2019

NT KONFERENCA

21. - 23. MAJ 2019

#ntk19

PROFESSIONAL SCRUM FOR DEVELOPMENT TEAMS WITH AZURE DEVOPS

Ana Roje Ivančić
Ognjen Bajić

Microsoft MVP for Developer Technologies
Professional Scrum Trainer (PST) for Scrum.org

Agilist IT
INFORMATION TECHNOLOGIES

Zagreb, Croatia

MVP Microsoft® Most Valuable Professional

PST Professional Scrum Trainer
Scrum.org

2019
NT KONFERENCA
21. - 23. MAJ 2019

Agenda

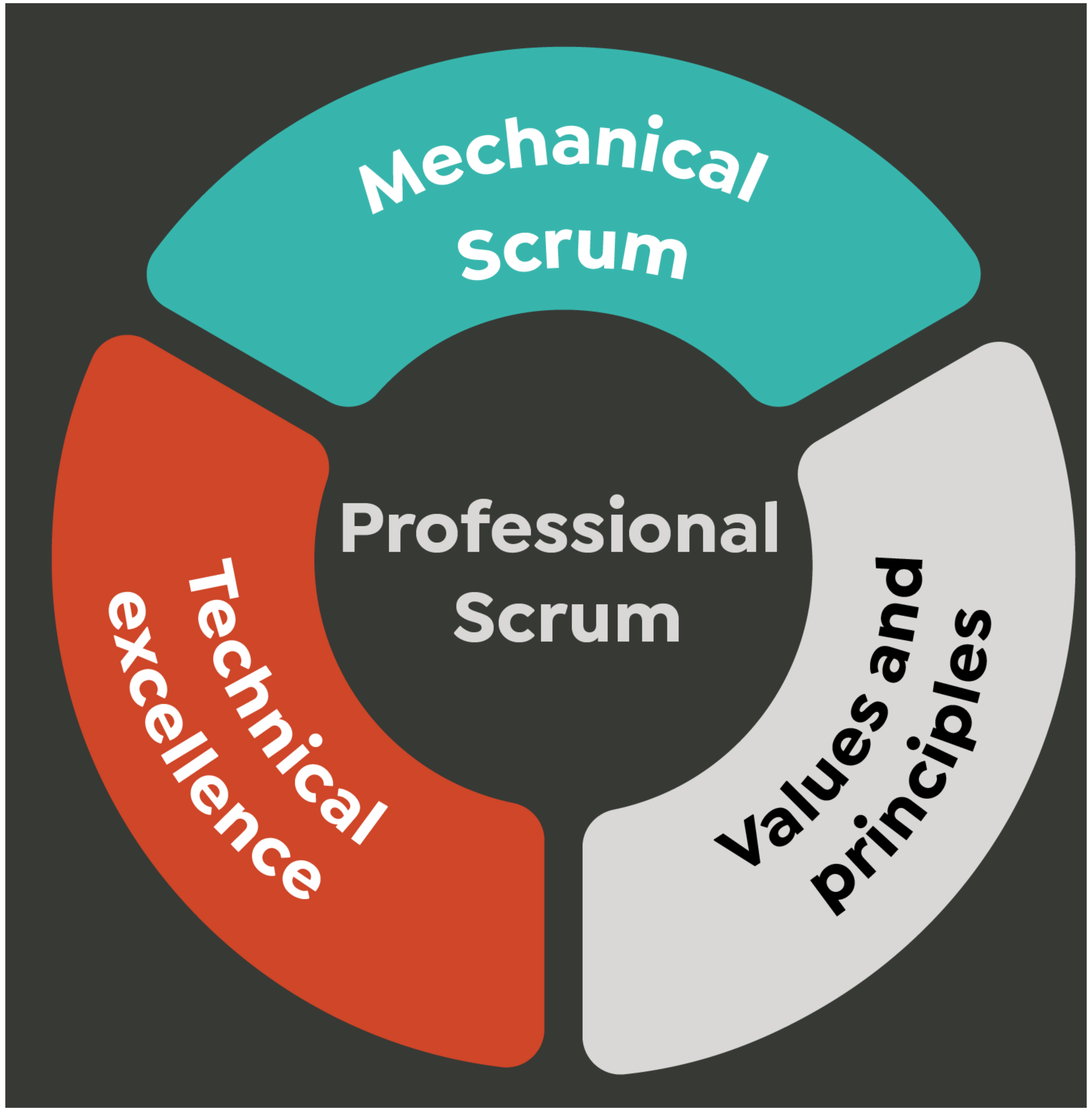
Mechanical Scrum

Scrum Values and Principles

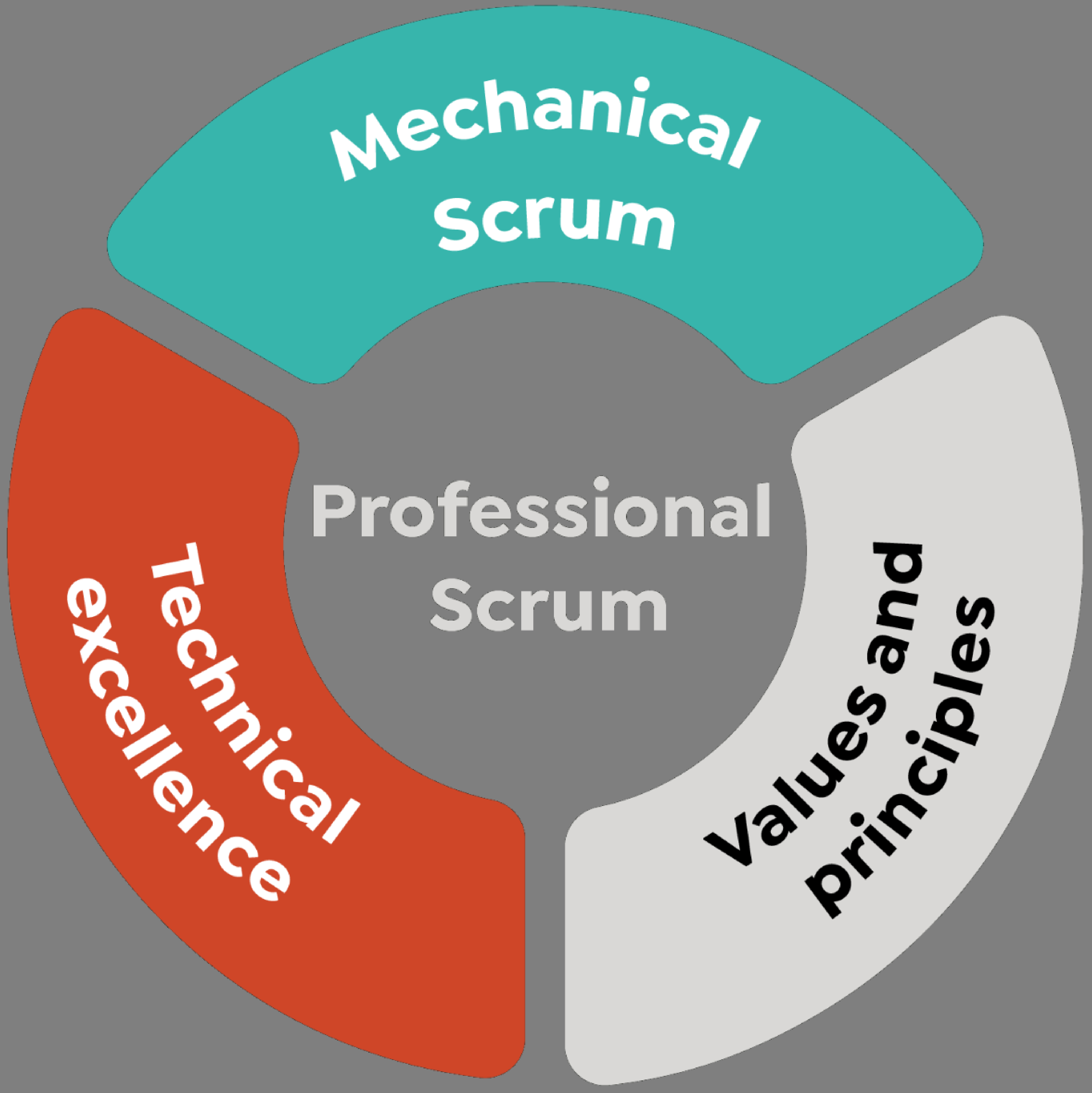
Professional Scrum

Technical Excellence – DevOps

Azure DevOps



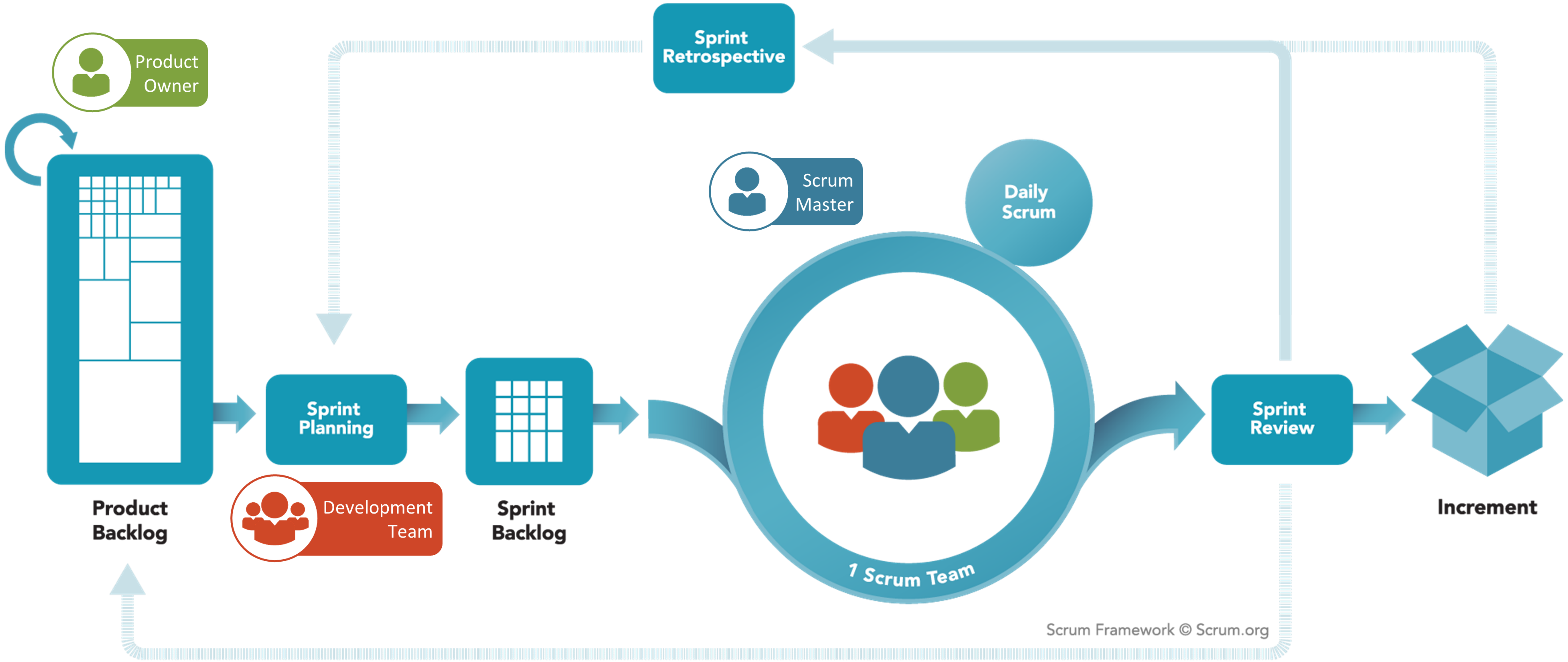
MECHANICAL SCRUM



Scrum In A Nutshell

#ntk19

1. The team forecasts to deliver working software in 30 days or less
2. The team creates working software
3. The software is presented for inspection to stakeholders
4. The plan is adjusted according to feedback and new insights
5. GOTO 1.



Scrum Framework © Scrum.org

Roles, Events, Artifacts, Rules

Rules

Roles
Product Owner
Scrum Master
Development Team

Note:
All Roles together form the Scrum Team

Rules

Rules

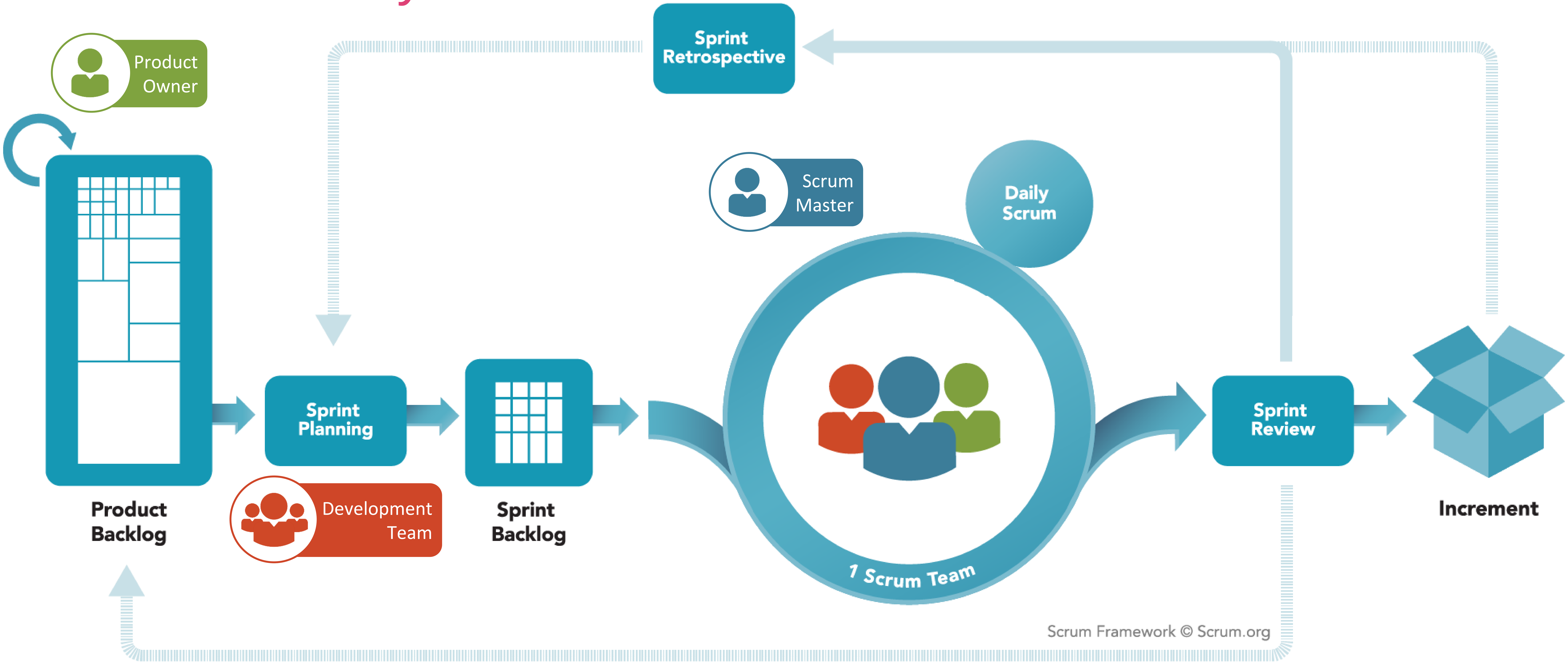
Events
The Sprint
Sprint Planning
Daily Scrum
Sprint Review
Sprint Retrospective

Note: All Events are timeboxed

Artifacts
Product Backlog
Sprint Backlog
The Increment (of potentially releasable product)

Scrum Lifecycle

#ntk19



Scrum Framework © Scrum.org

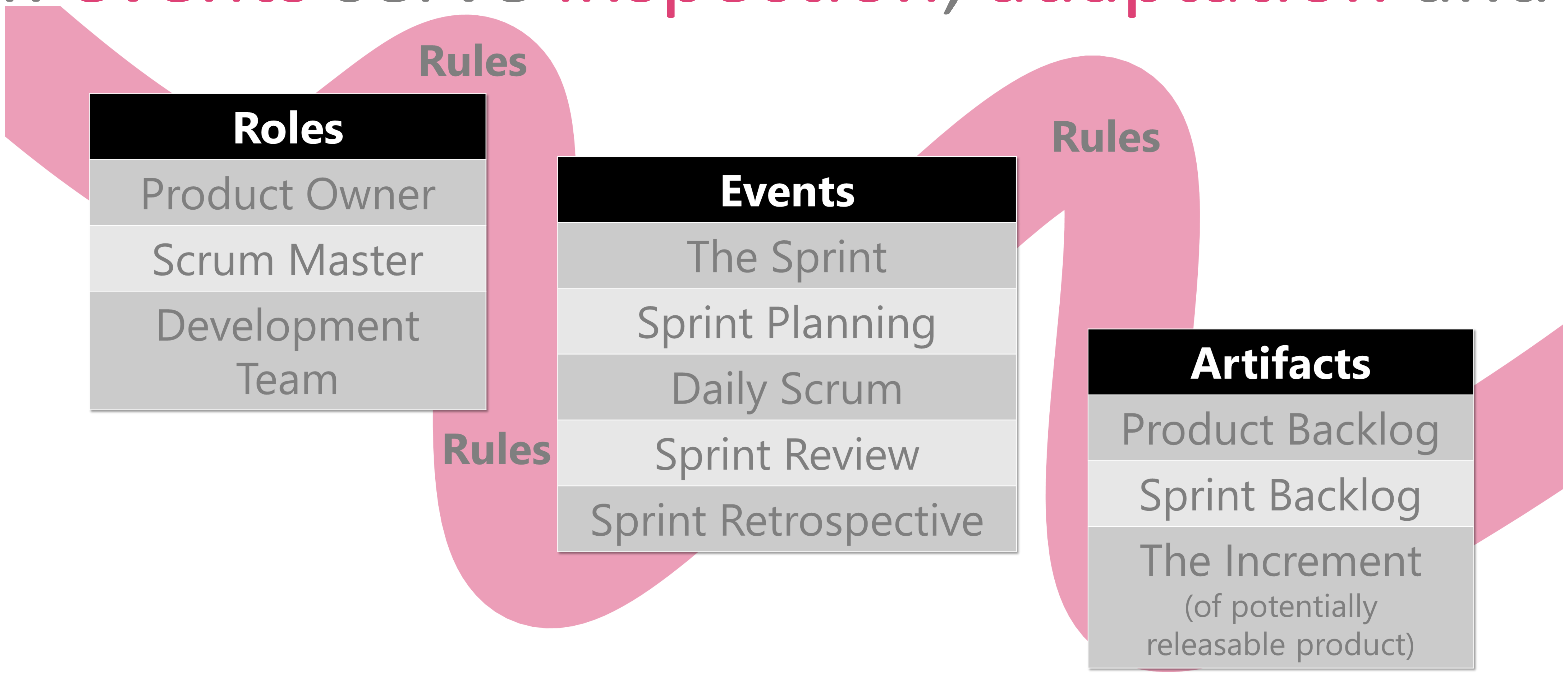
Every Scrum Component is Essential

Each **component** within the framework serves a specific purpose and is essential to Scrum's success and usage.

Every Scrum role has a clear accountability

All Scrum artifacts provide transparent information

All Scrum events serve inspection, adaptation and transparency



Roles
- Product Owner
- Scrum Master
- Development Team

Events
- The Sprint
- Sprint Planning
- Daily Scrum
- Sprint Review
- Sprint Retrospective

Artifacts
- Product Backlog
- Sprint Backlog
- The Increment (of potentially releasable product)

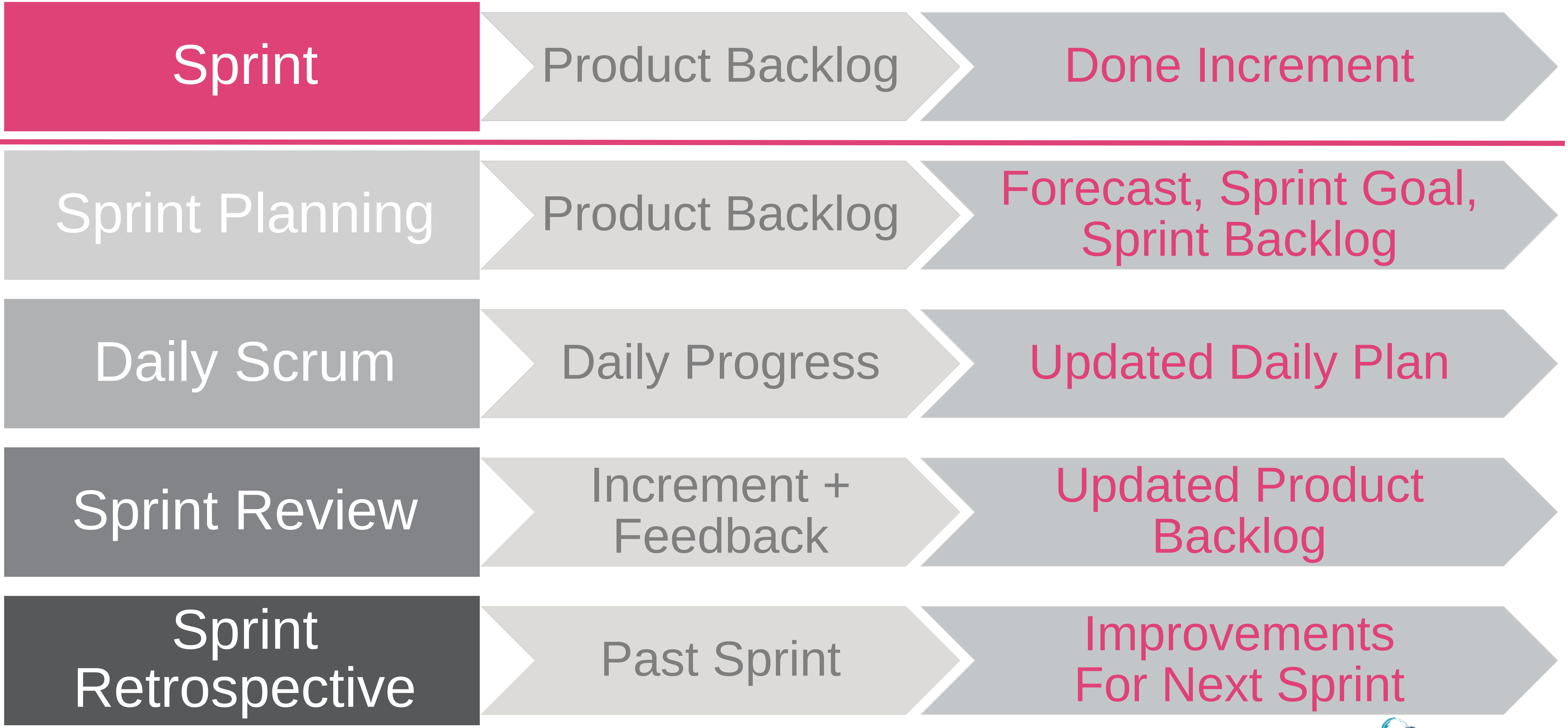
Roles: Each One Has a Clear Accountability

Product Owner **Single person**	**Development Team** **3-9 members**	**Scrum Master** **Single person**
Represents the Customer Drives the product vision Manages the Product Backlog Plans iterations Responsible for success/ROI	Plans iterations Runs Iterations Cross functional Self-organizing Accountable as a whole	Establishes Scrum practices and rules Shields the team Educates everyone Facilitates collaboration
Scrum Team		

Artifacts: Each One Contains Specific Information

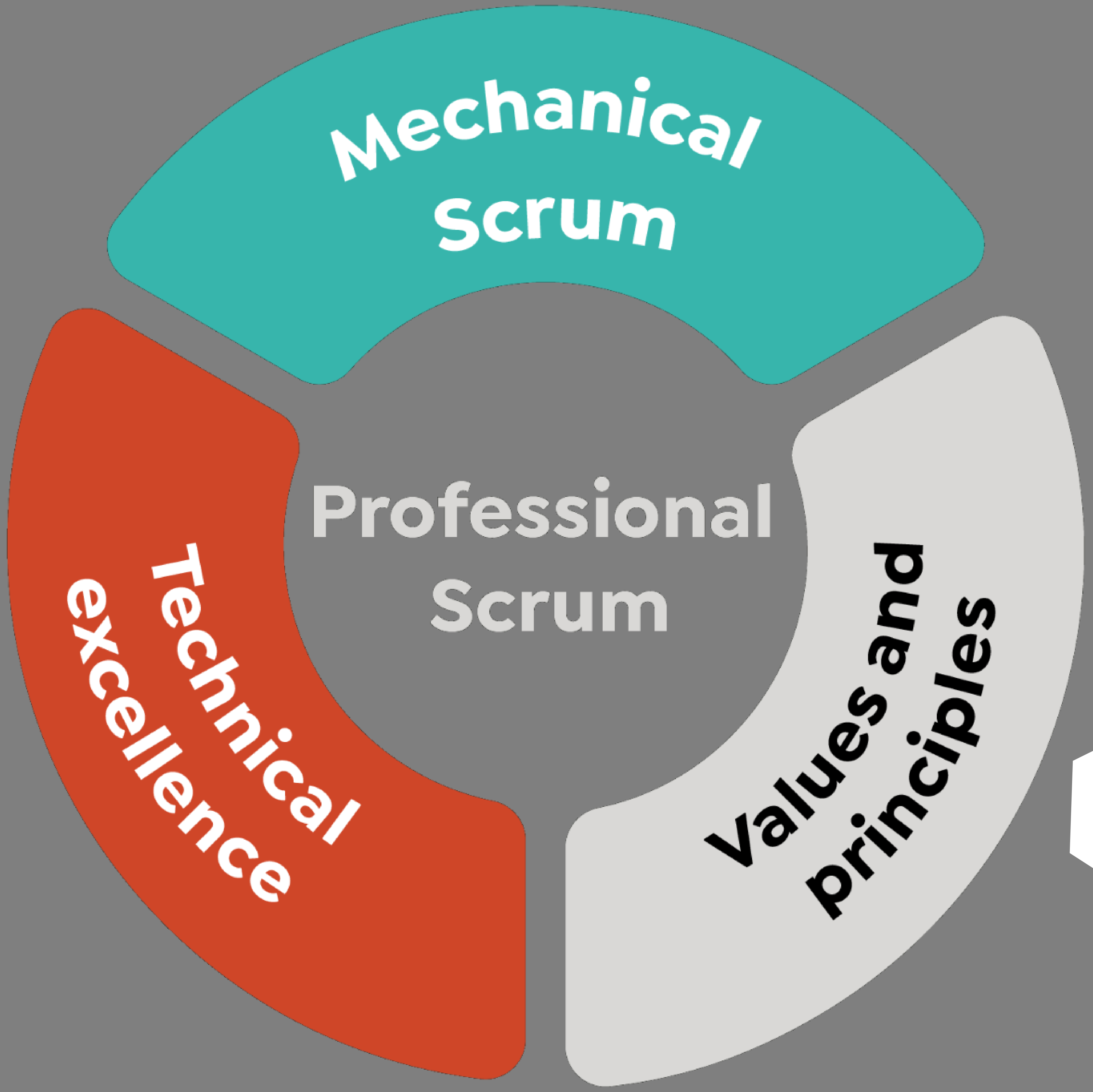
Product Backlog	• Holds the requirements for the product • Managed by the Product Owner
Sprint Backlog	• Holds all work for the Sprint Goal • Managed by the Development Team
Increment	• Working addition to the product • Done + potentially releasable

Events: Each One Has a Specific Purpose



Sprint	Product Backlog	Done Increment
Sprint Planning	Product Backlog	Forecast, Sprint Goal, Sprint Backlog
Daily Scrum	Daily Progress	Updated Daily Plan
Sprint Review	Increment + Feedback	Updated Product Backlog
Sprint Retrospective	Past Sprint	Improvements For Next Sprint

SCRUM VALUES AND PRINCIPLES



Scrum Values

The Scrum Team members have **courage**	• to do the right thing and work on tough problems. • in saying „no” to cutting quality under pressure • to not deliver undone work (and not even show it)
Everyone **focuses** on	• the work of the Sprint and the goals of the Scrum Team • what’s most important now • the simplest thing that might possibly work
People personally **commit** to	• achieving the goals of the Scrum Team • deliver working software • the Definition of “Done” and quality
Scrum Team members **respect** each other	• to be capable, independent people • to have good intentions and do their best • for diversity in knowledge and skills
The Scrum Team and its stakeholders agree to be **open** about	• all the work and the challenges with performing the work • your work status to create transparency • in sharing and receiving feedback

Scrum Principles

Empirical Process Control	• Knowledge comes from experience and making decisions based on what is known • Transparency, inspection, and adaptation
Self-Organizing Teams	• Brings more personal commitment, accountability, and creativity among team members
Time Boxing	• Helps to focus and manage risks • Enables consistent delivery of business value in every time-boxed Sprint
Shippable Software	• Ensure a potentially useful version of working product is always available • Maximizes opportunities for feedback

“Done” and Definition of Done (DoD)

- “Done” is the state when the Increment becomes releasable
  - Good quality, usable, providing value, etc.
  - At least at the end of each iteration
- In a modern CI/CD world the product should continuously be in the Done state
- DoD = Definition of "Done"
  - Explicit quality criteria
  - Auditable checklist of “Done” criteria
  - Owned and defined by the Development Team

SAMPLE DOD

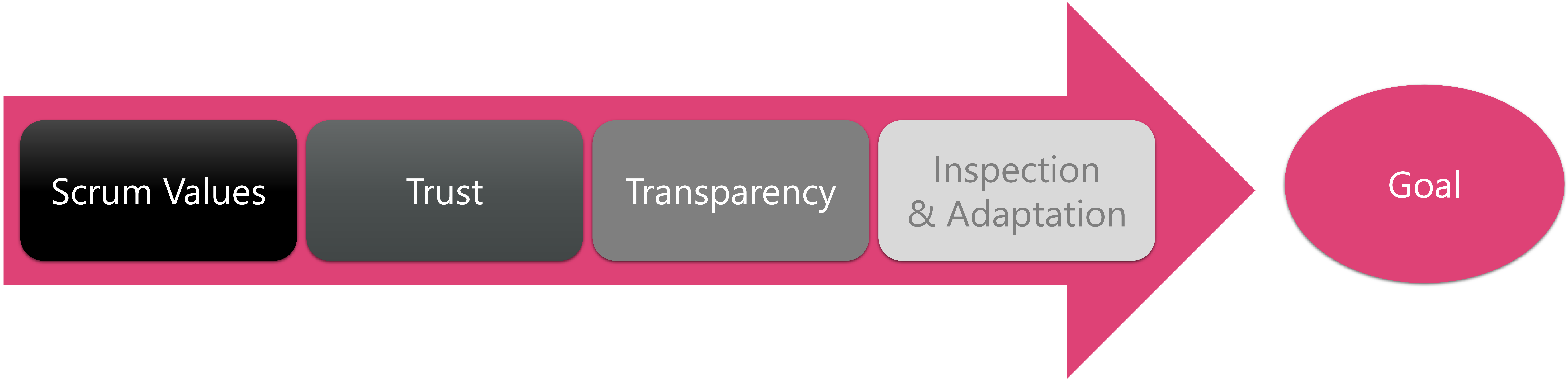
1. *Peer reviewed*
2. *All test cases pass (including security and performance tests)*
3. *No open blocking, critical, high or medium bugs*
4. *Automated tests have been created (unit or integration depending on what is more relevant)*
5. *Conditional coverage is at least 50+% for UI, 60+% for services, and 80+% for utility classes.*
6. *Documentation completed*
7. *Included in the installer*
8. *Reviewed by the Product Owner*
9. *Deployed to the DEMO environment*
10. *Remaining hours for the task set to zero and story/task is closed*

Scrum Helps Organizations Reach Goals

Iterative, value-based incremental delivery

Frequently gathering customer feedback and embracing change

Optimizing predictability and controlling risk



Scrum Values

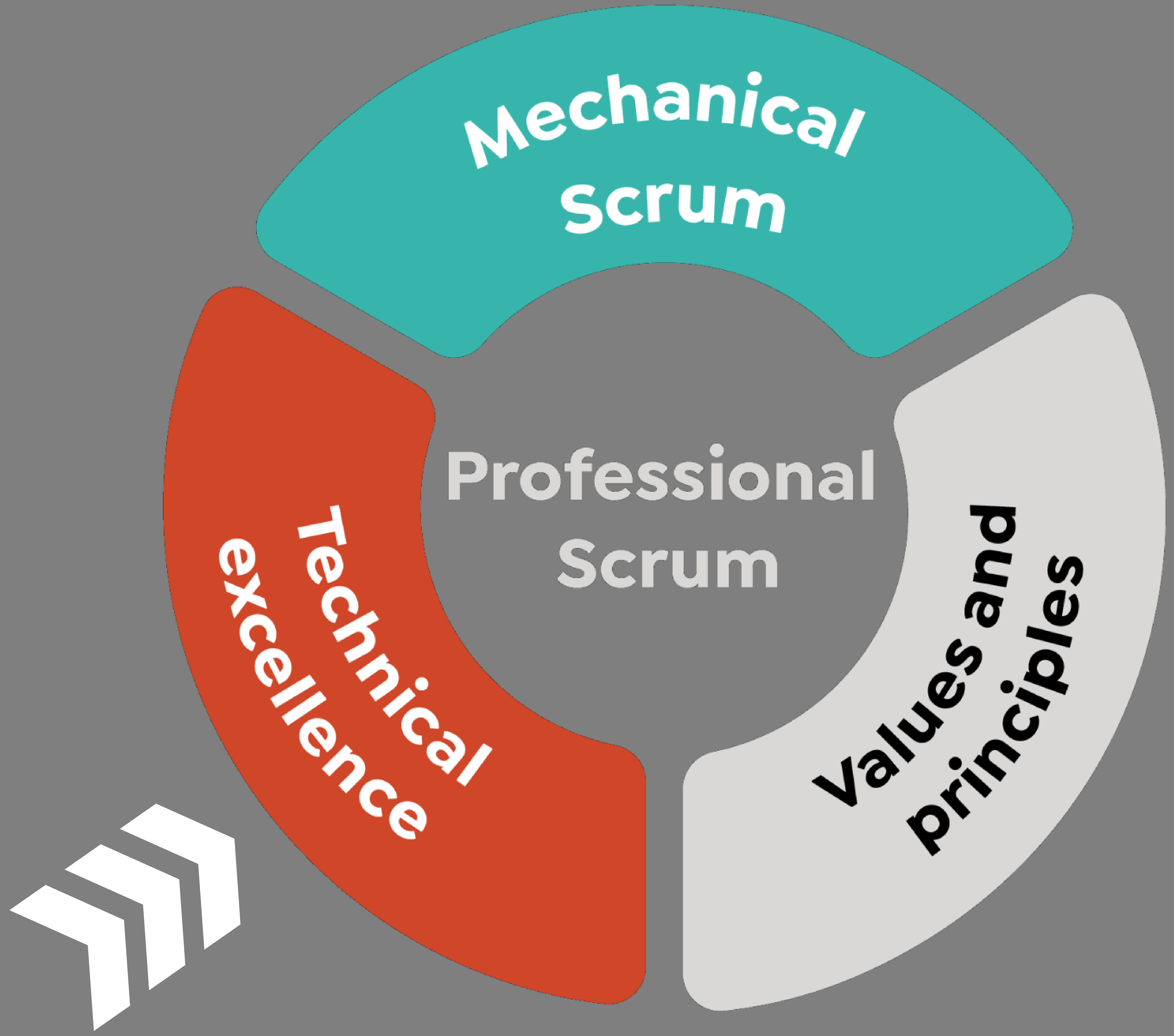
Trust

Transparency

Inspection & Adaptation

Goal

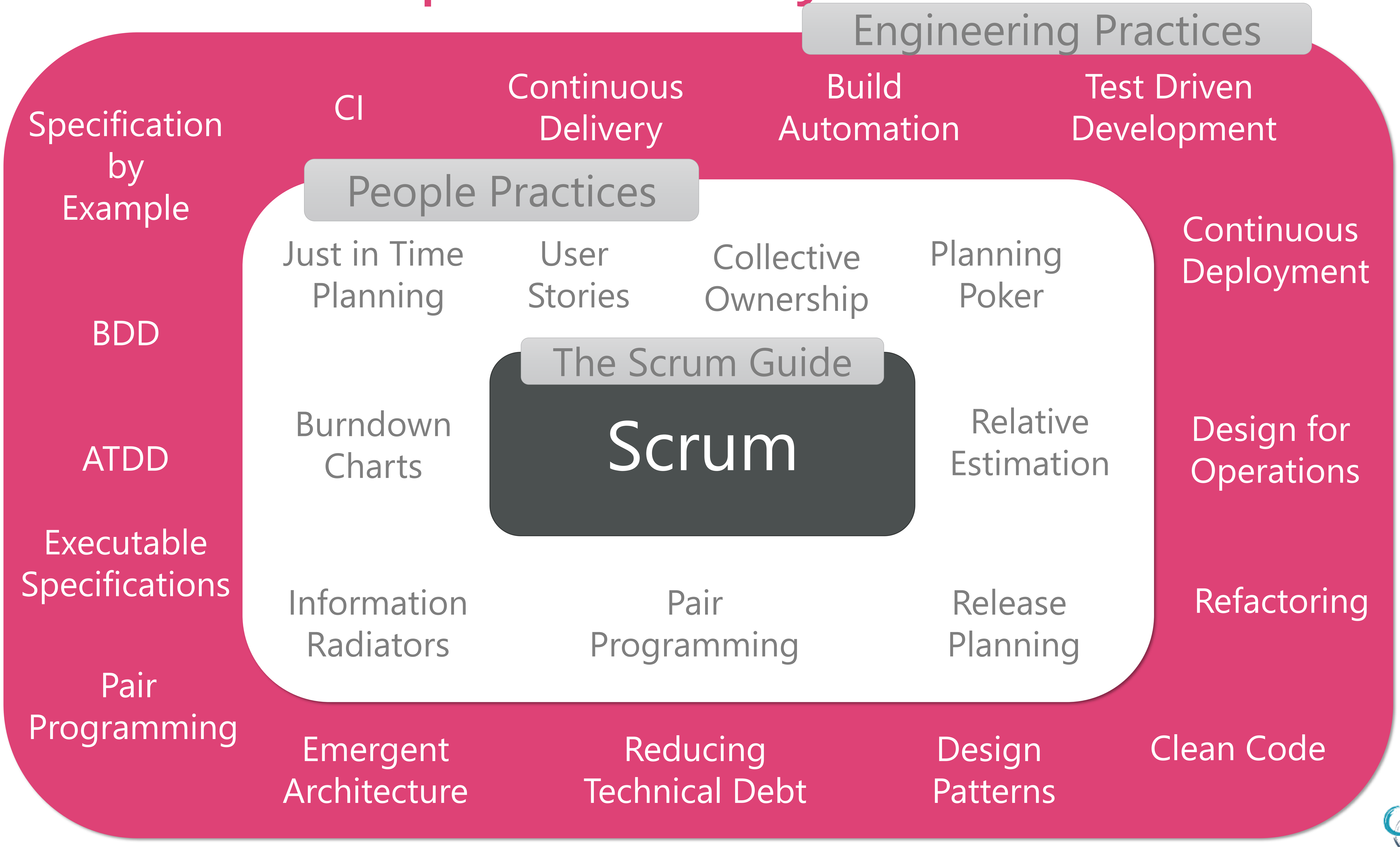
TECHNICAL EXCELLENCE



#ntk19

Practices Complementary to Scrum

#ntk19



Engineering Practices

Specification by Example

CI

Continuous Delivery

Build Automation

Test Driven Development

Continuous Deployment

BDD

ATDD

Executable Specifications

Pair Programming

Design for Operations

Refactoring

Emergent Architecture

Reducing Technical Debt

Design Patterns

Clean Code

People Practices

Just in Time Planning

User Stories

Collective Ownership

Planning Poker

Burndown Charts

Relative Estimation

Information Radiators

Pair Programming

Release Planning

The Scrum Guide

Scrum

Scrum.org

What is DevOps?

#ntk19

DevOps is the union of ***people****,* ***process****, and* ***products*** *to enable continuous delivery of value to your end users.*

Most Impactful DevOps Practices:

Automation and Speed

Automated delivery pipeline

Quality Fast

Automated tests for automated DOD

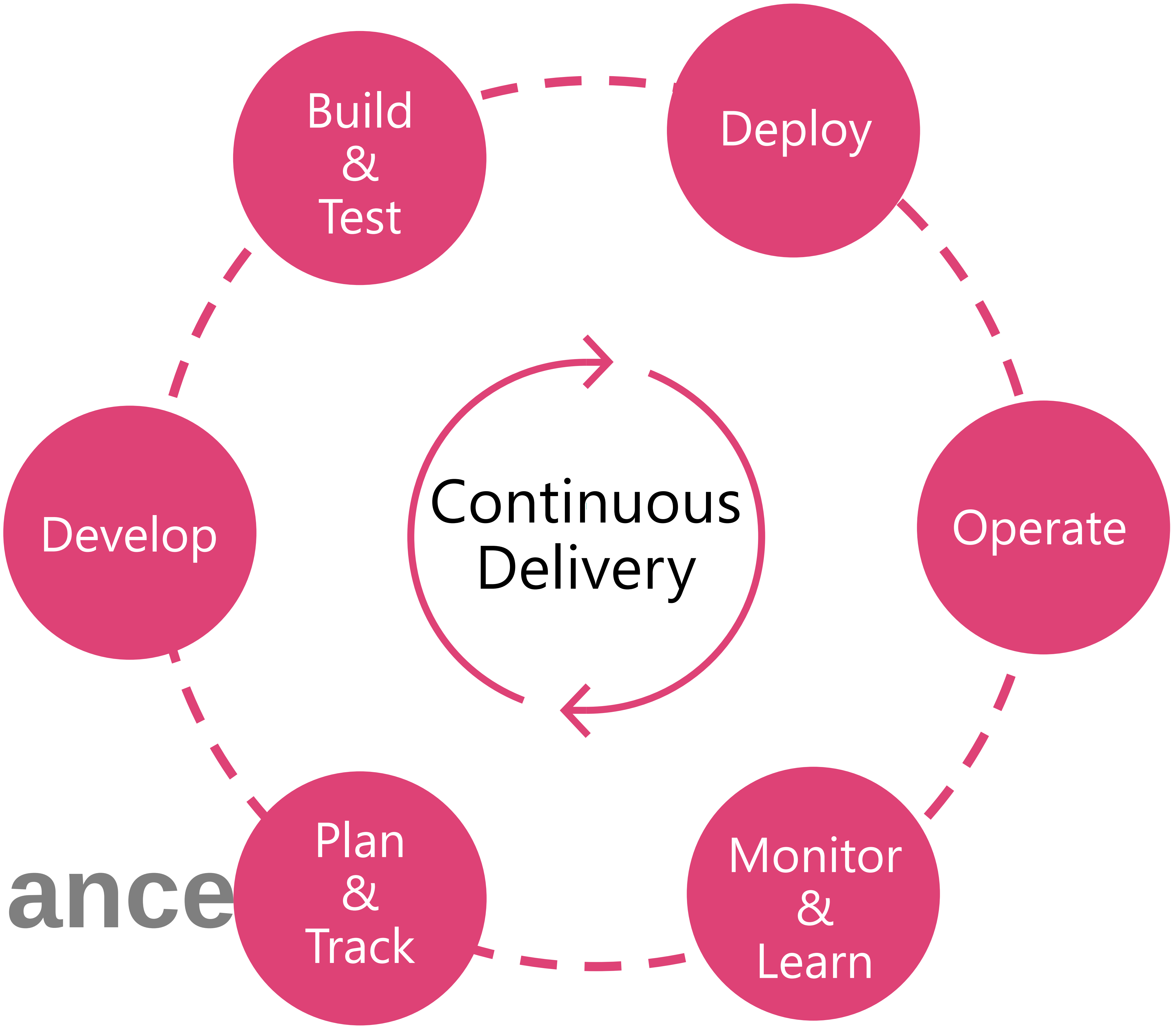
Test Shift Left

Code Analysis

Code Reviews (Pull Requests)

Insights – State, Behavior, Performance

Telemetry & feedback gathering



#ntk19

Meet Azure DevOps!

End-to-end DevOps toolchain consisting of integrated services for sharing code, tracking work, and shipping high quality solutions

Azure Boards

Deliver value to your users faster using proven **agile tools to plan, track, and discuss work across** your teams.

Azure Pipelines

Build, test, and deploy with **CI/CD** that works with **any language, platform, and cloud**. Connect to GitHub or any other Git provider and deploy continuously.

Azure Repos

Get unlimited, cloud-hosted **private Git repos** and collaborate to build better code with **pull requests** and advanced file management.

Azure Test Plans

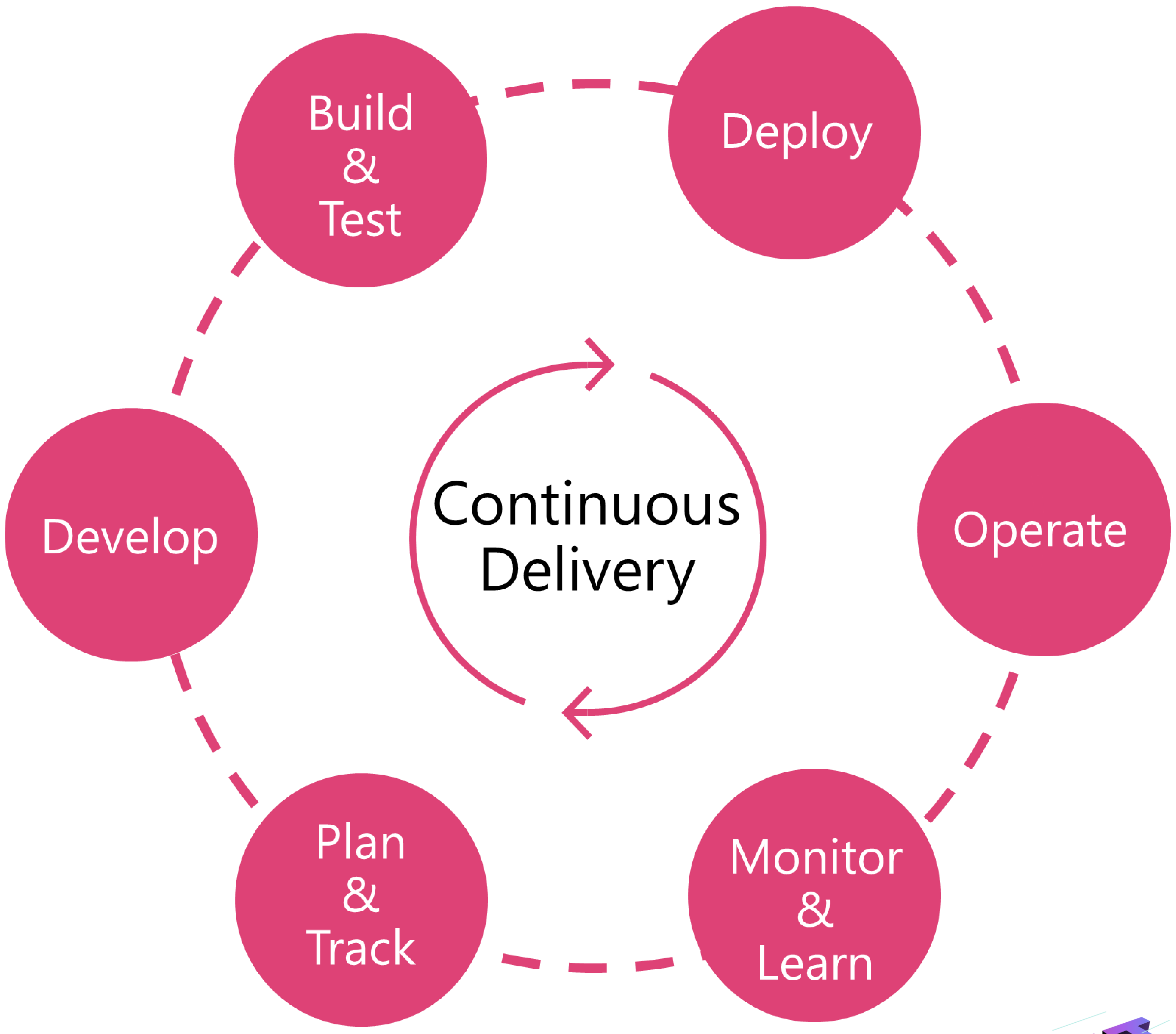
Test and ship with confidence using **manual and exploratory testing tools**.

Azure Artifacts

Create, host, and share packages with your team, and add artifacts to your CI/CD pipelines with a single click.

CORE DEVOPS PRACTICE

AUTOMATION AND SPEED



Automated Delivery Pipeline, CI, CD

#ntk19

Fully automated delivery through a number of environments all the way to production

Based on build and release management

Includes various types of automated testing

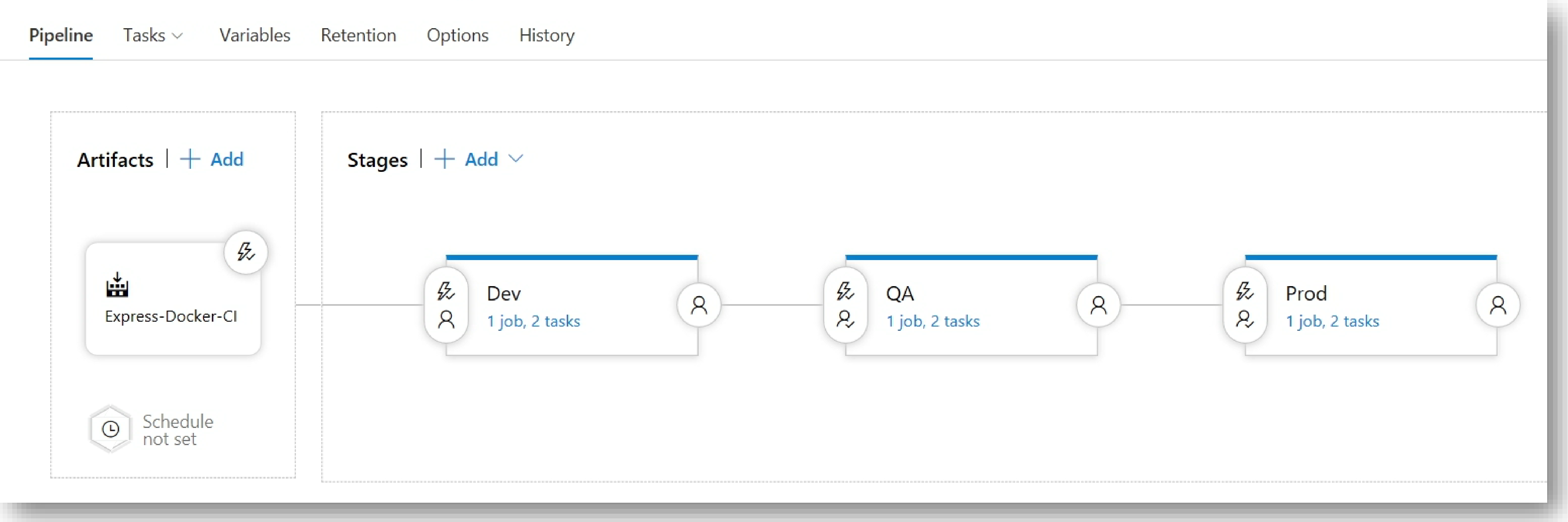
Control of automated delivery:

1. Manual Approvals
2. Automated Release Gates

   Manual test results, Bugs

   Stakeholder Feedback

   Integration with other systems (incident management, etc.)

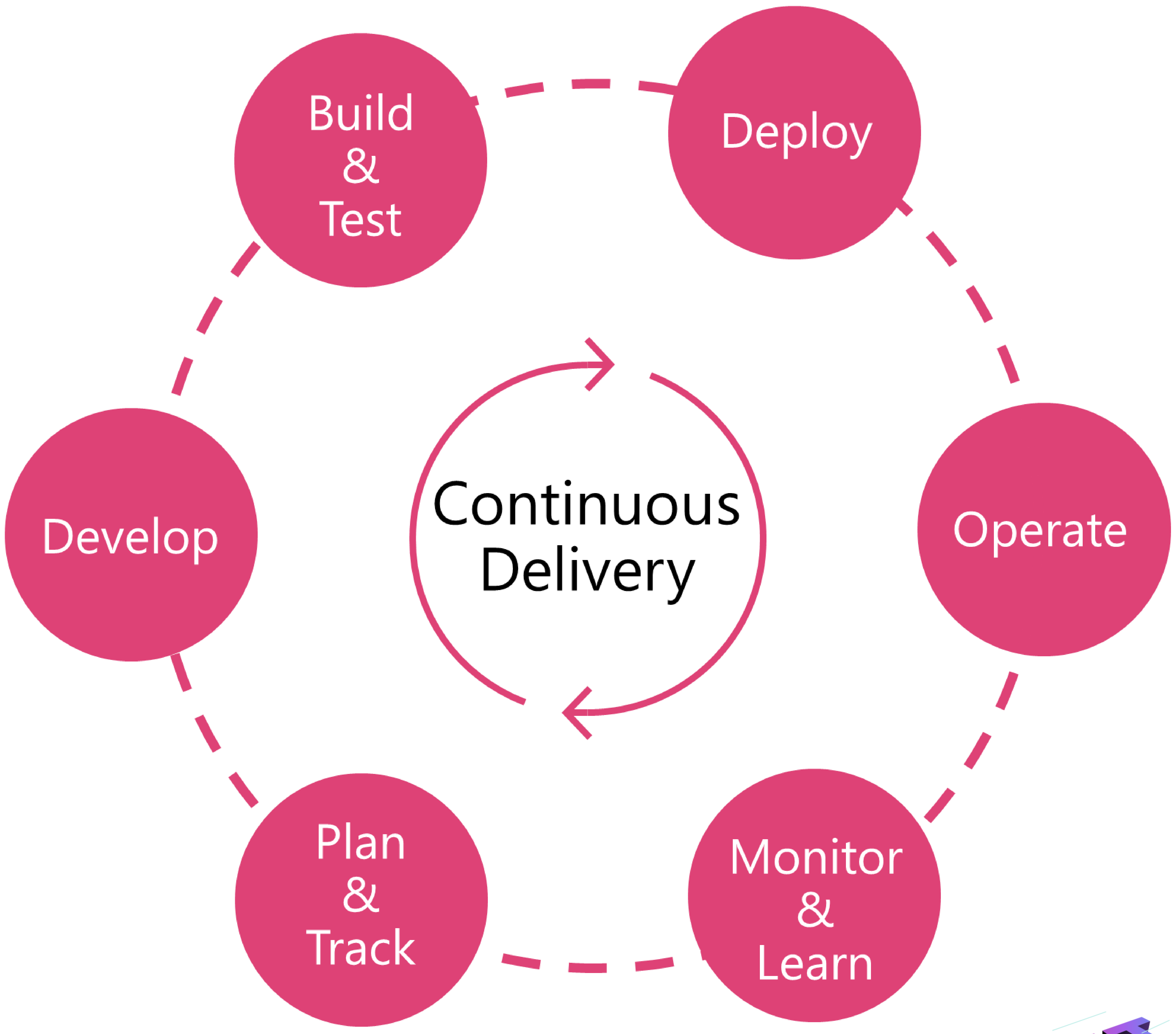


DEMO

AZURE PIPELINES
BUILD AND RELEASE
RELEASE GATES
MANUAL APPROVALS

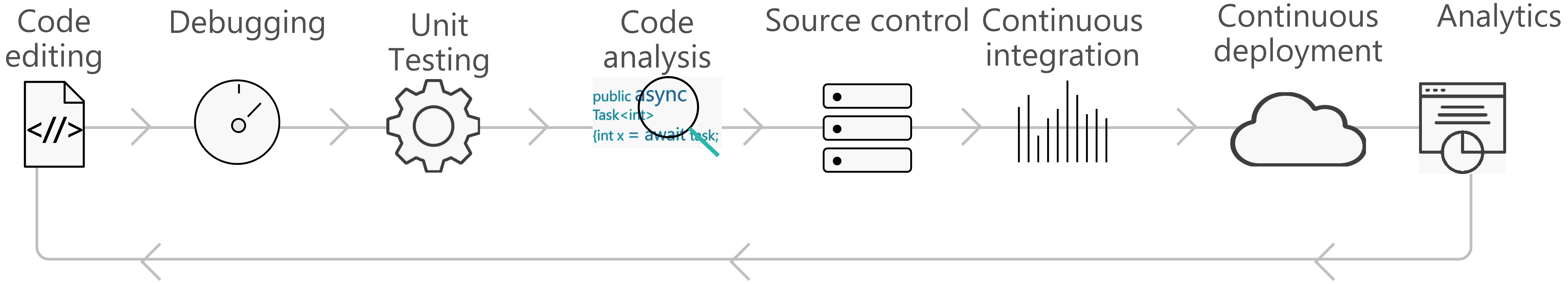
CORE DEVOPS PRACTICE

QUALITY FAST



#ntk19

Verifying Quality in Different Parts of the Cycle



Code editing

Debugging

Unit Testing

Code analysis

Source control

Continuous integration

Continuous deployment

Analytics

1. Test Shift Left – Start evaluating DoD as early as possible and then continue throughout the pipeline
2. Automate your delivery pipeline
3. We need some manual steps to verify quality too!

#ntk19

Test Shift Left – Automated Tests – Early DoD

- Automated testing - Safety net for rapid development!
  - Acceptance Tests → Regression Tests
    - Unit Tests, Functional or Integration Tests, Stress or Load Tests
    - Performance, Security, Usability etc.
- Test Shift Left = Write tests at the „lowest” level possible
  - More unit test like, less UI based tests
    - Replace fragile UI based (functional and integration) tests with robust unit test based (functional and integration) tests
  - Usually requires refactoring of the architecture
  - Tests become significantly faster, more reliable and can be executed everywhere
- Bugs are discovered earlier in the development cycle
  - Quick feedback on commits further reduces context switching

Automated Tests

Automating DoD Testing in the Delivery Pipeline

Continuous Integration (CI) with automated testing

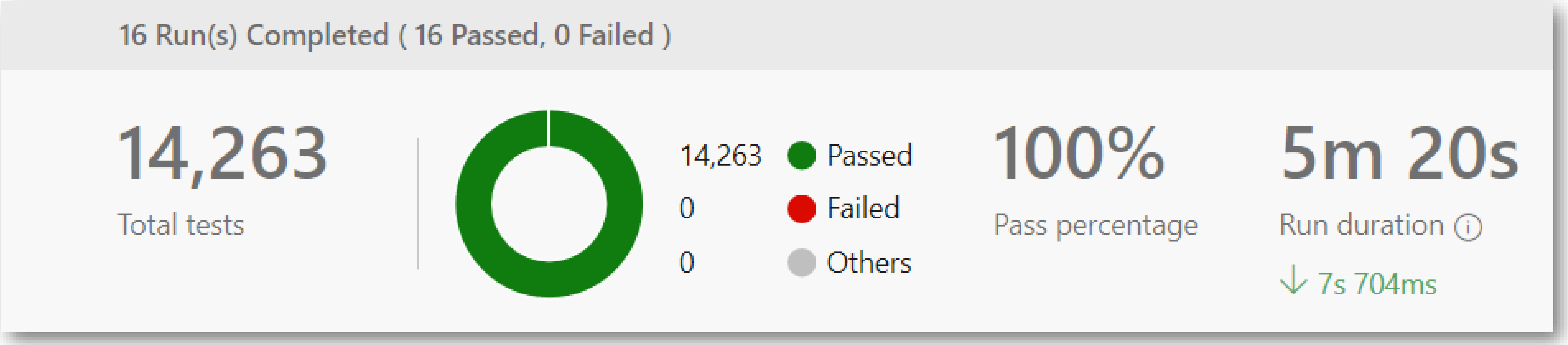
Speed is key: fast builds, fast test results, fast feedback

Always stay very close to „ready to deploy to production”

Fast tests can be executed everywhere

- Developer’s workstation
- Build servers
- Production environments

VS Code
Build
Pipeline



16 Run(s) Completed (16 Passed, 0 Failed)

14,263
Total tests

14,263	🟩 Passed
0	🟥 Failed
0	● Others

100%
Pass percentage

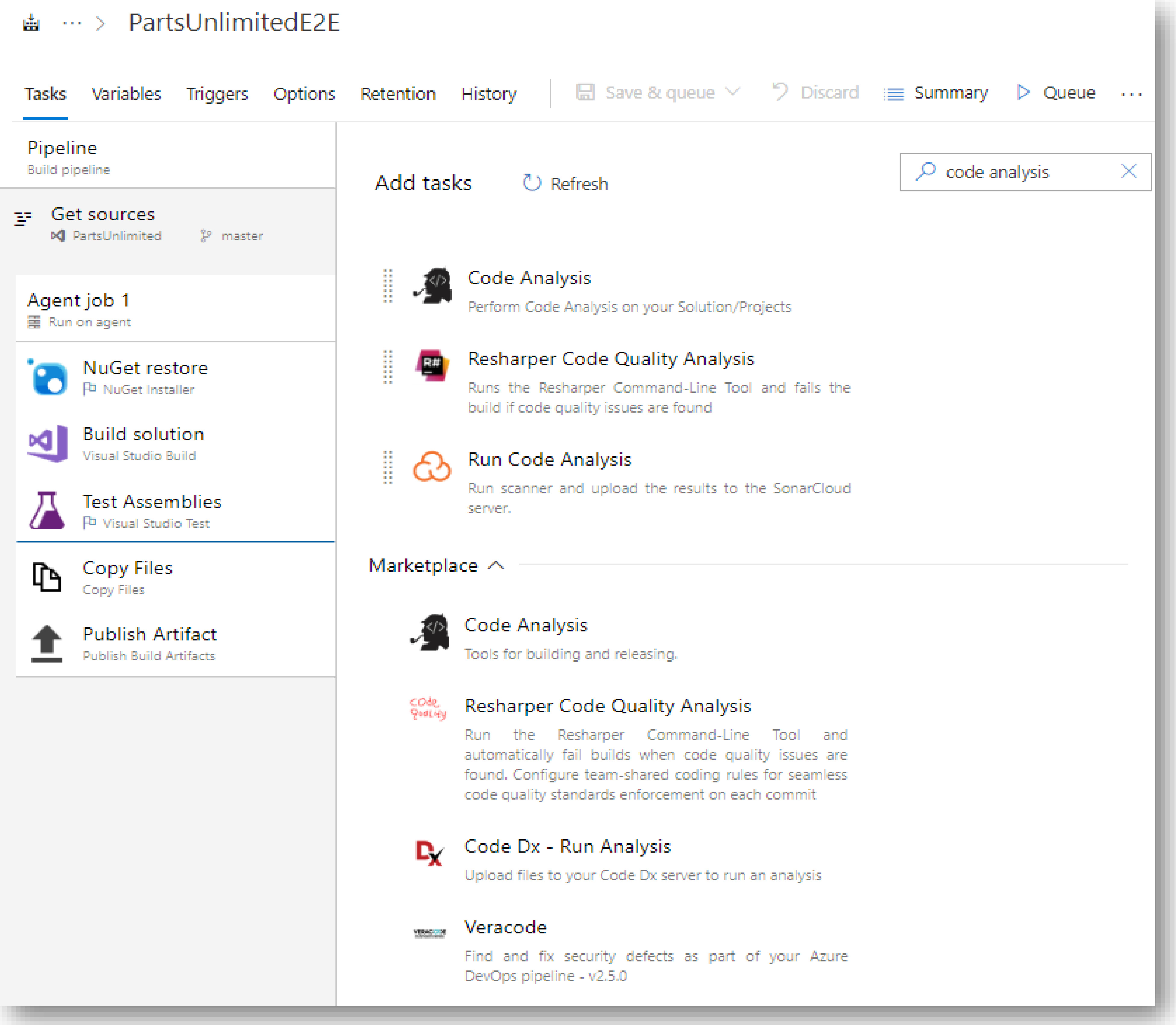
5m 20s
Run duration ⓘ
↓ 7s 704ms

https://dev.azure.com/vscode/VSCode/_build

Code Analysis

Automating DoD Criteria Verification in the Delivery Pipeline

- In the Build part of the pipeline
  - Free tools: R#, MS Code Analysis
  - Sonar Cloud (Free for OS): Static analysis, Bugs, Vulnerabilities, Code smells, Code coverage, Code duplication, etc.



Pipeline
Build pipeline

Get sources
PartsUnlimited master

Agent job 1
Run on agent

NuGet restore
NuGet Installer

Build solution
Visual Studio Build

Test Assemblies
Visual Studio Test

Copy Files
Copy Files

Publish Artifact
Publish Build Artifacts

Code Analysis
Perform Code Analysis on your Solution/Projects

Resharper Code Quality Analysis
Runs the Resharper Command-Line Tool and fails the build if code quality issues are found

Run Code Analysis
Run scanner and upload the results to the SonarCloud server.

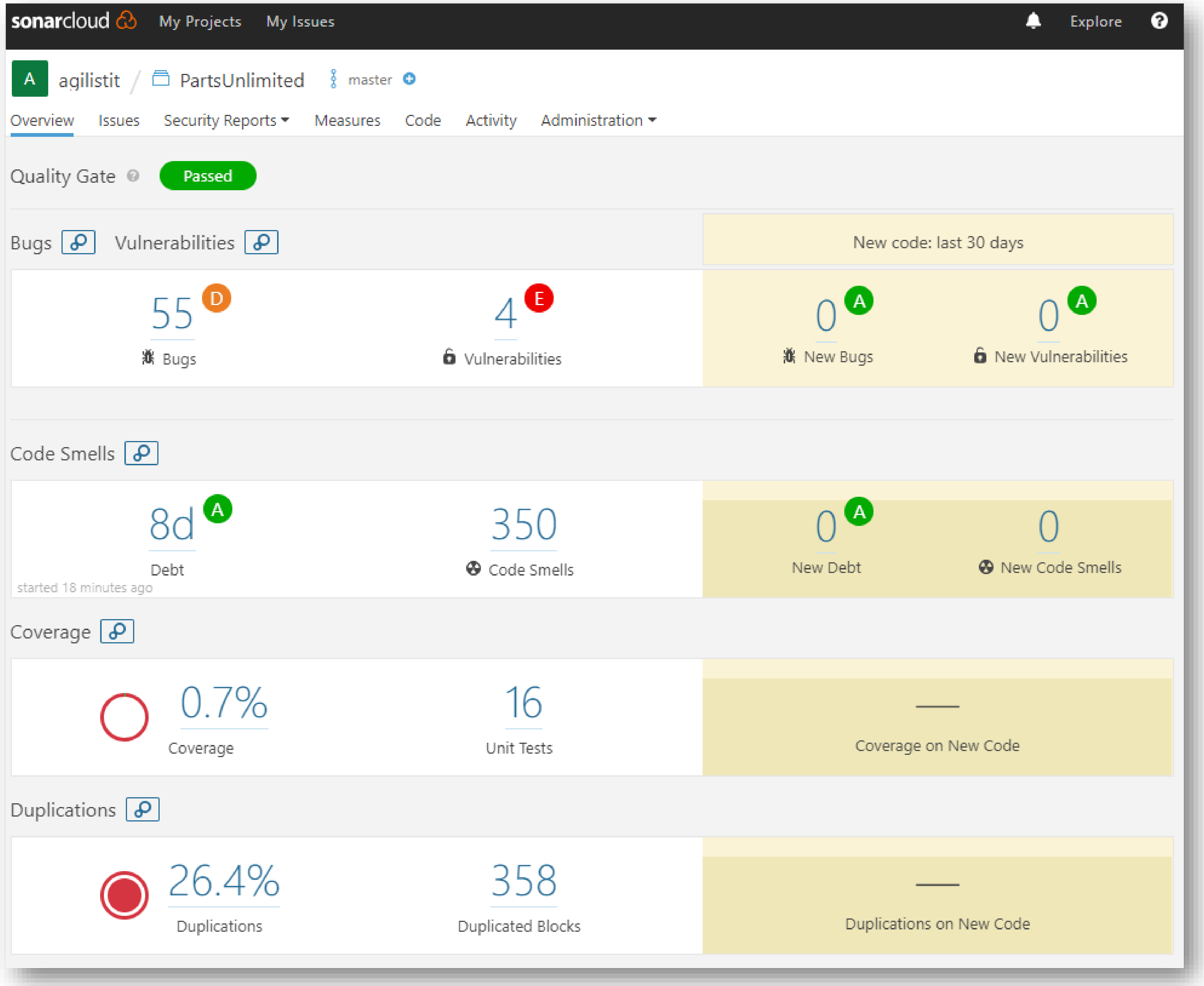
Marketplace

Code Analysis
Tools for building and releasing.

Resharper Code Quality Analysis
Run the Resharper Command-Line Tool and automatically fail builds when code quality issues are found. Configure team-shared coding rules for seamless code quality standards enforcement on each commit

Code Dx - Run Analysis
Upload files to your Code Dx server to run an analysis

Veracode
Find and fix security defects as part of your Azure DevOps pipeline - v2.5.0



Quality Gate Passed

Bugs Vulnerabilities | New code: last 30 days

55 D Bugs | 4 E Vulnerabilities | 0 A New Bugs | 0 A New Vulnerabilities

Code Smells

8d A Debt | 350 Code Smells | 0 A New Debt | 0 New Code Smells

started 18 minutes ago

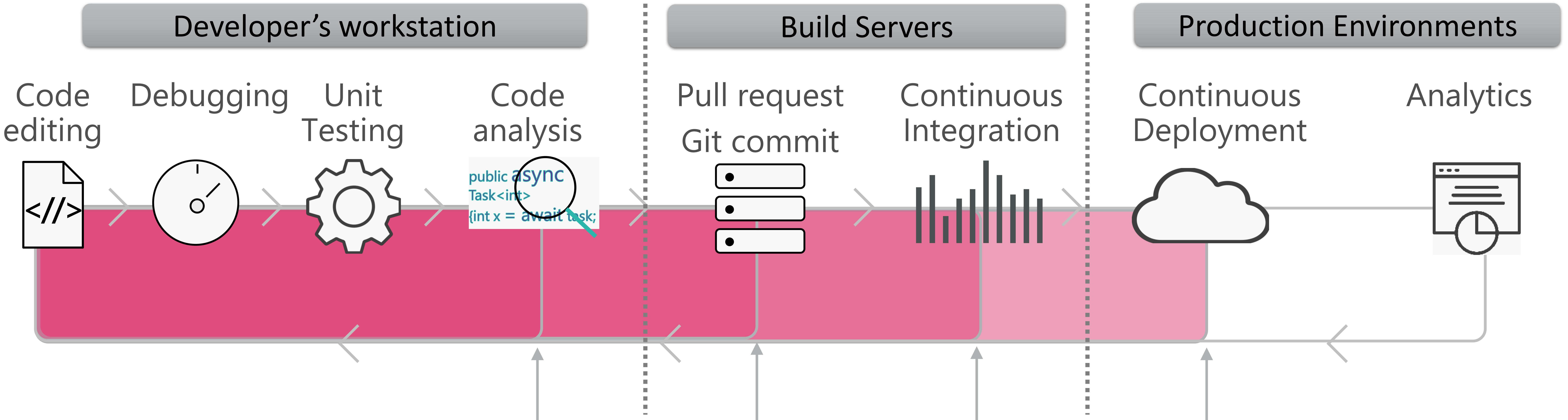
Coverage

0.7% Coverage | 16 Unit Tests | — Coverage on New Code

Duplications

26.4% Duplications | 358 Duplicated Blocks | — Duplications on New Code

Automated DoD Evaluation in Different Parts of the Cycle



DoD in the edit/debug/build cycle
Edit and Continue
Live Unit Testing
Live Code Analysis

Automated Release with further DoD criteria verification (additional integration tests)

Automated Build with automated analysis and tests

Branching policies: Automated Build with automated analysis and tests

Not Everything Should Be Automated

- Code Reviews - *Before code reaches master*
  - Git Pull Request combined with Branching Policies
  - Ensure Quality
  - Promote shared team code ownership
- Manual specified tests – *Manual approvals embedded in pipeline*
  - Acceptance tests defined as steps and expected results
  - Cover all usage scenarios
  - Explicit definition of scope for user stories
  - Automated during development (if that is a part of DoD)

DEMO

- AUTOMATED DOD IN AZURE PIPELINE – BUILD
- CODE ANALYSIS & SONAR CLOUD
- GIT PULL REQUEST & BRANCHING POLICIES
- TEST SHIFT LEFT
(OS PROJECTS EXAMPLES)

Flaky (Non-Deterministic) Tests Problem

Flaky tests – may pass or fail without any change in the code-under-test

Loss of productivity
Debugging non-deterministic test failure leads to loss of developer productivity
Low Confidence in Quality Signal
Once you know that some tests are flaky, your stop trusting test results
Bad Code left unnoticed
Test failure ignored as flaky (incorrectly) causing failures to reach customers

Some reasons for flakiness

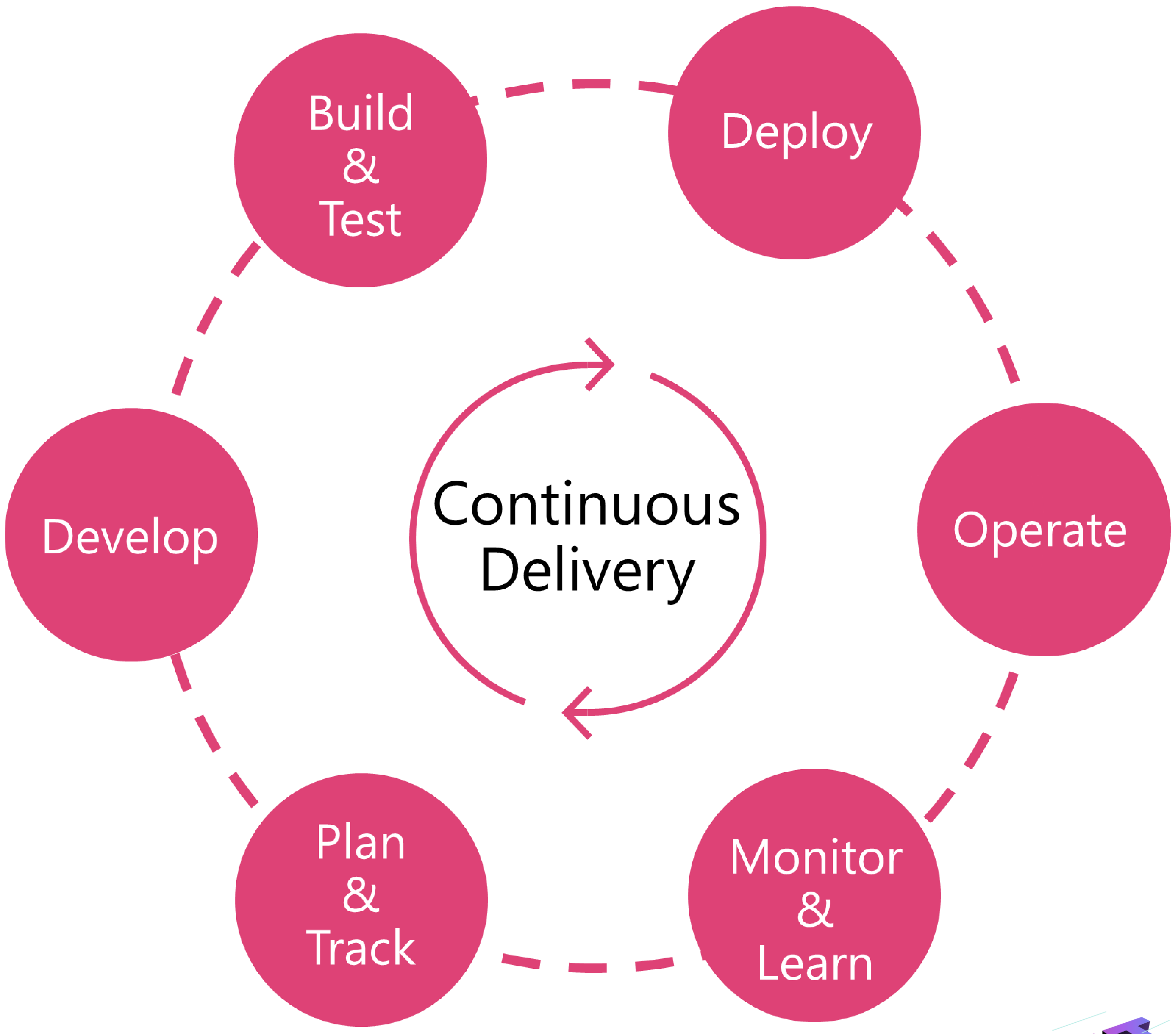
- Poor test isolation
- Flaky external services
- Timeouts not long enough
- Improper test setup or teardown

DEMO

FLAKY TESTS

CORE DEVOPS PRACTICE

INSIGHTS



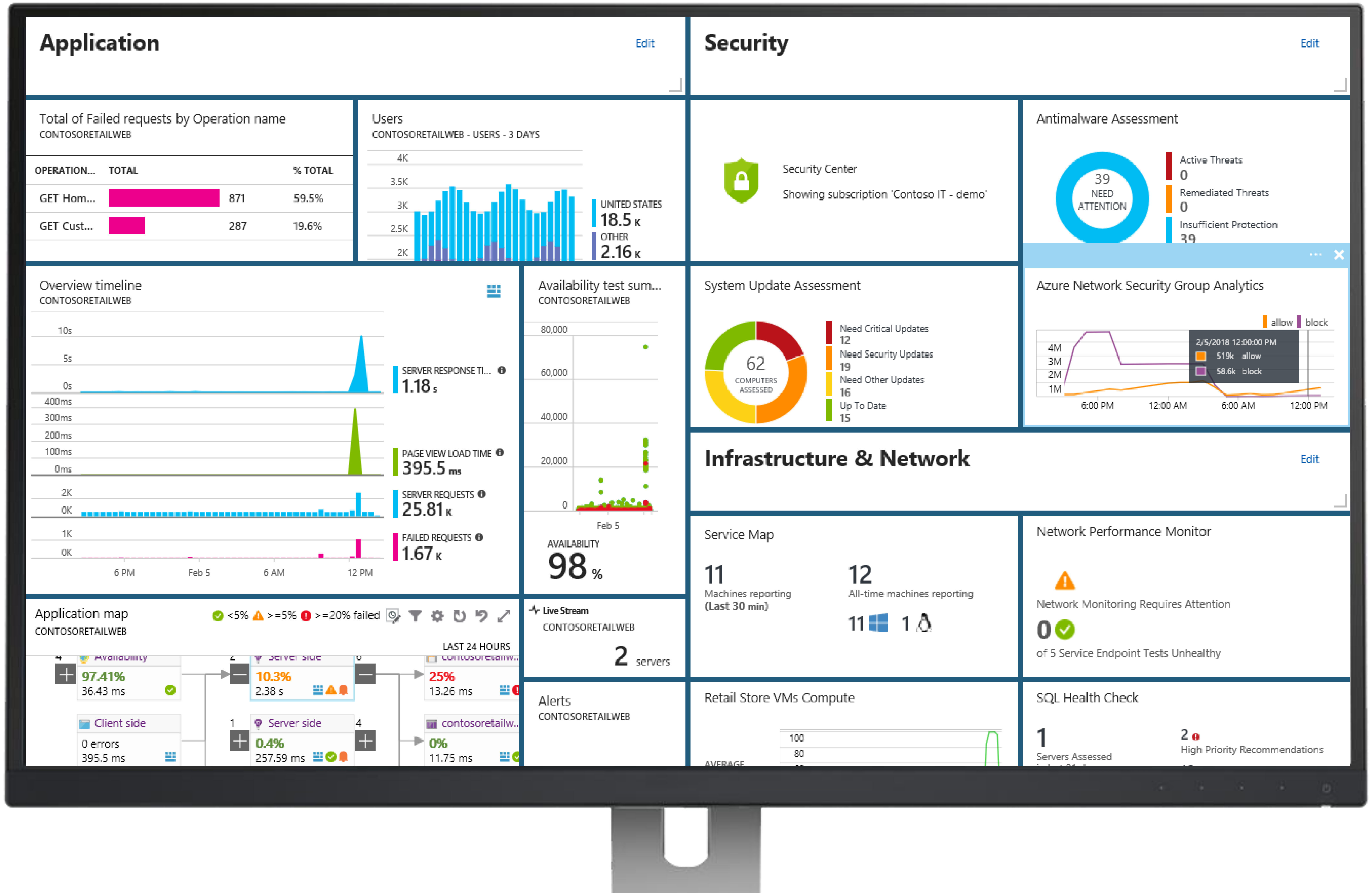
#ntk19

Production Monitoring

#ntk19

Live Monitoring of Software in Production Environment

- Monitor and Track Usage Patterns, Application Behavior, Performance, Availability and Scale
- Preemptively recognize, diagnose and resolve problems before users are affected
- Visualize data in intuitive and customizable dashboards
- Separate the signal from noise and accelerate root-cause analysis



Application

Total of Failed requests by Operation name
CONTOSORETAILWEB

OPERATION...	TOTAL	% TOTAL
GET Hom...	871	59.5%
GET Cust...	287	19.6%

Telemetry and Feedback Gathering

#ntk19

Outside-in monitoring

URL pings and web tests from multiple global points of presence

Observed user behavior

How is the application being used?

Developer traces and events

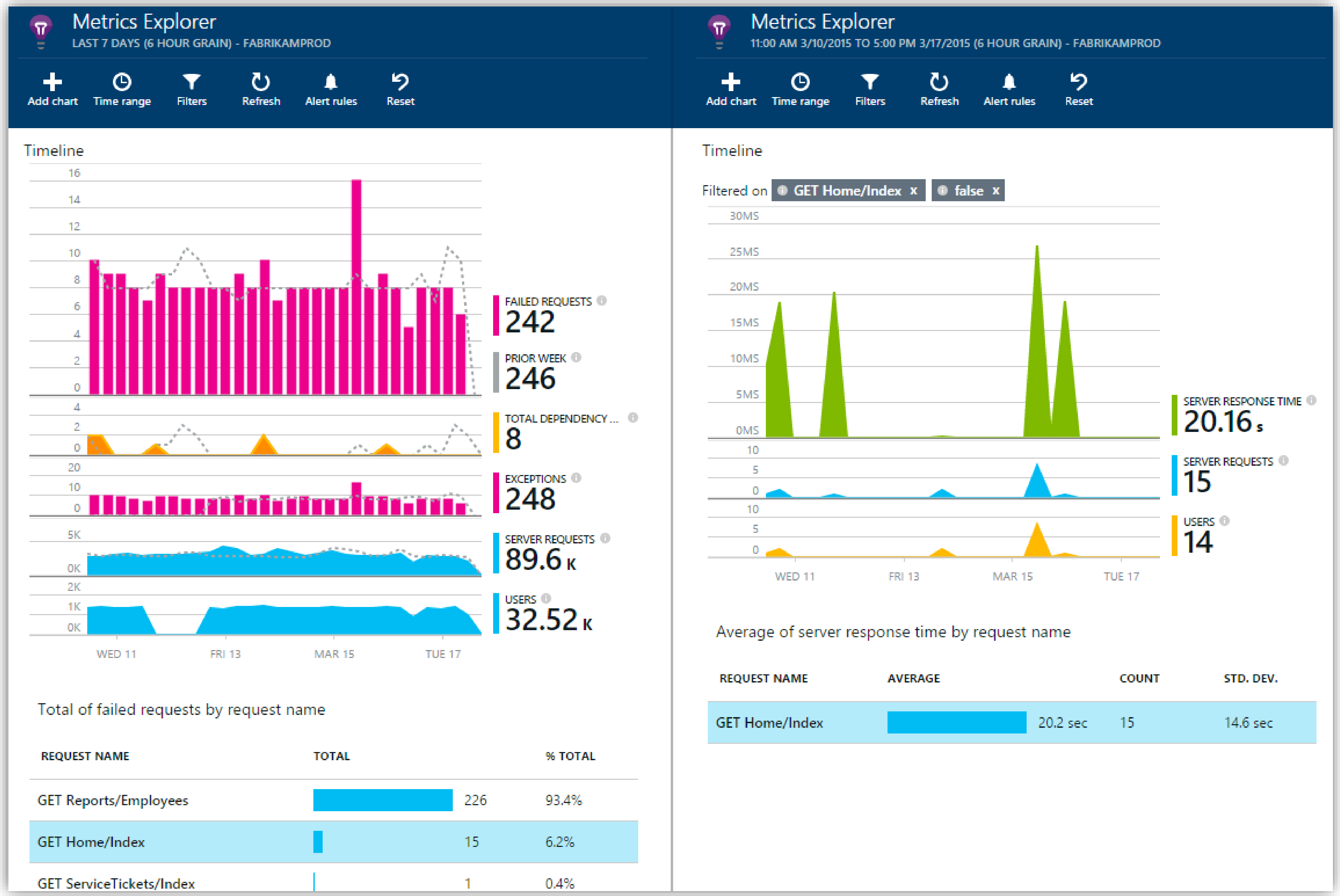
Selected events, exceptions, logs

Observed application behavior

Service dependencies, queries, response time, exceptions, logs, etc.

Infrastructure performance

System performance counters



Metrics Explorer

LAST 7 DAYS (6 HOUR GRAIN) - FABRIKAMPROD

Timeline

FAILED REQUESTS 242

PRIOR WEEK 246

TOTAL DEPENDENCY ... 8

EXCEPTIONS 248

SERVER REQUESTS 89.6 K

USERS 32.52 K

Total of failed requests by request name

REQUEST NAME	TOTAL	% TOTAL
GET Reports/Employees	226	93.4%
GET Home/Index	15	6.2%
GET ServiceTickets/Index	1	0.4%

Metrics Explorer

11:00 AM 3/10/2015 TO 5:00 PM 3/17/2015 (6 HOUR GRAIN) - FABRIKAMPROD

Timeline

Filtered on GET Home/Index x false x

SERVER RESPONSE TIME 20.16 s

SERVER REQUESTS 15

USERS 14

Average of server response time by request name

REQUEST NAME	AVERAGE	COUNT	STD. DEV.
GET Home/Index	20.2 sec	15	14.6 sec

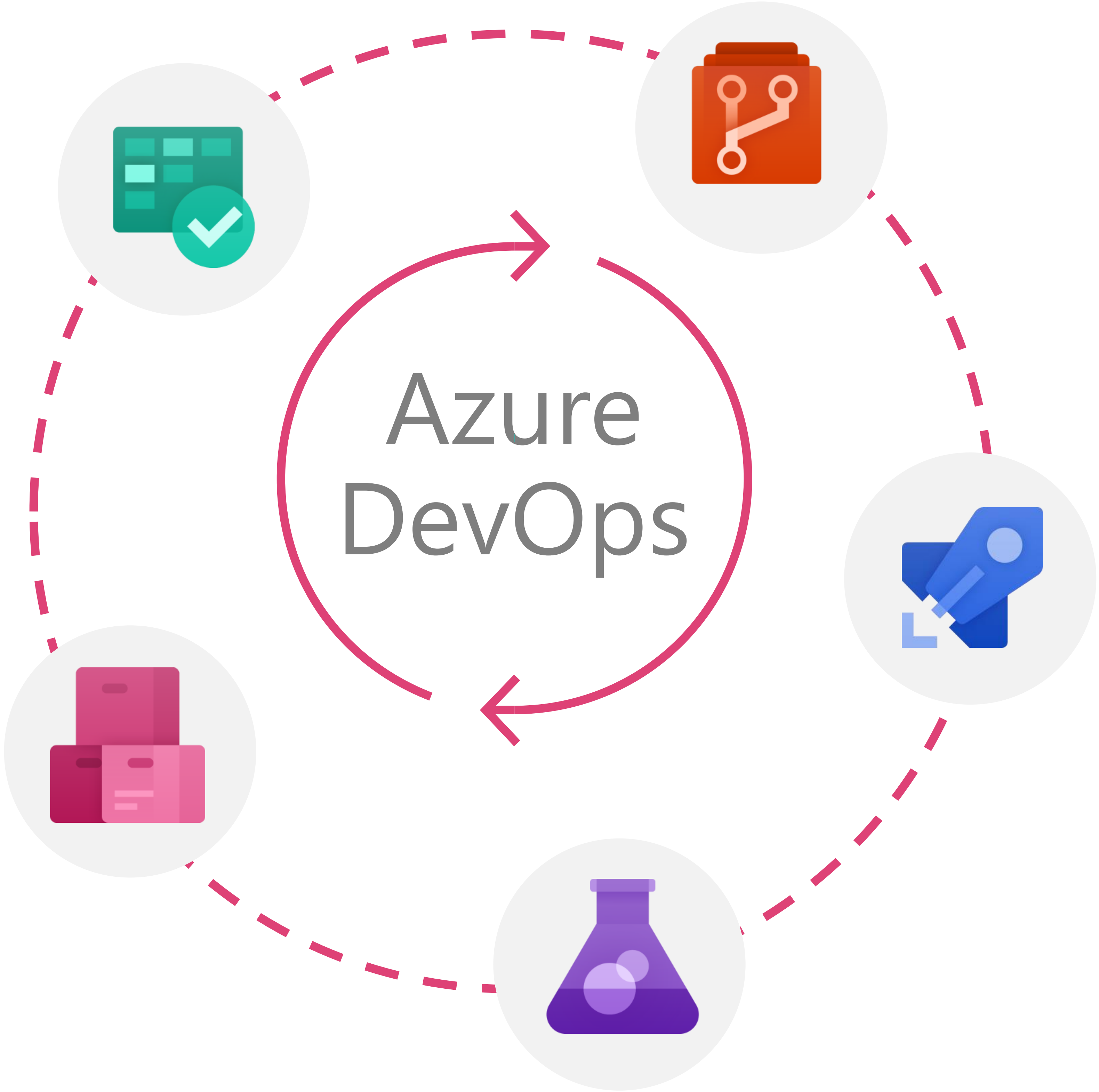
#ntk19

DEMO

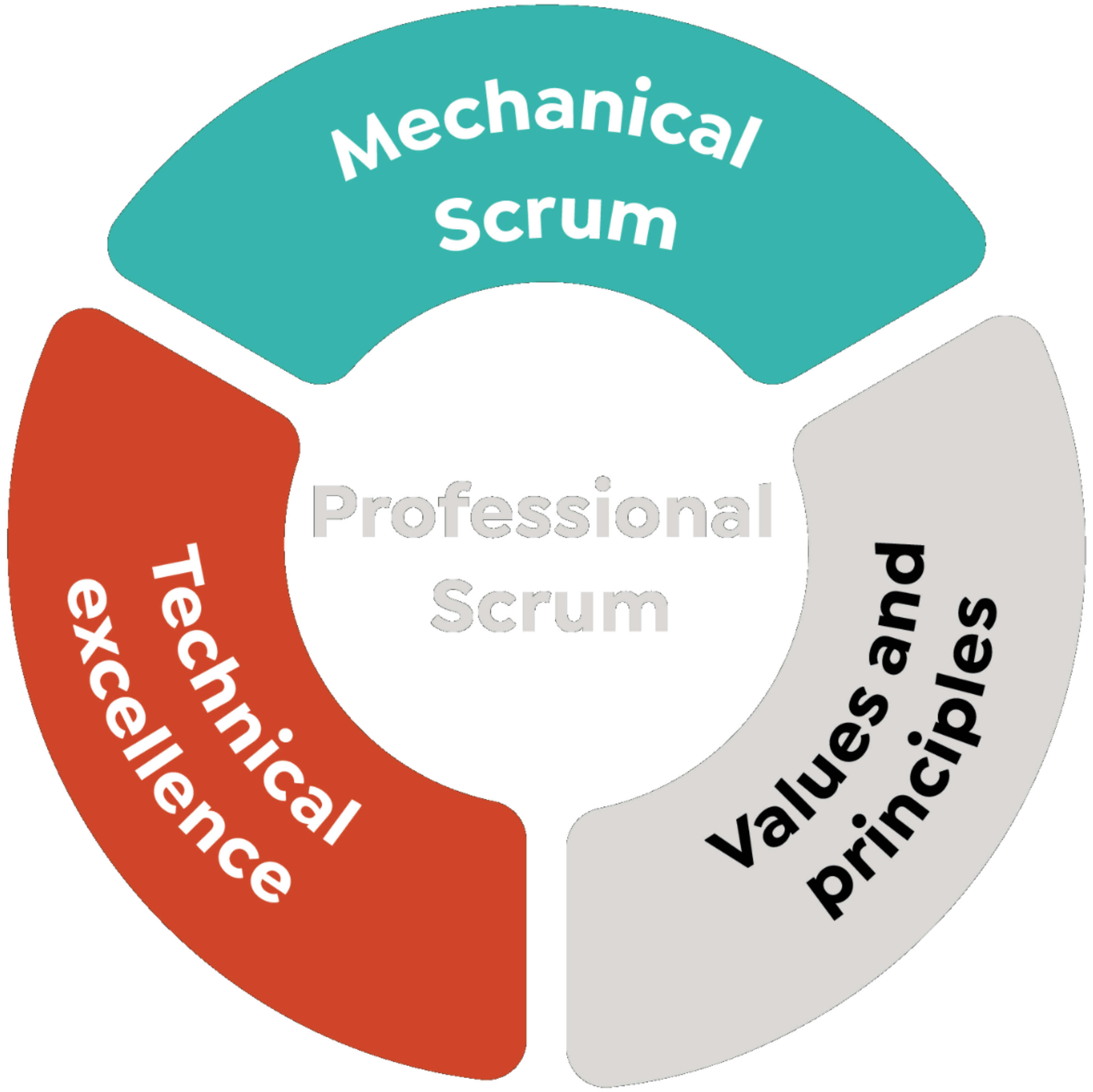
APPLICATION INSIGHTS

High Performance DevOps Enables Professional Scrum

#ntk19



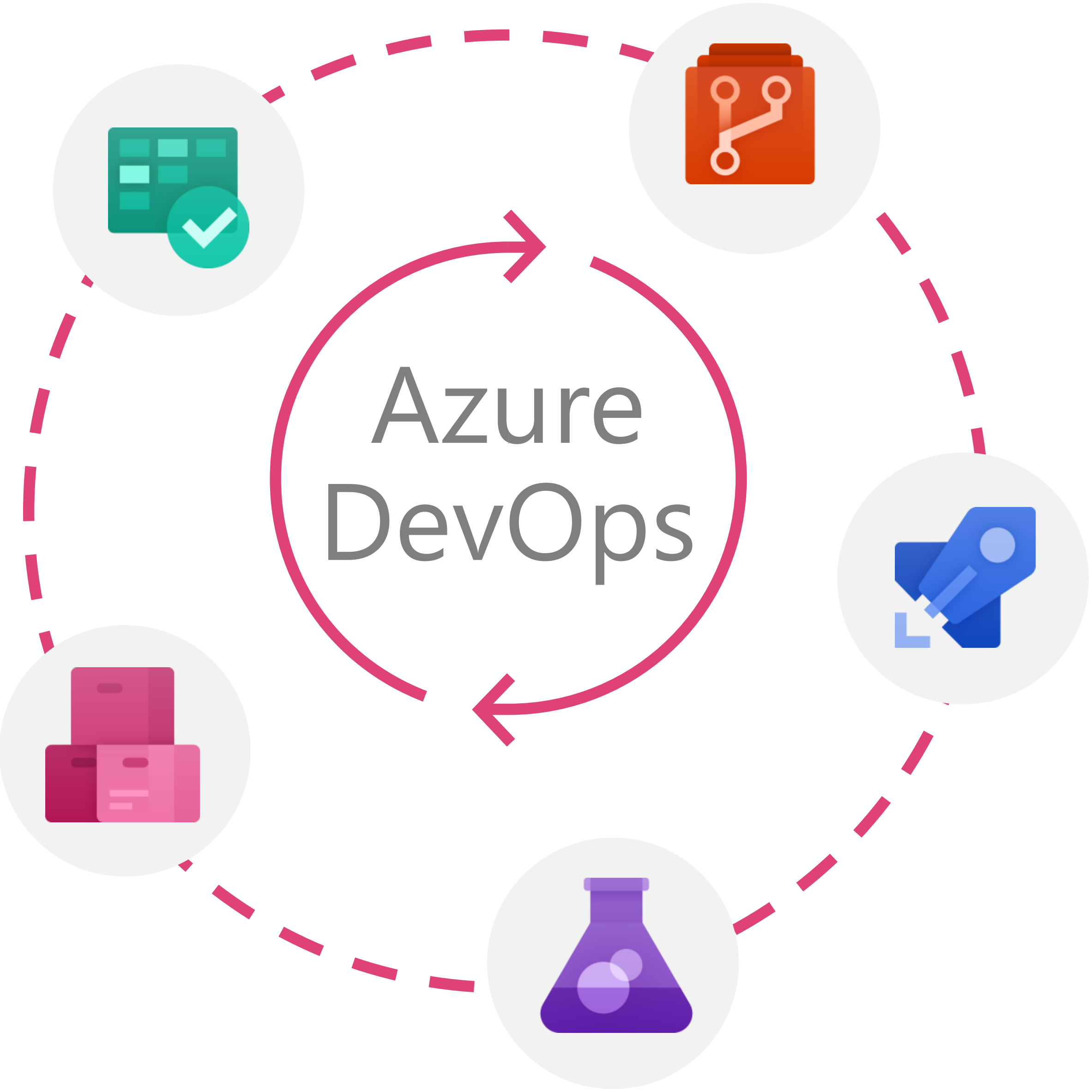
&



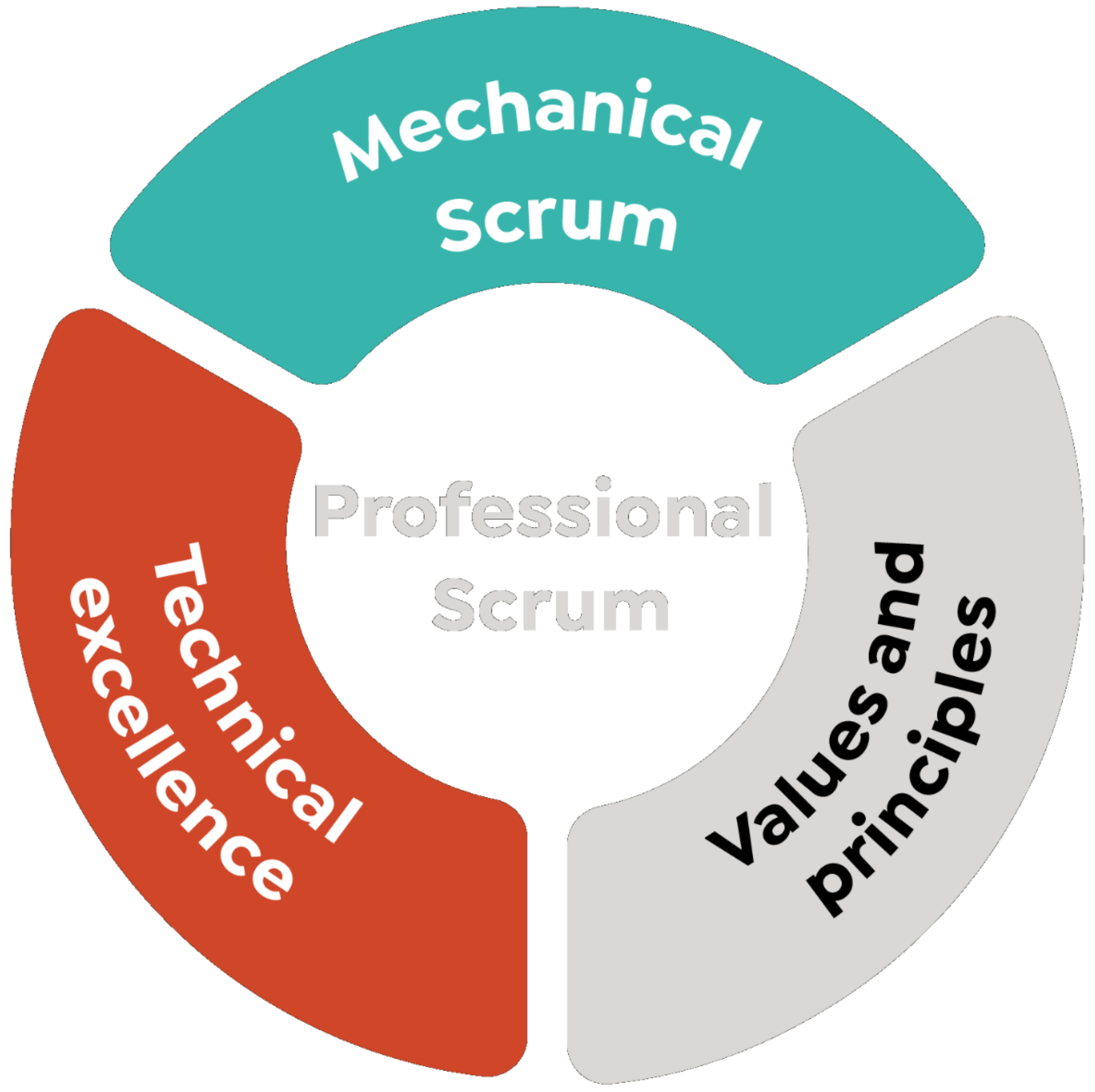
End-to-end DevOps toolchain consisting of integrated services for sharing code, tracking work, and shipping high quality solutions

Scrum instance implementing Scrum mechanics, Scrum values and principles and technical excellence

Questions ?



&



Ognjen Bajić, obajic@ agilist.hr
Ana Roje Ivančić, arojeivancic@ agilist.hr
Agilist IT, Zagreb, Croatia

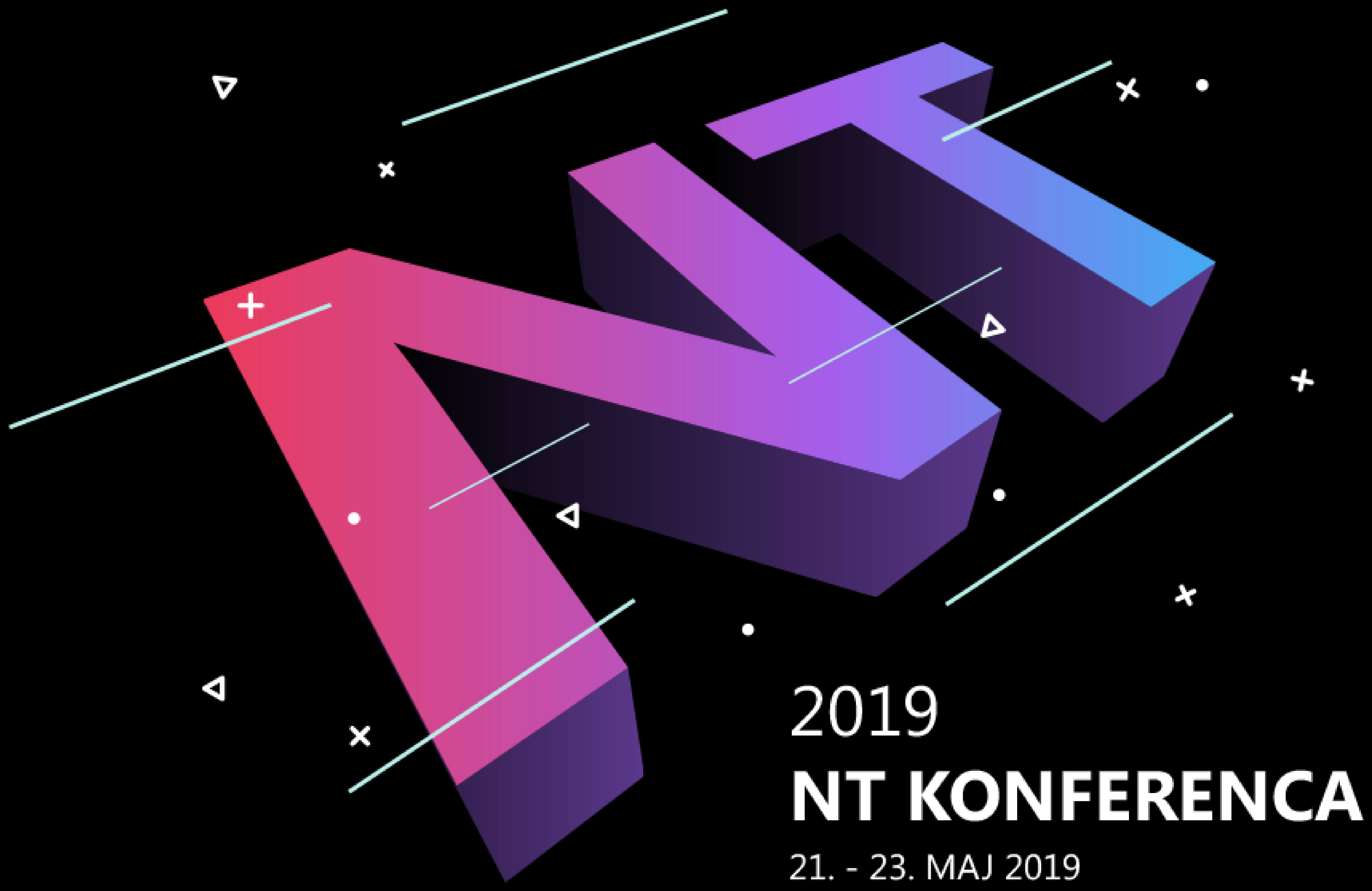
Microsoft® Most Valuable Professional

PST Professional Scrum Trainer Scrum.org

http://aglst.com/ScrumTraining_PSF

http://aglst.com/ScrumTraining_PSD

trainings@agilist.hr



2019

NT KONFERENCA

21. - 23. MAJ 2019

#ntk19