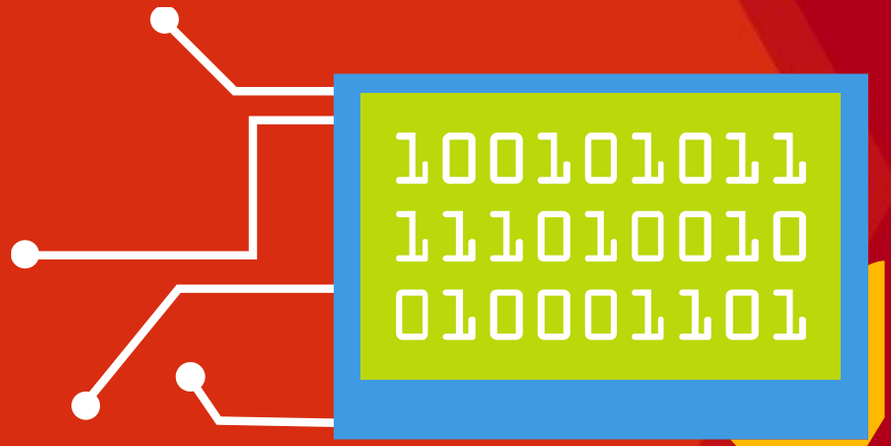
KONFERENCA

Microsoft

Patterns and Practices of Best DevOps Organizations



Ognjen Bajić, VS ALM MVP
Ana Roje Ivančić, VS ALM MVP
Ekobit

TEHNOLOGIJA

Speakers

Working with VS ALM tools since 2004.

WinDays 2005 preconf day on VSTS

Worked as Dev, PM, Test, RM, SM, PO…

VS ALM MVPs

Agenda

Overview of DevOps

DevOps Habits and Practices

- Plan +Track
- Develop + Test
- Release
- Monitor + Learn

DevOps Adoption

DevOps
is the union of people, process, and tools to enable continuous delivery of value to end users

Cornerstones of DevOps:

- Culture supporting sharing and collaboration
- Continuously optimized lean process
- Automated deployment pipeline

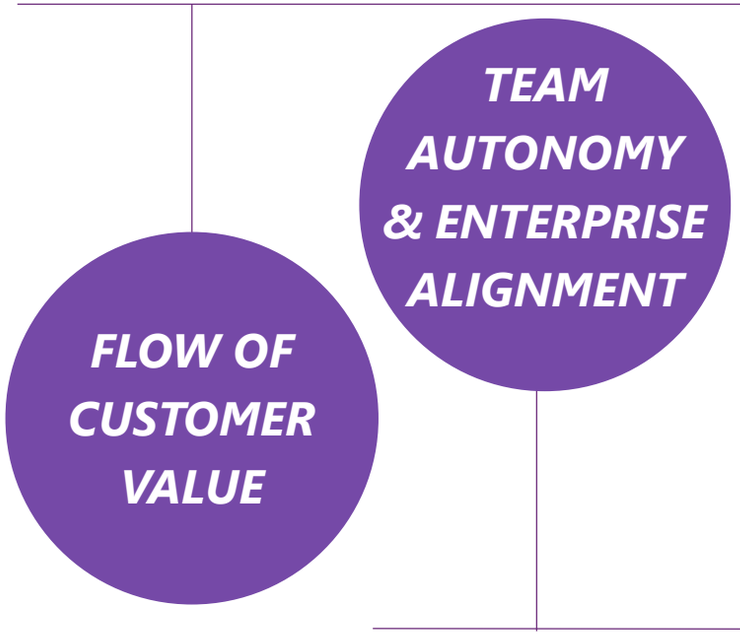
Best DevOps Organizations

- High performing organizations
  - Releasing many features frequently
- High Quality
- Low Cycle time
- Low Lead time
- Low MTTR , high MTBF
  - Mean Time To Recovery
  - Mean Time Between Failures

DevOps Habits

DevOps Habits

Insist on delivering more features more frequently, through automation, optimized process and collaborative culture. Shorter cycle times by reducing impediments and rework loops to support increased responsiveness, in turn fostering stakeholder satisfaction and trust.

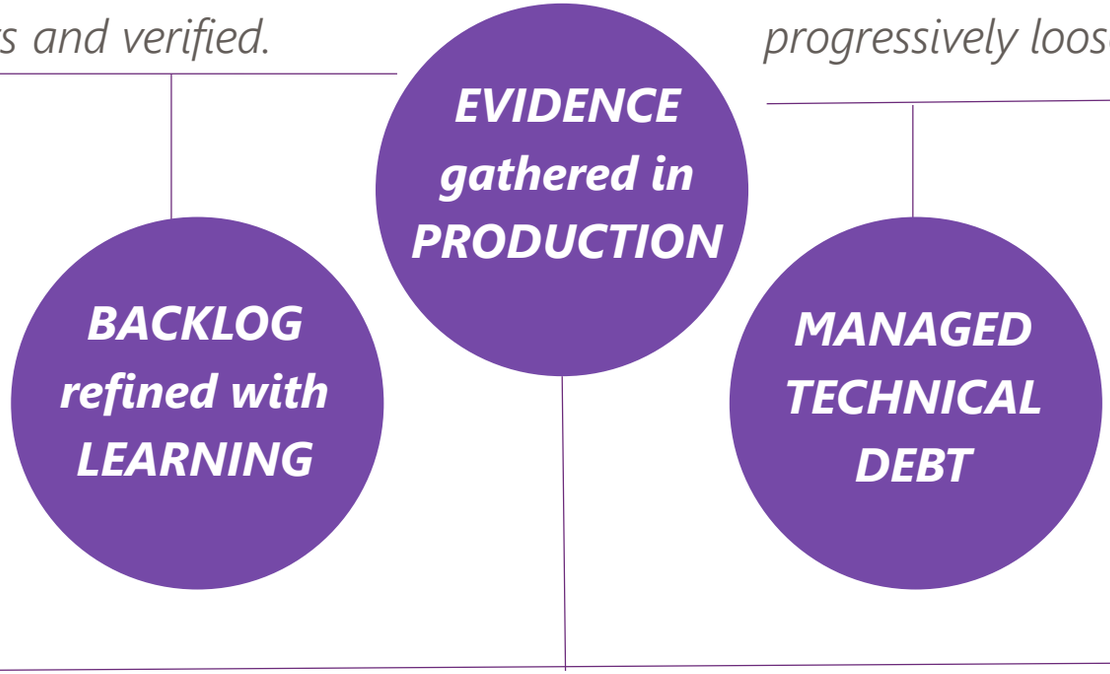


Strive towards multidisciplinary feature crews who have the freedom to best organize themselves in order to deploy after each sprint. At the same time, organize work around a common product backlog to keep them aligned with enterprise wide strategies.

DevOps Habits

Backlog items emerge from what we learn from customer feedback, experiments and active monitoring. Treat the backlog as a set of hypotheses, which are turned into experiments and verified.

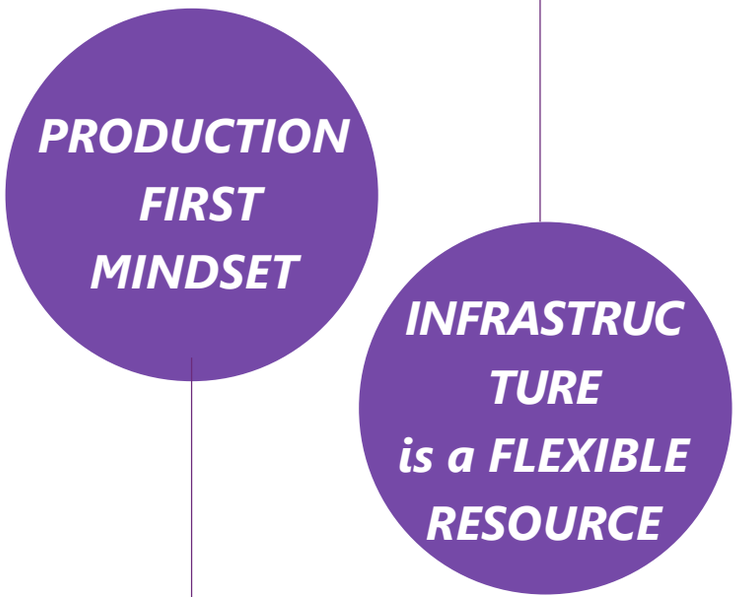
Technical debt are problems in a development effort that make forward progress on customer value inefficient. Without strict control of technical debt we progressively loose control over quality and progress.



Good decisions are informed by real data. Instrument everything, not just for health, availability, performance, and other qualities of service, but to understand usage and to collect evidence relative to the backlog hypotheses.

DevOps Habits

Dinamically manage your environments and resources. Make use of flexible and scalable cloud infrastructure and continually improve architecture to refactor into more independent, discrete services.

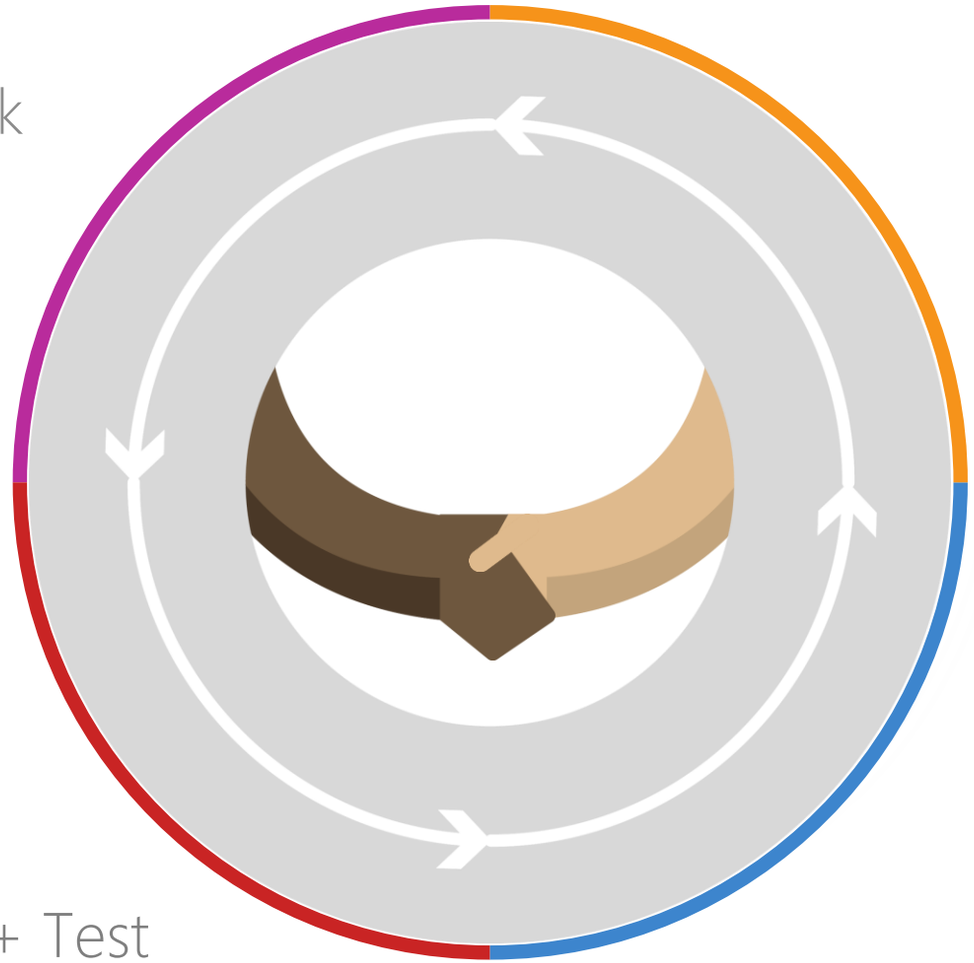


Production lies at the heart of any software delivery organization, and the best ones recognize that the real production should be top priority for every team member in every role.

End-to-End DevOps

1 Plan + Track

4 Monitor + Learn



Development

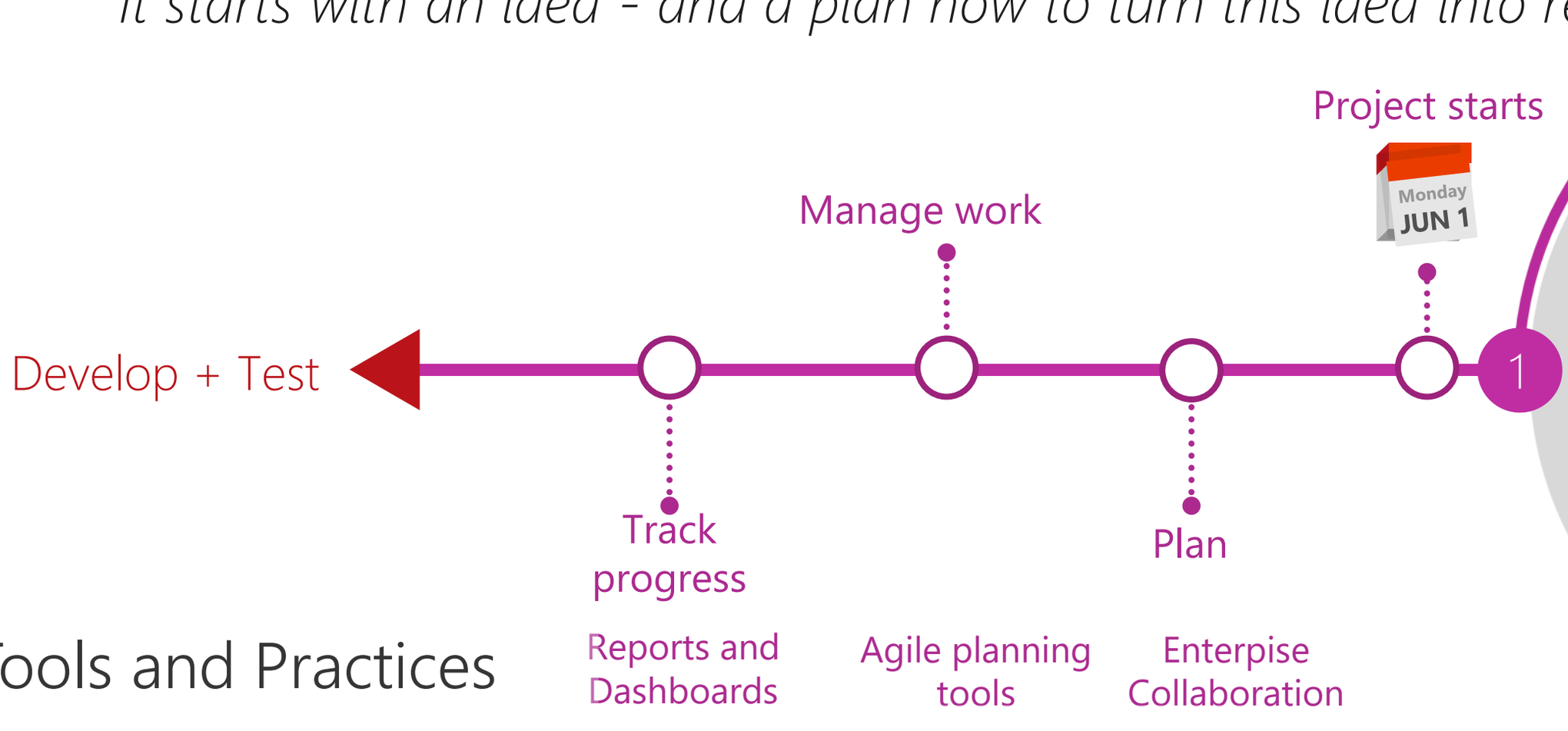
Production

2 Develop + Test

3 Release

Plan + Track

It starts with an idea - and a plan how to turn this idea into reality…



Tools and Practices

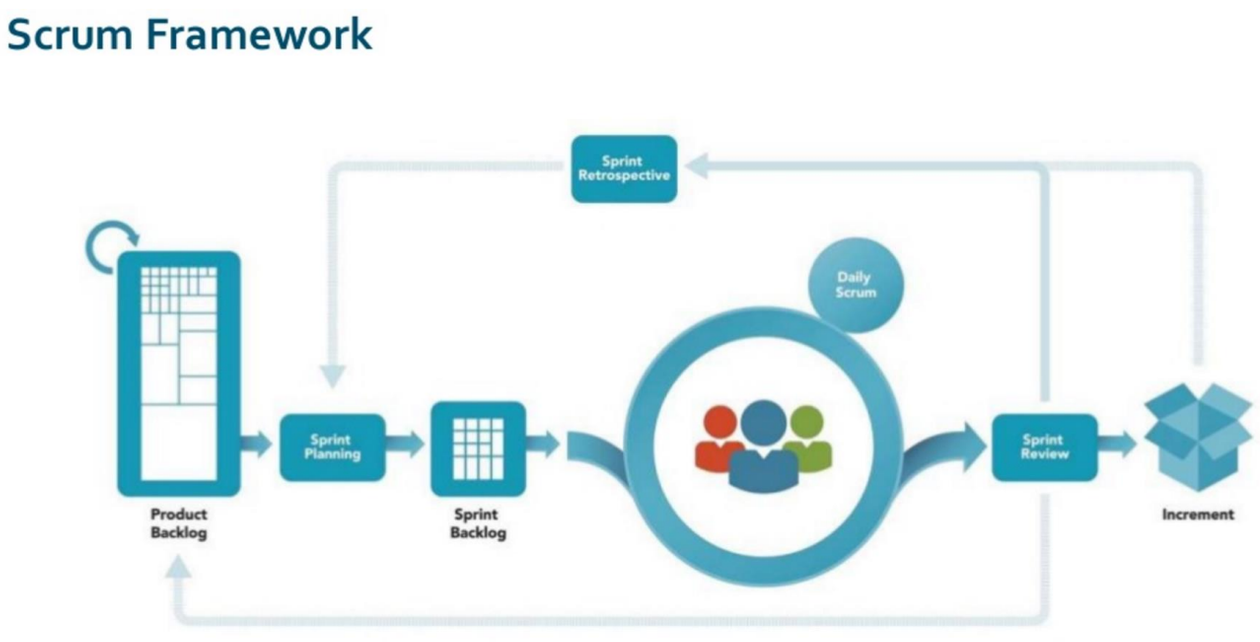
Process

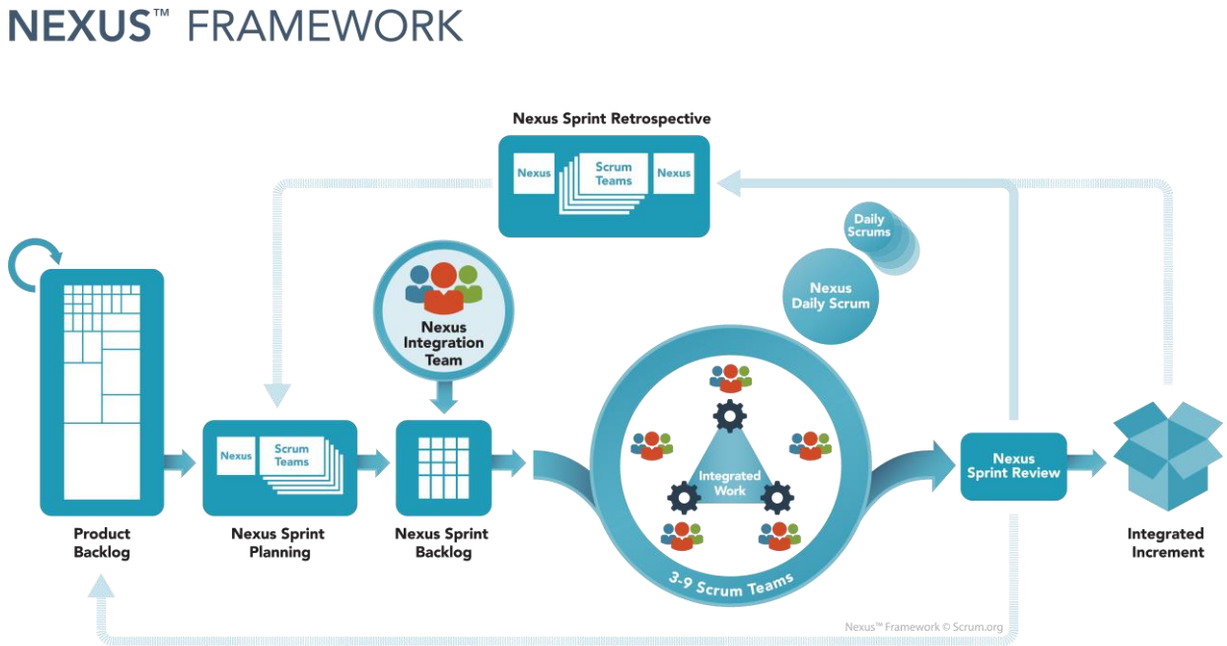
	Reports and Dashboards	Agile planning tools	Enterpise Collaboration
Process		Scrum Kanban	Scaled Scrum - Nexus

Process – Scrum, Nexus and Kanban

Scrum – Agile Process of Choice

Nexus – Scrum scaled to 3-9 Teams

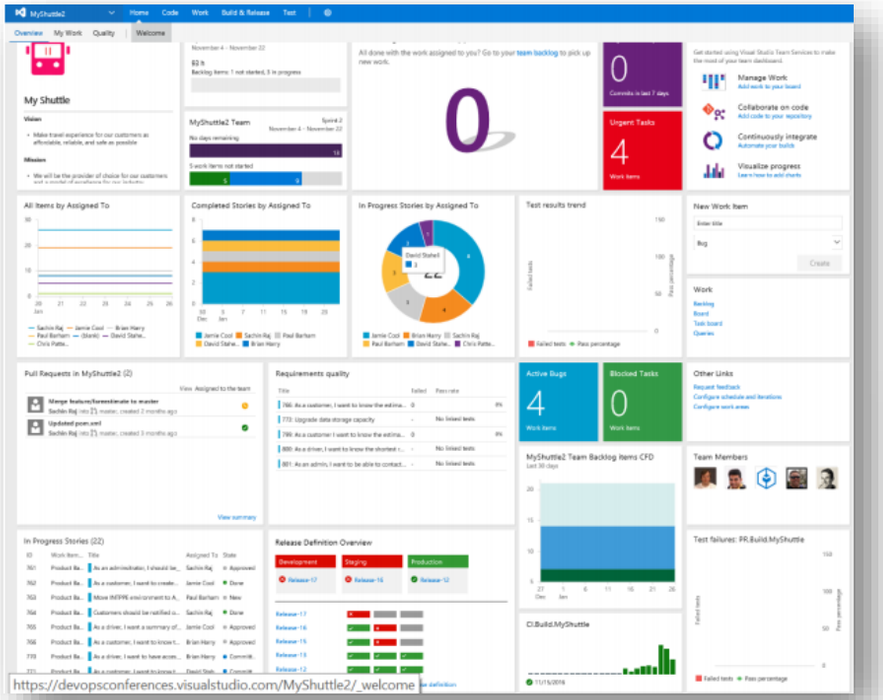




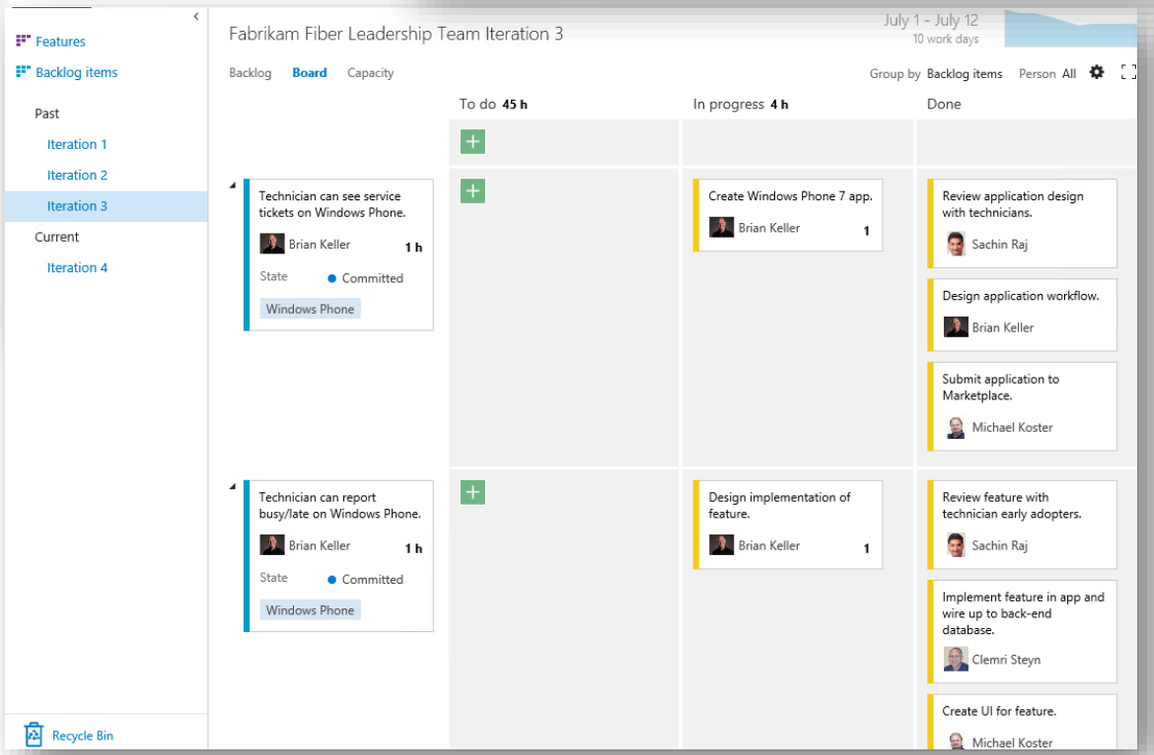
Kanban – For Support and Maintenance

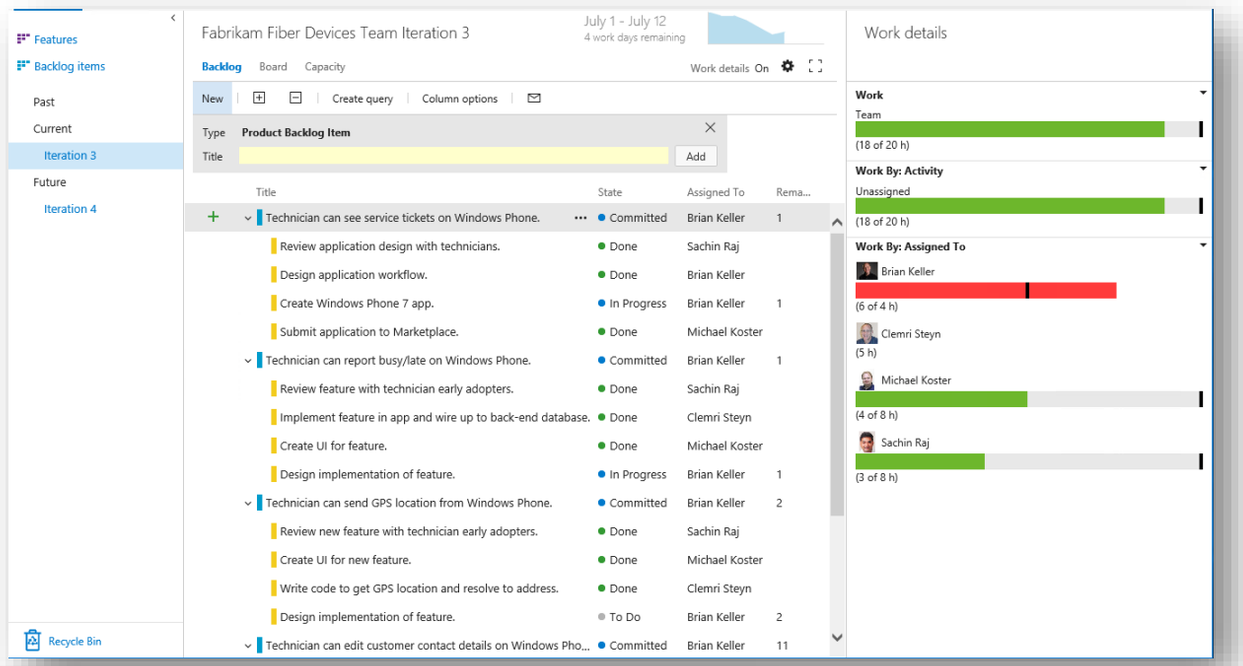
Plan + Track Tools

Dashboards



Tracking progress
Taskboard



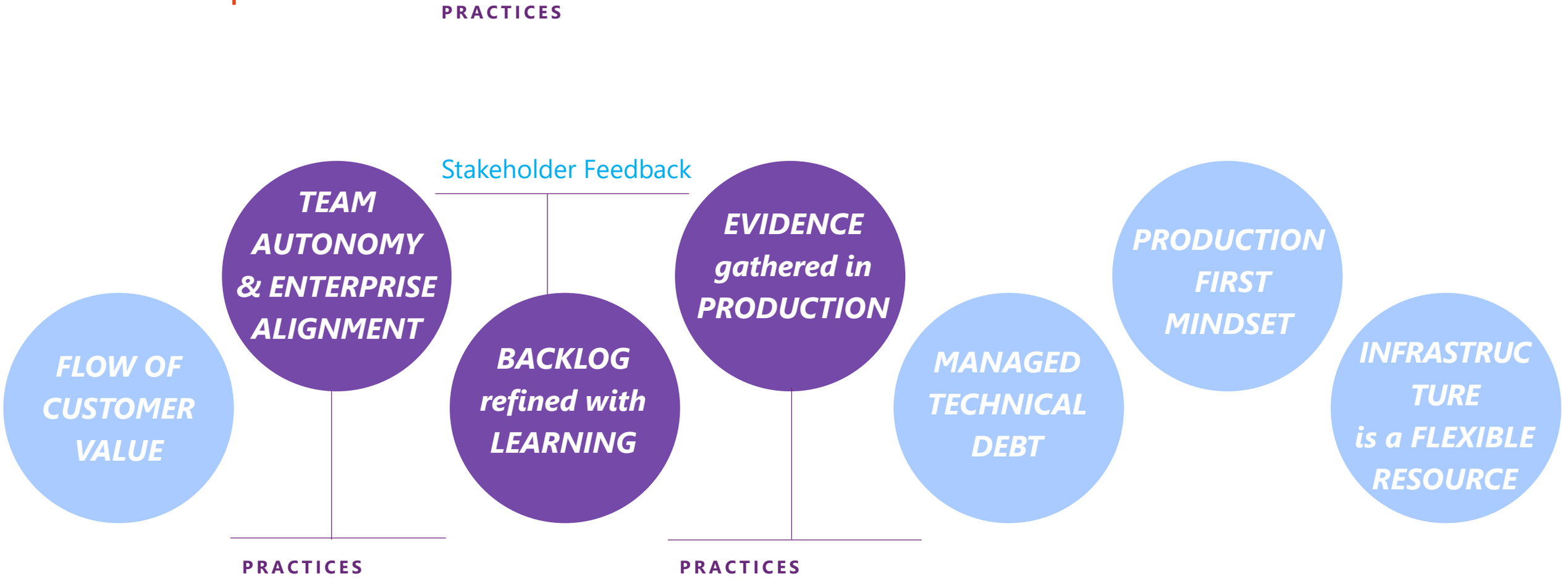


Agile planning tools
Backlog

Microsoft

KONFERENCA
Reach us with #ntk17

DevOps Habits and Practices – Plan & Track



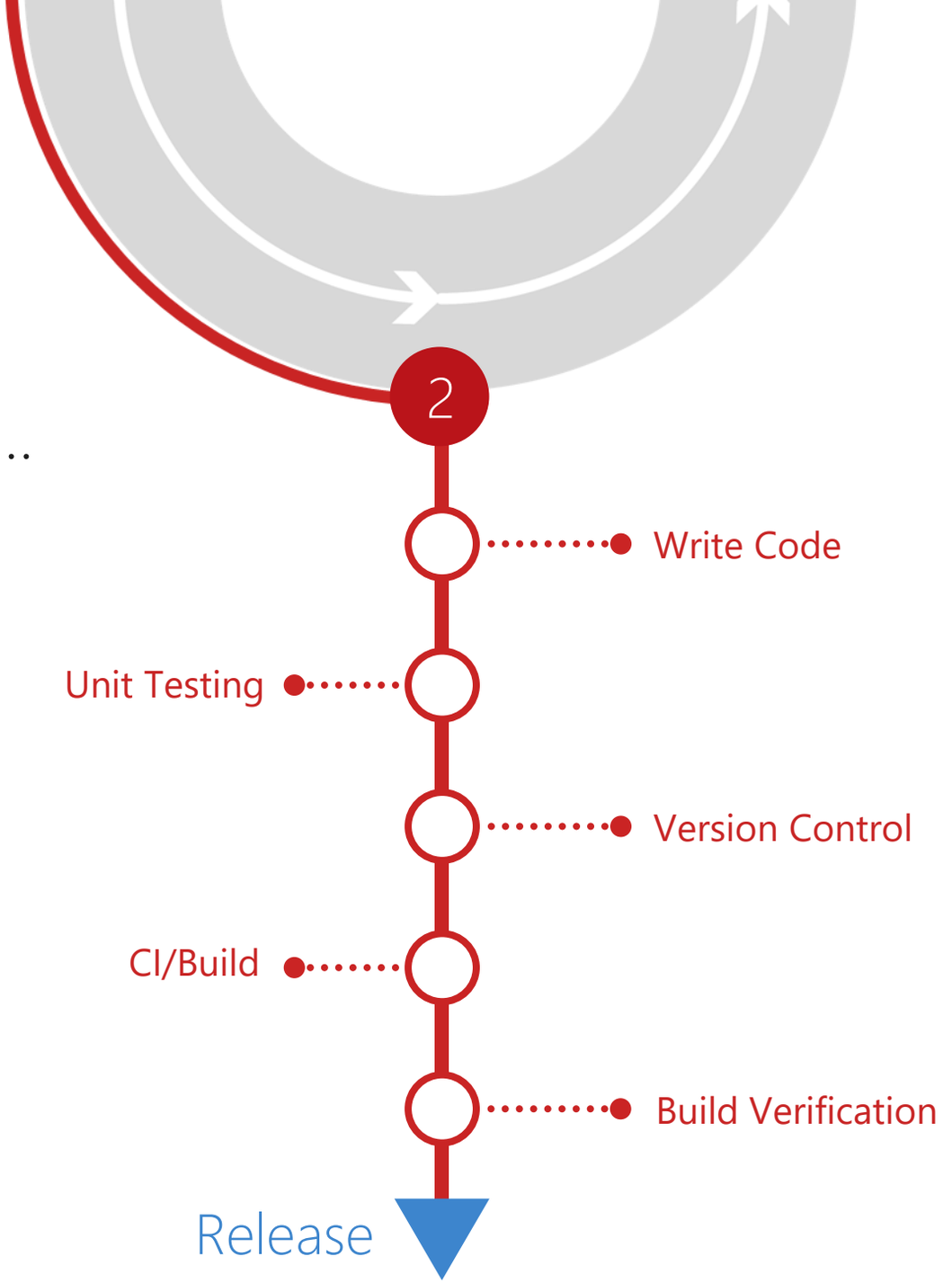
VSTS Agile Toolset Dashboards and Reporting Demo

Microsoft

KONFERENCA

Develop + Test

After the iteration starts, developers turn great ideas into features and functionality …



Tools and Practices

Version Control and Code Reviews

Automated Build

Automated Tests

Tests as a part of the Build

Code and Quality - Principles and Practices

Principle: always stay very close to ready to deploy to production

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

Invest in Continuous Integration (CI) with automated testing

Code Reviews

- Supported by Version Control Tools – Git Pull Request Experience
- Ensure Quality
- Promote Shared Team Code Ownership

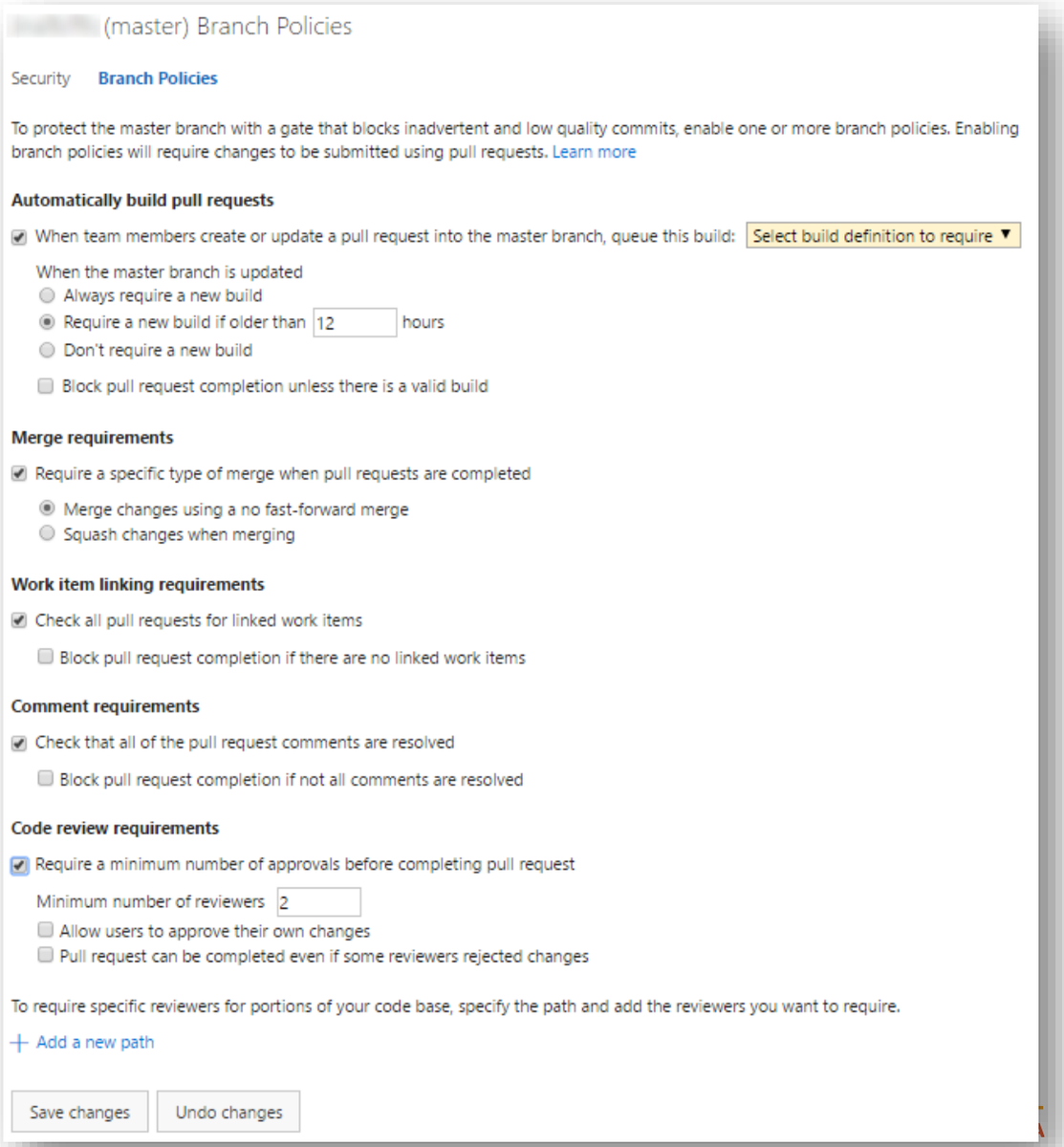
Pull Requests Demo

Microsoft

KONFERENCA

Advanced Code Practices – For high throughput

- Work in master (trunk)
  - Alternatively use short lived feature branches
- Don't use integration branches
- Put changes behind feature flags
- Branch policies
  - Must have a green build
  - Optional: functional test run



Testing - Automated and Manual Tests

Automated Testing

- Acceptance Tests
  - Unit Tests, Functional or Integration Tests, Stress or Load Tests
  - Performance, Security, Usability etc.
- Regression Tests
- Safety net for rapid development - Prerequisite for sustainable development effort
- Recommended practice: Test Shift Left

Manual Testing

- Doesn't scale - High throughput teams avoid manual testing
- Specified (following a specific usage scenario within a test plan)
- Exploratory

Best Practice – Test Shift Left

Tests should be written at the lowest level possible

Test early – more unit test like, less UI based tests

- Replace UI based functional tests with unit test based (functional and integration) tests
- Replace fragile UI tests with fast and robust (unit) tests
- Usually requires refactoring of the architecture

Tests become significantly faster and more reliable

Bugs are found earlier in the cycle

- Developers get quick feedback on their commit, which further reduces context switching

DevOps Habits and Practices - Develop & Test

PRACTICES

Automated Testing

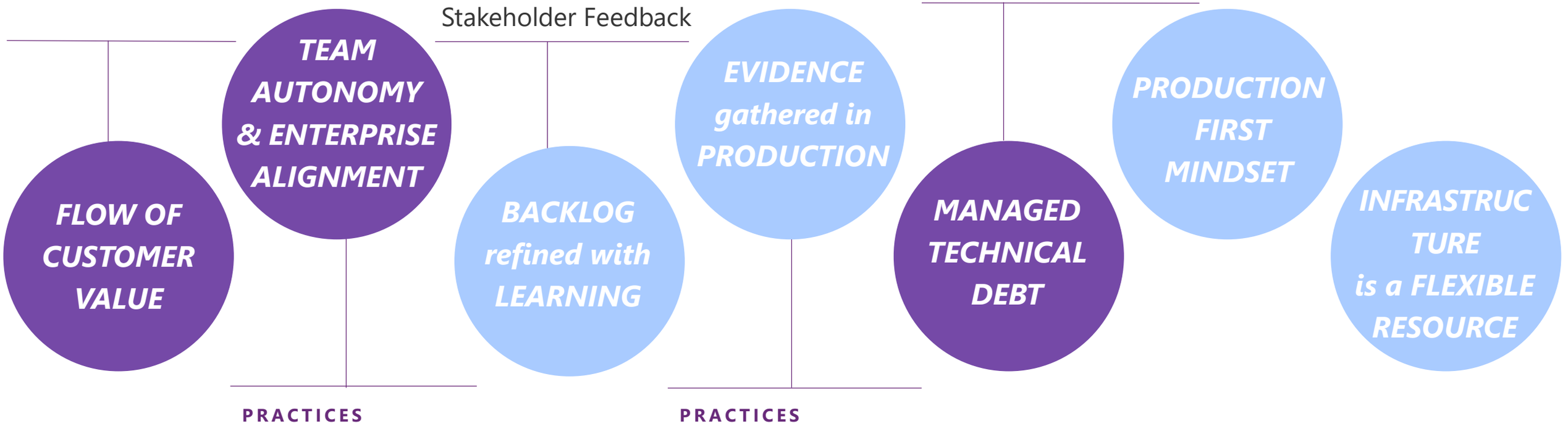
Continuous Integration

PRACTICES

PRACTICES

Code Reviews

Automated Testing



PRACTICES

Enterprise Agile

Continuous Integration

PRACTICES

Stakeholder feedback

Automated Tests – Unit and UI Demo

Microsoft

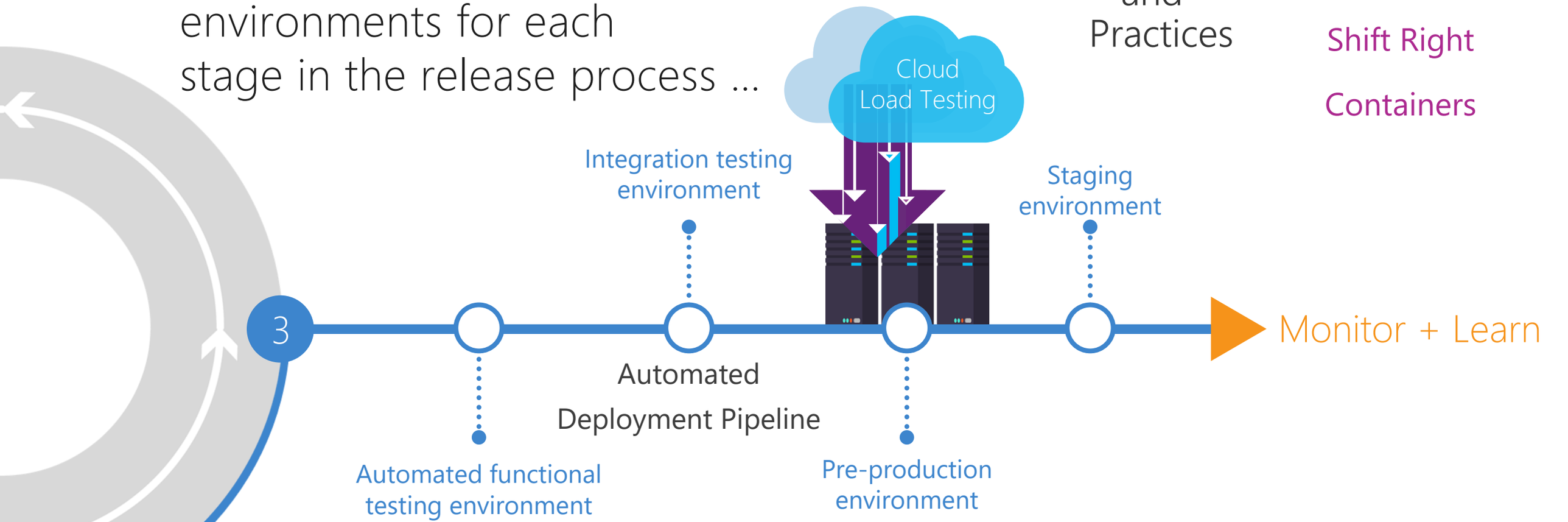
KONFERENCA

Release

When all tests pass, the build is automatically deployed and tests executed in test environments for each stage in the release process …

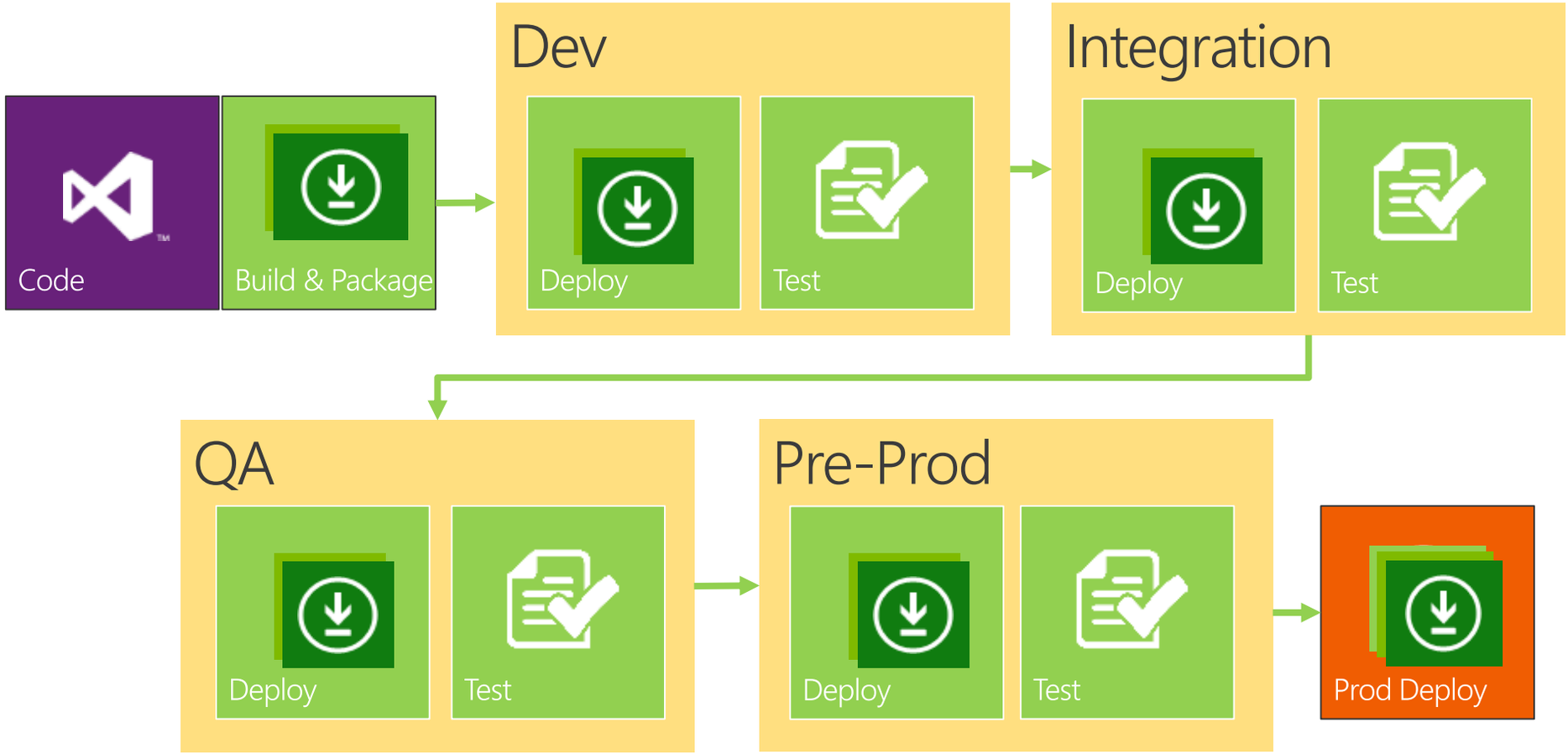
Tools and Practices

- Release Management
- Feature Flags
- Shift Right
- Containers



Release Management

Repeatable Automated Deployment Model



Deployment Principles

Invest in the Continuous Deployment pipeline

- Implement fully automated „Push button" deployments
- Allow for no down time

Execute tests in all phases of the delivery pipeline

- Treat test failures in CI/CD pipeline as high priority bugs

Promote joint ownership between engineering and ops

- Streamline and automate workflow between Dev & Ops to deliver higher quality software more frequently with less risk

Automated Recovery

- Built in mechanisms that detect a deployment failure and attempt an automated recovery to a working state

Feature Flags

- Decouple deployment of features from their exposure to end-users
  - All done code is deployed, but feature flags control exposure
  - Granular control of user exposure to features down to an individual user
  - Combined best with multiple rings of production environments
  - Users can be added or removed with no redeployment
  - Turned off quickly
- Enable progressive experimentation & refinement
- Support early feedback
- Decouple engineering and marketing
  - Enables „dark launch" of features
- *Note! Avoid using feature flags for dark delivery of unfinished code*
  - *Prefer delivering more granular features in done state*

Shift Right – Test in Production

Leverage end users as scalable manual testers

- Production is the best place to look for certain types of issues

Control exposure of features to users

- For web/cloud apps use multiple rings of production environments
- Use Feature flags

Monitor metrics and act on irregularities

- Proactively react, fix problems, redeploy

DevOps Habits and Practices - Release

PRACTICES

Automated Testing

Continuous Integration

Continuous Deployment

Release Management

FLOW OF CUSTOMER VALUE

TEAM AUTONOMY & ENTERPRISE ALIGNMENT

PRACTICES

Enterprise Agile

Continuous Integration

Continuous Deployment

Release Management

PRACTICES

Testing in Production

Stakeholder Feedback

BACKLOG refined with LEARNING

EVIDENCE gathered in PRODUCTION

PRACTICES

Testing in Production

Stakeholder feedback

Feature flags

PRACTICES

Code Reviews

Automated Testing

MANAGED TECHNICAL DEBT

PRODUCTION FIRST MINDSET

PRACTICES

Infrastructure as Code

Continuous Delivery

Release Management

Configuration Management

Automated Recovery

PRACTICES

Infrastructure as Code

Continuous Deployment

Release Management

Configuration Mng.

Automated Recovery

INFRASTRUCTURE is a FLEXIBLE RESOURCE

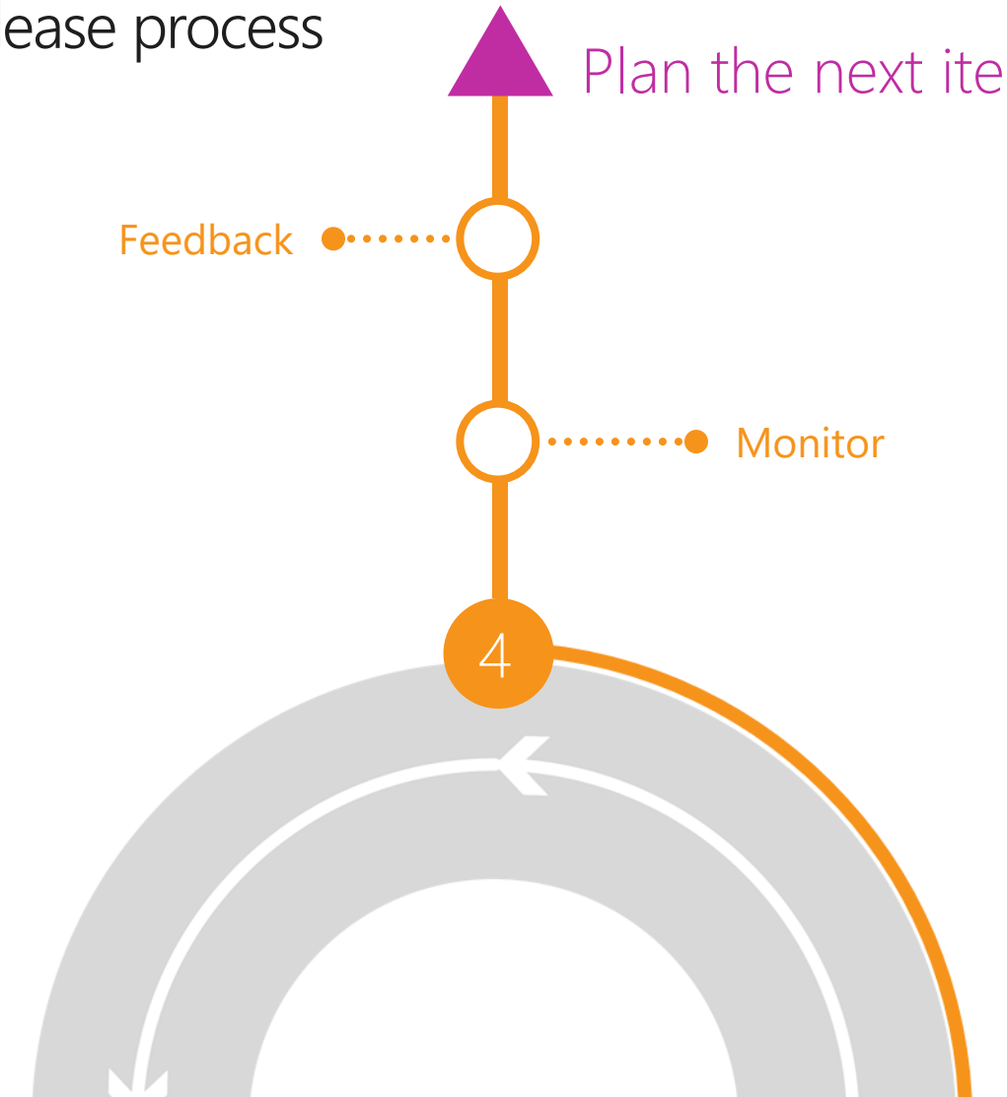
Release Management Demo

Microsoft

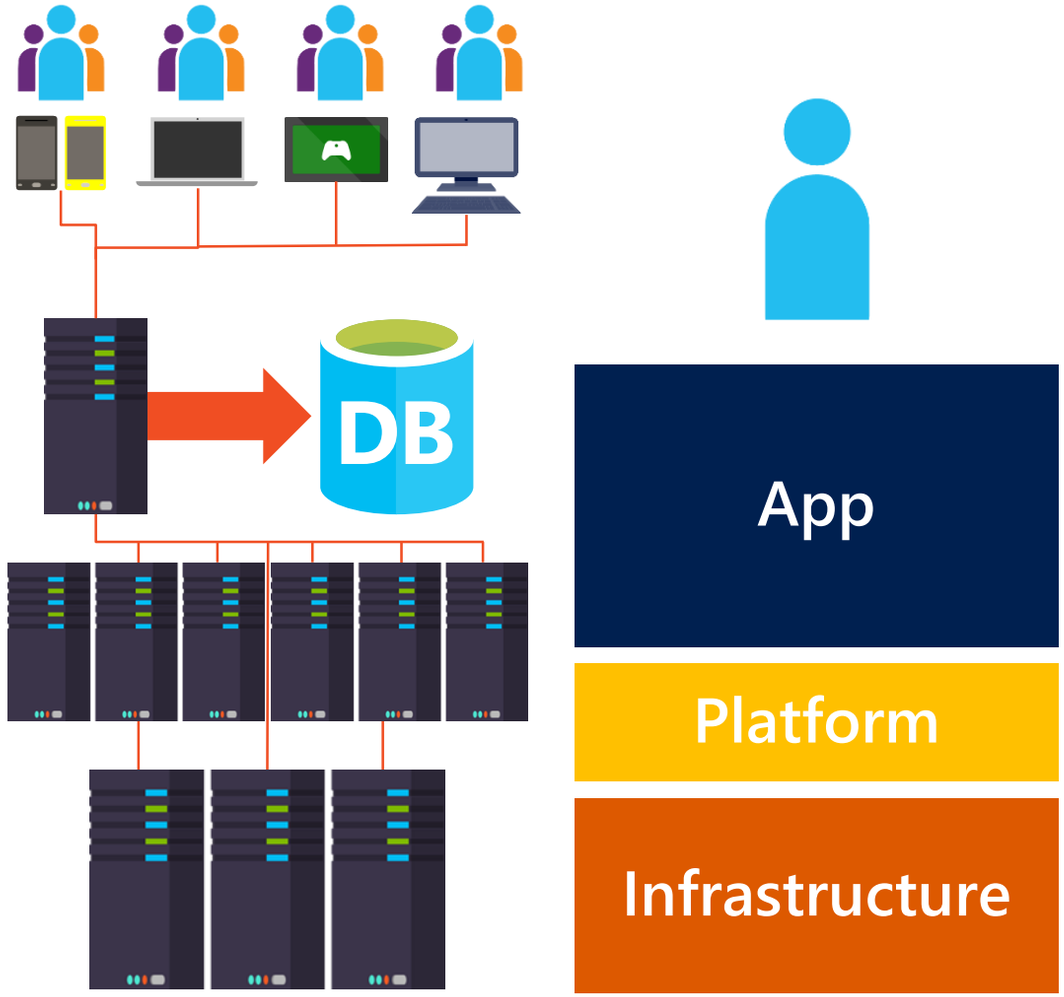
KONFERENCA

Monitor + Learn

When all tests pass, the build is deployed to testing environments for each stage in the release process

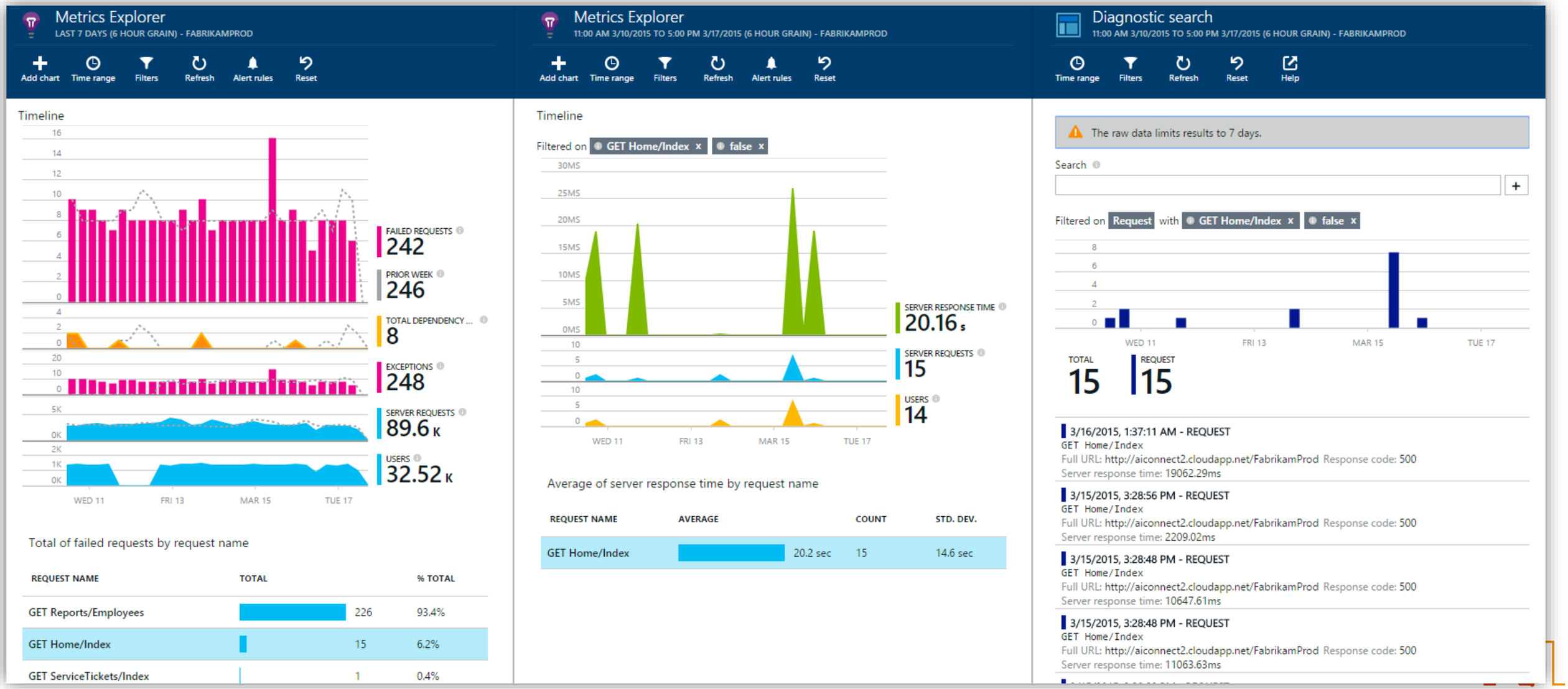


Sources of Telemetry



1. **Outside-in monitoring**
   URL pings and web tests from 16 global points of presence
2. **Observed user behavior**
   How is the application being used?
3. **Developer traces and events**
   Whatever the developer would like to trace
4. **Observed application behavior**
   No coding required – service dependencies, queries, response time, exceptions, logs, etc.
5. **Infrastructure performance**
   System performance counters

Application Insights - Drill Down Tools for Powerful Insights



Alerting is Key to Fast Detection

Alerts should create a sense of urgency

- False and redundant alerts dilute urgency
- Set right thresholds and tune often

Every alert must be actionable and represent a real system issue

- Response team
- On-call DRI (Designated Responsible Individual)

Optimize for fixing quickly

- Production First Mindset

DevOps Habits and Practices - Monitor & Learn

PRACTICES

Automated Testing
Continuous Integration
Continuous Deployment
Release Management

FLOW OF CUSTOMER VALUE

TEAM AUTONOMY & ENTERPRISE ALIGNMENT

PRACTICES

Enterprise Agile
Continuous Integration
Continuous Deployment
Release Management

PRACTICES

Usage Monitoring
Telemetry Collection
Testing in Production
Stakeholder Feedback

BACKLOG refined with LEARNING

EVIDENCE gathered in PRODUCTION

PRACTICES

Testing in Production
Usage Monitoring
User Telemetry
Stakeholder feedback
Feature flags

PRACTICES

Code Reviews
Automated Testing
Continuous Measurement

MANAGED TECHNICAL DEBT

PRODUCTION FIRST MINDSET

PRACTICES

Application Performance Management
Infrastructure as Code
Continuous Delivery
Release Management
Configuration Management
Automated Recovery

INFRASTRUCTURE is a FLEXIBLE RESOURCE

PRACTICES

Application Performance Mng.
Infrastructure as Code
Continuous Deployment
Release Management
Configuration Mng.
Automated Recovery

Microsoft

DevOps Habits and Practices - End to End

PRACTICES

Automated Testing
Continuous Integration
Continuous Deployment
Release Management

FLOW OF CUSTOMER VALUE

TEAM AUTONOMY & ENTERPRISE ALIGNMENT

PRACTICES

Enterprise Agile
Continuous Integration
Continuous Deployment
Release Management

PRACTICES

Usage Monitoring
Telemetry Collection
Testing in Production
Stakeholder Feedback

BACKLOG refined with LEARNING

EVIDENCE gathered in PRODUCTION

PRACTICES

Testing in Production
Usage Monitoring
User Telemetry
Stakeholder feedback
Feature flags

PRACTICES

Code Reviews
Automated Testing
Continuous Measurement

MANAGED TECHNICAL DEBT

PRODUCTION FIRST MINDSET

PRACTICES

Application Performance Management
Infrastructure as Code
Continuous Delivery
Release Management
Configuration Management
Automated Recovery

PRACTICES

Application Performance Mng.
Infrastructure as Code
Continuous Deployment
Release Management
Configuration Mng.
Automated Recovery

INFRASTRUCTURE is a FLEXIBLE RESOURCE

Microsoft

DevOps Adoption/Agile Transformation Agenda

Start with Scrum

Change Release Process

- Automate delivery pipeline

Change Testing Methodology

- Automate Tests and test environments, Shift left, Shift right

Increase the release cadence

Introduce automated processes without delay

Proceed cautiously – Cultural changes can't be rushed

Proceed driven by business needs

Thank you!

Questions?

obajic@ekobit.hr

aroje@ekobit.hr

Microsoft

KONFERENCA

Microsoft

KONFERENCA

PORTOROŽ, 15. DO 17. MAJ 2017