Changing IT Delivery with DevOps and Microservices

Andreas Lennevi

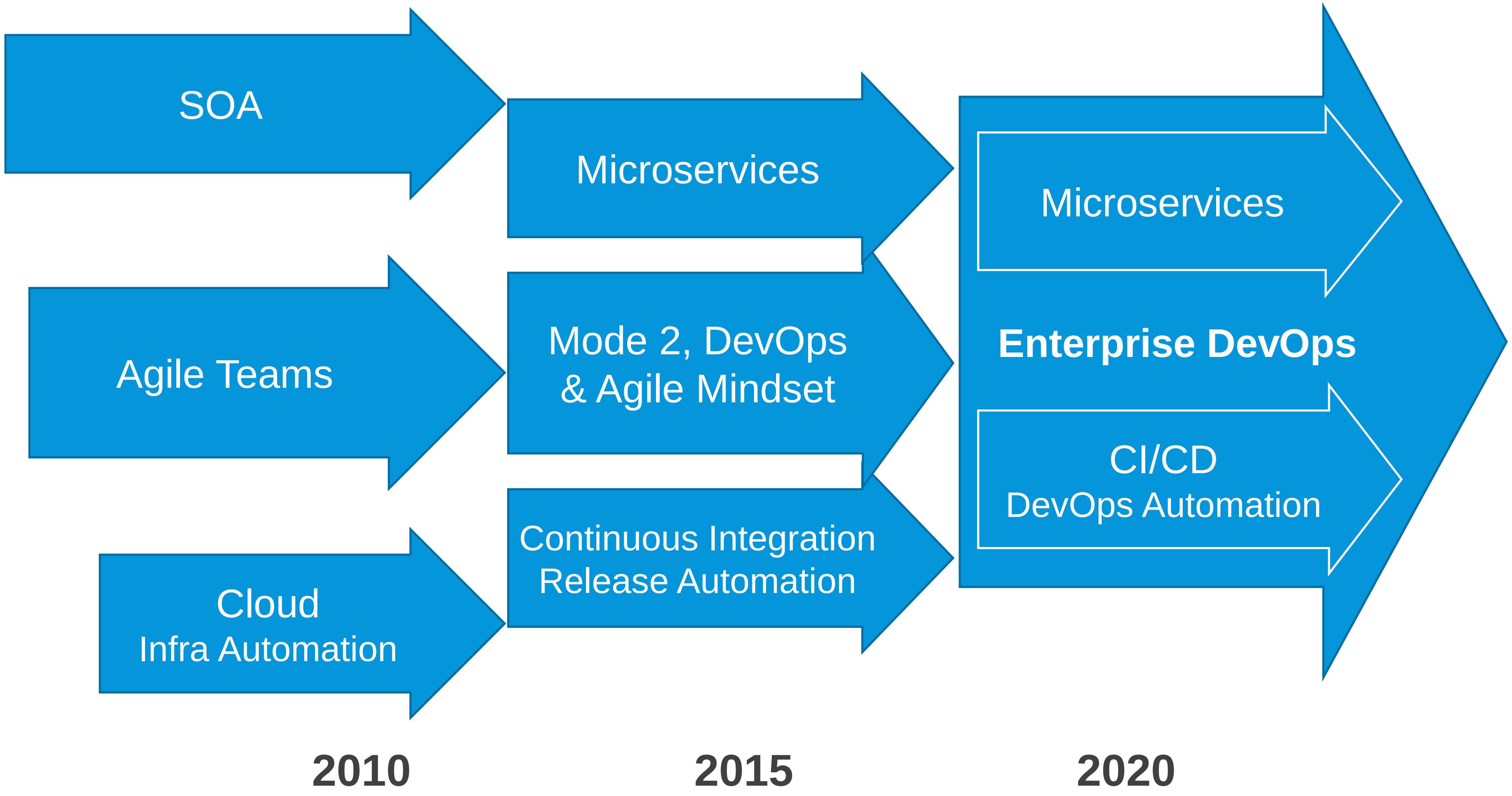
mx mendix

“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.***

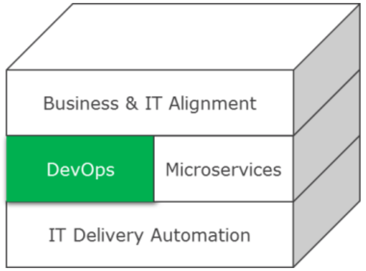
Trends merging into coherent Movement:



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

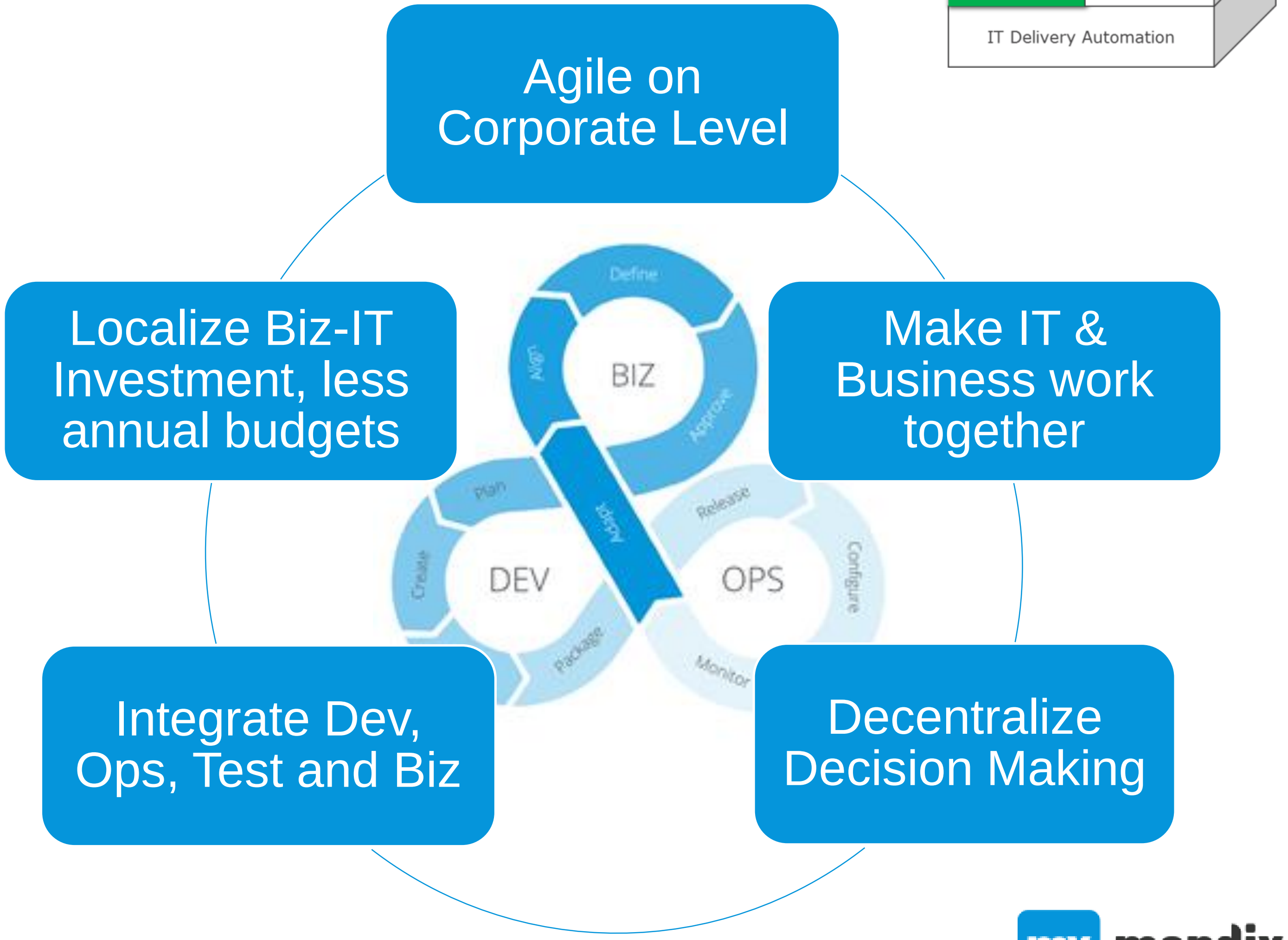
mx mendix

Enterprise DevOps

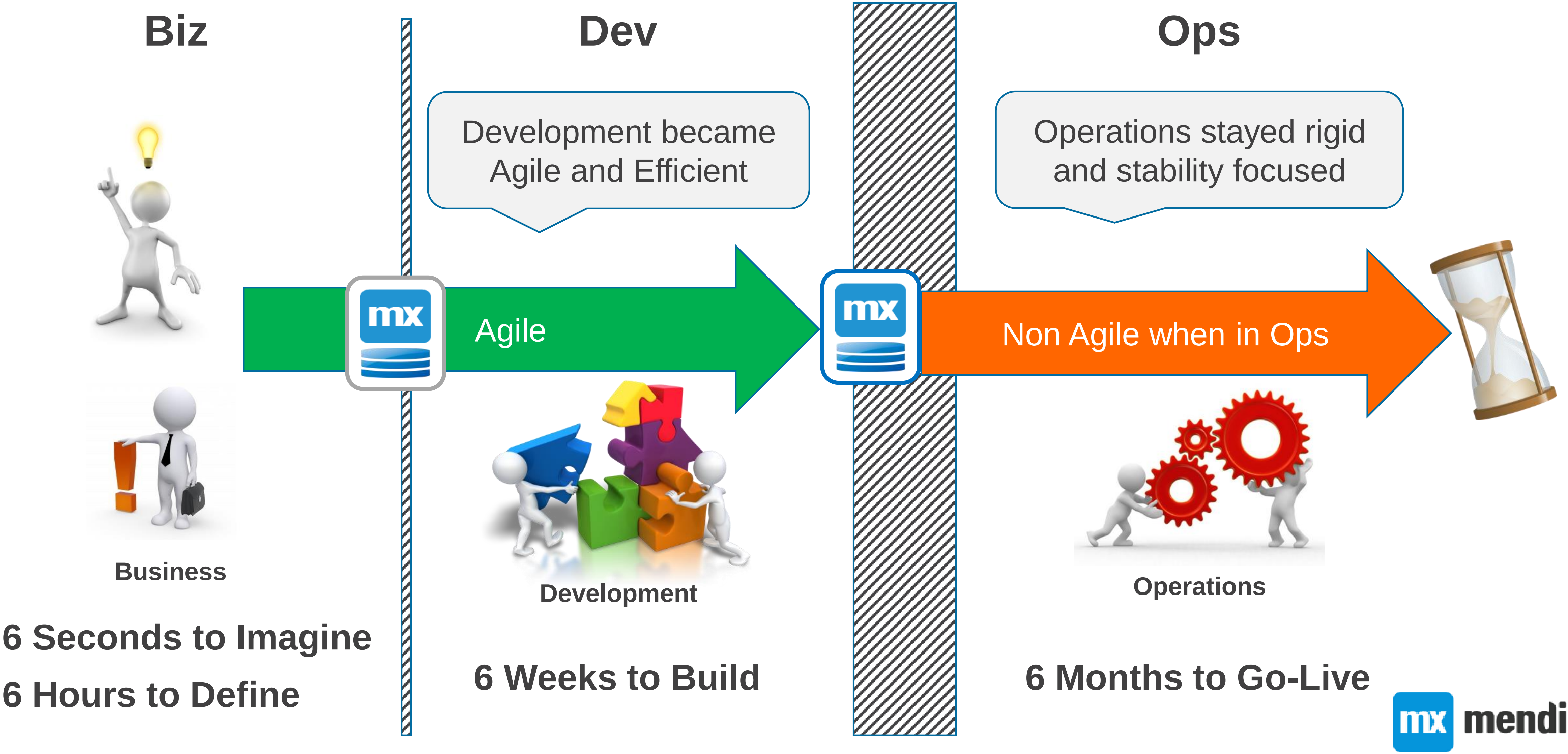


Why?

- Faster Product Delivery
- Manage Shifting Priorities
- More Value / Worked Hour


Agile on Corporate Level
Make IT & Business work together
Decentralize Decision Making
Integrate Dev, Ops, Test and Biz
Localize Biz-IT Investment, less annual budgets
Define
Align
BIZ
Approve
Plan
Create
DEV
Package
Adapt
Release
OPS
Configure
Monitor

Agile Development + Rigid Ops = Rigid IT



Biz

Dev

Ops

Development became Agile and Efficient

Operations stayed rigid and stability focused

Agile

Non Agile when in Ops

Business

Development

Operations

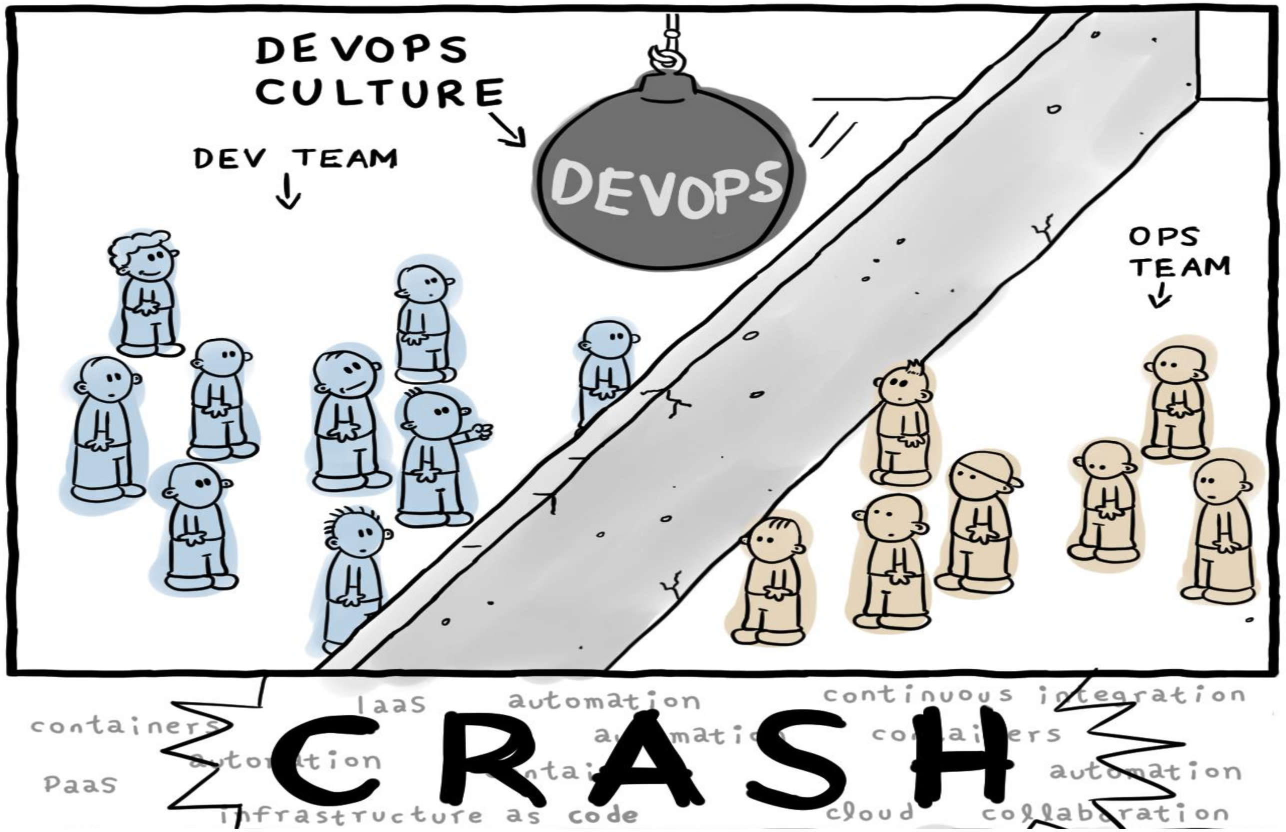
6 Seconds to Imagine

6 Hours to Define

6 Weeks to Build

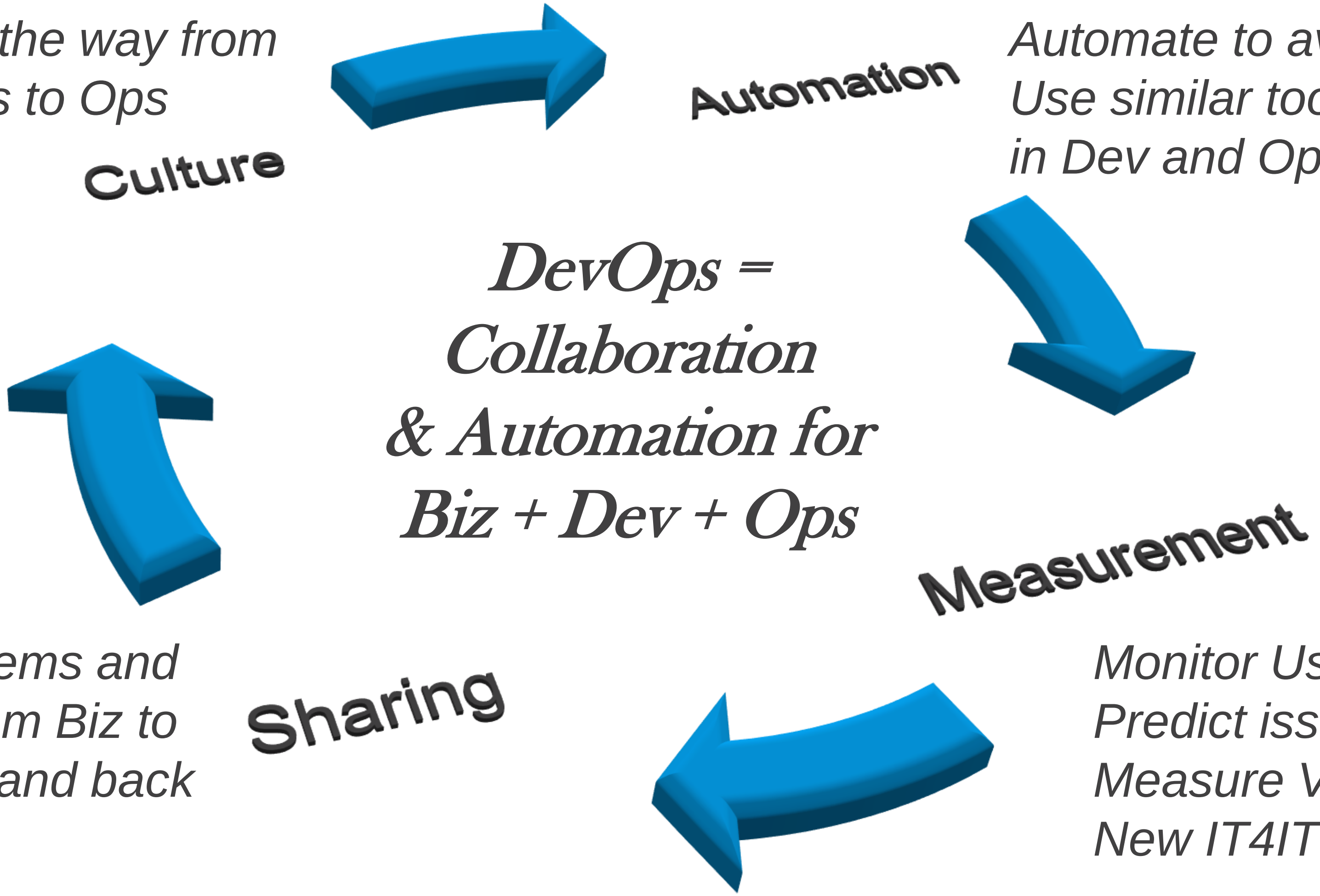
6 Months to Go-Live

DevOps – The Culture Wrecking Ball



DevOps Pillars

“CAMS”



Agile all the way from Business to Ops

Culture

Automation

Automate to avoid Ops work
Use similar tools and methods in Dev and Ops

DevOps =
Collaboration
& Automation for
Biz + Dev + Ops

Measurement

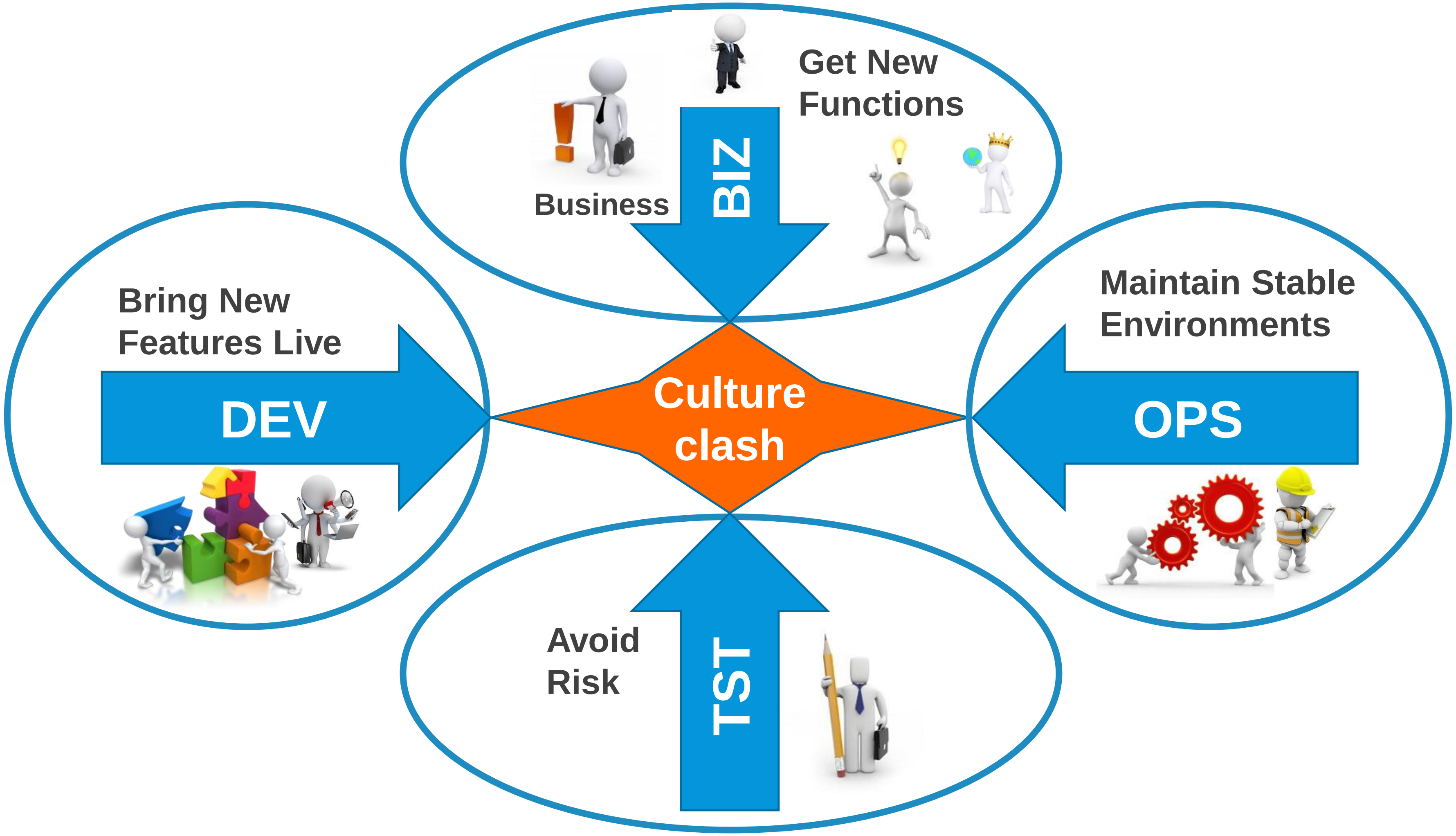
Monitor Use & Performance,
Predict issues in PRD
Measure Value & Feedback
New IT4IT type tools

Sharing

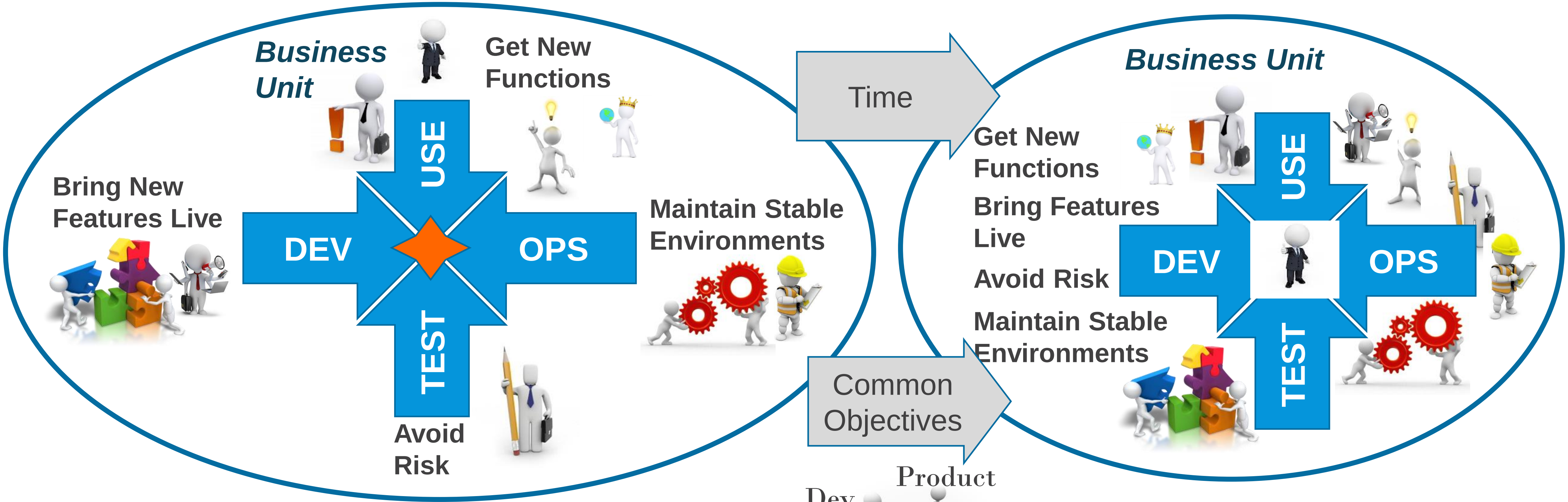
Share problems and solutions from Biz to Dev to Ops and back

“CAMS” by John Willis: [John Willis coining “CAMS”:]()

Conflicting Objectives of Departments



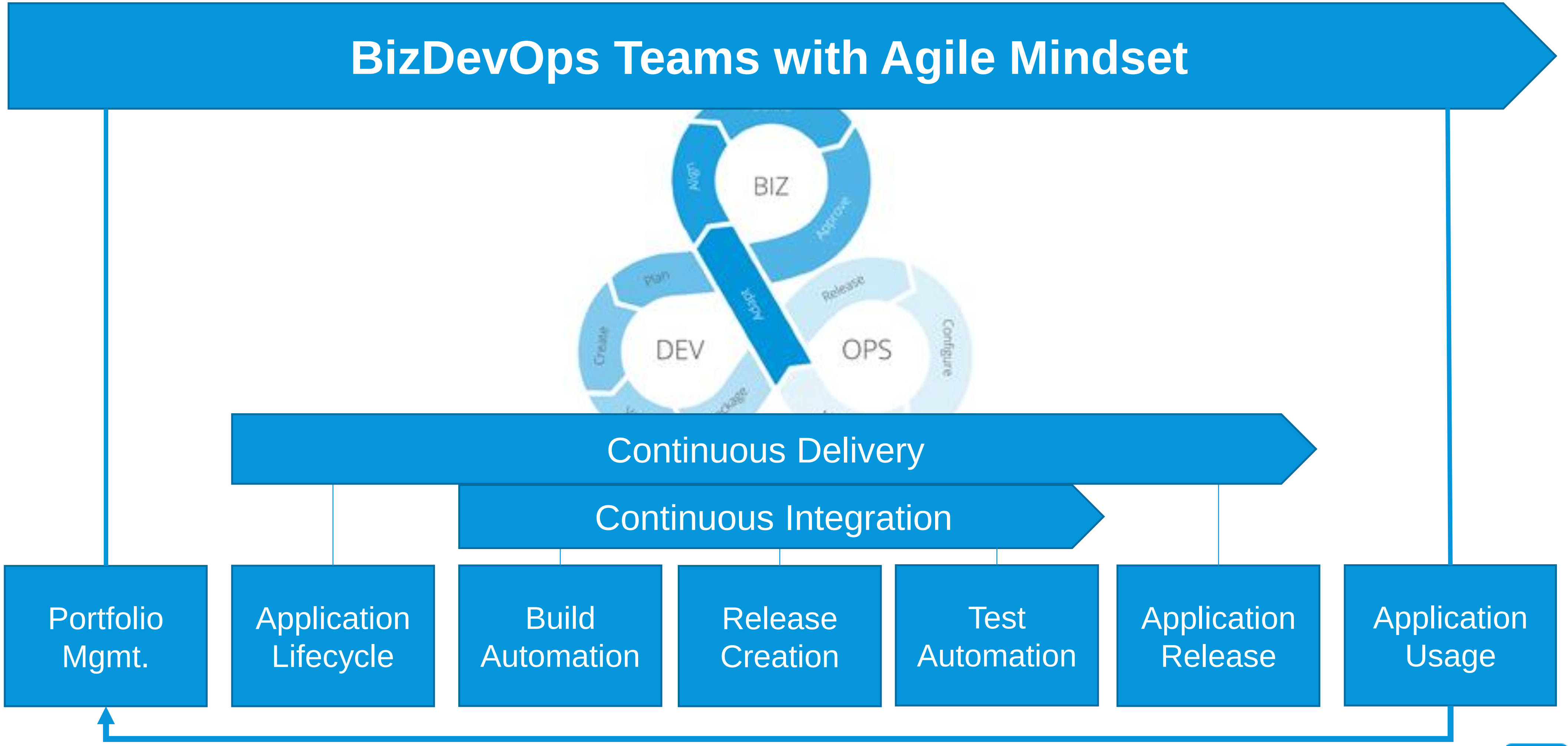
BizDevOps: Cooperating around Joint Objectives



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

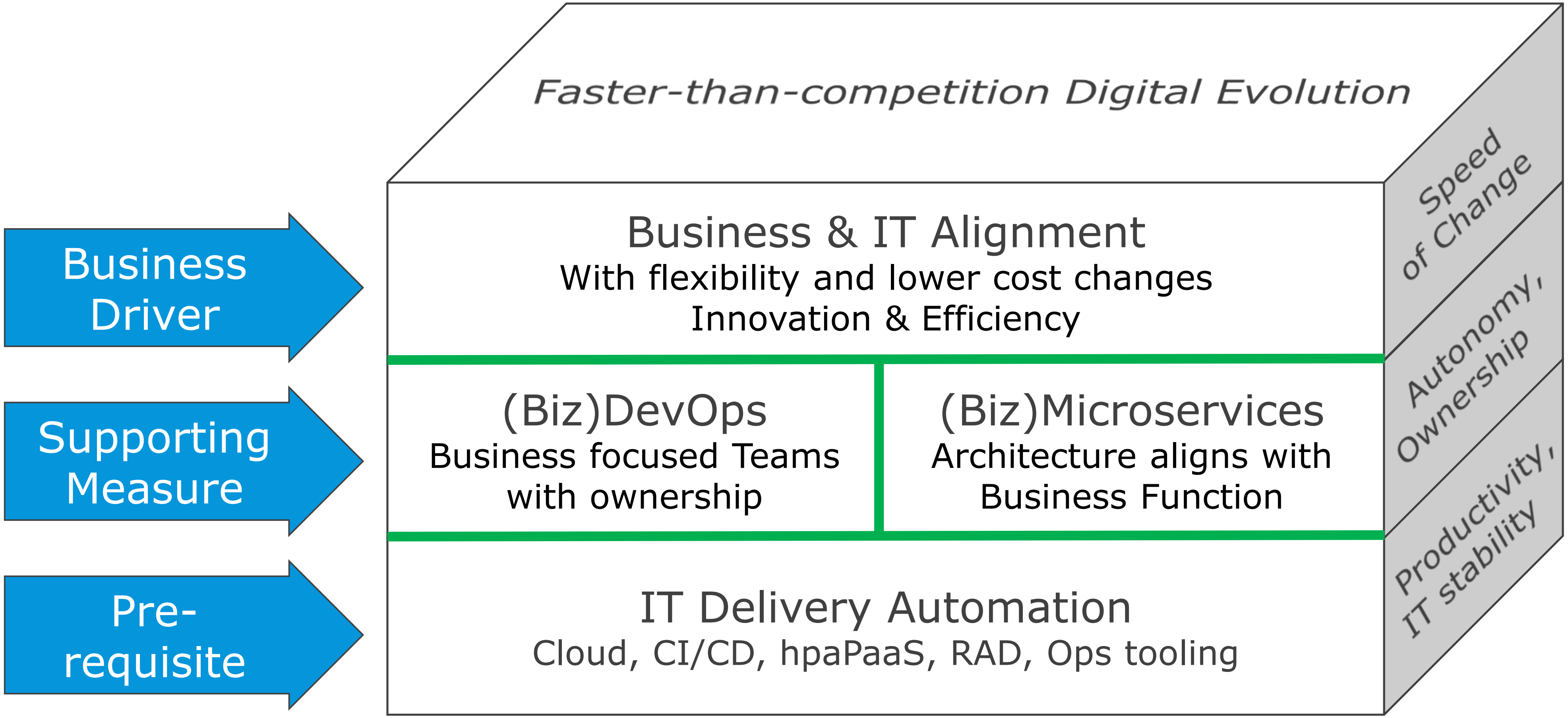
Release Creation

Test Automation

Application Release

Application Usage

Why is DevOps so Appealing:
Business & IT Alignement



Faster-than-competition Digital Evolution

Business Driver

Supporting Measure

Pre-requisite

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

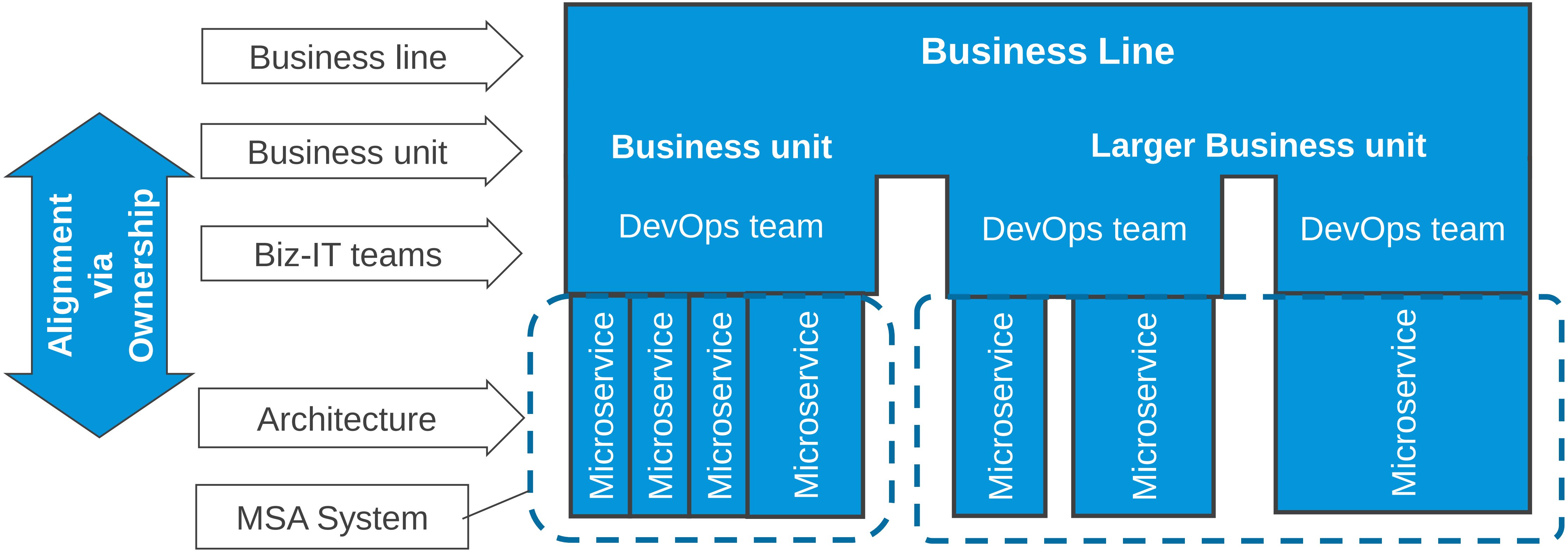
Speed of Change

Autonomy, Ownership

Productivity, IT stability

Business – IT Alignes by Maximum Ownership

By ownership and independence we get flexibility and innovation



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

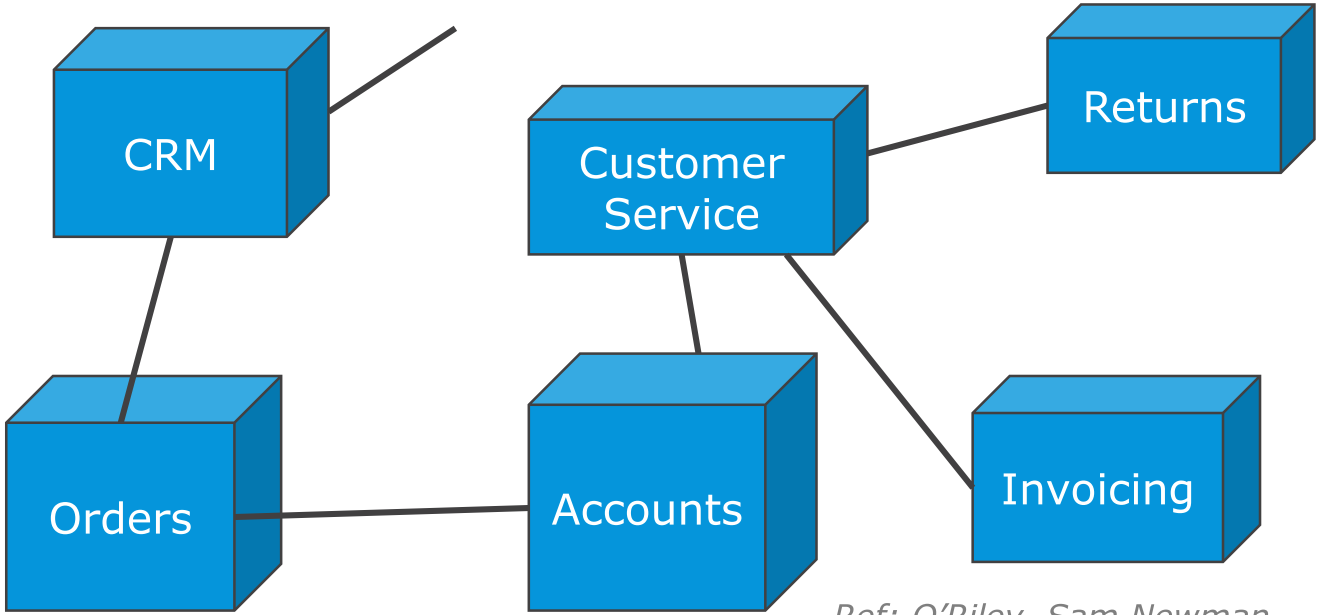
Components of the Future will be more Business Like:

“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

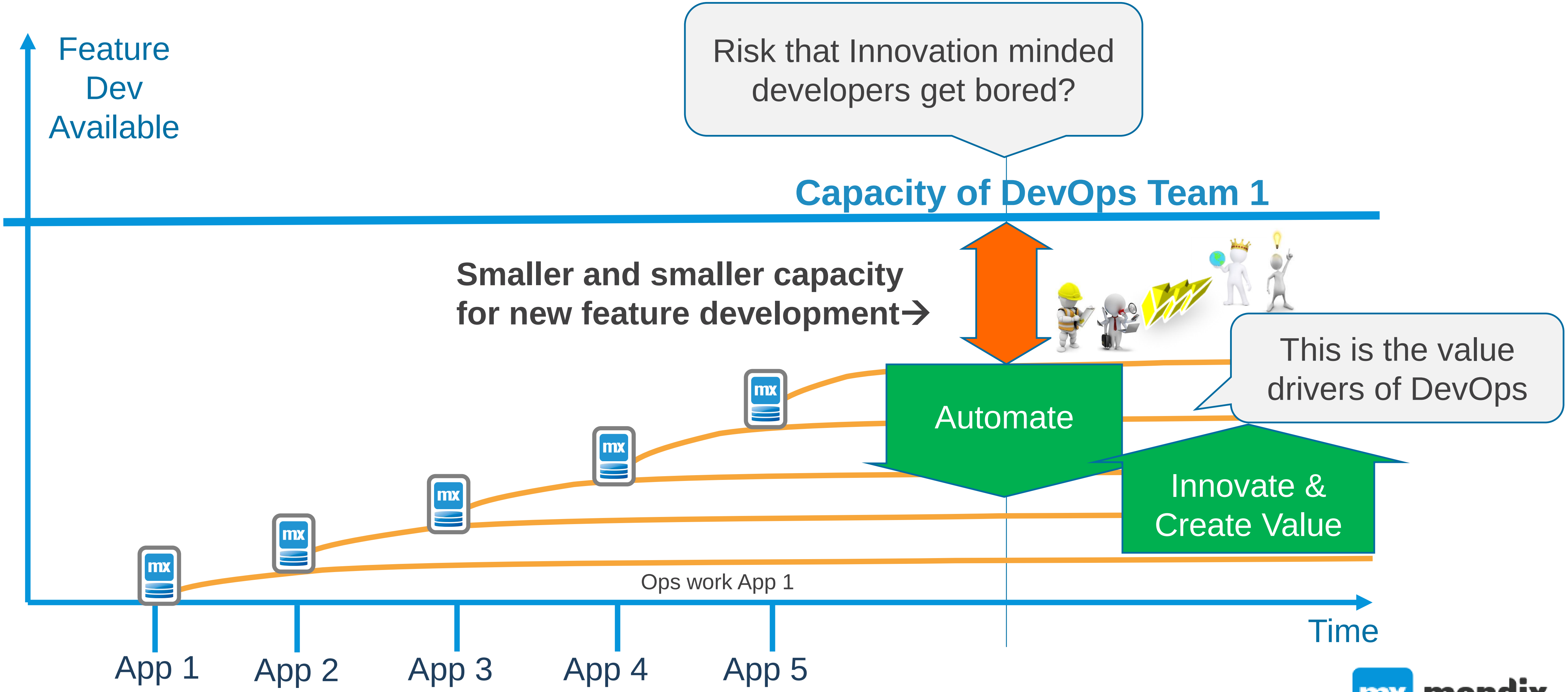
System Architect



Ref: O’Riley, Sam Newman

Final Stage DevOps (Team has beeper for alarms):

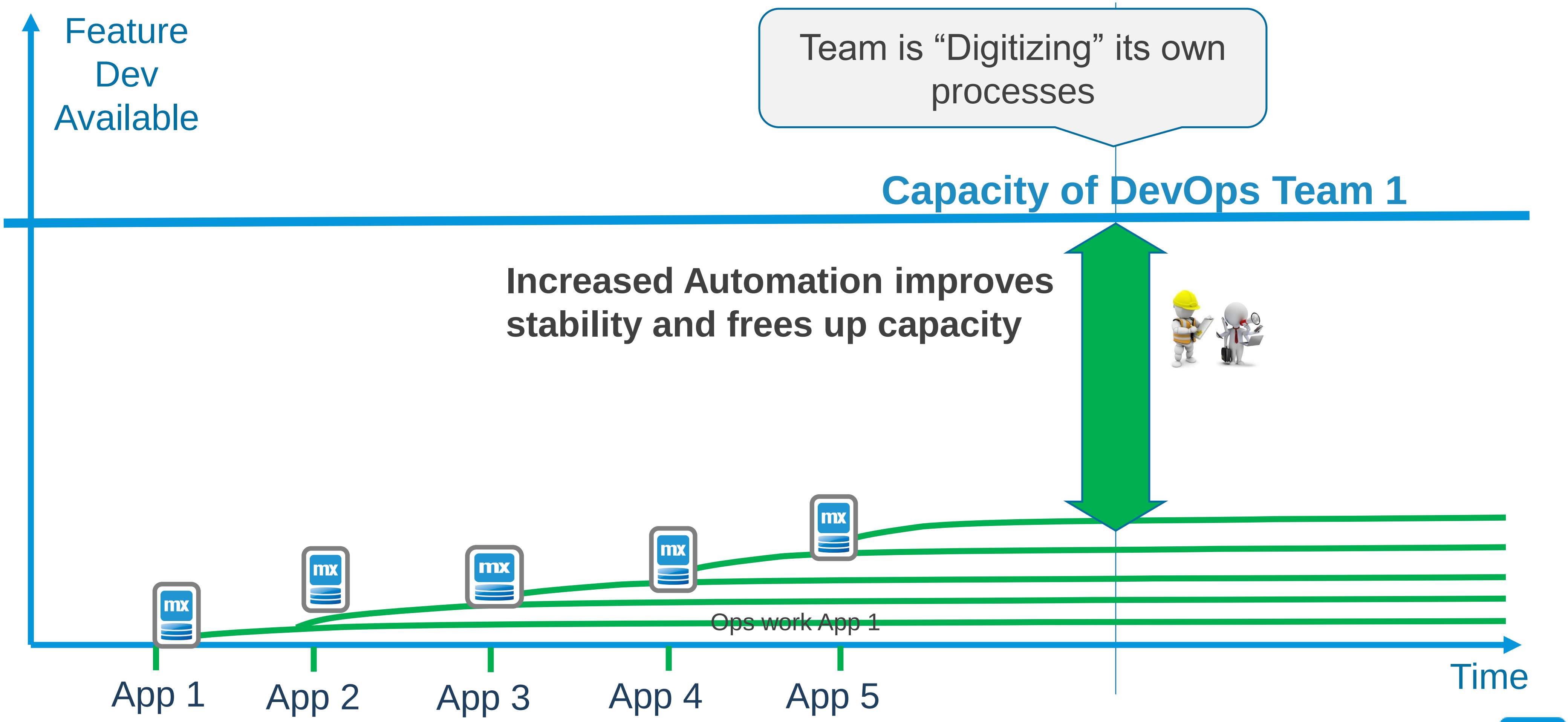
“Ops work keeps adding up for team with many Apps”


Feature
Dev
Available
Risk that Innovation minded developers get bored?
Capacity of DevOps Team 1
Smaller and smaller capacity for new feature development→
Automate
This is the value drivers of DevOps
Innovate & Create Value
Ops work App 1
Time
App 1
App 2
App 3
App 4
App 5
mx

mx mendix

Final Stage DevOps (Team has beeper for alarms):

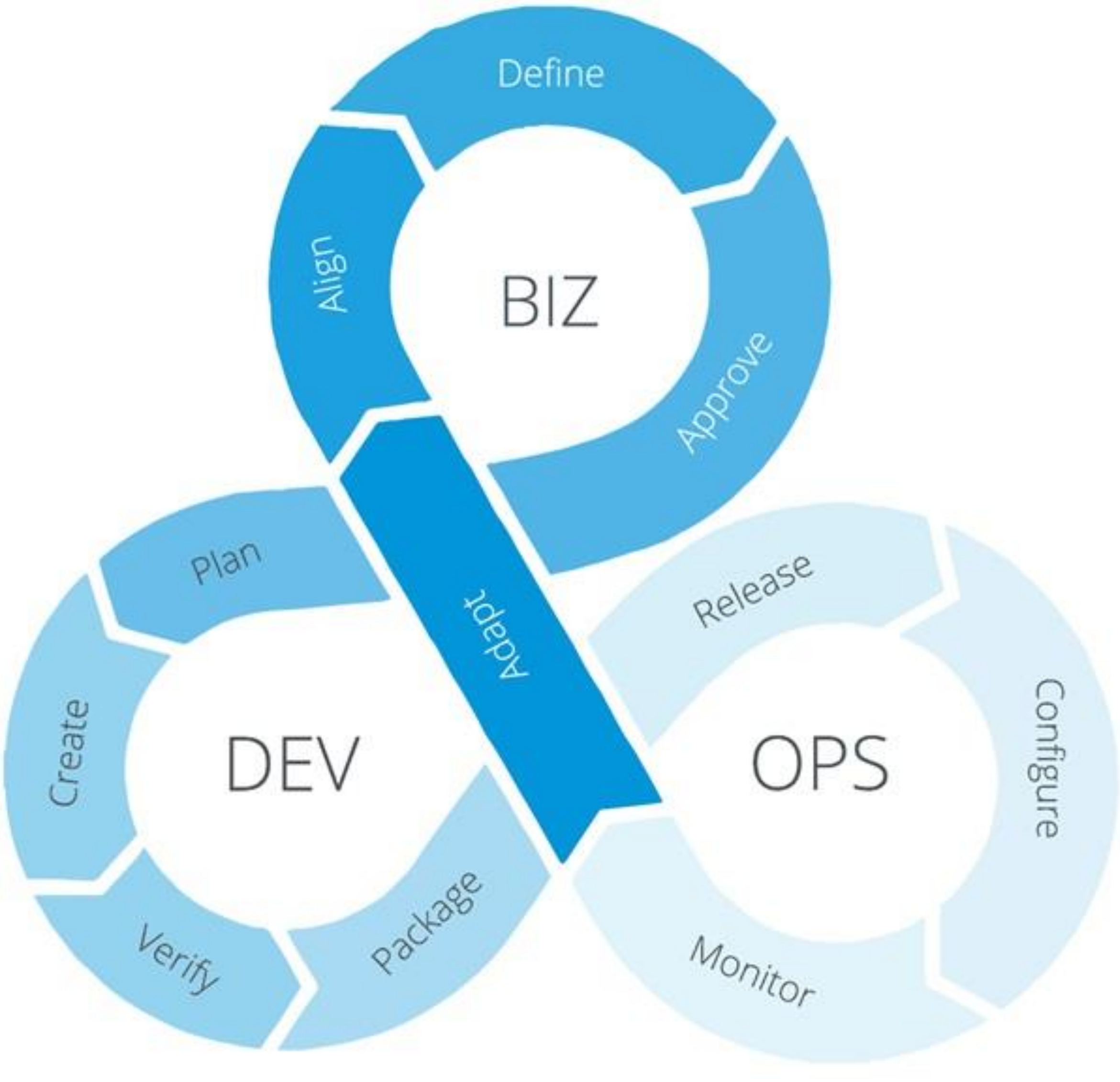
"Ops work keeps adding up for team with many Apps"



Team is "Digitizing" its own processes

Increased Automation improves stability and frees up capacity

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

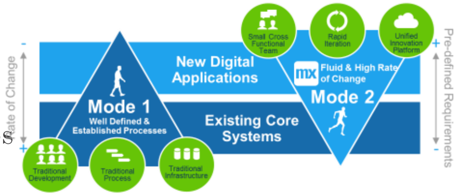
mx mendix

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

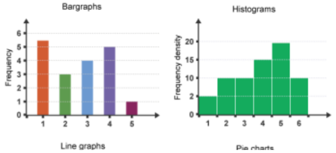
DevOps Focus Areas

Mode 2, Release Innovation



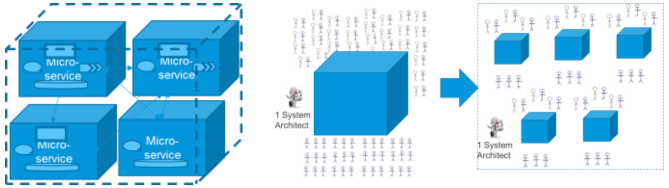
Mode 2 with Agile Mindset
Start with DevOps for Mode 2
Evolve into larger Scope

Automate and Measure



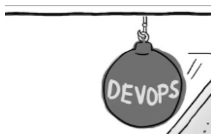
Tooling & Automation:
CICD – Auto-testing - Monitoring
Cloud – IT Tooling – Value Measure

RAD & Microservices



Architecture:
Move away from SOA and Monoliths

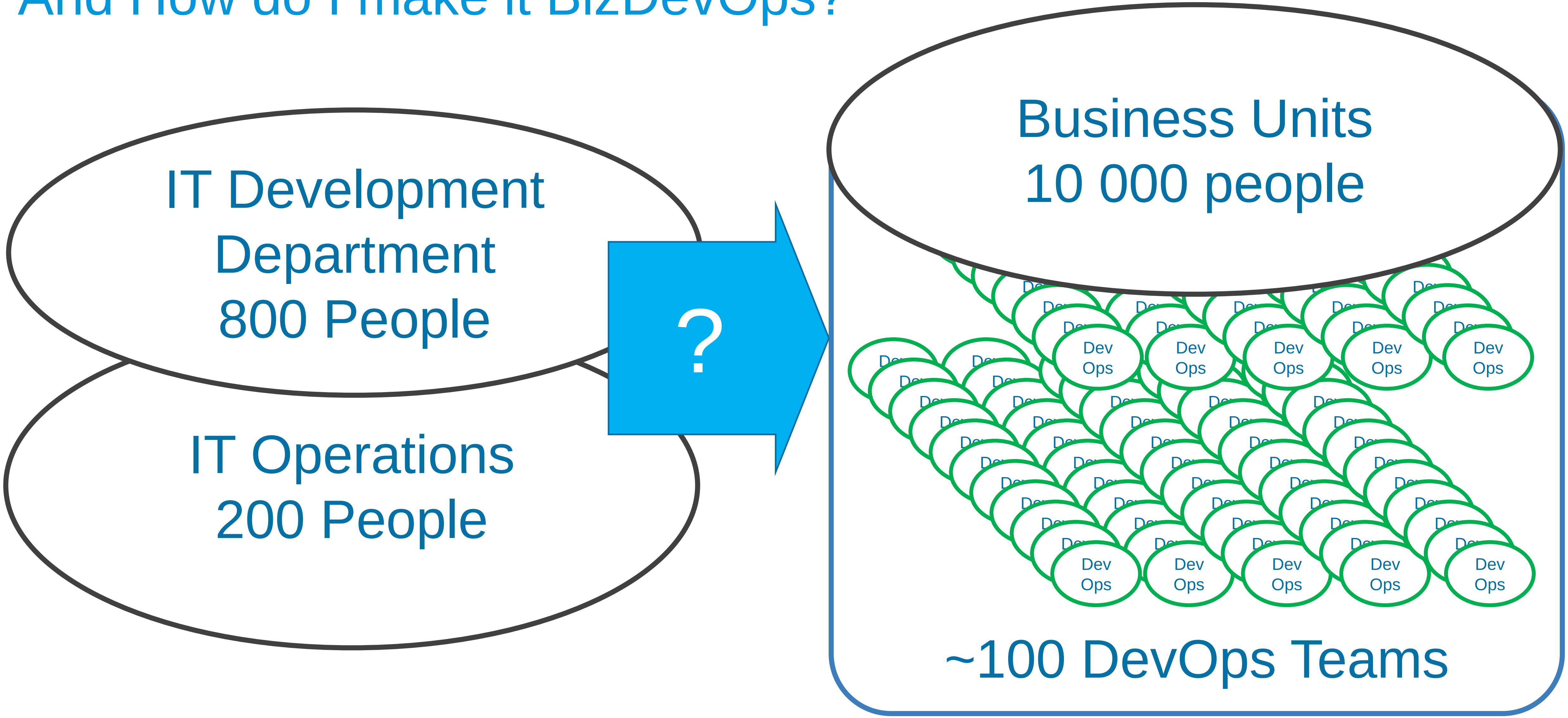
Corporate Change Program



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

How does this transition happen?

And How do I make it BizDevOps?



IT Development Department
800 People

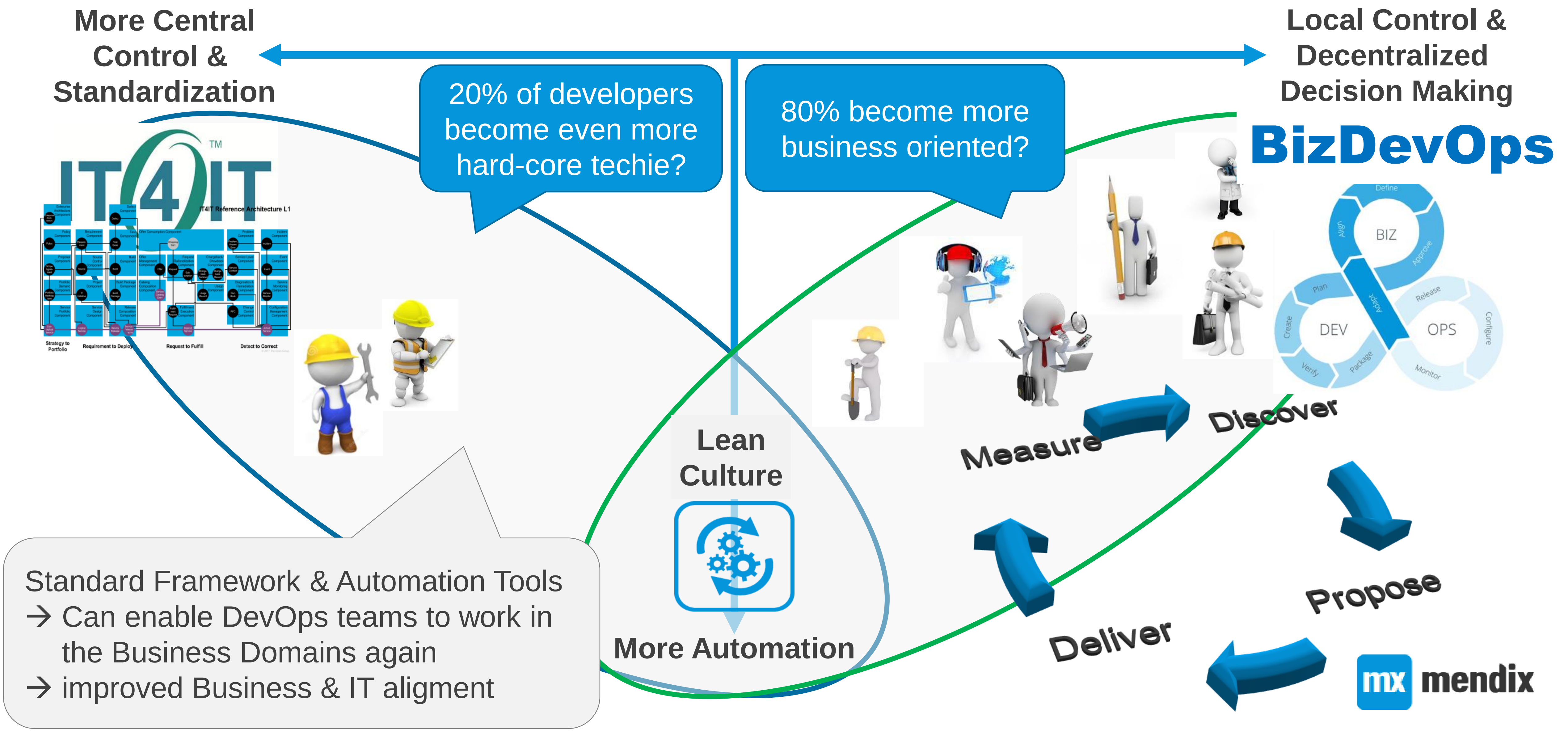
IT Operations
200 People

?

Business Units
10 000 people

~100 DevOps Teams

How does IT4IT compare with DevOps?



More Central Control & Standardization

Local Control & Decentralized Decision Making

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

80% become more business oriented?

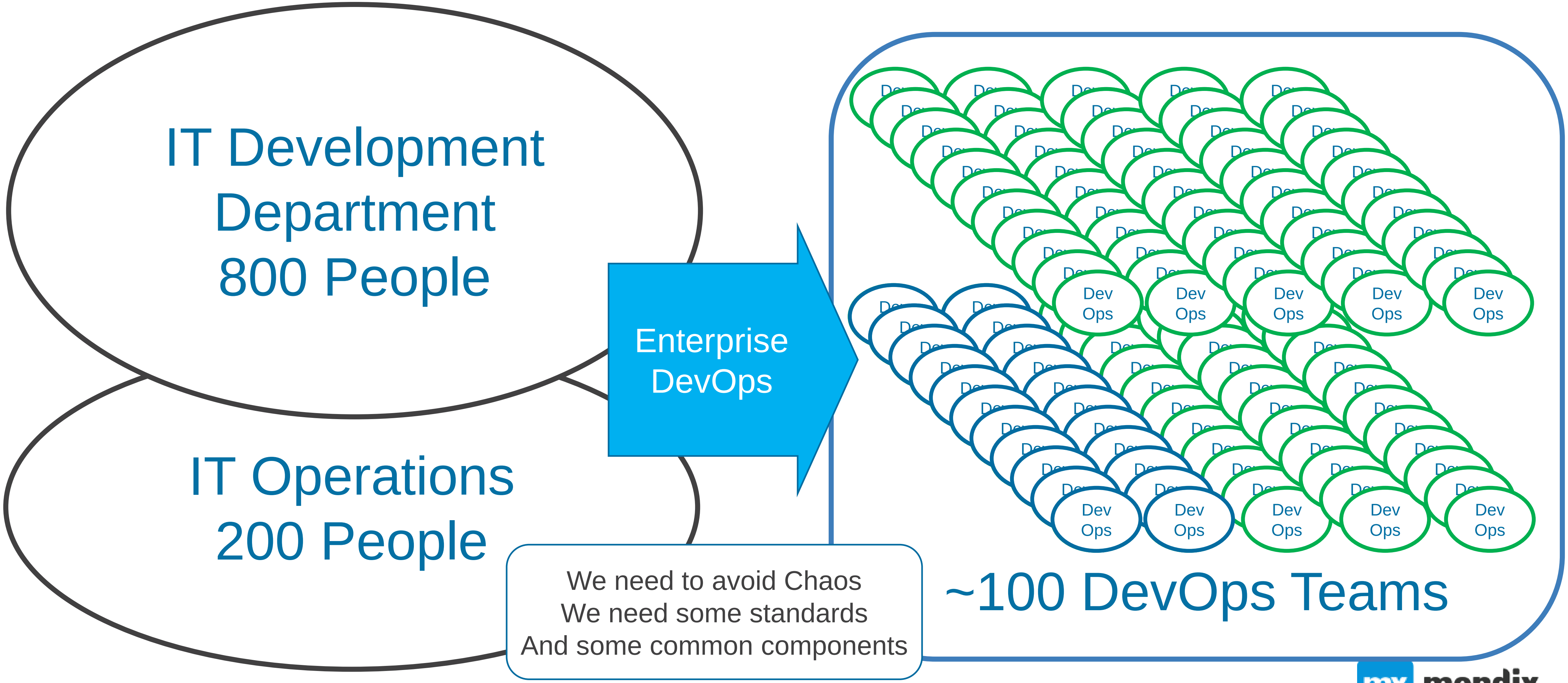
BizDevOps

Lean Culture

More Automation

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

Enterprise DevOps in Large Scale – Challenge:



IT Development Department 800 People

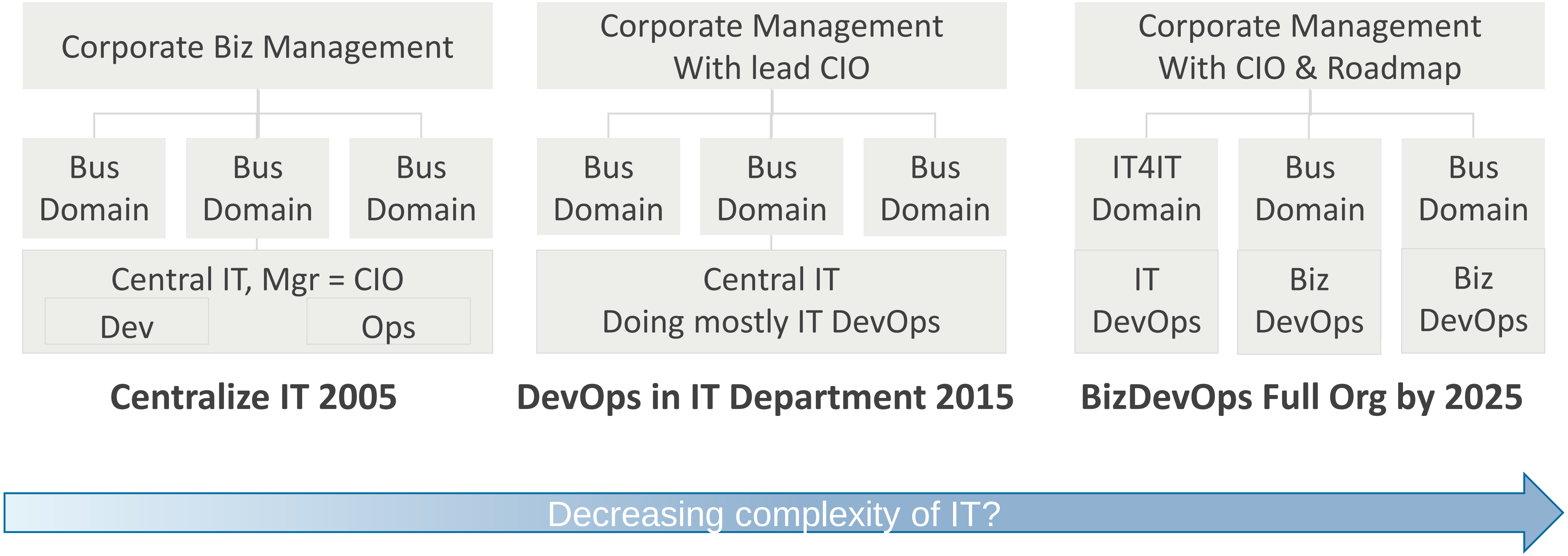
IT Operations 200 People

Enterprise DevOps

~100 DevOps Teams

We need to avoid Chaos
We need some standards
And some common components

Likely Evolution: Maximize Autonomy of Biz Units



Corporate Biz Management

Bus Domain | Bus Domain | Bus Domain

Central IT, Mgr = CIO

Dev | Ops

Centralize IT 2005

Corporate Management
With lead CIO

Bus Domain | Bus Domain | Bus Domain

Central IT
Doing mostly IT DevOps

DevOps in IT Department 2015

Corporate Management
With CIO & Roadmap

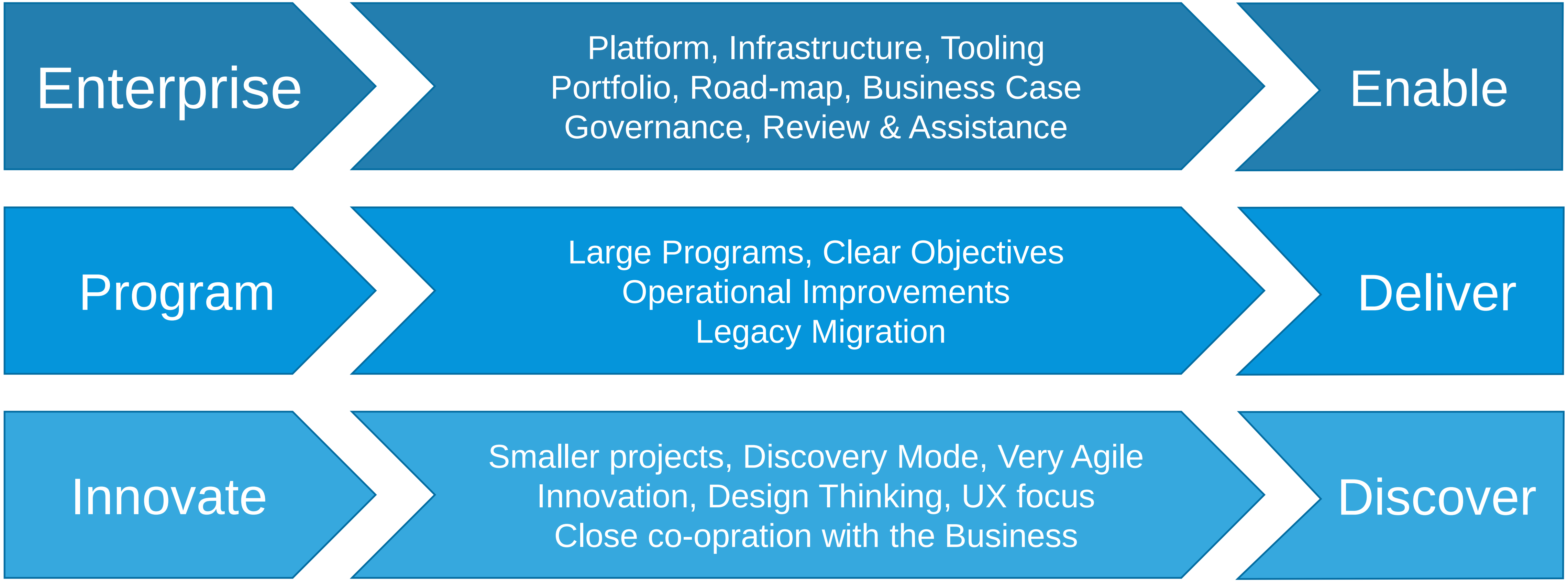
IT4IT Domain | Bus Domain | Bus Domain

IT DevOps | Biz DevOps | Biz DevOps

BizDevOps Full Org by 2025

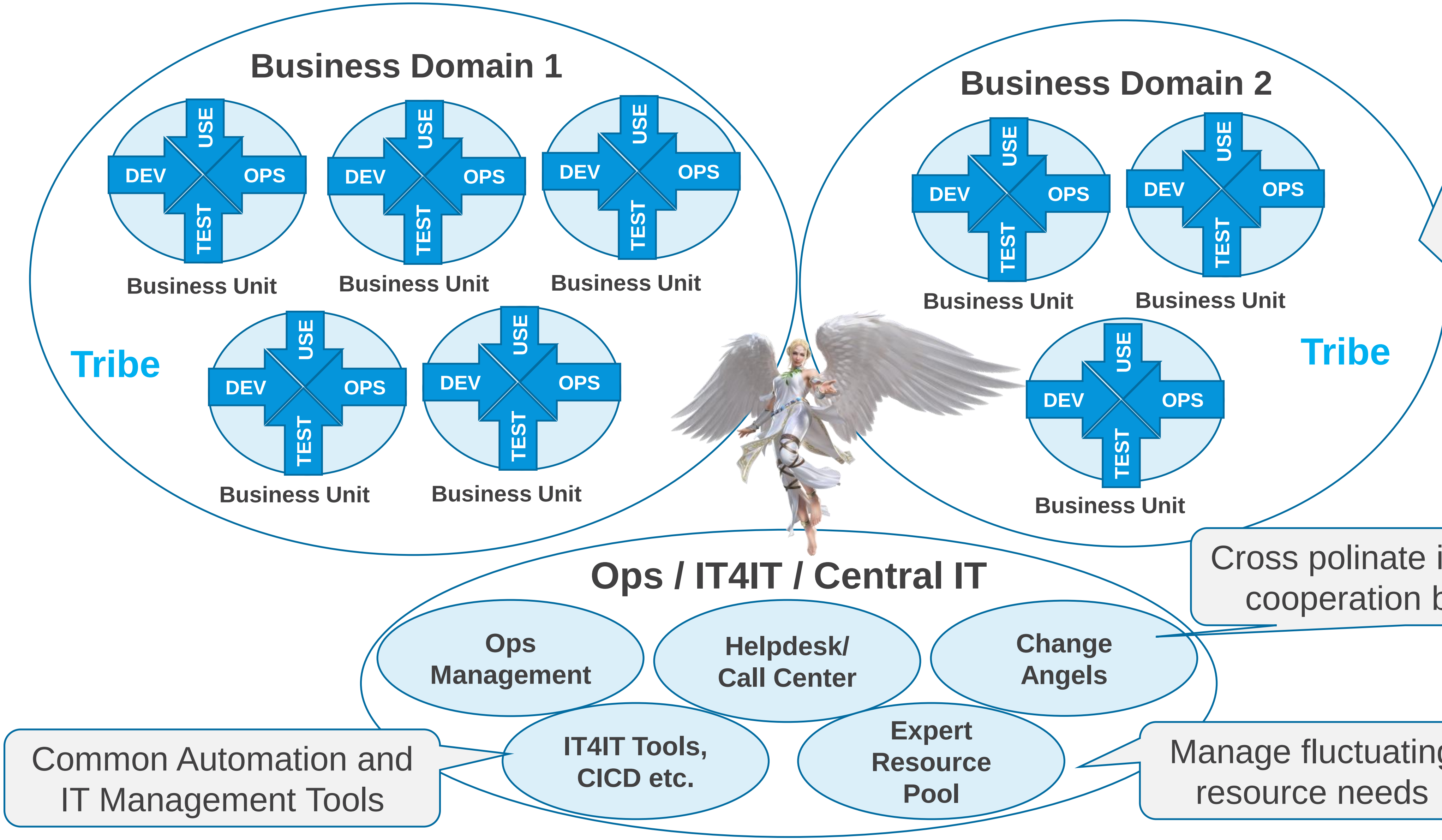
Decreasing complexity of IT?

Levels for DevOps Structures



Enterprise	Platform, Infrastructure, Tooling Portfolio, Road-map, Business Case Governance, Review & Assistance	Enable
Program	Large Programs, Clear Objectives Operational Improvements Legacy Migration	Deliver
Innovate	Smaller projects, Discovery Mode, Very Agile Innovation, Design Thinking, UX focus Close co-opration with the Business	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

mx mendix

Governance: Enterprise – Programs – Teams

LEADERS

- Mx Platform Owner
- Enterprise Architect

FOCUS

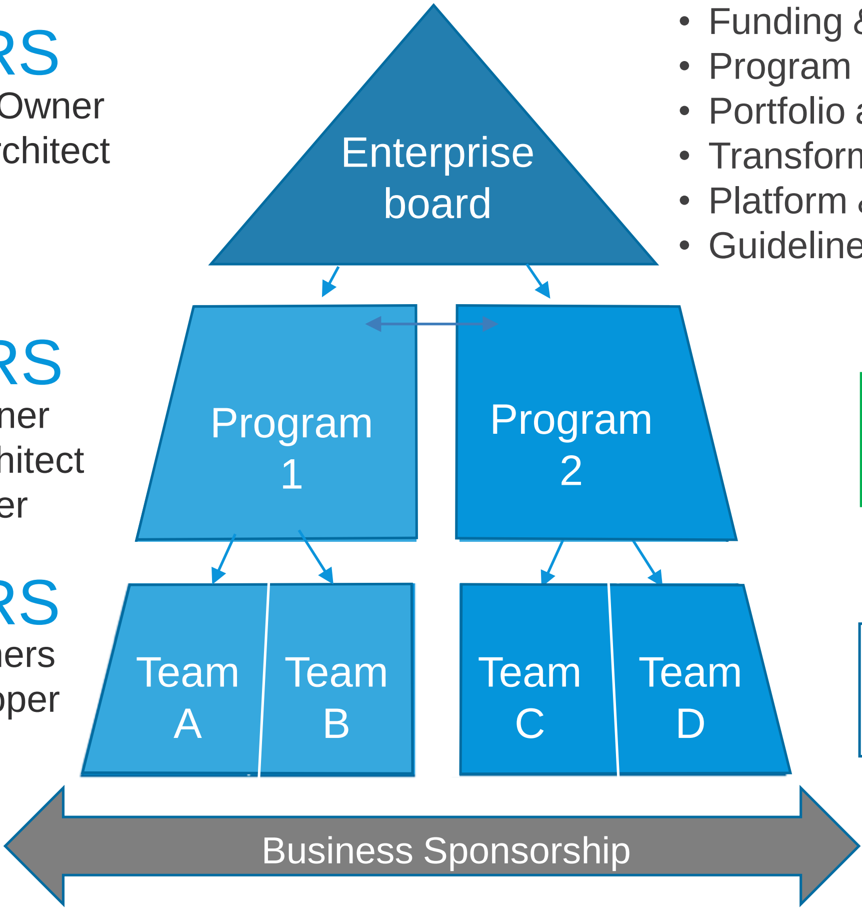
- Funding & Strategic return
- Program alignment
- Portfolio and Roadmap
- Transformation risk
- Platform & Common Components
- Guidelines and Epic level Review

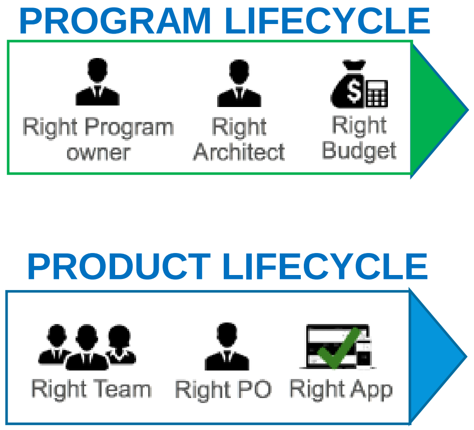
LEADERS

- Program owner
- Solution Architect
- Scrum Master

LEADERS

- Product owners
- Lead Developer





FOCUS

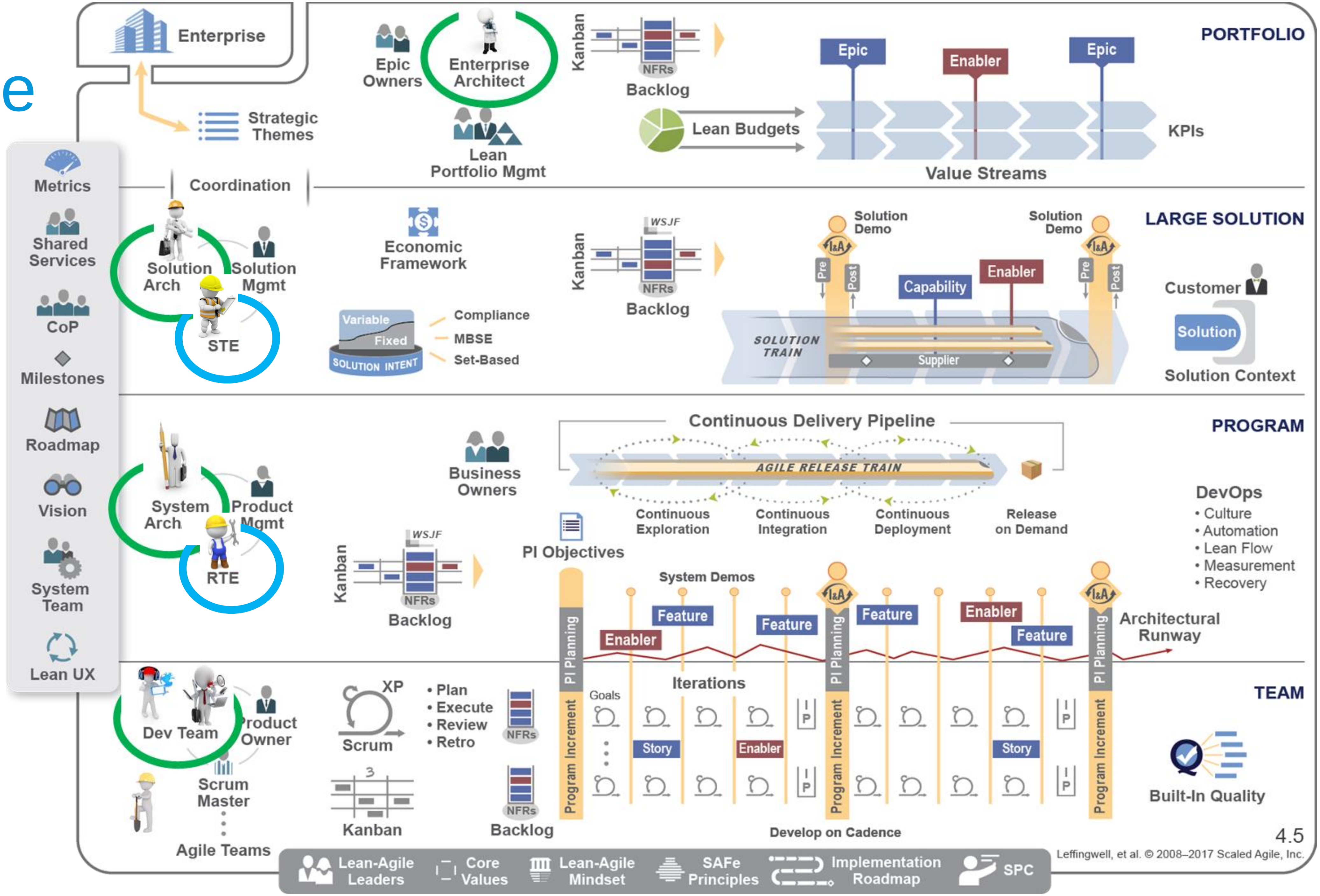
- Resources & Planning
- Guide, Assist, Review
- Coordinate & Communicate
- ROI

FOCUS

- Scope
- Time
- Cost
- Quality

mx mendix

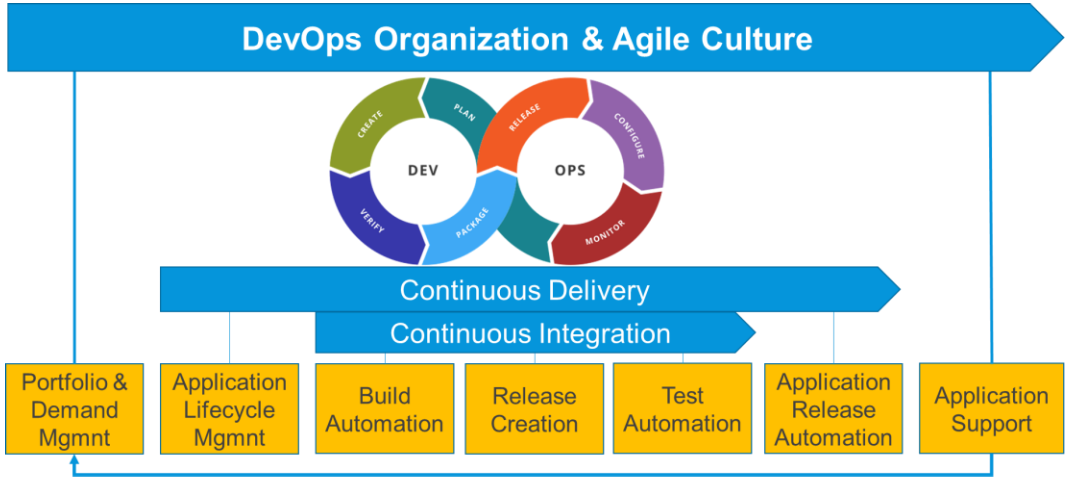
Full SAFe



Enterprise
Strategic Themes
Epic Owners
Enterprise Architect
Lean Portfolio Mgmt
Kanban
NFRs
Backlog
Lean Budgets
Epic
Enabler
Epic
KPIs
Value Streams
PORTFOLIO
Metrics
Shared Services
CoP
Milestones
Roadmap
Vision
System Team
Lean UX
Coordination
Solution Arch
Solution Mgmt
STE
Economic Framework
Variable
Fixed
SOLUTION INTENT
Compliance
MBSE
Set-Based
Kanban
WSJF
NFRs
Backlog
Solution Demo
Solution Demo
I&A
Pre
Post
Capability
Enabler
SOLUTION TRAIN
Supplier
LARGE SOLUTION
Customer
Solution
Solution Context
System Arch
Product Mgmt
RTE
Business Owners
Kanban
WSJF
NFRs
Backlog
Continuous Delivery Pipeline
AGILE RELEASE TRAIN
Continuous Exploration
Continuous Integration
Continuous Deployment
Release on Demand
PROGRAM
DevOps
• Culture
• Automation
• Lean Flow
• Measurement
• Recovery
PI Objectives
System Demos
Feature
Enabler
Feature
Feature
Enabler
Feature
PI Planning
Architectural Runway
Iterations
Goals
Story
Enabler
Story
P
Program Increment
Develop on Cadence
TEAM
Built-In Quality
Dev Team
Product Owner
Scrum Master
Agile Teams
XP
Scrum
• Plan
• Execute
• Review
• Retro
NFRs
Kanban
Backlog
Lean-Agile Leaders
Core Values
Lean-Agile Mindset
SAFe Principles
Implementation Roadmap
SPC
4.5
Leffingwell, et al. © 2008–2017 Scaled Agile, Inc.

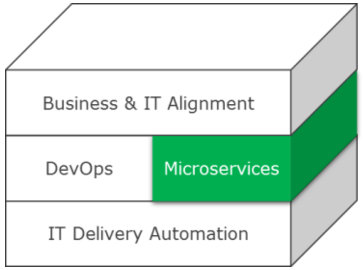
- Microservices

- Andreas
- Architect



Definition of Micro Services

(James Lewis & Martin Fowler)

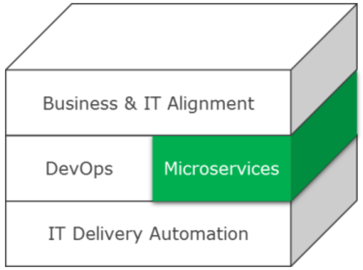


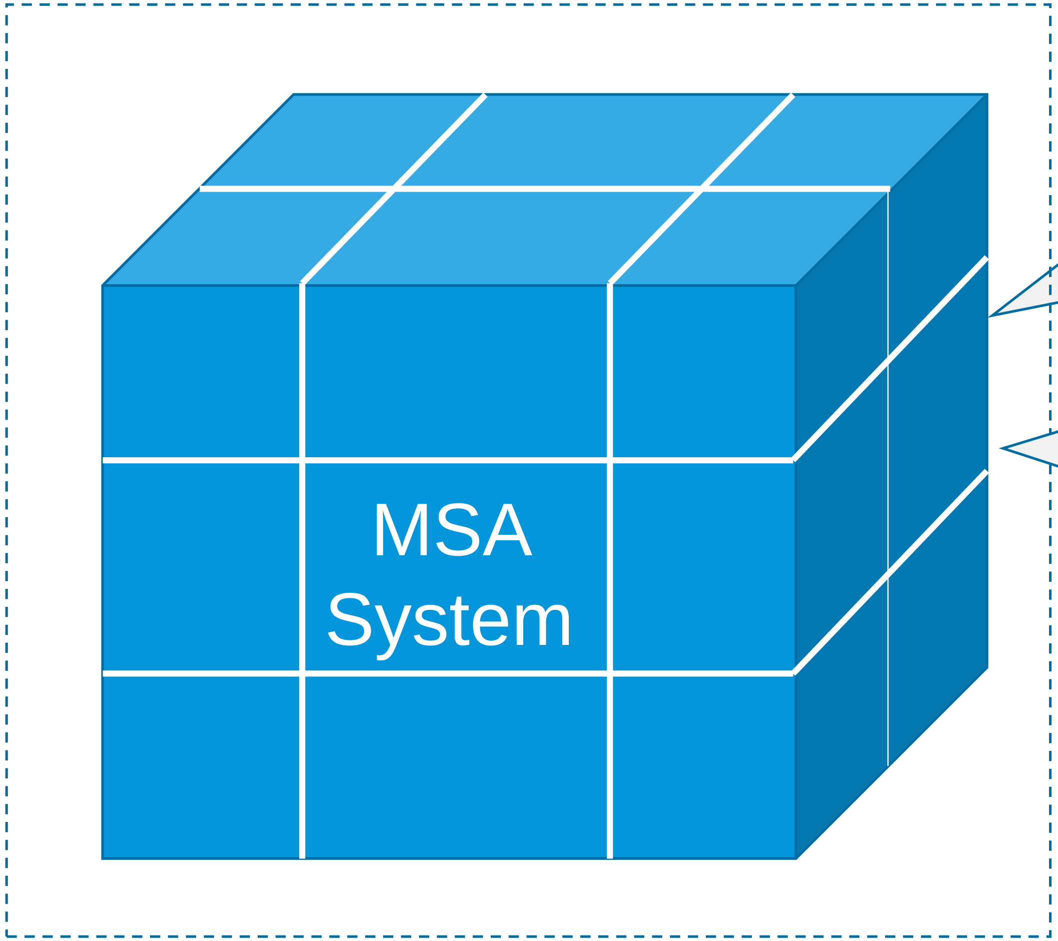
“The Microservice architecture style provides an approach to:

- *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)

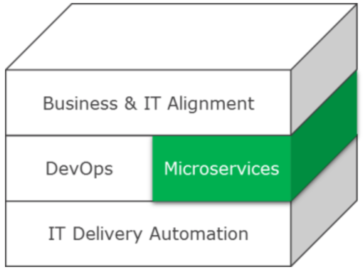


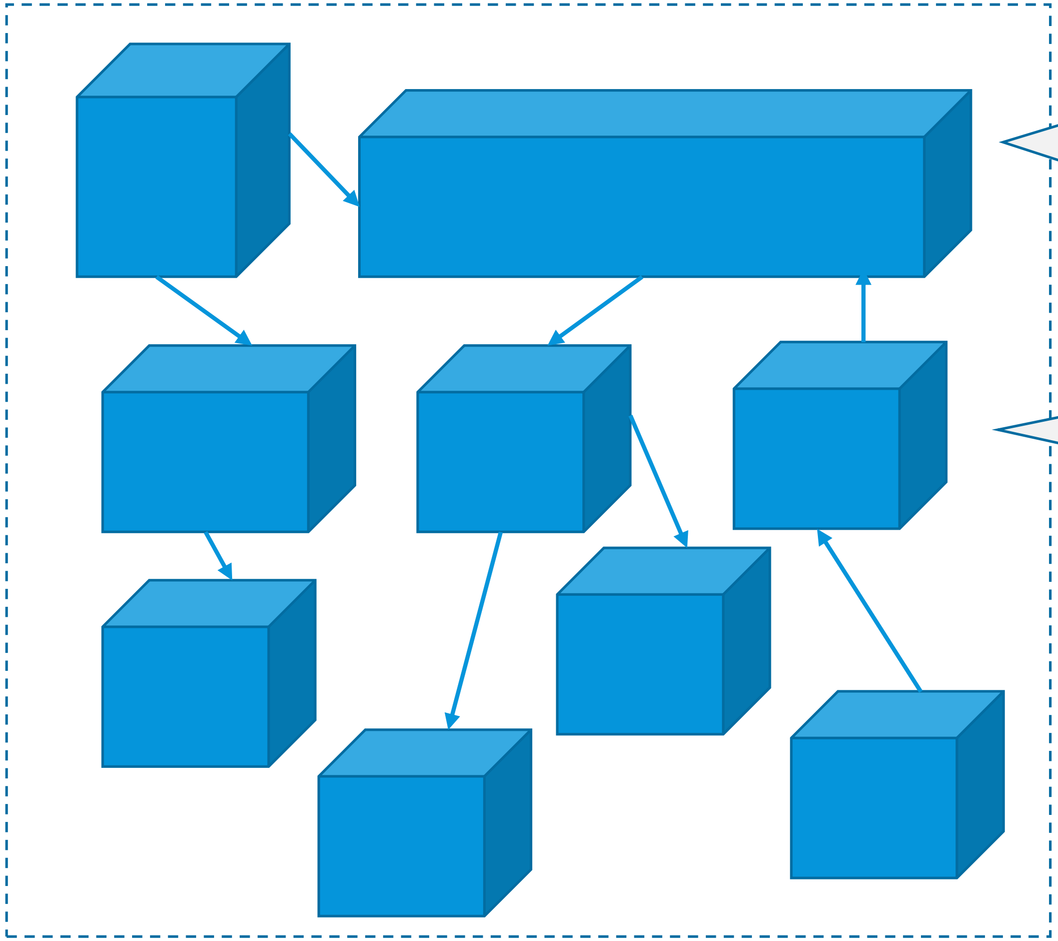


We split a large system in smaller independent pieces that work together

We call that a Microservice Architecture System

Definition means …
(James Lewis & Martin Fowler)

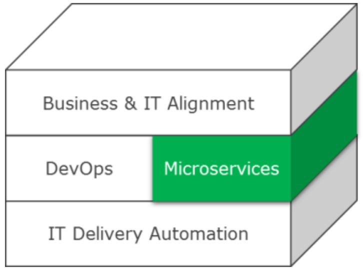




With autonomous services that all fill a business purpose

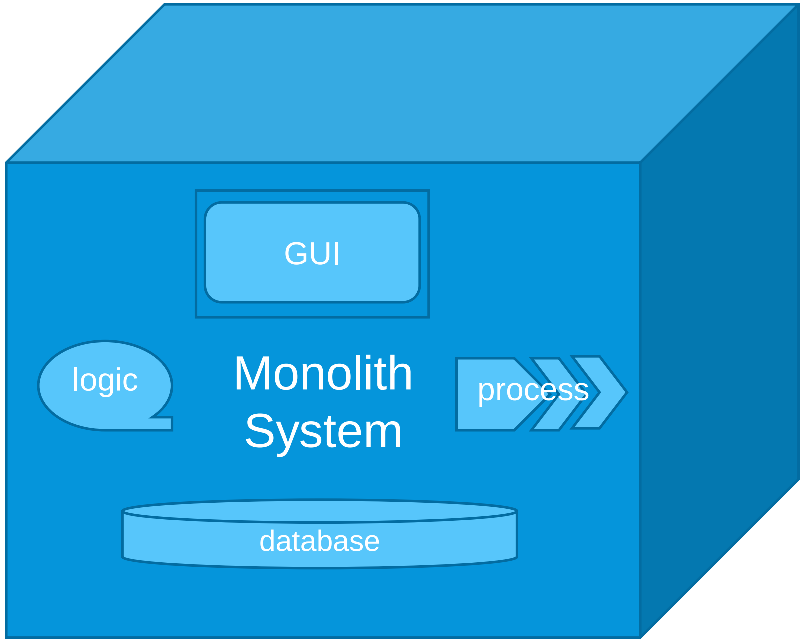
Communicating via services or deep-links

Overview Comparison



Traditional System

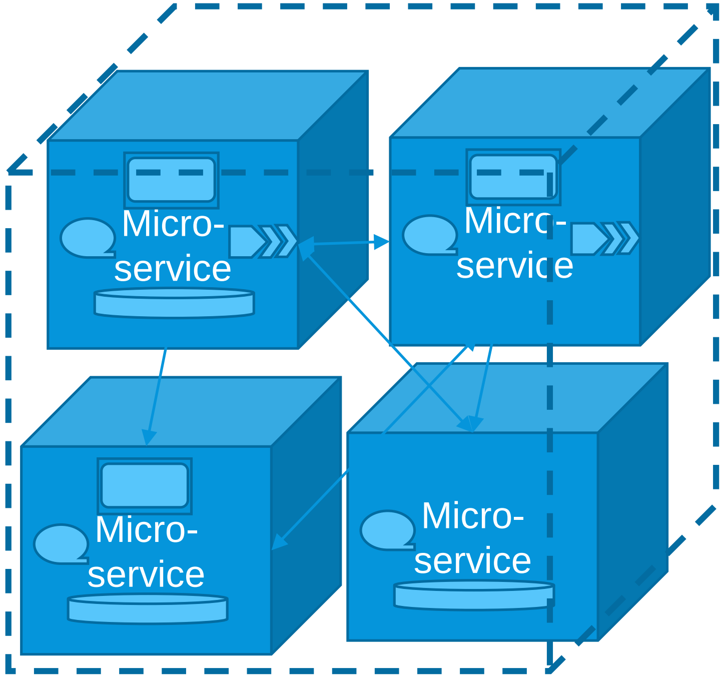
One large System with internal Modules



Monolith Application

Microservices System

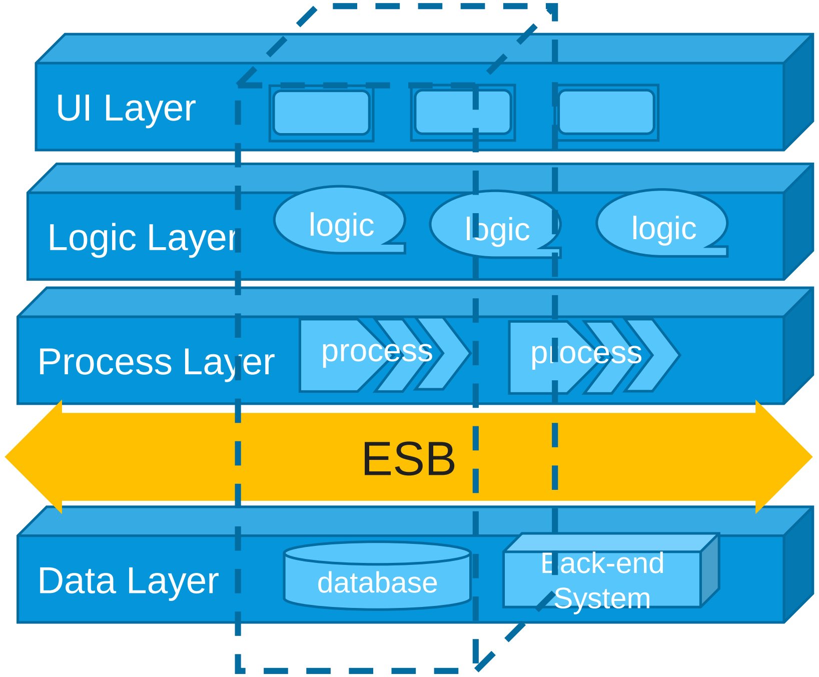
Independent Specific Components



Federated Application

SOA “System”

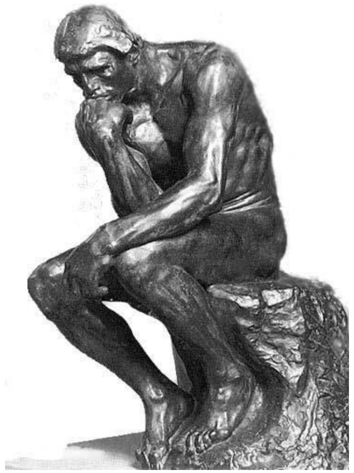
Shared Layered Components



Layered Application

There is More to Microservices than Size !

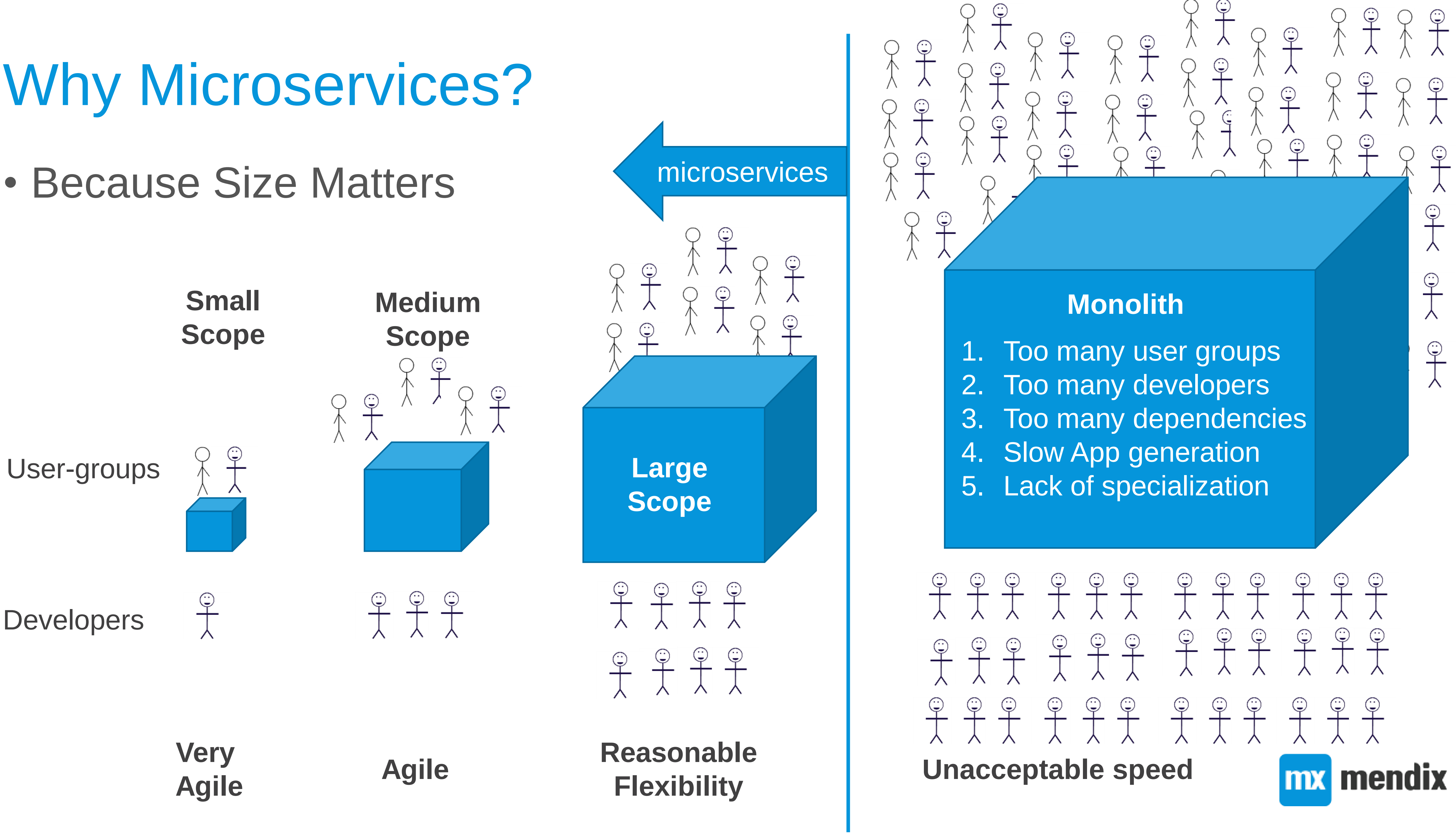
SOA and Monoliths	Microservices
“Share as much as Possible”	“Share as little as Possible”
“Do not Copy Data or Functions”	“Copy what I need to do my Job”
“Align teams with Technology”	“Cross functional Teams”
“Own a layer”	“Own a business function”



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

Why Microservices?

- Because Size Matters



Monolith

1. Too many user groups
2. Too many developers
3. Too many dependencies
4. Slow App generation
5. Lack of specialization

Real-life “Autonomous Units”

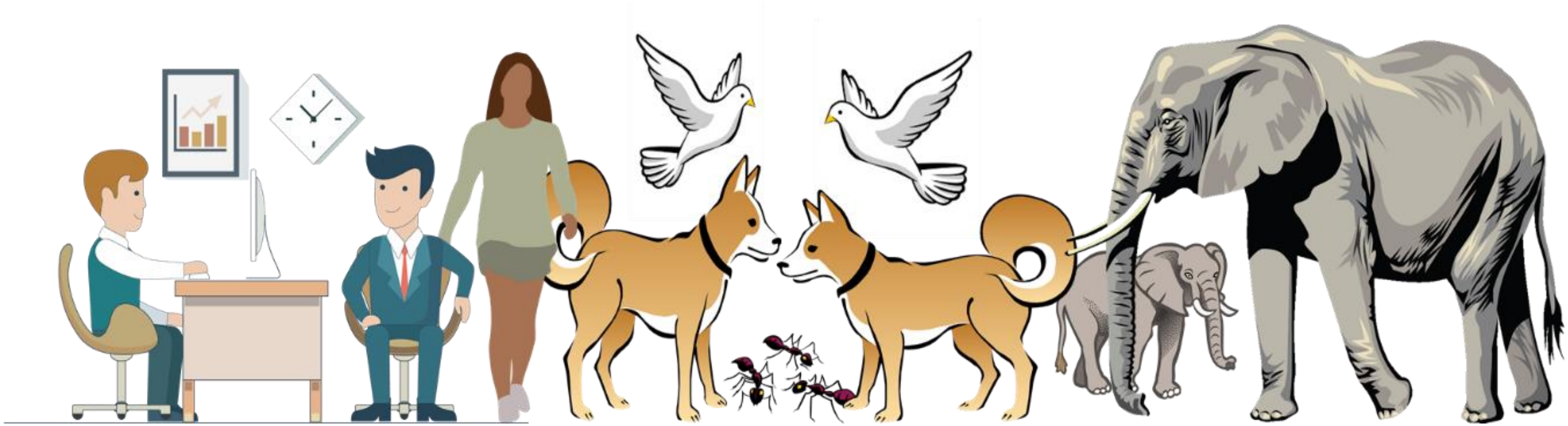
- Knowledge is better copied between individuals
- Merged with the experience and needs of each students
- Creates new valuable combinations of information and function

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



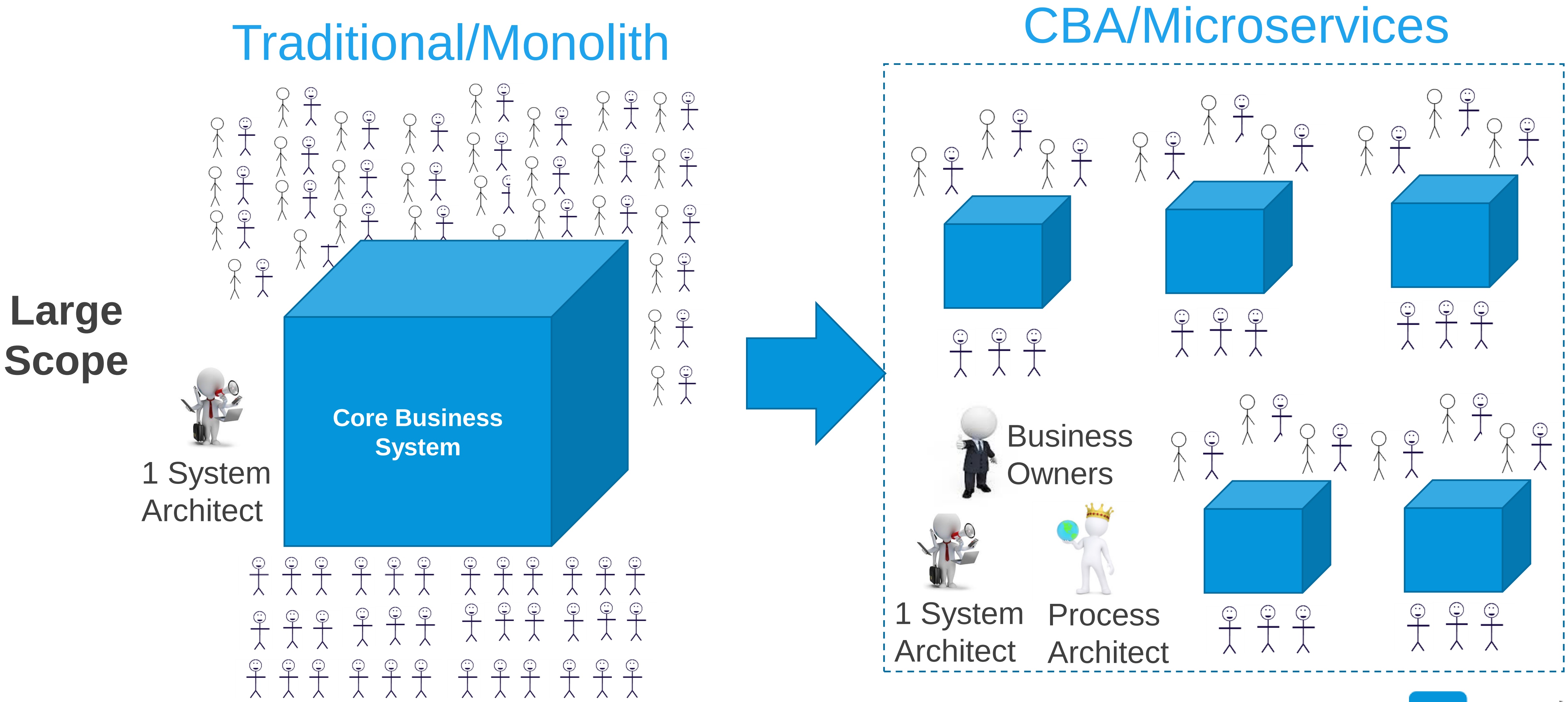
Building complete Business Functions as Microservices

Copy only the functions and data we really need to work properly

Some times for very stable functions to re-use as a service

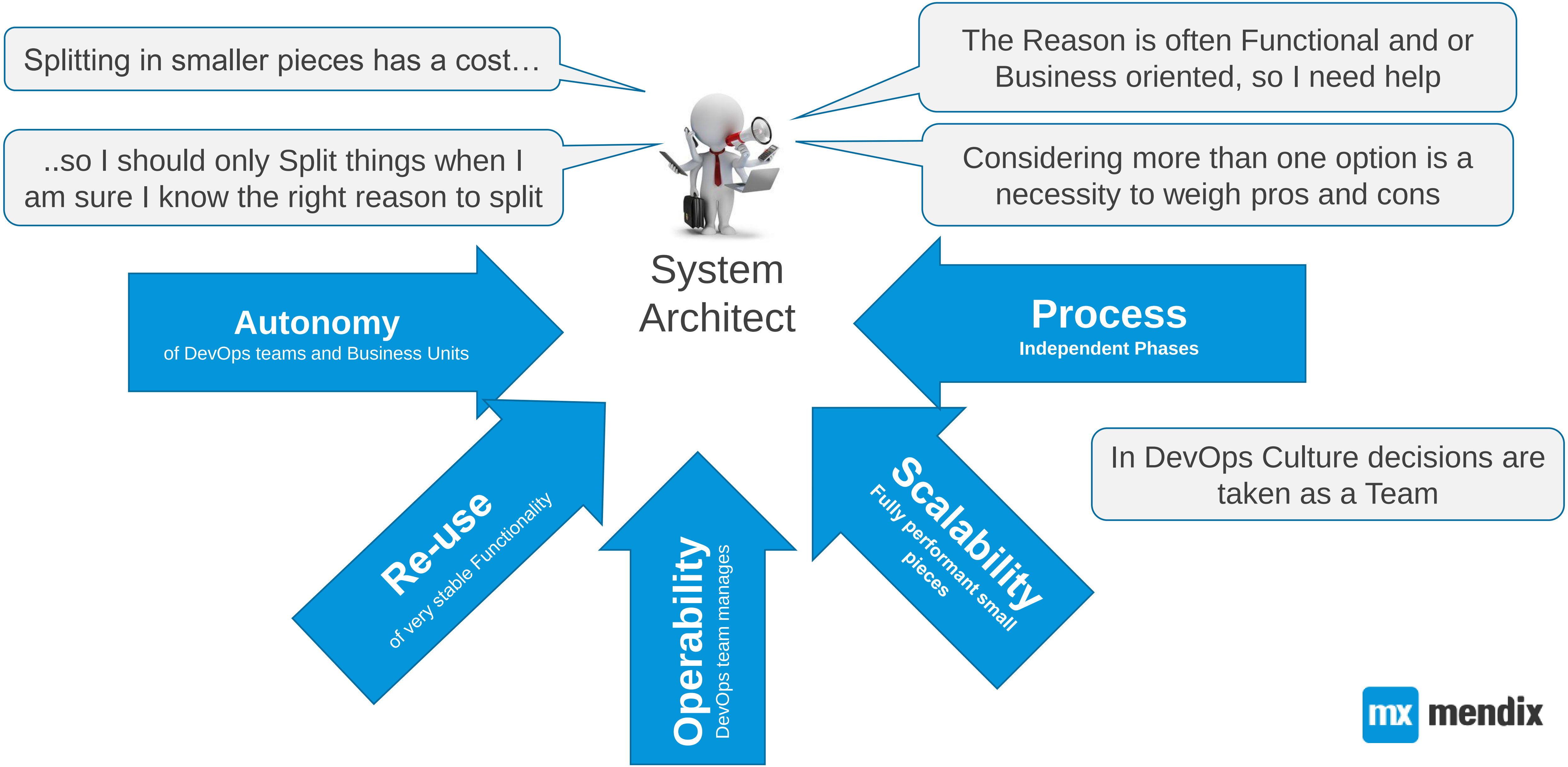
mx mendix

Autonomous Microservices for Core Systems



mendix

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

System Architect

Autonomy of DevOps teams and Business Units

Process Independent Phases

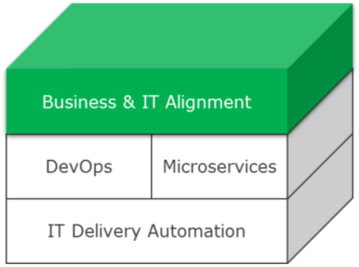
Re-use of very stable Functionality

Operability DevOps team manages

Scalability Fully performant small pieces

In DevOps Culture decisions are taken as a Team

Process & Business Flexibility in the Centre

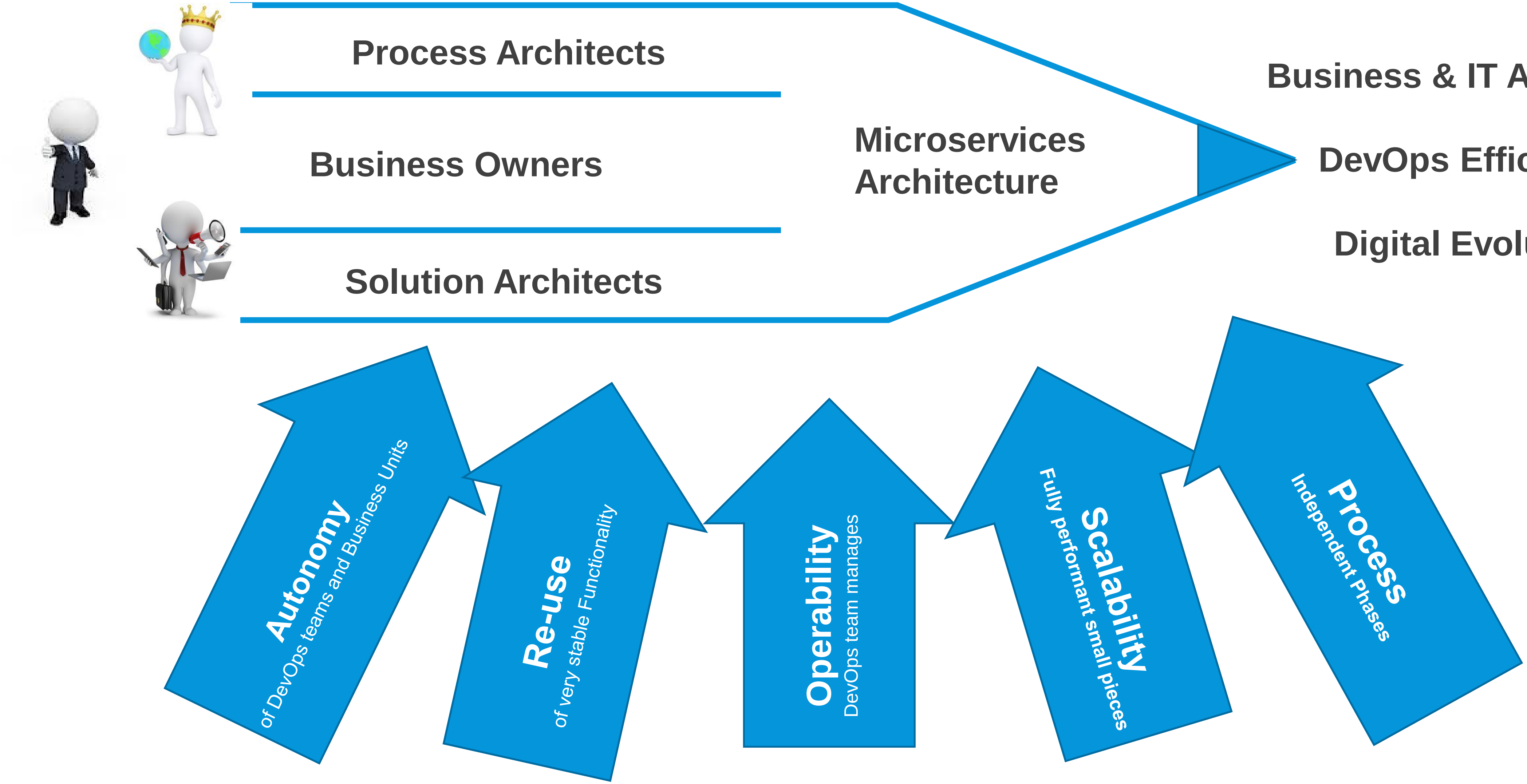


Business & IT Alignment

DevOps

Microservices

IT Delivery Automation



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 Functionality

Operability DevOps team manages

Scalability Fully performant small pieces

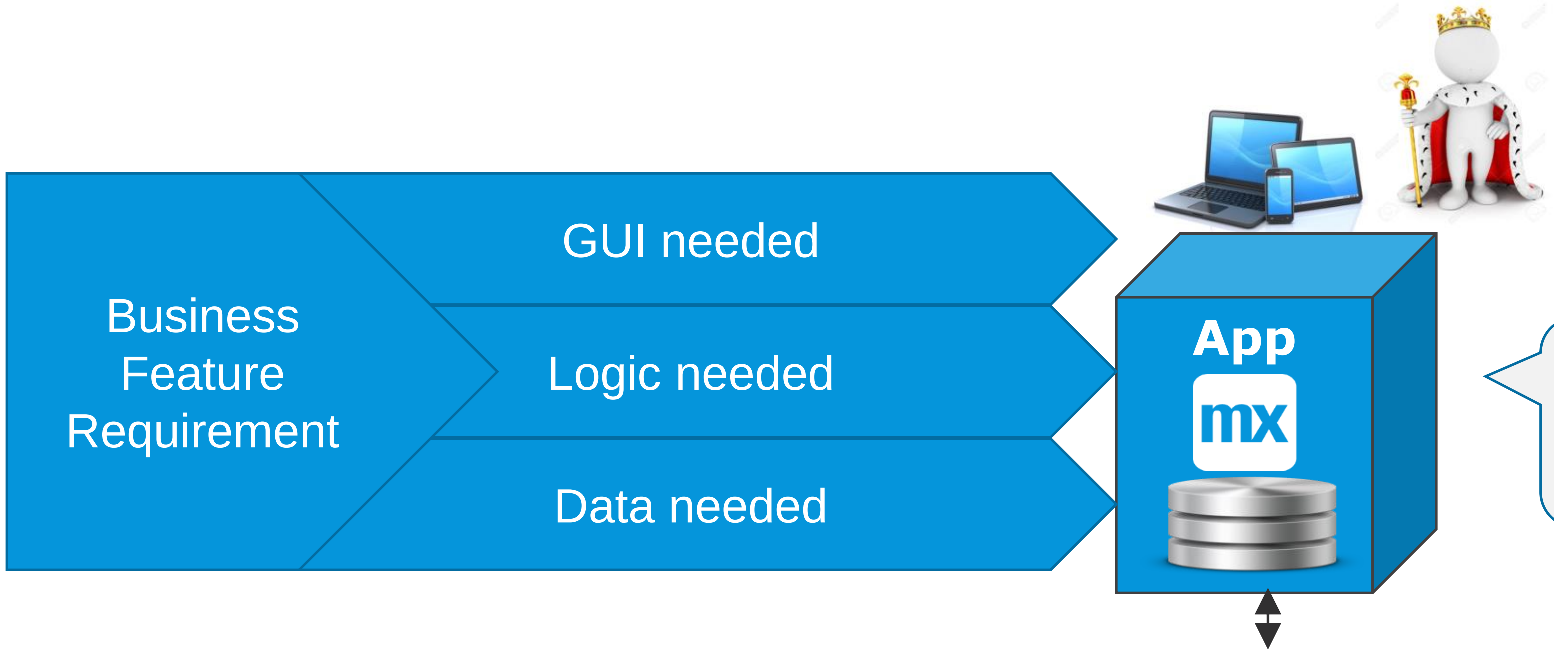
Process Independent Phases

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

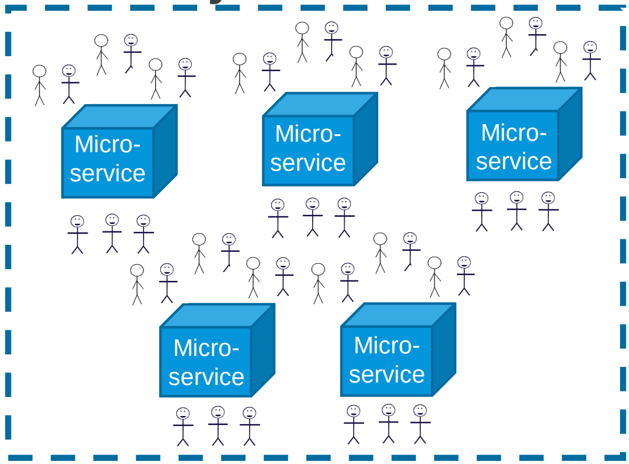
Some level of “Integration Test” and CICD when checking into team-server

Mendix Platform – Pre-built Automation

HPA
PaaS

*Buy
Automation*

Mendix Style Microservice System

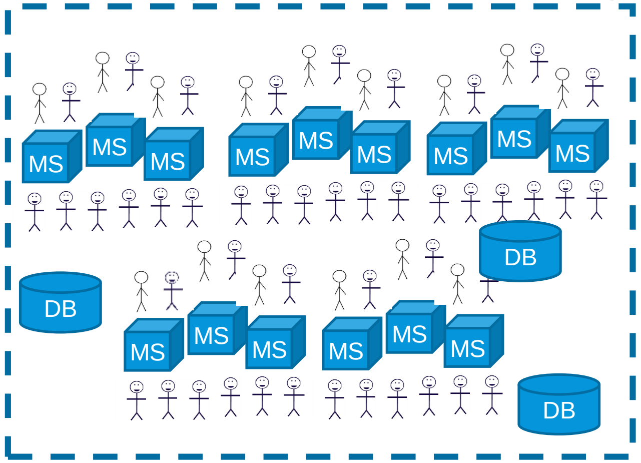


***5-10 Times
Faster
Results***

Open
Source

*Build
Automation*

Java Style Microservice System

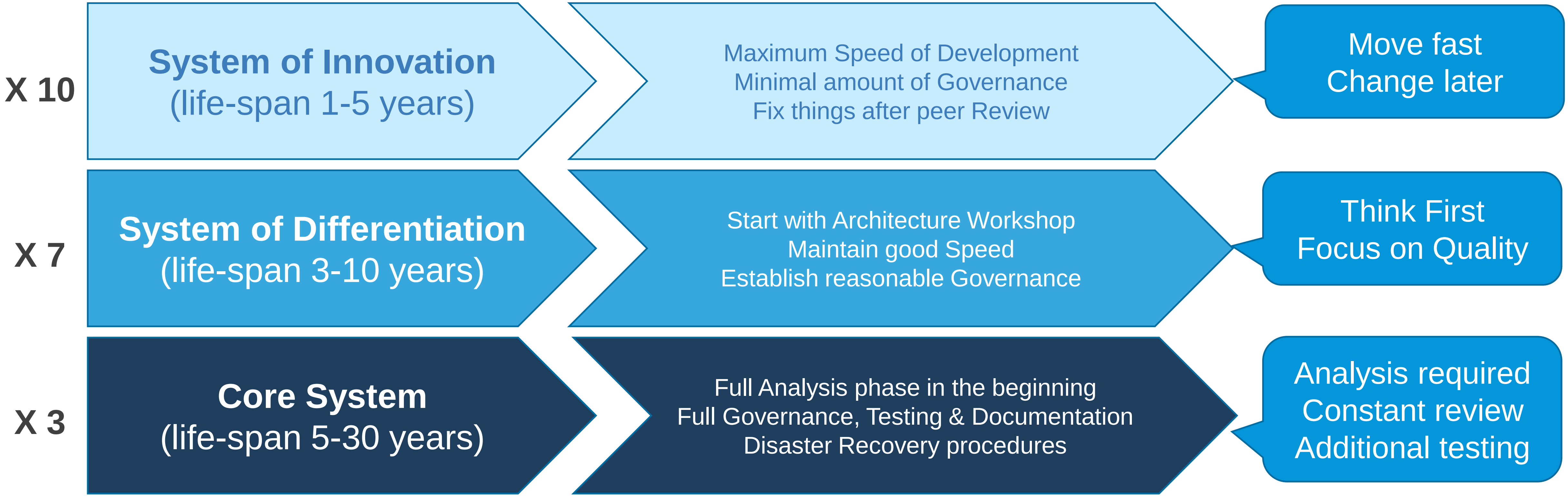


***Harder to
align IT with
Business
People***

mx mendix

Mendix for Core Systems

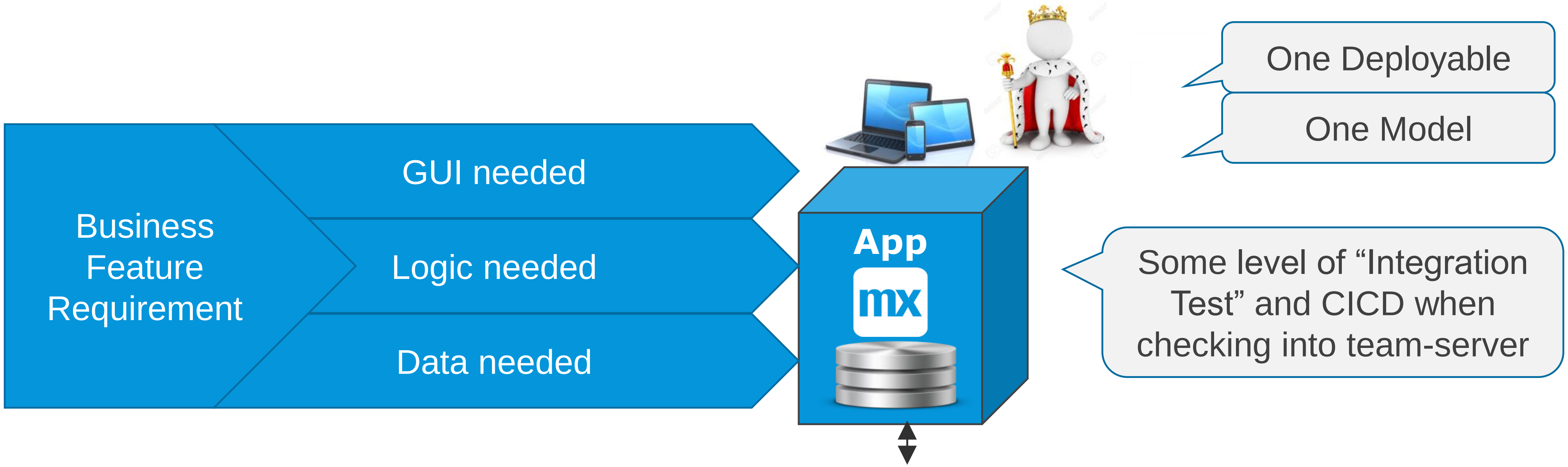
Adaptive Governance per System Type:



X 10	**System of Innovation** (life-span 1-5 years)	Maximum Speed of Development Minimal amount of Governance Fix things after peer Review	Move fast Change later
X 7	**System of Differentiation** (life-span 3-10 years)	Start with Architecture Workshop Maintain good Speed Establish reasonable Governance	Think First Focus on Quality
X 3	**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***



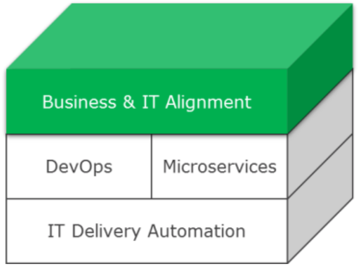
One Deployable

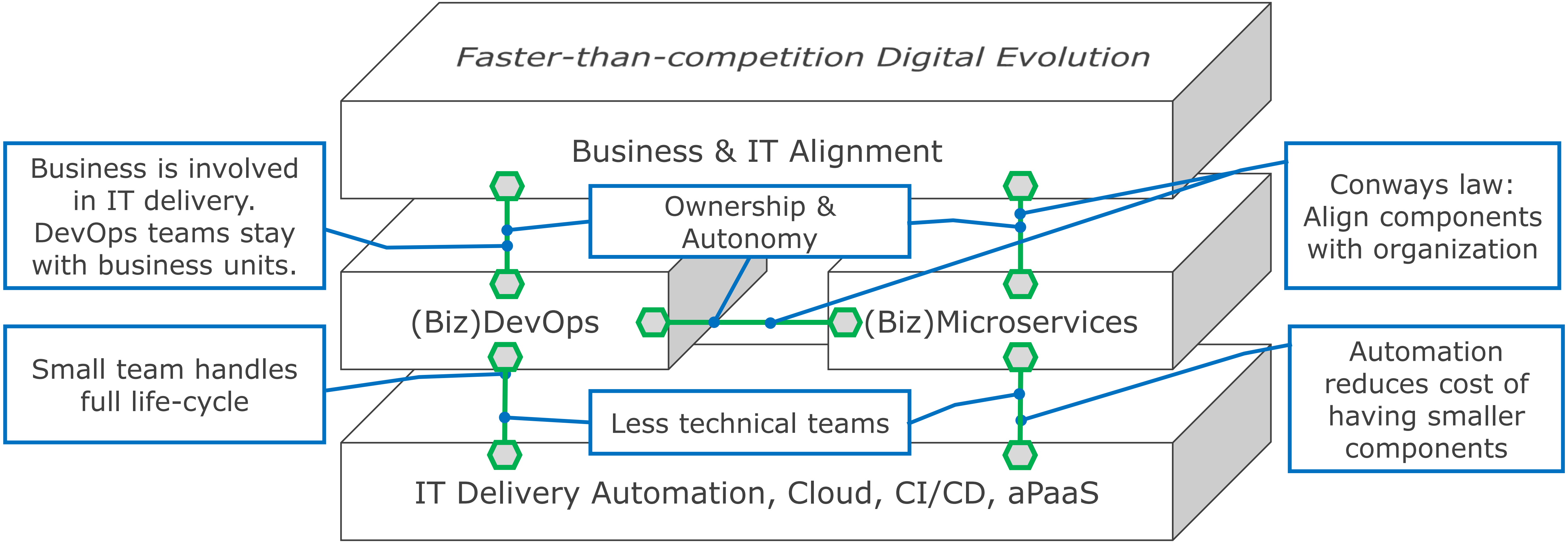
One Model

Some level of “Integration Test” and CICD when checking into team-server

mx mendix

The Business & IT Alignment Explained



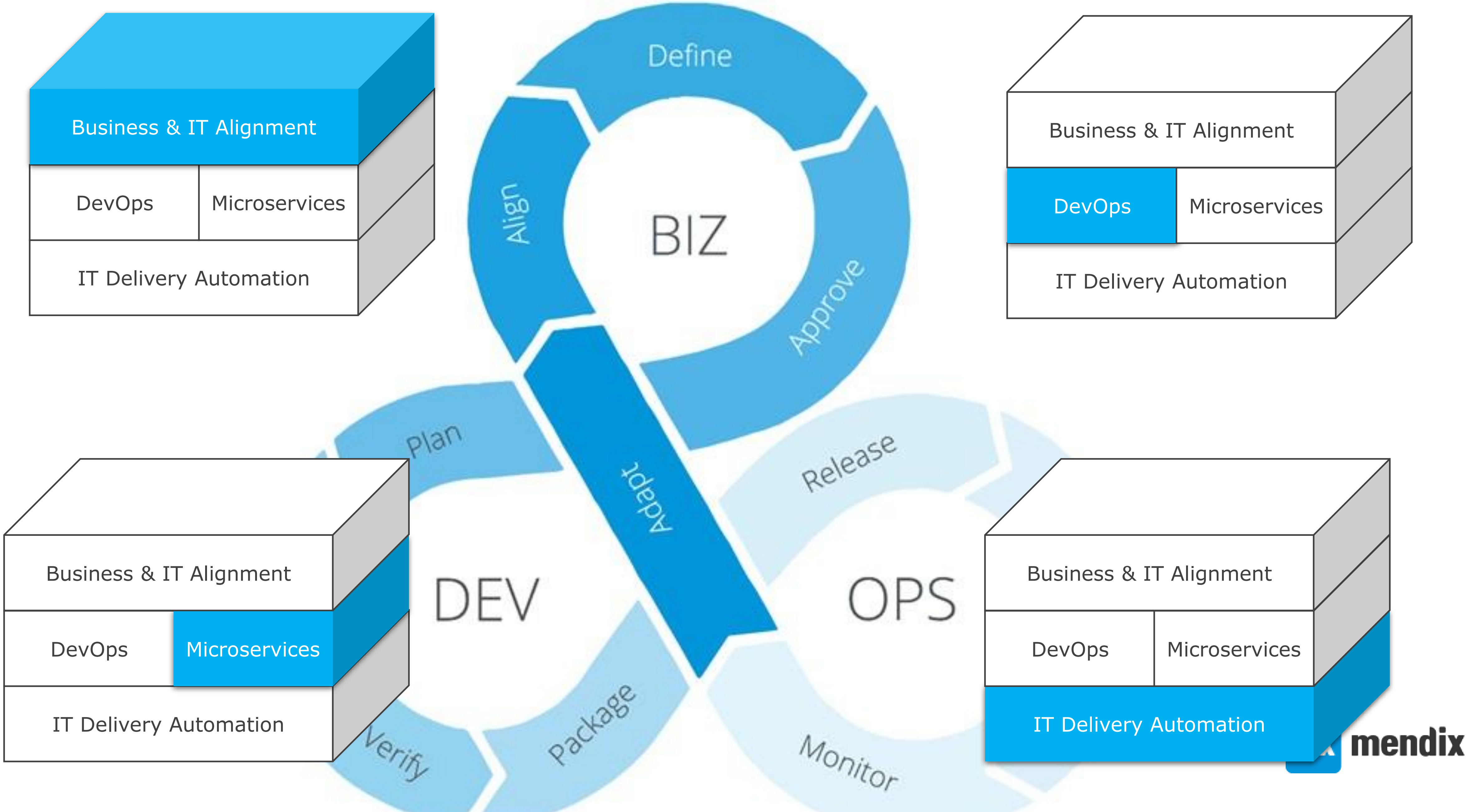


Business is involved in IT delivery. DevOps teams stay with business units.

Small team handles full life-cycle

Conways law: Align components with organization

Automation reduces cost of having smaller components



Business & IT Alignment

DevOps | Microservices

IT Delivery Automation

BIZ

Define

Align

Approve

Adapt

DEV

Plan

Verify

Package

OPS

Release

Monitor

Business & IT Alignment

DevOps | Microservices

IT Delivery Automation

Business & IT Alignment

DevOps | Microservices

IT Delivery Automation

Business & IT Alignment

DevOps | Microservices

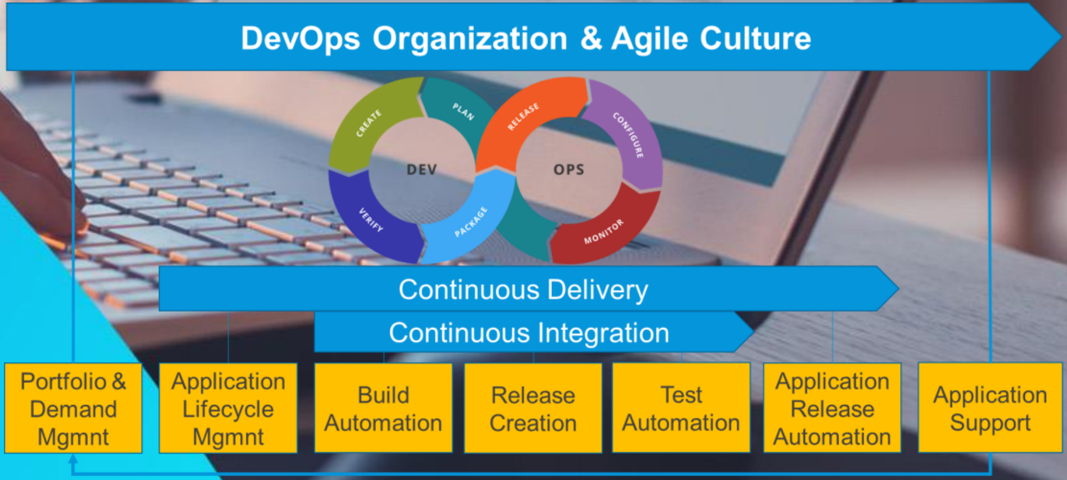
IT Delivery Automation

mendix

Questions?

Andreas

Architect



DevOps Organization & Agile Culture

Continuous Delivery

Continuous Integration

Portfolio & Demand Mgmnt

Application Lifecycle Mgmnt

Build Automation

Release Creation

Test Automation

Application Release Automation

Application Support

mx mendix