Management summary

“It is not the strongest of the species that survive, nor the most intelligent, but the one most responsive to change.”¹

Setting the scene

In the current world of IT and the development of IT-related products or services, companies from enterprise level to smaller sizes are starting to use the DevOps processes and methods as a part of their day-to-day organization process.

The goal is to reduce the time involved in all the software development phases, to achieve greater application stability and faster development cycles.

However not only on the technical side of the organization is DevOps changing the playing field, also an organizational change that involves merging development and operations teams is required with a hint of cultural changes.

And last, but not least, the skillset of all people involved is changing.

Goal of this Point of View

This point of view aims to create awareness around the transformation towards the DevOps way-of-working, to help gain understanding what DevOps is, why you need it and what is needed to implement DevOps.

An Enterprise Architecture perspective

Even though it is DevOps from an Enterprise Architecture perspective, this material has been gathered from our experiences with customers, combined with knowledge from subject matter experts and theory from within and outside Deloitte.

Targeted audience

It is specifically for the people within Deloitte that want to use this as an accelerator for conversations & proposals and to get in contact with the people who have performed these type of projects.

By all means, it is a deck that can be shared within Deloitte and with our customers to provide a more holistic view.

¹ Charles Darwin
DevOps practitioners

For questions or remarks, feel free to reach out to our DevOps practitioners

Mark Maijs
DevOps Overall Lead
Senior Manager
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 83 89 02 14
mmaijs@deloitte.nl

Donald Pondman
DevOps Transformation
Manager
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 83 33 02 56
dpondman@deloitte.nl

Jan Prüst
DevOps Technology
Senior Consultant
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 202 58 502
jprust@deloitte.nl

Sebastiaan Teerenstra
DevOps Process & People
Senior Consultant
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 22 98 09 09
steerenstra@deloitte.nl

Marlies Quekel
DevOps Organization & Team Setup
Consultant
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 50 07 02 16
mquekel@deloitte.nl

© 2022 Deloitte The Netherlands

Deloitte DevOps Point of View 3
## Contents

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is DevOps?</td>
<td>5</td>
</tr>
<tr>
<td>Why do I need DevOps?</td>
<td>9</td>
</tr>
<tr>
<td>Where is DevOps applicable?</td>
<td>12</td>
</tr>
<tr>
<td>What is needed for DevOps to work?</td>
<td>14</td>
</tr>
<tr>
<td>• People</td>
<td>16</td>
</tr>
<tr>
<td>• Process</td>
<td>24</td>
</tr>
<tr>
<td>• Technology</td>
<td>27</td>
</tr>
<tr>
<td>• Operating Model</td>
<td>33</td>
</tr>
<tr>
<td>How do I implement DevOps?</td>
<td>38</td>
</tr>
<tr>
<td>Deloitte Accelerators</td>
<td>42</td>
</tr>
<tr>
<td>Client Examples</td>
<td>46</td>
</tr>
</tbody>
</table>
What is DevOps?
What is DevOps?

DevOps is a new way-of-working that improves value delivery for the customer and enables benefits for both development and operations.

**Definition**

DevOps is a new approach to optimize and manage end-to-end service delivery and operations. It applies a set of principles to transform the entire software delivery lifecycle to introduce new practices enabled by technology.

**DevOps principles**

- Culture of shared responsibility and collaboration
- End-to-end ownership of services
- Multi-disciplinary teams
- Incremental value delivery
- Flow optimization in the delivery process
- Automate (almost) everything
- Measurement of everything
- Continuous improvement

**Software Delivery Lifecycle**

**Goal**

DevOps primary goal is to improve the flow from an idea towards value for the customer, enabled by an environment in which multidisciplinary teams work collaboratively to continuously deliver high quality solutions, in a faster pace, that qualify for operations.

**New DevOps practices:**

- Continuous Integration
- Continuous Testing
- Continuous Delivery
- Continuous Operations

**Benefits**

- Increases the frequency and quality of deployments and releases
- Improves innovation and risk-taking
- Realizes faster time to market
- Improves solution quality and operational reliability
- Improves the Mean Time to Recover (MTTR)

**Applying DevOps principles to the SDLC lead to new practices that benefit both Development and Operations**
DevOps is becoming the norm in software delivery and is increasingly being adopted & matured across enterprises, becoming the new best practice

Pre-2008

- DevOps

In IT, traditional waterfall methods of application development were losing ground to iterative methods such as agile. Speed became the goal, which took priority over development and deployment processes.

2008

- The chronic conflict between Dev & Ops is explored

Based on personal experience living in the world of Dev and Ops, Patrick Debois from Belgium starts investigating the chronic conflict between Dev and Ops.

2009

- The “DevOps” term is coined

Andrew Shafer and Patrick Debois meet at the DevOpsDays 2009 and later at Velocity conference, the term is picked up: “10+ Deploys a Day – a collaboration between Dev & Ops at Flickr” – Velocity, 2009.

2010

- The grass roots movement takes off

DevOps expands upon the practices of “infrastructure as code” and continuous integration and deployment. DevOps principles start being applied to the IT value stream.

2011

- “DevOps is the future”

March 2011, Gartner predicts “By 2015 DevOps will be adopted by 20% of the Fortune 2000.”

Most CIOs and IT organizations are looking into doing work differently.

2015

- DevOps incorporated into SAFe

SAFe is rapidly gaining traction in the enterprise arena, where DevOps is adopted and scaled across.

2016

- DevOps is the new norm for high-performing companies

“Clearly, what was state-of-the-art three years ago is just not good enough for today’s business environment.”

Most CIOs and IT organizations are looking into doing work differently.

2018

- State of DevOps report defines 5-stage approach

From level 0 to 5, a descriptive, pragmatic approach is introduced to guide teams and mature DevOps initiatives, a report sponsored by Deloitte

2019

- Enterprises embed more IT functions in their teams next to ‘Dev’ and ‘Ops’

“organizations are embedding security (DevSecOps), privacy, policy, data (DataOps) and controls into their DevOps culture and processes.”

Deloitte Tech Trends 2019

© 2022 Deloitte The Netherlands
DevOps practices

DevOps practices apply continuous automation cycles throughout software development and operations processes.

**Continuous Integration**
- The streamlining of internal development by integrating code into a shared repository several times a day. Each check in is then verified by an automated build, allowing teams to detect problems early in the cycle.

**Continuous Testing**
- Automating and integrating tests into the software delivery chain, and automatically executing those tests against each build of the code base.

**Continuous Delivery**
- Is the process of delivering code that is production ready and is kept in an always releasable state, so it can be deployed (automatically) to production at any given time based on business needs.

**Continuous Operations**
- Is proactively managing the solution based on feedback loops. Monitoring and telemetry become part of the backlog. Processes such as patching also fall under this practice.
Why do I need DevOps?
Traditional function separation for Development and Operations

Prior to DevOps change release frequency was low, Development and Operations worked separately to serve business demands, having completely opposite mindsets.

1. Focusses solely on development activities
2. Operational requirements are unclear between environments making the hand-over cumbersome
3. Operational feedback is only retrieved after completing a release
4. After releasing, developers are no longer involved

Development
Changes Features

Request features

Business

Demand
Stability

Operations
Safeguards Stability

1. Focusses solely on operational activities
2. Operational requirements are unclear, needing ad-hoc changes to the environment
3. Operational understanding and experience is gained only when the app is released
4. Operators manage and control software written by others

Deloitte DevOps Point of View
DevOps unifies the mindset of Development and Operations

Today business wants to release on demand. With DevOps, both functions continuously collaborate to align business demands within the software delivery lifecycle.
Where is DevOps applicable?
Indicators for DevOps Applicability
Several indicators help to determine if DevOps is applicable for your organization

**DevOps is Applicable**

- Management trusts delivery teams to work autonomously and only shares a product vision
- Multiple teams are responsible to manage the end-to-end lifecycle of a single product
- Environments where IT solutions are changing rapidly
- An incremental delivery process that focuses on early value delivery
- Desired product end-state is unknown, changing business requirements give guidance on steering development
- People have great affinity with software and technology and are not change averse
- Product is software that could be delivered as-a-service

**DevOps is Not Applicable**

- Management requires direct involvement in the delivery process and makes all decisions
- Product or service delivery does not require a multi-disciplinary (cross-functional) team
- Environments where IT solutions have low change rate
- Your delivery process has many sequential constraints, where outputs equal required inputs for consecutive process steps
- Desired product end-state is known and business requirements do not often change
- People have no affinity for new technologies, and are change averse
- Products are tangible, typically consisting of semi-finished products provisioned by multiple partners that don’t have a direct relation with each other
What is needed for DevOps to work?
DevOps dimensions

The DevOps operating model is structured along People, Process and Technology, each dimension is necessary for successful DevOps.

**People**
Establish a DevOps Organization & Culture with cross-functional teams that are open and trustful
- T-shaped employees
- Foster continuous learning and development to build cross-functional capabilities and a mindset open to continuous change
- Transformational leadership & balanced metrics to drive DevOps culture
- Open and transparent communication enable feedback and short learning cycles

**Process**
Establish standardized interconnected process in the software development (and operation) lifecycle
- Continuous Everything: integration, delivery, testing, monitoring, release management and planning
- Continuous Integration and Continuous Delivery are key to build quality into DevOps processes
- Establish interconnected processes across all phases of development and operations for consistent and predictable deployments

**Technology**
Improve toolset to support the delivery and automation of the process specifically to accelerate software delivery activities
- Container-based delivery and immutable infrastructure blocks
- Leverage the vast DevOps tooling landscape to automate and support Continuous Integration and Continuous Delivery and minimize user intervention
- Support of dynamic environment configuration to help remediate the current bottleneck in testing environment availability

**Operating Model**
- People, Process and Technology combined in a governance model for the DevOps way-of-working
- Teams deliver services end-to-end in the DevOps Target Operating Model
## DevOps Organization & Culture

The principles of the DevOps way-of-working have extensive implications on the organization structure, as well as on the culture of the workforce.

<table>
<thead>
<tr>
<th>DevOps principle</th>
<th>Implication for Organization &amp; Culture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Culture of shared responsibility and collaboration</td>
<td>Teams are accountable for progress and output, not an individual team member. Team setup is persistent and co-located to improve collaboration and performance.</td>
</tr>
<tr>
<td>End-to-end ownership of services</td>
<td>Team resources are allocated by services instead of organizational functions. Teams take end-to-end accountability and responsibility (vertically integrated) for the delivery of a service.</td>
</tr>
<tr>
<td>Multi-disciplinary autonomous teams</td>
<td>Teams are setup vertically, end-to-end responsible for the whole lifecycle of a product. It contains balanced T-shaped skilled personnel from various domains (cross-functional) to achieve its targets.</td>
</tr>
<tr>
<td>Incremental value delivery</td>
<td>Work is broken down into small pieces to continuously deliver value to the business using iterative and frequent releases.</td>
</tr>
<tr>
<td>Flow optimization in the delivery process</td>
<td>Elimination of waste, shift left and limit work in progress optimizes the flow in the delivery process. Teams test as early and as often as possible, minimize handoffs and maximize checkpoints to reduce dependencies and risks.</td>
</tr>
<tr>
<td>Automate (almost) everything</td>
<td>Tools automate as many tasks and process steps as possible in the delivery process to drastically reduce time, effort, and risk of human errors.</td>
</tr>
<tr>
<td>Measurement of everything</td>
<td>Everything is monitored and measured by a balanced metric system focused on the speed and stability of service delivery.</td>
</tr>
<tr>
<td>Continuous improvement</td>
<td>Teams organize retrospectives, (automated) feedback loops, and touchpoints with the business in order to continuously improve their delivery and way-of-working.</td>
</tr>
</tbody>
</table>
DevOps principles are the starting point for an organization structure

Based on DevOps principles, an organization allocates resources by service instead of functions to enable end-to-end ownership and increase agility within teams.
Key DevOps roles and responsibilities

Several key roles should be represented in a cross functional DevOps team; a team member with a T-shaped profile can fulfill more than one role.

The **Developer** writes and verifies code, fixes bugs, executes patch management, maintains asset and configuration repository and functions as 2nd line support.

The **Test Engineer** creates and executes test scripts, automates tests, supports usability testing & UAT, and manages test environments and test data.

The **Business Analyst** engages the business for requirements, helps defining features, user stories & test cases, and validates designs.

The **Scrum Master** facilitates the team on processes & approach, manages impediments and enables continuous improvement.

The **Operations Specialist** executes day-to-day technology operations (functional maintenance), monitors technology operations, performs Problem Management, manages change processes (Approves/Rejects).

The **DevOps Engineer** configures, maintains and monitors the CI/CD toolset.

*The scope of DevOps engineer role depends on the maturity of the shared infrastructure function.*
## T-shaped profiles

Ideally, DevOps team members have a T-shaped profile, teams have a combination of different profiles covering all knowledge and skills areas.

### Why we advise T-shaped profiles

A T-shaped profile entails that a team member covers different knowledge areas and skills in varying levels of expertise.

A team with T-shaped profiles does not have a hierarchy since everyone’s skills and knowledge complement each other.

A lack of hierarchy brings a team closer together and creates a sense of shared ownership.

### Knowledge Areas*

- Business Value Optimization
- Business Analysis
- Architecture & Design
- Programming
- Continuous Delivery
- Test Specification
- Infrastructure Engineering
- Security, Risk & Compliance
- Courage
- Teambuilding
- DevOps Leadership
- Continuous Improvement

### Skills Areas*

- Business Value
- Business Optimization
- Agile
- Architecture & Design
- Continuous Delivery
- Test Specification
- Infrastructure Engineering
- Security, Risk & Compliance
- Business Analysis
- Continuous Improvement

### Levels

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 1</td>
<td>Novice&lt;br&gt;Strict obedience to rules, no experience, little situational perception, no discretionary judgement</td>
</tr>
<tr>
<td>Level 2</td>
<td>Competent&lt;br&gt;Still limited with situational perception, knows the aspect guidelines and treats all attributes and aspects separately, yet equally</td>
</tr>
<tr>
<td>Level 3</td>
<td>Proficient&lt;br&gt;Sets priorities, actions are seen partly in longer term goals, deliberate planning, standardized procedures</td>
</tr>
<tr>
<td>Level 4</td>
<td>Expert&lt;br&gt;Perceives deviations from the normal patterns, makes decisions more easily, assesses situations as part of the “big picture”</td>
</tr>
<tr>
<td>Level 5</td>
<td>Master&lt;br&gt;Has a wealth of experience, creative solutions and visions, breaks the rules when needed, uses analytic approaches sparingly, makes good decisions quickly yet professionally</td>
</tr>
</tbody>
</table>

---

*DevOps Agile Skills Association (DASA)

© 2022 Deloitte The Netherlands
The mindset of a DevOps team member

DevOps team members foster certain cultural aspects contributing to the end-to-end ownership of services

**A mindset of effectiveness**
We continuously improve our delivery to improve our effectiveness. We define effectiveness as our ability to adapt to "market" circumstances and the success (value) of the product features delivered. Note that this also includes the effectiveness of activities, such as backlog prioritization

**A mindset of taking responsibility**
All members of our team are responsible for the complete product, which includes the full delivery cycle as well as operating/providing customer support throughout the lifecycle of the product in a collaborative mindset

**An engineering mindset**
We have the desire to utilize our knowledge, skills, and creativity to solve problems, implement product features, and optimize our delivery process. We do not settle for the current status quo. We strive to improve our craftsmanship

**Build quality in**
Quality is built-in from the initiation of the teams up to discharge. It is at the heart of every activity. It is never compromised. We value full transparency

**Inspirational and fun environment**
An environment in which people perform at best, where they feel inspired, where they want to be, feel welcomed and are encouraged to think out of the box

**Continuous learning & Continuous improvement**
We have the desire to explore and learn in all activities we do. We strongly believe that working together, transparency, and sharing knowledge is vital. We care about our job enough to not pass the buck, we want to learn all the parts as a whole and not just our little world

**Experimentation & Risk taking**
We always conduct experimentation using solid methodologies to ensure ideas are evaluated on the real value instead of the assumed value

**A mindset of product thinking**
Our application is our product. It must deliver value when it runs in production. We need continuous improvements to ensure the application delivers value now and in the future
Factors influencing DevOps organization design

DevOps theory doesn’t always apply to practice, client specific factors need to be taken into account for an applicable and effective DevOps organization design.

DevOps organization design is client specific

Setting up DevOps teams or an entire DevOps organization requires understanding of existing, but also future organizational structures.

Client specific factors might increase the complexity and effort that is required to transform towards a DevOps organization.

There is no “one-size-fits-all” DevOps organization design. Client specific factors must be taken into account.

Example considerations for an effective “to-be” DevOps organization design are:

- Keeping some functional hierarchy intact to facilitate collaboration with the enterprise
- Re-architecting the technology stack to enable DevOps practices
- Adhering to some degree of separation of duties to comply with regulations
Process
The DevOps model is significantly different from the traditional IT model.

DevOps integrates the application lifecycle into an end-to-end, iterative process.

**Traditional IT Model**

- **Define**
- **Design**
- **Develop**
- **Test**
- **Release & Deploy**
- **Operate**
- **Retire**

**Implications**
- Straightforward sequential process assuming all is known
- Big chunks of work
- Maximizes each process-step, big-bang delivery
- Many separated functions (silos) with (manual) handoffs
- Specialization (I-shaped roles)
- Rigid change ability

**Target State DevOps Model**

- **Plan**
- **Continuous Integration (Source)**
- **Continuous Testing**
- **Continuous Delivery**
- **Continuous Operations**
- **Monitor**
- **Retire**

**Continuous Improvement**

**Implications**
- Complex iterative process to manage unknowns
- Small chunks of work
- Maximizes flow, incremental delivery
- Fewer handoffs (less silos)
- Generalists (T-shaped roles)
- More flexible to adapt to change
DevOps leverages technologies to automate the software delivery process

While extending agile, DevOps optimizes the software delivery process by leveraging CI/CD which automatically promotes developer’s source code to operational solutions.

DevOps extends the Agile process and puts emphasis on the software delivery cycle and operations...

... its practices focus on bridging the stage gate gaps between phases to accelerate throughput by promoting more frequently with smaller products...

... next to this, DevOps practices incorporate feedback loops continuously in the process for value creation and learning by experience.

CI/CD pipelines integrate process into technology:

- **Automates almost everything** - Automation drastically reduces time, effort, and risk of human errors.
- **Done means released** - Deliver releases for pre-deployment or deploy new releases to productions in minutes instead of months.
- **Everything in Version Control** - Versioning ensures that no work gets overwritten and that the latest versions are built upon.
- **Builds Quality Into the Process** - The quality of every deliverable is guaranteed; errors and problems are detected early.
Technology
Architecture design in a DevOps implementation

Realizing the full potential of DevOps requires a flexible, secure and agile technology stack, which often brings the need for architectural redesign.

Increased end-to-end delivery speed and maturity poses increased demands on an organization’s technology stack. Re-architecting for DevOps therefore requires decisions on some key technology areas:

- **Modularity & Independent Deployment**: External dependencies on other technologies and processes can put at risk the rapid deployment of new features. Minimize dependencies by designing modular systems.

- **Resource & Infrastructure Management**: Even with the fastest deployment pipeline, going live will not be possible without the proper supporting infrastructure. Effectively managing (standardized) resources is critical for effective DevOps.

- **Fit-for-purpose Supporting Technology**: The supporting technology to develop a CI/CD pipeline must be consistent with the organization’s objectives, available capabilities and maturity level.

- **Responsible DevOps**: In modern IT landscapes, speed cannot be achieved at the cost of security, privacy or compliance. Integrating a risk-based approach in the DevOps process and pipeline is fundamental to ensure full control.
Modularity as a mean to boost the impact of DevOps

By decreasing technical & functional dependencies among components, modularized systems can empower DevOps and further increase delivery speed and quality.

Modularity is a **spectrum**, whose level can be influenced by decisions on both **software functional segregation** and **system architecture** (e.g. having a messaging layer, splitting DBs etc.)

### Features of modular systems

<table>
<thead>
<tr>
<th>Impact on DevOps</th>
<th>Modularity Level</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Independent Deployments</strong></td>
<td>Monolithic Architecture</td>
</tr>
<tr>
<td>Small components enable organization to create small, autonomous teams (e.g. Amazon rule), which are <strong>free to deploy</strong> their changes <strong>independently</strong>, without having to <strong>wait</strong> for the <strong>entire application</strong> to be deployable.</td>
<td>Service Oriented Architecture</td>
</tr>
<tr>
<td><strong>Contract-based Coordination</strong></td>
<td>Microservices Architecture</td>
</tr>
<tr>
<td>By enforcing standard services’ interfaces (i.e. how the different services talk to each other), delivery teams can <strong>decrease</strong> the <strong>amount</strong> and <strong>impact</strong> of <strong>inter-team dependencies</strong> and make collaboration <strong>more efficient</strong>.</td>
<td></td>
</tr>
<tr>
<td><strong>Localized Failure</strong></td>
<td></td>
</tr>
<tr>
<td>As <strong>failures</strong> are <strong>localized</strong>, teams can more confidently <strong>deploy stand-alone software</strong> since, if the service is faulty, the system entirety will not be compromised.</td>
<td></td>
</tr>
</tbody>
</table>

© 2022 Deloitte The Netherlands
Cloud is an accelerator for DevOps

Though DevOps can be applied in an on-premise or hybrid environment, cloud infrastructure is an accelerator that enables high levels of automation.

**Cloud**

- DevOps can be **accelerated** by leveraging resource automation principles that are widely applied in the cloud. Several capabilities of cloud infrastructure offer **distinct advantages** when implementing DevOps:
  - Easy and rapid creation and replication of environments ensures **scalability** and **business continuity**.
  - Fully automated (de-)provisioning of infra using **infrastructure-as-code** enables resource flexibility, resulting in a **resilient infrastructure**.
  - Services provided by cloud vendors focus on **high usability** and **easy integration**, reducing the additional expertise required in teams.

**On-premise**

- Achieving an **agile** and **modular** technology stack is definitely possible in an on-premise environment. However, there are **some disadvantages** to building and managing your own infrastructure:
  - **Limited control** over networking and security policies often prevent easy replication of environments and requires alignment with a wide range of stakeholders.
  - Hard boundaries on resource flexibility due to **hardware constraints** and integration with **legacy systems** limit an infrastructure’s resiliency.
  - On-premise often leads to a significant **increase in scope** of DevOps teams, and require experts with **specific knowledge**.

---

© 2022 Deloitte The Netherlands

DevOps Deep Dive by TT&A
A legion of tools are available to support DevOps practices
As DevOps is a tool intensive practice, a thorough tool selection incorporating client maturity is a crucial part of the DevOps transformation

DevOps is a tool intensive practice
The voluminous amount of tools available, delivering one or multiple capabilities brings consequences when transforming towards a DevOps organization

• Selecting the right tools requires an iterative approach (a procedure per capability)
• Selection is, among others, based on engineering skills, prior experience, or tools (architecture) already in place

XebiaLabs published A sample selection of DevOps tools categorized in capabilities
Patterns to setup a CI/CD pipeline

Selecting the right set of tools (Best-of-Suite, Best-of-Breed or hybrid) for the CI/CD pipeline depends heavily on IT maturity and tech-savviness of the organization.

**Applicability of the toolset**

### Best-of-Suite

**“Bundle of end-to-end enterprise software applications”**

**Advantages**
- Control – one central place to manage users, applications etc.
- User experience – one similar user interface for the pipeline
- One integrated platform to process the pipeline from

**Disadvantages**
- Standard solution – Often a bit more rigid than best-of-breed solutions, offering less room for specialization
- Partner dependency – The performance and development of the features depend on a single provider
- Integration focus – New features have the objective to integrate with the core instead of being the best of its kind

### Hybrid

**“Best of both worlds”**

**Advantages**
- Quality cascade – iterate upon the current setup and consider best option available

**Disadvantages**
- Effort to determine concurrent tools – The hybrid approach considered a thorough reconsideration for every requirement between Best of Breed and Suite

### Best-of-Breed

**“Selecting the best product of its kind”**

**Advantages**
- Flexibility – you are not depending on a one-size-fits-all solution
- Independent – you can pick and choose new capabilities regardless of the core solution

**Disadvantages**
- Maintenance – requires knowledge of the setup of each, and dependencies between applications
- Vendor segregation – issue solving might cover multiple vendors with different support models

---

Deloitte DevOps Point of View
A basic functional flow through a CI/CD pipeline

A CI/CD Pipeline can be built in various ways considering the desired tooling patterns, covering the same functional flow with different tools and integrations between them.
Security at every step of the software delivery lifecycle

DevOps requires a strong focus on integrating security automation in the deployment pipeline, as well as including security* in the regular development process.

The items above are examples of DevOps activities/controls that can be put in place across the different stages of a CI/CD pipeline, for both software and infrastructure.

* Although security is an inherent component of every step of the DevOps cycle, security begins by incorporating it into the design of the application.
Operating Model
The changing IT Operating Model
DevOps disrupts multiple dimensions of the IT Operating Model; synergies arise when combined with next generation technologies and ways-of-working

<table>
<thead>
<tr>
<th>Organizational Model</th>
<th>Architecture Model</th>
<th>Delivery Model</th>
<th>Release Model</th>
<th>Practice Model</th>
<th>Hosting Model</th>
<th>Sourcing Model</th>
<th>Support Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hierarchical Organization</td>
<td>Monolithic</td>
<td>Waterfall</td>
<td>Annually &amp; Quarterly</td>
<td>Process Driven</td>
<td>Physical Datacenter</td>
<td>In-house</td>
<td>Tier Based</td>
</tr>
<tr>
<td>Traditional</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product Line Organization</td>
<td>N-Tier</td>
<td>Agile</td>
<td>Monthly</td>
<td>Practice Driven</td>
<td>Virtual Machines</td>
<td>Traditional Outsourcing (Towers)</td>
<td>Skilled Support</td>
</tr>
<tr>
<td>Evolved</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Autonomous Teams</td>
<td>Microservices &amp; API</td>
<td>Continuous Delivery</td>
<td>On Demand</td>
<td>Value Stream</td>
<td>Cloud &amp; Containers</td>
<td>Service Brokering</td>
<td>Self Service &amp; Self Help</td>
</tr>
<tr>
<td>Next Gen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DevOps Target Operating Model (TOM)

A DevOps TOM lands the DevOps way-of-working in the organization by describing the DevOps practices using Deloitte’s framework ‘Technology TOM in a box’

The Technology TOM in a box is a holistic framework to describe the governance structure of an IT organization and how it functions as an entity.

The capabilities of the DevOps practices will be described through the dimensions of the Technology TOM in a box.

The DevOps Target Operating Model (TOM) outlines the governance structures on how an IT organization should govern and operate the DevOps practices.
Our proven approach to DevOps TOM design

DevOps disrupts the IT Operating model (shape and size). Using a number of inter-related dimensions we describe in 3 steps *What, How* and *Who* needs to change

DevOps TOM

Using a number of inter-related dimensions we describe in 3 steps:

1. **What** services the DevOps organization is going to deliver, and to whom

2. **How** are these services delivered, how to structure and source required capabilities

3. **Who** is responsible for service delivery in a DevOps culture; roles, processes and tooling

The DevOps TOM is a number of inter-related dimensions that describe the governance structure of an IT organization and how it functions as an entity.
Detailing the DevOps practices

The DevOps TOM describes capabilities that are part of the DevOps practices

DevOps capabilities (non-exhaustive)

- System Architecture
- Development Practices
- Development Environments
- Source Code & Version Control
- Automated Code Build

- Test Automation
- Test Environments & Data
- Bug Reporting & Defect Management
- Data Security

- Release Orchestration
- Deployment Automation
- Configuration & Asset Management
- Environment Management
- Infrastructure automation
- CI/CD Metrics and Tracking

- Telemetry & Logging
- Metrics / Dashboards
- Self-Service Portals
- Incident Response
The DevOps way-of-working has several key changes on the IT operating model that differentiate from the traditional way-of-working.

### Differentiators of the DevOps TOM

#### People
- **Service Oriented Organization**
  - Resources organized around services, focused on value
- **Service Brokering**
  - As-a-service delivery of composed products with an ecosystem of partners
- **Autonomous teams**
  - Self-organizing teams, autonomously in their decision making
- **Cross functional teams**
  - Multiple disciplines in a team that share E2E responsibility

#### Process
- **Release On Demand**
  - DevOps practices (CI/CD) for iterative and frequent releases
- **Continuous delivery**
  - Slicing the work into smaller chunks that add value to customers
- **E2E traceability**
  - Greater visibility and transparency on service delivery
- **Value stream**
  - Siloes broken down between business and IT
- **Self Service & Self Help**
  - Service request portals automate request fulfillment

#### Technology
- **Automate everything**
  - Tools automate processes, minimize handoffs and maximize checkpoints
- **Modular App. Architectures**
  - Architecture (e.g. Micro-Services & API) enables autonomous teams to develop and release independently
- **Infrastructure as code**
  - Next gen hosting (e.g. Cloud & Containers) enables automated environment provisioning
- **Venture-Capital Budgeting**
  - Funding dependent on minimal viable product, and its performance

Source: Deloitte Technology TOM in a box
How do I implement DevOps?
DevOps transformation approach

The transformation approach outlines the Deloitte service used to collaboratively drive and coordinate a successful DevOps transformation, consisting of four different phases.

<table>
<thead>
<tr>
<th>Start Project</th>
<th>Transition Delivery</th>
<th>Coach &amp; Scale</th>
<th>Roll-off</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 – 6 months</td>
<td>3 months</td>
<td>6 – x months*</td>
<td>1 – 2 months</td>
</tr>
</tbody>
</table>

**Project Management**
- **Stream 1:** DevOps TOM
- **Stream 2:** CI/CD

**Key objectives**
- Define mission and IT services in scope
- Start design track for DevOps TOM and roadmap
- Start execution track to for CI/CD implementation

**Start Project**
- Start with two project tracks; design and execute

**Transition Delivery**
- Transition into a bi-modal organization where coaching and co-creation is key

**Key objectives**
- Setup a DevOps CoE as driver of the transformation
- Onboard CI/CD Services as the first DevOps team
- Replace traditional governance with DevOps ceremonies

**Transition Delivery**
- Onboard, coach and handover new virtual teams to scale the bi-modal organization

**Key objectives**
- Shape the DevOps TOM by learning from experiments
- Experiment with CI/CD to accelerate value delivery
- Show value using DevOps Metrics

**Coach & Scale**
- Roll-off once the client is mature enough to continue themselves

**Key objectives**
- Stop coaching efforts once the organization is mature enough
- Handover of transformation responsibilities

---

* Depending on # of teams to onboard
What are the challenges of implementing DevOps?

DevOps majorly challenges skills of everyone involved: management and team. It may lead to development slowdown and will not compensate for lack of responsibility

<table>
<thead>
<tr>
<th>People</th>
<th>Process</th>
<th>Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Challenges:</strong></td>
<td><strong>Challenges:</strong></td>
<td><strong>Challenges:</strong></td>
</tr>
<tr>
<td>• <strong>Skill Challenge:</strong> Difficult to replicate skills across every DevOps team in your organization and acquiring people with the DevOps skills may be challenging</td>
<td>• <strong>Lack of Overview:</strong> Progress and stability are spread across the teams and overview may lack</td>
<td>• <strong>Application architecture:</strong> Transforming an organization from being monolithic to work with microservices and API’s</td>
</tr>
<tr>
<td>• <strong>Management Challenge:</strong> DevOps can mean management challenges for your team leads</td>
<td>• <strong>Dev Slowdown:</strong> Operations may hamper progress in development</td>
<td>• <strong>Choice of Tools:</strong> difficult to choose appropriate tooling when knowledge about the technology behind it is lacking</td>
</tr>
<tr>
<td>• <strong>No Magic:</strong> DevOps will not compensate for potential lack of responsibility in your organization</td>
<td>• <strong>Self-organization Challenge:</strong> Clear Service-Level Structures in operations may be challenging</td>
<td>• <strong>Integration with existing IT landscape:</strong> Pipeline platform integration with enterprise IT (network, infra, hardware, security, applications) is often experienced as difficult</td>
</tr>
</tbody>
</table>
### Key Lessons Learned

During our engagements we gathered the following key takeaways that we will bring to future projects:

<table>
<thead>
<tr>
<th>People</th>
<th>Management support is crucial</th>
<th>Break down silos</th>
<th>Assign champions from client</th>
</tr>
</thead>
<tbody>
<tr>
<td>You cannot “buy DevOps”</td>
<td>Management involvement is crucial in the DevOps transformation, as change starts and stops with them</td>
<td>Break down silos. Not only between departments, but also between organizations</td>
<td>Ensuring the support of the client can be accelerated by having a champion from their side spreading the DevOps culture and principles</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Process</th>
<th>Collaboration is key</th>
<th>DevOps ≠ Agile</th>
<th>Focus on E2E responsibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consider secondary impacts</td>
<td>DevOps requires close collaboration across dev, test, operations and business teams to effectively deliver value to the organization</td>
<td>DevOps can be seen as an extension of Agile, with the same level of agility driven into development, test and operations</td>
<td>Limit handovers as much as possible, teams must adopt an end-to-end responsibility for the product or service they deliver</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Technology</th>
<th>Show value quickly</th>
<th>DevOps ≠ Automation</th>
<th>Use Cloud as an accelerator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modern architecture is critical</td>
<td>'Prove' the DevOps concept by demonstrating working solutions early and often (e.g. CI/CD tooling)</td>
<td>Release and Deployment Automation or App Release Automation are only a part of DevOps. End-to-end automation is key</td>
<td>Ensure parity between cloud and on-premise implementations (e.g. Azure DevOps)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operating Model</th>
<th>Change incrementally</th>
<th>DevOps ≠ Organization</th>
<th>DevOps journey is client specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaborate with Tech Stream</td>
<td>Apply an agile approach for adopting DevOps and introduce change incrementally with focus on the outcomes</td>
<td>There are different organizational patterns to setting up DevOps and it doesn’t always have to be making it a separate organization</td>
<td>DevOps target operating model transformation depends heavily on where the client is in their DevOps journey</td>
</tr>
</tbody>
</table>

Platforms built on modern architectures based on modular design, decoupling and good componentization enables deeper adoption of DevOps

'Prove' the DevOps concept by demonstrating working solutions early and often (e.g. CI/CD tooling)

Release and Deployment Automation or App Release Automation are only a part of DevOps. End-to-end automation is key

Ensure parity between cloud and on-premise implementations (e.g. Azure DevOps)

Design the DevOps operating model in parallel and close collaboration with the technology implementation

Apply an agile approach for adopting DevOps and introduce change incrementally with focus on the outcomes

There are different organizational patterns to setting up DevOps and it doesn’t always have to be making it a separate organization

DevOps target operating model transformation depends heavily on where the client is in their DevOps journey

© 2022 Deloitte The Netherlands
Deloitte Accelerators
Deloitte DevOps Maturity Assessment Offerings

To understand the current state DevOps capabilities and to identify areas for improvement, we have two assessment methodologies available.

What is it?

DORA provides a SaaS questionnaire that benchmarks DevOps performance against 2000+ leading Enterprises across industries

- "Gold standard" for DevOps assessments
- Compare your results against others in industry
- Two assessments included – one to baseline and one to measure progress
- Provides priorities for capability improvement

Benefits

- 180 questions along each of the DevOps domains (Release Planning, Continuous Development, Continuous Integration, Continuous Testing, Continuous Deployment, Continuous Monitoring)
- Assesses maturity against desired future state
- Identifies areas for capability improvement

Cost

Deloitte receives a 30% discount from DORA; will be an additional cost on top of pilots

Included in price of DevOps KickStart or DevOps Dojo

Sample Output

To understand the current state DevOps capabilities and to identify areas for improvement, we have two assessment methodologies available.

Deloitte’s DevOps Maturity Assessment (DDMA) is an extensive questionnaire for assessing current state against desired future state maturity of DevOps capabilities across the DevOps domains: from Release Planning to Continuous Deployment and Monitoring.
# Deloitte Global Accelerators

Our Global DevOps Community of Practice has a wide variety of accelerators available that we can use in our engagements

<table>
<thead>
<tr>
<th>Learning</th>
<th>Sales Materials</th>
<th>Tools and Enablers</th>
<th>Eminence and Point of Views</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Learning Series</strong></td>
<td>Proposal templates &amp; DevOps brochures</td>
<td>DevOps Local toolkit</td>
<td>Eminence</td>
</tr>
<tr>
<td>Basic introduction course to various DevOps practices</td>
<td>Templates &amp; brochure to help you kickstart your DevOps proposal</td>
<td>An integrated toolkit of local DevOps tools to gain hands-on experiences</td>
<td></td>
</tr>
<tr>
<td><strong>Learning Resources</strong></td>
<td>DevOps Qualifications</td>
<td>Deloitte supplied tools</td>
<td>DevOps Point of Views (PoV)</td>
</tr>
<tr>
<td>A collection of documents to assist learning DevOps and specific elements or specific vendors</td>
<td>‘Quals’ to help you display Deloitte’s capability to deliver DevOps transformations, including tooling</td>
<td>Tools that can be supplied for client engagements:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Agile Manager</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• HP Application Lifecycle Management</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Fortify</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• JRebel</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Performance Center</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• SonarQube</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Unified Functional Testing and UFT Pro</td>
<td></td>
</tr>
<tr>
<td><strong>Videos &amp; Demos</strong></td>
<td>DevOps Case studies</td>
<td>Enablers</td>
<td></td>
</tr>
<tr>
<td>A collection of videos and demos regarding Deloitte methodologies and instructions for DevOps tooling</td>
<td>Case studies of client engagements, with success stories and demos. The Client demo can showcase DevOps automation capabilities</td>
<td>Enabling materials for specific vendors, industries, such as the Cloud Compass, PoC for SAP or Google Cloud enablers, Cards for Agility, Technology TOM in a Box</td>
<td></td>
</tr>
</tbody>
</table>

Note: non-exhaustive, Global examples, which may be updated continuously

© 2022 Deloitte The Netherlands
Client example 1

DevOps journey and CI/CD pipeline implementation
Client example 1: Global parcel delivery services company

We took our client on a DevOps transformation journey across all five dimensions to streamline the software delivery lifecycle of their mission critical system.
Client example 2
A CI/CD Pipeline for a Banking Platform
Client example 2: CI/CD pipeline

Deloitte developed an Open Banking Platform as a global asset with an CI/CD pipeline to ensures continuous integration and deployment.
Client Example 3

Test Automation for a Banking Application
Client example 3: Automating End-to-End Testing

We helped our customer to setup their testing capability, a vital but time-consuming part of the software delivery process.

Customer aims to speed up release and ensure software quality through UI-based E2E Integration and Acceptance testing with the Selenium Framework.

<table>
<thead>
<tr>
<th>Test Phase</th>
<th>System</th>
<th>Integration</th>
<th>E2E</th>
<th>Acceptance</th>
<th>PAT</th>
</tr>
</thead>
</table>
| Type of tests | • System tests  
  • Bug testing  
  • Unit testing  
  • Deployment tests | • Deployment tests  
  • Integration tests based on user stories  
  • Regression tests (E2E) | • Deployment tests  
  • Business Acceptance Regression tests (E2E) | | • Non-functional tests  
  • Performance  
  • Security  
  • Disaster Recovery  
  • Prepare for go-live execution |
Client example 3: Automating End-to-End Testing
Using a test automation framework based on Selenium, Deloitte delivered a total of 15 automated end-to-end test cases

![Diagram of DevOps process]

- **Jenkins**
  - CI/CD Automation Server

---

**Given** I navigate to “Deloitte.com”
**And** I click “About Deloitte”
**Then** I should see title “About us”

---

- **Develop**
  - Behavior-Driven Development
  - cucumber

- **Build**
  - Build Automation Tool
  - Apache Ant

- **Run**
  - Web Browser Automation Framework
  - Trigger test run based on
    - Time
    - Code commit
    - JIRA update

---

© 2022 Deloitte The Netherlands

Deloitte DevOps Point of View 53
Recommended resources
In case you got excited and would like to learn more...

Books:
• **The DevOps Handbook**: How to Create World-Class Agility, Reliability, and Security in Technology Organizations
  – by Gene Kim, John Willis, Patrick Debois, Jez Humble

• **The Phoenix Project**
  – by Gene Kim, Kevin Behr, George Spafford

• **Continuous Delivery**: Reliable Software Releases through Build, Test, and Deployment Automation
  – by Jez Humble and David Farley

• **Accelerate**: Building and Scaling High Performing Technology Organizations
  – by Nicole Forsgren, Jez Humble, Gene Kim

Websites:
• [https://notafactoryanymore.com/](https://notafactoryanymore.com/)

Video’s:
• John Smart (Deloitte colleague) at the DevOps Summit: [https://www.youtube.com/watch?v=-Rq-fuiKNCU](https://www.youtube.com/watch?v=-Rq-fuiKNCU)
Additional contributors

A big thanks to all our colleagues that contributed to this Point of View

Andries van Dijk
Director
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 52 04 87 05
anvandijk@deloitte.nl

Tessa van den Berg
Senior Manager
Human Capital
Deloitte Consulting
+31 6 10 04 25 78
tesvandenberg@deloitte.nl

Andreas Boon
Manager
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 83 33 96 09
aboon@deloitte.nl

Boris Smits
Consultant
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 82 67 44 36
bsmits@deloitte.nl

Tom Pennings
Consultant
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 22 17 87 17
tpennings@deloitte.nl

Jan Prüst
Consultant
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 82 51 86 04
jprust@deloitte.nl

Audrey Sie
Analyst
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 50 06 26 13
ausie@deloitte.nl

Jasper Lelijveld
Analyst
Technology, Strategy & Transformation
Deloitte Consulting
+31 6 50 06 25 77
jajelijveld@deloitte.nl
Deloitte refers to one or more of Deloitte Touche Tohmatsu Limited, a UK private company limited by guarantee ("DTTL"), its network of member firms, and their related entities. DTTL and each of its member firms are legally separate and independent entities. DTTL (also referred to as "Deloitte Global") does not provide services to clients. Please see www.deloitte.nl/about to learn more about our global network of member firms.

Deloitte provides audit, consulting, financial advisory, risk advisory, tax and related services to public and private clients spanning multiple industries. Deloitte serves four out of five Fortune Global 500® companies through a globally connected network of member firms in more than 150 countries and territories bringing world-class capabilities, insights, and high-quality service to address clients’ most complex business challenges. To learn more about how Deloitte’s approximately 245,000 professionals make an impact that matters, please connect with us on Facebook, LinkedIn, or Twitter.

This communication contains general information only, and none of Deloitte Touche Tohmatsu Limited, its member firms, or their related entities (collectively, the "Deloitte Network") is, by means of this communication, rendering professional advice or services. Before making any decision or taking any action that may affect your finances or your business, you should consult a qualified professional adviser. No entity in the Deloitte Network shall be responsible for any loss whatsoever sustained by any person who relies on this communication.

© 2022 Deloitte The Netherlands