DevOps

2 day introduction course

Dare to Challenge

Quint
Wellington Redwood

Content

- Day 1
  - What is DevOps?
  - Dev and Ops
  - Which problem does DevOps solve
  - Automation

- Day 2
  - Performance
  - Behavior and Attitude
  - Organization
  - Governance
  - Transformation

What is DevOps

Quint
Wellington Redwood

Definitions of DevOps

'DevOps (a portmanteau of "development" and "operations") is a software development method that stresses communication, collaboration (information sharing and web service usage), integration, automation and measurement of cooperation between software developers and other information-technology (IT) professionals. DevOps acknowledges the interdependence of software development and IT operations. It aims to help an organization rapidly produce software products and services and to improve operations performance - quality assurance' (http://en.wikipedia.org/wiki/DevOps)

Jez Humble

DevOps is "a cross-disciplinary community of practice dedicated to the study of building, evolving and operating rapidly-changing resilient systems at scale."

the agile admin

DevOps is the blending of tasks performed by a company's application development and systems operations teams.

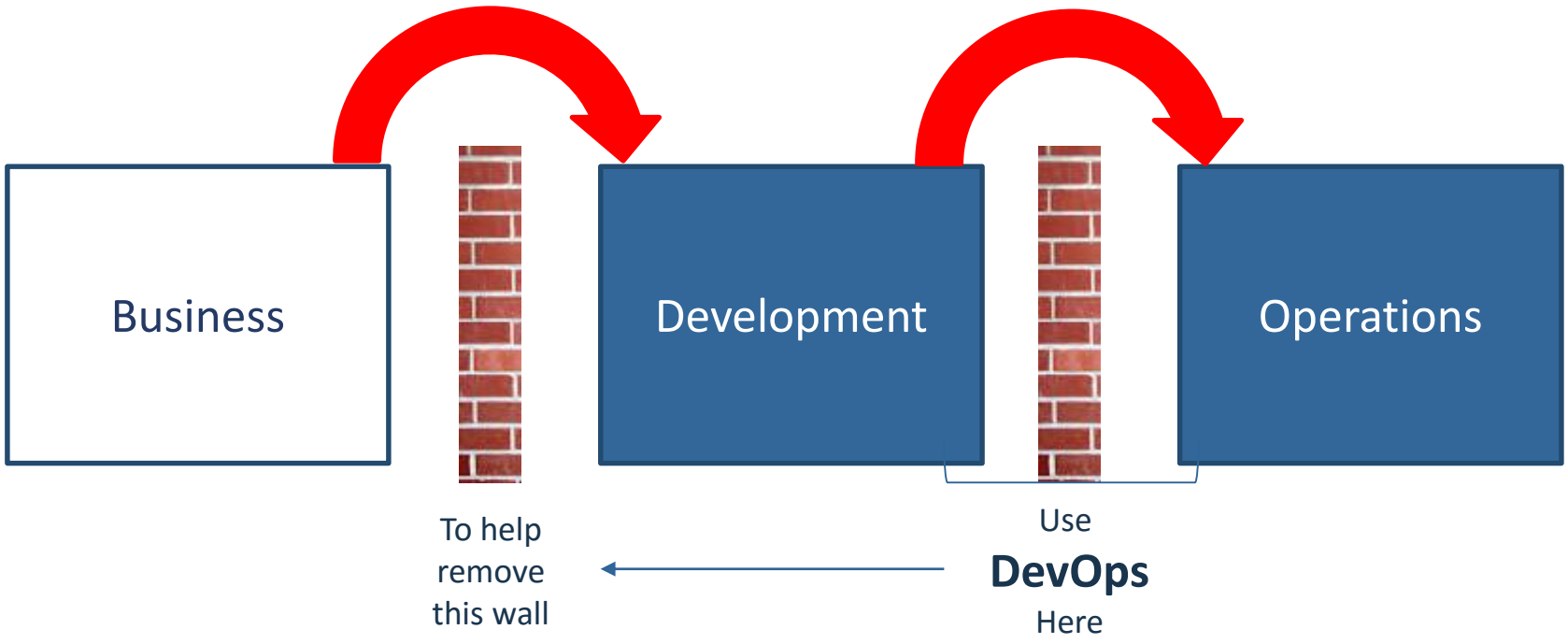
"DevOps is just the Agile principle taken to the full enterprise"

'DevOps is just the Agile principle taken to the full enterprise. It provides a more comprehensive end-to-end perspective that enables enterprise-scale transformation'

'DevOps is just a natural extension of Agile. It takes the guiding principles and best practices that Agile brought to developers and applies them on a much broader scale across the organization.'

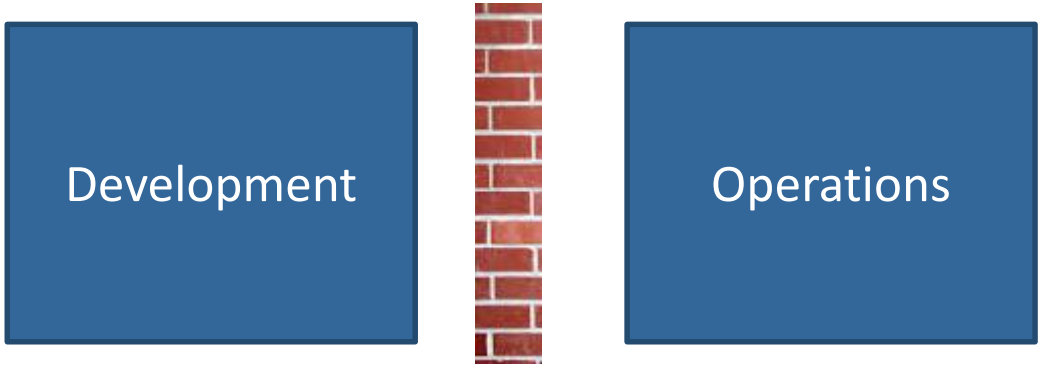
* DevOps is Agile for the rest of the company
Mar 4, 2015 Posted by Tony Bradley In Blogs, (http://devops.com/blogs/devops-is-agile-for-the-rest-of-the-company/)

What is the current situation? We have built walls – intentionally or not – that limit performance



"DevOps is a response to the growing awareness that there is a disconnect between what is traditionally considered development activity and what is traditionally considered operations activity. This disconnect often manifests itself as conflict and inefficiency"
(What is DevOps? Dev2ops.org)

The wall of confusion



Extreme focus on change

- More urgent, date/driven projects
- Fragile code going into production
- Releases with ´turbulent installs´
- Longer release cycles amortize `cost of deployments`
- Backlog of infrastructure projects that could fix root cause and reduce costs

Extreme focus on stability

- Fragile applications failing
- Difficulties identifying root cause
- Taking too long to restore service
- Extensive firefighting and unplanned work
- Unfinished planned project work
- Frustrated customers
- Market share decreasing

DevOps is …

DevOps: A culture shift or movement that encourages great collaboration (aka teamwork) to foster building better quality software more quickly with more reliability.

DevOps is NOT:

- A role, person, or organization
- Something only systems administrators do
- Something only developers do
- Writing Chef and Puppet scripts
- Tools

DevOps is a way of building software while still getting a speed to market without sacrificing quality and reliability.

What is DevOps

- DevOps is about identifying bottlenecks and removing the waste from the system to continuously improve your organization to deliver better software. Bottlenecks can be in the form of long approval processes, manual processes, and even certain resources.

The market sees the solution from various points of view

What are the points of view of DevOps?

Organization	• From functional siloes … • To multidisciplinary customer-focused teams
Performance	• From different KPIs for dev and ops … • To common goals and measures
Culture	• From “different” … • To “together”
Flow	• From complex and slow … • To end-to-end and simple
Automation	• From own tools … • To integrative tooling

DevOps helps to achieve the goals of IT

Quint Definition of DevOps:

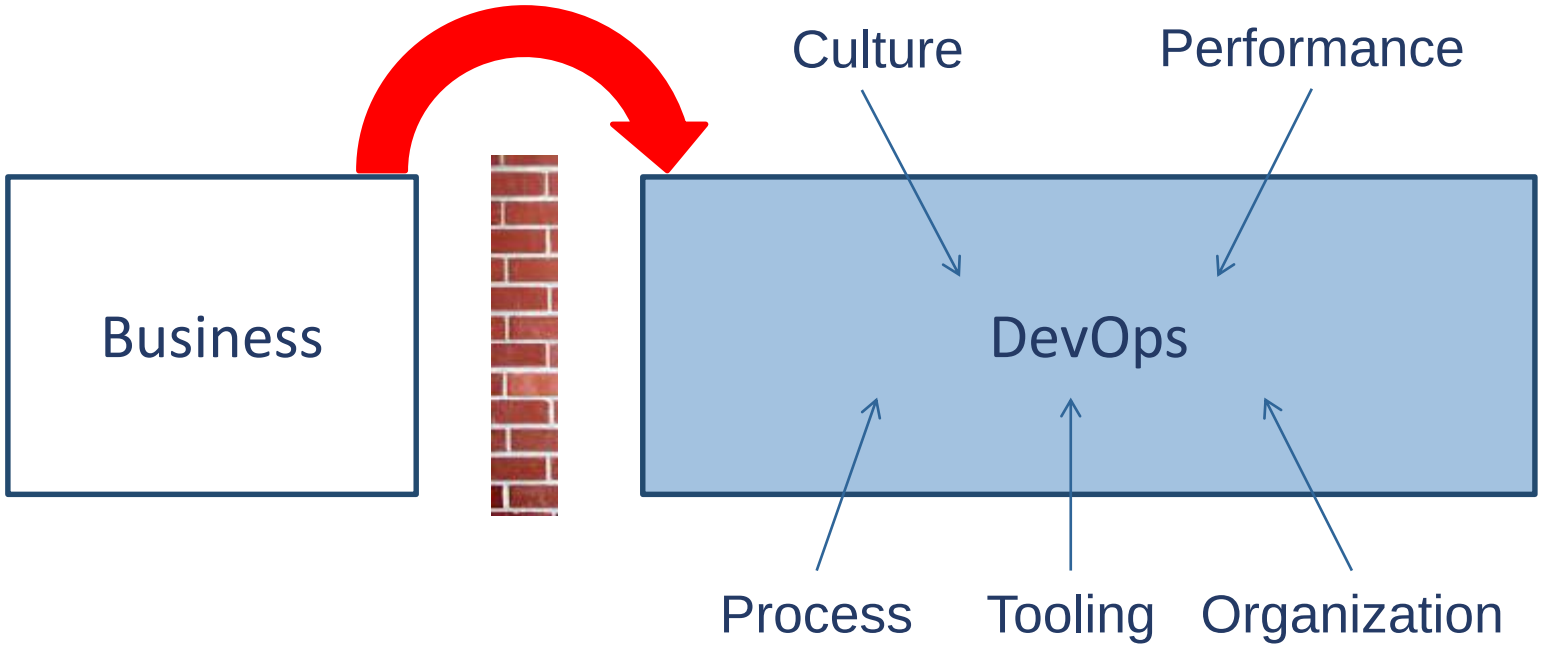
DevOps (a portmanteau of development and operations) is an organizational philosophy that stresses communication, collaboration and integration between information technology (IT) software developers and IT operations professionals. DevOps uses the interdependence of software development and IT operations to create higher quality products and services. It aims to help an organization rapidly deliver the value sought by the customers of IT

DevOps is, in short, “movement” to bring Development and Operations closer together to achieve four key goals:

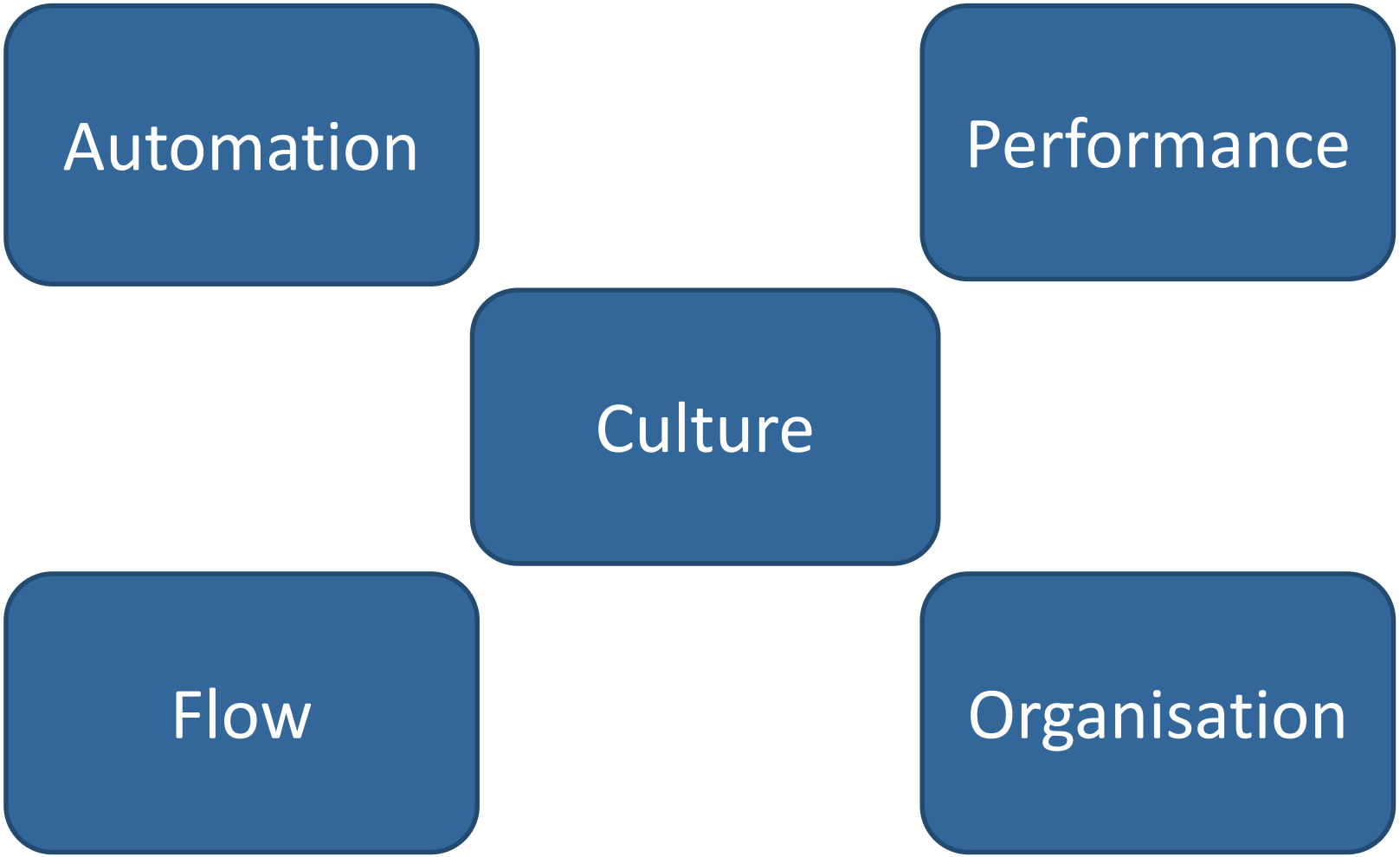
- Improve **Quality** of delivery (First time right)
- Deliver higher **Business Value** (Customer)
- Increase **Speed** of delivery (Performance)
- Improve **Efficiency** (Reduce Cost)

4.2 New Situation

Making life easier?



DevOps is the solution to a problem



Quint
Wellington Redwood

DevOps vocabulary

Continuous Delivery is a software strategy that enables organizations to deliver new features to users as fast and efficiently as possible. The core idea is to create a repeatable, reliable and incrementally improving process for taking software from concept to customer. The goal is to enable a constant flow of changes into production via an automated software Continuous Delivery pipeline.

Continuous integration is the practice, in software engineering, of merging all developer working copies to a shared mainline several times a day. The main aim of CI is to prevent integration problems

Test-driven development is a software development process that relies on the repetition of a very short development cycle: first the developer writes an (initially failing) automated test case that defines a desired improvement or new function, then produces the minimum amount of code to pass that test, and finally refactors the new code to acceptable standards.

Test automation is the use of special software (separate from the software being tested) to control the execution of tests and the comparison of actual outcomes with predicted outcomes

DevOps adoption

The adoption of DevOps is being driven by factors such as:

1. Use of agile and other development processes and methodologies
2. Demand for an increased rate of production releases from application and business stakeholders
3. Wide availability of virtualized and cloud infrastructure from internal and external providers
4. Increased usage of data center automation and configuration management tools
5. It has also been suggested that side-effect of the dominant, traditional US management style ("the Sloan model" vs. "the Toyoda model") guarantees the development of silos of automation, thus creating "the DevOps gap" that then requires DevOps capabilities to address

DevOps Goals

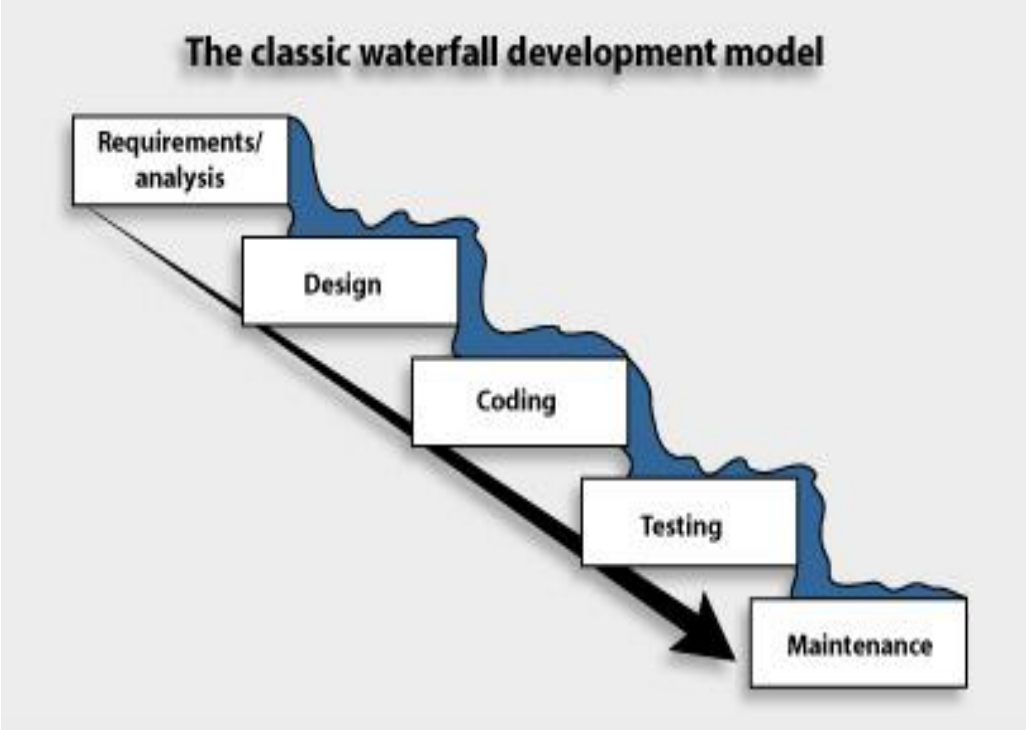
- Better align IT responsiveness and capabilities to business needs
- Produce smaller, more frequent software releases
- Reduce effort with software development, transition and operation
- Improve time to market
- Improve quality of code
- Improve quality of software deployments
- Reduce cost of product iteration and delays
- Intill a culture of communication and collaboration
- Improve productivity
- Improve visibility into IT requirements and processes

Other frameworks

Quint
Wellington Redwood

Classic software development

1. Determine requirements
2. Complete the design
3. Do the coding and unit testing
4. Perform functional and integration tests and fix bugs
5. Perform acceptance and deploy



Agile Software Development

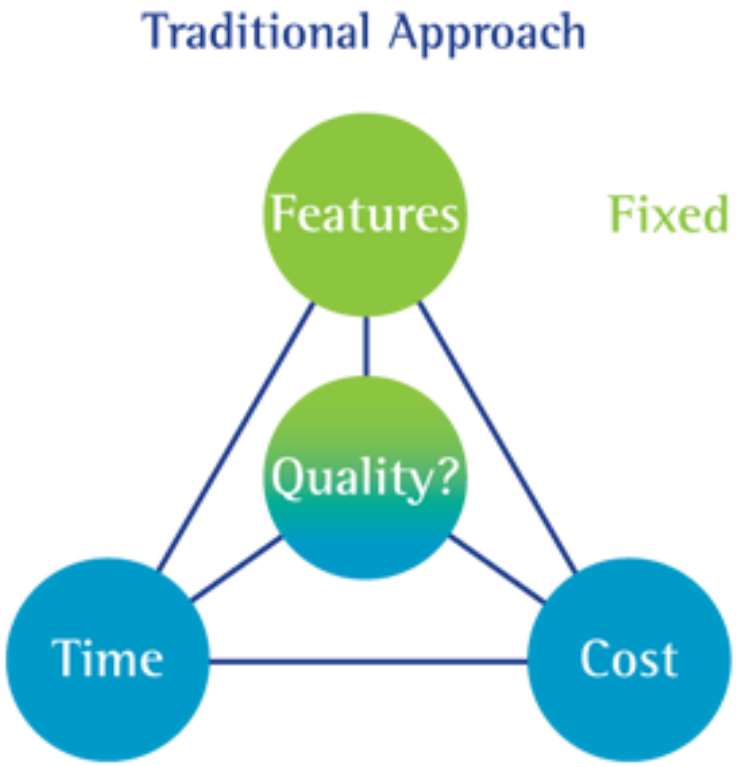
Most Agile implementations use a routine and formal daily face-to-face communication among team members. This specifically includes the customer representative and any interested stakeholders as observers.

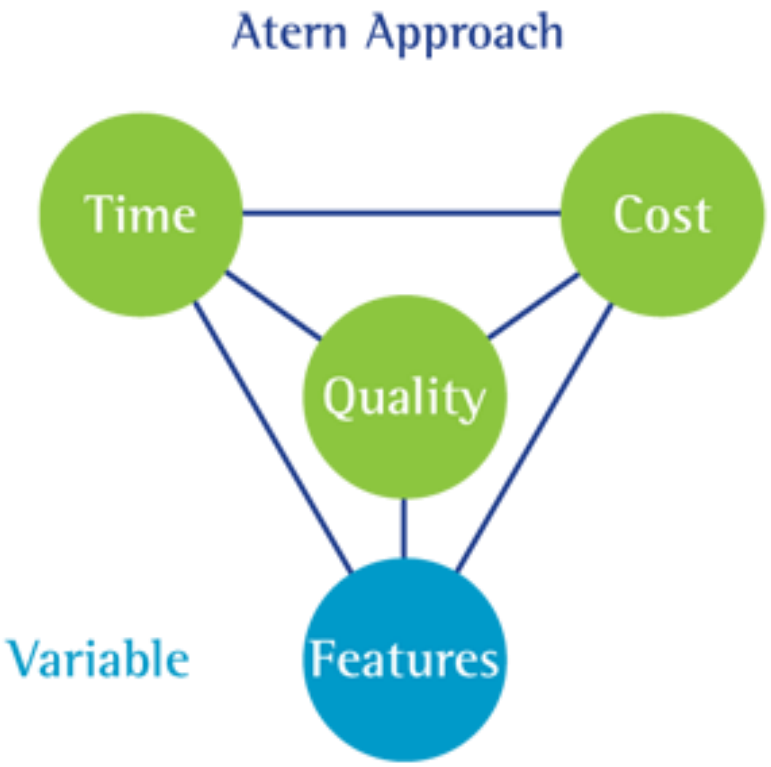
Agile development emphasizes working software as the primary measure of progress. This, combined with the preference for face-to-face communication, produces less written documentation than other methods.

Agile methods are sometimes characterized as being at the opposite end of the spectrum from plan-driven or disciplined methods. Agile methods lie on the adaptive side of this continuum. Adaptive methods focus on adapting quickly to changing realities.

Individuals and Interactions – In agile development, self-organization and motivation are important, as are interactions like colocation and pair programming.
Working software – Working software will be more useful and welcome than just presenting documents to clients in meetings.
Customer collaboration – Requirements cannot be fully collected at the beginning of the software development cycle. Therefore, continuous customer or stakeholder involvement is very important.
Responding to change – Agile development is focused on quick responses to change and continuous development.

Traditional versus Agile





Waterfall

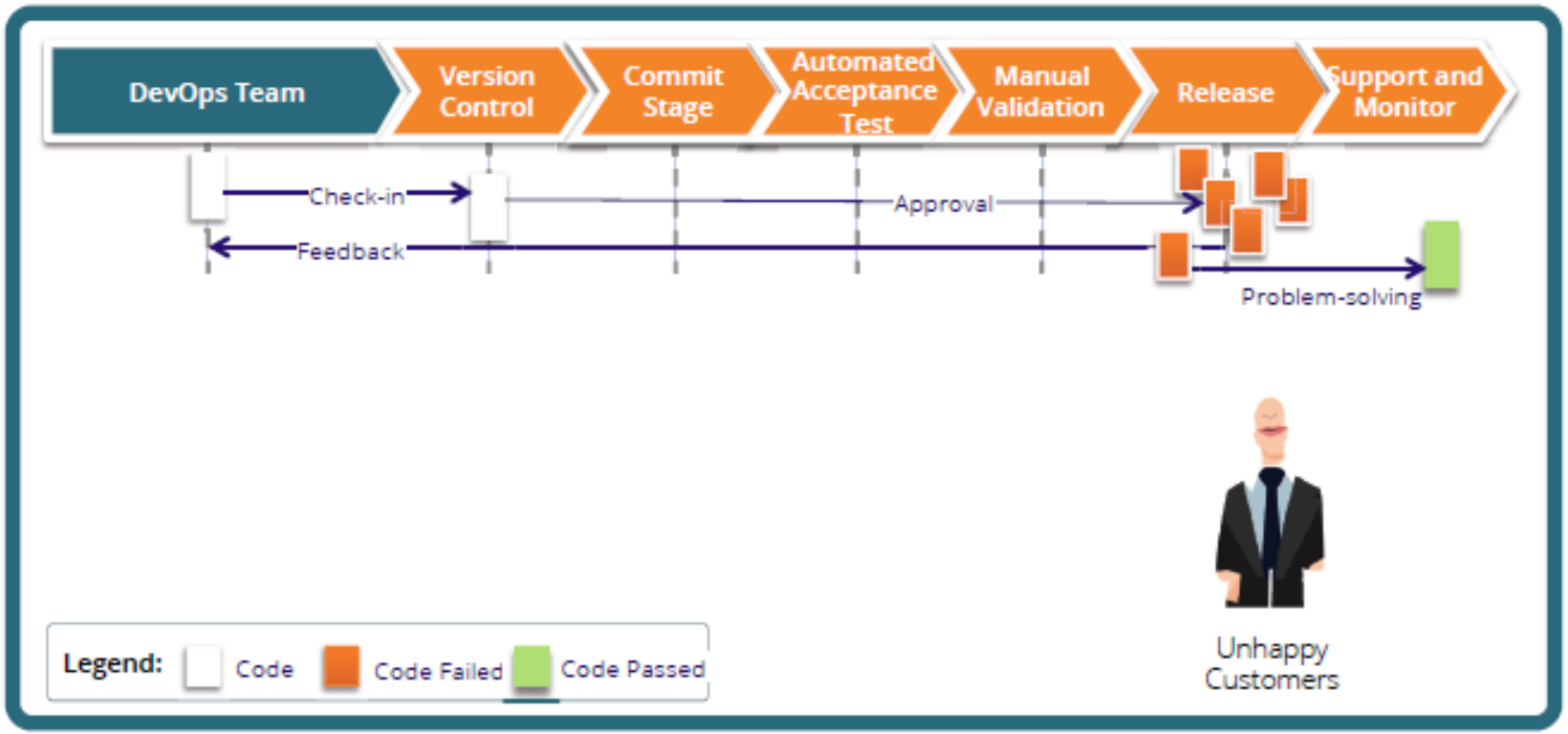
- Start with a complete design
- Building is followed by testing the final product
- Finally, testing in practice
- No feedback loops
- Plan-driven

Agile

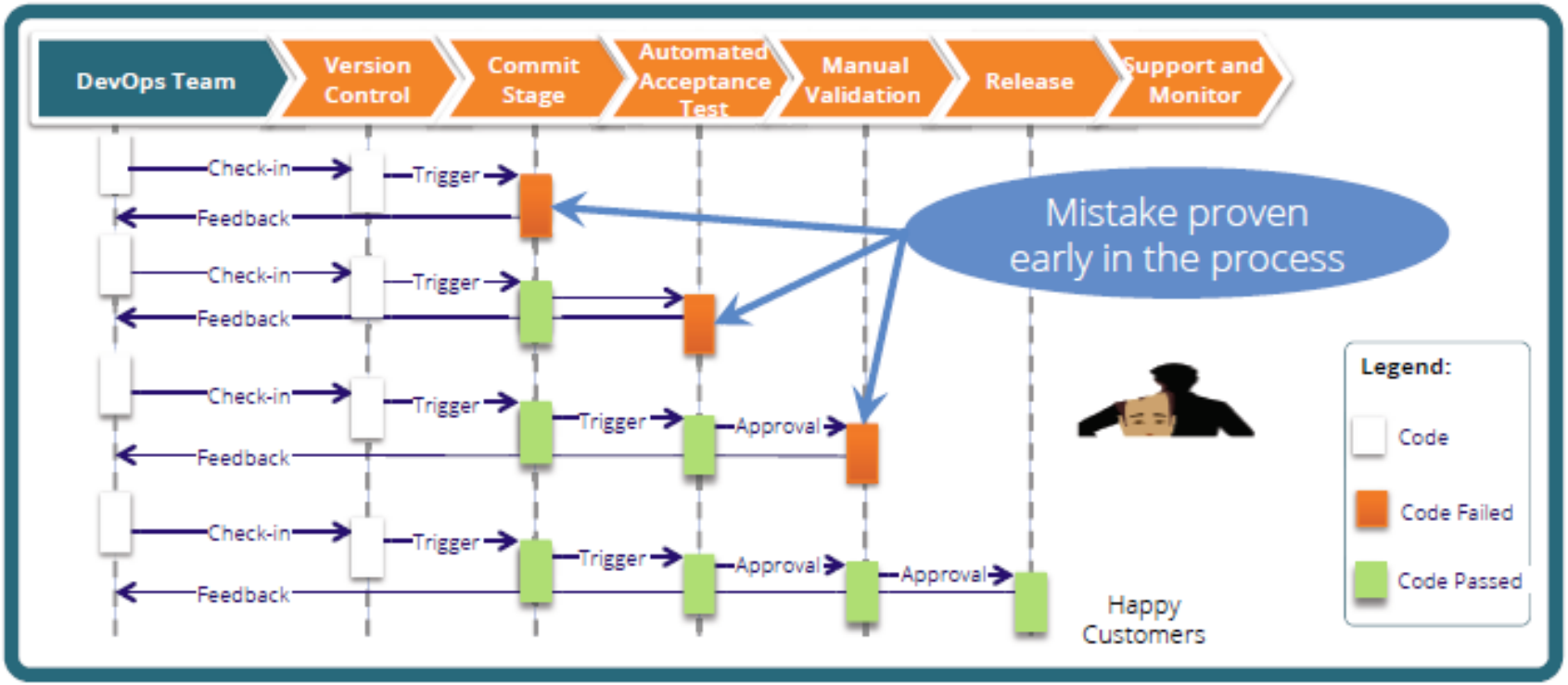
- Every Sprint delivers working software that can be used in practice
- Starts with delivering basic functionality to which features are added
- Value-driven

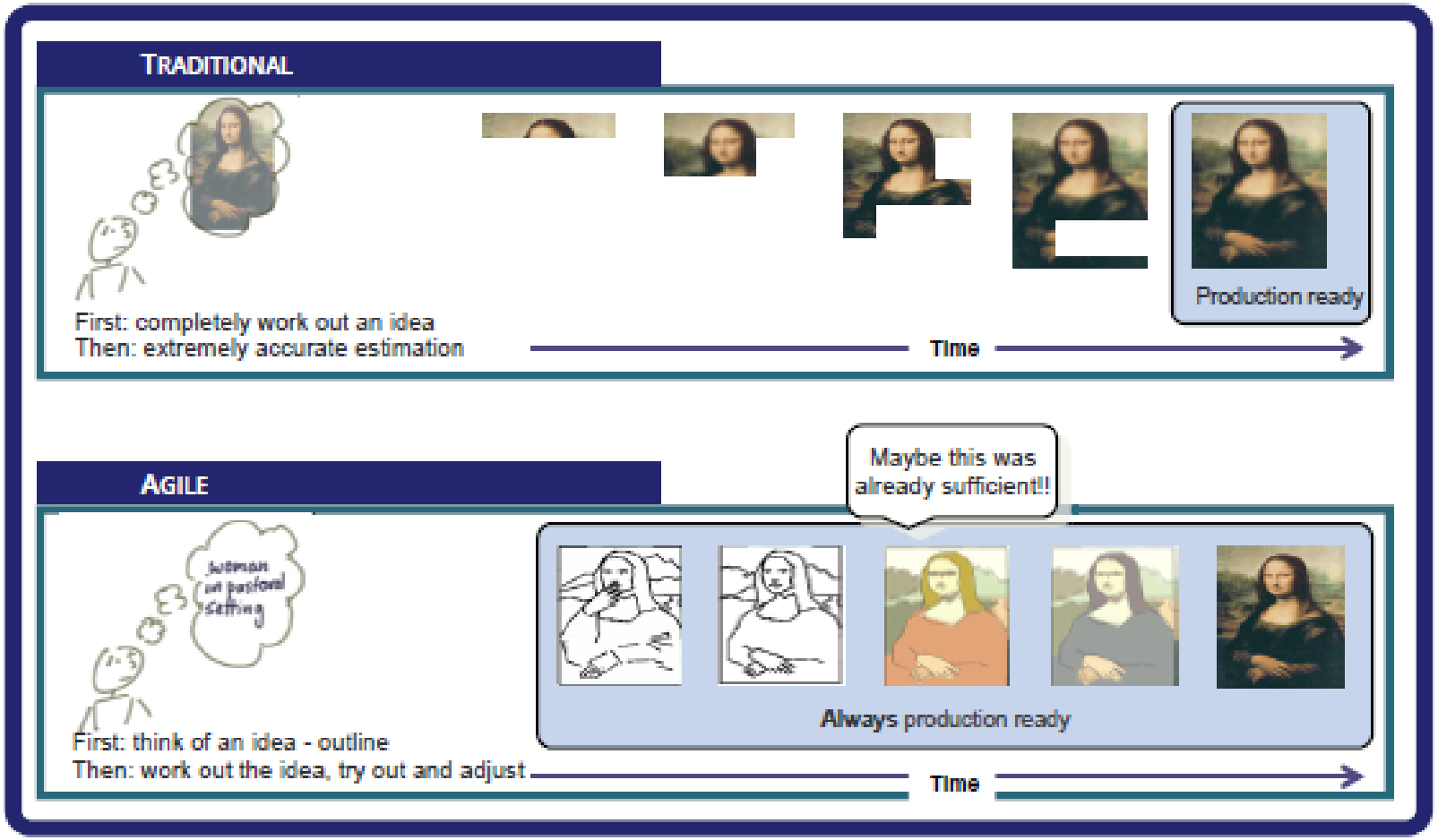
Traditional Approach and No Quality at the Source

- In traditional organizations IT processes are not automated.
- There is a long time between the update of a service and the moment problems are uncovered and solved
- as shown in the following figure.



- A DevOps team having the service mindset develops a product using the lifecycle as shown in the following figure




TRADITIONAL
Production ready
First: completely work out an idea
Then: extremely accurate estimation
Time
AGILE
Maybe this was already sufficient!!
Always production ready
First: think of an idea - outline
Then: work out the idea, try out and adjust
Time

Dev and Ops Realities

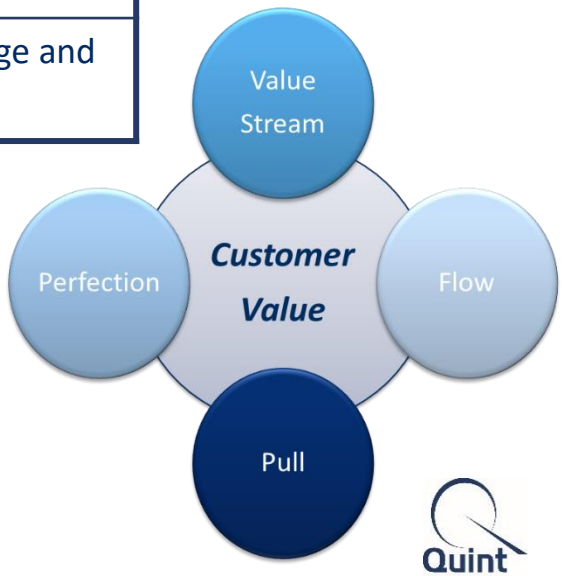
Dev and Ops are each taking steps to improve their quality and velocity

- Agile and lean software development practices
- Agile and lean service management practices
- Virtualized and cloud infrastructure from internal and external providers
- Infrastructure as a code
- Data center automation and configuration management tools
- Monitoring and self-healing technologies

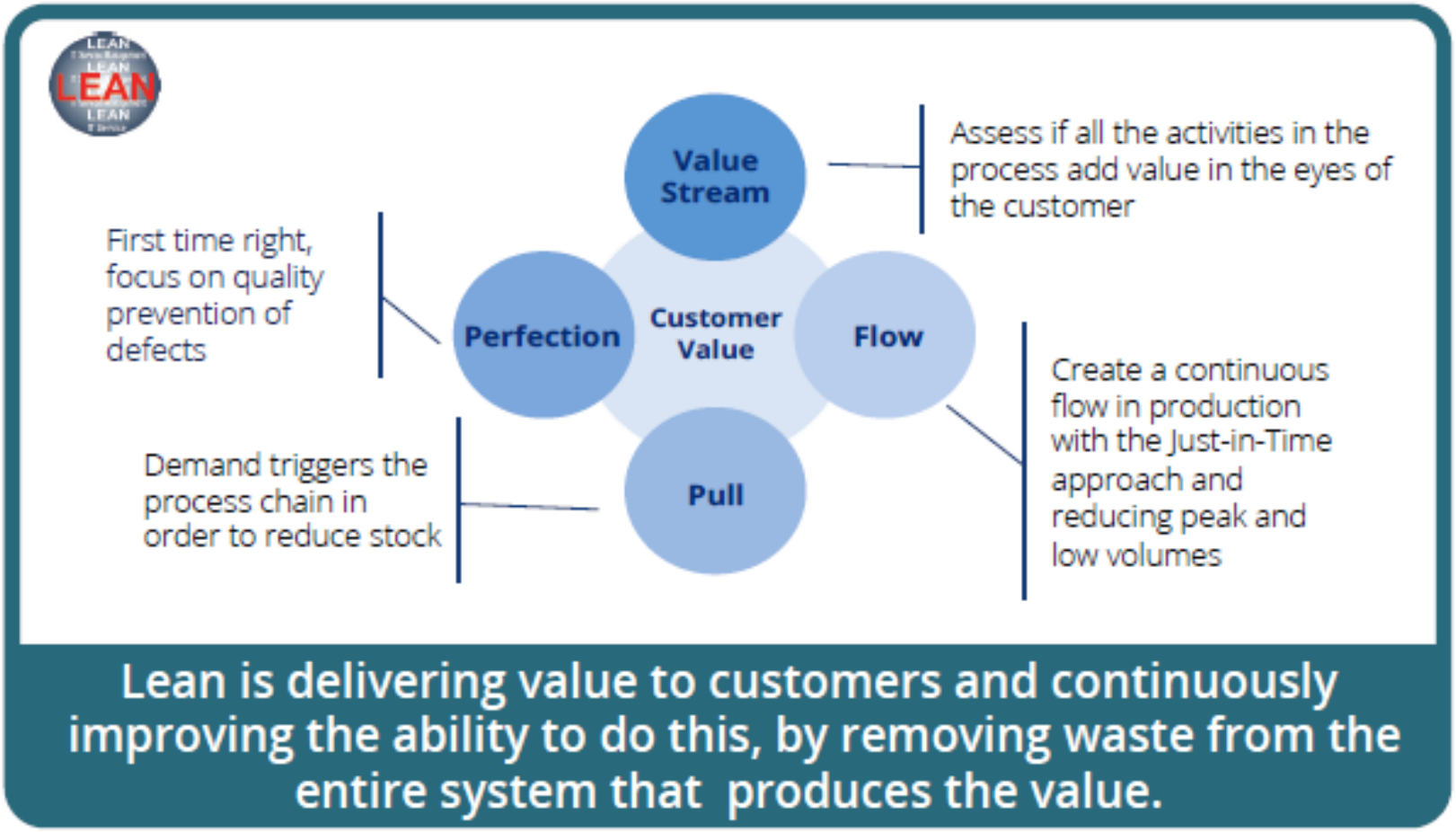
Unfortunately, Dev and OPS are not working together on these initiatives

Lean is a paradigm shift

Traditional	Lean
Managers have all the answers	Manager should ask the right questions, employees should have the answers as a team
Managers do the thinking, workers concentrate on doing	Managers facilitate the workers to add value
Activities are done, because they are asked to be done	Activities are only done if they add value for the customer
A certain rate of defects is unavoidable	Defects can be eliminated
Knowledge is power	Power comes from sharing knowledge and applying it as a team



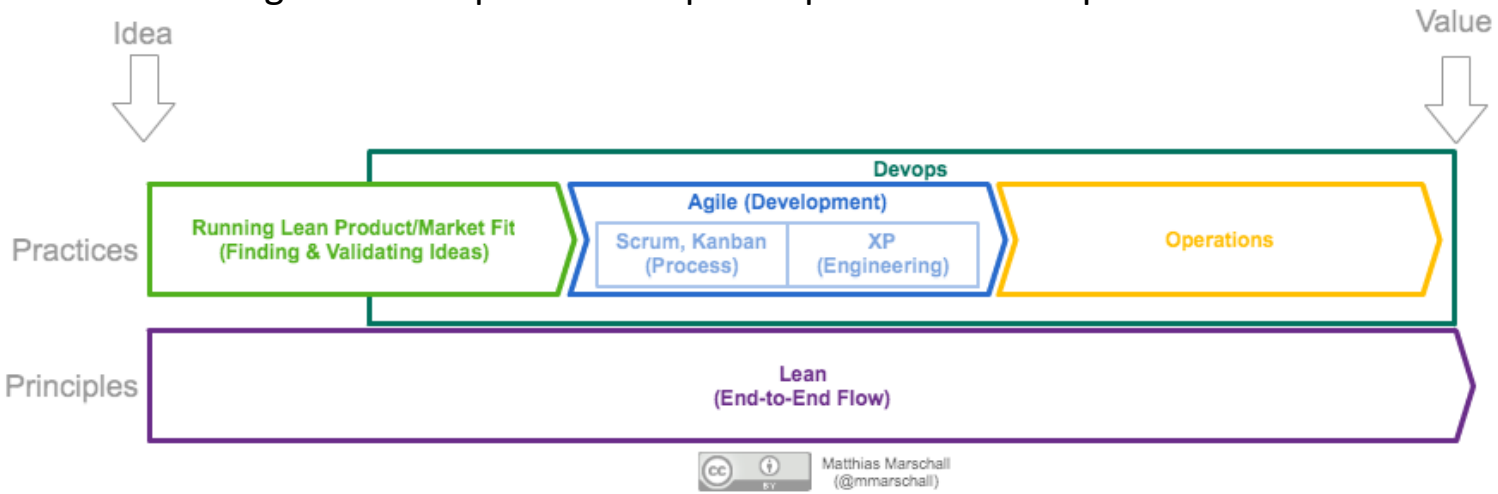
Lean focuses on creating the value for customer.



Lean is delivering value to customers and continuously improving the ability to do this, by removing waste from the entire system that produces the value.

Lean IT, Agile, ITIL and DevOps

- LSD – “Lean Software Development” Application of Lean IT to Software Development – same principles as Agile.
- “Since one of the Agile principles (and also very Lean) is to make business people and developers work together daily throughout the project, shouldn’t we call it BusDevOps…or value stream management
- Although they were independent approaches, they have lots of synergies. A DevOps team could use:
  - Agile approaches to accelerate TTM mainly in Dev side (use SCRUM and XP methodologies), and involve business (use roles of “product owners” as “business ambassadors”)
  - Lean techniques to create and evolve DevOps team both in Dev and Ops sides (Team Vision, Visual Management, Team Leader Agenda, Retrospective Meeting, VSM, VoC, …)
  - ITIL knowledge and best practice to speed up continuous improvement.

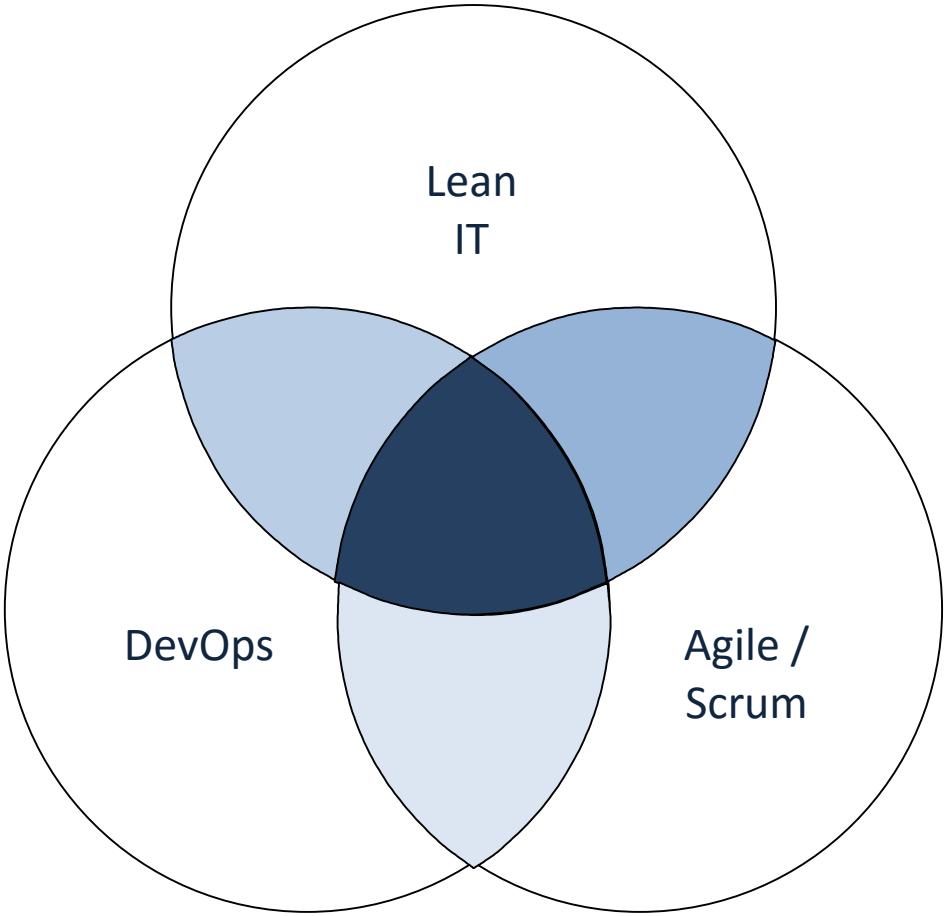


How are Lean, Agile, and DevOps related to each other?, Matthias Marschall

Lean, Agile and DevOps – Similarities and differences

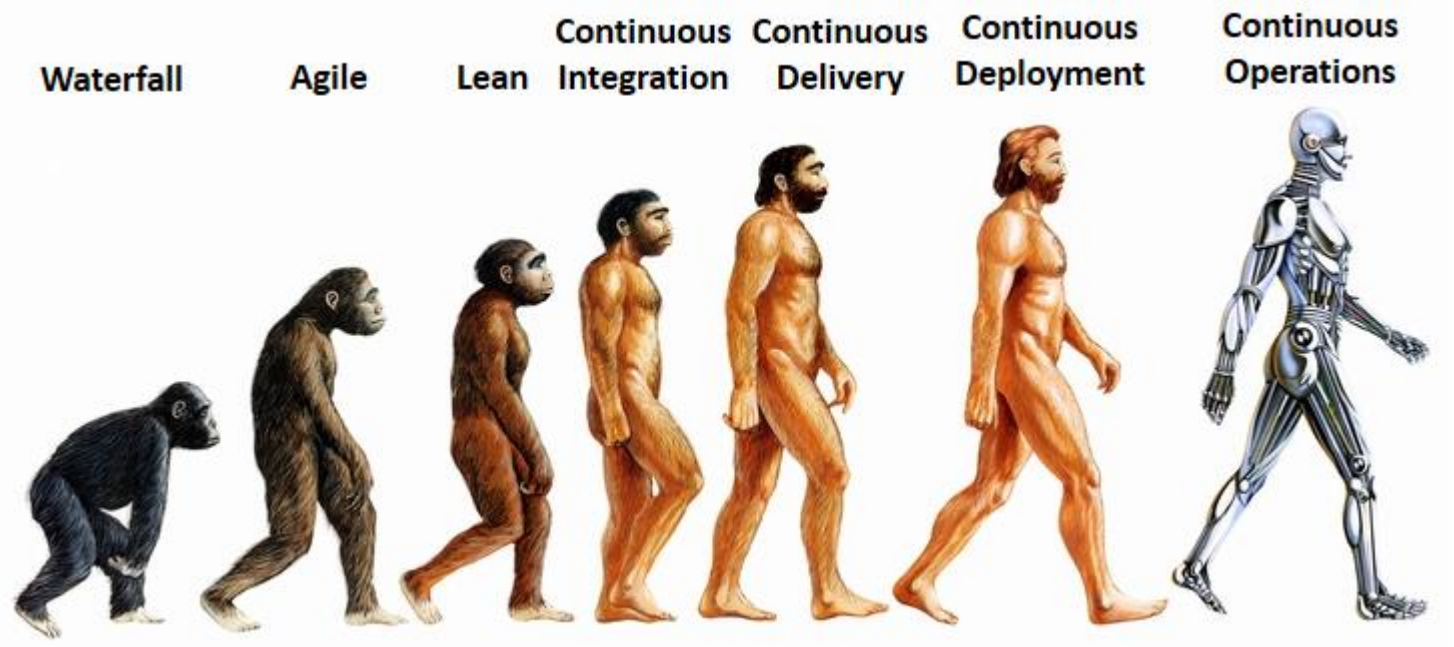
	Lean IT	Agile/Scrum	DevOps
Main Focus	Value for Customer	Improve TTM	Reduce silos inefficiencies
Organization	All or a small process	Business & Projects	Projects & Operations
Visual Mgmt.	Core	Core	Recommended
People (B&A)	Integral transformation	Project Management And Devs Teams	Join Teams (Dev and Ops)
Processes	Value Flow	Project Management (SCRUM, XP, ..)	ITSM and PPM processes
Tools / Artifacts	CTQ, VSM, Team Vision, DILO, Kaizen, A3…	User stories, Kanban, Product Owner	Automatic Test. and Deploy. Continues deployment
Goals / Indicators	% Initiatives for improvement % Savings Time & Costs by waste reduction Customer and employee Satisf Quality	% Successful projects % User stories implemented Business satisfaction	% Post-release incidents % Successful changes/releases IT Team performance (E2E)

Lean, Agile and DevOps synergies



- There are **synergies** in the 3 approaches, although you can follow your path **independently**.
- **There is no recommended path**– although abroad Lean focus could guide you to cover the most urgent necessities
- **They are no and end by themselves,** but they help you to **obtain your goals** and fulfil your requirements (agility, efficiency, continuous deployment, continuous delivery, continuous improvement, …)
- **Lean** is the **most global approach**, you can use this philosophy both for a particular department or the global Enterprise

DevOps is an evolution

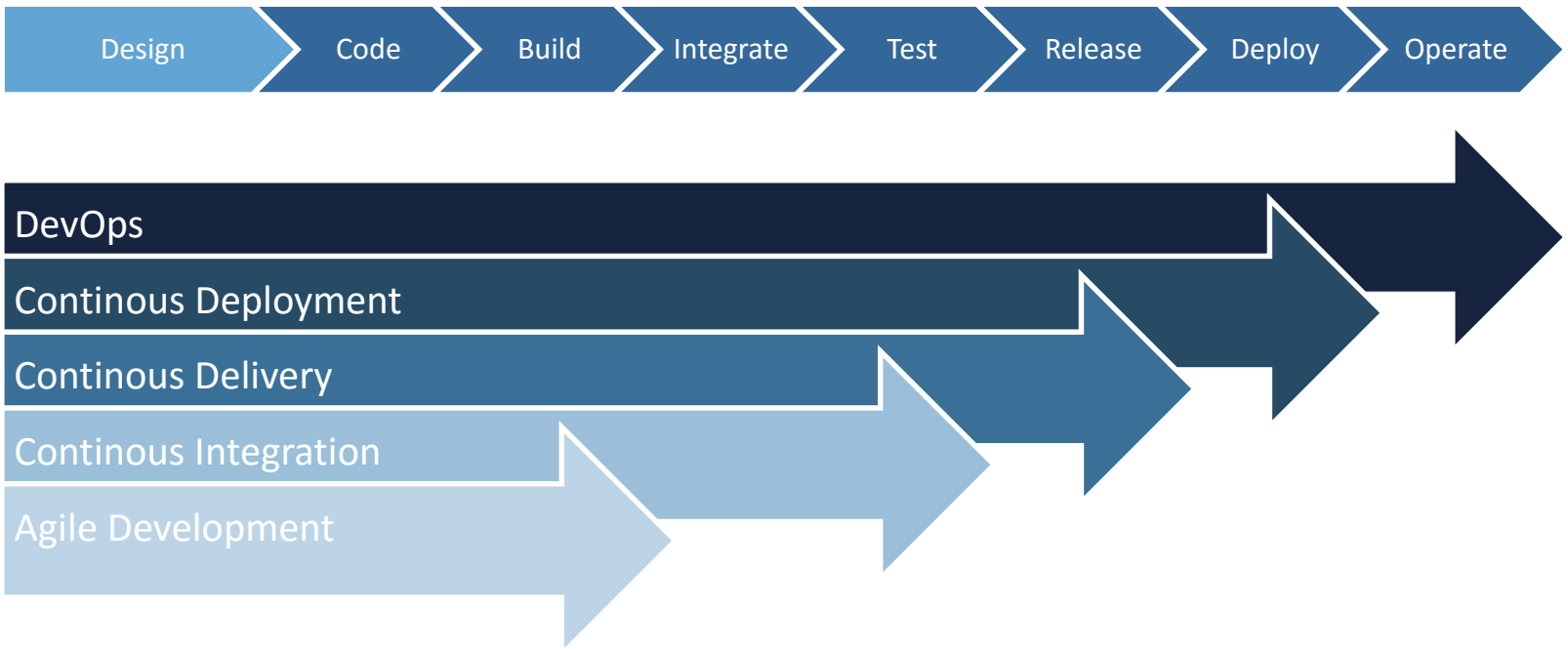


DevOps an evolution

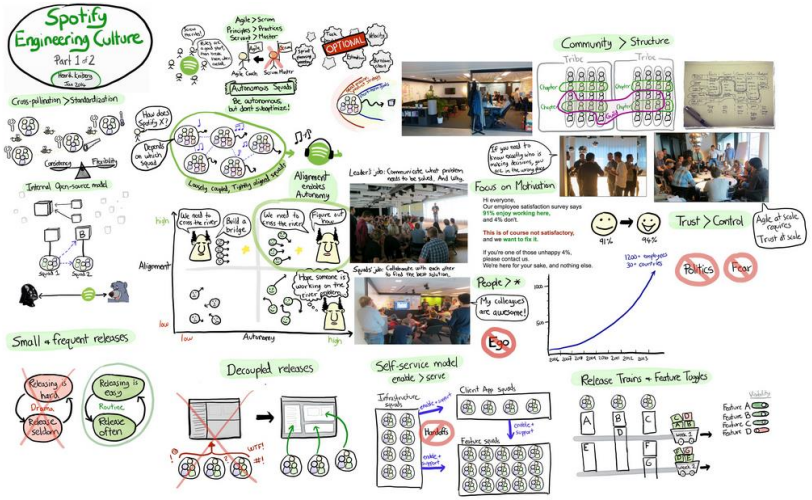
New Way of Delivering Software

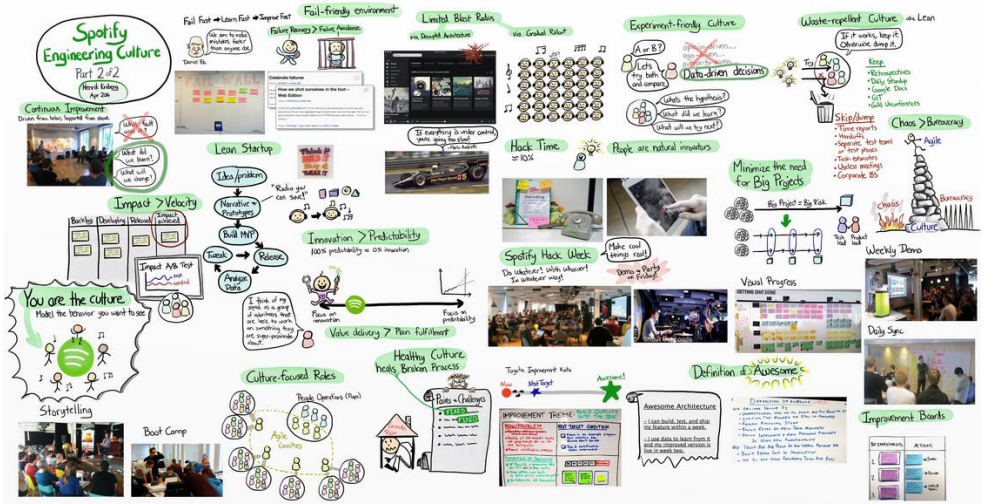
Old Way	New Way
Focus on Ship Date	Focus on Working Software
Prolong Deployments	Release More Frequently
Heroic Efforts	Repeatable & Predictable Processes
Done when code is built	Never done – Always On
Ops runs applications	Everyone is responsible
Ops and Security dictate	Ops and Security Participate
Story Points	Customer Satisfaction
Handoffs	Feedback Loops

Evolution DevOps



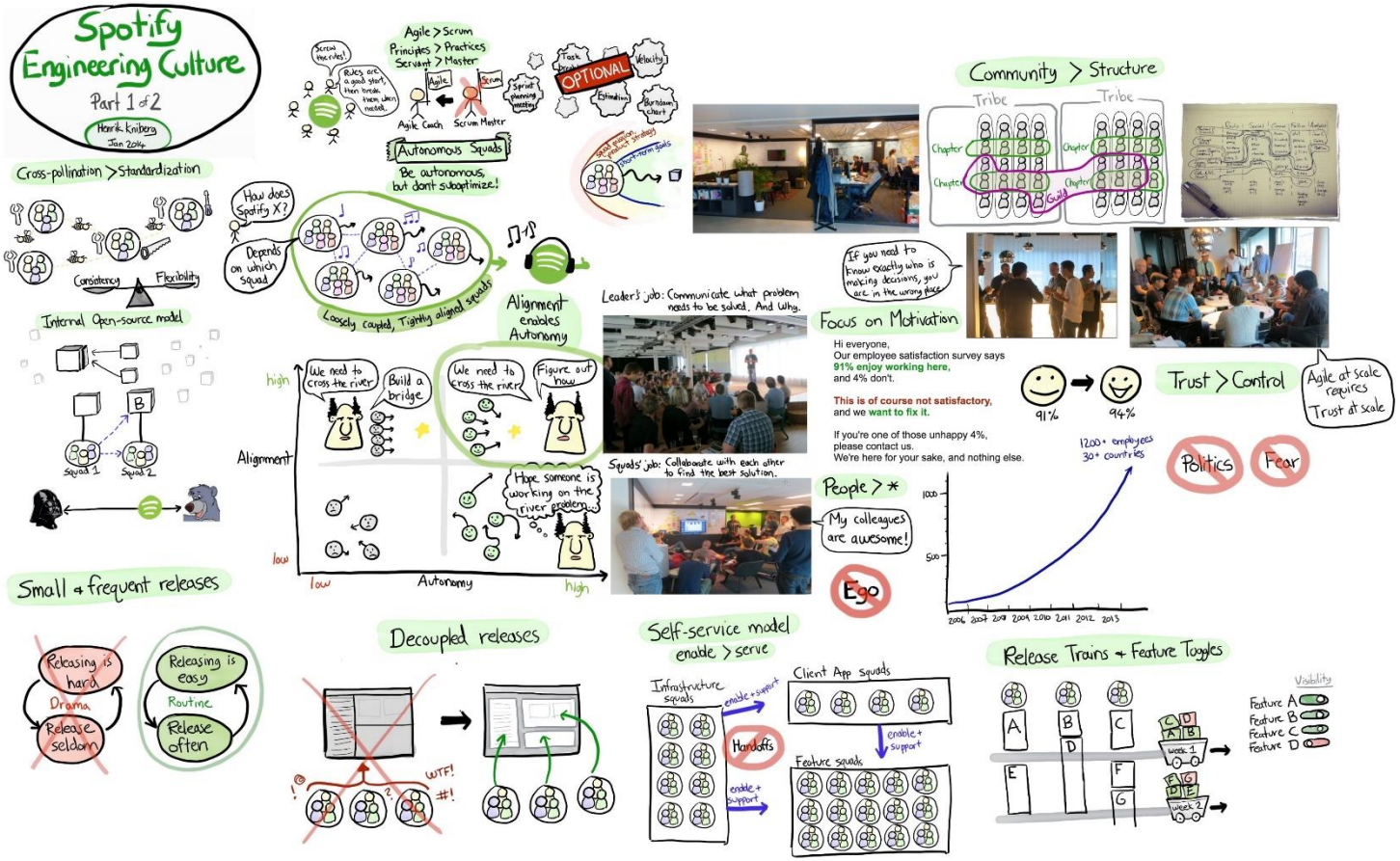
Integrate all the useful bits





Tip: View clips at 'Google Spotify Agile part 1 & 2'

Integrate all the useful bits



Tip: View clips at 'Google Spotify Agile part 1'

Integrate all the useful bits (continue…)

Tip: View clips at 'Google Spotify Agile part 2'

Which problem does DevOps solve

Quint
Wellington Redwood

Why did DevOps develop? It is the solution to one or more problems …

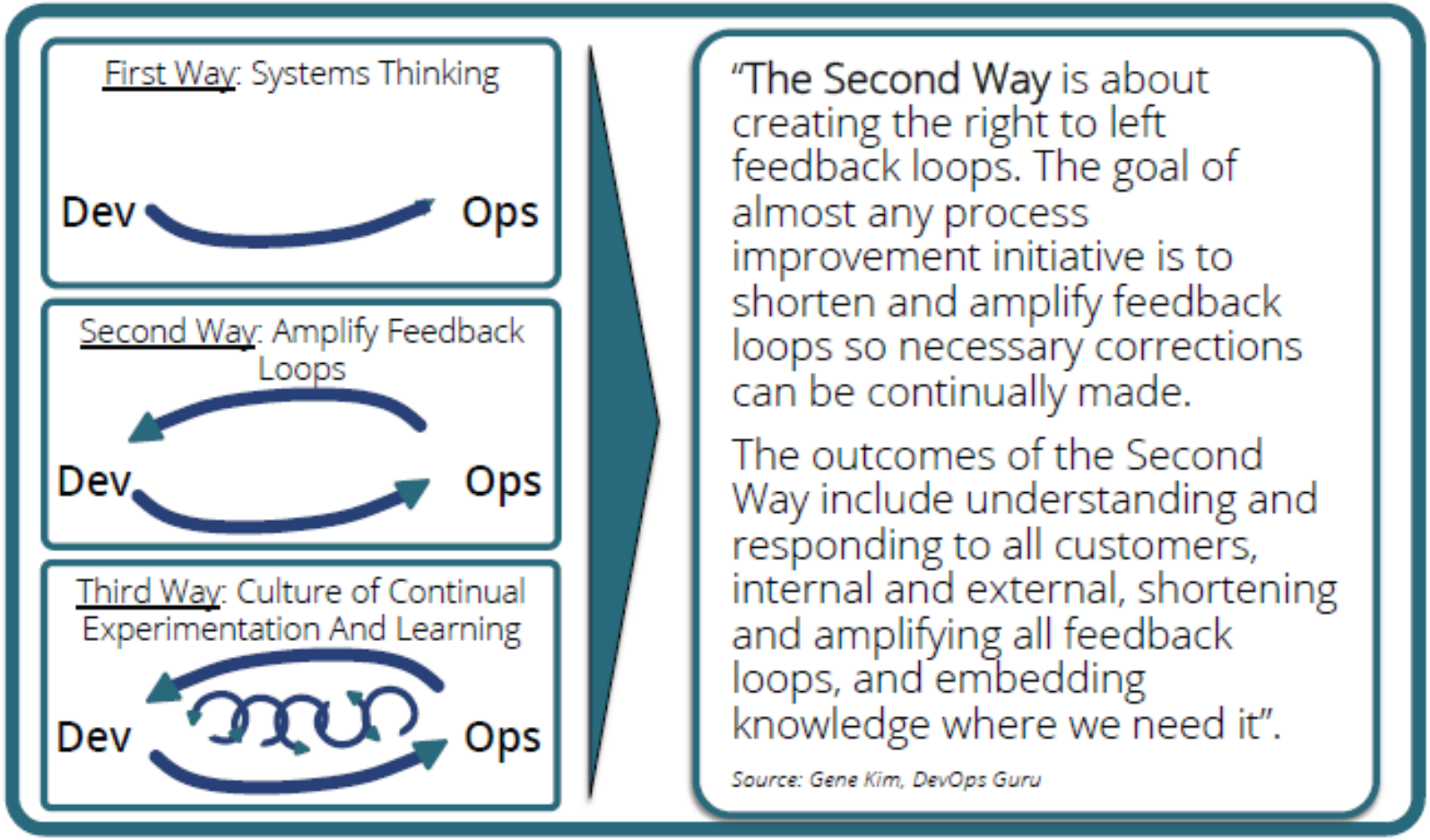
Which issues do we solve?

- Speed of delivery
- Quality of delivery
- Lack of understanding between developers and operations people
- Ability to ensure that customer requests are seamlessly processed
- Bring people who work on the same IT service closer together
- Preventing burnouts

If you can't measure, you can't improve!

A successful DevOps implementation will measure everything it can as often as it can... performance metrics, process metrics, and even people metrics.

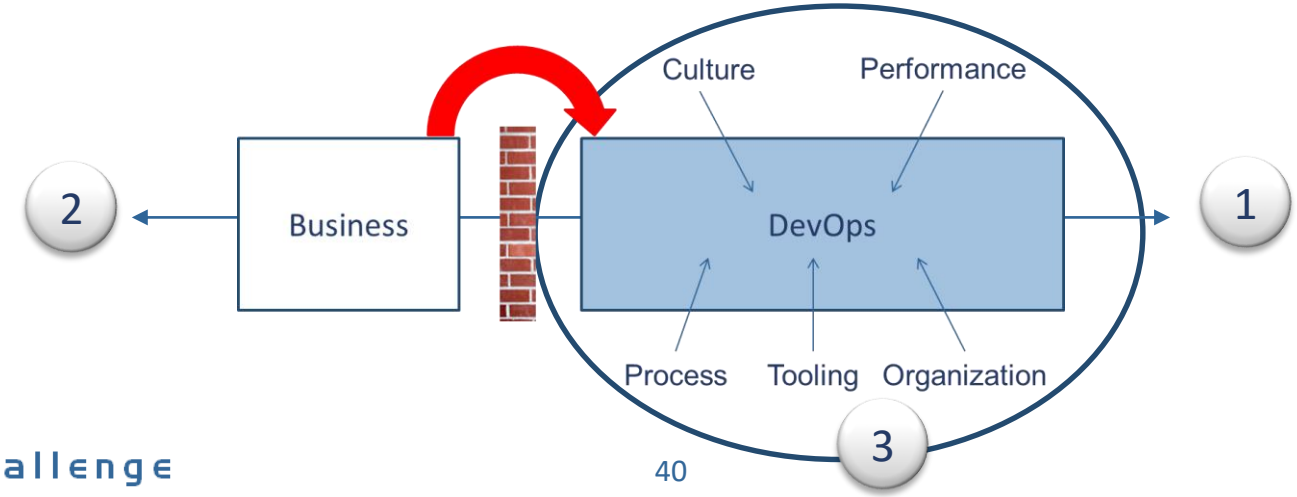
Importance of Feedback: Three Ways Model



Source: http://itrevolution.com/the-three-ways-principles-underpinning-devops/

The Three Ways

- The First Way – Flow
  - Understand and increase the flow of work (left to right)
- The Second Way – Feedback
  - Create short feedback loops that enable continuous improvement (right to left)
- The Third Way – Continuous experimentation and learning
  - Create a culture that fosters
    - Experimentation, taking risks and learning from failure
    - Understanding that repetition and practice is the prerequisite to mastery



The promise of DevOps

Quint
Wellington Redwood

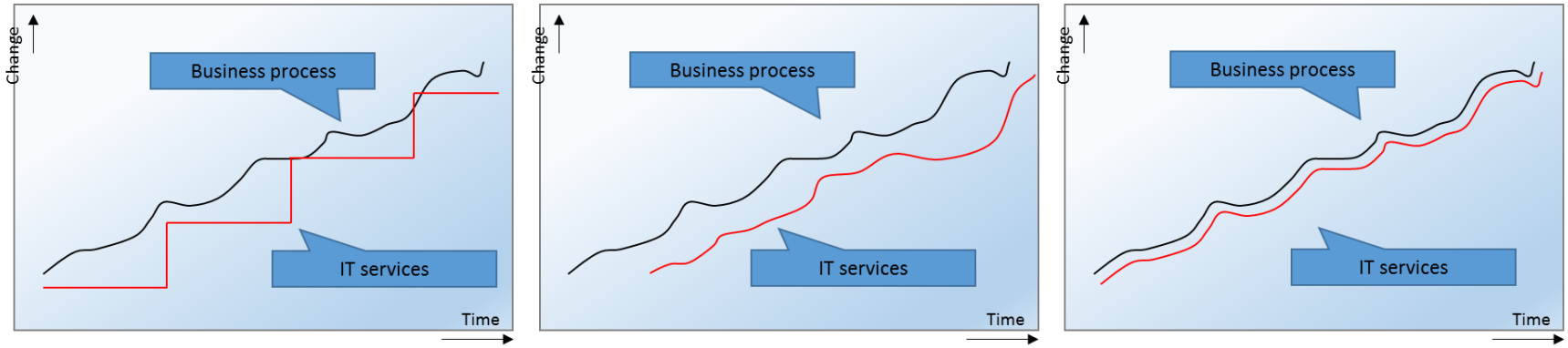
DevOps increases Agility and stability

- Organizations that implement DevOps are more likely to be high performing
- Organisations are more agile
  - Code is shipped 30 times faster
  - Deployments are completed 8000 times faster
- Services are more reliable
  - There are 50% fewer failures
  - Service is restored 12 times faster

(source: 2013 State of DevOps Report – Puppet labs and IT revolution press)

Same cadence between business and IT

Business processes change over time in a evolutionary way.
So the IT services that enable those processes need to change as well



Classic IT
(big gaps and big steps)

First Agile effect
(small gap, but lagging behind)

DevOps promise
(small gap, no lag)

Who is best

"Amazon has become famous for its high-velocity deployment of more than 2500 software releases per day across its various Cloud solution offerings. And there are similar storied successes at eBay, Etsy, Facebook, Netflix, Spotify, Twitter."
http://blog.saugatucktechnology.com/

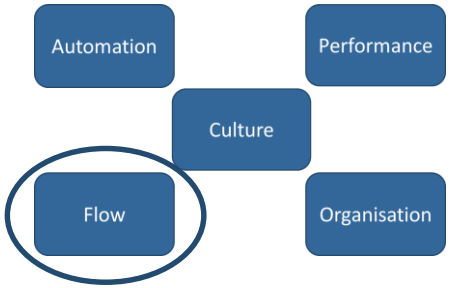
Flow

2 The Flow Solution

The problem is because Dev and Ops use different processes

*"The important theme of DevOps is that **the entire development-to-operations lifecycle must be viewed as one end-to-end process**. Individual methodologies can be followed for individual segments of that processes (such as Agile on one end and Visible Ops on the other), so long as those processes can be plugged together to form a unified process (and, in turn, be managed from that unified point-of-view). Much like the question of measurement and incentives, each organization will have slightly different requirements for achieving that unified process."*

What is DevOps? Dev2ops.org



Key considerations

- Development uses Agile, Scrum, TDD
- Operations uses ITIL
- Processes are seen to be different, even incompatible
- Nobody takes an end-to-end view

3 Key aspects per dimension: Flow

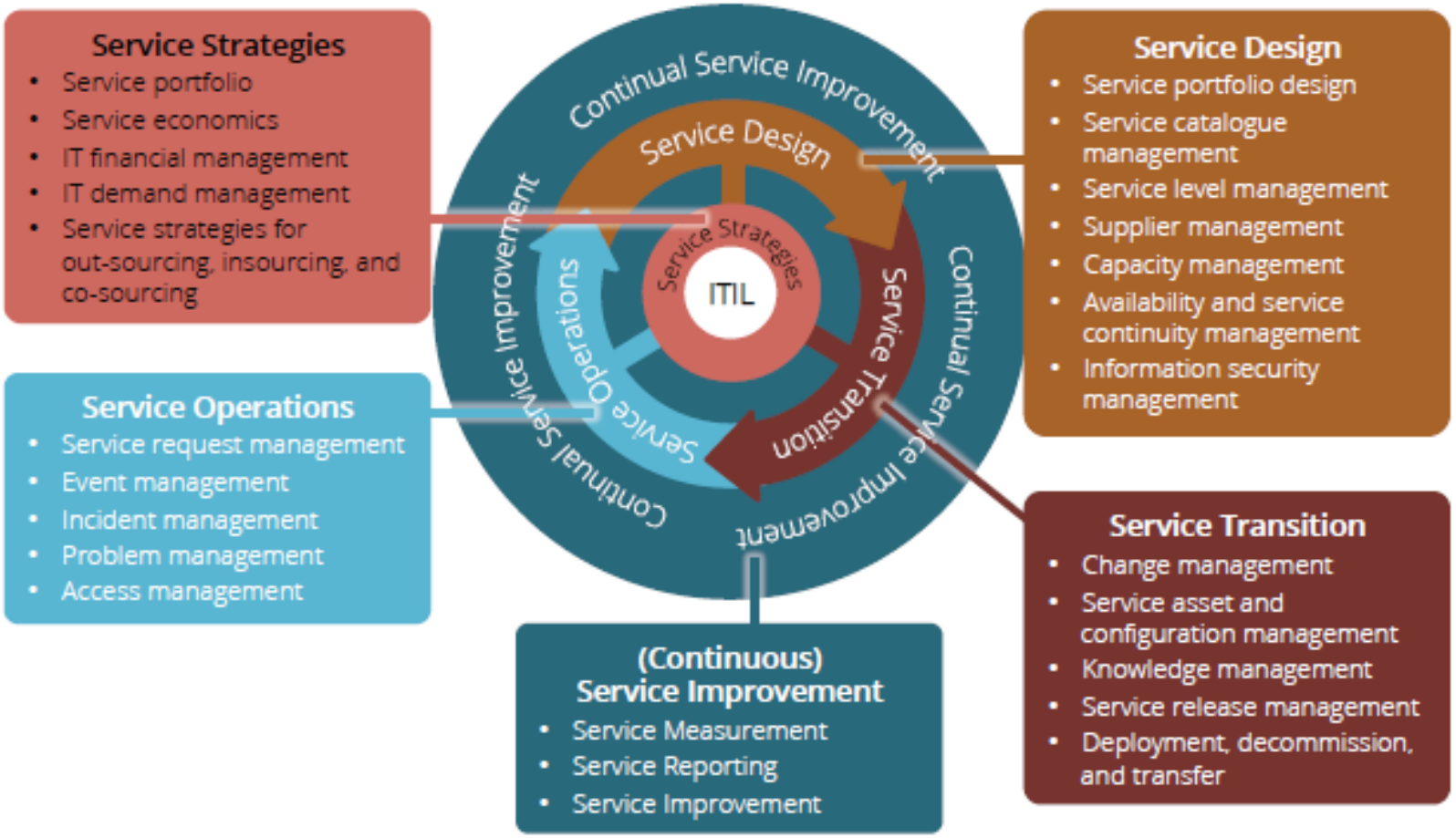
Getting units of work to Flow through the team to deliver value

Flow

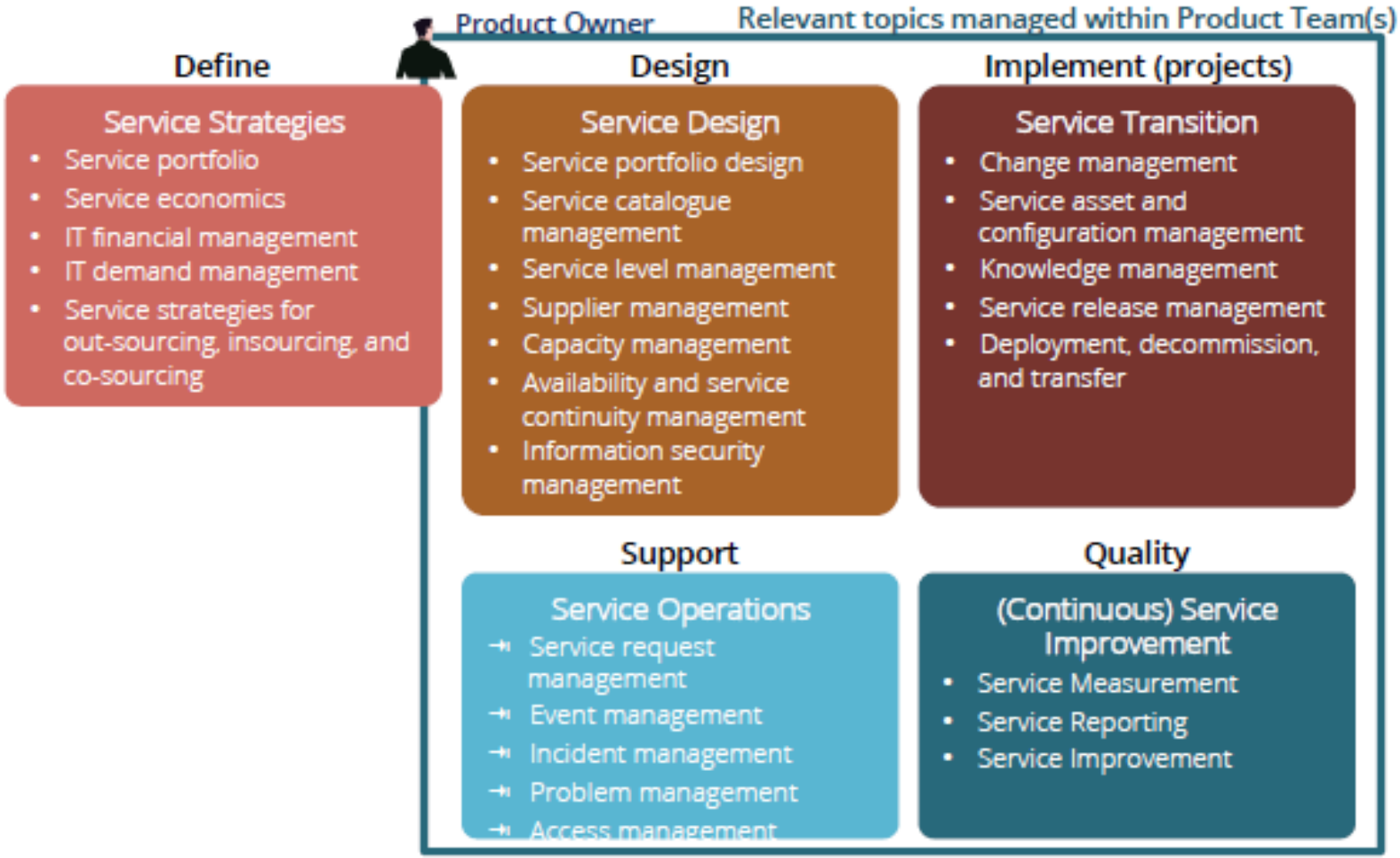
Flow is about ensuring that units of work move swiftly through the process

	What is it	Why is it important
Knowledge of units of work	The things the team spends time on	improve its performance on each unit of work
Awareness of processes	How each unit of work needs to be processed	Developing successful habits supports team performance
Documentation of processes	How extensively processes are documented	Helps to standardize the way of working
Monitoring of progress	Understanding the status of each unit of work	Encourages flow efficiency
Customer orientation	Each process starts and ends with a customer, and delivers value	Focuses team on ensuring that processes deliver value

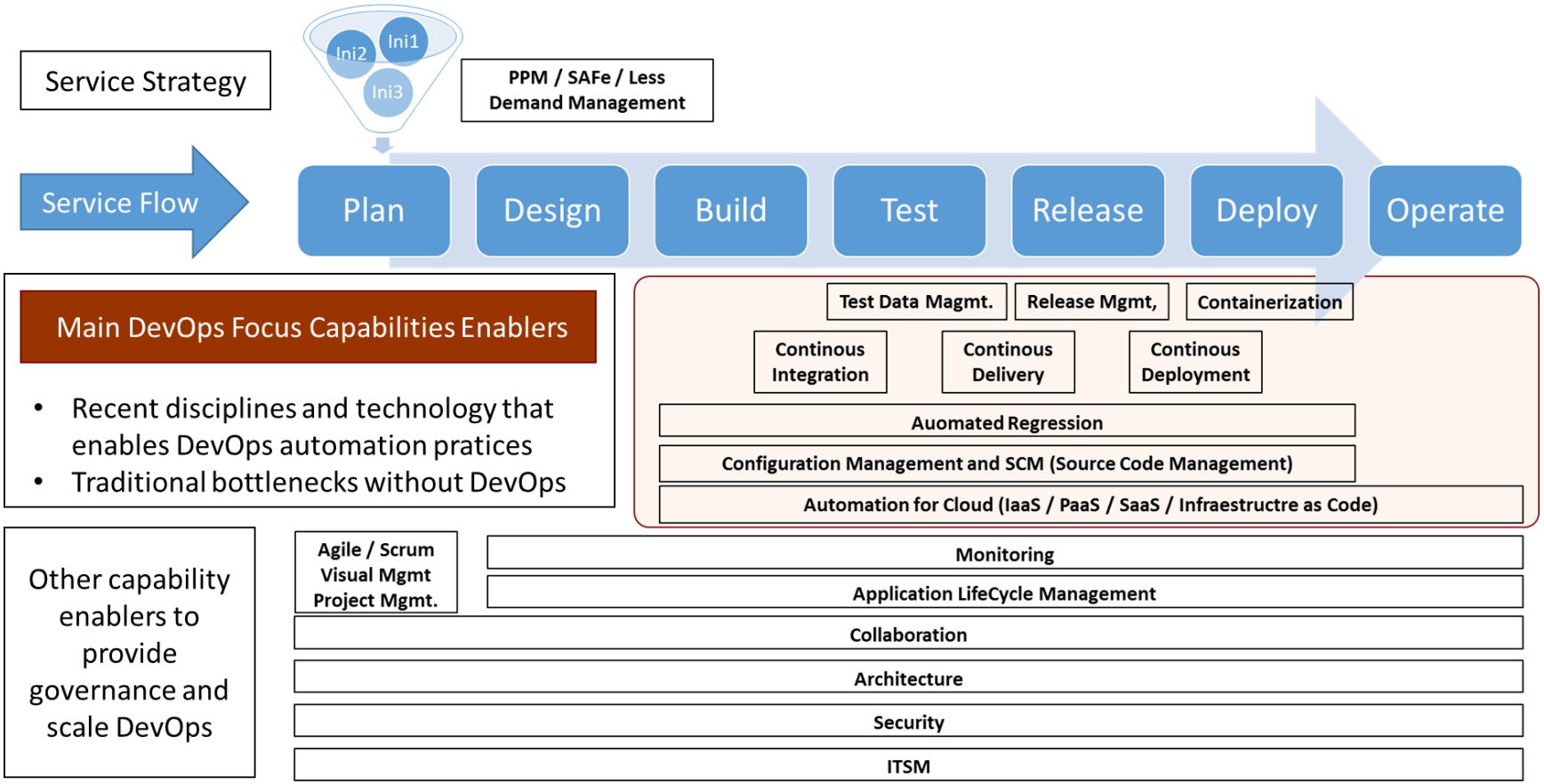
DevOps and ITSM



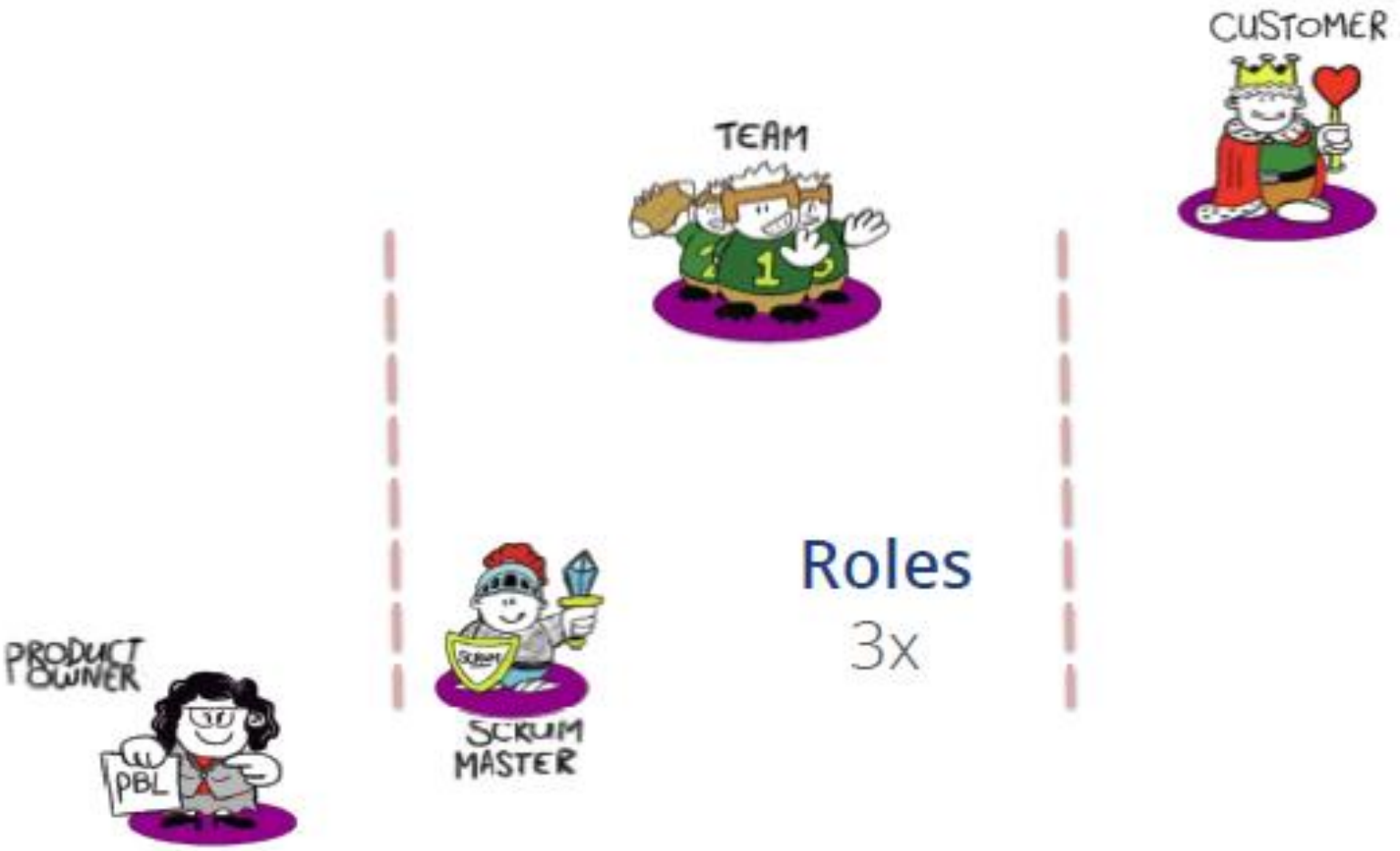
DevOps and ITSM (Contd.)



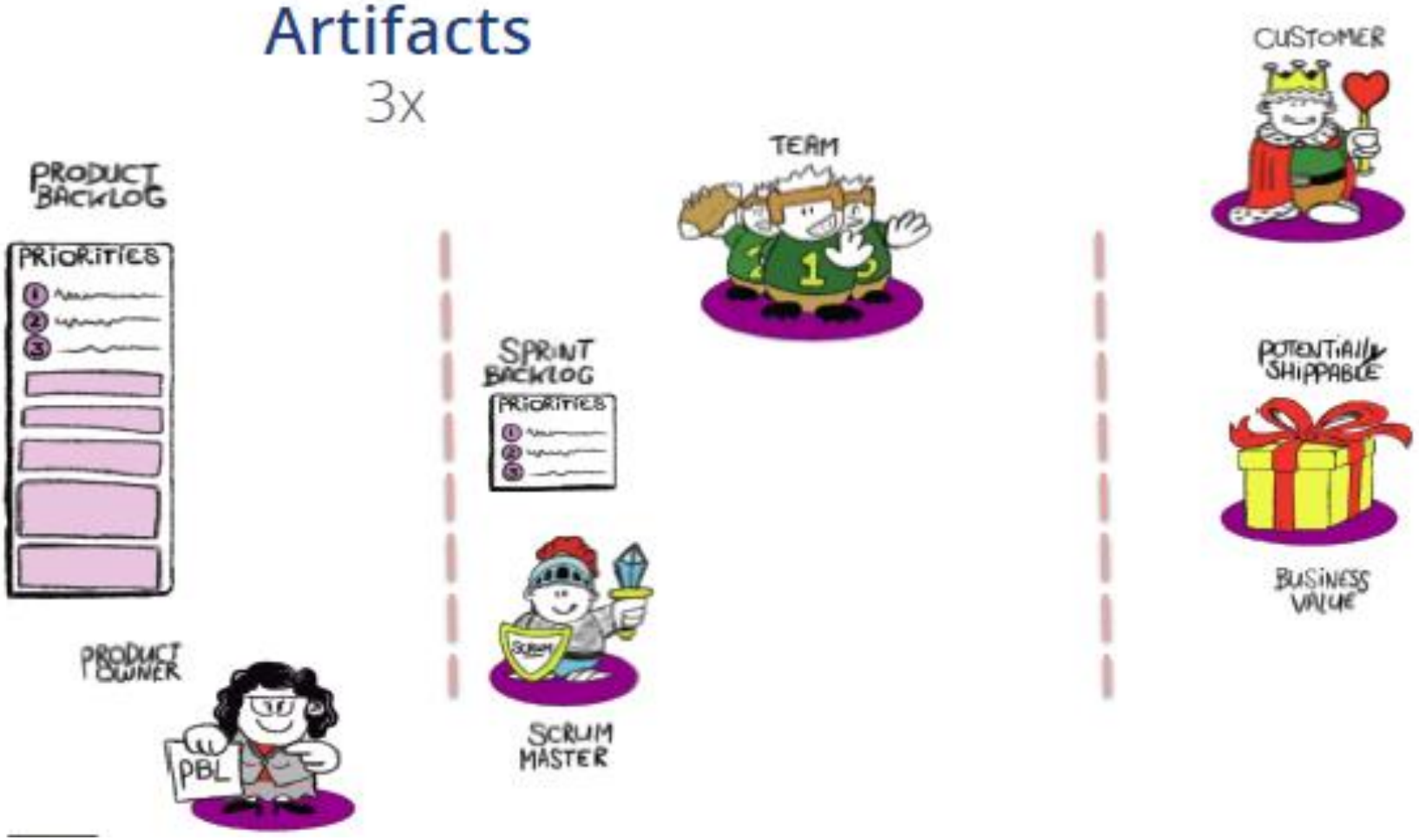
Flow for Service Creation and Operation



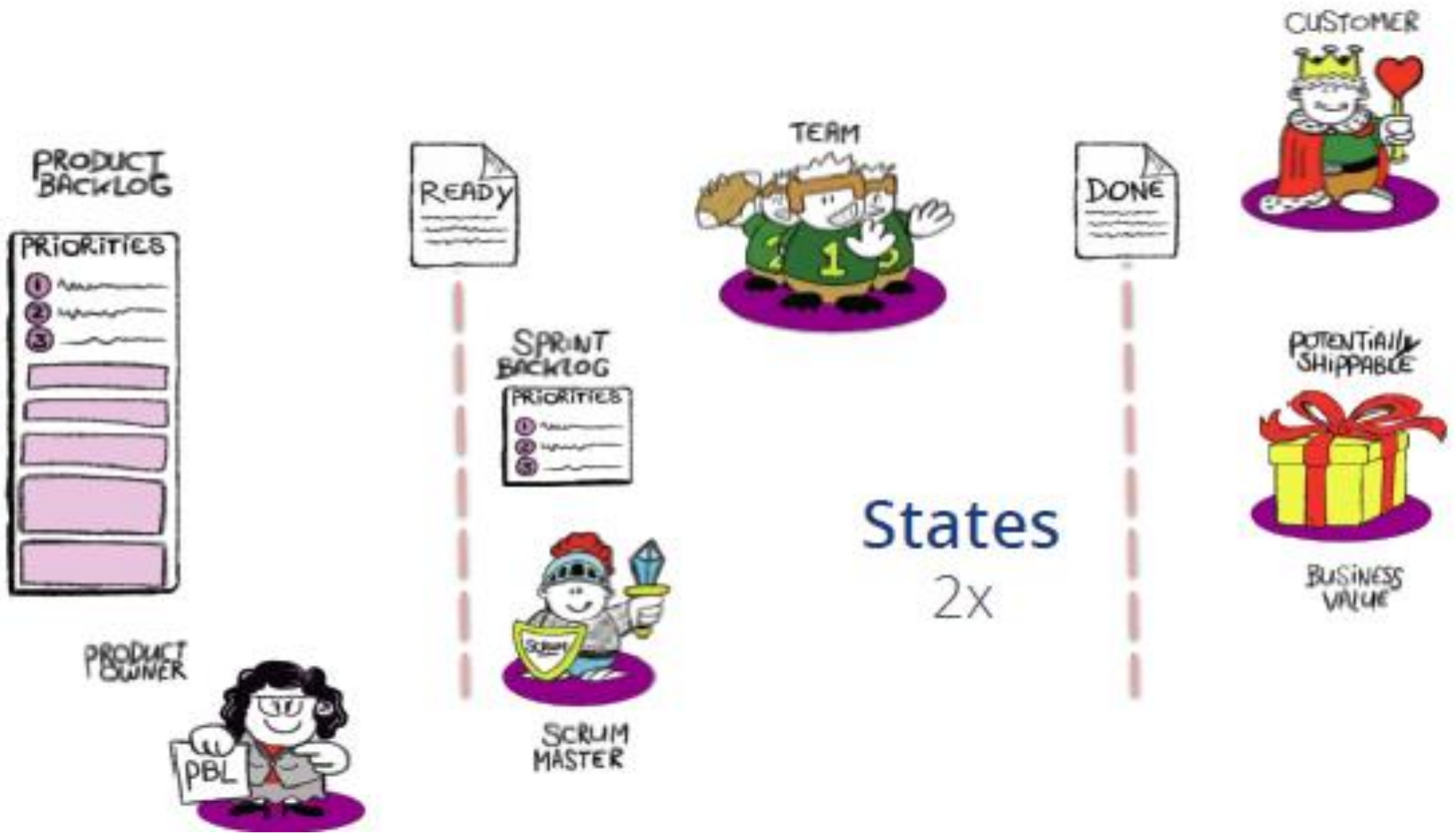
Scrum Approach



Scrum Approach (Contd.)

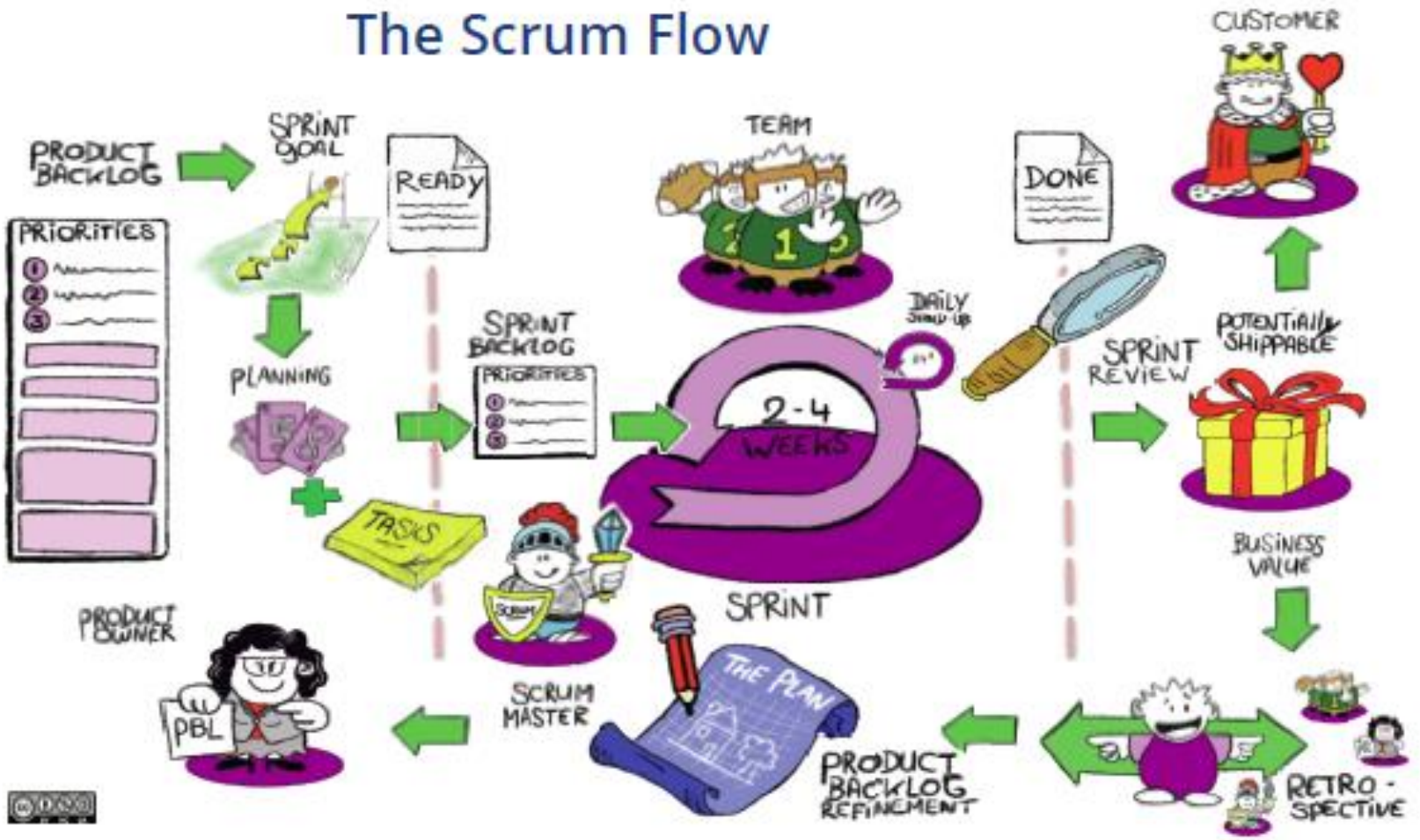


Scrum Approach (Contd.)

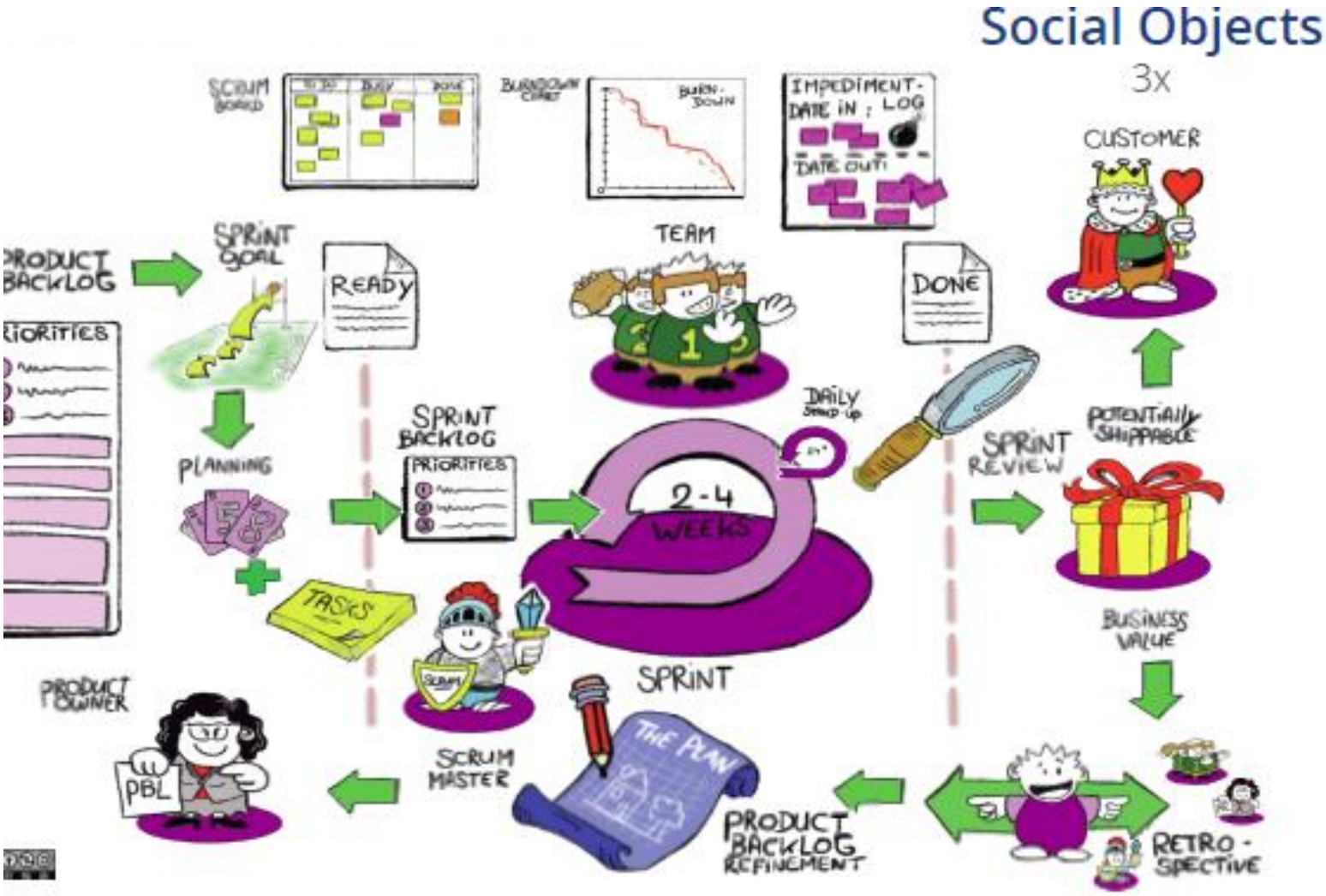


Scrum Approach (Contd.)

Scrum Approach (Contd.)

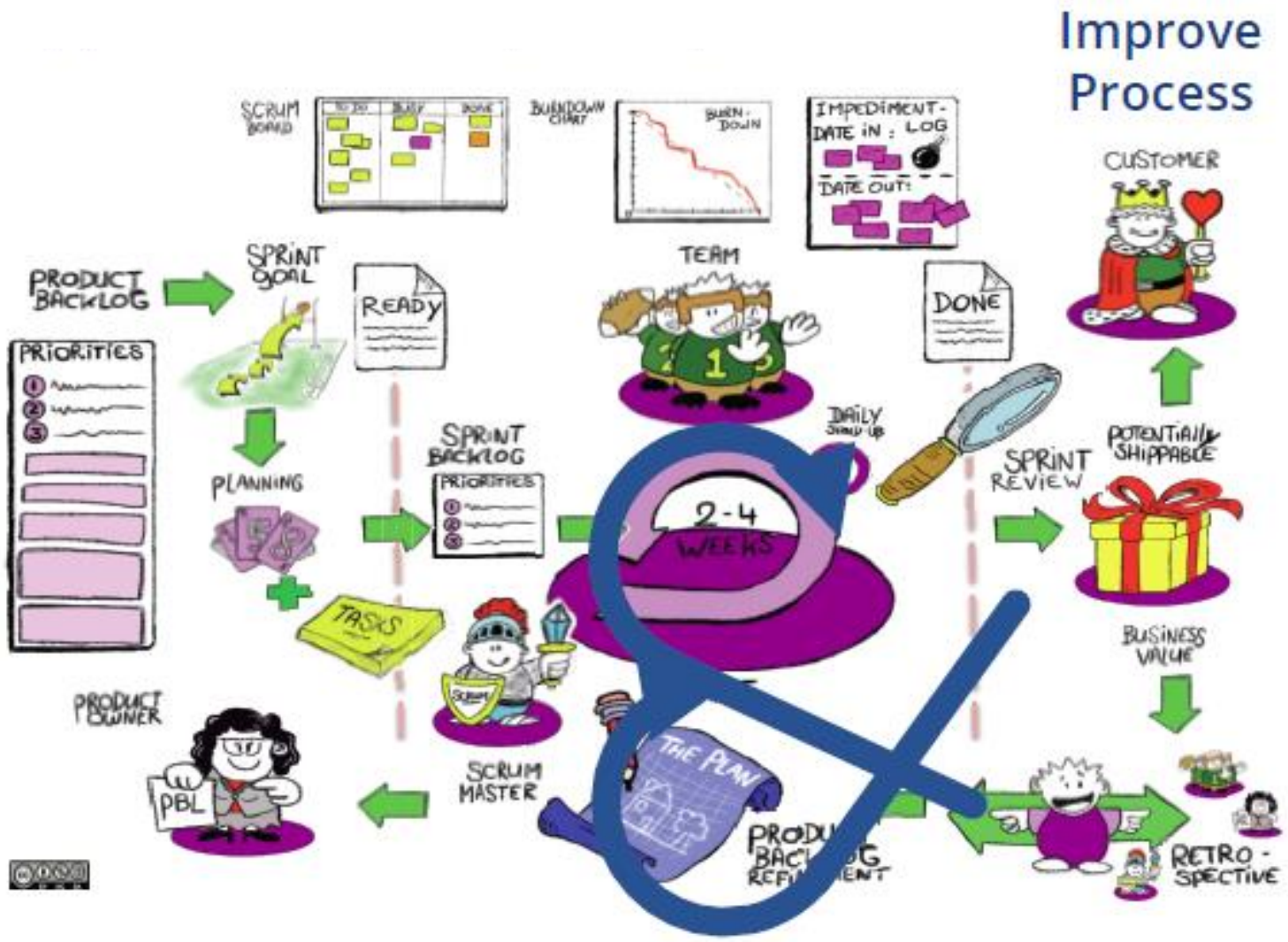


Scrum Approach (Contd.)

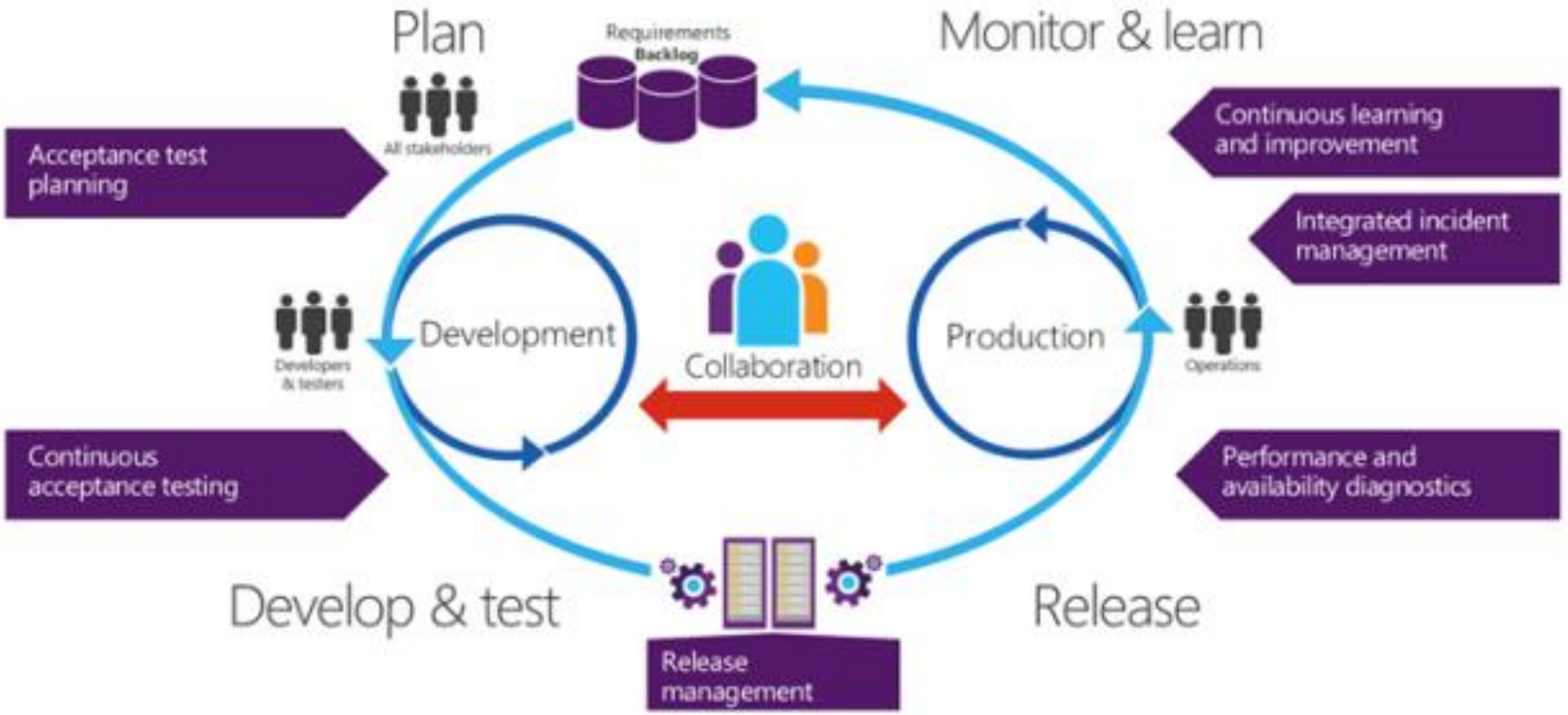


Scrum Approach (Contd.)

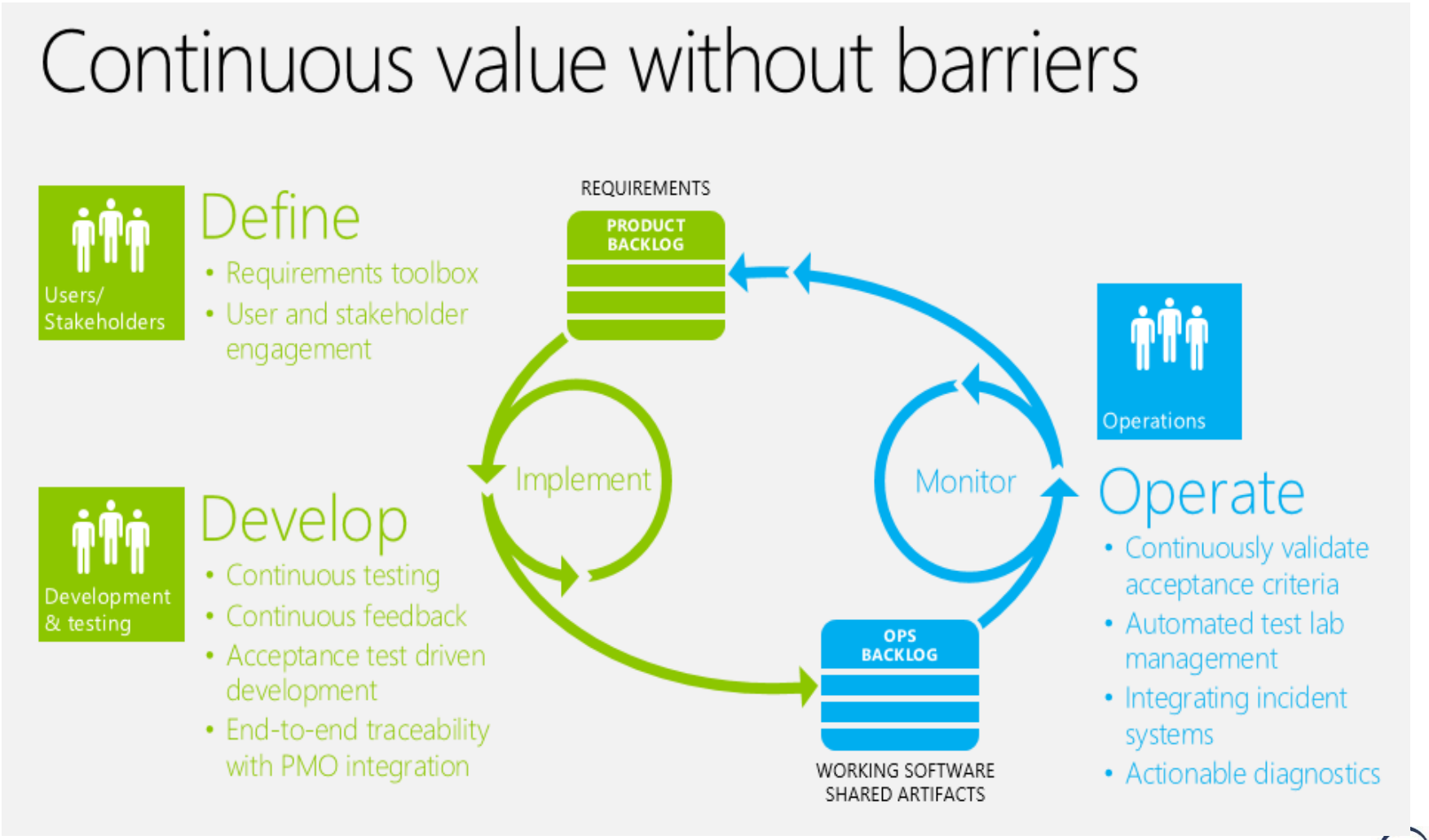
Scrum Approach (Contd.)



Continues flow

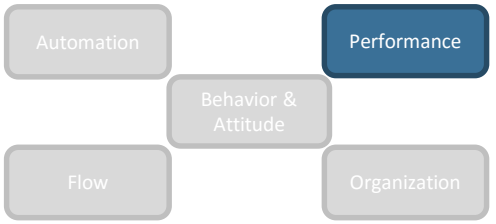


Continues flow



Performance

Performance



The problem is because Dev and Ops measure performance in a different way

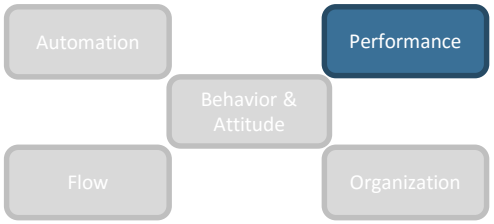
"*DevOps promises* ***more releases in a short timeframe in order to adapt to problems or changes in the market faster****. […] It also means that the metrics defined by each team can be automatically measured and automatically communicated and shared with everybody that relies on them to make decisions.*"

Best Practices for Performance-focused DevOps, ampblog.compuware.com

Key considerations

- Change vs. Stability
- Fast change vs. slow change
- Different KPIs used in development and operations
- Dev and Ops have different goals

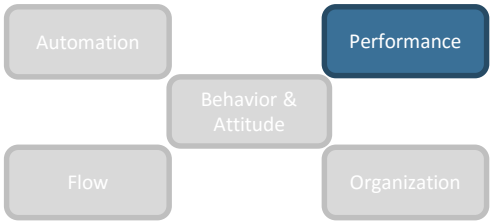
Key aspects per dimension



Performance is all about understanding how well the team is doing. Delivering value for the customer; performance must be sustainable.

	What is it	Why is it important
Key Performance Indicators	Metrics for understanding performance of the team	Show the team how it is doing. Identify improvements
Technical Debt reduction	Health of the service	Directs solutions towards architectural choices
Level of performance (Inc, SR, St. Ch.)	Understand performance on small units of work	Basic hygiene factor for customers.
Level of performance (creative work)	Understanding of performance on larger units of work or specials	Steer greatest business value
Time usage (Earning/Burning)	Understanding of how time is used within the team	Steering the usage of time to optimize the time available for creative work

Measuring Success



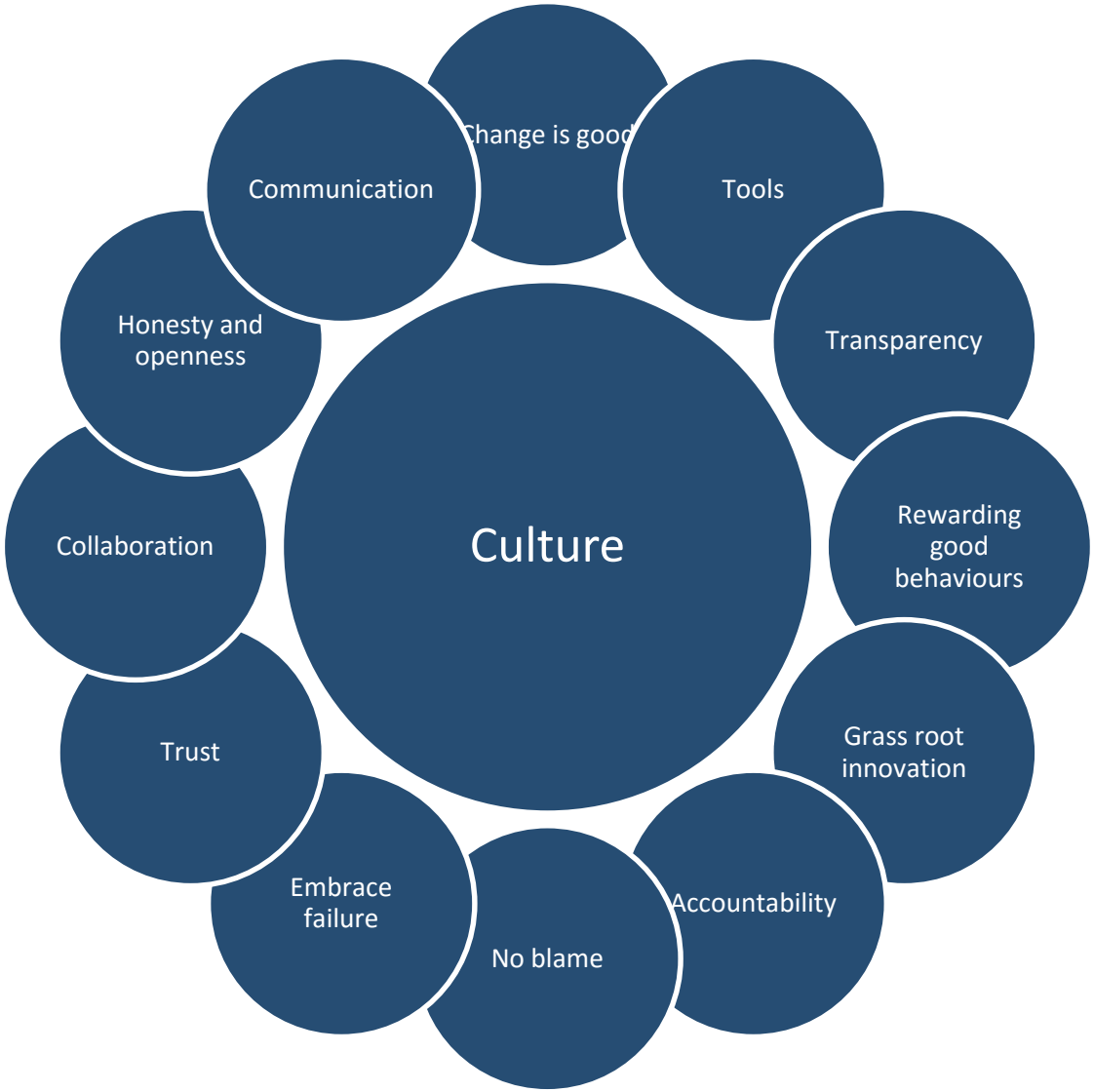
Practice that enable organizations to understand and improve flow enable improved IT performance

- IT performance is measured in terms of throughput and stability
- Throughput is measured by deployment frequency and lead time for changes
- Stability is measured by mean time to recover and the ability to preemptively detect and mitigate problems

  - Deployment frequency – mean time between deploys
  - Cycle time – Time from start of work to ready for delivery
  - Change Failure rate
  - Mean time to detect incidents (MTTD)
  - Mean time to recover (MTTR) – Component
  - Mean time to restore (MTRS) - Service

Behavior & Attitude

Culture is all about



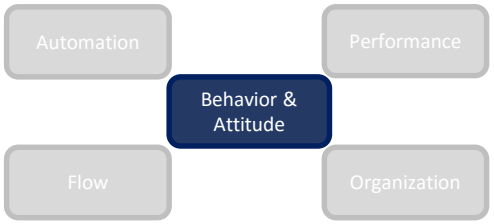
The culture in Dev and Ops

The problem is a mismatch of cultures between Dev and Ops

*"DevOps is simply a **culture** that brings development and operations teams together so that through understanding each others' perspectives and concerns, they can build and deliver resilient software products that are production ready, in a timely manner. DevOps is not NoOps. Nor is it akin to putting a Dev in Ops clothing. DevOps is synergistic, rather than cannibalistic."*
Fresh Stats Comparing Traditional IT and DevOps Oriented Productivity, DevOps.com

Key considerations

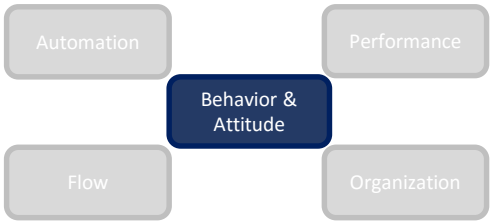
- Change vs. Stability
- Fast change vs. slow change
- Development people are different from Operations people



Key aspects per dimension

A team of skilled professionals with the right attitude, behavior and motivation to deliver excellent performance

	What is it	Why is it important
Team-forming	Understanding of where the team is in its evolution	Helping a team to 'gel' to deliver high performance
Collaboration	An environment in which people are free to act together	This is the basis for continuous improvement and learning
Problem-solving	A method and mindset for resolving the issues	Ensure progress of and excellence in the service
Leadership Fit	Leadership in relation to the requirements of the team	Leader's style must be compatible with team needs
Motivation	Understanding level of motivation of the team	Only motivated people can produce a sustained excellent performance



Burn down the old silos…

Burn down the silos and create integrated cross functional teams

- Overcome the challenge of change versus stability
- Create shared team responsibility for all requirements,
- analysis, design, engineering, testing, integration, delivery
- and deployment
- both new development as well as maintenance and support

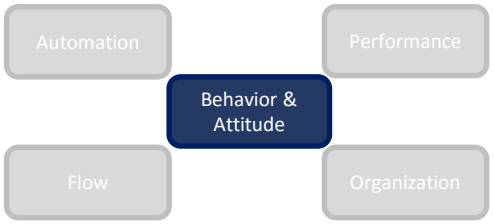
Results:

- shorter lead time!
- no more walls of confusion
- more trust and fun
- higher quality, lower risk

Stable, co-located teams with a common goal and enough mandate

- Product teams and Platform teams
- Team KPIs instead of personal KPIs
- Speak the same language
- Seek first to understand, then to be understood
- Tools (software; Scrum board)
- Retrospective and daily stand-up:
- honest and open feedback
- Training, knowledge sharing
- Team building events

… and create transparency

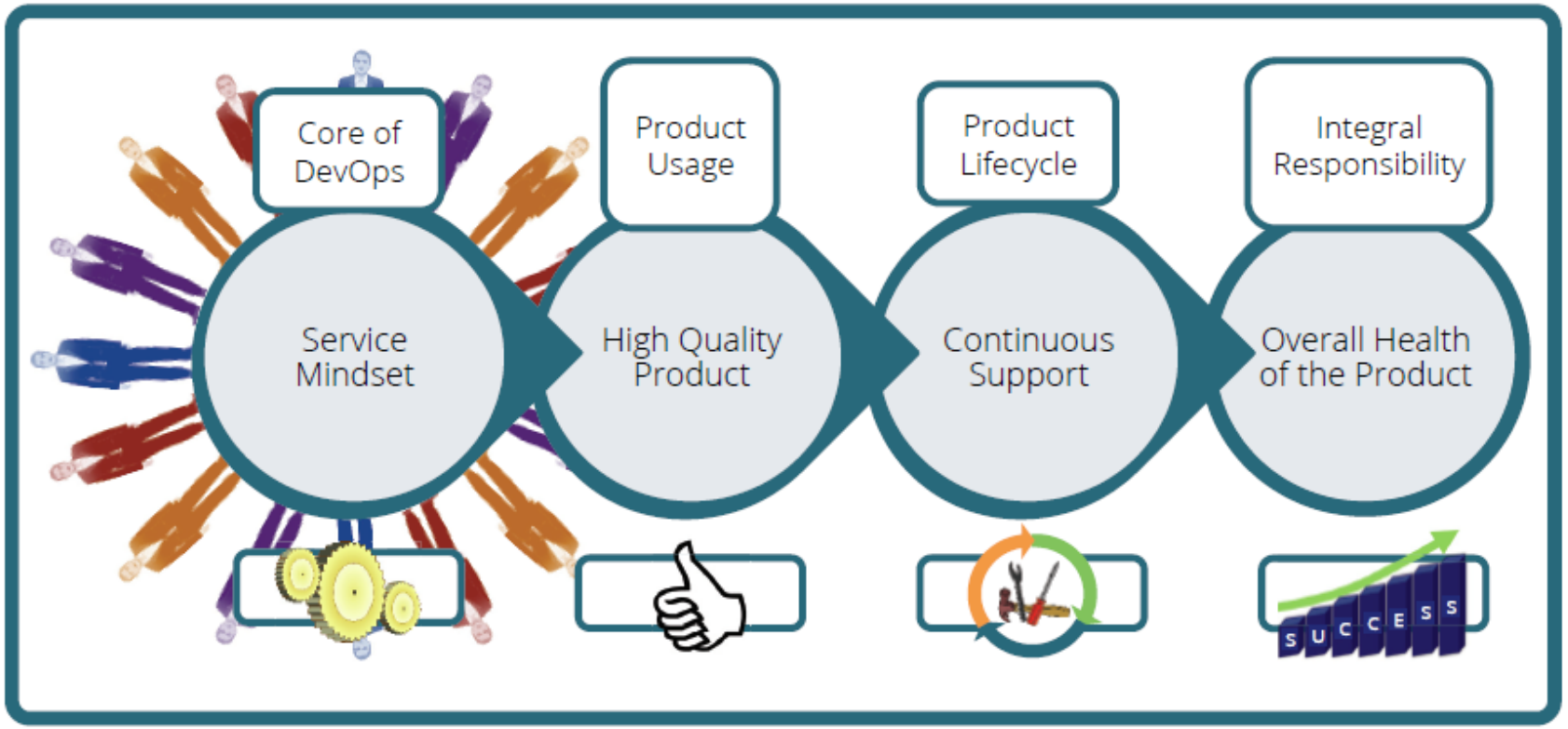


Transparency is about getting feedback

- from the customer / business
- from the team
- from the software / system in order to continually improve: inspect and adapt
- Transparency is about visualizing metrics

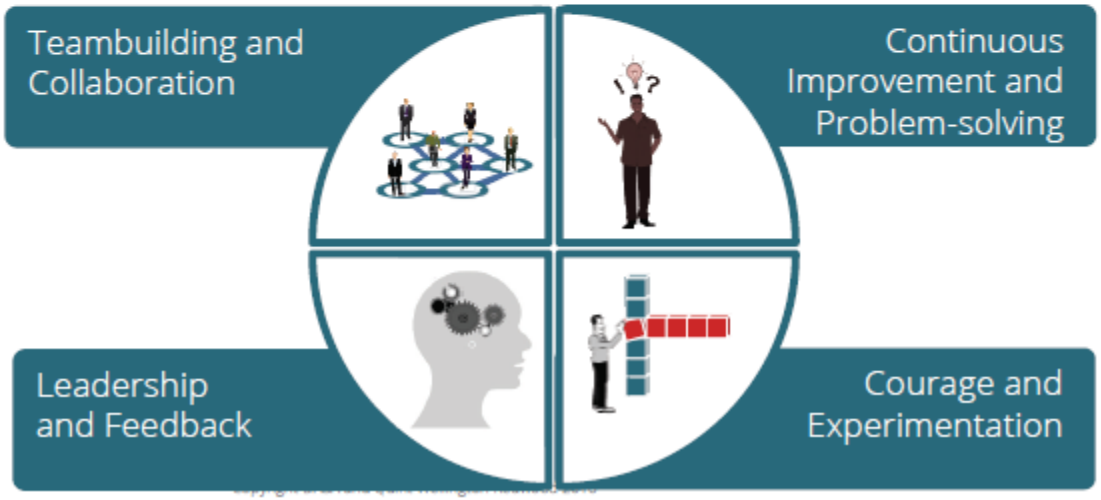
Core of DevOps Culture

- The core of the DevOps culture is the emphasis on service.
- The following figure shows how this mindset help to deliver a high quality product.



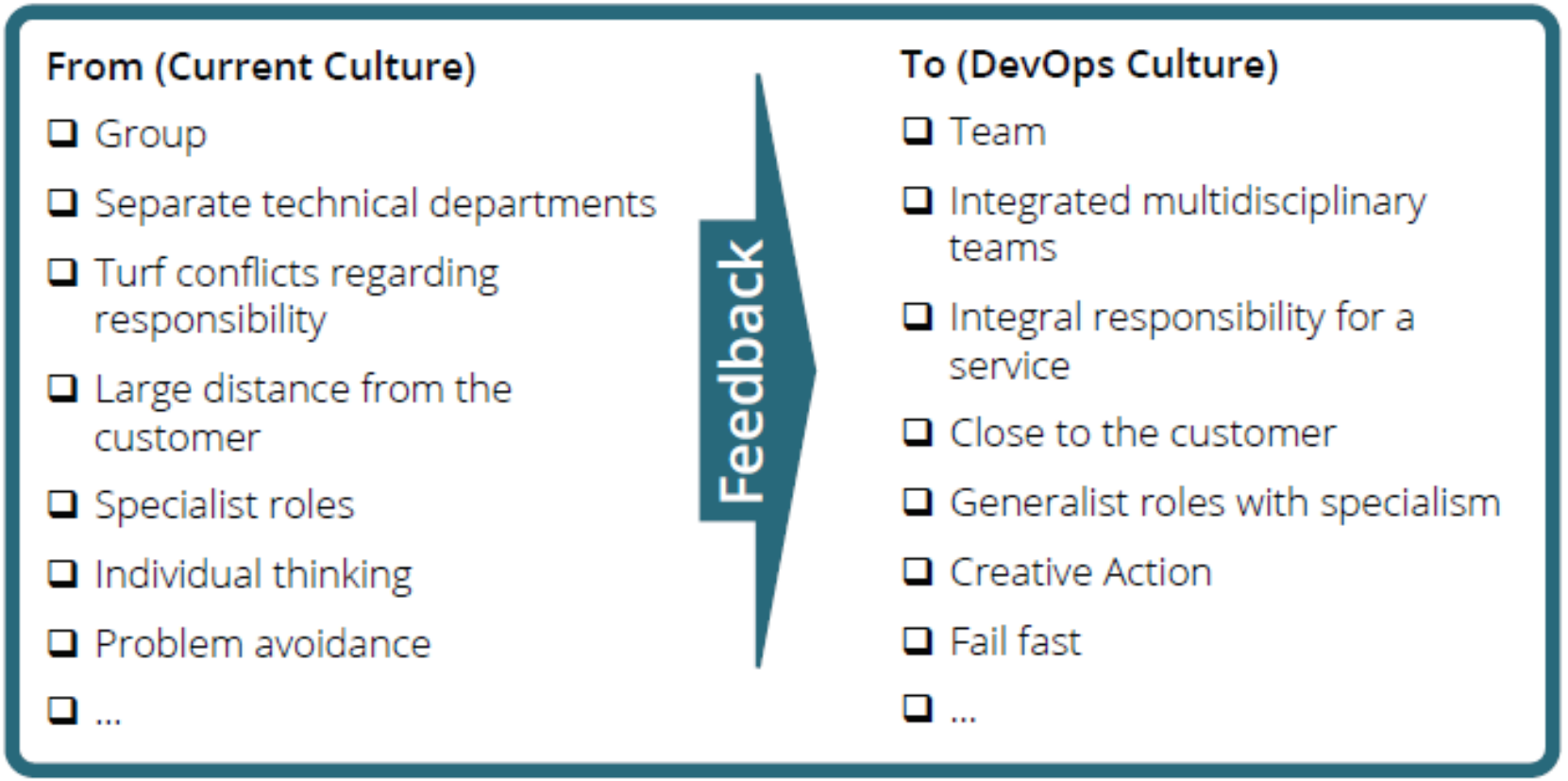
Cultural Elements of DevOps

- **DevOps is all about people, processes, and tools.**
  - People across the groups - working together - determine DevOps success.
  - The primary goal is to build a collaborative and respectful culture across the IT organization, to bring a single piece of flow in software delivery.
- **A team cannot become a DevOps team just by using a set of tools.**
  - They need to ensure that tools support DevOps requirements and workflows.
  - A culture that supports the ideals of DevOps is crucial.
  - Some cultural elements that can help develop an effective and successful DevOps culture are:



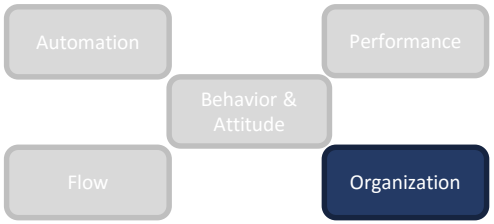
Changing the Culture

- **Changing the culture is a collective movement in which behavior, attitude, and mindset are adjusted through feedback. Using feedback, the team can move from its current status to the desired situation, as shown in the following figure.**



Organization

Cultural Shift "Connect" your Teams



The problem is because Dev and Ops have artificially been organizationally kept apart

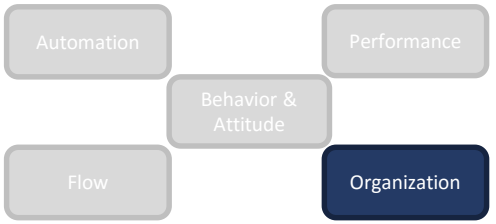
When we talk about "The Wall" between the Teams (Dev, Test, Ops, Business) – most industry professionals we have conversed with understand and recognize the concept. So – these walls exist – and we have to tear them down. ***The best way to start is by inviting Testing and Operations to your daily development Stand-Ups****. Let them listen in what you are currently working on. Invite them to your sprint reviews and planning meetings so that they see what is coming down the pipeline.*

Best Practices for Performance-focused DevOps, ampblog.compuware.com

Key considerations

- Traditionally, we have organized skills in functional siloes
- Development people sit next to development people, ops next to ops, etc.

Key aspects per dimension



Team responsibilities and accountability are, and that it is capable of fulfilling them.

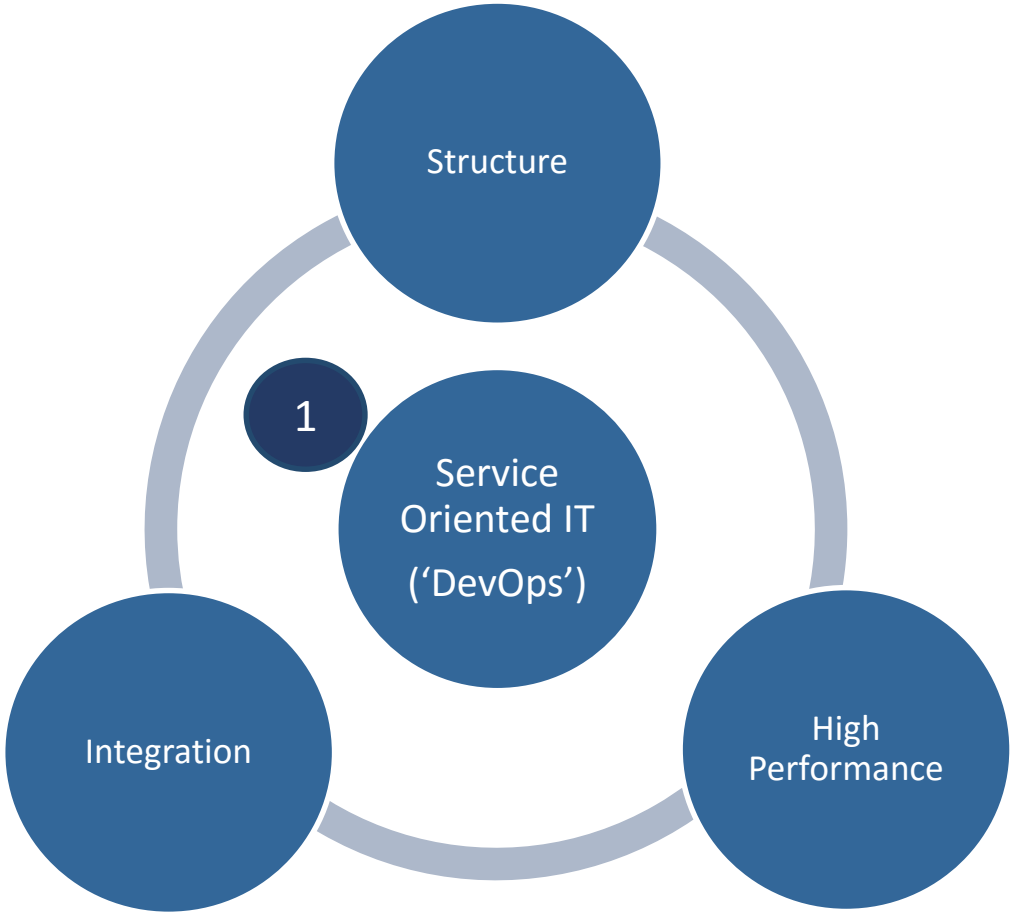
	What is it	Why is it important
Service Definition	Unambiguous definition of the service	Sense of purpose Responsibility and accountability
Customer definition	Clear definition of for whom the service is provided	Understanding of value of the service
Completeness of team	Skills & knowledge and roles to provide the service	Ensure that the team can take full responsibility for the service
Dependency on other teams	Understanding of shared resources and dependencies	Optimize freedom to act on behalf of the customer
Management practices	Visual management and other governance practices to facilitate the team	Facilitate team in delivering value

Make it easy for customers to do business with you

Automation | Performance | Behavior & Attitude | Flow | Organization

2

Structure
Organizing teams and the governance



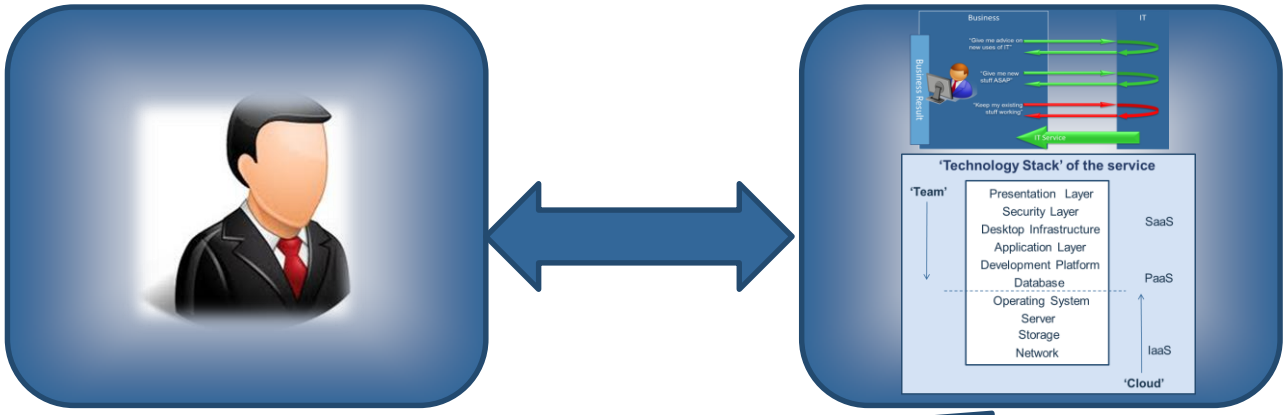
3

Integration
DevOps Integration Model to guide the integration of Dev and Ops teams

4

High Performance
High Performing Team model to guide both Management Team and DevOps teams

1 Define Service-oriented IT based on customer, service and required skills

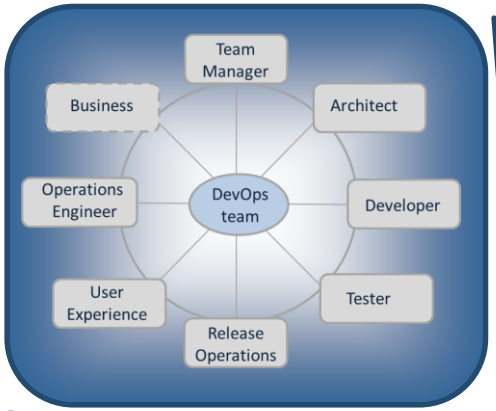


The customer

- Person responsible for the business process
- People using the service

IT Service(s)

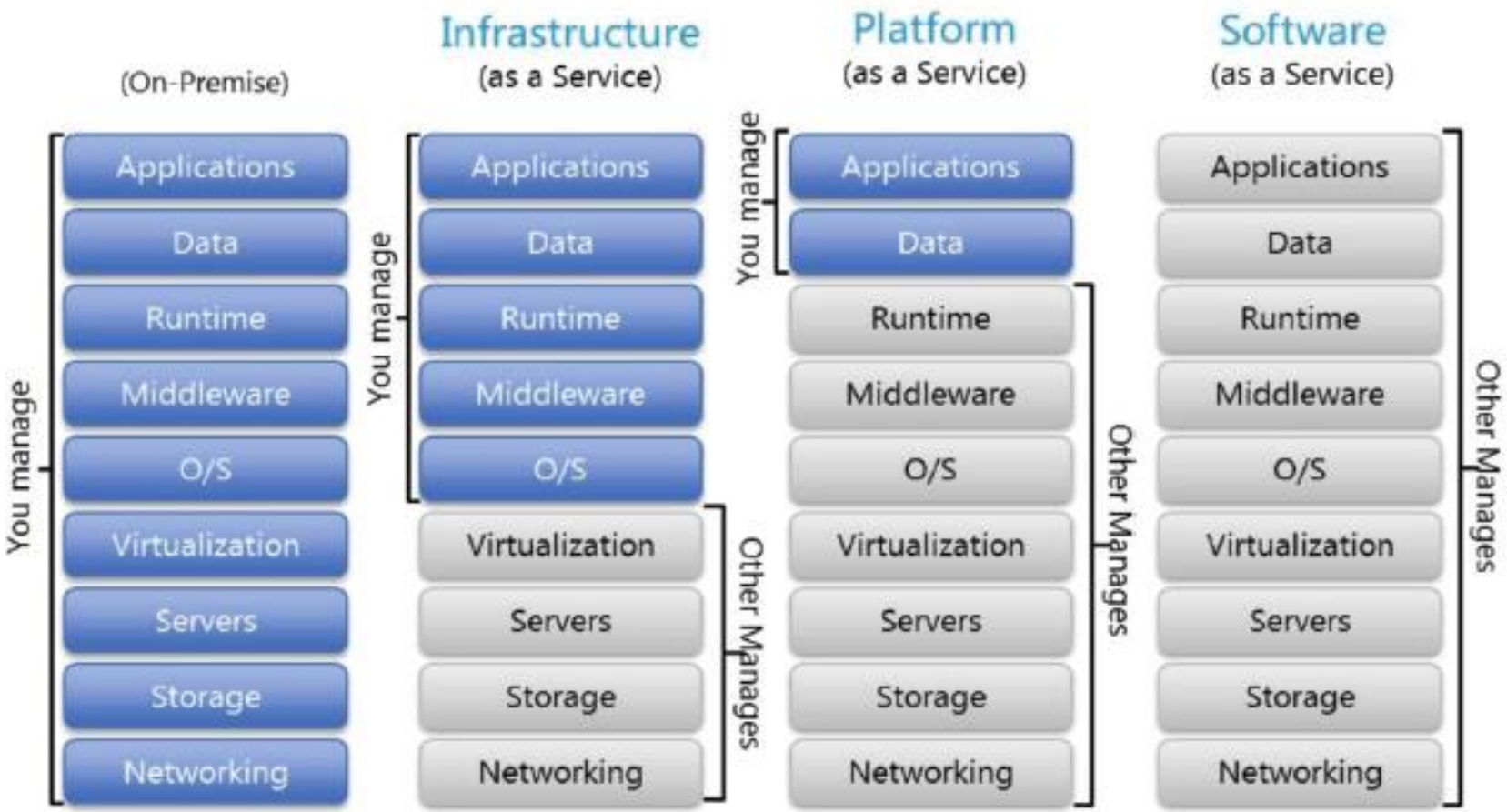
- All functionality that supports business process
- Deliver the value customers require



DevOps Team

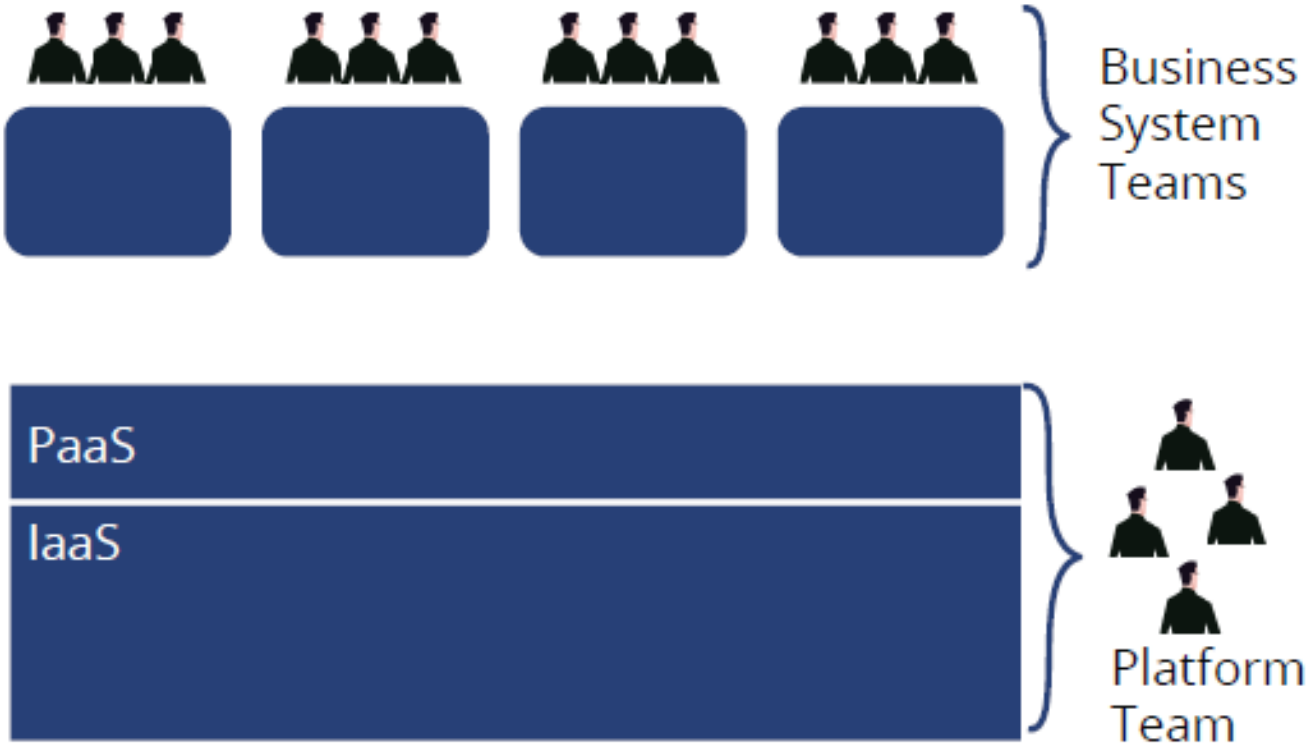
- Responsible for the 'health' of a service
- Contains all knowledge and skills required to support and develop the service

Cloud Service



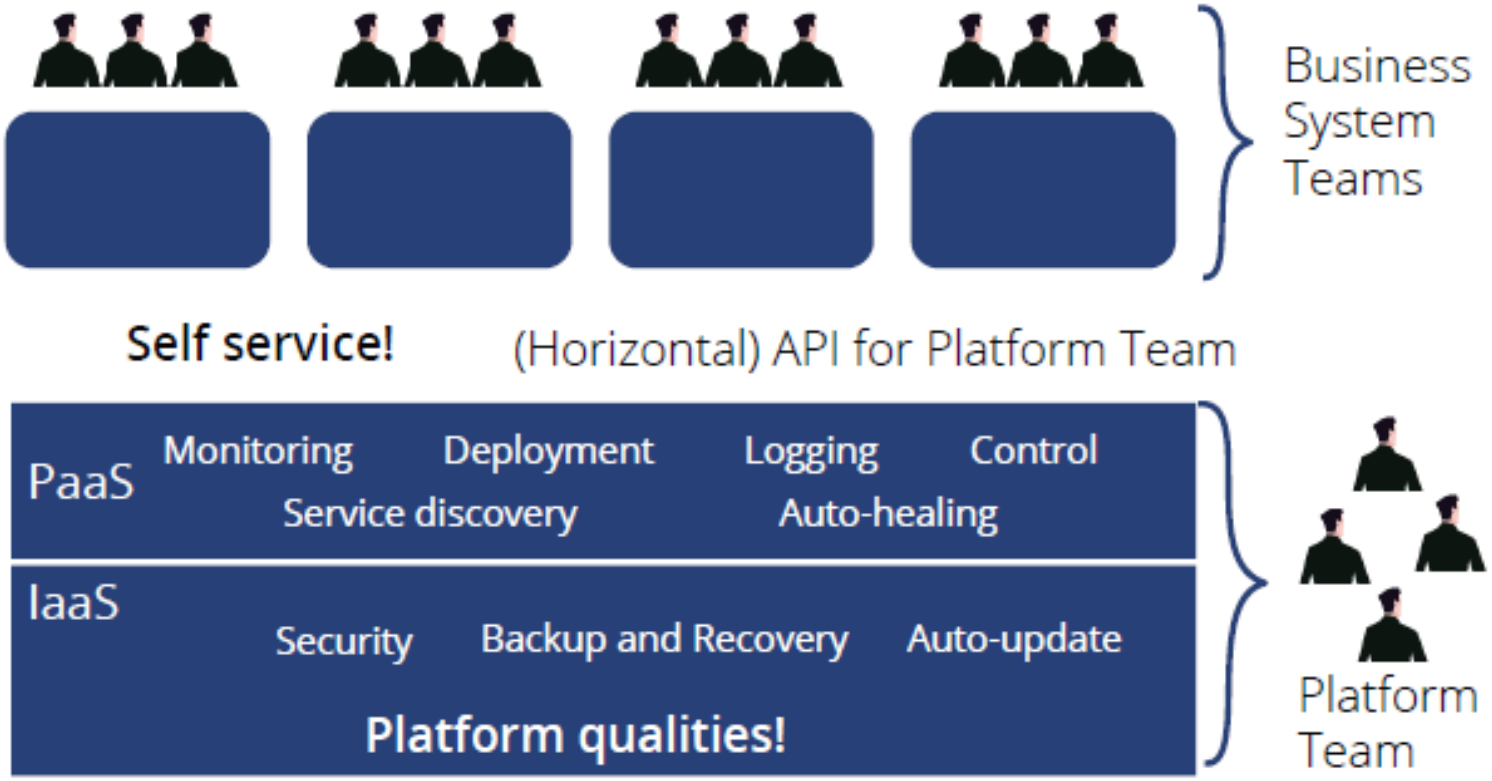
Source: http://download.microsoft.com/download/3/4/D/34D3CCF9-DF32-45BD-AD99-A004433D66FE/Microsoft%20Private%20Cloud%20Fast%20Track%20Reference%20Architecture%20Guide.pdf

Organizational Model: DevOps

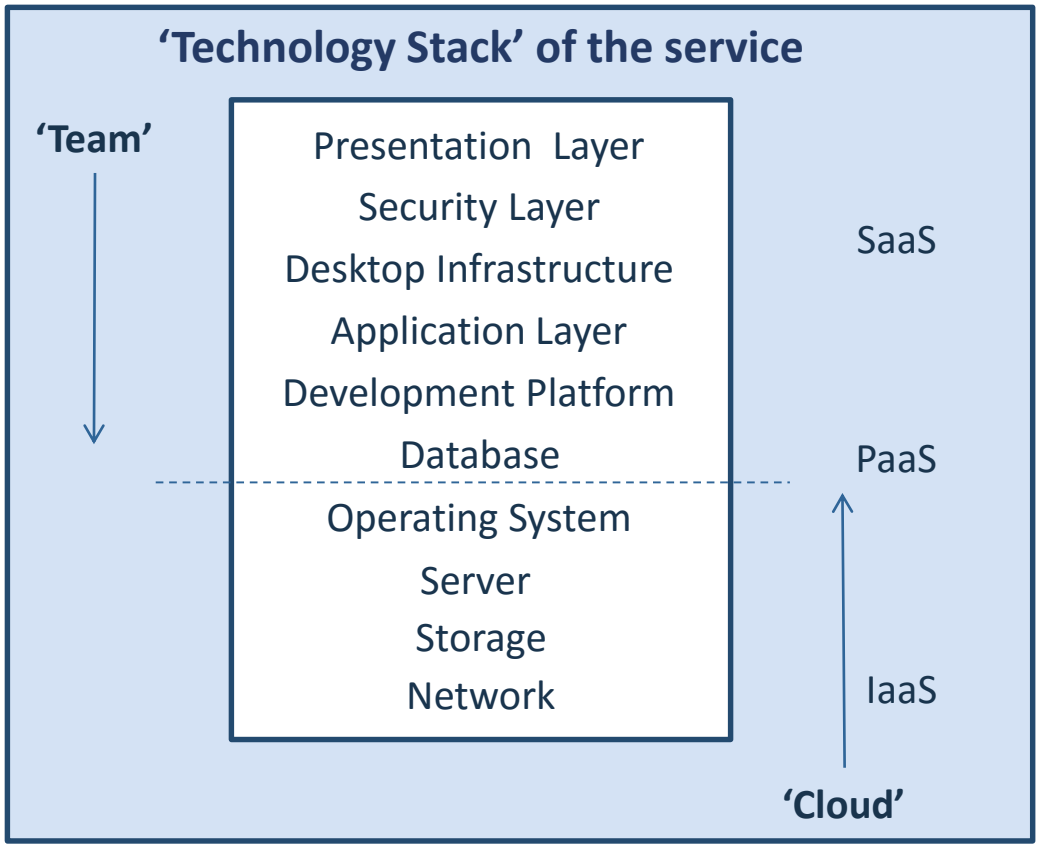


Business System Team, manage end user (customer), services and products (applications)
Platform Team, manage platform product used by Business System Teams

Autonomy of Teams



1 Determining the scope of a DevOps team



'Technology stack' determines:

- The area of responsibility of the team
- The required skill set of the team

A DevOps team is responsible for all facets of:

- The 'health' of a service
- The continued match of the service with the business process it supports

Focus on Products and Services

Features of product-focused approach:

- **Responsibility of product-team extended all the way into production.**
- **All are responsible and accountable for a fully working product.**

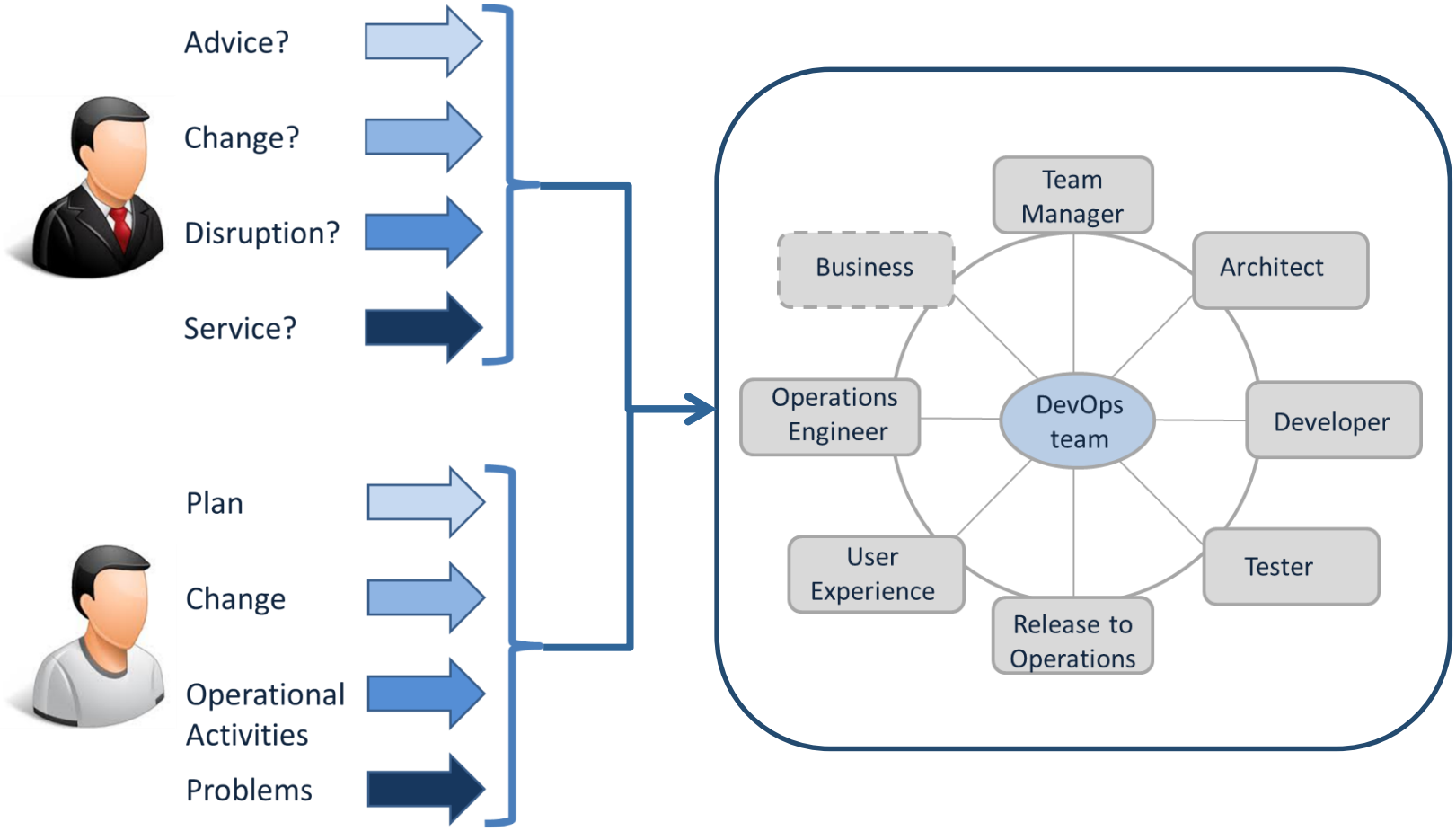
'Activity-Focused' (Siloed)

- Specialty Oriented
- Functionally Organized
- Project Focused
- Work with Individuals

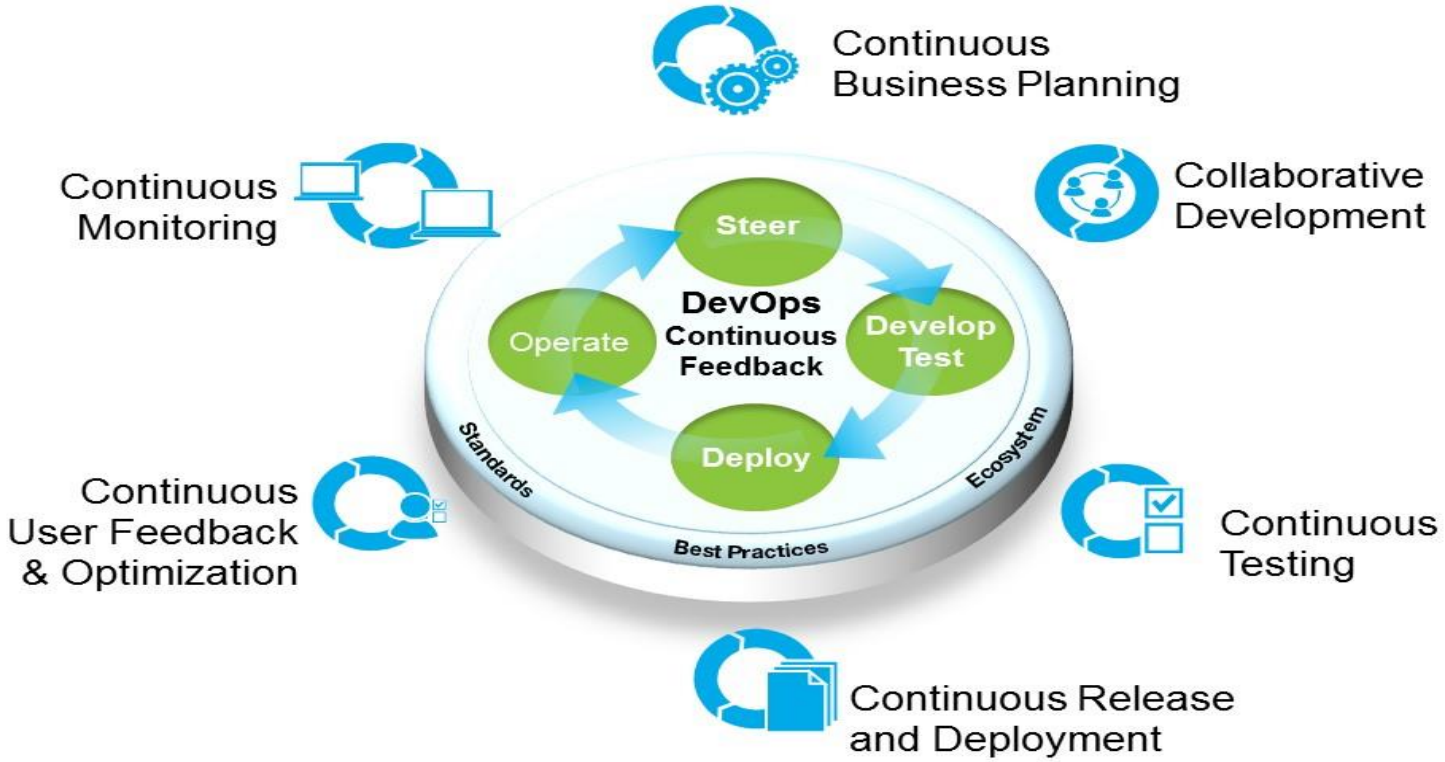
'Product-Focused' (Team)

- Work Oriented
- Team Organized
- Product Focused
- Work with Teams

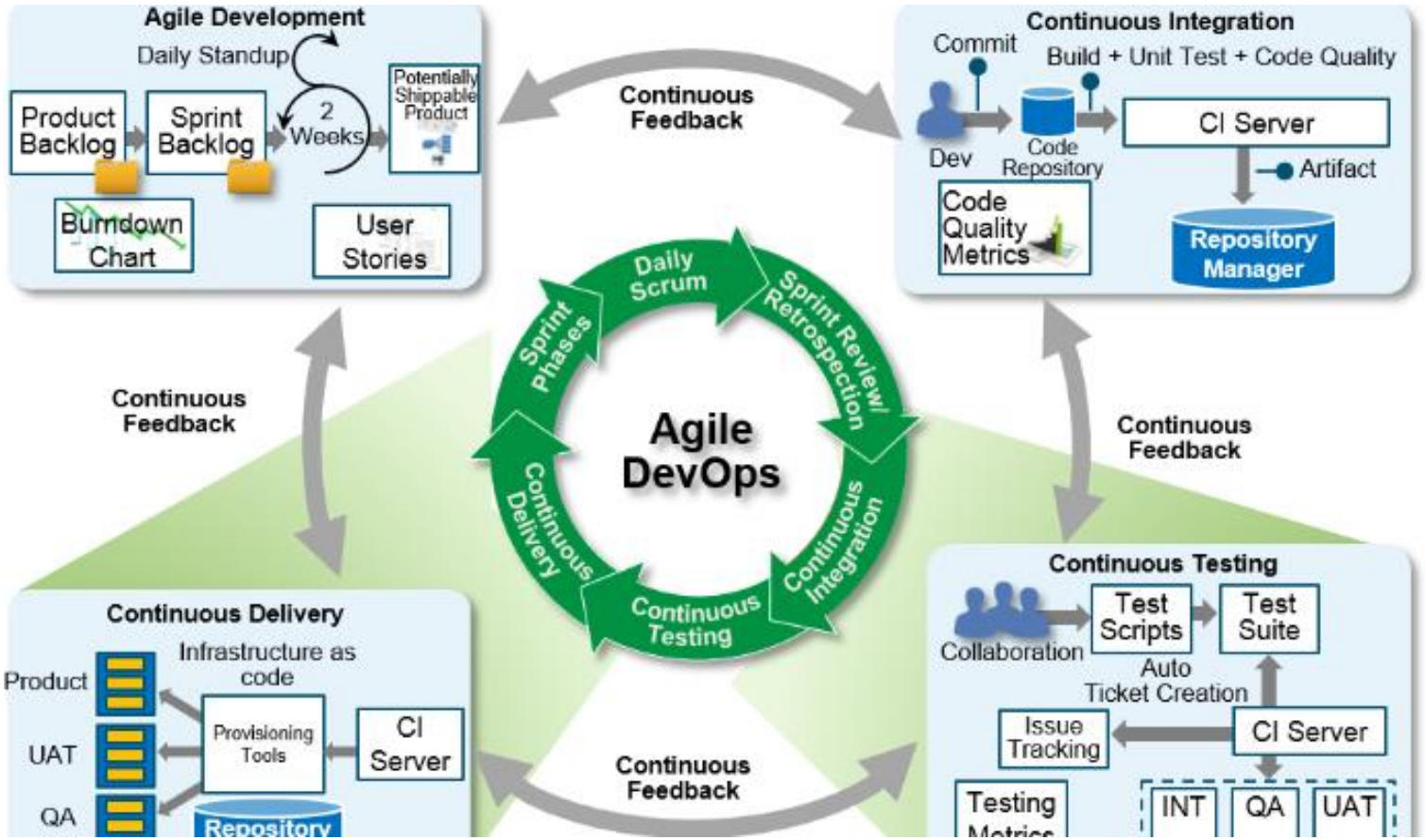
1 Understand demand



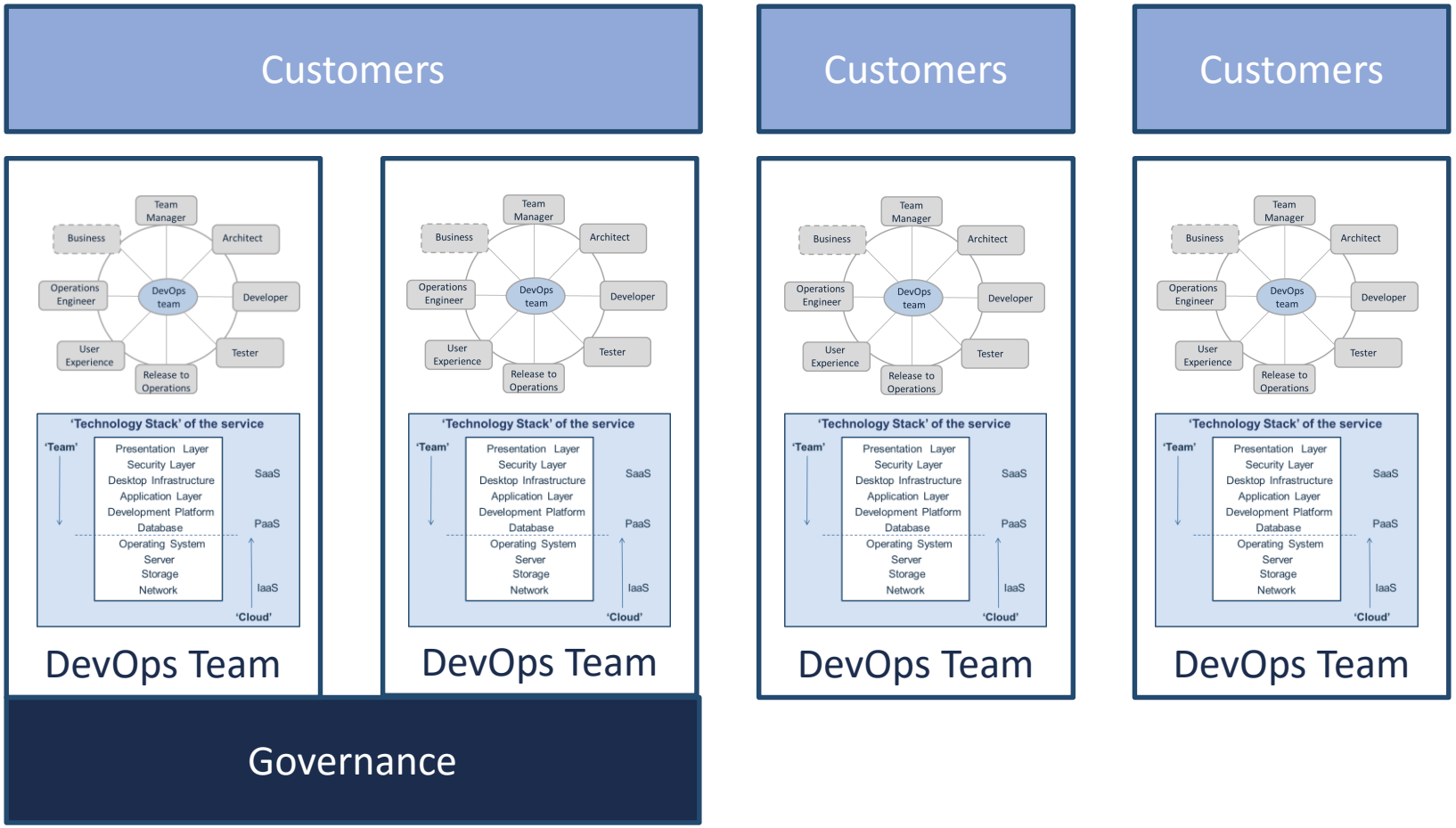
1 Continues deployment within the team



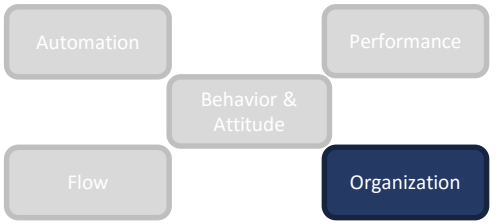
1 way of working within the teams



2 Create the structure with DevOps teams and necessary governance



3 Integrate Dev and Ops, based on a holistic view, starting from the current situation



5 DevOps Levels of Integration

Level 1: Low

Traditional IT organization

Level 2: Partial

Traditional IT organization with some DevOps; no clear DevOps vision

Level 3: Initial

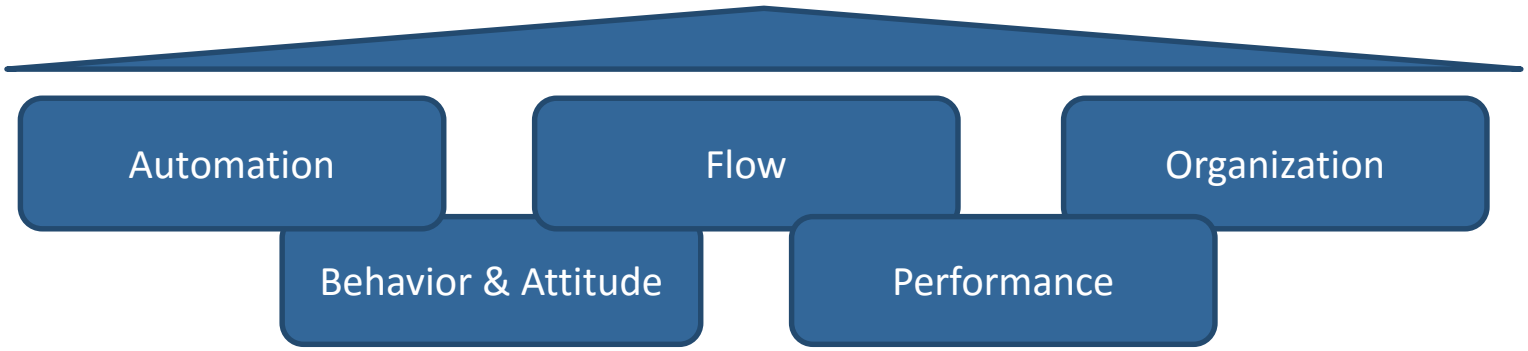
Dev and Ops closer together based on benefits

Level 4: Advanced

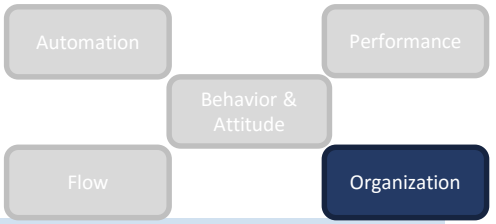
Dev and Ops together in teams based on vision

Level 5: Full

DevOps teams based on a integrated vision of providing value to customers



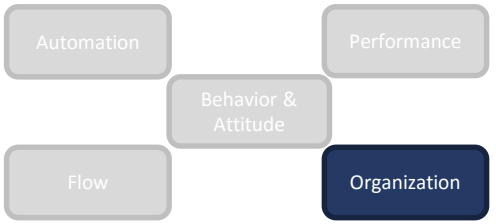
3 Combine dimensions and integration levels to achieve DevOps

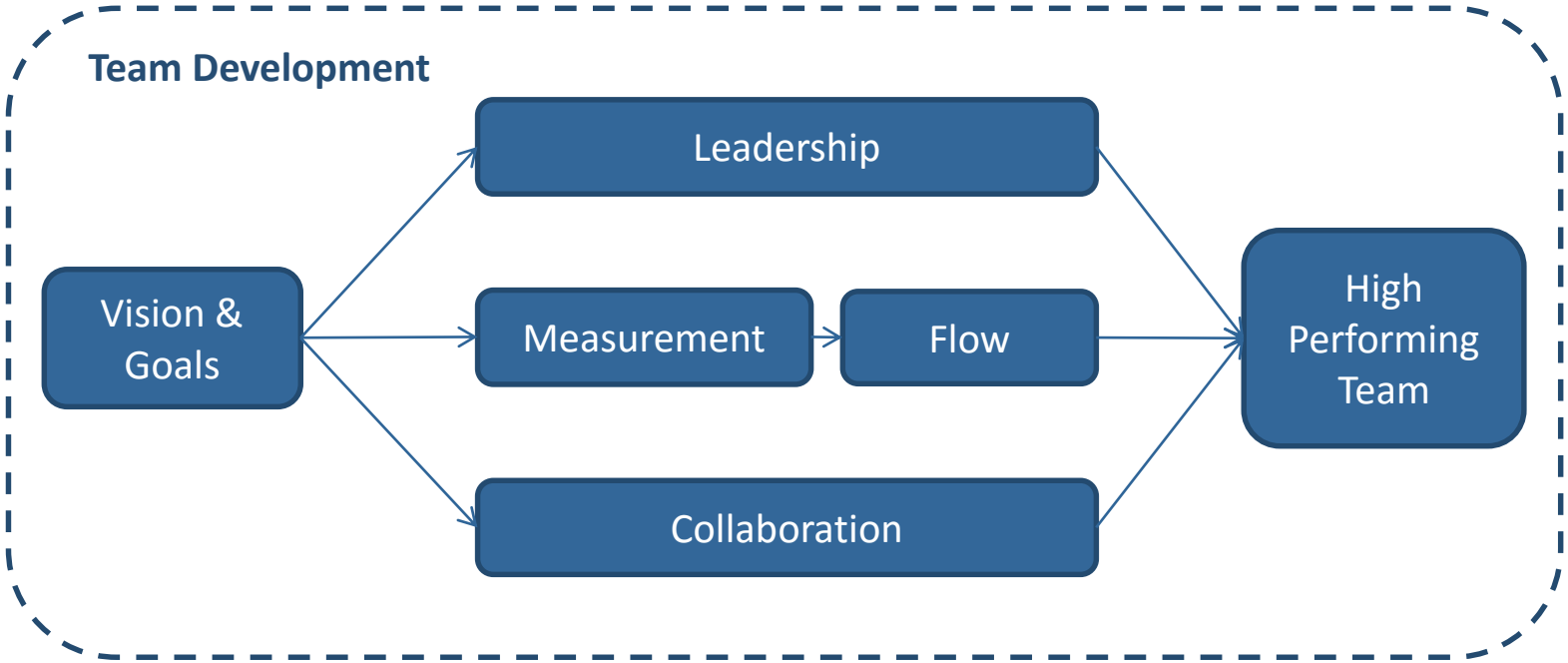


DevOps is characterized by integration rather than maturity

	Level 1	Level 2 Level 3 Level 4	Level 5
Organization	Individuals, lack of management and technical practices	→	High-performing integrated team with agreed practices
Performance	No measures, no steering	→	Measures integrated into way of working
Behavior & Attitude	Basic leadership, low motivation	→	Facilitating leadership, high motivation and continuous improvement
Flow	No customer orientation, poor processes	→	Processes are simple and short, few descriptions
Automation	No or little tooling for testing and deployment	→	Fully integrated suite of tooling for development, testing, deployment and service management

4 Focus on turning the DevOps team into a High Performance Team that delivers Customer Value

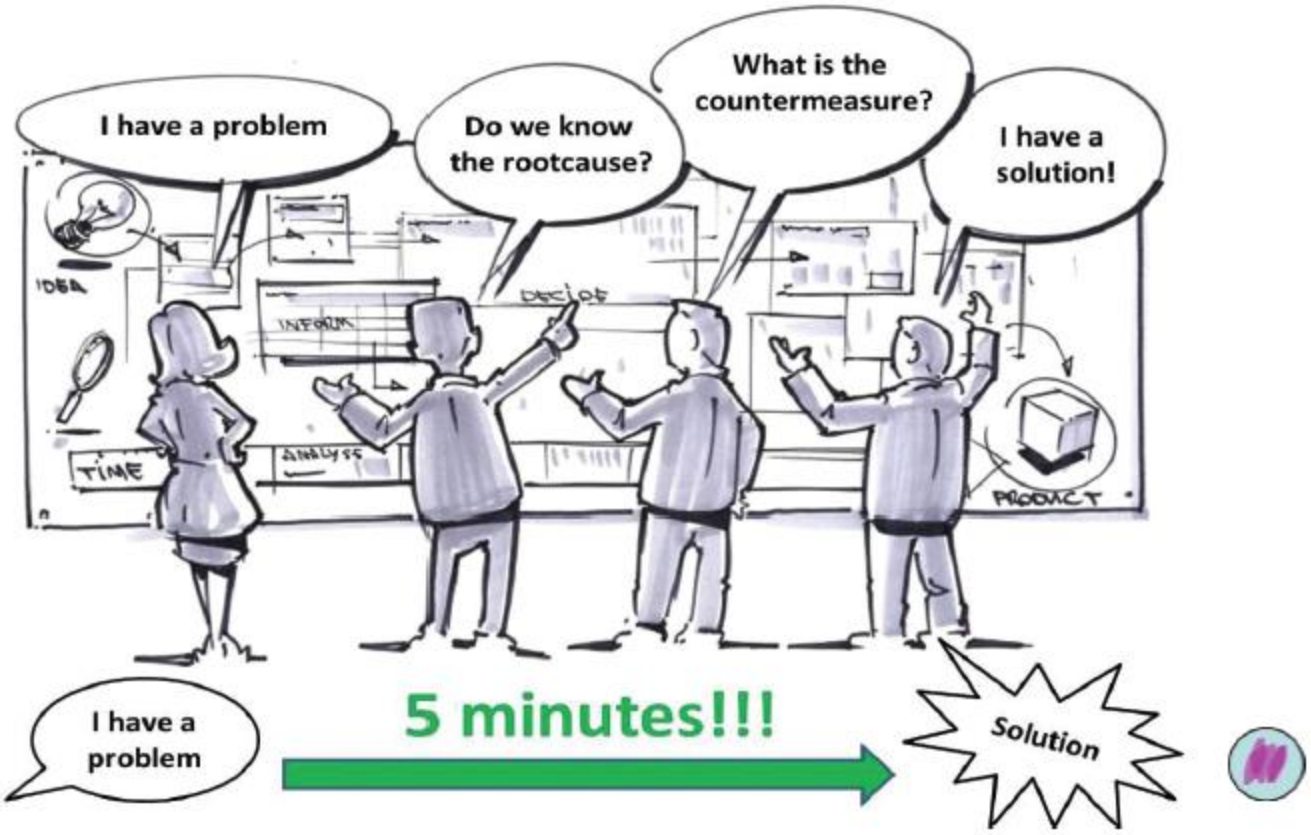




Governance structures

Governance within Teams

- Good governance within teams is the responsibility of everyone within the team.



BI Team use agile techniques to manage workload

Customer

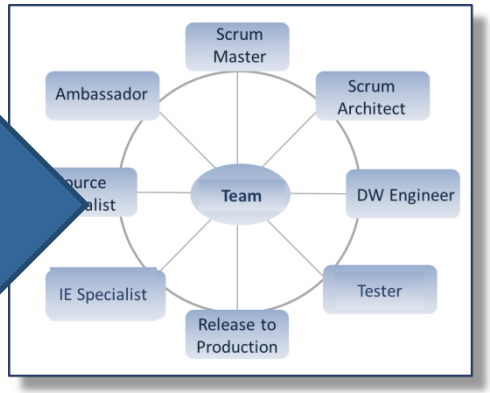
Incidents

BI Governance Team

Product Backlog

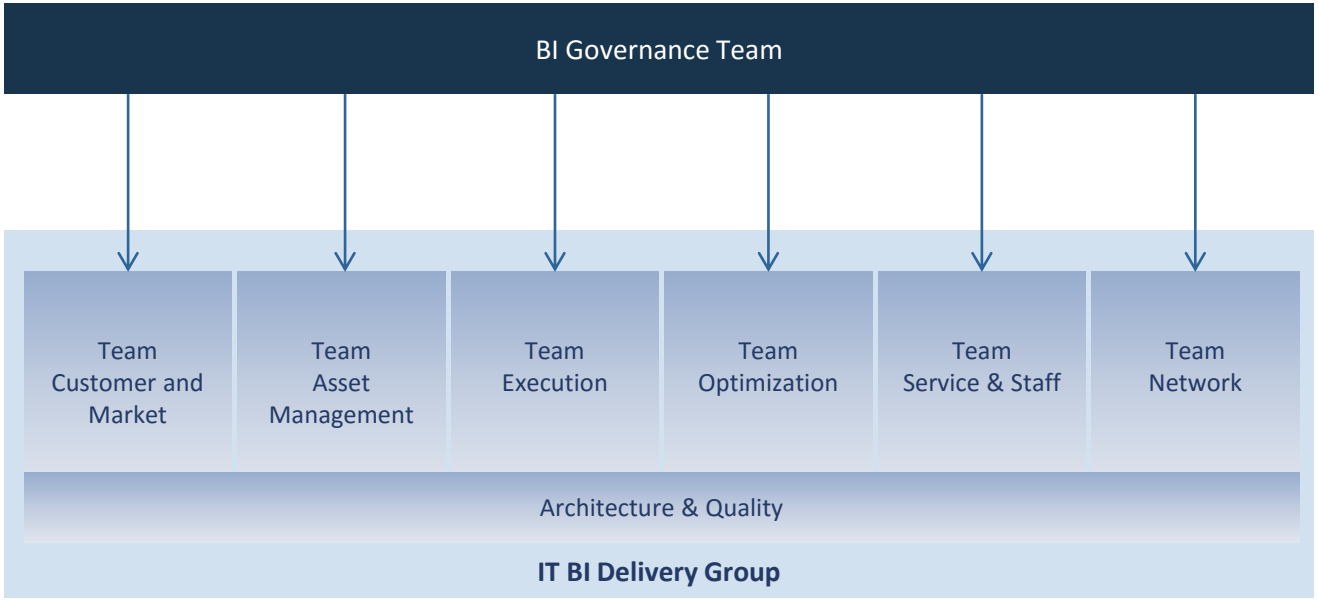
- Intakes
- Projects
- Functional changes
- Source changes
- Problems
- Advice
- Technical debt
- …

Team



Operational activities, service requests

BIGT directly linked to teams



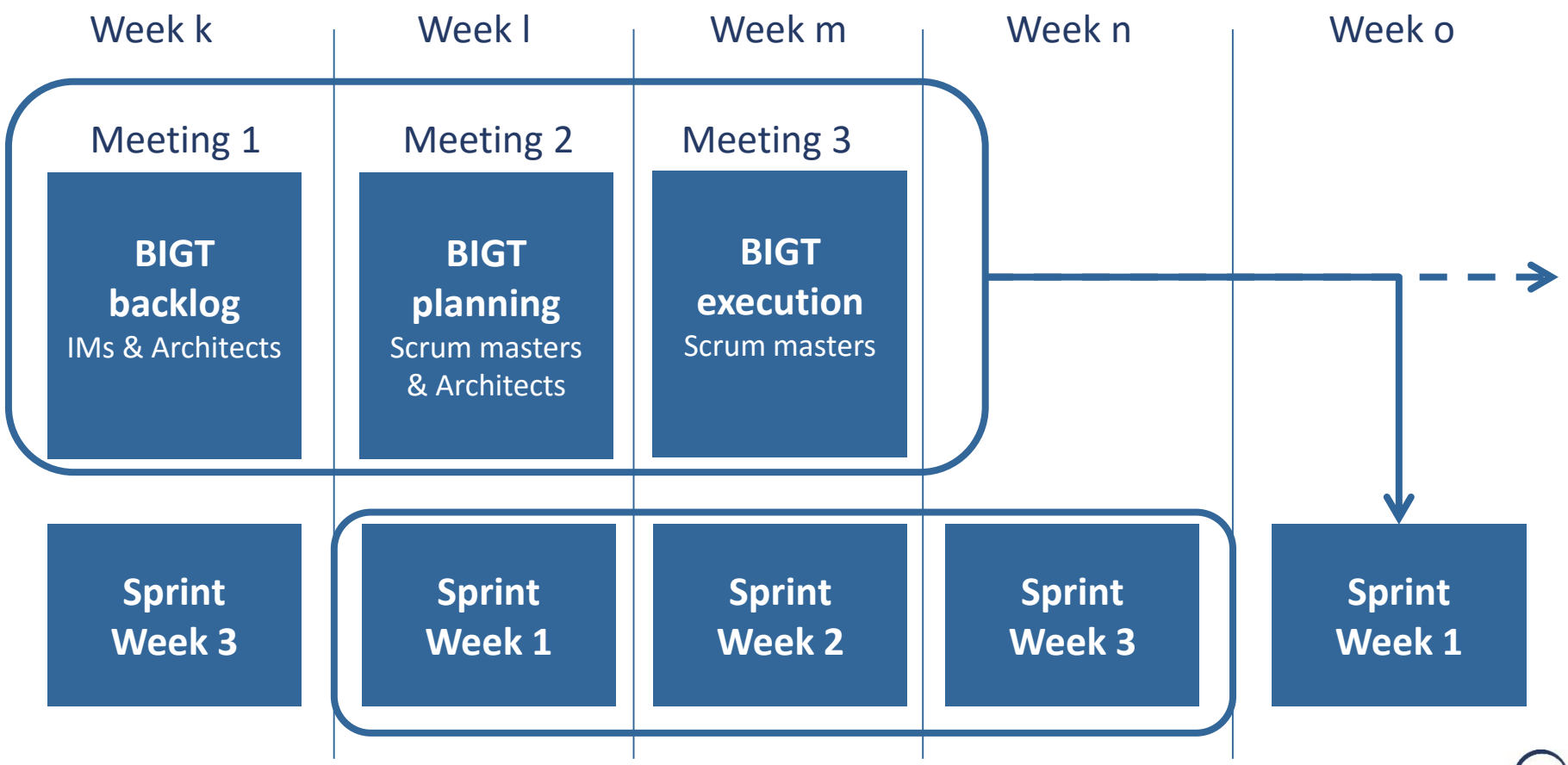
BIGT:

- All changes according to architecture
- Checks execution with availability of resources
- Changes allocated to most suitable team

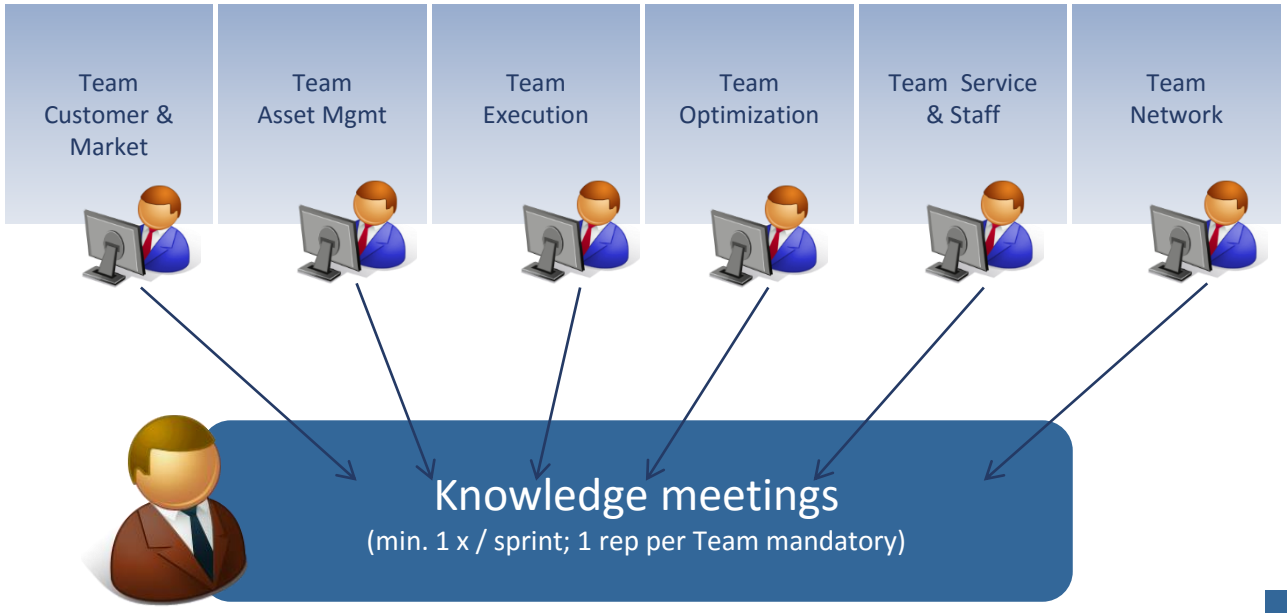
Benefits

- Direct steering of delivery reduces conflicting changes
- Central quality check to avoid new technical debt

BI Governance Team: Way of Working and Timing



Key part of improving the service: Knowledge development



Solution architect determines

- Knowledge and learning goals
- Technical Guidelines

Experts discuss

- Knowledge & learning goals
- Technical coordination

Benefits

- Up-to-date-knowledge
- Common technical way of working
- Prevents solutions that do not fit in architecture

Transformation

Approach creates predictability and stability in a changing organization and space for realizing ambitions

"Current state?"
"Future state?"
"Commitment?"

- Service Oriented IT analysis
- Agreeing design principles for DevOps teams and governance
- Leadership workshop(s) on DevOps principles, goals and behavior

Results in

- Current state of Service Oriented IT
- Design principles
- Commitment leadership to DevOps principles and goals

- Management

" Organizing teams and governance"

- DevOps training
- Detailed team and governance designing
- Defining technology stacks
- Transformation roadmap workshops

Results in

- Detailed definition of customers, services, teams and technology stacks
- Roadmap and communication plan

- Management, Architects, Lead technologists, Team leads, HR

"Working in DevOps teams"

- Form DevOps teams
- Awareness and training
- Prepare Team for integration
- Analysis of improvement potential on DevOps goals

Results in

- Detailed Integration plan per team
- Trained employees
- Continuous service improvement roadmap

- Management, Teams in scope

"High performance on collaboration and technology "

- Intensify collaboration
- Improve flow
- Extensive automation
- Multi-skilled team members
- Focus on customer-oriented performance

Results in

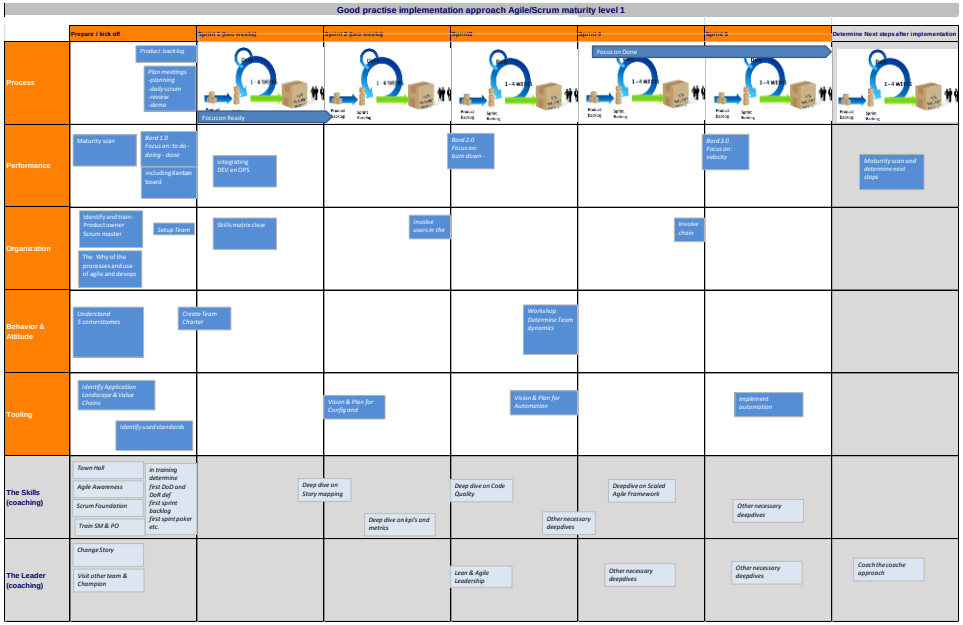
- Improved Quality of delivery
- Higher Business Value
- Increased Speed of delivery
- Improved Efficiency

- Management, DevOps teams

Getting DevOps to work

DevOps is an individual path per team

- Choose a common reference framework (DevOps Integration Model) to facilitate learning
- Assess the current situation per team/department
- Create individual plan per team to achieve the selected level of integration
- Pay attention to the governance of the DevOps teams



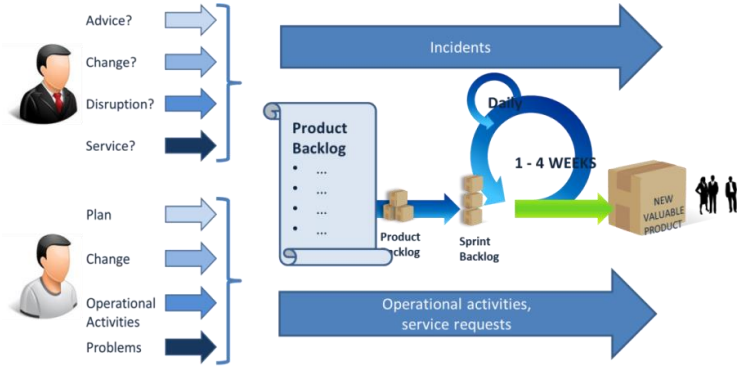
Start with assessing the current situation of the team, using the Quint DevOps Integration Model

The Integration model from Dev and Ops to a DevOps team, and on to a high-performing DevOps team

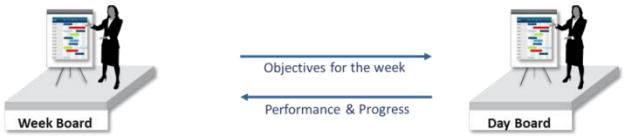
		DevOps Integration						
			Low	Partial	Initial	Advanced	Full	
		Level 0	Level A	Level B	Level C	Level D	Level E	
Organization	**Service Definition**	1	2	3	4	5		
	Customer definition	1	2	3	4	5		
	Completeness of team		1	2	3	4	5	
	Dependency on other teams			1	2	3	4	5
	Management practices		1	2	3	4	5	
Performance	**Key Performance Indicators**		1	2	3	4	5	
	Technical Debt reduction		1	2	3	4	5	
	Level of performance (Incidents, SRs, St Ch.)	1	2	3	4	5		
	Level of performance (Creative work)		1	2	3	4	5	
	Time usage (Earning/Burning)		1	2	3	4	5	
Behavior & Attitude	**Team-forming**	1	2	3	4	5		
	Collaboration	1	2	3	4	5		
	Problem-solving		1	2	3	4	5	
	Leadership fit	1	2	3	4	5		
	Motivation		1	2	3	4	5	
Flow	**Knowledge of units of work**		1	2	3	4	5	
	Awareness of processes		1	2	3	4	5	
	Documentation of processes		1	2	3	4	5	
	Monitoring of progress		1	2	3	4	5	
	Customer orientation	1	2	3	4	5		
Automation	**Automated Test**		1	2	3	4	5	
	Development Tooling	1	2	3	4	5		
	Automated Deployment to Production		1	2	3	4	5	
	Service Management Tooling	1	2	3	4	5		
	Use of other tools		1	2	3	4	5	

Integrate Technical and Management Methods

Ensuring the right habits makes the DevOps team more effective



Agile & Continuous Delivery



Lean & Visual Management

- Standardize methods to encourage learning
- Define the interface with the customer, together with the customer
- Define the flow of units of work through the team, e.g. Waterfall, DSDM, Agile,
- Define the management principles used by the team, e.g. Lean, TOC,

Success factors

- Start small with quick wins
- Attention to culture and tools
- Transparency and adoption
- Ensure the process is end to end
- The attitude of ‘doing it right now’
- Automation of testing and deployment
- Track and analyze impact of changes
- Constant cooperation within the teams and over the teams
- Get buy in from senior management

Big Bang VS Snowball

Big Bang

Disadvantages

- Critical processes can't be tested first
- Changes apply to all, could be difficult to master and the impact could be huge
- No flexibility in scheduling
- Impact of changes are difficult to measure, it all comes at once

Advantages

- It applies to everyone, no jealousy
- Requires little or no planning.

Snowball

Disadvantages

- Number of people are first and some are last, jealousy my occur.
- Requires planning.

Advantages

- Changes only apply to small groups, impact is small
- Incidents can be sorted out and conquered first, to stabilize before taking the next step
- The process are tested frequently and become more and more reliable.
- The process is reviewed over and over, inefficiencies could be optimized every time.
- Small changes early results

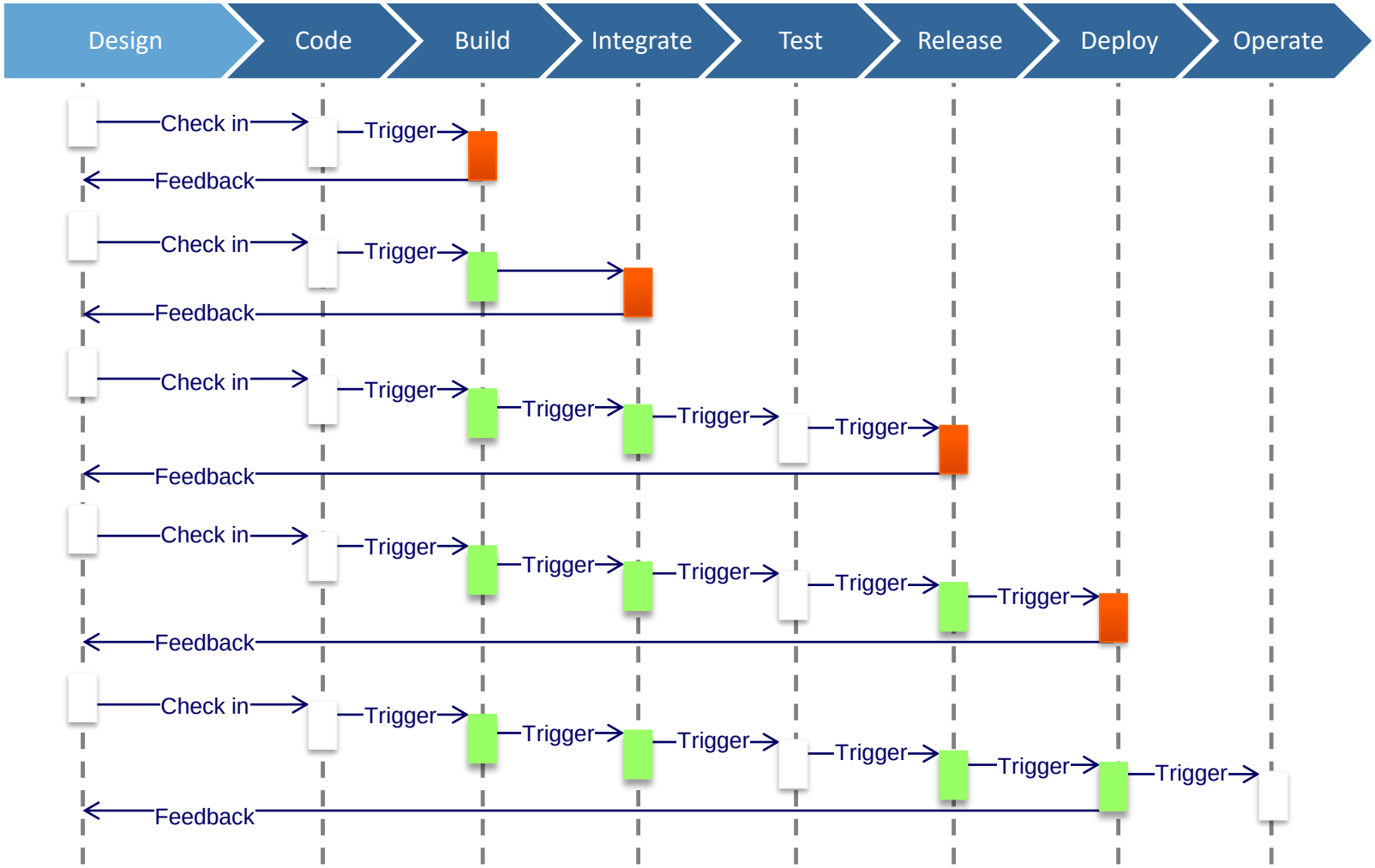
Automation

“We find ourselves now with two camps in the DevOps community: technical and non-technical. Either side shifts the conversation to where they feel most comfortable: the domain they work in and understand. Technical folks talk about tooling. Non-technical folks talk about culture.”
http://dustinrcollins.com/devops-a-house-divided

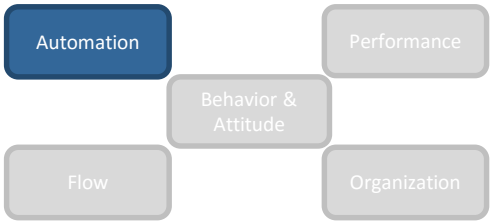
Continuous delivery covers the full deployment pipeline

- Treating infrastructure as code by programmatically provisioning and managing infrastructure resources
- Repeatable and reliable deployment processes
- Development and testing (preferably automated testing) performed against production-like systems
- On-demand creation of development, test, staging and production environments
- Proactive monitoring of infrastructure components, environments, systems and services

ENGINEERING FLOW OF CONTINUOUS DELIVERY



Continuous Integration, delivery & deploy



Continuous Integration: Integrate early and often to minimize the gap between development and operations to increase:

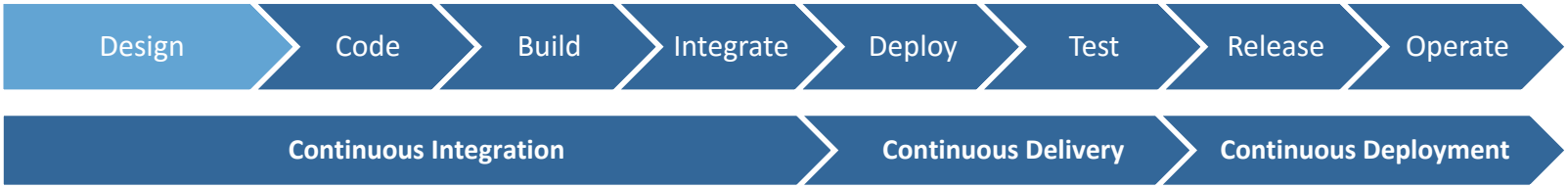
- quality assurance
- test driven development
- configuration management

Continuous Delivery: extension of continues integration and ensures that every change is releasable:

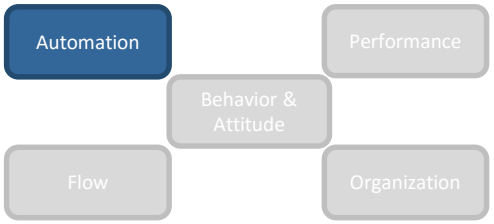
- Automation to improve processes software delivery including integration and acceptance testing.
- Release in one push on the button

Continuous Deployment: end to end software delivery which includes continues integration and delivery:

- Automate the release of the delivered software
- Automated infrastructure, Release Management
- reduced lead time, earlier feedback, reduced risk

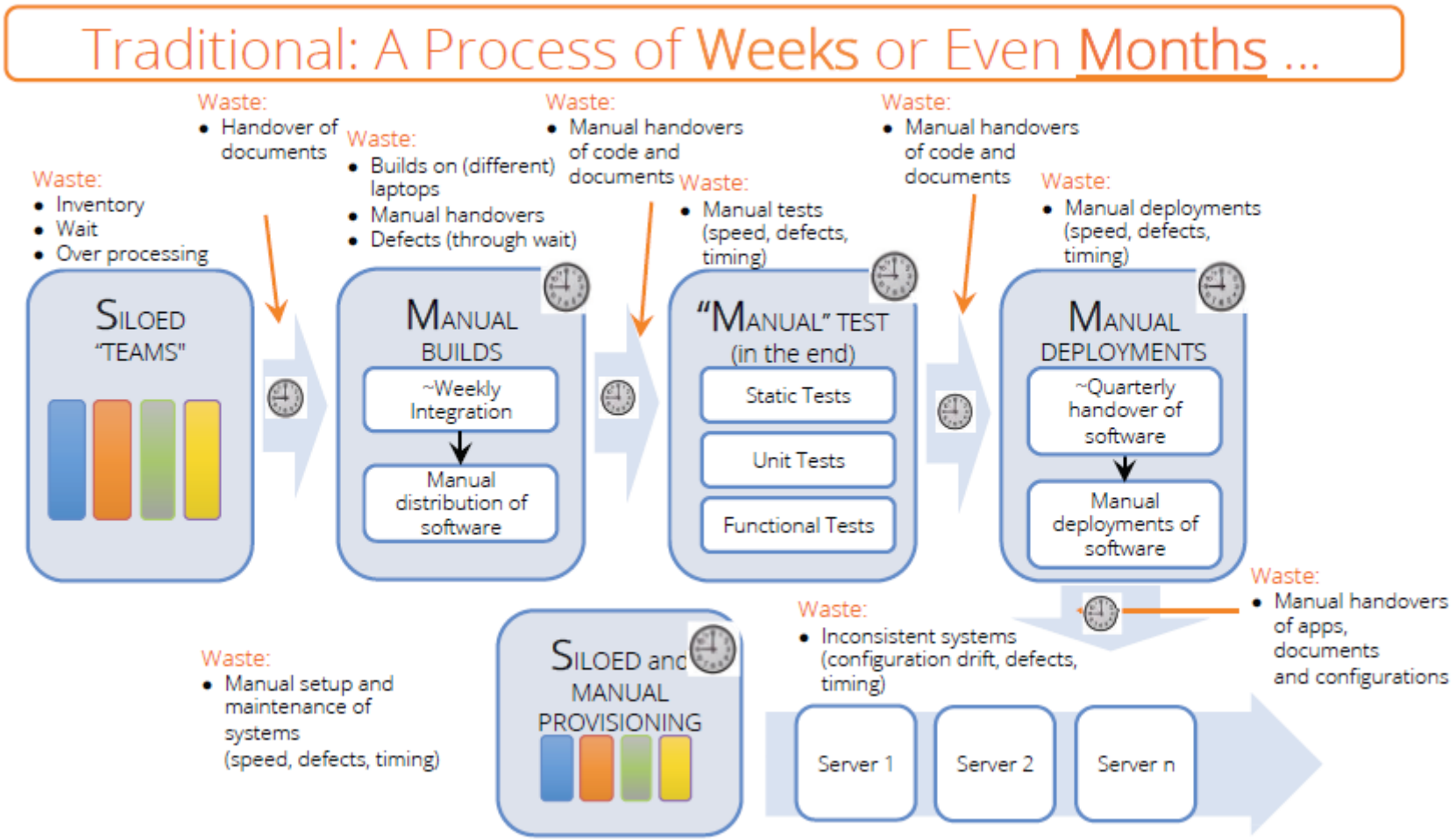


Automation is vital for ensuring speed, consistency and quality of work

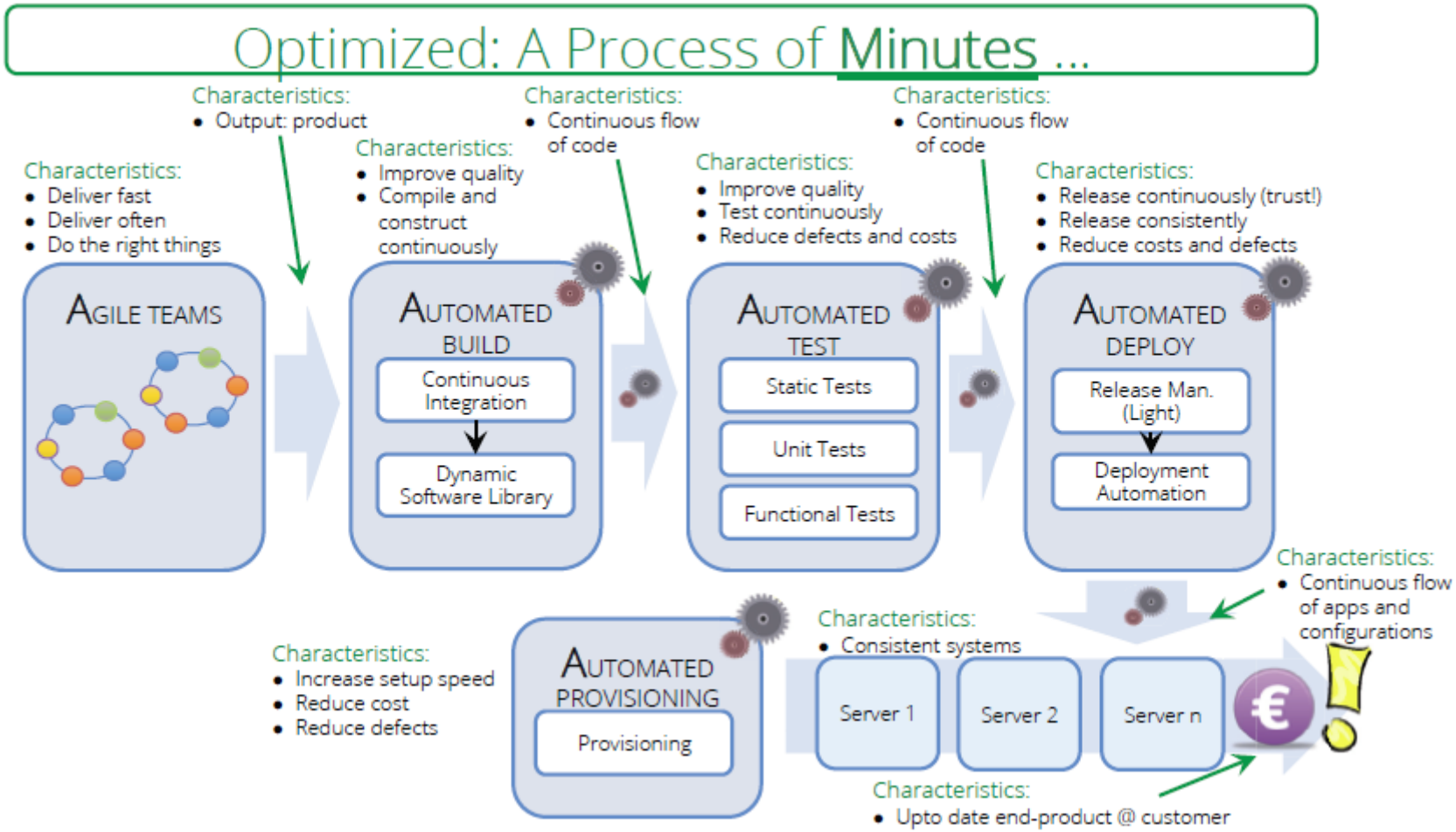


	What is it	Why is it important
Service Management Tooling	Tool in which all units of work are registered	Monitoring progress and data for the overall KPIs
Automated Test	Tooling to ensure the quality, completeness and speed of the test process.	A tool can generally test more quickly and accurately
Automated Deployment to Production	Automated flow from dev through test and acceptance to production	Reduces the chance of mistakes; ensures the consistency of environments
Development Tooling	Tooling supporting the development process	Reduces the time required for developing working software
Use of other tools	Non-technical tools e.g. checklists, guidelines and architecture drawings.	Not everything is integrated into tooling, but does need to be available for the team

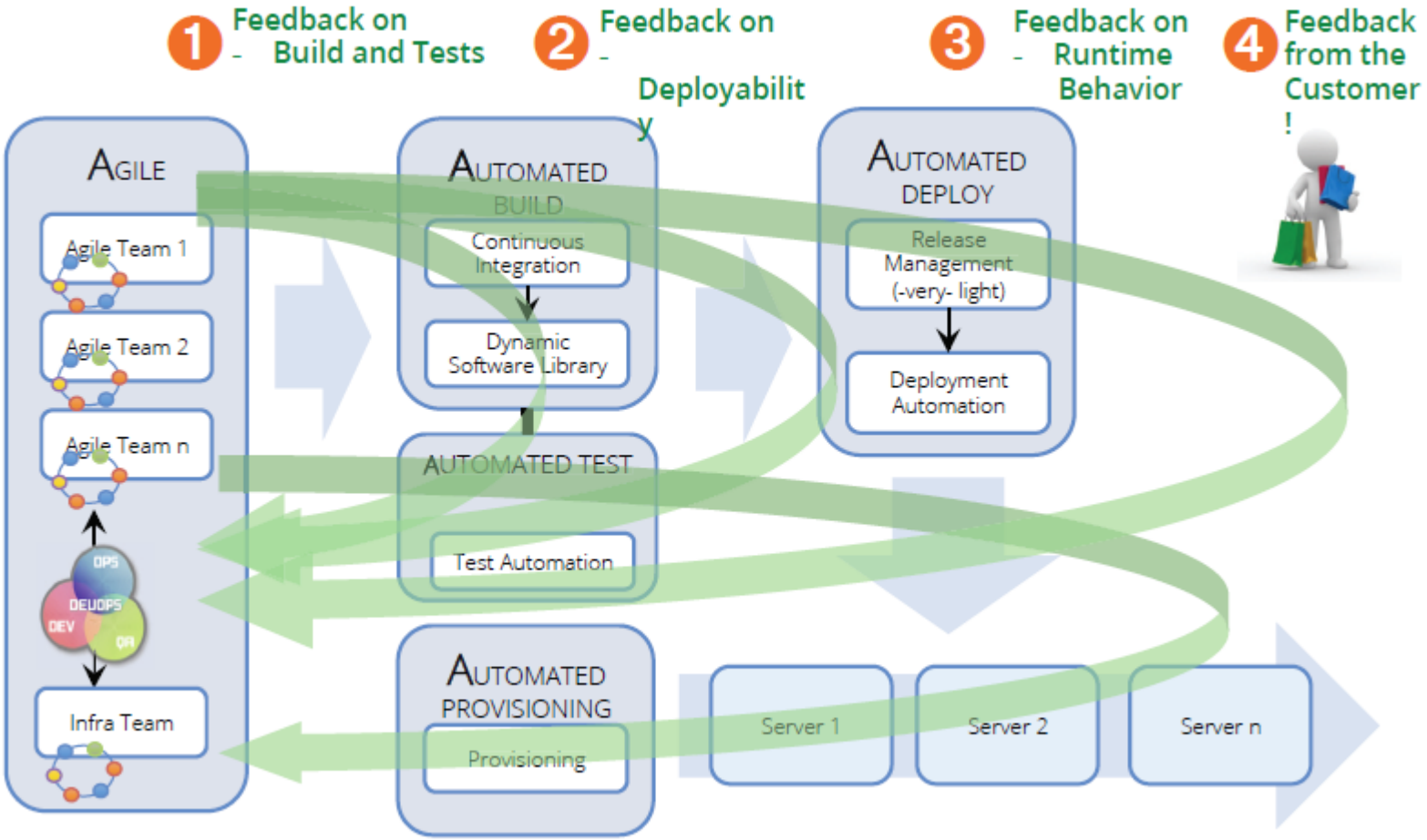
Continues Delivery Implies: Software has to Flow



Continues Delivery Implies: Software has to Flow (Contd.)

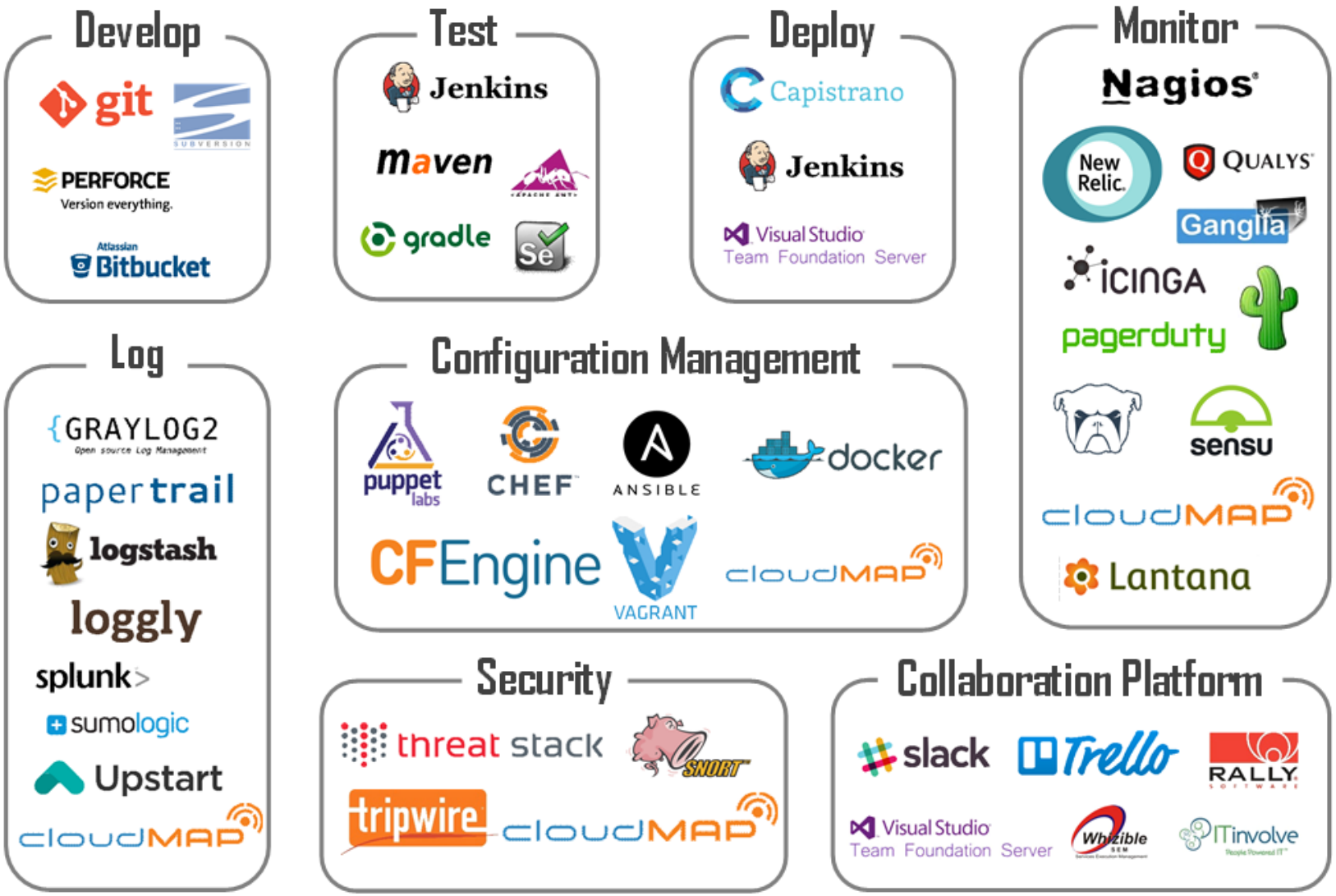


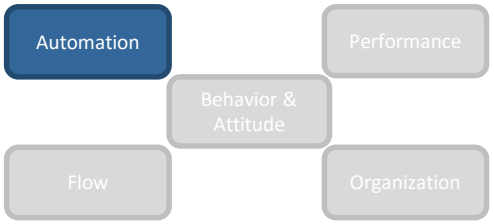
Type of Feedback



TYPICAL TOOLS THAT SUPPORT THIS PRINCIPLE OF CONTINUOUS DELIVERY

Tools for automation





Benefits of automation

Automation supports

- Faster lead times
- More frequent releases
- Less turbulent releases
- Fewer errors
- Higher quality
- Faster recovery
- Business and customer satisfaction

Automation enables computers to do rote tasks and allows people to

- Solve problems
- Make decisions based on feedback
- Use their skills, experience and Judgment

Considerations

Considerations with DevOps

- Legacy systems: first do DevOps then let the team clean up the legacy
- Team forming
- Short cycles of feedback from customer to DevOps team
- Short cycles of feedback from P to DTA
- Real service definition
- Encourage the self-cleansing capability of the team (“Eat your own dog food”)
- Sizing for DevOps – how big must my organization be?
- Shared resources or underutilized resources
- What happens to all the process management roles?

Other considerations … (1)

There are other details to be ironed out

How to deal with the old-school segregation of duties?

- The release manager role ensures that there is no unauthorized access to the production environment.
- As a result of the increase in deployments, the need for quick, unauthorized access is reduced as the process fulfills the needs of the team.
- Tooling can also ensure that SoD is facilitated

How to deal with the ‘shared resources’ phenomenon?

- “I have one DB expert who reviews all major queries on our database system. But now I have 4 DevOps teams? In which one should he be?”
- The key is to understand how much time the expert needs to spend per team.
- If, for example, there is 1 day of work per team, then the shared resource should focus on sharing knowledge and skills with people in the team. The expert can then be used for the specialist tasks and project support.
- This expert should be evaluated on flow efficiency rather than resource efficiency.

Other considerations … (2)

There are other details to be ironed out

How to deal with the promise of increased efficiency?

- Since I get faster, automated and better at things…, I guess this whole DevOps thing is a great way as a drastic cost saver, no?
- There is a basic minimum size to a DevOps team; below this level, the service can no longer be guaranteed. The aim is to continually increase the amount of value added by the team. The IT service can always be further automated and improved.
- If there is truly too little work, expand the scope of the team.

How to deal with customers?

- “Since one of the Agile principles (and also very Lean) is to make business people and developers work together daily throughout the project, shouldn’t we call it BusDevOps…or value stream management?
- The intention of DevOps is to improve the flow efficiency of the units of work that enter the team. Involving the customer gives a greater certainty that the team is also dealing with the right things
- A DevOps team must have its processes in place so that the customer is enticed to join in

Other considerations … (3)

There are other details to be ironed out

How to deal with visualization?

- My Agile board starts with epics, to features, to sprint items, to stuff in integration and then ‘done’…; so what columns do I add once Ops is in my team? Are we still working on the same units of work?
- The units of work are identical only we use generic common names for them (incident, change, project, service request, problem). These are used on the team’s VM boards. The details per unit of work will differ

How to deal with governance?

- “I’m currently a project manager. I already have two scrum masters on board of my development team plus a product owner. And now there’s this team leader of the Ops team. Who is going to be the appointed manager in this circus and what is its main responsibility?
- It works the other way round. First, there is a service. Second, there is a team that supports it. Within the team, all units of work are processed. The ‘project manager’ may have the role of team leader, or someone else may have the role. In this case, the project manager reports to the team leader.

Other considerations … (4)

There are other details to be ironed out

How to deal with ‘interest’ or lack thereof?

- One of the traditional arguments against DevOps is that the people involved are not interested in each other’s work: a developer does not understand and is not interested in operations and vice versa
- This is the cultural change that needs to take place in the team: there is a common goal – the health of the IT service and everyone plays an equally important part. It is vital for the team spirit that each team member is aware of the contribution that each of the team members makes.
- The lack of interest is something that we have created as a result of out way of steering and structuring the organization and individuals

You've been introduced to DevOps

Now what?