Zukunft der Anwendungsentwicklung im Enterprise Umfeld

ADC 2016

Keynote

Rainer Stropek

software architects gmbh

Web http://www.timecockpit.com

Mail rainer@timecockpit.com

Twitter @rstropek

Advanced Developers Conference 16

Development for Professionals!

Agenda

Microsoft ist dabei, sich drastisch zu verändern, das ist nicht zu übersehen. Roslyn, .NET Core, Visual Studio „15“, TypeScript, Open Source, Container, wöchentlich neue Azure-Dienste – es ist schwierig, auf dem Laufenden zu bleiben. Speziell im Enterprise-Umfeld wird die „neue Microsoft“ kritisch beäugt. Sind die neuen Technologien Enterprise-tauglich oder handelt es sich nur um Spielereien für Startups? Was wird besser durch sie? Warum ist die Veränderung überhaupt notwendig? Rainer Stropek, langjähriger Azure MVP und MS Regional Director geht in seiner Keynote auf diese Fragen ein. Ausgehend von generellen Trends in der Softwarearchitektur und Organisation von IT-Projekten wie DevOps und Microservices zeigt er, welches Potential in den neuen Technologien steckt. Rainer spricht darüber, wie Microsoft eine interessante Strategie verfolgt, die Softwareentwicklung im Enterprise-Umfeld auf ein ganz neues Niveau heben kann.

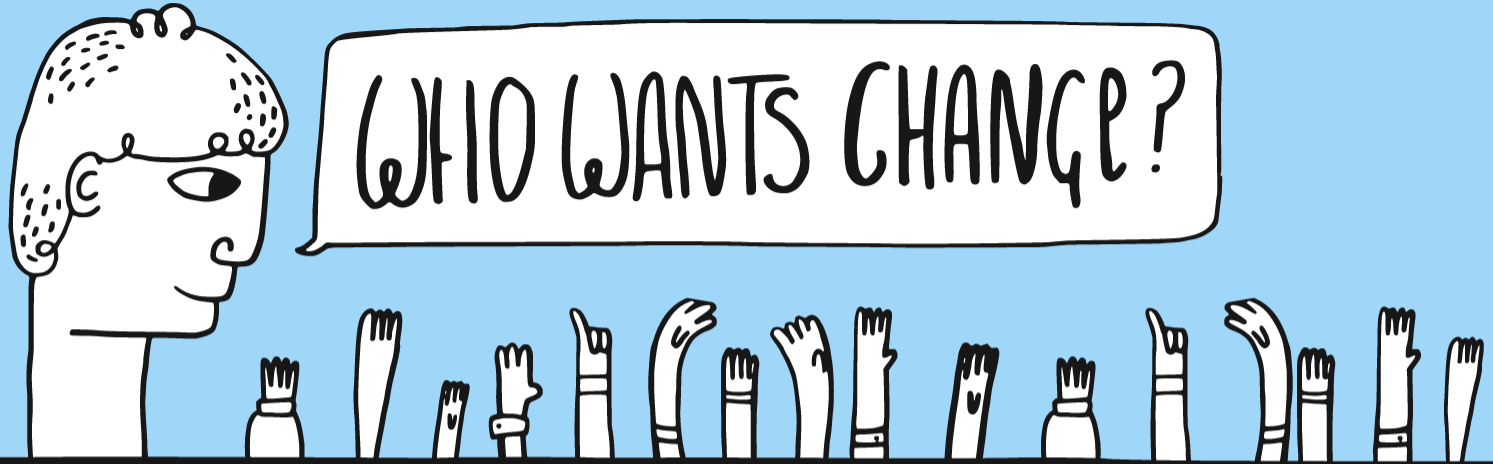
Your Host

Rainer Stropek

Developer, Entrepreneur
Azure MVP, MS Regional Director
IT-Visions

Contact

software architects gmbh
rainer@timecockpit.com
Twitter: @rstropek



Microsoft is Changing – Examples

Ubuntu Subsystem in Windows

Running unmodified, native Linux binaries in Windows without VM or Container
https://msdn.microsoft.com/en-us/commandline/wsl/about

Open Source PowerShell on Linux

https://github.com/PowerShell/PowerShell

Containers, Participating in Docker Ecosystem

E.g. *microsoft/dotnet*, *microsoft/powershell*
Docker on Windows

```
Start Bash on Windows
  cd /mnt/c/…
  vim some-js.js
  node some-js.js

Run Powershell on Linux
  Docker: microsoft/powershell
  $something = „asdf“
  Write-Host $something
  Get-Item tmp

Run Docker on Windows
  docker run -it –rm microsoft/windowsservercore cmd
  docker run –d –p 8080:80 microsoft/iis
```

Demo

Microsoft is changing

Microsoft's Change

Earnings Release FY17 Q1

Selected Product and Service Revenue Constant Currency Reconciliation

	Thre
	Percentage Change Y/Y (GAAP)
Office commercial products and cloud services	5%
Office 365 commercial	51%
Office consumer products and cloud services	8%
Dynamics products and cloud services	11%
Server products and cloud services	11%
Azure	116%
Enterprise Services	1%
Windows OEM	0%
Windows commercial products and cloud services	0%
Phone	(72)%
Gaming	(5)%
Search advertising excluding traffic acquisition costs	9%

Source: Microsoft Investor Relations

Enterprises are Changing

Digital Interdependence

Digital ecosystem readiness

"79% of [...] top performers [...] participate in a digital ecosystem"

Interoperability

External mindset

Focus on managing interdependence

BI/Analytics and Cloud Services

Top two investment areas of top performers

Source: 2017 CIO Agenda, Gartner Inc.; available at http://www.gartner.com/imagesrv/cio/pdf/Gartner_CIO_Agenda_2017.pdf

Environment is Changing

Azure Machine Learning, R, Cognitive Services, IoT Hub

Hololens, Windows Holographic, Azure BaaS

Intelligent

Artificial Intelligence and Advanced Machine Learning

Intelligent Apps

Intelligent Things

Digital

Virtual Reality and Augmented Reality

Digital Twins

Blockchains and Distributed Ledgers

Conversational Systems

Digital Technology Platforms

Mesh App and Service Architecture

Adaptive Security Architecture

Mesh

Bot Framework, Azure Platform

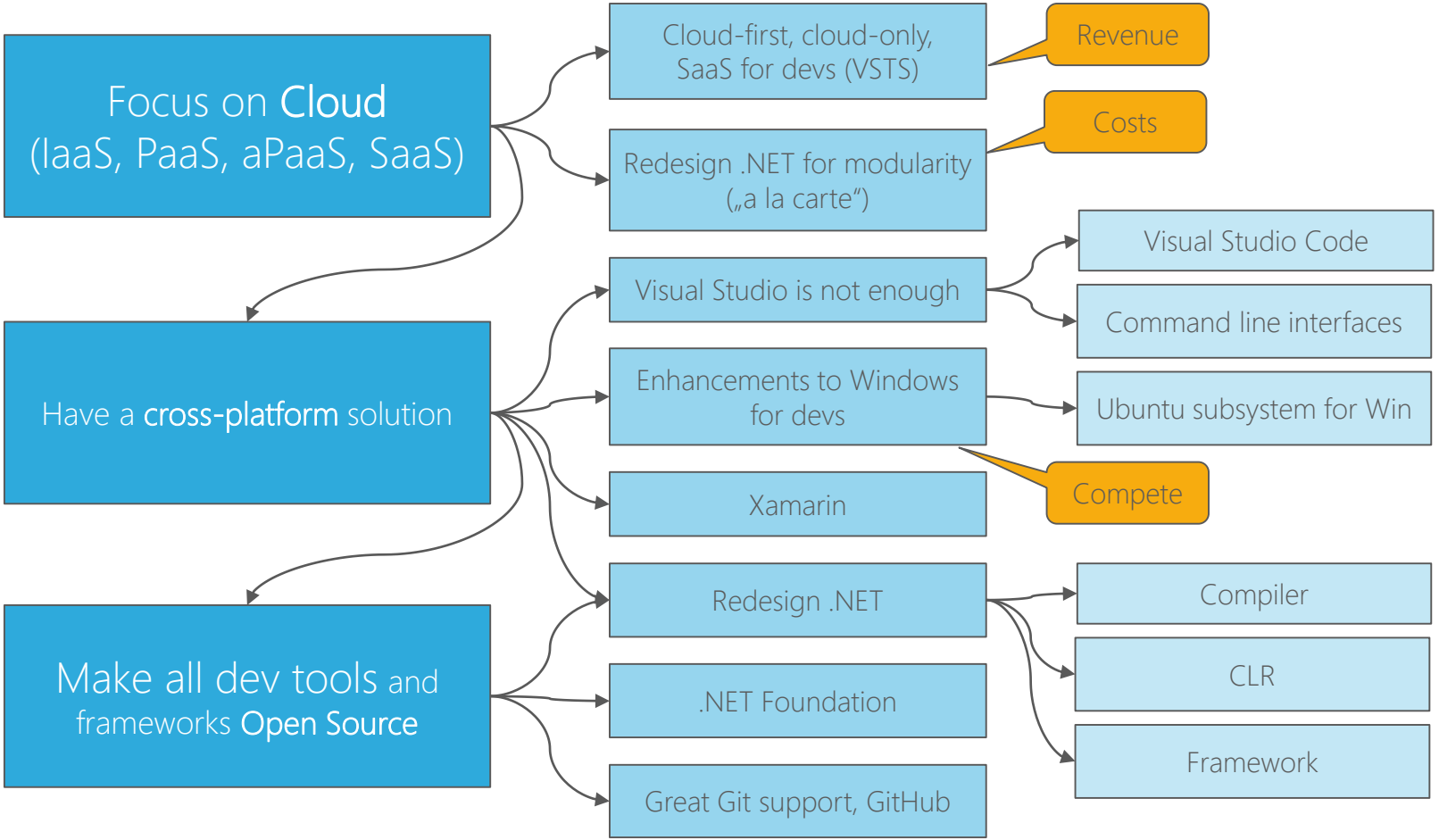
Xamarin, Web Technologies, DevOps, APIs

Source: Gartner, Oct. 2016, available via http://www.gartner.com/smarterwithgartner/gartners-top-10-technology-trends-2017/


What does that mean
for me as an
enterprise developer?

Consequences on .NET

Why and how had .NET to change?


Focus on Cloud
(IaaS, PaaS, aPaaS, SaaS)
Have a cross-platform solution
Make all dev tools and frameworks Open Source
Cloud-first, cloud-only, SaaS for devs (VSTS)
Revenue
Redesign .NET for modularity („a la carte“)
Costs
Visual Studio is not enough
Visual Studio Code
Command line interfaces
Enhancements to Windows for devs
Ubuntu subsystem for Win
Compete
Xamarin
Redesign .NET
Compiler
CLR
Framework
.NET Foundation
Great Git support, GitHub

Switch to .NET Core

Don‘t rush things

Build components based on .NET Standard

Getting ready step-by-step

Re-think your software architecture

Mini- and Microservices
APIs
Various UI platforms

PLATFORM NAME	ALIAS								
.NET Standard	netstandard	1.0	1.1	1.2	1.3	1.4	1.5	1.6	2.0
.NET Core	netcoreapp	→	→	→	→	→	→	1.0	vNext
.NET Framework	net	→	4.5	4.5.1	4.6	4.6.1	4.6.2	vNext	4.6.1
Mono/Xamarin Platforms		→	→	→	→	→	→	→	vNext
Universal Windows Platform	uap	→	→	→	→	10.0	→	→	vNext
Windows	win	→	8.0	8.1					

Microservices

What are Microservices?

Small, autonomous services working together

Single responsibility principle applied to SOA
See also concept of Bounded Context

Best used with DevOps and continuous deployment

Enhance cohesion, decrease coupling, enable incremental evolvement

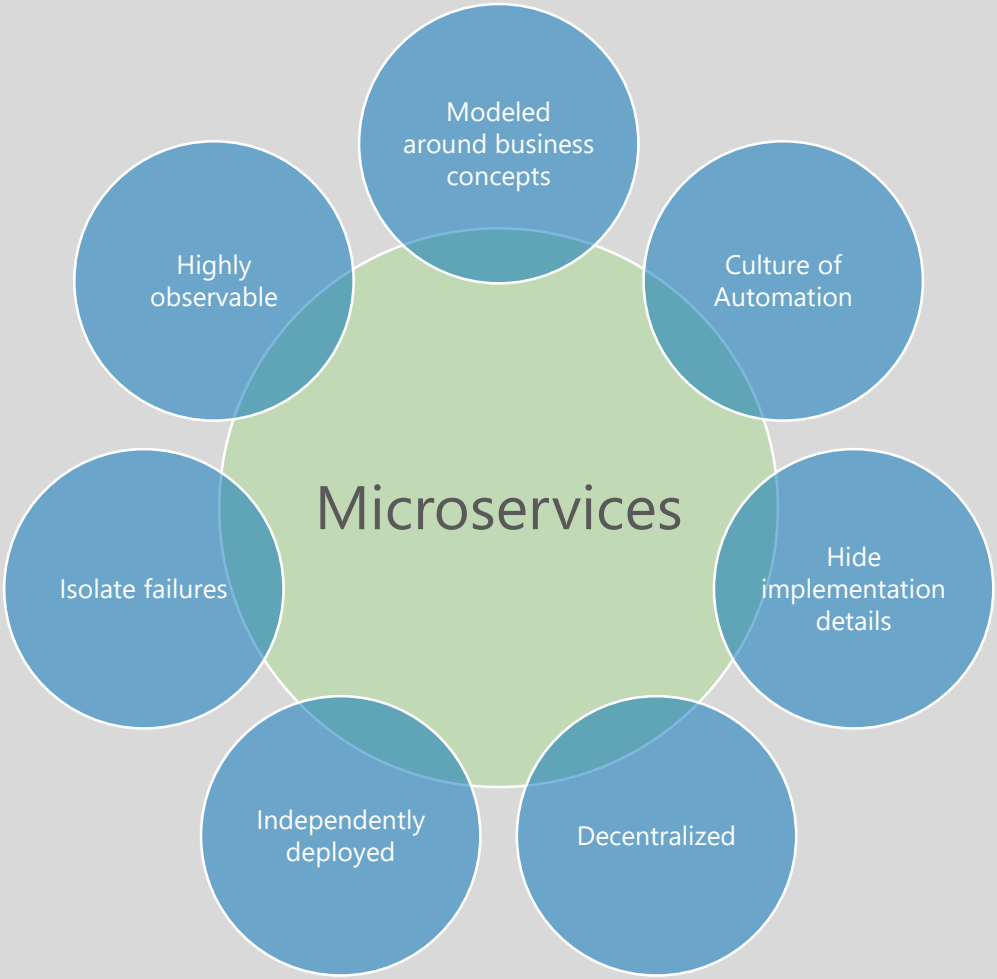
How small are Microservices?

It depends (e.g. team structure, DevOps maturity, etc.)
"… one agile team can build and run it", "… can be rebuilt by a small team in two weeks"
Find an individual balance

Autonomous = deploy changes without affecting others

Technology- and platform-agnostic APIs

See also https://en.wikipedia.org/wiki/Microservices



Microservices

Fundamental ideas

Work alongside many state-of-the-art approaches for software development

- Agile development techniques
- Continuous Integration/Delivery
- DevOps
- Cloud Computing
- Containers

Why Microservices?

Work well in heterogeneous environments

- Right tool for the job
- Available skills of team members
- Grown environment (e.g. M&A, changing policies, changing overall designs)

Easier to test/adopt new technologies

- Reduce risk and cost of failure
- New platforms (e.g. Node.js instead of .NET), new versions (e.g. .NET Core),

Resilience

- Reduce single point of failures
- Support different SLAs for difference modules (costs, agility)
- Separation of services add complexity (e.g. network) → criticism of Micrservices

Why Microservices?

Let people take responsibility

Teams "own" their services
You build it, you run it

Scaling

Fine-grained scaling is possible

Simplify deployment of services

Overall, deployment of many Microservices might be more complex → criticism
Deployment patterns: https://www.nginx.com/blog/deploying-microservices/

Why Microservices?

Composability

Hexagonal architecture

Ability to replace system components

Outdated technology
Changed business requirements

Why Not? (Examples)

Harder to debug and troubleshoot

Distributed system

Possible mitigation: Mature logging and telemetry system

Performance penalty

Network calls are relatively slow

Possible mitigation: Remote calls for larger units of work instead of chatty protocols

No strong consistency

We are going to miss transactions!

Possible mitigation: Idempotent retries

Why Not? (Examples)

System is too small

For small systems, monolithic approach is often more productive

Cannot manage a monolith (e.g. deployment)? You will have troubles with Microservices!

Environment with lots of restrictions

Microservices need a high level of autonomy

Harder to manage

You have to manage lots of services which are redeployed regularly

Possible mitigation: DevOps, Automation

DevOps

The converged DevOps lifecycle

Shift to DevOps

Old World

- Focus on planning
- Compete, not collaborate
- Static hierarchies
- Individual productivity
- Efficiency of process
- Assumptions, not data

New World

- Focus on delivering
- Collaborate to win
- Fluent and flexible teams
- Collective value creation
- Effectiveness of outcomes
- Experiment, learn and respond

DevOps habits and practices

FLOW OF CUSTOMER VALUE

PRACTICES
Automated Testing
Continuous Integration
Continuous Deployment
Release Management

TEAM AUTONOMY & ENTERPRISE ALIGNMENT

PRACTICES
Enterprise Agile
Continuous Integration
Continuous Deployment
Release Management

BACKLOG refined with LEARNING

PRACTICES
Usage Monitoring
Telemetry Collection
Testing in Production
Stakeholder Feedback

EVIDENCE gathered in PRODUCTION

PRACTICES
Testing in Production
Usage Monitoring
User Telemetry
Stakeholder feedback
Feature flags

MANAGED TECHNICAL DEBT

PRACTICES
Code Reviews
Automated Testing
Continuous Measurement

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 Management
Infrastructure as Code
Continuous Deployment
Release Management
Configuration Management
Automated Recovery

How to Change?

Conway's Law

„Any organization that designs a system will inevitably produce a design whose structure is a copy of the organization's communication structure"

Organizational hurdles for Microservices

- Tightly-coupled organizations
- Geographically distributed teams
- Missing tools (e.g. self-service cloud infrastructure, CI/CD tools)
- Inappropriate security policies
- Unstable or immature service that frequently changes
- Missing culture of taking ownership (need someone to blame)
- Cope with many different and new technologies

Source: Conway, How Do Committees Invent, Datamation magazine, April 1968

Organisational Helpers

Co-locate teams

One team responsible for a single service should be co-located

Embrace open source development style

Works internally, too

Internal consultants, custodians and trusted committers

Quality gateways
Servant leaders

Step-by-step approach

Be clear in communication

E.g. responsibilities, long-term goals, changing roles

Modern Architects…

…don’t create perfect end products

…help creating “a framework in which the right systems can emerge, and continue to grow”

…understand the consequences of their decisions

…code with the team (“architects should code”, “coding architect”)

…aims for a balance between standardization and freedom

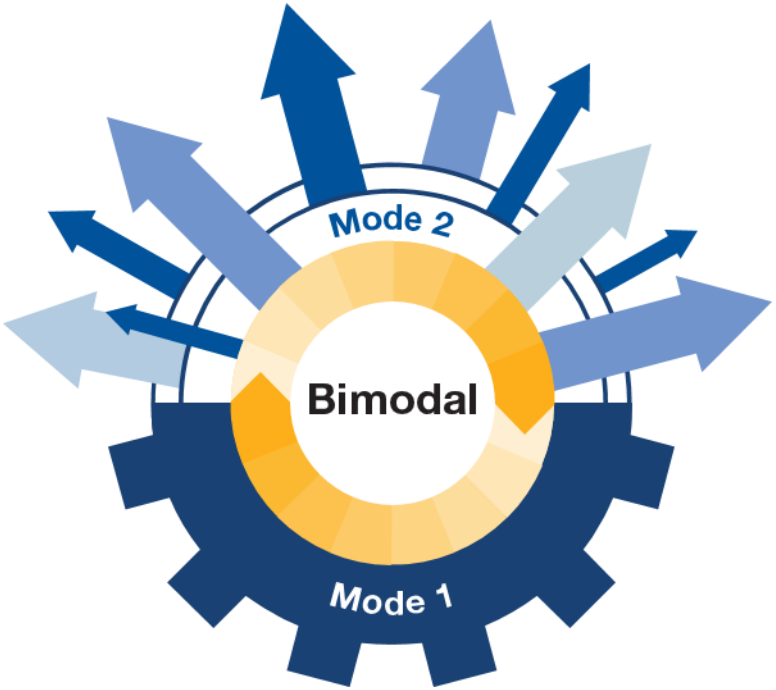
Build skills for a certain technology vs. right tool for the right job

…create guiding principals and practices

Example for principals (largely technology-independent): https://12factor.net/
Example for practices (often technology-dependent): .NET Core Coding Guildelines

Recommended reading: Newman, Sam. Building Microservices, O'Reilly Media

Bimodal Enterprise



Mode 1:
Predictability and Stability

Mode 2: Exploratory

We have to deliver in mode 1 to get trusted for mode 2

Source: Gartner, Deliver on the Promise of Bimodal, Feb. 2016, available via http://www.gartner.com/it-glossary/bimodal/

Zukunft der Anwendungsentwicklung im Enterprise Umfeld

Q&A

Thank your for coming!

Rainer Stropek

software architects gmbh

Mail rainer@timecockpit.com
Web http://www.timecockpit.com
Twitter @rstropek

Advanced Developers Conference 16
Development for Professionals!