Microsoft

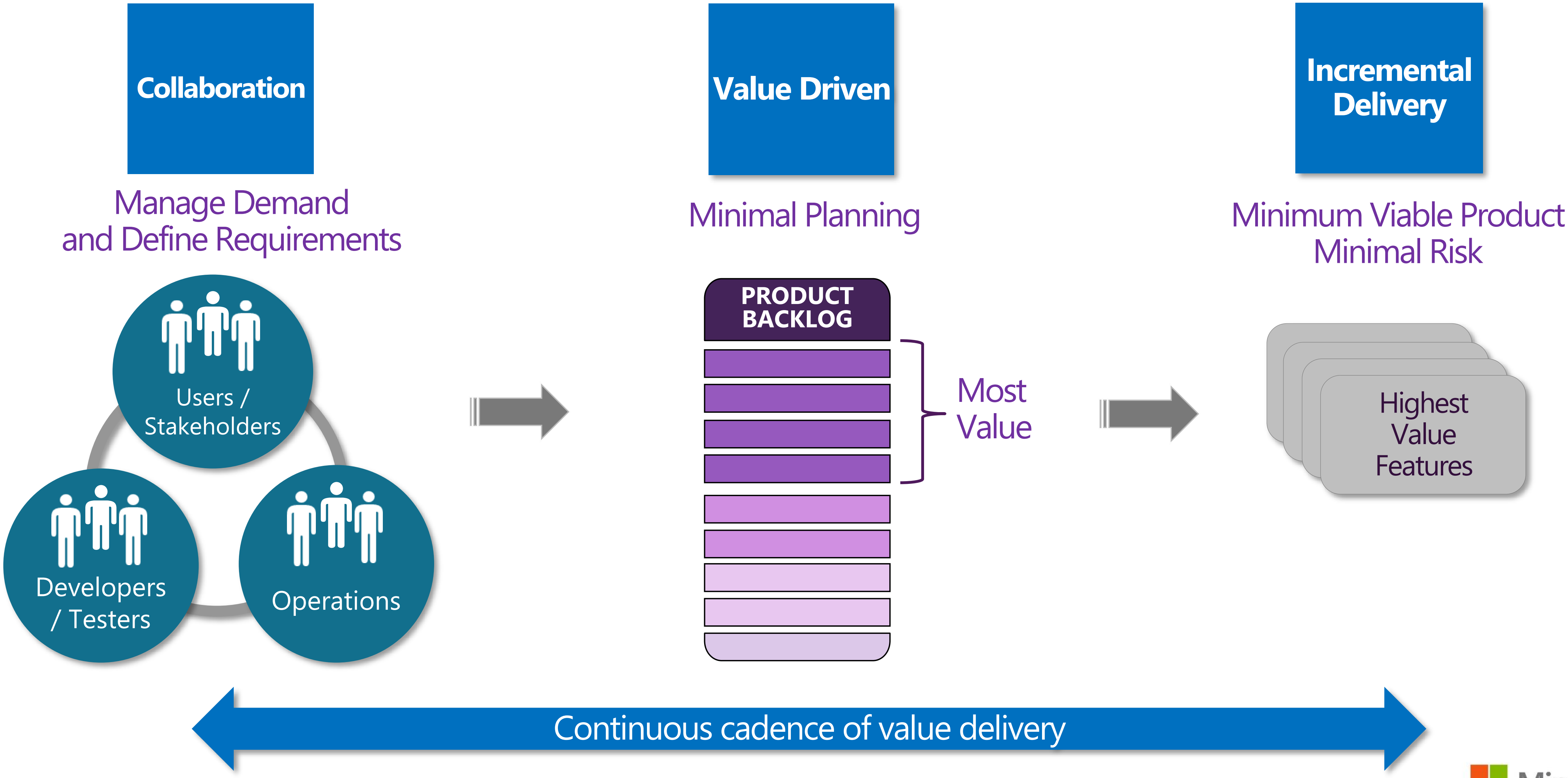
Agile 101

Crystal Tenn
Crystal.Tenn@microsoft.com

What is Agile?

A set of Values and Principles.

Planning for continuous delivery of value



Collaboration

Manage Demand
and Define Requirements

Users /
Stakeholders

Developers
/ Testers

Operations

Value Driven

Minimal Planning

PRODUCT BACKLOG

Most
Value

Incremental Delivery

Minimum Viable Product
Minimal Risk

Highest
Value
Features

Continuous cadence of value delivery

Agile Manifesto

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.

Agile Principle #1

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

Agile Principle #2

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

Microsoft

Agile Principle #3

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

Agile Principle #4

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

Agile Principle #5

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

Agile Principle #6

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

Agile Principle #7

Working software is the primary measure of progress.

Microsoft

Agile Principle #8

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

Agile Principle #9

Continuous attention to technical excellence and good design enhances agility.

Agile Principle #10

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

Agile Principle #11

The best architectures, requirements, and designs emerge from self-organizing teams.

Agile Principle #12

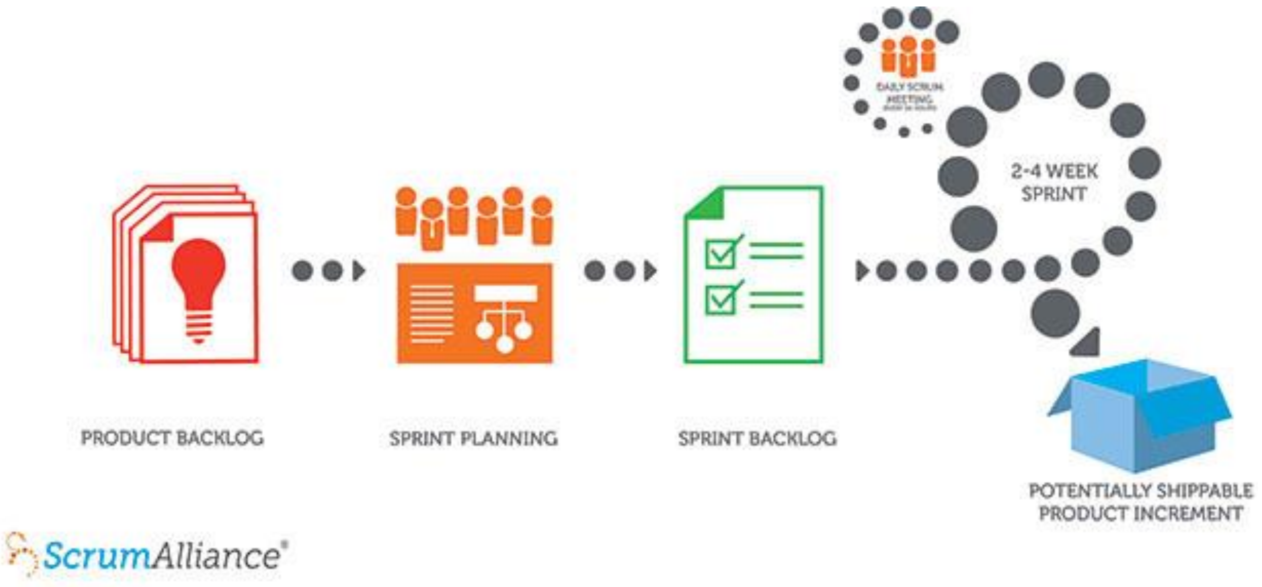
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile Methodologies: Scrum and Kanban

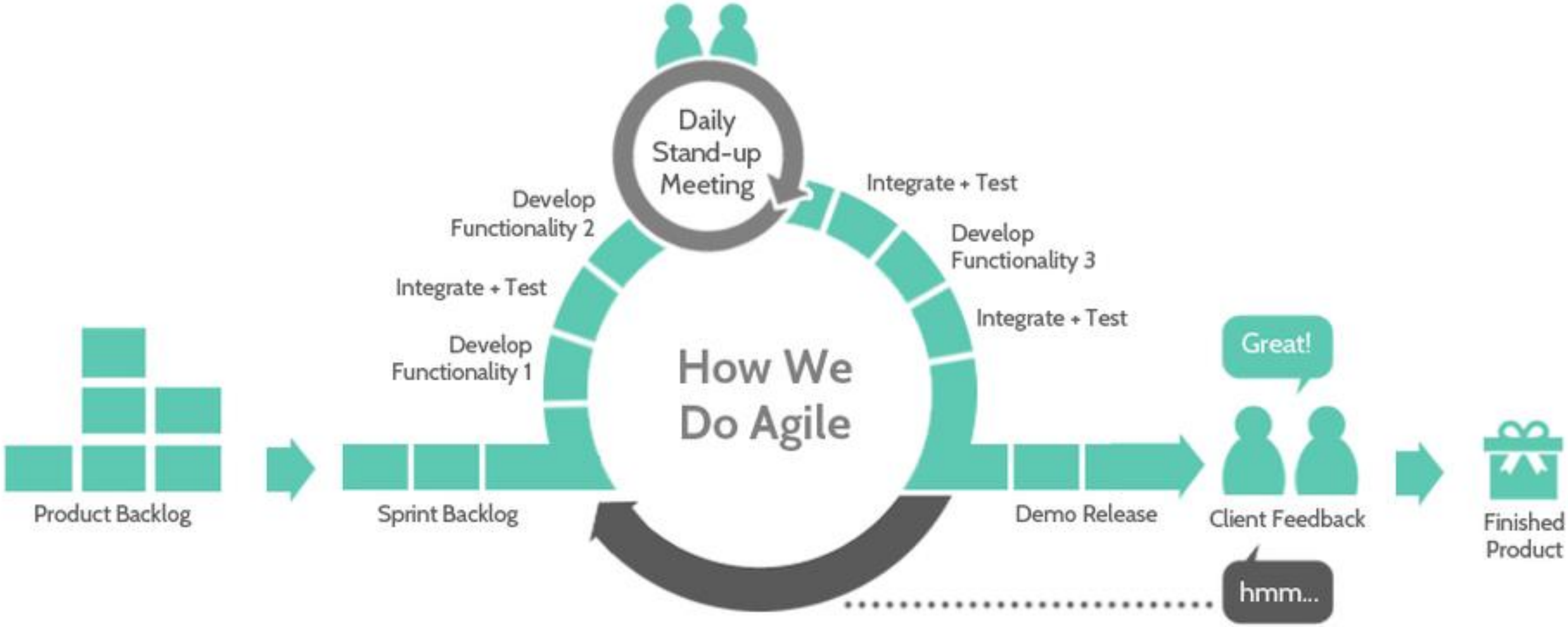
What are Scrum and Kanban?

Scrum is an Agile framework for completing complex projects.

Scrum in 30 seconds

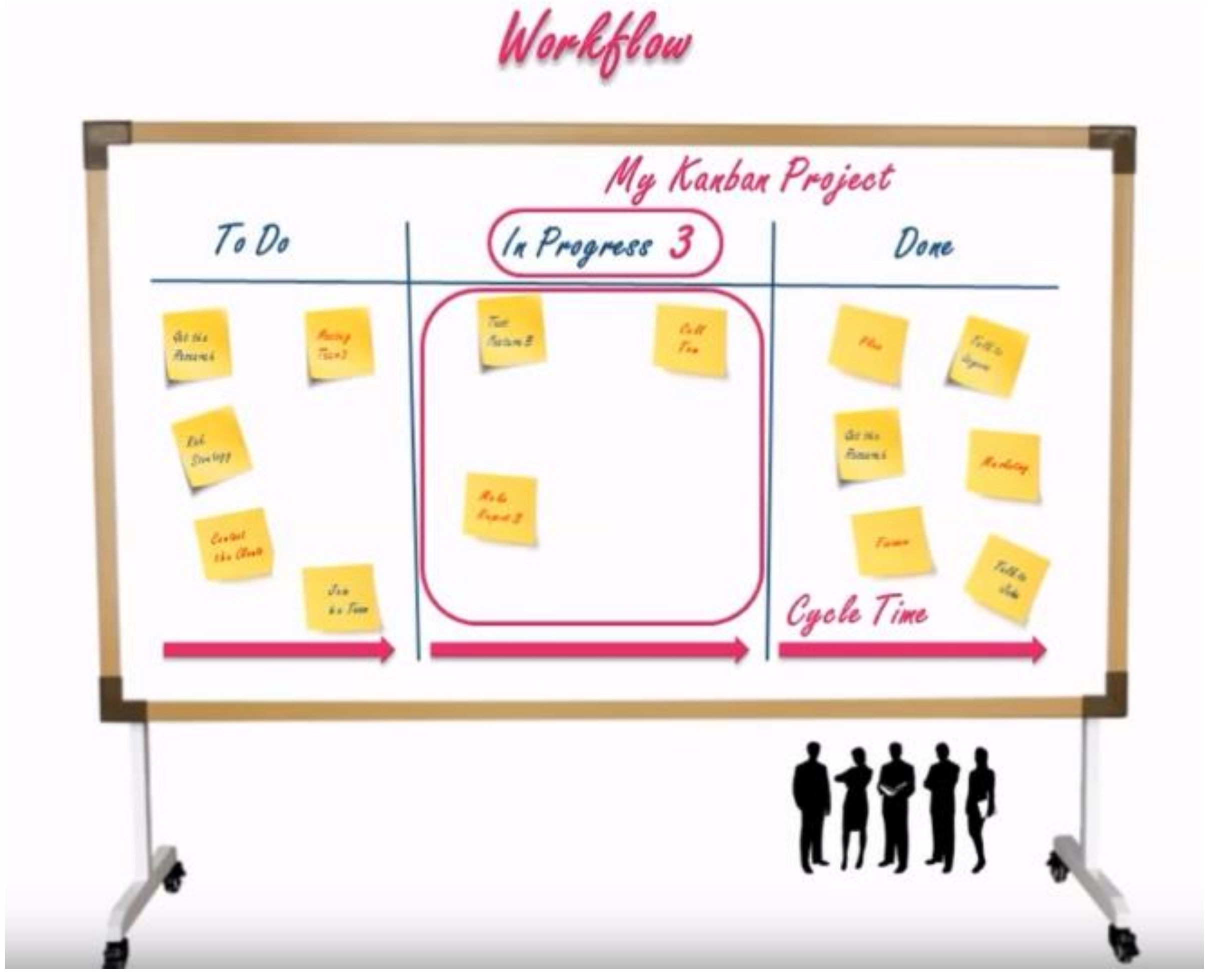


- A product owner creates a prioritized list called a product backlog.
- During sprint planning, the team pulls a small chunk from the top of the sprint backlog, and decides how to implement those pieces.
- The team has a sprint (usually 2-4 weeks) — to complete its work, but meets daily to assess progress (daily Scrum standup).
- The ScrumMaster keeps the team focused.
- At the end of the sprint, the work should be potentially shippable: ready to hand to a customer or demo.
- The sprint ends with a sprint review and retrospective.


Daily
Stand-up
Meeting
Develop
Functionality 2
Integrate + Test
Develop
Functionality 1
Integrate + Test
Develop
Functionality 3
Integrate + Test
How We
Do Agile
Great!
Product Backlog
Sprint Backlog
Demo Release
Client Feedback
Finished
Product
hmm...

Kanban is also an agile framework that requires real-time communication of capacity and full transparency of work.

Kanban in 30 seconds



- A project has a board that lives as long as the project.
- As new work is needed, it is added to the **to do** side and color prioritized.
- Limited work should be in progress at one time.
- Once work is completed it is moved to the done side.
- Work is completed based on priority and as team members get capacity.

Let’s compare Scrum and Kanban…

Differences between Scrum and Kanban: Scheduling

- Scrum processes place heavy emphasis on completing a shippable product within each 2-4 week sprint. Anything outside of the sprint must be pushed to a future sprint.
- On a Kanban team, there are no required time boxes or iterations. While the Kanban method is iterative in nature, the continual improvement is expected to occur as work is continually completed.

Differences between Scrum and Kanban: Team Members

- On scrum teams, 3 roles must exist: the Product Owner, Scrum Master, and Team Members. A team must have all the resources necessary to complete the entire sprint's work on their own.
- Under Kanban, no set roles are prescribed. Teams can share resources. Members can fulfill multiple roles.

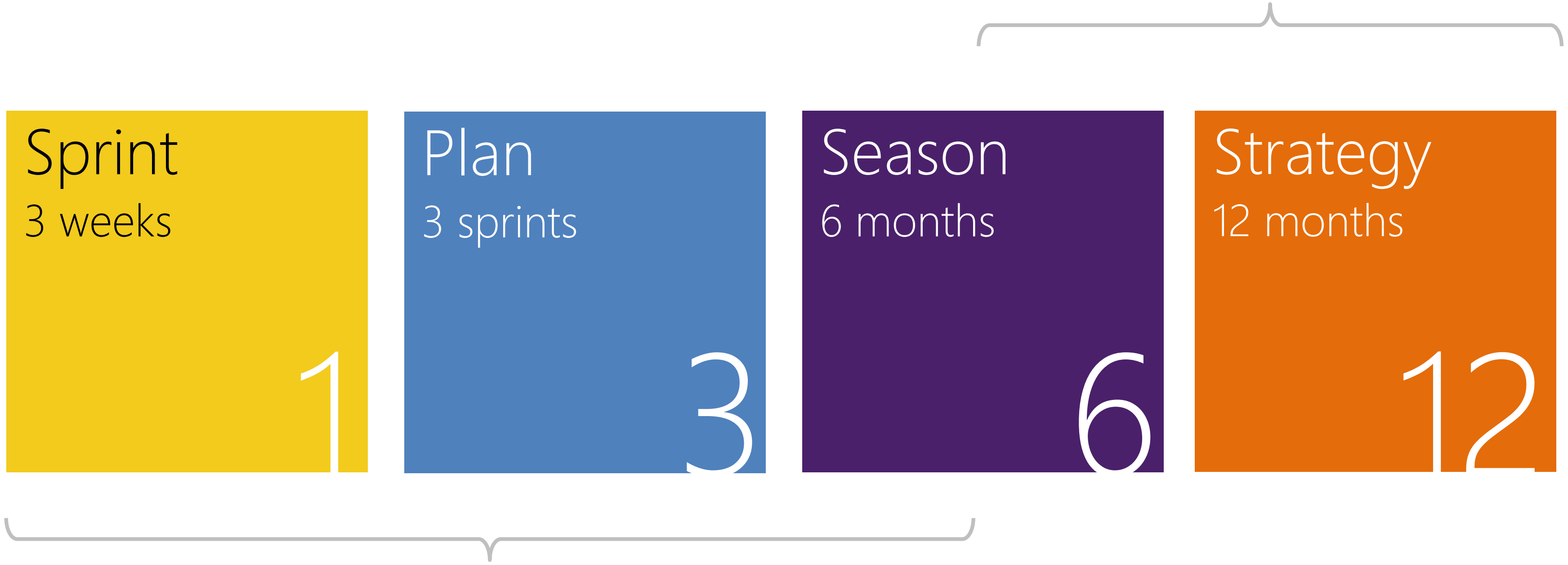
Differences between Scrum and Kanban: **Boards**

- In Scrum, columns are labeled to reflect periods in the work flow. All the stories added to the board at the beginning of each sprint should be found in the final column at the end of that sprint or the sprint was unsuccessful. Board is reset for each new sprint.
- On a Kanban board, the columns show work flow states and a maximum value for the In Progress state. No reason to reset the Kanban board as work progresses. It will continue to flow for as long as the project continues, with new stories being added as the need arises.

“Big” Enterprise Agile Alignment

Planning

Leadership is responsible for the big picture



Sprint
3 weeks
1

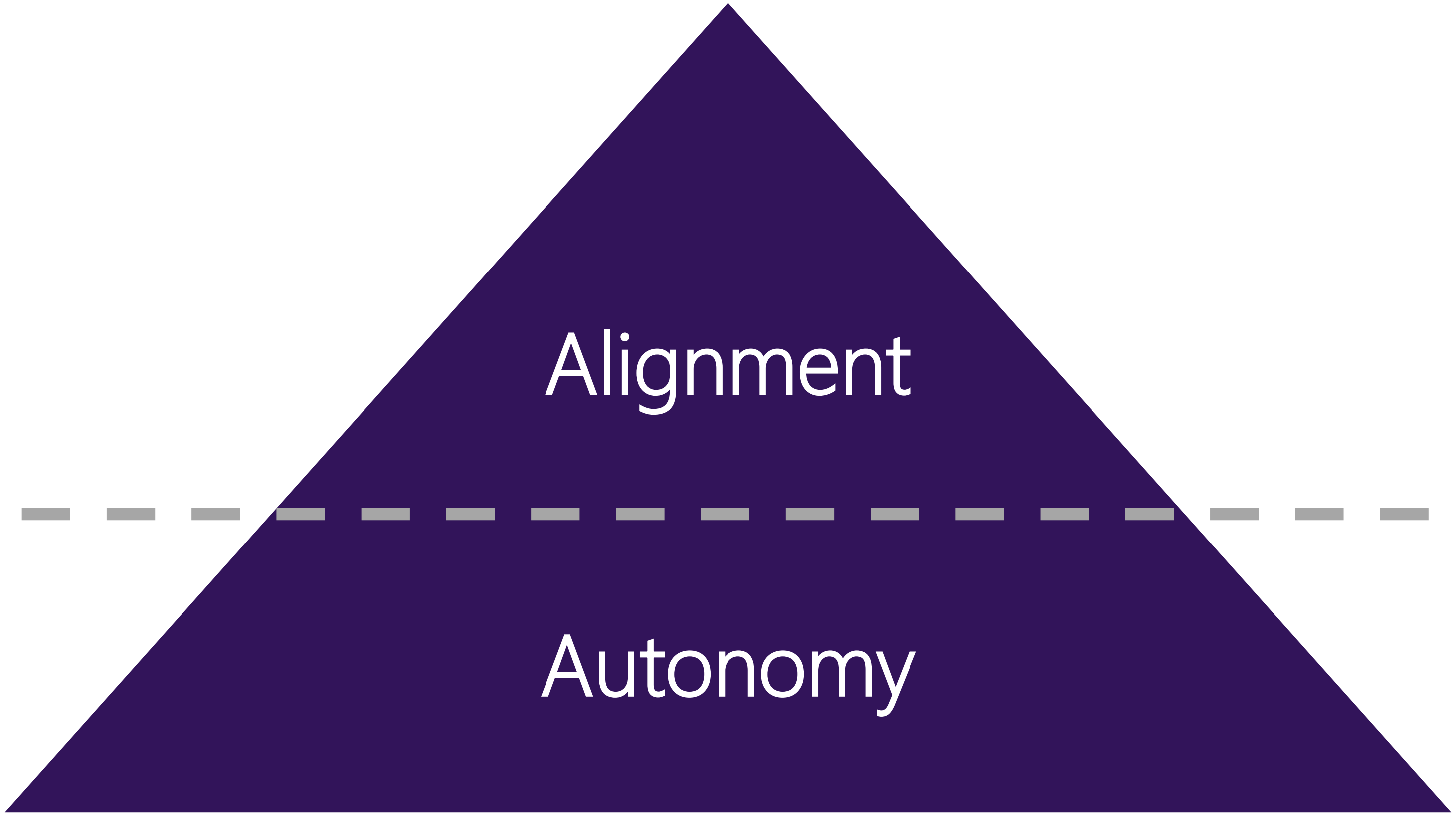
Plan
3 sprints
3

Season
6 months
6

Strategy
12 months
12

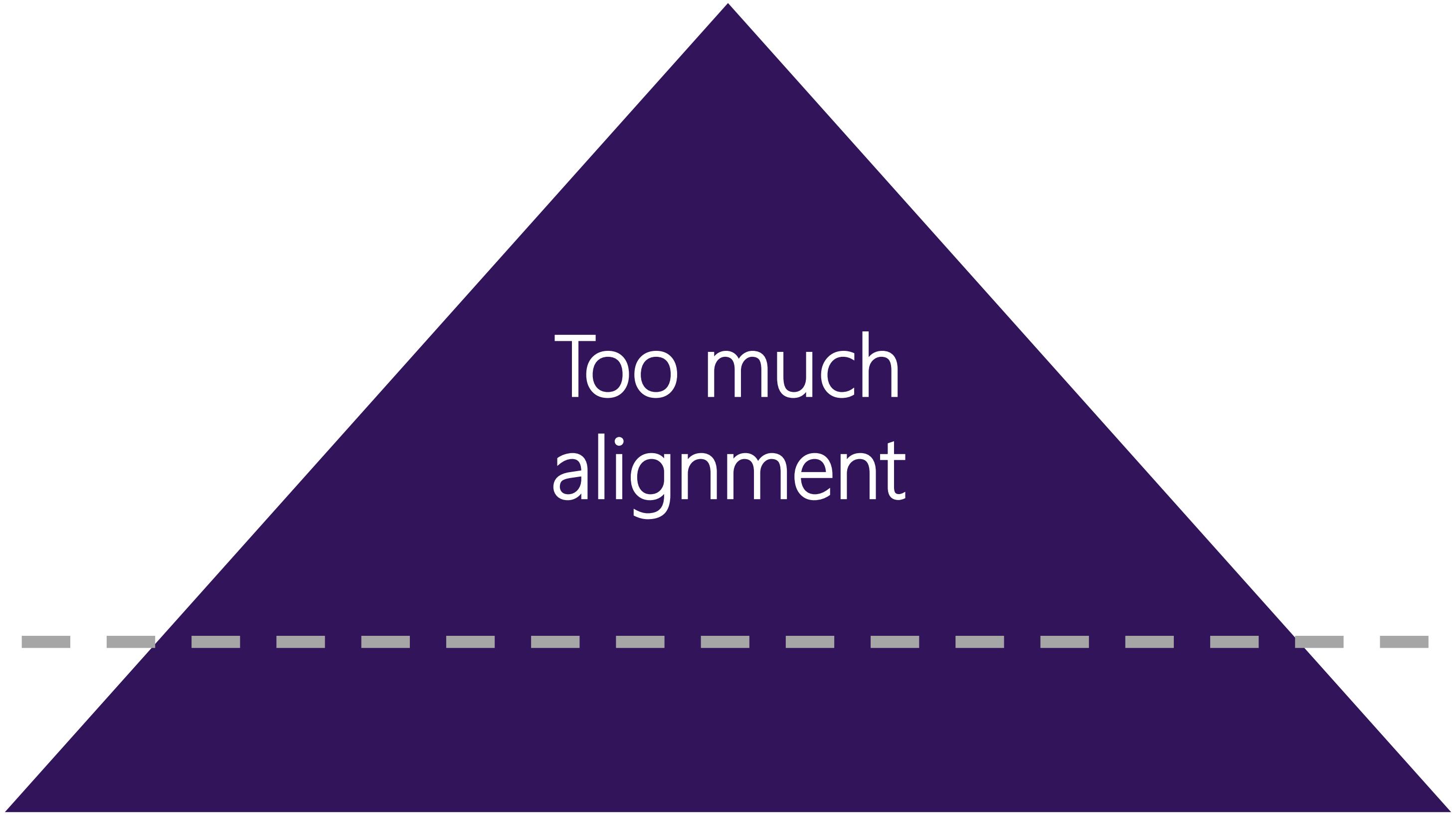
Teams are responsible for the detail

Aligned Autonomy


Alignment

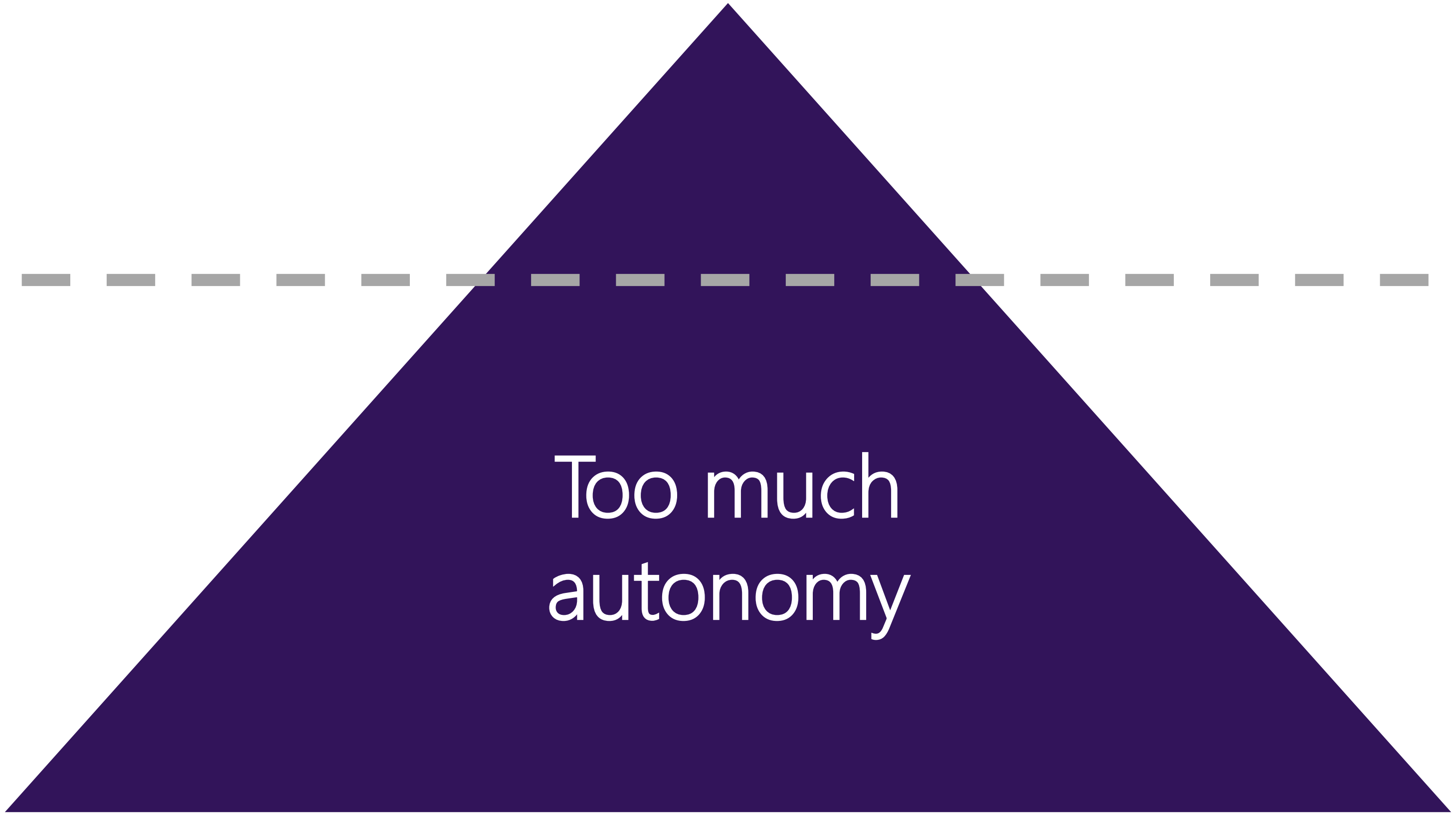
Autonomy

Aligned Autonomy



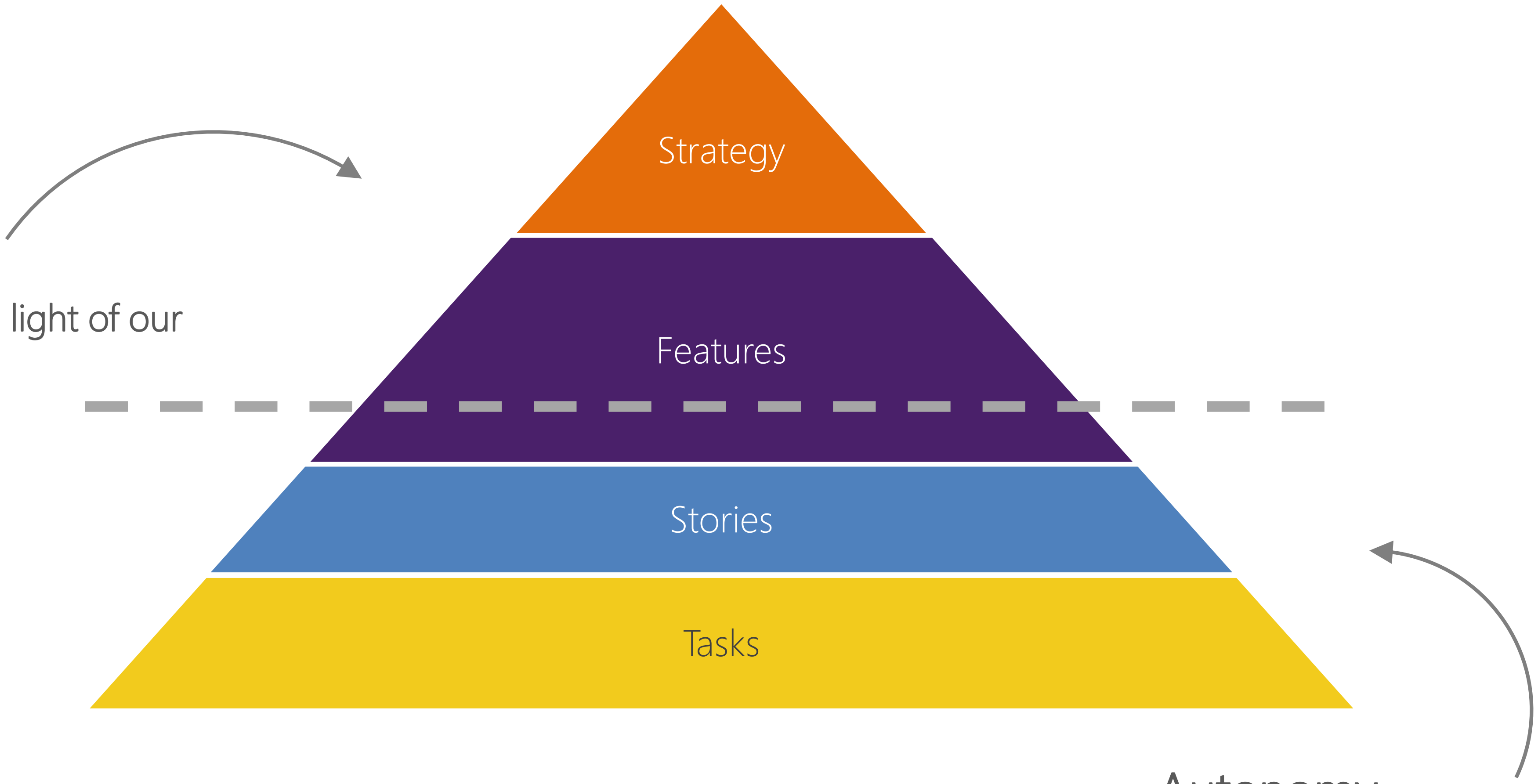
Microsoft

Aligned Autonomy



Too much autonomy

Aligned Autonomy



Strategy

Features

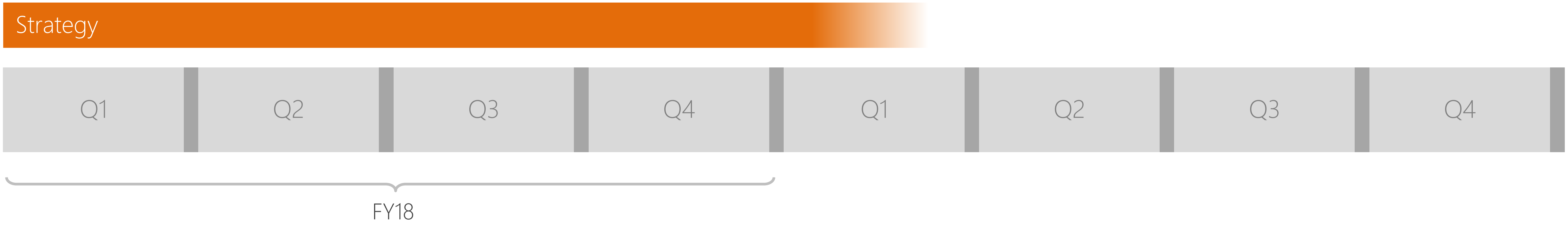
Stories

Tasks

Alignment
The big picture in light of our business goals

Autonomy
The detail about what we'll deliver to achieve our business goals

Planning

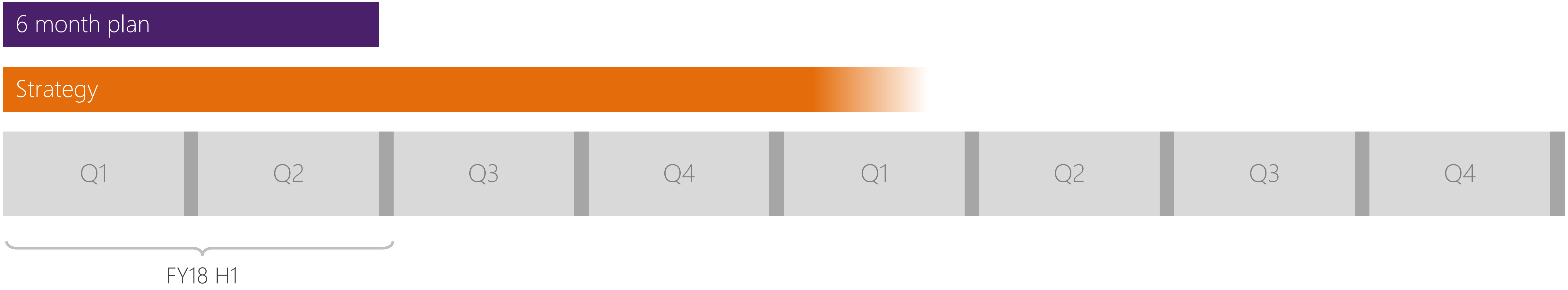


Strategy

Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4

FY18

Planning



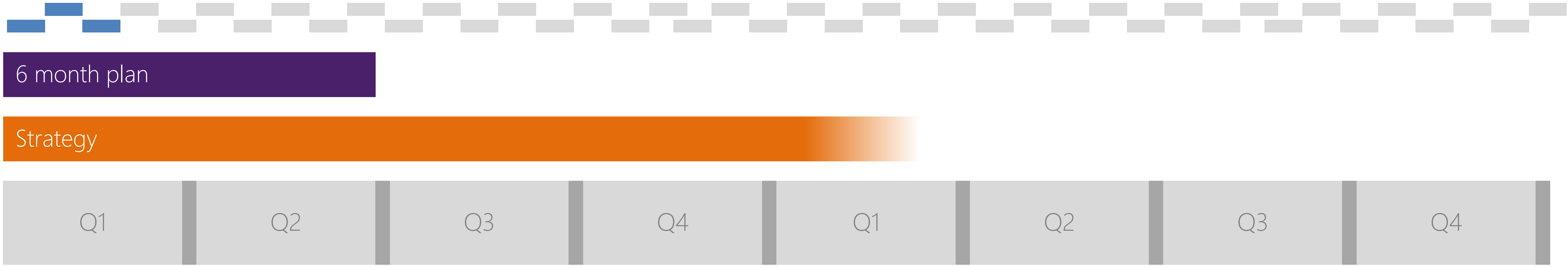
6 month plan

Strategy

Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4

FY18 H1

Planning

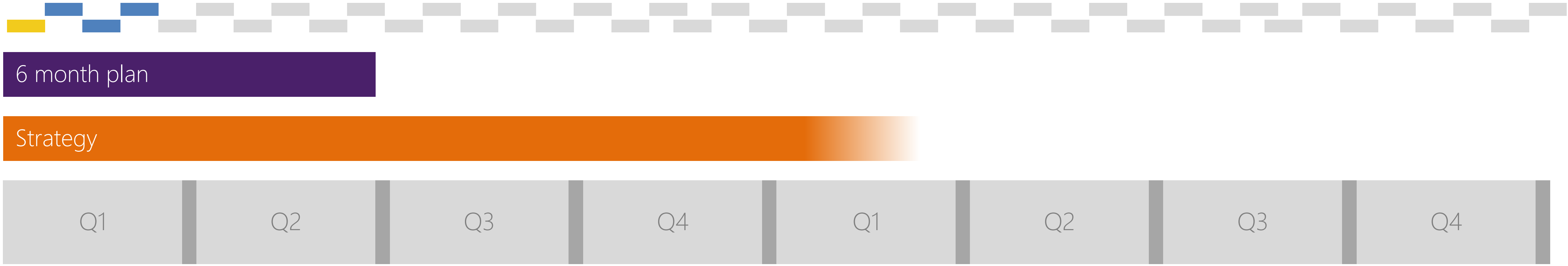


6 month plan

Strategy

Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4

Planning

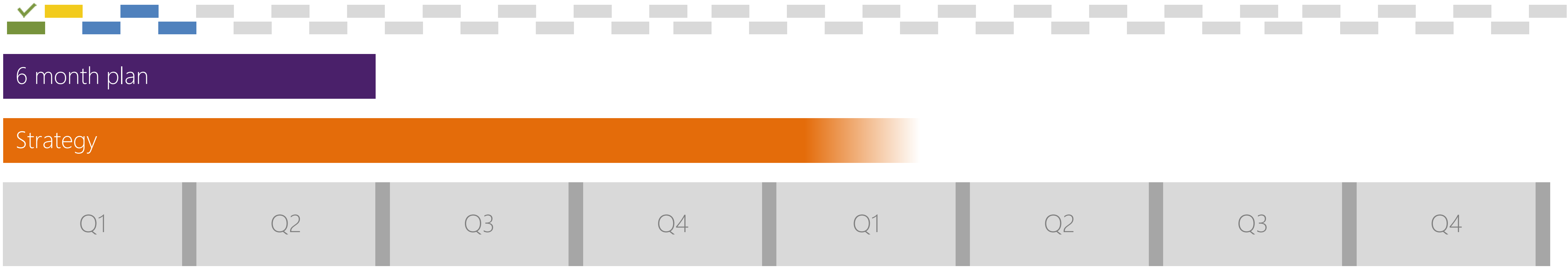


6 month plan

Strategy

Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4

Planning

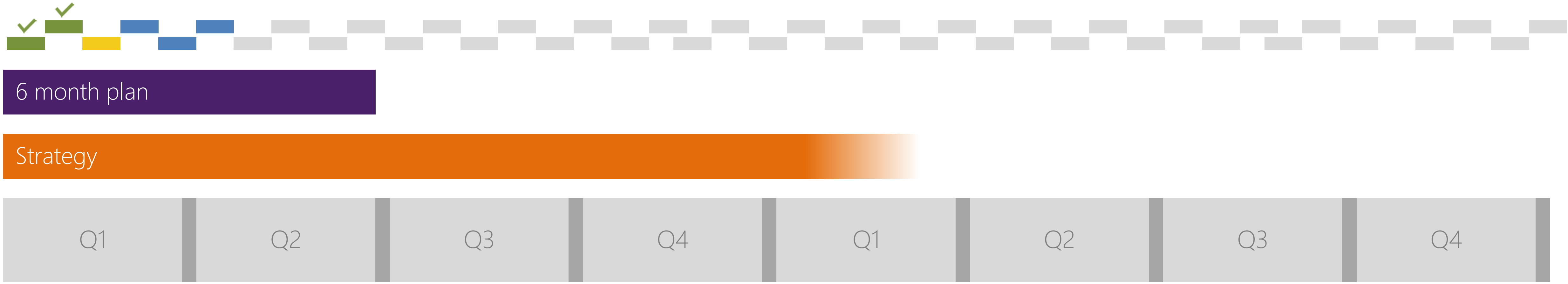


6 month plan

Strategy

Q1 Q2 Q3 Q4 Q1 Q2 Q3 Q4

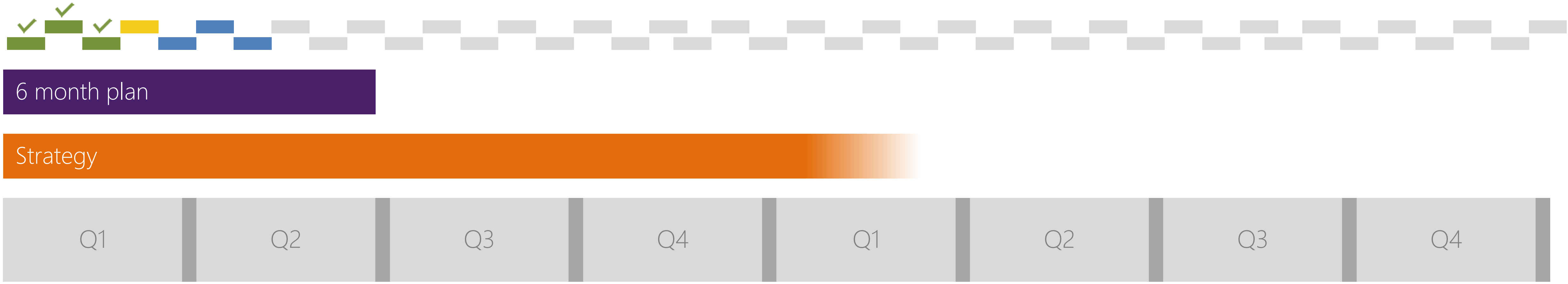
Planning



6 month plan

Strategy

Planning

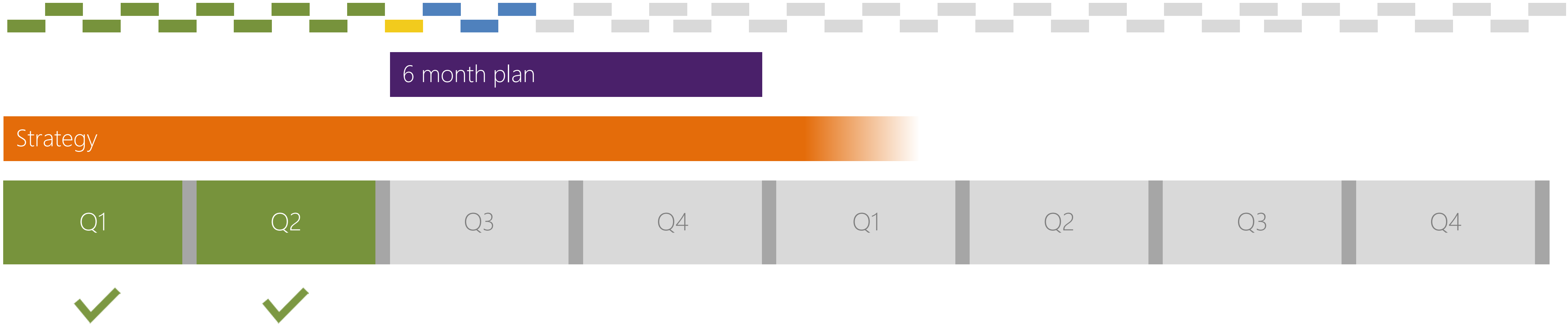


6 month plan

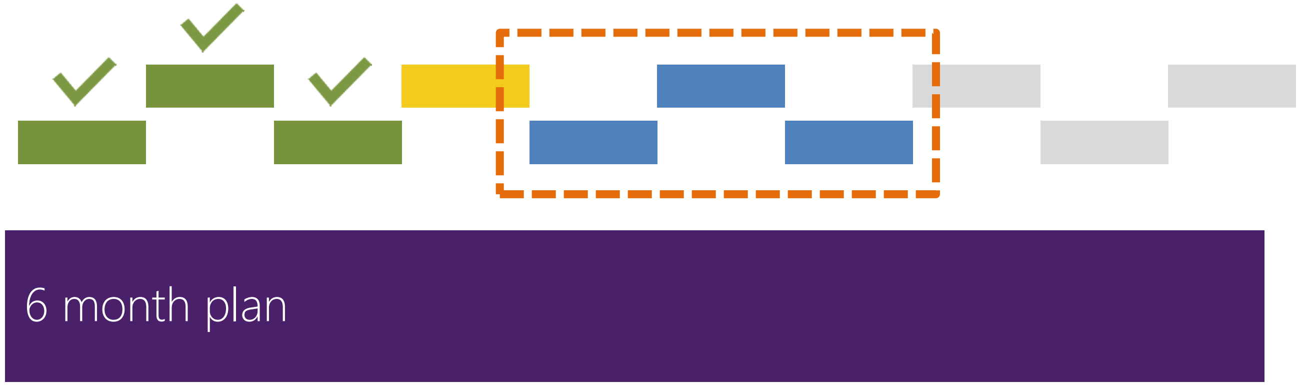
Strategy

Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4

Planning



Quarterly Feature Team Chats



Each team comes in and reviews with leadership three things:

1. What is the plan for the next 3-sprints?
2. Is the team healthy?
3. Any risks or issues to highlight?

Agile planning takeaways

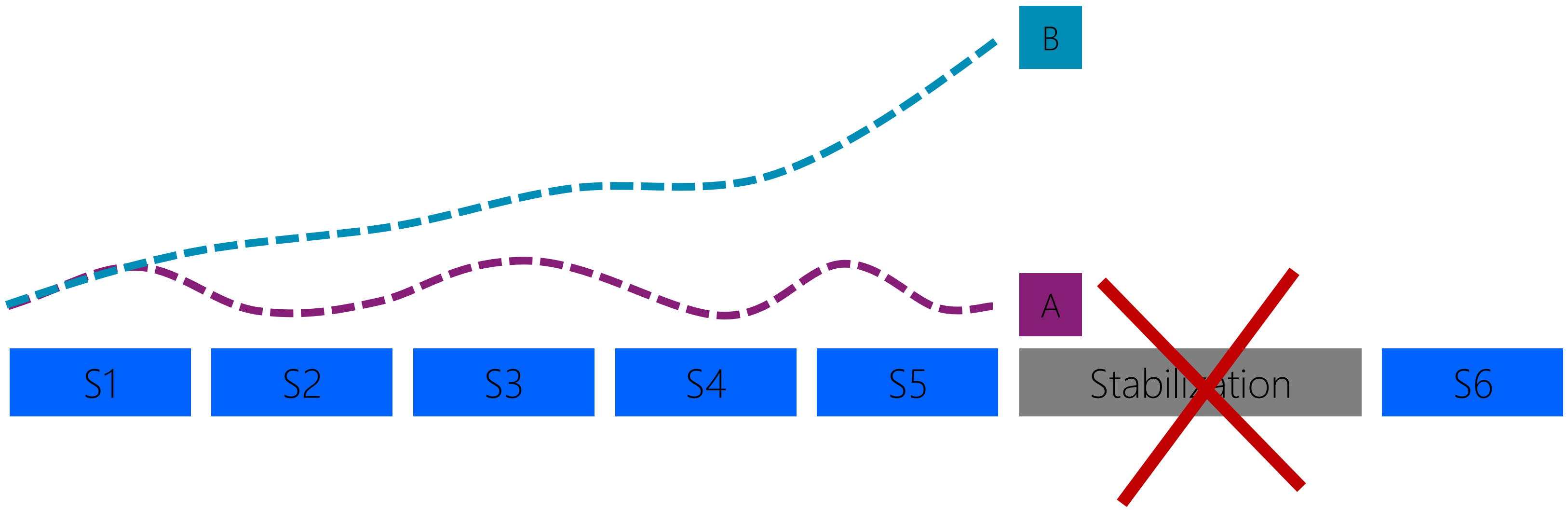
The schedule

- No stabilization

Alignment and Autonomy

- You need both

Continuous planning (and learning)

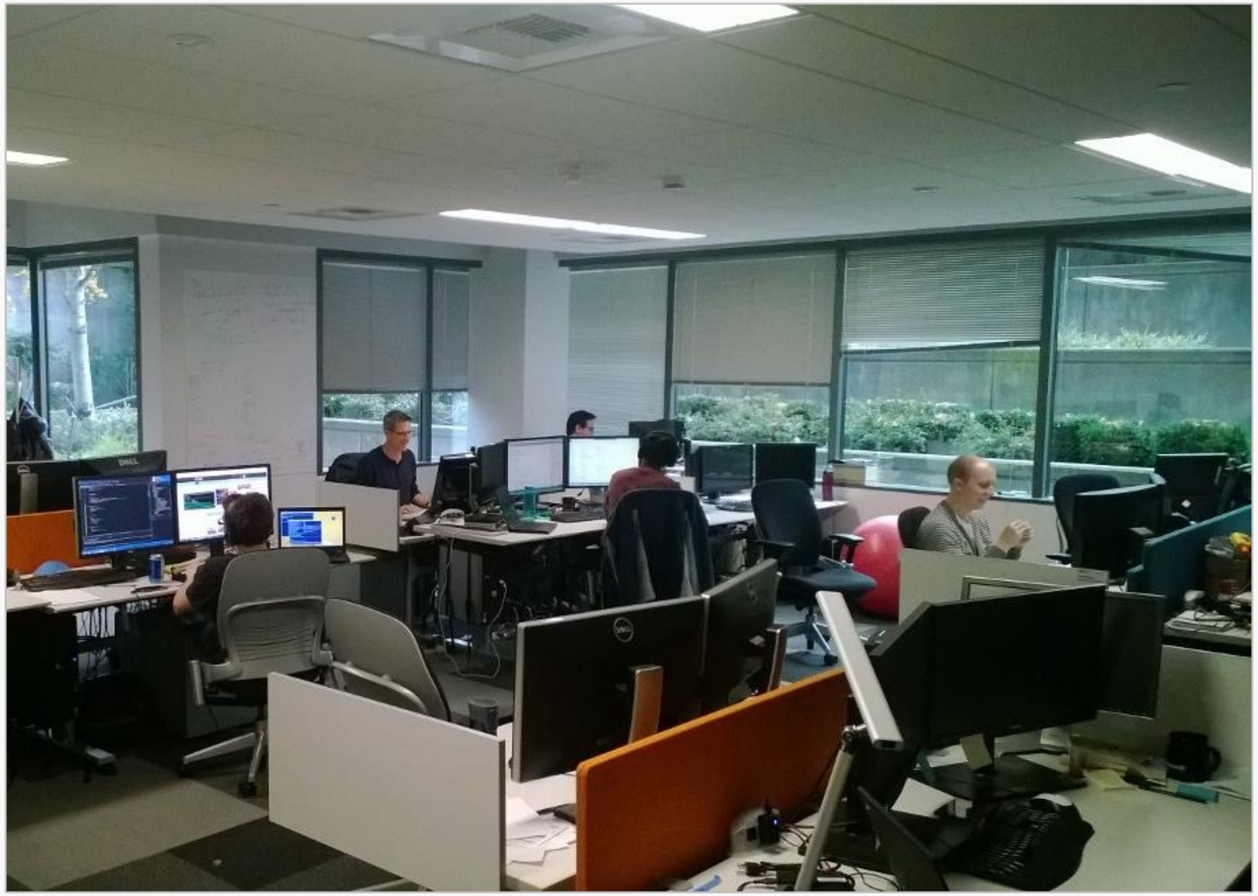


Microsoft

The Team and Roles

Teams

Cross discipline
10-12 people
Self managing
Clear charter and goals
Intact for 12-18 months
Physical team rooms
Own features in production
Own deployment of features



Microsoft

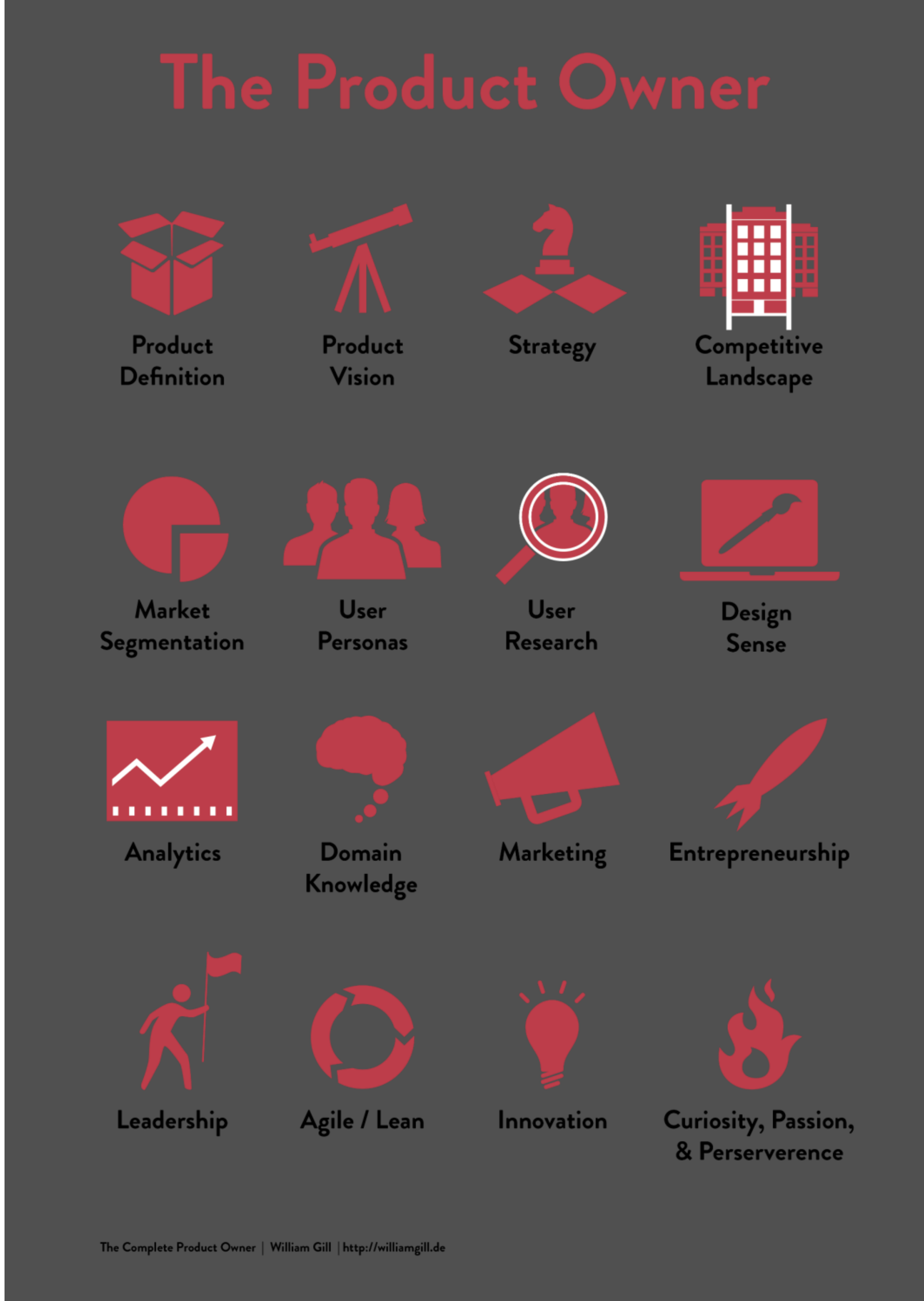
Scrum Master

- Helps the team to follow the Scrum **process**
- Assists the team in **removing blockers** and in giving them as much time possible to work
- Helps the Product Owner understand and create the product
- Runs Scrum meetings

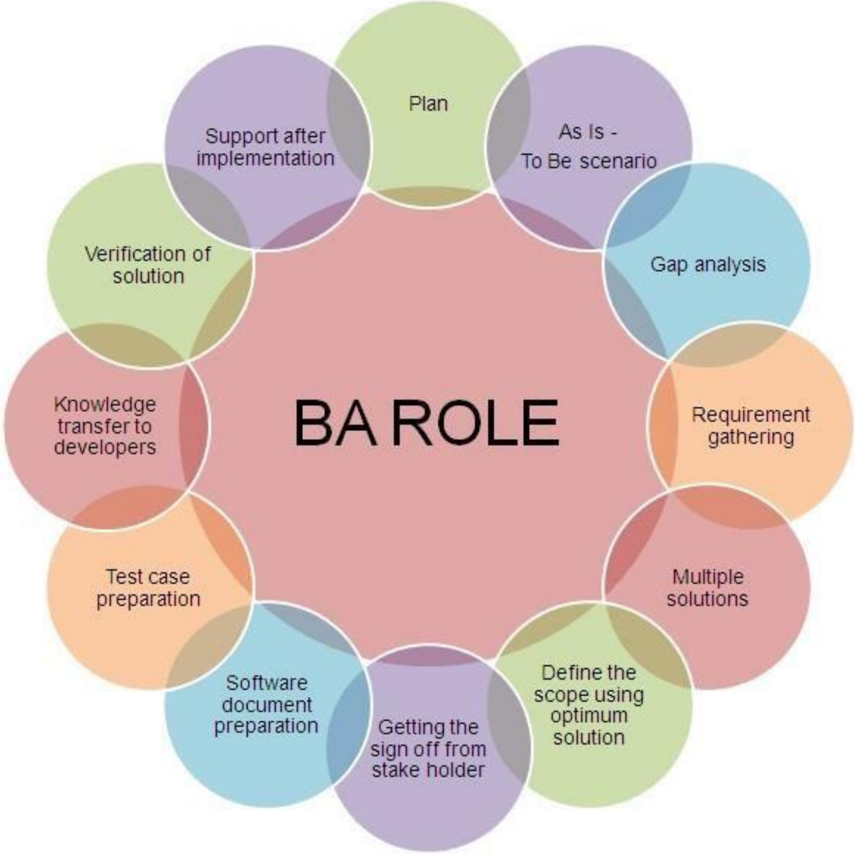
Microsoft

Product Owner

- **Product Vision** and final decisions on backlog item priority and questions
- Prioritize and manage backlog
- Really **own** the backlog and keep track of all changes to it and update the team



The Complete Product Owner | William Gill | http://williamgill.de


Plan
As Is -
To Be scenario
Gap analysis
Requirement gathering
Multiple solutions
Define the scope using optimum solution
Getting the sign off from stake holder
Software document preparation
Test case preparation
Knowledge transfer to developers
Verification of solution
Support after implementation
BA ROLE

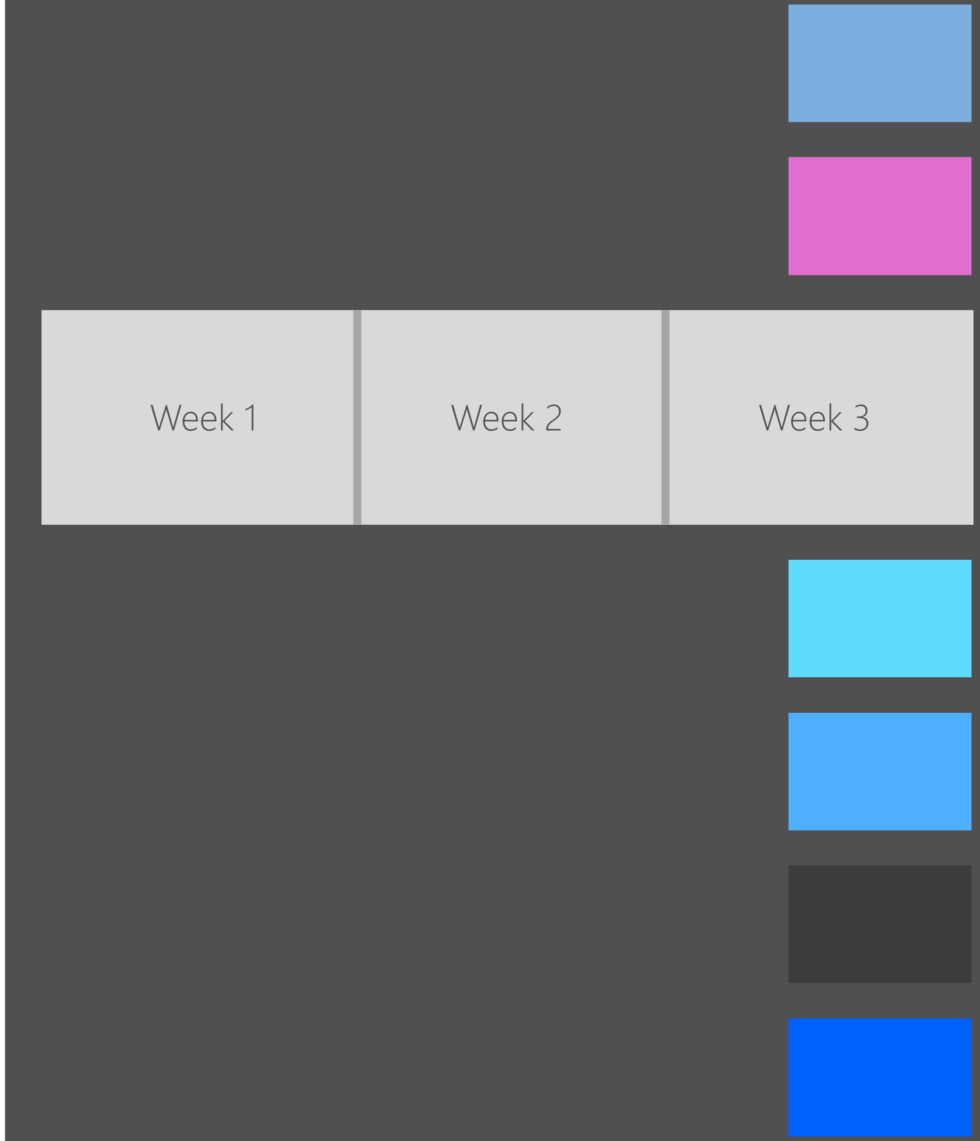
Business Analysts

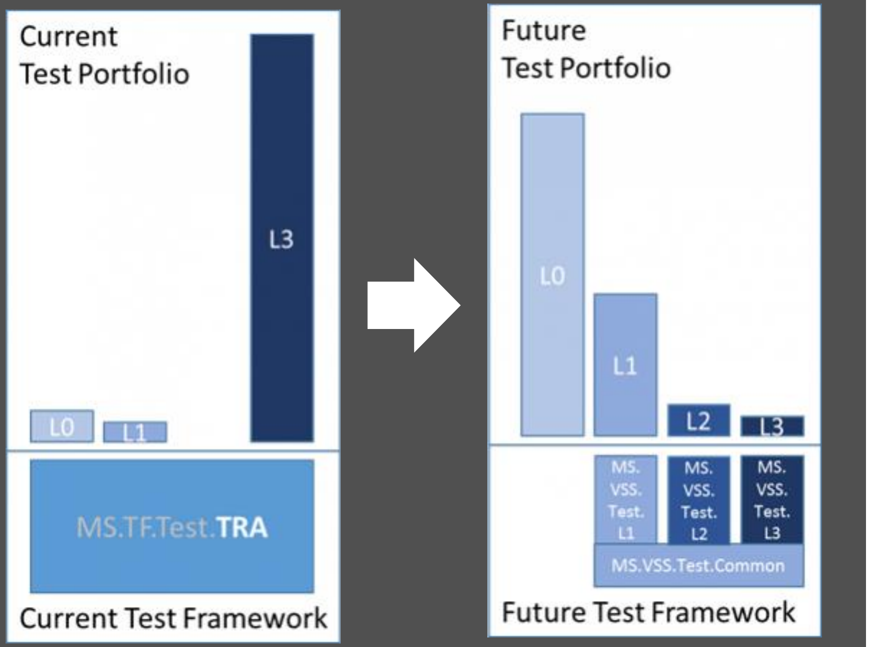
- Gather requirements and create user stories
- **Relate** user stories to one another so links are clear
- **Convey** user stories to the team to help them build

Microsoft

Day in the life of an Engineer

- **Work** on Code, build the product
- Process and tools should help you do more. **Communicate** with your team to ensure product is built right!
- Continuous deployment



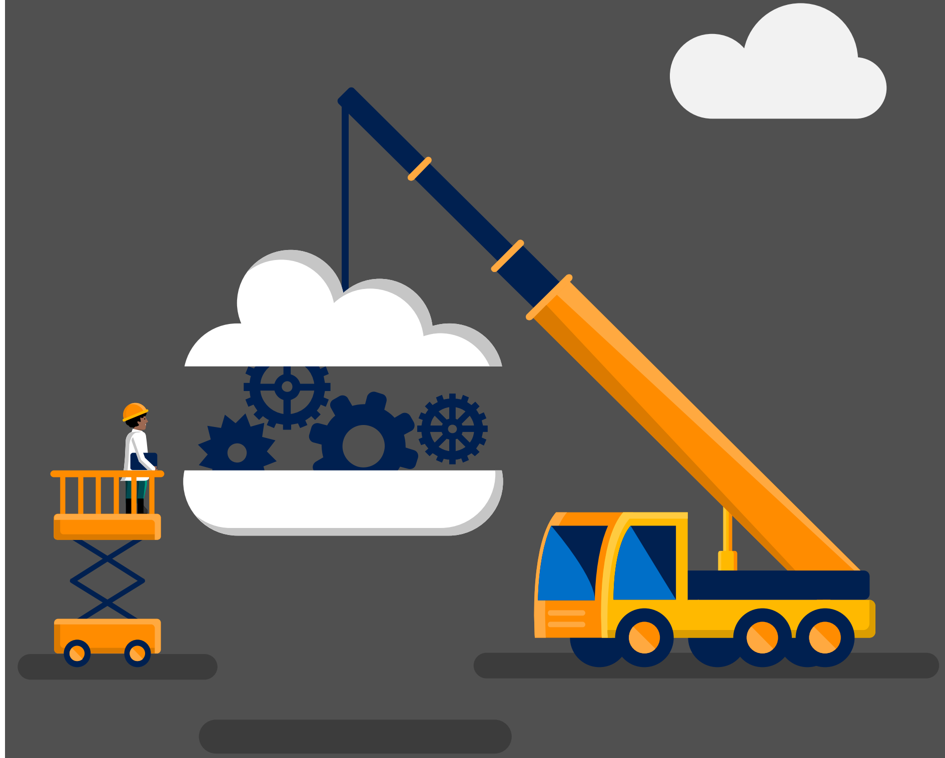


Quality and Testing

- Test accountability with dev
- Zero tolerance for Flakey tests or Flakey code
- Test closer to the code
- Test **early** and test often!
- Ensure shippable product is **solid**.

Architecture

- Move to the cloud & evolve in flight
- Oversee and **design** code patterns
- Prevent cascading failure
- **Communicate** technical needs to business



Microsoft

Security

- Throw formality aside! Make it real
- Assume breach, plan the reaction
- Humans are human

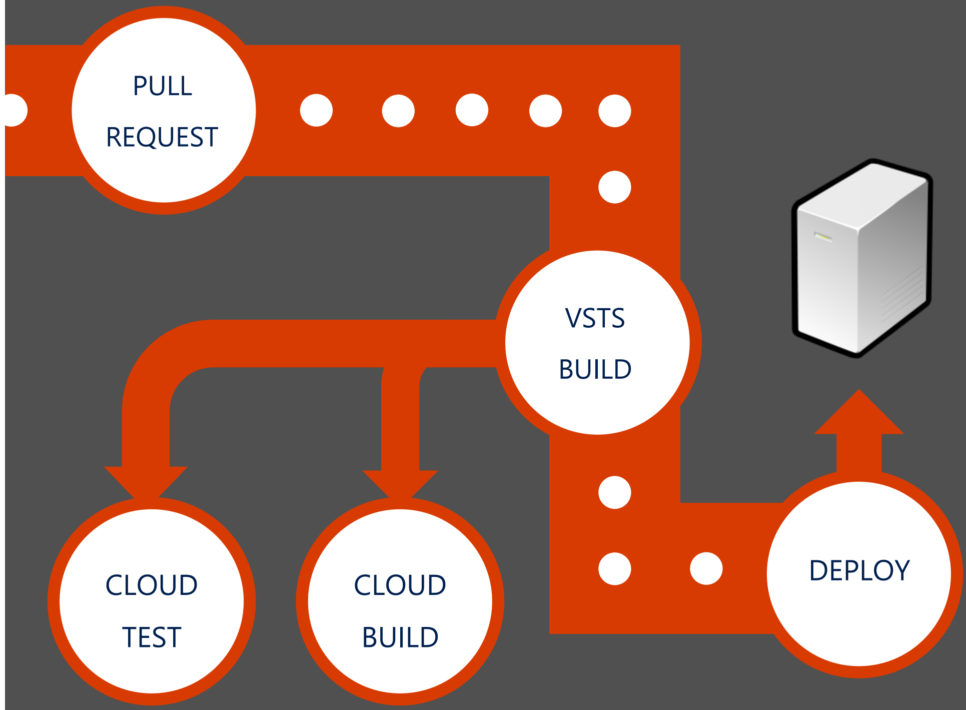
Microsoft

Deployment Practices

Deploy early and often so testers can test sooner in the Sprint!

Make a team that handles deployment and managing the CI/CD processes

Stay green throughout the sprint



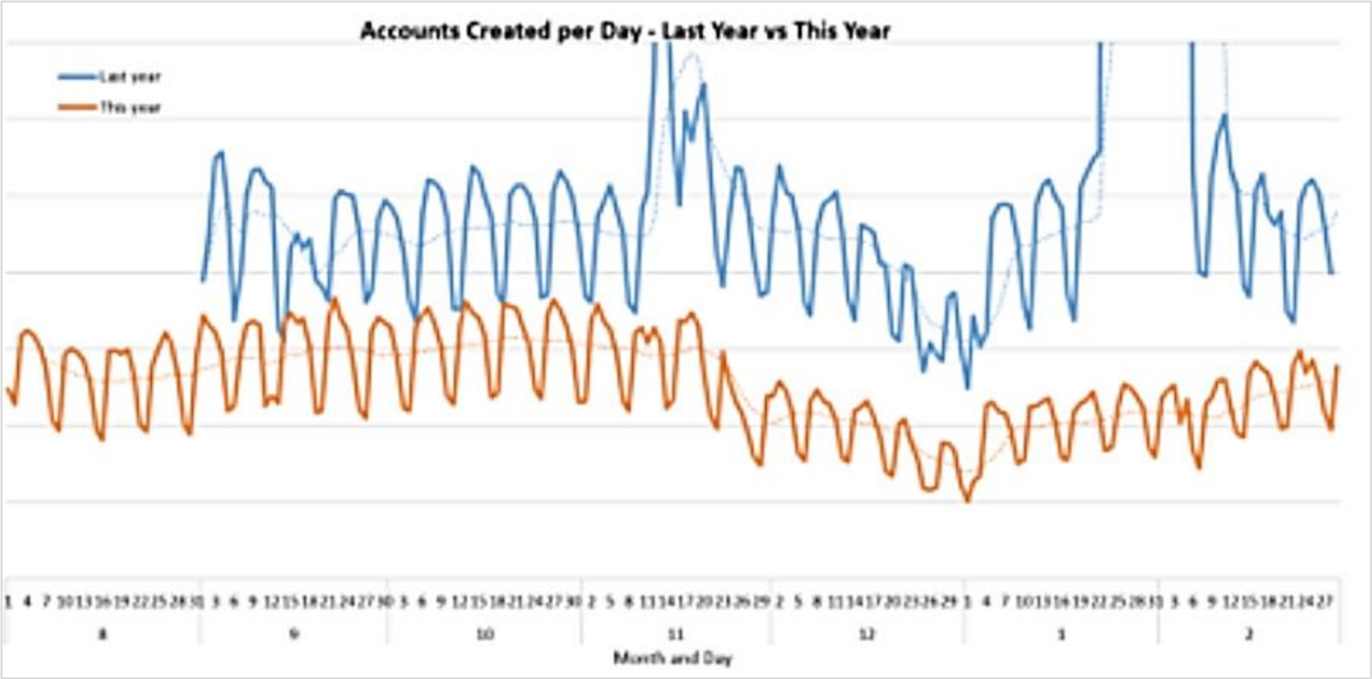
Microsoft

Running the Business on Metrics

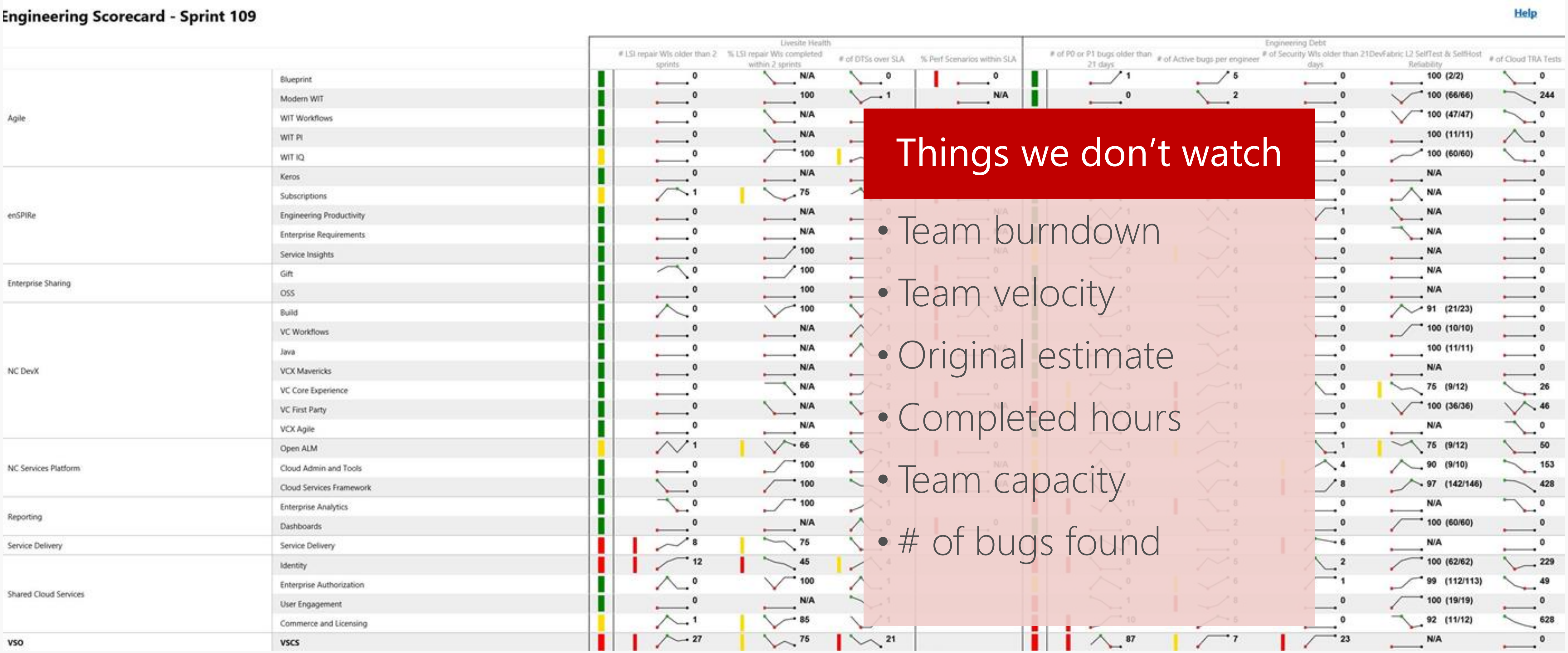
Start with what is most important/painful & evolve

Designing metrics is as hard as designing features

Bake it into the review culture to spur activity



VSTS Scorecard



Engineering Scorecard - Sprint 109

Help

		Livesite Health				Engineering Debt				
		# LSI repair WIs older than 2 sprints	% LSI repair WIs completed within 2 sprints	# of DTSs over SLA	% Perf Scenarios within SLA	# of P0 or P1 bugs older than 21 days	# of Active bugs per engineer	# of Security WIs older than 21 days	DevFabric L2 SelfTest & SelfHost Reliability	# of Cloud TRA Tests
Agile	Blueprint	0	N/A	0	0	1	5	0	100 (2/2)	0
	Modern WIT	0	100	1	N/A	0	2	0	100 (66/66)	244
	WIT Workflows	0	N/A					0	100 (47/47)	0
	WIT PI	0	N/A					0	100 (11/11)	0
	WIT IQ	0	100					0	100 (60/60)	0
enSPIRe	Keros	0	N/A					0	N/A	0
	Subscriptions	1	75					0	N/A	0
	Engineering Productivity	0	N/A					1	N/A	0
	Enterprise Requirements	0	N/A					0	N/A	0
	Service Insights	0	100					0	N/A	0
Enterprise Sharing	Gift	0	100					0	N/A	0
	OSS	0	100					0	N/A	0
NC DevX	Build	0	100					0	91 (21/23)	0
	VC Workflows	0	N/A					0	100 (10/10)	0
	Java	0	N/A					0	100 (11/11)	0
	VCX Mavericks	0	N/A					0	N/A	0
	VC Core Experience	0	N/A					0	75 (9/12)	26
	VC First Party	0	N/A					0	100 (36/36)	46
	VCX Agile	0	N/A					0	N/A	0
NC Services Platform	Open ALM	1	66					1	75 (9/12)	50
	Cloud Admin and Tools	0	100					4	90 (9/10)	153
	Cloud Services Framework	0	100					8	97 (142/146)	428
Reporting	Enterprise Analytics	0	100					0	N/A	0
	Dashboards	0	N/A					0	100 (60/60)	0
Service Delivery	Service Delivery	8	75					6	N/A	0
Shared Cloud Services	Identity	12	45					2	100 (62/62)	229
	Enterprise Authorization	0	100					1	99 (112/113)	49
	User Engagement	0	N/A					0	100 (19/19)	0
	Commerce and Licensing	1	85					0	92 (11/12)	628
VSO	**VSCS**	27	75	21		87	7	23	N/A	0

Things we don't watch

- Team burndown
- Team velocity
- Original estimate
- Completed hours
- Team capacity
- # of bugs found

> “Culture eats strategy for breakfast.”

Peter Drucker

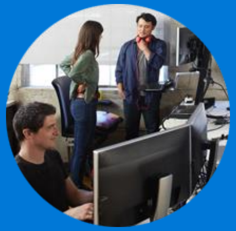
It all starts with the Culture

The DevOps conversation

PEOPLE

Collaborate more
Share common goals
Focus on improvement

BRINGING PEOPLE TOGETHER



PROCESS

Eliminate waste
Increase efficiency
Streamline feedback

DELIVERING VALUE FASTER

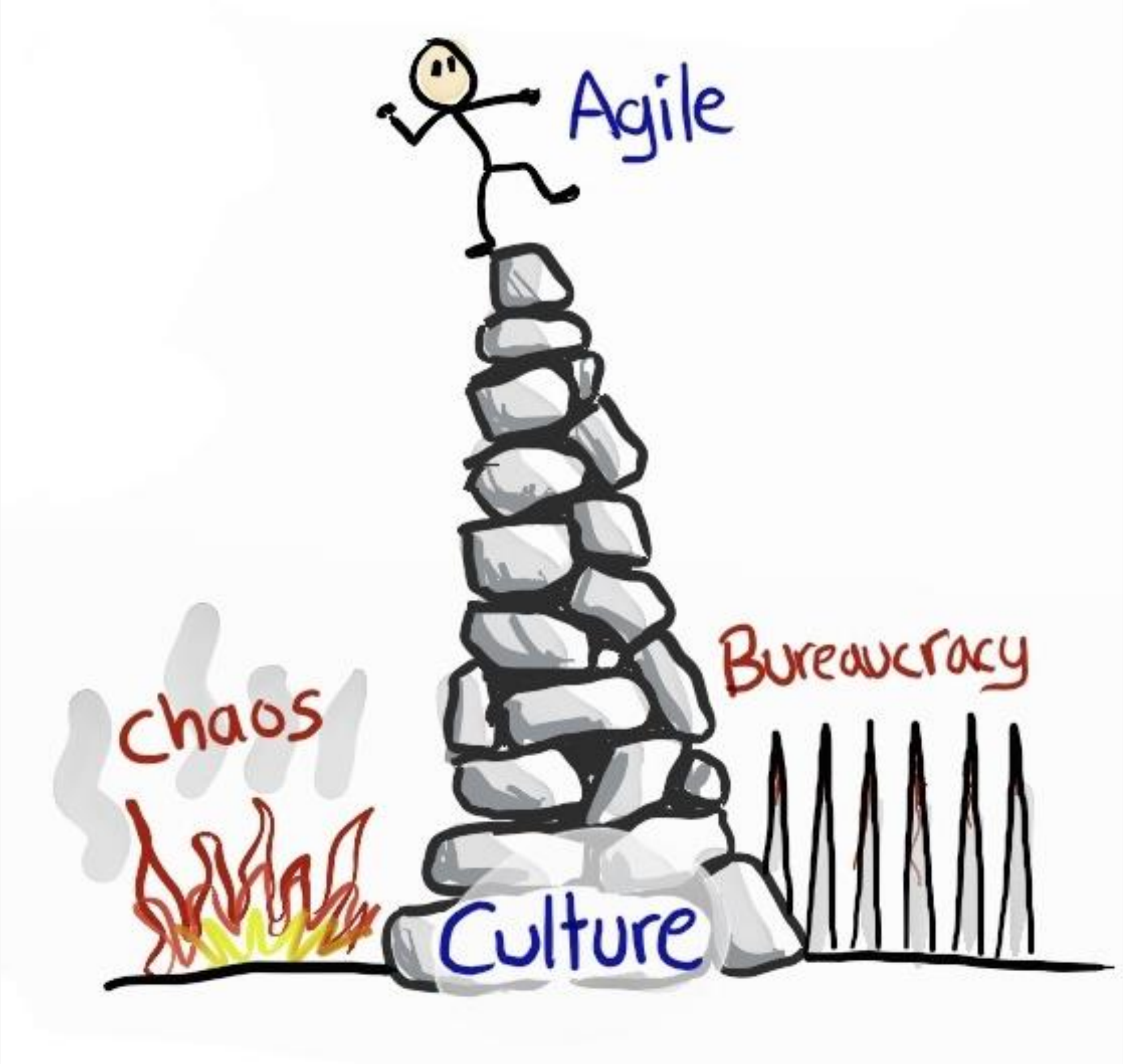
TOOLS

Enhance productivity
Enable collaboration
Facilitate experimentation

EXECUTING A DEVOPS STRATEGY

Testimonial 3: Agile Planning at Scale

Build and Protect the Culture!



Agile Methodologies – Retrospective

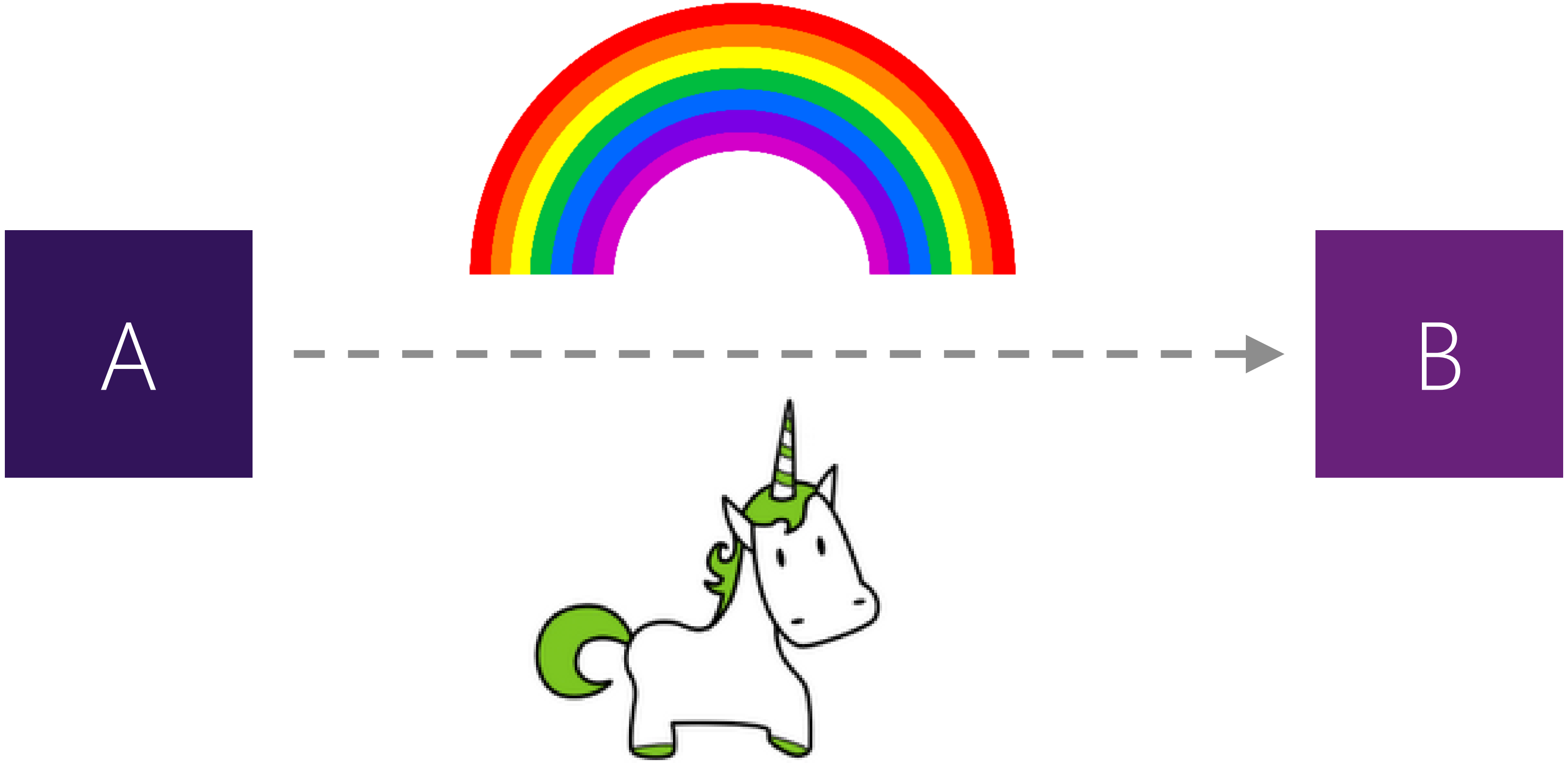
Created by Håkan Forss @hakanforss http://hakanforss.wordpress.com

Microsoft

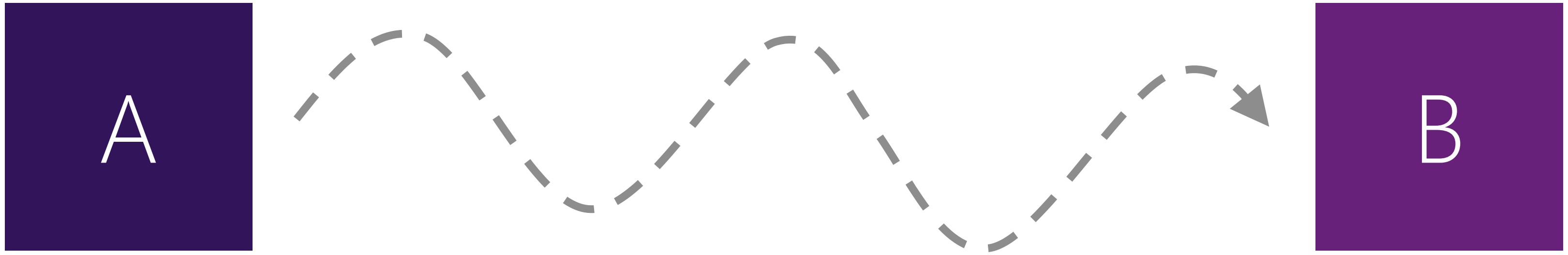
Transformation

Before	After
4-6 month milestones	3-week sprints
Horizontal teams	Vertical teams
Personal offices	Team rooms
Long planning cycles	Continual Planning & Learning
PM, Dev, Test	PM & Engineering
Yearly customer engagement	Continual customer engagement
Feature branches	Everyone in master
20+ person teams	8-12 person teams
Secret roadmap	Publicly shared roadmap
Bug debt	Zero debt
100 page spec documents	Specs in PPT
Private repositories	Open source
Deep organizational hierarchy	Flattened organization hierarchy
Success is a measure of install numbers	User satisfaction determines success
Features shipped once a year	Features shipped every sprint

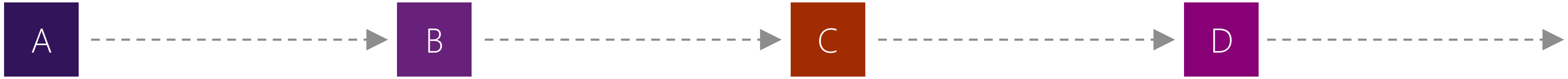
Too good to be true?



… a journey of continued improvement.



This journey does not end.



Demo + Lab Time!