

Introduction to Microservices

Ronnie Mitra
Director of Design
@mitraman
ronnie.mitra@ca.com



What are
Microservices?

James Lewis & Martin Fowler on Microservices

- Componentization via Services
- Organized around Business Capabilities
- Products not Projects
- Smart endpoints and dumb pipes
- Decentralized Governance
- Decentralized Data Management
- Infrastructure Automation
- Design for failure
- Evolutionary Design



Fred George's Principles of Micro Services

- Very, very small
- Loosely coupled (including flow)
- Multiple versions acceptable (encouraged)
- Self-execution monitoring of each service
- Publish interesting stuff (w/o requirement)
- "application" seems to be poor conceptualization



Fred George's Principles of uServices

- Very, very **small**
- **Loosely coupled** (including **flow**)
- **Multiple versions** acceptable (encouraged)
- **Self-execution monitoring** of each service
- **Publish** interesting stuff (w/o requirement)
- "application" seems to be poor conceptualization



Adrian Cockcroft

On Microservices

“Loosely coupled service-oriented architecture with bounded context”

<http://www.hpts.ws/papers/2015/cockcroft-hpts.pdf>



Adrian Cockcroft

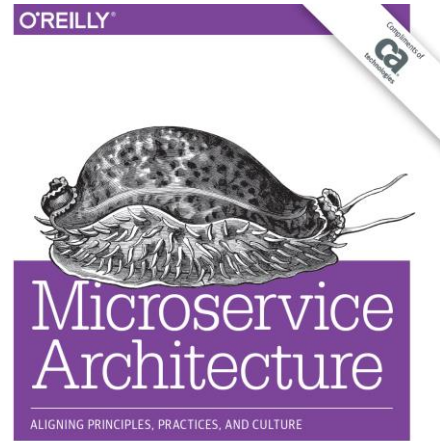
On Microservices

“Loosely coupled **service-oriented architecture** with bounded context”

<http://www.hpts.ws/papers/2015/cockcroft-hpts.pdf>

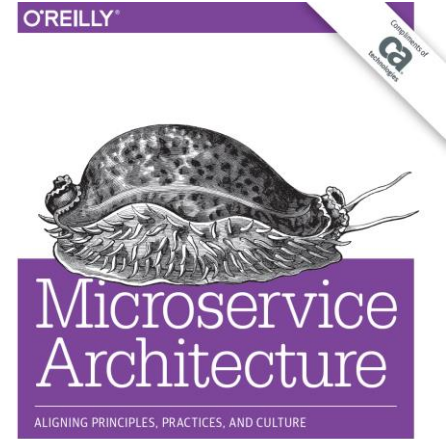


“A **microservice** is an independently deployable component of bounded scope that supports interoperability through message based communications.”



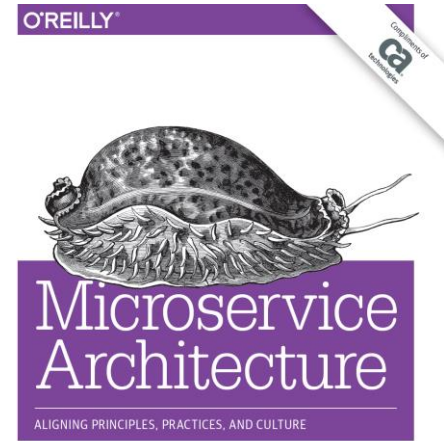
Irakli Nadareishvili, Ronnie Mitra,
Matt McLarty & Mike Amundsen

“**Microservice Architecture** is a style of engineering highly-automated, evolvable software systems made up of capability-aligned microservices.”



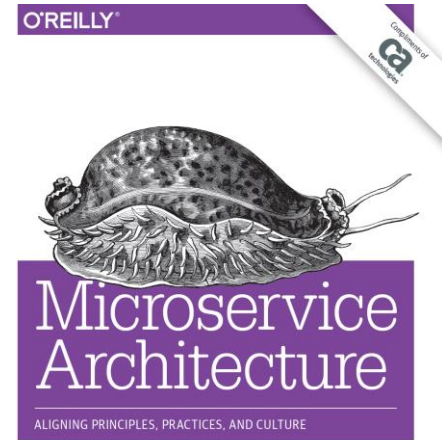
Irakli Nadareishvili, Ronnie Mitra,
Matt McLarty & Mike Amundsen

“A **microservice** is an **independently deployable** component of **bounded scope** that supports **interoperability** through **message based** communications.”



Irakli Nadareishvili, Ronnie Mitra,
Matt McLarty & Mike Amundsen

“**Microservice Architecture** is a style of engineering **highly-automated, evolvable** software systems made up of **capability-aligned** microservices.”



Irakli Nadareishvili, Ronnie Mitra,
Matt McLarty & Mike Amundsen

Small	Continuous improvement	Evolutionary	Automated	Loosely Coupled
Container-based	Conway's law	Independent Deployability	Decentralized Governance	Decentralized Data
Immutable	Message Based	Service-oriented	Products not Projects	Smart Endpoints Dumb Pipes
Bounded in Scope	Modular	Smart Endpoints	Continuous Deployment	Asynchronous Messaging
Interoperable	Capability Aligned	Design for Failure	Event Based	Autonomous

Small	Continuous improvement	Evolutionary	Automated	Loosely Coupled
Container-based	Conway's law	Independent Deployability	Decentralized Governance	Decentralized Data
Immutable	Message Based	Service-oriented	Products not Projects	Smart Endpoints Dumb Pipes
Bounded in Scope	Modular	Smart Endpoints	Continuous Deployment	Asynchronous Messaging
Interoperable	Capability Aligned	Design for Failure	Event Based	Autonomous

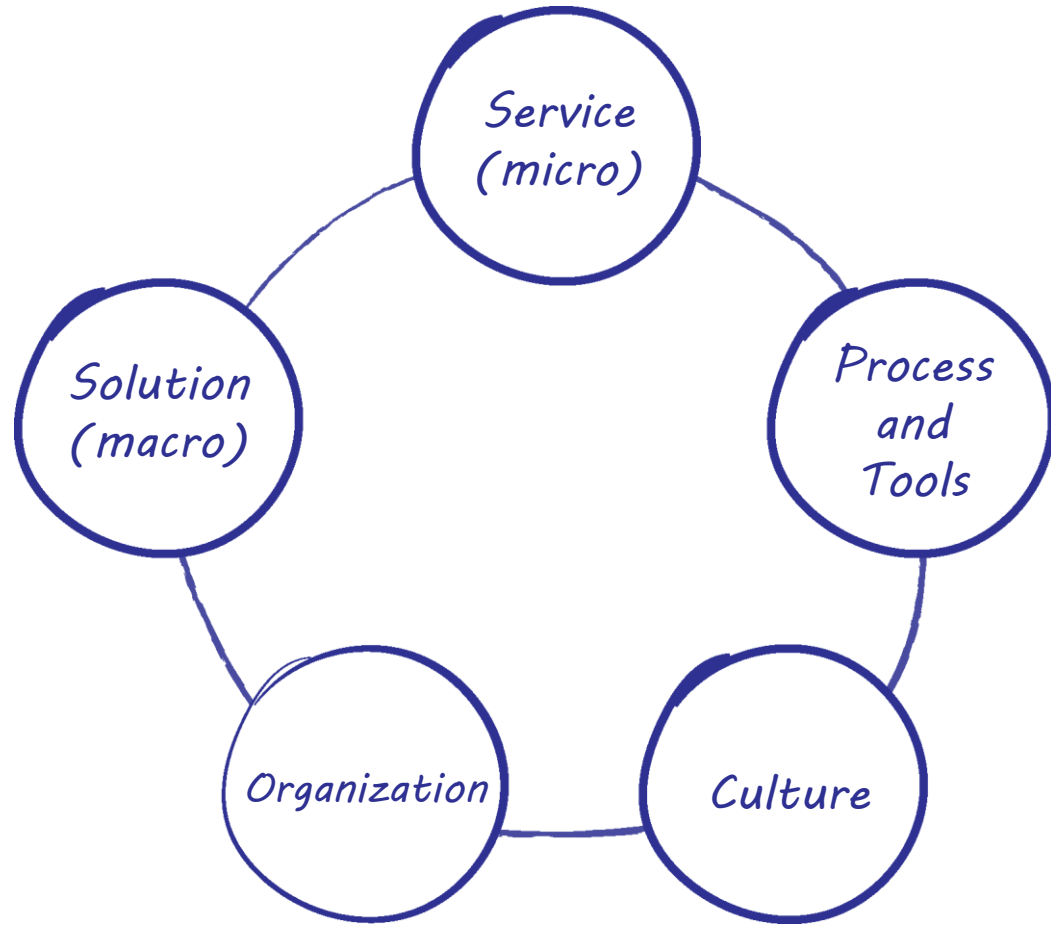
Microservice Complexity

- Microservices are simple
- Microservice systems are complex

Fred Brooks on Software Complexity

“The complexity of software is an essential property not an accidental one. Hence descriptions of a software entity that abstract away its complexity often abstract away its essence.”





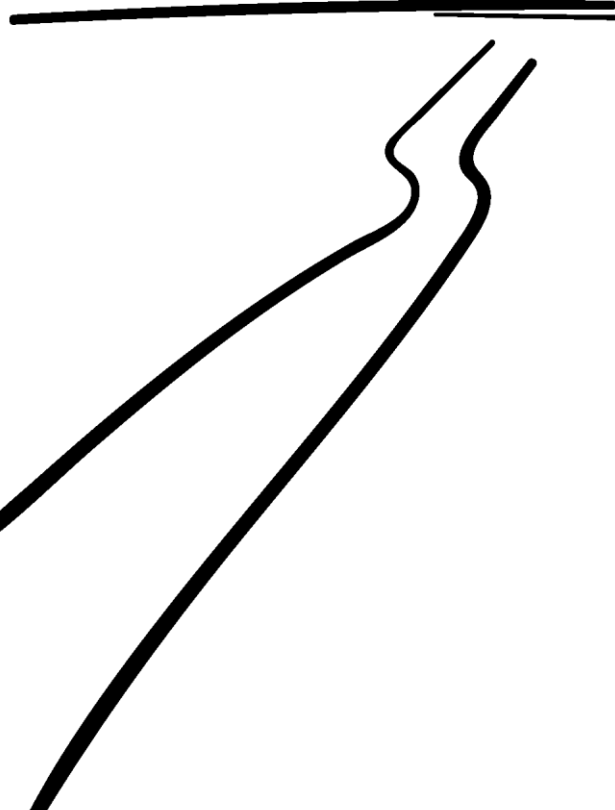
Small	Continuous improvement	Evolutionary	Automated	Loosely Coupled
Container-based	Conway's law	Independent Deployability	Decentralized Governance	Decentralized Data
Immutable	Message Based	Service-oriented	Products not Projects	Smart Endpoints Dumb Pipes
Bounded in Scope	Modular	Smart Endpoints	Continuous Deployment	Asynchronous Messaging
Interoperable	Capability Aligned	Design for Failure	Event Based	Autonomous

Small	Continuous improvement	Evolutionary	Automated	Loosely Coupled
Container-based	Conway's law	Independent Deployability	Decentralized Governance	Decentralized Data
Immutable	Message Based	Service-oriented	Products not Projects	Smart Endpoints Dumb Pipes
Bounded in Scope	Modular	Smart Endpoints	Continuous Deployment	Asynchronous Messaging
Interoperable	Capability Aligned	Design for Failure	Event Based	Autonomous

Why Microservices?

a useful way of managing
change in software
applications

The Microservices Way



Speed and Safety at Scale
and in Harmony

Speed and Safety at Scale and in
Harmony

We Want Change and We Want it Now

Business Models

Products

Experiences

What Would Amazon Do?

Business Models

Products

Experiences



Business Models

Products

Experiences

Applications

Data Models

Architecture

What Would Netflix Do?

Applications

Data Models

Architecture

NETFLIX

Jeff Bezos Wants Amazon to be Fast

“We want to combine the extraordinary customer-serving capabilities that are enabled by size with the **speed** of movement, nimbleness, and risk-acceptance mentality normally associated with entrepreneurial start-ups.”

Jeff Bezos

Amazon 2015 Letter to Shareholders



Examples of Increasing Speed

- Zero beauracracy, don't ask for permission
- Zero validation, don't test anything
- Increase change frequency
- Higher faster programmers
- Use un-constrained languages and tools
- Change production directly

Facebook's Motto

“Move Fast and Break Things”

Mark Zuckerberg



Speed and **Safety** at Scale and in
Harmony

Why Safe?

- Sometimes failures are catastrophic
- Startups can afford to be less-safe, endups are usually more cautious

Examples of Increasing Safety

- Strong governance, control everything
- Always validate, test everything
- Reduce change frequency
- Higher safer programmers
- Use constrained languages and tools
- No access to production environments

Speed and Safety at **Scale** and in
Harmony

The Scaling Factor

A changes in size can have a big impact on our system:

Scaled **demand**

Scaled **distance**

Scaled **organizations**

Speed and Safety at Scale and in
Harmony

Speed

Safety

Autonomy

Control

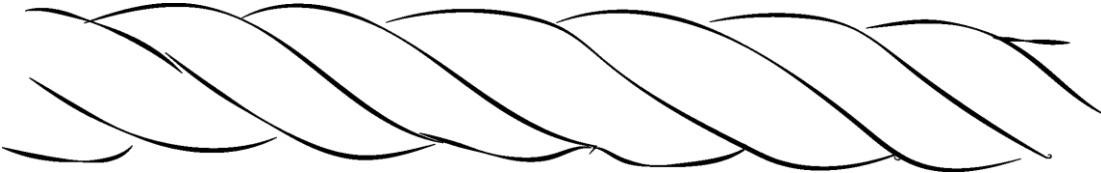
De-centralization

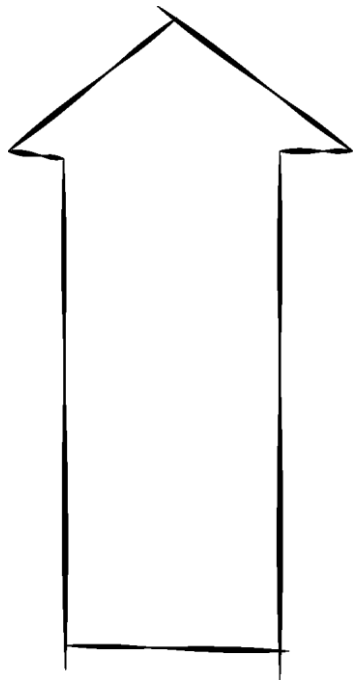
Centralization

Embrace Risk

Avoid Risk

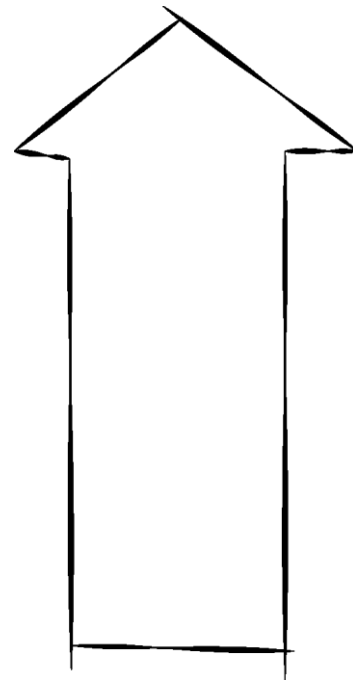
Trade-Offs

Speed  *Safety*



Speed

Harmony



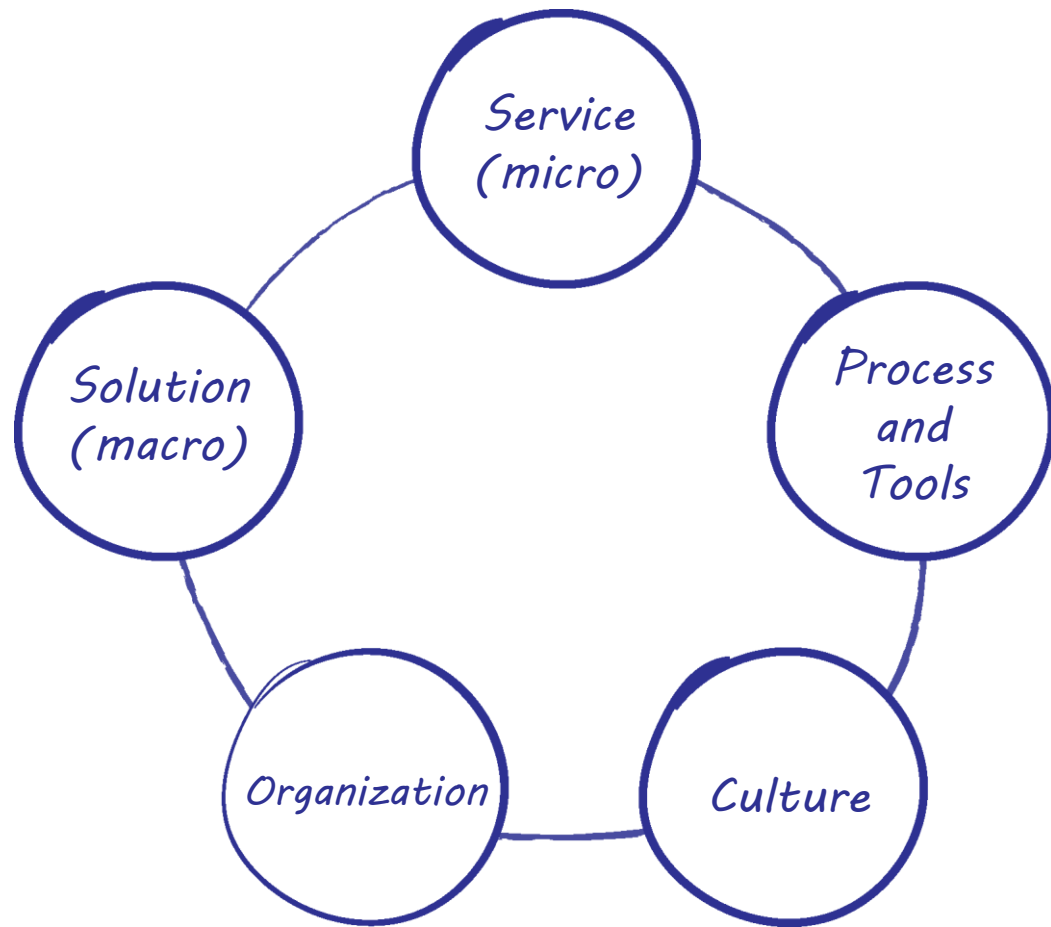
Safety

Harmony

“Cheating” the speed-safety trade-off:

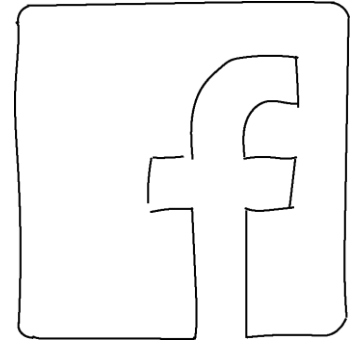
- Introduce automation
- Change process and tooling
- Alter the organizational design
- Improve the architecture
- Design the culture
- Move the change boundaries

Small	Continuous improvement	Evolutionary	Automated	Loosely Coupled
Container-based	Conway's law	Independent Deployability	Decentralized Governance	Decentralized Data
Immutable	Message Based	Service-oriented	Products not Projects	Smart Endpoints Dumb Pipes
Bounded in Scope	Modular	Smart Endpoints	Continuous Deployment	Asynchronous Messaging
Interoperable	Capability Aligned	Design for Failure	Event Based	Autonomous



Facebook's Motto (reprised)

“Move Fast and Break
Things”

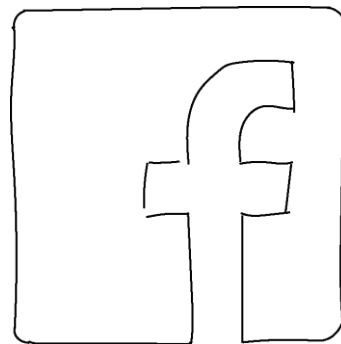


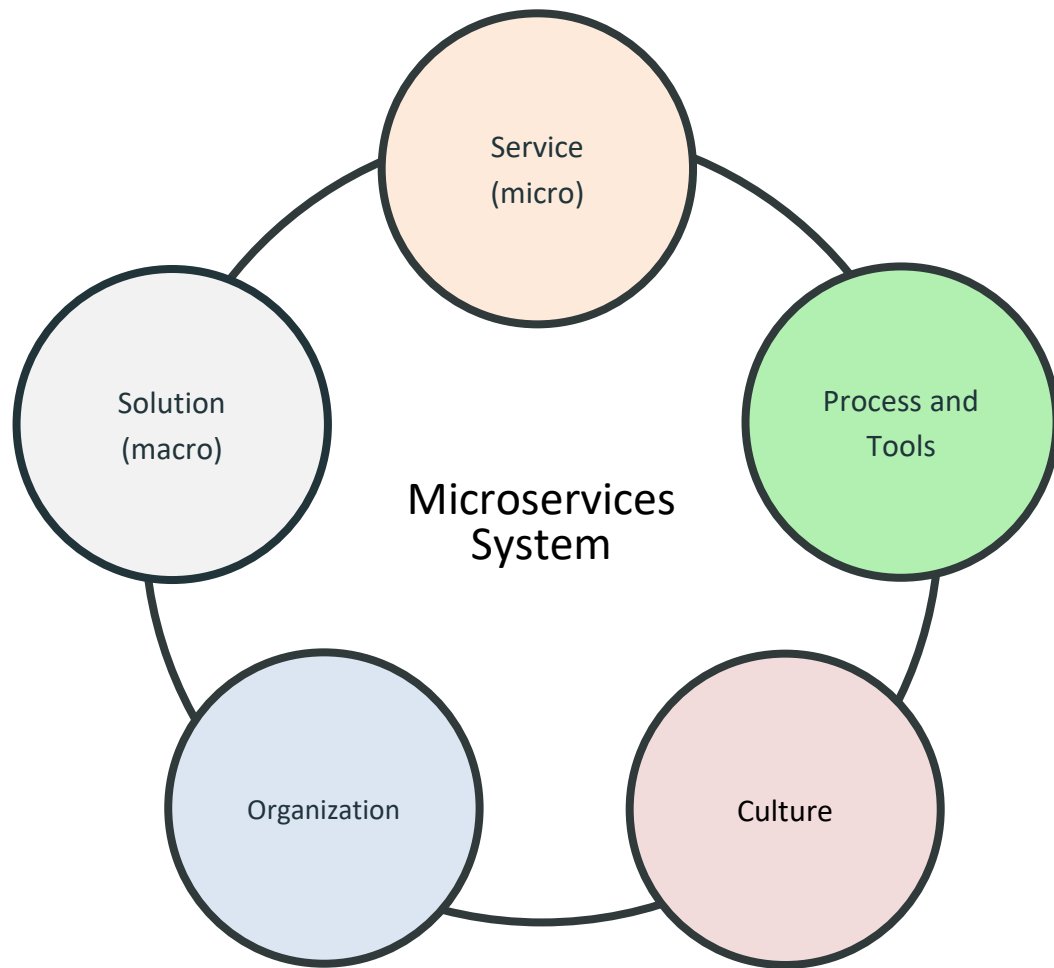
Facebook's Motto (reprised)

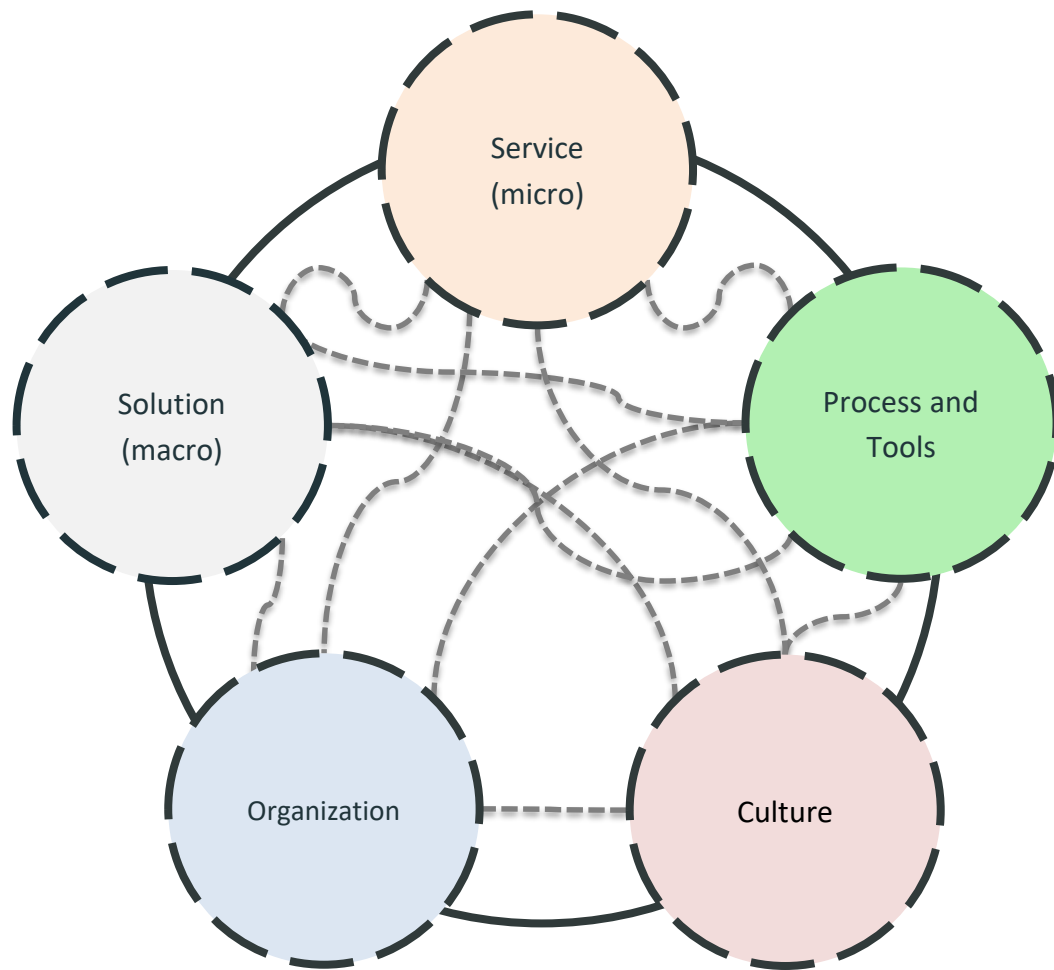
“Move Fast and Break
Things”

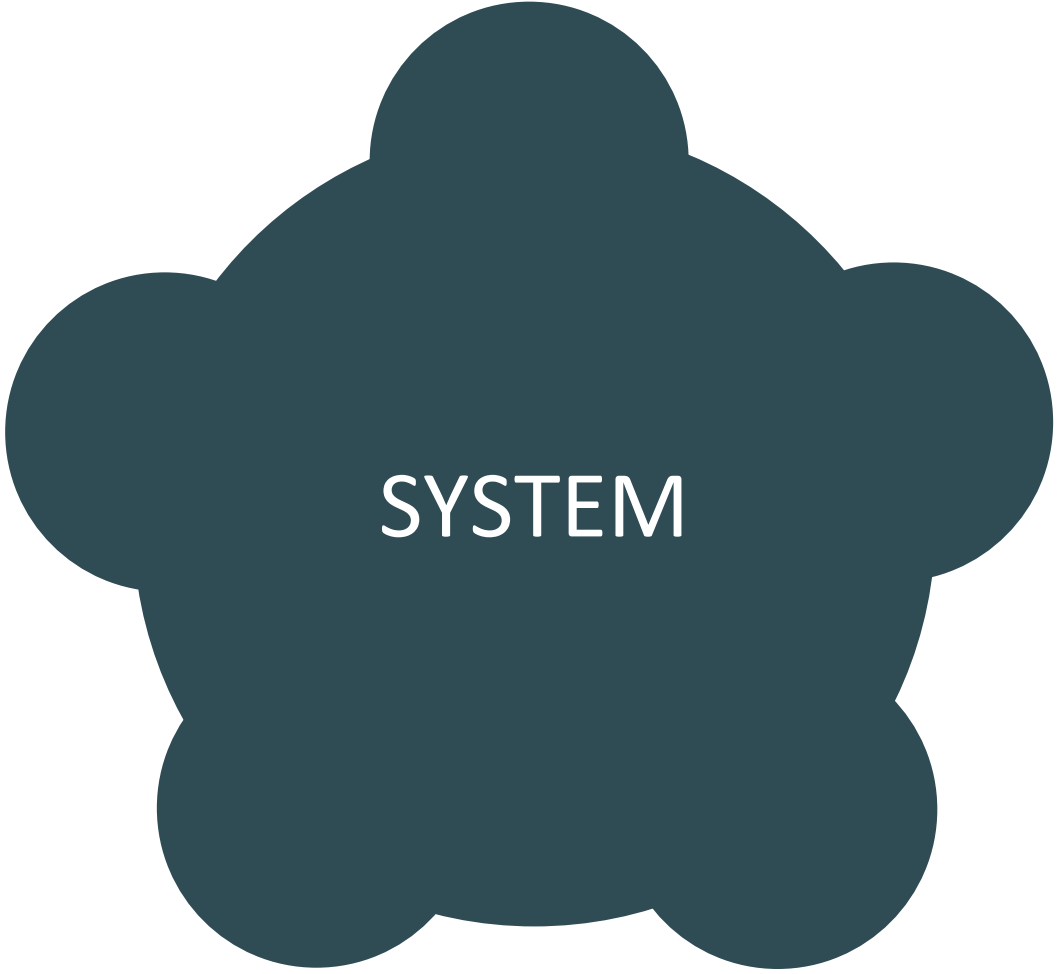
“Move Fast with
Stable Infra”

Facebook's motto in 2014







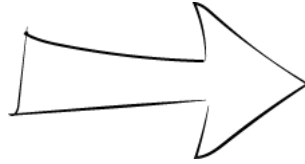


SYSTEM

How We Work Today

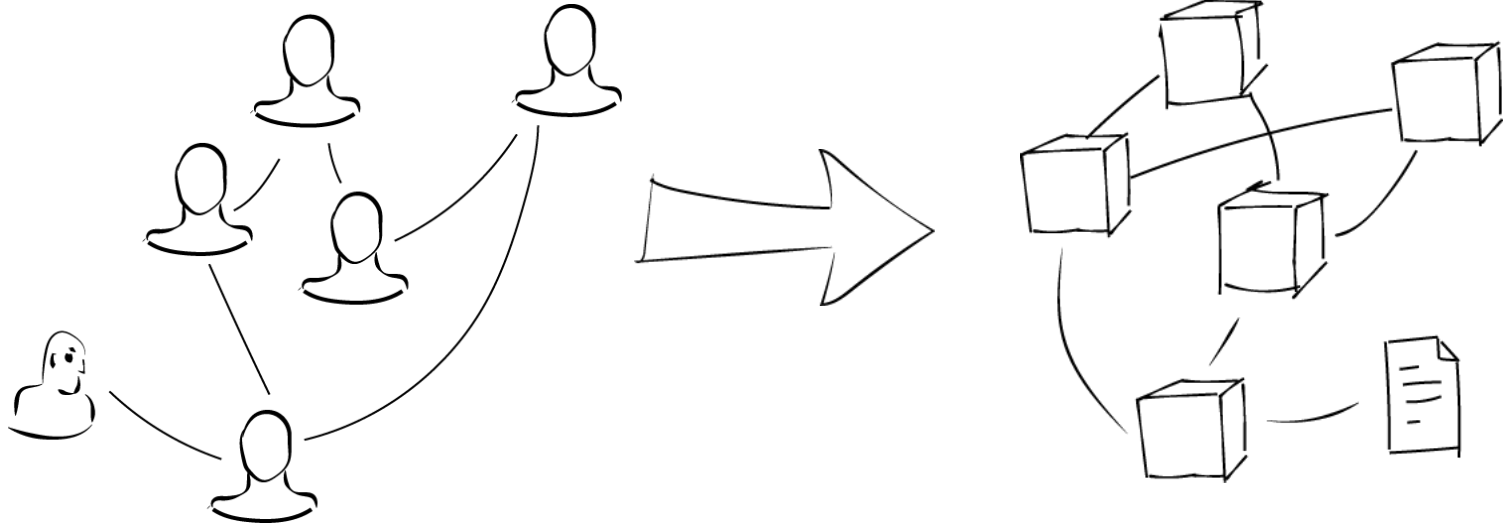


adaptive

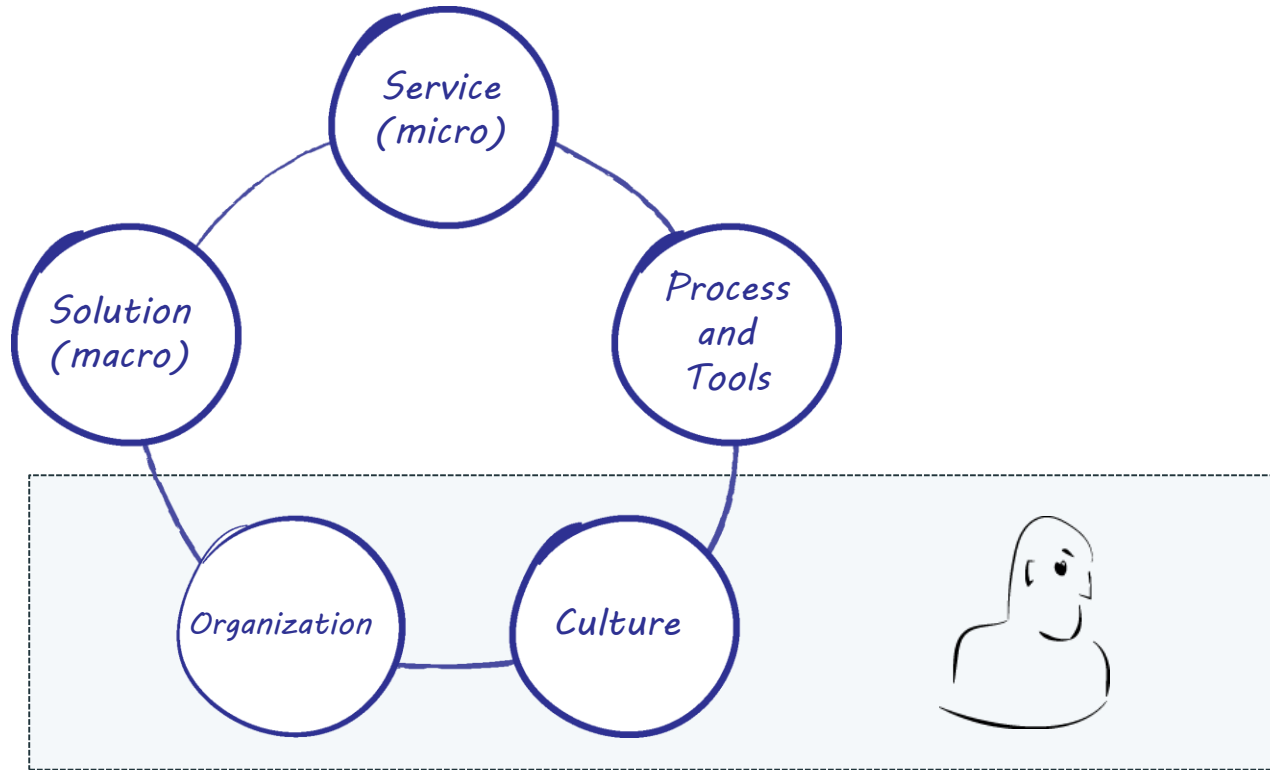


prescribed

The System That Outputs a System



The People Elements



Towards Harmony: Traditional Obstacles

- Co-ordination costs are too high
- Change is laborious
- Risk levels are too high (human error)

Emerging Trends

	Lighter Co-ordination	Labour Reduction	Risk Reduction
Solution (macro)	De-Centralization	Evolvability	Adaptiveness / “Anti-Fragility”

Emerging Trends

	Lighter Co-ordination	Labour Reduction	Risk Reduction
Solution (macro)	De-Centralization	Evolvability	Adaptiveness / “Anti-Fragility”
Service (micro)	Independent Deployability	Implementation Autonomy	Constrained Size

Emerging Trends

	Lighter Co-ordination	Labour Reduction	Risk Reduction
Solution (macro)	De-Centralization	Evolvability	Adaptiveness / “Anti-Fragility”
Service (micro)	Independent Deployability	Implementation Autonomy	Reduced Size
Process and Tools	Smaller Units of Change	Work Automation	Test Automation

Emerging Trends

	Lighter Co-ordination	Labour Reduction	Risk Reduction
Solution (macro)	De-Centralization	Evolvability	Adaptiveness / “Anti-Fragility”
Service (micro)	Independent Deployability	Implementation Autonomy	Reduced Size
Process and Tools	Smaller Units of Change	Work Automation	Test Automation
Culture	Trust	Tool Making	Accountability & Responsibility

Emerging Trends

	Lighter Co-ordination	Labour Reduction	Risk Reduction
Solution (macro)	De-Centralization	Evolvability	Adaptiveness / “Anti-Fragility”
Service (micro)	Independent Deployability	Implementation Autonomy	Reduced Size
Process and Tools	Smaller Units of Change	Work Automation	Test Automation
Culture	Trust	Tool Making	Accountability & Responsibility
Organization	Distributed Authority	Specialization	Better Talent

Speed and Safety at Scale and in
Harmony

Introduction to Microservices

Ronnie Mitra
Director of Design
@mitraman
ronnie.mitra@ca.com

