Changing IT Delivery with DevOps and Microservices

Andreas Lennevi
“Enterprise DevOps”

Forrester in Oct 2017 predicted:

• “2018 will be The Year Of Enterprise DevOps”
• More than 50% of organizations are implementing DevOps
• Discussion shifted from:
  • “What is DevOps?” to
  • “How do I implement DevOps at scale?”.
• DevOps momentum is occurring in all industry sectors.
• Although many organizations are in the experimentation stage with single or multiple pilots – they all are transitioning toward DevOps across their entire enterprise.
Trends merging into coherent Movement:

- **2010**
  - SOA
  - Agile Teams
  - Cloud Infra Automation

- **2015**
  - Microservices
  - Mode 2, DevOps & Agile Mindset
  - Continuous Integration Release Automation

- **2020**
  - Microservices
  - Enterprise DevOps
  - CI/CD DevOps Automation

**Smaller Scale**
- Less Central IT

**Autonomous Teams**
- Autonomous Components
- Business & IT Alignment

**Specialization for Purpose**
- High Productivity Tooling
- Automate Everything
- Measure Everything
- Develop Testability
- Develop Operability
- Local Ownership

_mendix_
Enterprise DevOps

Why?
• Faster Product Delivery
• Manage Shifting Priorities
• More Value / Worked Hour
Agile Development + Rigid Ops = Rigid IT

Development became Agile and Efficient

Operations stayed rigid and stability focused

6 Weeks to Build

6 Months to Go-Live

6 Seconds to Imagine

6 Hours to Define
DevOps – The Culture Wrecking Ball
DevOps Pillars

“CAMS” by John Willis: John Willis coining “CAMS”:

DevOps = Collaboration & Automation for Biz + Dev + Ops

- Agile all the way from Business to Ops
- Automate to avoid Ops work
- Use similar tools and methods in Dev and Ops
- Monitor Use & Performance, Predict issues in PRD
- Measure Value & Feedback
- New IT4IT type tools
- Share problems and solutions from Biz to Dev to Ops and back

“CAMS”
Conflicting Objectives of Departments

- **BIZ**
  - Business
  - Get New Functions

- **DEV**
  - Bring New Features Live
  - Avoid Risk

- **TST**
  - Culture clash

- **OPS**
  - Maintain Stable Environments
BizDevOps: Cooperating around Joint Objectives

Putting people in the same team, makes them see the same problems and solve them together.

After some time with joint Objectives, priorities align & team works as a unit.
**BizDevOps**: Small Team Owns entire Life Cycle

**BizDevOps Teams with Agile Mindset**

**Continuous Delivery**

**Continuous Integration**

- Portfolio Mgmt.
- Application Lifecycle
- Build Automation
- Release Creation
- Test Automation
- Application Release
- Application Usage
Why is DevOps so Appealing: Business & IT Alignment

Faster-than-competition Digital Evolution

Business & IT Alignment
With flexibility and lower cost changes
Innovation & Efficiency

(Biz)DevOps
Business focused Teams with ownership

(Biz)Microservices
Architecture aligns with Business Function

IT Delivery Automation
Cloud, CI/CD, hpaPaaS, RAD, Ops tooling

Business Driver
Supporting Measure
Pre-requisite
Organization, Process, IT Teams & IT Components are not independent from each other. Rather the opposite. This means the Business has a strong stake into Architecture.
Microservices For Alignment

Components of the Future will be more Business Like:

“Microservices interact as Actors in a Business process”

Maximize Probability that a new Feature request lands in one single Microservice?

Owned by one BizDevOps team

System Architect
Final Stage DevOps (Team has beeper for alarms): “Ops work keeps adding up for team with many Apps”

Risk that Innovation minded developers get bored?

Smaller and smaller capacity for new feature development→

This is the value drivers of DevOps

Automate
Innovate & Create Value

Feature Dev Available

Capacity of DevOps Team 1

Ops work App 1

App 1  App 2  App 3  App 4  App 5
Final Stage DevOps (Team has beeper for alarms):

“Ops work keeps adding up for team with many Apps”

Team is “Digitizing” its own processes

Increased Automation improves stability and frees up capacity

Ops work App 1

Feature
Dev
Available

Capacity of DevOps Team 1

Time

App 1  App 2  App 3  App 4  App 5

Increased Automation improves stability and frees up capacity
People often take 2-3 Roles in BizDevOps

BizDevOps Team – of 4-10 People

- PO, BA, Biz SME, Test, Data Admin, Support
- SA, Process, Agile Coach, Lead Dev, Test
- DevTest | OpsTest | OpsDeploy
Origins and Evolution of DevOps

• DevOps starts from a Developer’s revolt against Agile
  • Too much process, no real ownership, not Agile on business level
  • The rest of the organization is not aligned with agile

• BizDevOps now Expands Agile to Business, Management and Operations
  • It addresses some weaknesses with a central IT department

• But this requires Changes in HR, Teams and Organization
  • Enterprise DevOps now driven from CIO/CEO level
  • Perfectly suited for building Microservices and using Mendix
DevOps Focus Areas

Mode 2, Release Innovation

- Mode 2 with Agile Mindset
  - Start with DevOps for Mode 2
  - Evolve into larger Scope

RAD & Microservices

Architecture:
Move away from SOA and Monoliths

Corporate Change Program

Full Organizational Change
- Join all Teams & Priorities, use IT4IT
  - DevOps, Microservices and CICD

Automate and Measure

Tooling & Automation:
CICD – Auto-testing - Monitoring
Cloud – IT Tooling – Value Measure
How does this transition happen?
And how do I make it BizDevOps?
How does IT4IT compare with DevOps?

More Central Control & Standardization

Local Control & Decentralized Decision Making

Lean Culture

More Automation

Standard Framework & Automation Tools
→ Can enable DevOps teams to work in the Business Domains again
→ improved Business & IT alignment

20% of developers become even more hard-core techie?

80% become more business oriented?

BizDevOps
We need to avoid Chaos
We need some standards
And some common components
Likely Evolution: Maximize Autonomy of Biz Units

Centralize IT 2005

Corporate Biz Management

- Bus Domain
- Bus Domain
- Bus Domain

Central IT, Mgr = CIO

Dev

Ops

DevOps in IT Department 2015

Corporate Management With lead CIO

- Bus Domain
- Bus Domain
- Bus Domain

Central IT

Doing mostly IT DevOps

BizDevOps Full Org by 2025

Corporate Management With CIO & Roadmap

- IT4IT Domain
- Bus Domain
- Bus Domain

IT DevOps

Biz DevOps

Biz DevOps

Decreasing complexity of IT?
Levels for DevOps Structures

**Enterprise**
- Platform, Infrastructure, Tooling
- Portfolio, Road-map, Business Case
- Governance, Review & Assistance

**Program**
- Large Programs, Clear Objectives
- Operational Improvements
- Legacy Migration

**Innovate**
- Smaller projects, Discovery Mode, Very Agile
- Innovation, Design Thinking, UX focus
- Close co-operation with the Business

**Enable**

**Deliver**

**Discover**
Business Domains, Delegation and what’s left of Ops

Domains with delegated responsibility having Tribes Specialized at supporting a specific Domain

Cross polinate ideas & increase cooperation between tribes

Common Automation and IT Management Tools

Manage fluctuating resource needs
**Governance: Enterprise – Programs – Teams**

**FOCUS**
- Funding & Strategic return
- Program alignment
- Portfolio and Roadmap
- Transformation risk
- Platform & Common Components
- Guidelines and Epic level Review

**PROGRAM LIFECYCLE**
- Right Program owner
- Right Architect
- Right Budget

**PRODUCT LIFECYCLE**
- Right Team
- Right PO
- Right App

**LEADERS**
- Mx Platform Owner
- Enterprise Architect

- Program owner
- Solution Architect
- Scrum Master

- Product owners
- Lead Developer

**Focus Areas**
- Resources & Planning
- Guide, Assist, Review
- Coordinate & Communicate
- ROI

- Scope
- Time
- Cost
- Quality

**Enterprise board**

**Programs**
- Program 1
- Program 2

**Teams**
- Team A
- Team B
- Team C
- Team D

**Business Sponsorship**
Full SAFe
• Microservices
• Andreas
• Architect
Definition of Micro Services
(James Lewis & Martin Fowler)

“The Microservice architecture style provides an approach to:

• Build larger applications as a suite of smaller services (=IT components or Apps), where each service:
  • Is built around a business capability,
  • Runs its own process,
  • Communicate via a light weight mechanism,
  • And is independently deployable by an automated deployment machinery”

https://martinfowler.com/microservices/#what
Definition means …
(James Lewis & Martin Fowler)

We split a large system in smaller independent pieces that work together.

We call that a Microservice Architecture System.
Definition means …
(James Lewis & Martin Fowler)

With autonomous services that all fill a business purpose

Communicating via services or deep-links
Overview Comparison

Traditional System
One large System with internal Modules

Monolith Application

Microservices System
Independent Specific Components

Federated Application

SOA “System”
Shared Layered Components

Layered Application

Monolith System

GUI

logic

process

database

Microservices System

Micro-service

Micro-service

Micro-service

ESB

UI Layer

Logic Layer

Process Layer

Data Layer

Back-end System
There is More to Microservices than Size!

<table>
<thead>
<tr>
<th>SOA and Monoliths</th>
<th>Microservices</th>
</tr>
</thead>
<tbody>
<tr>
<td>“Share as much as Possible”</td>
<td>“Share as little as Possible”</td>
</tr>
<tr>
<td>“Do not Copy Data or Functions”</td>
<td>“Copy what I need to do my Job”</td>
</tr>
<tr>
<td>“Align teams with Technology”</td>
<td>“Cross functional Teams”</td>
</tr>
<tr>
<td>“Own a layer”</td>
<td>“Own a business function”</td>
</tr>
</tbody>
</table>

“Business alignment starts by understanding each other, and ends with working together”
Why Microservices?

• Because Size Matters

[Diagram showing the relationship between user groups, developers, scope size, and microservices]

- Small Scope: Very Agile
- Medium Scope: Agile
- Large Scope: Reasonable Flexibility
- Monolith:
  1. Too many user groups
  2. Too many developers
  3. Too many dependencies
  4. Slow App generation
  5. Lack of specialization

Unacceptable speed
Real-life “Autonomous Units”

• Knowledge is better copied between individuals
• Merged with the experience and needs of each student
• Creates new valuable combinations of information and function
Years of Architecture habits to Un-learn

“We have been learning to maximize re-use and minimize duplication.”

What is the cost of sharing a component?
The Microservice key word is Autonomy

Building complete Business Functions as Microservices
Copy only the functions and data we really need to work properly
Some times for very stable functions to re-use as a service
Autonomous Microservices for Core Systems

Traditional/Monolith

Large Scope

1 System Architect

Core Business System

CBA/Microservices

Business Owners

1 System Architect

Process Architect
No Silver Bullet anymore – and not One Option

Splitting in smaller pieces has a cost…

...so I should only Split things when I am sure I know the right reason to split

The Reason is often Functional and or Business oriented, so I need help

Considering more than one option is a necessity to weigh pros and cons

In DevOps Culture decisions are taken as a Team

**Autonomy**
of DevOps teams and Business Units

**Process**
Independent Phases

**Re-use**
of very stable Functionality

**Operability**
DevOps team manages

**Scalability**
Fully performant small pieces
Process & Business Flexibility in the Centre

- Process Architects
- Business Owners
- Solution Architects

Microservices Architecture

Business & IT Alignment
- DevOps Efficiency
- Digital Evolution

Autonomy
- of DevOps teams and Business Units

Re-use
- of very stable, functional parts

Operability
- DevOps team manages

Scalability
- Fully performant, small pieces

Process
- Independent Phases
Microservices Thinking is Good for Mendix

- Microservices is 100% in line with the DevOps trend
- Microservices has become ‘the new thing’ after SOA
- *Mendix makes Microservices by Default*
Business Features usually have Data, Logic & UI

- **Mendix makes Microservices by Default**

  - One Deployable
  - One Model

  Some level of “Integration Test” and CICD when checking into team-server
Mendix Platform – Pre-built Automation

Mendix Style Microservice System

- HPA PaaS
- Buy Automation

Java Style Microservice System

- Open Source
- Build Automation

5-10 Times Faster Results

Harder to align IT with Business People
Mendix for Core Systems
Adaptive Governance per System Type:

**System of Innovation**
(life-span 1-5 years)
- Maximum Speed of Development
- Minimal amount of Governance
- Fix things after peer Review

Think First
Focus on Quality

**System of Differentiation**
(life-span 3-10 years)
- Start with Architecture Workshop
- Maintain good Speed
- Establish reasonable Governance

Move fast
Change later

**Core System**
(life-span 5-30 years)
- Full Analysis phase in the beginning
- Full Governance, Testing & Documentation
- Disaster Recovery procedures

Analysis required
Constant review
Additional testing
Business Features usually have Data, Logic & UI

- Mendix makes Microservices by Default

- One Deployable
- One Model

Some level of “Integration Test” and CICD when checking into team-server
The Business & IT Alignment Explained

- Business is involved in IT delivery. DevOps teams stay with business units.
- Small team handles full life-cycle
- Automation reduces cost of having smaller components
- Conways law: Align components with organization

- Business & IT Alignment
- Ownership & Autonomy
- (Biz)DevOps
- (Biz)Microservices
- Less technical teams
- IT Delivery Automation, Cloud, CI/CD, aPaaS

Faster-than-competition Digital Evolution
Questions?

Andreas
Architect