White paper

devops
the journey
of a continuous delivery implementation

Your business technologists. **Powering progress**

Devops the journey

The journey of a continuous delivery implementation

Contents

Introduction

Currently DevOps (an abbreviation of Development and Operations) is creating a worldwide buzz due to its ability to decrease IT operational costs while improving software quality and speeding up time to market. DevOps reorganizes software development and operational departments in order to remove the operational boundaries that currently generate inefficiencies in terms of time to market for new features and software quality. It includes the implementation of Continuous Delivery, Continuous Integration, Automated testing, Application monitoring and other best practices in Software Development and Operations.

Gartner

Gartner finds that 90% of the interviewed organizations describe themselves as being busy with DevOps. 44% openly admit that they are still trying to find out what it means. DevOps is known to improve a number of KPIs in development, operations and maintenance of applications: e.g. manual testing effort, ops effort, regression errors, time to move a feature from request to production.

Any organization that claims to work with DevOps but has not seen any change in these KPIs, did not truly implement the best practices of DevOps.

Best practices: calms

It is the introduction of a number of best practices and transformations that each carries a small benefit towards the larger goal of faster, higher quality releases.

These best practices are summarized as 'CALMS', see figure 1.

Each of these transformations has an effort and a cost. The importance depends on the organization's current situation, the type of application and the desired benefit of the DevOps implementation. For instance, not every organization and not every application requires multiple releases on a daily basis; the benefit of moving an organization from one release per week towards multiple releases per day, may not justify the costs.

Figure 1: CALMS – Best practices

Culture	▸ DevOps is not a methodology or framework, rather some high minded principles and a willingness to break down silos ▸ Without culture, improvement attempts will fail ▸ Value Dev AND Ops encourage collaboration
Automation	▸ Many processes can be automated ▸ Automation removes manual, error prone tasks allowing people to focus on Quality ▸ Automation is the enabler for continuous delivery
Lean	▸ Reduce waste ▸ Focus on purpose, process and people
Measurement	▸ If you can't measure, you can't improve ▸ Measure current processes and plan for improvement ▸ A successful DevOps implementation will measure everything it can as often as it can...
Sharing	▸ Sharing is the way to break down silos ▸ Sharing is also part of the "feedback loop" ▸ Resolving issues is about working together

Starting DevOps: approach

To start a DevOps implementation program the organization assesses its current situation and desired targets for each application silo. Using a DevOps maturity model, experts are able to assess the current 'as-is' situation of the organization in a number of different DevOps key areas. They propose a 'to-be' situation based on the organizations needs and budget.

Figure 2 shows an overview of the activities.

Based on the outcome of this assessment, a DevOps transformation program is defined. It is advisable to start with a number of pilot projects. These will prove the viability of the DevOps approach to the rest of the organization and create an internal DevOps pool of competence that can coach and help transform the other parts of the organization.

So, starting DevOps in the organization is looking for the right project to start with.

Criteria for choosing a pilot project

The choice of the pilot project(s) is important and should take into account a number of criteria:

1. Benefit of DevOps to the applications
Back-office applications which are end-of life, and hardly change any more are not good candidates. Focus on applications that require regular and frequent updates / releases.

2. Risk and importance of the application
It is not a good idea to start with a business critical application, or with an application that is integrated with all other applications.

3. Technology stack
DevOps principles and benefits are general, and apply to all software, regardless if it is custom development, configuration of packages (ERP), mainframe or SAP. However, the relative maturity and impact of the DevOps best practices, and certainly the required tooling are different depending on the technology stack, so the organization must choose what technology stack would benefits most or most easily.

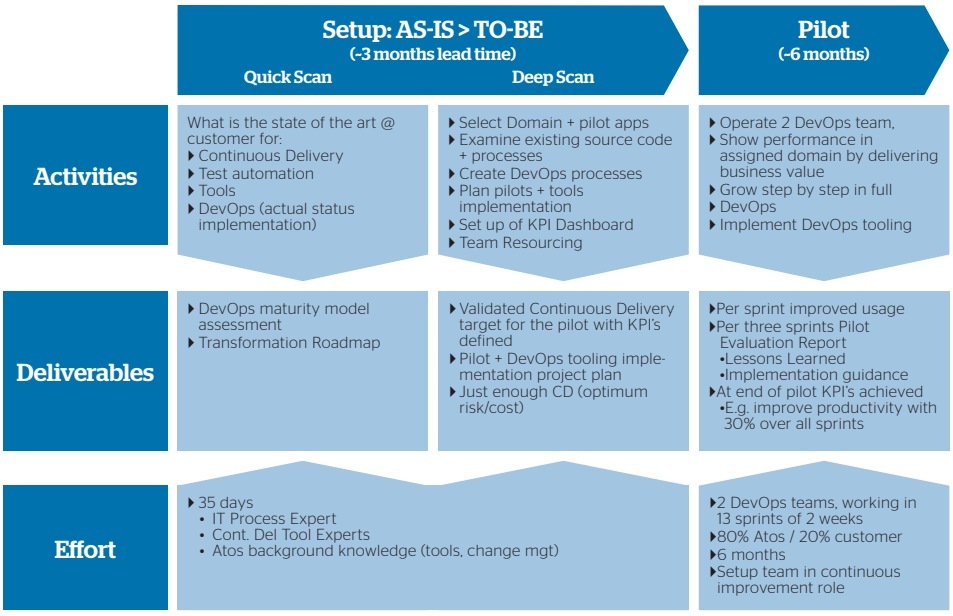
Implementing DevOps and the IT operations budget

The costs for implementing DevOps is a lot lower for new applications than it is for existing applications in a maintenance mode because investments such as increasing the automated test coverage will only be recovered if they can be executed frequently, which is more likely for new application development projects.

On the other hand, the benefits of a DevOps approach are more easily measureable when introducing DevOps in existing applications, because then a clear 'before ' and 'after' comparison is possible.

For new development projects, the effort of implementing Devops, such as installing the tooling can easily be absorbed in the project setup, but for existing applications, it means additional budget and resources need to be foreseen to write the automated testcases, and that is usually paid from the IT operations budget.

Figure 2: Timeline of a DevOps transformation project



DevOps implementation

Figure 3 shows the different aspects involved with transforming an organization to a DevOps way of working, grouped according to 3 principal axis: Mindset, methodology and tooling. Note that, while Devops may be perceived as 'just a new set of tools and automation to implement, experience shows that a DevOps implementation is actually 75% organizational and processes transformation and only 25% tooling and technology implementation.
So any DevOps implementation that neglects the organizatorical, and people components is not likely to realize all the expected benefits.

Where to start?

DevOps transformations can start from different parts within the organization[1].

Each approach has advantages, disadvantages and risks:

Start from the Operations teams
This requires is a relatively easy place to start, with gradual transition path where resources migrate from a central operations organization into DevOps teams, while maintaining an Operations core organization that maintains the shared resources, like the PaaS platform. However, it requires that the operations teams have the skills and time to start automating and scripting their processes.

Set up a skunk-works team
Before rolling out DevOps to the entire organization, handpick resources from different teams (or recruit externally) to form a DevOps skunk-works project. This team must have the freedom to work out their own DevOps methodology.

Start from Development
Allow Development teams to work in a DevOps fashion in Dev and Test, and allow them to engage their own Ops members. Initially they hand-over their deliverables to DevOps experts with the operations teams.

Greenfield implementation
This is the rare situation, most likely to occur within startup organizations. These can start a completely new team using DevOps methodologies from the start to develop new applications. In this case there is no real transition to implement.

Force DevOps on the complete organization
This is the highest risk transformation, whereby teams are just put together and told to work in a DevOps manner. The risk is that many of the existing resources will just not be able to work in a DevOps manner, and leave the company, creating gaps in the organization, knowledge, and impacting the existing projects and SLAs. This method is only advised if there are a sufficient number of DevOps able resources, or the organization believes they can re-staff fast enough to make up for the resources lost through attrition.

Organizational transformation

The organizational transformation is a lot more than telling the existing Development and the Operations teams that as from now they are one 'DevOps-team' working in a DevOps way and reporting to just one manager. The purpose of the organizational transformation is to instill within an IT organization the sense of 'Lean' that currently exists in factory floors such as Toyota's.

This 'Lean' attitude is based on four key principles:

- Focusing on the customer
- Doing things right the first time
- Empowerment, including the ability to 'stop the line' to fix issues immediately when they occur
- Continuous improvement and having a learning organization

Successful implementations

For successful implementations it's important to take into account the following:

Incentivation
Dev & Ops teams were traditionally incentivized differently: Dev teams typically on effort per feature, and features per time, Ops teams on availability SLA's. Within a DevOps organization both now should be incentivized in the same way, and rather on business KPIs, like 'time between request and deployment'.

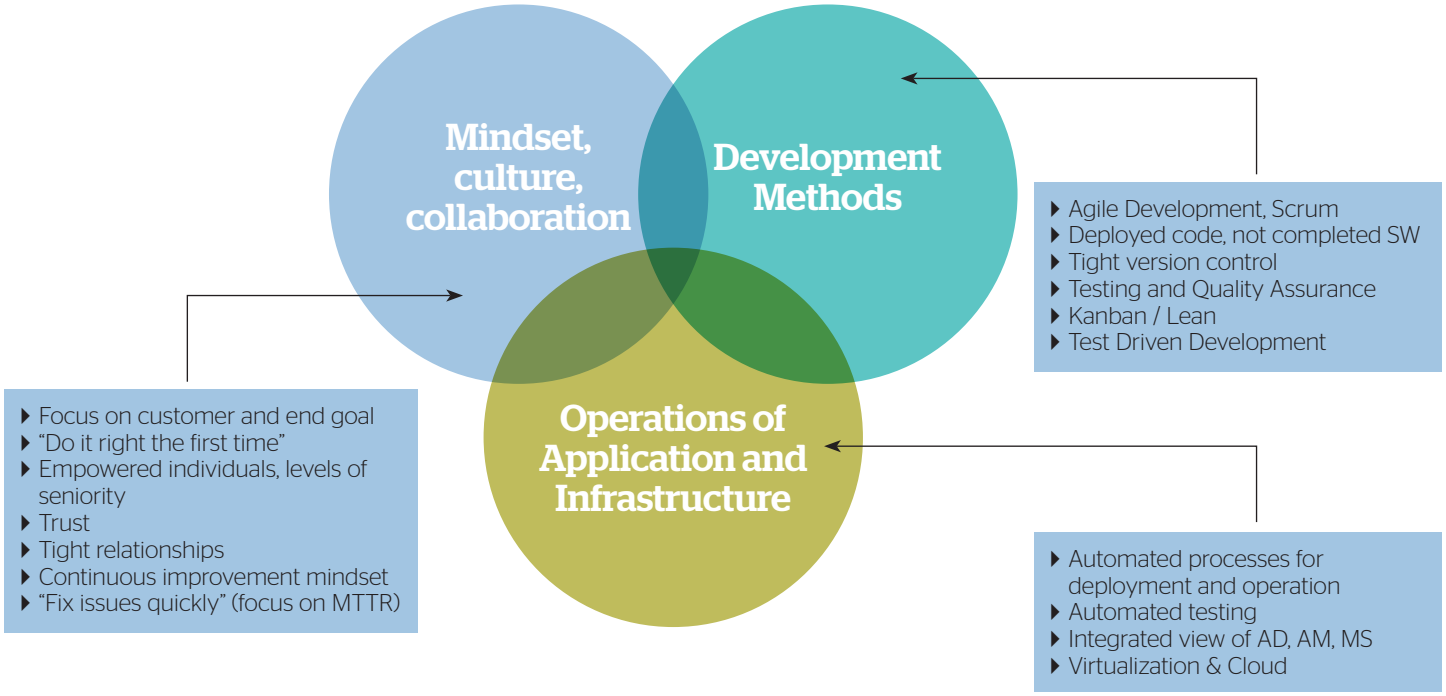
Training & staffing
Working in a DevOps way includes the ability to 'stop the line' if problems occur, followed by 'swarming', whereby the whole team assist in fixing the problem. That implies that everyone should be able to assist in any role in the team. It also requires an attitude of collaboration, self-improvement and enablement that may not be present in every member of the current Dev & Ops teams, without possibly extensive retraining and reskilling, and the assistance of experienced DevOps coaches.

Organizational structure
Implementing DevOps implies that Dev and Ops are reorganized into different teams following the application silo structure. However, the definition of what this implies, specifically from the Ops side, is open to debate. It makes no sense, for instance, that the operational maintenance of items that are inherently shared, such as the Private Cloud infrastructure, or the NAS, is divided up over the DevOps teams. It should remain in a central service team. Other items, such as the database operations, need to be examined on a case by case basis.

It does not imply that no common governance organization is needed. As we see, the importance of overall governance teams for enterprise architecture, QA, shared IT infrastructure and portfolio management remains. They become even more important when implementing DevOps, because they are the only way to avoid that different applications teams go too far in their own way, and collaboration between teams and application silo's collapses.

[1] Enterprise Devops, Aligning Development and Operations", Martin Maisey, BJSS.

Figure 3: Different aspects involved in the transformation to a DevOps way of working



"Through 2015, 80% of the risks associated with attaining DevOps program objectives will stem from how organizational change is managed" (Gartner, G00236104)

Processes

Implementing DevOps with the purpose of speeding up the release cycle of applications will likely require a change in the existing processes and governance within the organization.

The first step in the DevOps transformation process is to create a value chain chart of the processes required to bring a feature from business request to production: what steps, quality gates need to be passed, how much time is spent in each step and how much time is spent waiting for the next step to start.

Based on this value chain chart the organization can identify the bottlenecks that need to be resolved. For example: if the purpose of a DevOps implementation is to be able to go from Requirement to Production in a matter of weeks, then it is insufficient to have development and operations to collaborate to release an approved feature in a matter of days, if the feature approval and budgeting process still takes months.

A number of processes need to be examined when implementing DevOps:

- Project methodology
- Quality gates & Compliance
- Development & architectural discipline
- Working with outsourcers

Project methodology

Classical software development methodologies - such as 'push' Agile with sprints - start to break down when DevOps is taken to go to a release-to-production schedule more frequent than once every week.

In that case project management needs to be augmented to a 'pull' methodology such as Kanban[2]. For larger projects, multiple DevOps teams will work in parallel towards delivering features that span multiple applications or components of an application. Implementing Advanced Agile project methodologies like Scaled Agility, Disciplined Agile Delivery and Atos ACE to coordinate work between teams can help here. The concept of an Agile Sprint within a project doing Kanban can still be maintained for portfolio planning reasons and to align milestones between Devops and Agile projects, even if the DevOps project is actually delivering releasable features on a daily basis.

Quality gates and Compliance

DevOps will actually help and improve Quality Gates and Compliance, but the number of releases can be more frequent than doing in-person Quality Gates meeting schedule allows. A different implementation of the gates approval process is therefore needed.

Automated testing
Automated testing will (automatically) generate reports for quality gates; quality gates are changing from formal meetings to an approval workflow. Integration with the quality systems in place (e.g. CMMi) will require effort.

Compliance
Compliance will improve as well since Automated Deployment scripts leave logs of who did what and when and obviate manual access to servers for doing installations. Of course, this needs to be discussed with the auditors.

Development and architectural discipline

Strictly enforced coding guidelines
DevOps requires, in order to implement some of its key practices like single branch development, that all developers use a number of strictly enforced coding guidelines. Think about the use of feature toggles to disable the activation of features that are still in development when code is in production. These must be strictly enforced and deviations must be captured early through automated code quality checks and peer reviews.

Another issue is the implementation of Continuous Integration; this requires a clean decoupling of applications so they can be guaranteed to be testable in isolation, as well as having an easy way to test version X of application A, with version Y of application B, in situations where updates to application A and B may be as frequent as daily. This is done by having applications only interface with each other through documented APIs, in adherence to an enterprise architecture and governance as well as the introduction of the practice of peer code reviews to capture deviations early. The use of web-services, and the implementation of a SOA governance methodology focused on managing the Application Lifecycle Management (ALM) of these web-services, enforced by an enterprise architecture team, is a good way to keep things under control.

When your IT-operations have been outsourced

If a company uses an external company to deliver software in a project mode, then it is important that the processes are integrated and adapted to the DevOps principles of the organization that will receive the application. The tool chains used by both parties should be adapted and integrated so that, for instance, deployment scripts written by the outsourcers also function on the infrastructure of the receiving organization. No manual tests are needed when code is handed over from the outsourcers to the customer's organization.

2 Kanban, Successful Evolutionary Change for your Technology Business", David J Anderson.

Tooling

The first day of a new DevOps project is always to get a 'Hello World' program from Development into a production-ready deployment state. This 'clears the pipe' of the development and deployment tool-chain, and makes sure that as of day 1 the software is fully compliant to the DevOps methodology. This 'Hello world' program then evolves during the rest of the project into the desired application.

In order to be able to do this, constructing this tool-chain should be one of the starting points of any DevOps transformation.

The term 'tool-chain' not only refers to actual tools used during development, but also to middleware components such as the application databases. These may require upgrading or implementing a number of additional middleware components to support DevOps best practices.

This includes components such as:

- Databases that now need to be able to support multiple versions of the same client, with roll-back and roll-forward capability
- The CMDB, which now needs to be able to be updated automatically whenever deployments are done
- Monitoring infrastructure

A full DevOps tool-chain includes:

- Version management system that triggers a build at least daily
- Fully automated unit, integration, acceptance and performance testing tools
- Fully automated code quality auditing
- Tools to schedule peer reviews
- Packagers
- Artifact repositories

As large batches of tests run frequently and therefore are not allowed to take a long time, use is made of Cloud infrastructure in order to parallelize tests, and for their ability to allow automatic creation of servers from scratch, in adherence to scripts that describe the desired production infrastructure ('Infrastructure-as-code')

New DevOps support tools are released in the market on an almost daily basis. Existing tools are upgraded by their vendors to support DevOps methodologies. Though it is possible to use a complete, performant, high quality and supported DevOps toolset based only on open source components, this may not be the ideal solution for every organization. That's the reason that a DevOps transformation project should include an audit of the existing tooling and additional needs of both the Development and Operations parts of the organization prior to choose a tool chain vendor.

The existing development, testing and production infrastructure must also be re-evaluated. As mentioned before, testing infrastructure can benefit from the Cloud in order to rapidly and automatically create servers that resemble the production infrastructure in order to run all the automated tests with a high throughput. Production infrastructure and the way applications are deployed on the production infrastructure may also require changes to perform automated deployments with zero downtime and minimum risks. The introduction of dedicated clusters of production servers per application silo is a perfect solution to do this.

Atos and DevOps

The easiest and fastest way to implement DevOps is to hire experienced DevOps practitioners for coaching the teams and assisting in the organizational transformation. However, since DevOps is relatively new, there is little chance of finding these people within the existing organization. And because of high demand, they are also hard to find in the market, and command a premium price. Atos has been using DevOps methodologies on a number of internal projects as well as on customer's projects. We're able to provide a wide range of offerings to assist customers in implementing DevOps. (see figure 4)

In addition to the possibilities below, Atos provides a number of specific offerings to resolve a number of typical DevOps implementation issues:

- Atos DevOps Academy trains customer resources in project management, tooling and general Agile and DevOps skills.
- Atos automated testing tools allow customers to reduce the cost of introducing automated testing. This includes 'as-a-service' offerings, for specific pain points, such as running tests for mobile apps on a very wide selection of mobile devices
- The Atos Canopy Cloud PaaS offering provides all the necessary Dev and Test cloud infrastructure to support a DevOps methodology
- Web-service SOA governance implementation experience
- Template DevOps tool-chains based on open source components

Figure 4: How Atos can assist in DevOps transformations and add employees to DevOps teams

Type	Capacity	Management	Way of working	Result
Consulting: QuickScan, Deepscan	Atos	Customer	Customer	Transformation plan/ implementations
Secondment (time/material)	Customer	Customer	Customer	Professionals/ Coaching
Resource pool (co-sourcing)	Atos	Customer	Customer	Expertise/DevOps tooling
Execution (Project)	Atos	Atos	Customer	Project result in continuous delivery
Service (Out tasking existing project)	Atos	Atos	Atos	Product of service with SLA
Outsourcing (continuous enterprise)	Customer/Atos*	Customer/Atos*	Customer/Atos*	Business KPI's

About Atos

Atos SE (Societas Europaea) is an international information technology services company with 2013 annual revenue of € 8.6 billion and 76,300 employees in 52 countries. Serving a global client base, it delivers IT services through Consulting & Systems Integration, Managed Operations, and transactional services through Worldline, the European leader and a global player in the payments services industry. With its deep technology expertise and industry knowledge, it works with clients across different business sectors: Manufacturing, Retail & Transportation; Public Sector & Health; Financial Services; Telcos, Media & Utilities.

Atos is focused on business technology that powers progress and helps organizations to create their firm of the future. It is the Worldwide Information Technology Partner for the Olympic & Paralympic Games and is listed on the NYSE Euronext Paris market. Atos operates under the brands Atos, Atos Consulting, Worldline and Atos Worldgrid. For more information, visit: www.atos.net.

For more information:
Please contact **marketing-nl@atos.net**