Industrial-Grade DevOps

Balancing Speed with Extreme Quality

Frank Buschmann w/ Joachim Fröhlich, Lars Gelbke, Fei Li, Marc Zeller, Peter Zimmerer
Siemens AG, Corporate Technology
DevOps is state-of-practice in enterprise IT software engineering

**Leaders in DevOps experience measurable benefits**

<table>
<thead>
<tr>
<th>Benefit</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Market attractive product quality¹</td>
<td>18% increase in revenue</td>
</tr>
<tr>
<td></td>
<td>19% improvement in product quality and performance</td>
</tr>
<tr>
<td>Short time-to-market¹</td>
<td>21% reduction of time to market for product innovations</td>
</tr>
<tr>
<td>Minimal cost of downtime²</td>
<td>60% decrease of deployment downtime costs per year</td>
</tr>
<tr>
<td>IT performance²</td>
<td>On demand product deployment</td>
</tr>
<tr>
<td></td>
<td>3x lower deployment failure rate, 24x faster recovery from failure</td>
</tr>
<tr>
<td></td>
<td>&gt;2000x shorter lead times for product changes</td>
</tr>
</tbody>
</table>

¹) CA Technologies: DevOps: Enterprise Organizations’ Newest Best Practice, July 2015
²) Puppet 2016 State of DevOps Report
The Digitalization of the industry requires an adoption of DevOps

It is the “how” to achieving business agility
The Digitalization of the industry requires an adoption of DevOps. It is the “how” to achieving business agility.

Digitalization @ Siemens

- Digital Services
- Vertical Software
- Digitally Enhanced Electrification and Automation

8% growth p.a.

Source: Siemens AG
Yet the adoption of DevOps in industry is challenging. Stringent industry-specific constraints and requirements apply.
Yet the adoption of DevOps in industry is challenging. Stringent industry-specific constraints and requirements apply.
Yet the adoption of DevOps in industry is challenging. Stringent industry-specific constraints and requirements apply.
Ops environment complexity complicates continuous deployment

*Operations environments are often polyglot, critical infrastructures*

- Multiple IT, OT and field networks
- Restricted network access
- Multi-platform execution environments
- Federated cloud to edge deployments
- Flexible deployment demands

- What delivery pipeline architectures enable continuous deployment into industry operations environments?
- What technologies support flexible deployment of features to multi-platform execution environments?
Delivery pipelines extend to the field
Centrally managed, federated on-premise deployment infrastructure

Ñ Dedicated (restricted) DevOps management networks per (restricted) operations network
Ñ Secure connection of DevOps management networks with the delivery pipeline via VPN
Ñ Dedicated on-premise deployment execution stacks per DevOps management network

Ñ Management of on-premise deployment execution stacks as deployable services
Ñ Provisioning of on-premise deployment execution stacks using containerization
Diverse deployment scenarios
No two operations networks are likely to be the same

Direct Connection (for open Ops networks)

- On-premise deployment service directly accesses the main repository via Internet
- On-premise deployment service runs on edge device and updates it via feature pull

Repository mirror with the latest feature versions
Asset management to orchestrate devices for update
Deployment service updates devices via feature push

Push Update

Repository mirror with the latest product features
Deployment service runs on edge device and updates it via feature pull
Delivery pipelines support multi-platform build, test, and release

Parallelization of build, integration and test in virtual, emulated environment

- Versioning and release management strategies for products with flexible deployment demands
- Parallel build stage with dedicated build infrastructure per product execution platform
- Features w/ deployment flexibility packed for all execution environments on which they can run
- Common integration and test stage based on virtualization and emulation
- Repository with images for all deployment platforms available to the deployment service
Leveraging modern orchestration and provision technologies simplify the creation and management of industrial DevOps environments.

Most state-of-the-art technology need some adaptation to fulfill industry-specific needs.

Ops environment complexity enforces tailored solutions.

Single enterprise standard does not work.

Orchestration requires proper tooling.

Secure networking required to bridge dev and ops.

“Environment as asset” is a key mindset change!

Suggests role and practice extensions in existing scaled agile frameworks, e.g., SAFe.
Regulated domains challenge agility and cycle time

*Regulatory compliance like functional safety is often mandatory*

- Agility and continuous delivery create high volatility in ensuring regulatory compliance
- Current practice is time and effort intense
- Cross-cuts multiple engineering disciplines
- DevOps environments are subject of certification

- What enhancements to agile practices help address regulatory compliance in development fast enough?
- What methods and technologies support continuous delivery in the face of regulations and certification?
Safety assurance lifecycle in a nutshell

Strict compliance to (domain-specific) norms is mandatory

Regulatory compliance requires dedicated activities in addition to “normal” development
Methods and technologies for safety assurance are emerging

*Research driven by funded projects on EU and national level*

- Component Fault Trees (CFT)
- Safety Concept Trees (SCT)
- Conditional Safety Certificates (ConSerts)
- Component-based Failure Mode and Effect Analysis (FMEAexpress)
- Digital Dependability Identities (DDI)

*) EU H2020 project DEIS (Dependability Engineering Innovation for Cyber Physical Systems), national BMBF project CrESt (Collaborative Embedded Systems)
Non-agile culture and limited automation are core challenges

*Negative impact on agility and cycle time*

- Agile practices are available but rarely practiced
- Though tool supported, activities are mostly manual
- Compilation of safety case is typically tedious and error-prone paperwork
- Certification by authorities not integrated with development
Automation of delivery process is a viable first step
*Improves traceability and reduces pre-certification efforts*

- Automatic generation of safety tests from component fault trees (CFTs)
- Integration of safety tests with test automation in delivery pipeline
- Adaptive test and simulation environments with hardware in the loop for test execution
- Automatic generation of safety cases using digital dependability identifiers (DDIs)
Component Fault Trees (CFTs)*

Extend classic fault trees with a component concept

- Focus of are failure modes of an encapsulated system component
- Failures visible at the inport / outport of a component are modeled using Input / Output Failure Modes

Divide-and-conquer strategy for systems

- Modular, hierarchical decomposition of system fault trees
- Potential reuse of CFTs in other systems

Automated Test Case Generation from CFTs

Derive test cases from the failure behavior specification (safety tests)

* Show that in case of failure the implemented behavior is compliant to the specified behavior
* Use of test results to verify if safety mechanisms are correctly implemented
* Provide inputs and expected outputs for fault injection tests

Digital Dependability Identity (DDI)*

Ensures interoperability of safety information across organizations

- Contains all information that uniquely describes the safety characteristics of a system component and its interaction
- Supports encapsulation of model-based descriptions of a system component’s safety artifacts
- Provides standardized, machine-readable interfaces to access the safety artifacts
- Component DDIs can be connected to describe safety characteristics of an entire system

Composition of safety cases from DDIs

Definition and execution of modular, hierarchical safety cases

- Continuous and defined extension along the system’s lifecycle
- Basis for automatic execution of formal safety tests
- Support for reliable and traceable certification by authorities

Integration at design time
Integration at configuration time
Integration at runtime
Component Fault Trees (CFT), Conditional Safety Certificates (ConSerts) and Digital Dependability Identities (DDI) are promising technologies to automate the delivery process.

Design and implementation of adaptive test and simulation environments is a challenge.

Automated delivery process is important, but not sufficient to practice DevOps in regulated domains.

Additional methods and technologies required, e.g., for rapid design, change / impact analysis.

Agile practices available, but practices do not automatically create culture!

Certification of tools is a challenge by itself, especially if DevOps environments evolve.
Operational quality demands side-effect-free deployment
Correct system behavior, availability and performance must not degrade

- Many systems run 24/7, some with 99.999% availability – especially in Operations and Control
- System correctness and (real-time) performance must be guaranteed when needed
- Actual configuration may differ from test configuration, e.g., hardware change, exact position of machine

- What product designs enable continuous deployment in run and test-in-production without side effects?
- What runtime architectures ensure a side-effect-free deployment of features in run?
Product features must be prepared for continuous deployment

*Design for independence*

- Realize product features as independent, self-contained Microservices
- Define and block dedicated time and resource budgets for testing in production

![Diagram of Microservice architecture]( attachment:diagram.png )

Microservice

Driver

Sensor Proxy

Control Logic

Actuator Proxy

Driver

Input

Output
Product features must be prepared for continuous deployment

*Design for testability*

Integrate a permanent TEST PROBE* to dynamically configure and execute tests in production

Product features must be prepared for continuous deployment

*Use AUTHORITY DETERMINATION* paired with PRIMARY ACTIVATION* for seamless version switch and fall back in case of deployment failure

*) Publication in preparation
Product features must be prepared for continuous deployment

*Design for availability*

Integrate CONNECTORS* to separate domain logic and infrastructure domains from one another

Product features must be prepared for continuous deployment

*Design for independence, testability and availability*

- Realize product features as independent, self-contained Microservices
- Define and block dedicated time and resource budgets for testing
- Integrate a permanent TEST PROBE to dynamically configure and execute tests in production
- Use AUTHORITY DETERMINATION paired with PRIMARY ACTIVATION for seamless version switch and fall back in case of deployment failure
- Integrate CONNECTORS to separate domain logic and infrastructure domains from one another

Ensure all timing and resource requirements are met!
Product features must be prepared for continuous deployment

*Design for independence, testability and availability*

---

How can we enable field tests of an industrial microservice to access internal process data, possibly distributed across several, internal components, without impairing the real-time behavior of the microservice under test?

Build a **TEST PROBE** permanently into each microservice. Encapsulate testable process data. Make the **TEST PROBE** programmable, so that test programs can exactly control read and write accesses to testable process data. Let microservices treat **TEST PROBES** as ordinary components with reserved and limited system resources, so that test programs cannot accidentally change the microservice behavior.

---


---

---

---

---
Deploy new microservice version, connect it with the availability infrastructure of the current version, let it receive input but “mute” is output

Execute pre-activation tests on the new microservice version via its TEST PROBE

Ensure the device has dedicated resource budgets for continuous deployment
Notify the AUTHORITY DETERMINATION infrastructure to determine the new primary

Let the PRIMARY ACTIVATION infrastructure activate the new microservice version when seamless switch-over is possible.
Deployment architecture must support zero downtime

*Design for fail-over*

- Continuously test the new microservice version in production via its TEST PROBE
- Fall back to old microservice version in case of problems, otherwise uninstall it
Deploy new microservice version to device, let it receive input but “mute” is output

Execute pre-activation tests on the new microservice version via its TEST PROBE

Notify the AUTHORITY DETERMINATION infrastructure to determine the new primary

Let the PRIMARY ACTIVATION infrastructure activate the new microservice version when seamless switch-over is possible

Continuously test the new microservice version in production via its TEST PROBE

Fall back to old microservice version in case of problems, otherwise uninstall it

Ensure the device has dedicated resource budgets for continuous deployment
Operational quality demands side-effect-free deployment

*Proof-of-Concept up and running*

**Demonstrator set-up**

- **DevOps pipeline**
  - Control dashboard
  - Gatekeeper
- **Demonstrator**
- **Lana**
- **T-Rux**

**Expected behavior**

- **μService 2**
- **μService 2a**
- **Feature**

**Lab set-up**

- **Activated**
- **Test**
- **Activate**
- **Activated**
Operational quality demands side-effect-free deployment

Key aspects addressed, several research questions remain

- Design for continuous deployment adopts and enhances patterns for fault-tolerant systems
- Design concepts are feasible for independent microservices that implement a self-contained control program
  - PRIMARY ACTIVATION is a sensitivity point regarding correctness and availability
- Additional design concepts necessary for
  - Control programs that are federated across multiple (distributed) microservices
  - Assurance that deployment is side-effect-free to other microservices
- Applicability of common test-in-production practices like chaos monkey uncertain
First adoptions of DevOps in industry target at the industrial IoT

Cloud to edge analytics applications are in focus

Siemens Industrial Edge offers users the possibility of executing a range of descriptive, diagnostic, predictive and prescriptive analytical applications. This allows cloud connectivity (data to cloud) to be used in combination with Edge Apps from Siemens, third party providers or end users themselves in an integrated hardware and software ecosystem (Edge App to Device) for automation components.
First adoptions of DevOps in industry target at the industrial IoT

Key aspects are development efficiency and delivery automation

- Adaptive Development Workspace
  - Centrally managed, locally adaptable by partners
  - Set-up and configuration < 30 min
  - Project templates to get developers productive < 1 day
  - Connectivity to delivery pipeline and edge devices

- Delivery Pipeline and App Management
  - Multi-staged pipeline with defined quality gates
  - Orchestration of apps federated across cloud and edge
  - Provisioning of runtimes for all target ops environments
  - App management infrastructure
DevOps proof of concepts for industrial operations systems in work

*MOM and SCADA are the target*

Manufacturing Operations Management

Supervisory Control and Data Acquisition

Source: Siemens AG
DevOps proof of concepts for industrial operations systems in work

Deployment flexibility and operational quality in focus

- Delivery pipeline for flexible deployment
  - Automatic orchestration of federated, distributed applications: cloud, on-premise, cloud to edge
  - Automatic provisioning of runtime images for all target ops environments
  - Firmware updates
  - App and firmware management

- Operational quality during deployment
  - Reliability
  - IT Security
  - Availability (on-premise and edge)
Research on industrial-grade DevOps ongoing

Tough industry challenges in focus

Operational Quality
IT Security / Regulated Domains
Systems Engineering

Continuous Testing of Software-intense Systems
Agile Culture and Automation
PLM and ALM Integration and Automation

Source: Siemens AG
Continuous testing of software-intense industrial systems

Adaptive test and simulation environment to balance speed and quality

Enabler for shift-left and shift-right in software intense-systems, e.g., industry automation

- Model-based Digital Twin of the software to get immediate and early feedback on its quality in industrial environments
- Virtual integration and deployment as digital sandbox
- X-in-the-loop to step-wise frontload physical reality into software development

![Diagram of Adaptive Test- and Simulation Environment]

Evaluation | Emulation | Prove

*) concept plus prototype
Research in industrial-grade DevOps is well founded

Core technologies developed over a longer time period


Industrial-grade DevOps is key for the Digitalization of Industry

Helps mastering the digital transformation at speed and scale

Defines the “How” of Digitalization

Industry-specific challenges to master

Successful adoption for Industrial IoT

Research ongoing for Industrial Operations and Control

Source: Siemens AG
Industrial-Grade DevOps
Balancing Speed with Extreme Quality

Frank Buschmann w/ Joachim Fröhlich, Lars Gelbke, Fei Li, Marc Zeller, Peter Zimmerer
Siemens AG, Corporate Technology