined in the change management process.
Level 0 – Repeatable: Processes are documented and some are automated	Standard builds and tests are automated. All builds can be recreated with an automated procedure using a source management system.	Deployment is automated in some environments. New environments can be created easily. All configuration management information is used externally for version management.	Releases are considered a nuisance, and are infrequent and unreliable. Traceability of release requirements is also limited.	Automated tests are written as part of story development.	An automated script is used to make changes to the database, and the versions of both the script and the application are managed.	Everything that is needed for software creation (source code and settings files, build and deployment script, data migration, etc.) is managed using a version management system.
Level -1 – Frequent regression errors: Processes are not repeatable and management is inadequate. These issues are being addressed.	Software build procedures are manual. Deliverables and build results are not managed.	Software deployment procedures are manual. Binary is environment-dependent. Environment distribution is manual.	Releases are infrequent and unreliable.	Testing is done manually after development.	Data migration is done manually and versions are not managed.	A version management system is not used, or check-ins are rarely done.

Quick but rough vs. polished but slow
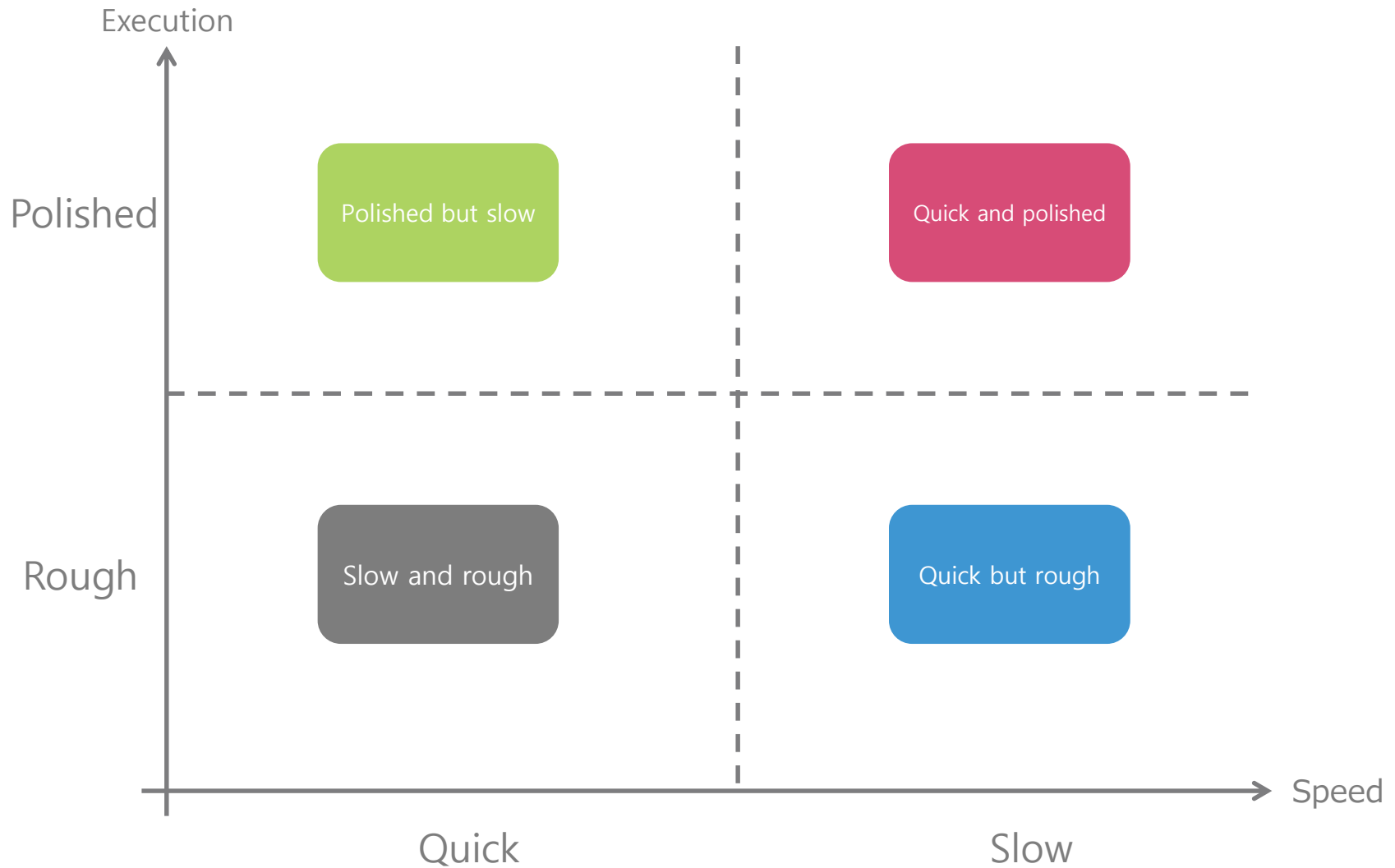
Quick but rough vs. polished but slow
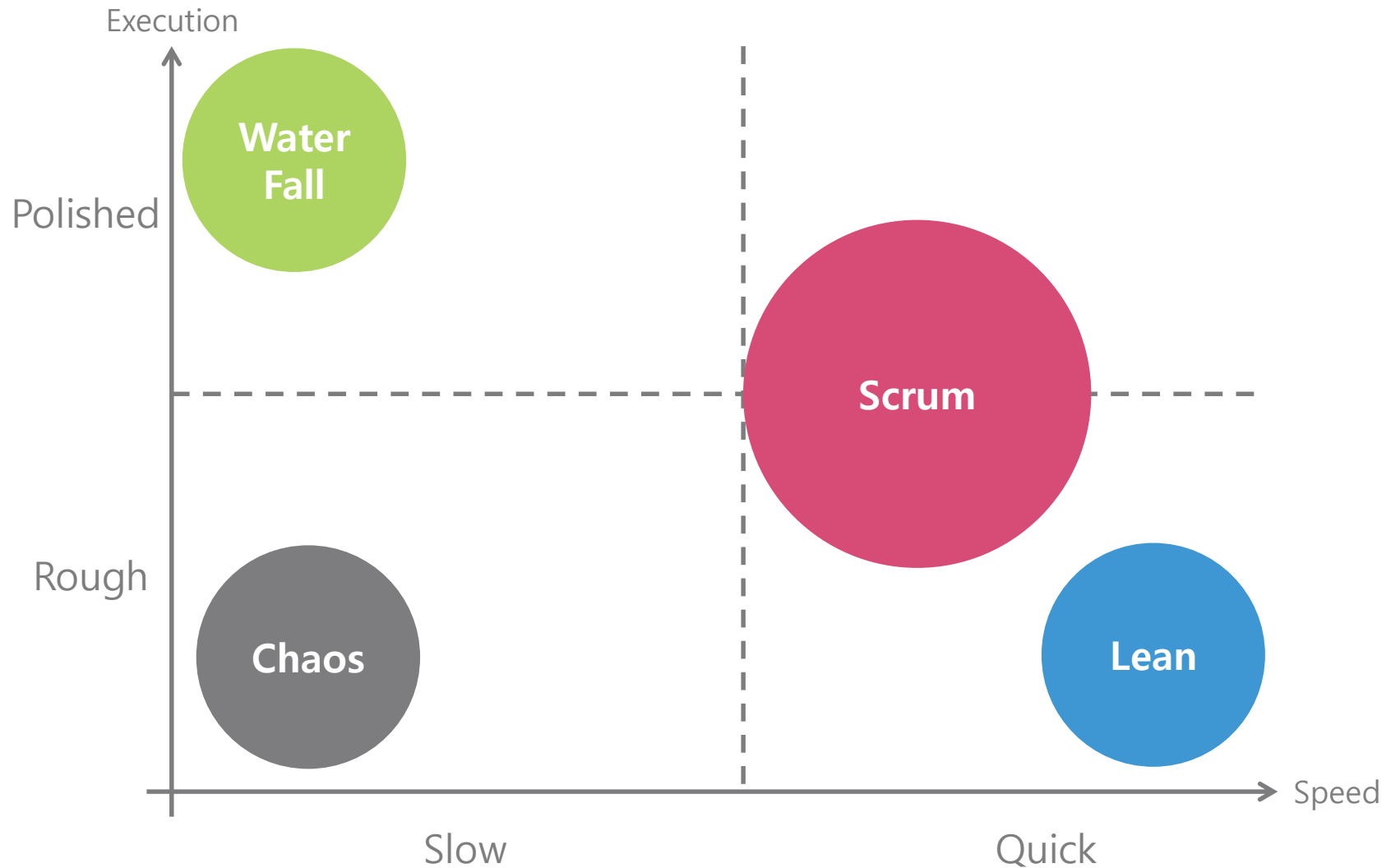
Scaling agile

Top 5 Tips for Success with Scaling Agile

Respondent indicated the most valuable in helping them scale agile practices were:

1 INTERNAL AGILE COACHES

2 CONSISTENT PRACTICES AND PROCESSES ACROSS TEAMS

3 IMPLEMENTATION OF A COMMON TOOL ACROSS TEAMS

4 EXTERNAL AGILE CONSULTANTS OR TRAINERS

5 EXECUTIVE SPONSORSHIP

*Respondents were able to make multiple selections.

VersionOne 12th Annual State of Agile Report
http://stateofagile.versionone.com/