EFICODE QUICK GUIDE

DEVOPS

WWW.EFICODE.COM

IN A NUTSHELL

Devops is a software development method for a post-agile world. It is based on automation, virtualization and smart tool choices. Boring, repetitive work is left for robots while people get to focus on the best part of software production: creative and customer-oriented development work.

CONTENTS

EFICODE QUICK GUIDE

DEVOPS

Management

”How many things have been denied one day, only to become realities the next!”

Jules Verne (1828 - 1905)

DEVOPS – A THIRD GENERATION SOFTWARE DEVELOPMENT METHOD

DevOps is part of a global change in the life of businesses and public organizations where IT and IT infrastructure are becoming increasingly virtualized. Consulting firm Deloitte named devops as one of the top technology trends of 2014.[1]

BY EFICODE'S SURVEY

66 % of Finnish IT decision-makers see DevOps as an interesting prospect for their company.[2]

Software development begins with a business need. A company might be looking for a web service that makes the cash register sing, or a digital tool that makes life easier for the company's employees. Once the idea has been formed and the goals have been set, software development may begin. After a while the development team presents a finished product and the users embrace it with fanfare. Or not. All too often cost estimates are exceeded, deadlines are missed and – worst of all – the users are unhappy.

What, then, is the problem?
Primarily, it comes down to the way software is developed.

The first mainstream software development method is known as the waterfall model. It is centered on rigid, predetermined requirements, the completion of which is closely monitored according to a painstakingly detailed agreement. As one cannot anticipate everything that happens during development, this inflexible process typically results in a disastrous user experience. If nothing else, the budget becomes bloated and the original schedule has to be forgotten at the stage when the supposedly finished software is found to require fixing.

FACTS

18 %

Smaller development and maintenance costs

= **A reduction of nearly one fifth in expenses.[3]**

50 %

Fewer failures

= **A two-fold increase in the chance of a successful product release.[4]**

30 ×

Faster release

= **Deployments 30 times more frequent and 8 000 times faster.[4]**

20 %

More business

= **A 17–23% increase in revenue and number of new clients.[3]**

[1] - Deloitte Development LLC (2014) Tech Trends 2014: Inspiring Disruption. Deloitte University Press.

[2] - Market research company Value Clinic Oy (2014) Devops Suomessa (Devops in Finland) research report.

[3] - CA Technologies (2013) TechInsights Report: What Smart Businesses Know About DevOps. CA Technologies.

[4] - Puppet Labs (2013) 2013 State of DevOps Report. IT Revolution Press.

FOCUS

WATERFALL

Focused on **the agreement**

﹀

AGILE METHODS

Focused on **the customer**

﹀

DEVOPS

Focused on **the delivery**

To solve the problem of requirements that change during development, agile software development methods were introduced. Agile methods approach software development as an iterative process and emphasizes communication. Agile methods have removed many of the issues plaguing the waterfall model, but it does not address the collaboration with the IT operations teams. Software buyers are also worried about the "open tab" mentality often associated with agile development, threatening to bloat the budget of the project.

Software development is becoming virtualized and automated

What is good about agile is also present in DevOps, a third generation development method that relies on virtualization and automation. With DevOps, software development is moved to the cloud and all manual, repetitive work stages in, for example, quality assurance and software release are reduced.

The results of the development process may be presented to buyers and end users as they are completed. DevOps also makes it possible to combine system requirements with automatic quality assurance, which means everyone involved, from developer to manager, may monitor the progress of the development process in real time. The product development process is always focused on delivering the customer benefit.

An organization that makes use of DevOps principles is testing and releasing software continuously and automatically. The development work is usually done in a virtualized environment. This reduces errors and costs resulting from infrastructure management and the development work itself. When the developers and the IT operations crew share common metrics, it helps these professionals who usually prefer to work on their own to work together for a better user experience.

DevOps answers the challenges of its predecessors

Agile methods solved many of the waterfall model's problems. DevOps picks up where agile leaves off and also gets rid of many of the downsides of the agile methods.

No more bottlenecks in the development chain

Agile methods made the practices of software production flexible. However, agile methods fail to take into account IT environments and the tools used in software production. This creates bottlenecks in the development chain that slow down the development process. In DevOps, technological solutions also play an important role. Old data center solutions may well be replaced by cloud services or virtual environments, and manual processes with automatic ones.

PROBLEMS / SOLUTIONS

01 02

PROBLEM WITH THE WATERFALL MODEL	VS	AGILE DEVELOPMENT SOLUTION
Software is planned like buildings - before any actual work is done. Changing plans later is expensive or impossible.	— —	Software is developed in small iterations and in cooperation with the customer throughout the project.
Projects often have a fixed agreement with regard to price and requirements. Negotiations are slow and based on guesswork.	— —	The agreement made at the beginning of the project concerns common procedures and a prioritized list of requirements, not the final product.
Software is developed in stages where testing starts after the development is "ready".	— —	Software is developed in small, independent iterations, the contents of which can be altered before an iteration begins.
Visibility into the development process is poor.	— —	The customer sees the actual status of the project at regular intervals.
Changes that occur during the development process are not prepared for.	— —	Changes are welcome and they are prioritized as parts of the future iterations.
The system is only tested at the end of the implementation phase when making changes is expensive. Some of the necessary fixes may even be impossible to implement.	— —	The functionality of the system is already tested during the implementation phase to ensure that the requirements selected for an iteration may all be done from start to finish during the iteration.

03 04

PROBLEM WITH AGILE DEVELOPMENT	VS	DEVOPS SOLUTION
Delivery of new features to the customer is often delayed.	— —	DevOps tools are used to test and release new features as they are completed.
Completed software components are not compatible with each other.	— —	Open interfaces and test automation make it possible to divide development into independent yet compatible parts.
Quality of the product is not ensured properly prior to release.	— —	DevOps tools and practices help automating quality assurance and reduce the need for repetitive manual work.
New features break old functions.	— —	The quality of existing functions is ensured quickly and automatically after each change.
Budget goals and deadlines are missed.	— —	The tools and procedures of DevOps increase the transparency and predictability of the development work.
Developer teams and IT operations crews are not cooperating.	— —	Developer teams and IT operations crews agree upon responsibilities together. Their goals are unified.

It's no wonder cure

DevOps has a lot to offer for an organization willing to learn, but it is no silver bullet. DevOps will not get rid of manual work completely. What it does is move repetitive and numbing work and tasks that require speed to the machines. DevOps allows developers to focus on the work that adds value – on the quality of the product and the user experience.

Here are a business director's steps to a more efficient future:

01

Talk to the IT director in your organization about the status of your software production and how DevOps principles might suit your needs. Also show him or her the technical section of this guide.

02

If you cannot find a person who is enthusiastic about DevOps to lead the project from within your organization, find one outside of it. Make sure that the service provider not only completes the mechanical installations, but is also ready to turn DevOps into knowledge capital for your organization.

DEVELOPMENT

DEVELOPMENT IN A TRADITIONAL ENVIRONMENT	DEVELOPMENT WITH DEVOPS
IT environments are inflexible and expensive to maintain.	A virtualized environment is an affordable way to maintain basic functions. It also helps to prepare for load spikes.
Dependence on one supplier narrows the skills palette involved and forces unnecessary compromises.	The principles of DevOps support modularity and open interfaces. One does not need to get stuck with only one supplier.
The completeness of customer requirements is difficult to monitor. Predicting the release schedule might be impossible.	With DevOps, customer requirements may be centralized into one place and attached to the quality assurance system. When a feature is released, the customer can be notified immediately.
Human errors in quality assurance lead to uneven quality.	Test automation always handles quality assurance at peak efficiency.
People who work on product development are frustrated by repetitive work tasks.	Boring tasks are left for the machines while people are free to work creatively.

DEVOPS IS IDEAL FOR THOSE WILLING TO CHANGE

DevOps is ideal for organizations that:

01. Develop, maintain or buy software
02. Want to develop better digital services
03. Are looking for cost savings
04. Wish to improve their operations.

MORE ABOUT DEVOPS

So, your thirst for knowledge has not yet been quenched? Great! Dive deeper into the world of DevOps with the following sources:

THE PHOENIX PROJECT: A NOVEL ABOUT IT, DEVOPS, AND HELPING YOUR BUSINESS WIN (GENE KIM ET AL.) · A hit novel written by three major-league IT professionals that tells a fictional story about an organization struggling with a delayed project with a bloated budget. The web begins to untangle when DevOps comes into town...

BUILDING A DEVOPS CULTURE (MANDI WALLS) · A brief and concise guide about how a new kind of development culture may be built and how the change can be implemented within an organization.

DEVOPS.COM · Devops.com contains a comprehensive collection of DevOps-related writings from news and feature stories to blogs and articles.

EFICODE.COM/BLOG · The Eficode blog is a place where the writers of this guide, among others, offer their fresh views on DevOps. Find out whether your DevOps transformation will ever be over (it needs an expiration date!) and get clued in on the 5 principles for successfully scaling your DevOps transformation.

EFICODE ASSESSMENT

eficode

Get started by using Eficode's DevOps and Automation Maturity Assessment Service. The assessment consists of our experts impartially determining the current status of your internal and outsourced software projects and providing you with a detailed roadmap to improve the efficiency of your processes. It takes into account the pressure points in your software production infrastructure, environments, processes, tools and development culture. After the plan has been drawn, you may decide which actions to take.

WWW.EFICODE.COM/BLOG

eficode

DEVOPS MATURITY MODEL

WWW.EFICODE.COM

	001	002	003	004
	Development operations have been separated from the business knowledge. Starting a new development project is laborious.	Starting new development projects is agile, and there are practices in place for steering the project.	New projects can be connected to organization's strategic targets. Starting a new pilot project is easy.	There are real-time metrics available from the development, supporting decision-making and tracking the completion of strategic targets.
ORGANIZATION AND CULTURE	Design, development and quality assurance are separate from each other. Communication is primarily in writing.	Work is conducted in teams but development and quality assurance are separate from each other.	The teams work independently. They have total liability for the development and quality assurance of features.	The teams communicate with each other regularly and work together to improve their practices. Communication with the IT operations is continuous.
ENVIRONMENTS AND RELEASE	Products are environment-specific and they are compiled manually. Environments are installed and configured manually.	The system is divided into parts and the compiling environment is known. Some releases are automated.	Environments can be installed and configured automatically. Build and release processes are automated.	Releases may be conducted automatically and continuously. Migration and recovery processes work as expected.
BUILDS AND CONTINUOUS INTEGRATION	Product integration is automatic, but configuration and deployment are controlled manually. No artifact or change logs management.	The process starts team-specifically after every change. Tools are shared. Integration does not involve testing.	Integration covers the entire product and it is connected to acceptance testing. Dependencies are known and managed.	The development organization meets regularly. Collected metrics aim to speed up the feedback cycle and improve visibility.
QUALITY ASSURANCE	Quality assurance is conducted completely by hand and primarily after development.	Unit testing or static code analysis is in place for some parts of the product.	Features visible to the end users are covered with automatic tests. Testers participate in the development process.	Acceptance tests present system requirements clearly and guide the development of the system as much as possible.
VISIBILITY AND REPORTING	Reports are made by hand when necessary.	Code integration, unit testing and code analysis are visible to the team.	The status of requirements can be monitored in real time in relation to tests and released features.	Metrics are collected from the product development process and used as a basis for improvement.
TECHNOLOGIES AND ARCHITECTURE	Technologies and tools are obsolete or are not fit for current requirements.	Technologies are growing old and the architecture is only partially adaptive or the interfaces are lacking.	Technologies are modern or well supported. The interfaces are well documented and exist for all key functionalities.	The architecture and technologies are appropriate and enable reaching business targets efficiently.

DEVOPS

Experts

“Perhaps we think up our own destinies,
and so in a sense deserve whatever happens to us,
for not having had the wit to imagine
something better.”

Iain Banks (1954 - 2013)

WHAT IS DEVOPS?

DevOps binds together the software production chain from a customer need to a delivered product or service.

BY EFICODE'S SURVEY

72 % of Finnish IT decision-makers think time-to-market is the most important feature of DevOps. [1]

While agile development focuses on communication and software development in small units, DevOps keeps the focus on the customer, automation and visibility. The term 'DevOps' describes a culture of doing that involves modern tools and processes.

DevOps brings together all the parties involved with the development process: the customer, developers, software suppliers and quality assurance and service maintenance teams. Many DevOps principles, such as continuous delivery, depart from the traditional division of responsibilities within the organization. DevOps places focus on smooth cooperation that is typically supported by centralized requirements management and automatic quality assurance.

The principles of DevOps are not entirely new. Many parts of DevOps, virtualization and automatic quality assurance, for instance, have already been around long before the term 'DevOps' was coined. The purpose of DevOps is to bring all good and efficient practices together and use them to build a coherent whole.

EVOLUTION OF VALUES

WATERFALL

Methodical Approach

AGILE

Cooperation
Incrementality

DEVOPS

Virtualization
Visibility
Automation

[1] - Market research company Value Clinic Oy (2014) Devops Suomessa (Devops in Finland) research report.

BENEFITS

WHAT ARE THE BENEFITS OF DEVOPS?

DevOps unifies development and maintenance and extends visibility all the way to the customer:

01. Requirements can be found in one clearly defined place.

02. Development environments are known and can be provisioned automatically.

03. Efficient version control enables controlled development and maintenance.

04. New features are tested automatically during different stages of the process.

05. Automatic quality assurance is connected to the original requirements.

06. New product versions can be released either completely automatically or with the push of a button.

07. Server environments are virtual or in the cloud. They can be configured and deployed automatically.

08. The operation and efficiency of services can be monitored across the entire organization.

PROBLEMS / SOLUTIONS

PROBLEM WITH TRADITIONAL DEVELOPMENT	VS	DEVOPS SOLUTION
Requirements from the customer, flaws in the system and new development ideas are scattered all over – and where is the documentation?		To make cooperation possible, the requirements and documentation must be located in one place where they are visible to everyone involved.
System integration is risky and time-consuming.		Continuous integration helps to ensure functionality and compatibility as the product changes.
System testing is done only in conjunction with a potential version release.		Automatic acceptance testing and continuous integration make it possible to conduct Quality Assurance as changes are made.
System developers and production server administrators do not cooperate with each other.		Product release is automatic. The functionality of a release is the joint-responsibility of the developers and IT.
Server environments are created with the worst-case scenario in mind. This results in increased expenses and less flexibility.		Virtualization of server environments helps to scale down the service and control expenses when the service is on low use.
Development cycles drag on too long.		Automation and the common goals set for the development teams ensure that the entire production chain operates at lightning speed.
Teams in the production chain have their own separate goals that contradict each other.		The entire chain has a unified goal, which is then divided into smaller sections.
The different parts of the production chain receive feedback too slowly and the feedback is not directed properly.		Automatically generated feedback is instantly delivered to the correct team or person.

KEY COMPONENTS OF DEVOPS

DevOps is a combination of tools and a development culture shared by an organization. Automation is the core of DevOps. It facilitates product development that flows as a constant stream through the entire organization. The need expressed by the customer is automatically transferred from one work stage to the next all the way to the release and maintenance of a completed feature.

REQUIREMENTS MANAGEMENT

Problem: Requirements and bug reports are scattered in different places.

01. Requirements and system documentation are collected into one location where they can be accessed by the entire development organization.

02. The status of the requirements is monitored throughout the development process all the way to the release of the software. This way it is always known when a feature has been released and what changes are included in the release.

DEVELOPMENT ENVIRONMENT

Problem: Establishing a development environment takes an insane amount of time. The developers also work in different development environments.

01. It should be possible to install environments for the developers automatically with the help of configuration libraries and tools.

02. The environments of the developers should be connected to centralized requirements management and version control and, via continuous integration, to automatic quality assurance and deployment.

03. Development is guided by quick and automatic feedback via continuous integration, automatic quality assurance and code analysis.

CONTINUOUS RELEASE AND DEPLOYMENT

Problem: Making production releases takes a lot of time and work, and is prone to errors.

01. Continuous delivery aims to achieve a situation where a product can be released for production automatically after each change.

02. Configurations used during development should be connected to the configuration and virtualization of the services so that the release process can be automated.

03. The continuous delivery process may contain quality gates that halt the automated process until the results are reviewed and the process is allowed to continue.

04. Continuous delivery is an attitude that aims to automate repetitive processes so reliably that the automation may be trusted to release changes all the way to the end users.

HOW DNA DID IT: Annual savings range from millions to tens of millions.
https://www.eficode.com/cases/dna

ACCEPTANCE TESTING

Problem: Requirements are not connected to tests. Testing is primarily done manually and only after the development stage.

01. Requirements are connected to quality assurance so that their actual status is clear at all times.

02. At least that part of the quality assurance process that has to be conducted after every change is automated.

03. Automated test cases are written in natural language so that both customers and experts understand why and how the quality of the system is ensured.

```
Transferring money using a phone number
  Login to payment service        ${user}
  Input recipients phone number   ${phone}
  Input the payment amount        €50
  Accept payment
  Confirm the payment has been transferred
```

VIRTUALIZATION

Problem: Maintaining servers is expensive and the capacity far exceeds normal demand. The configurations on the test and pre-production servers differ from those on the production servers.

01. One of the cornerstones of devops is the service virtualization.

02. Virtualization makes it possible to keep the environments identical throughout the development process.

03. Virtualization allows for significantly reduced maintenance costs as virtual servers can be ramped up or down as required.

04. Virtualization makes development and quality assurance faster. Errors stemming from environments are reduced significantly.

MONITORING

Problem: The service is slow, but nobody knows why.

01. Service monitoring allows performance tracking of both the server and the service itself.

02. Modern monitoring tools know how to report the largest problems and help improve service performance.

03. Monitoring facilitates the long-term development and tracking of the service.

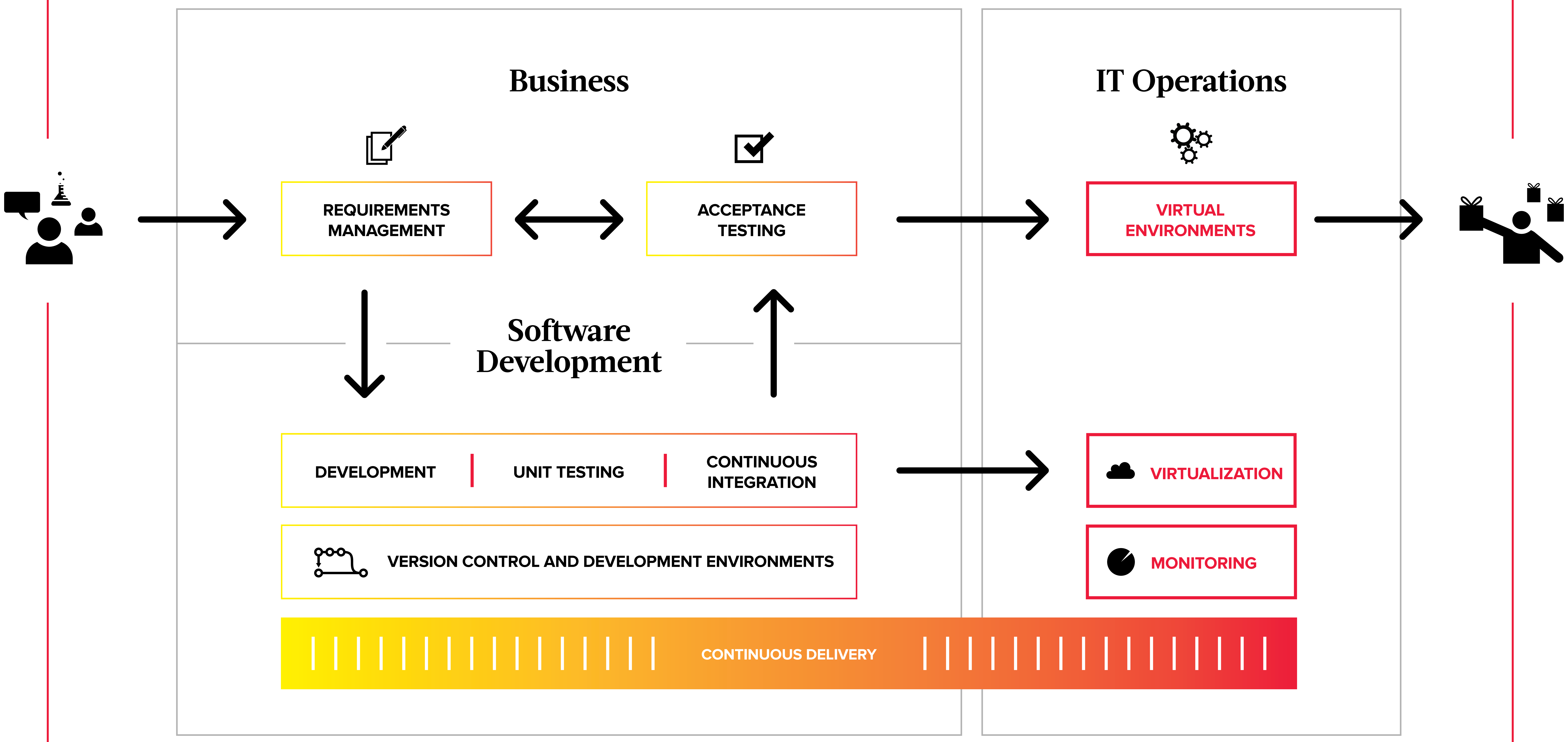
INTERFACES

Problem: The system cannot be tested automatically.

01. DevOps shifts the focus of product development from completing a monolith to small and independent units that communicate via interfaces.

02. The interfaces are well documented and follow both standards and the best practices.

KEY COMPONENTS OF DEVOPS



HOW TO GET STARTED

You can begin familiarizing yourself with DevOps right now. Start by thinking about where the biggest bottleneck is in your production chain and try to solve the problem with one of the methods mentioned earlier. Try automating your manual test cases or find out if your servers could be virtualized.

DO YOU WANT TO INTRODUCE YOUR TEAM TO DEVOPS? TRY THESE STEPS:

01. Chart the status of your software development processes.

02. Build a roadmap that allows you to conduct development in small increments.

03. Experiment with areas where the culture is open to change, or where most of your brightest and wildest innovators work.

04. Conduct a technical knowledge audit and try out different methods.

05. Form common principles and make them the practices of your organization.

06. Take a leap into the unknown! Example: allow developers to release changes directly into the production.

ON THE EFICODE BLOG:
(Acceptance) Test-Driven Development: An Introduction
https://www.eficode.com/blog/tdd-atdd

CONTENTS

DEVOPS MATURITY MODEL

WWW.EFICODE.COM

	001	002	003	004
LEADERSHIP	Development operations have been separated from the business knowledge. Starting a new development project is laborious.	Starting new development projects is agile, and there are practices in place for steering the project.	New projects can be connected to organization's strategic targets. Starting a new pilot project is easy.	There are real-time metrics available from the development, supporting decision-making and tracking the completion of strategic targets.
ORGANIZATION AND CULTURE	Design, development and quality assurance are separate from each other. Communication is primarily in writing.	Work is conducted in teams but development and quality assurance are separate from each other.	The teams work independently. They have total liability for the development and quality assurance of features.	The teams communicate with each other regularly and work together to improve their practices. Communication with the IT operations is continuous.
ENVIRONMENTS AND RELEASE	Products are environment-specific and they are compiled manually. Environments are installed and configured manually.	The system is divided into parts and the compiling environment is known. Some releases are automated.	Environments can be installed and configured automatically. Build and release processes are automated.	Releases may be conducted automatically and continuously. Migration and recovery processes work as expected.
BUILDS AND CONTINUOUS INTEGRATION	Product integration is automatic, but configuration and deployment are controlled manually. No artifact or change logs management.	The process starts team-specifically after every change. Tools are shared. Integration does not involve testing.	Integration covers the entire product and it is connected to acceptance testing. Dependencies are known and managed.	The development organization meets regularly. Collected metrics aim to speed up the feedback cycle and improve visibility.
QUALITY ASSURANCE	Quality assurance is conducted completely by hand and primarily after development.	Unit testing or static code analysis is in place for some parts of the product.	Features visible to the end users are covered with automatic tests. Testers participate in the development process.	Acceptance tests present system requirements clearly and guide the development of the system as much as possible.
VISIBILITY AND REPORTING	Reports are made by hand when necessary.	Code integration, unit testing and code analysis are visible to the team.	The status of requirements can be monitored in real time in relation to tests and released features.	Metrics are collected from the product development process and used as a basis for improvement.
TECHNOLOGIES AND ARCHITECTURE	Technologies and tools are obsolete or are not fit for current requirements.	Technologies are growing old and the architecture is only partially adaptive or the interfaces are lacking.	Technologies are modern or well supported. The interfaces are well documented and exist for all key functionalities.	The architecture and technologies are appropriate and enable reaching business targets efficiently.

Management

HEIKKI HÄMÄLÄINEN
Head of DevOps
heikki.hamalainen@eficode.com
+358 (0) 40 766 2610

RISTO VIRKKALA
CEO
risto.virkkala@eficode.com
+358 (0) 40 501 8434

Experts

MARKO KLEMETTI
CTO
marko.klemetti@eficode.com
+358 (0) 44 522 5927

KAJ JOKINIEMI
Vice President of DevOps
kai.jokiniemi@eficode.com
+358 (0) 40 592 6257

eficode

Eficode Oy
Pohjoinen Rautatiekatu 25, 00100 Helsinki Finland - +358 (0) 207 40 11 22

Eficode is a software company with offices in Finland, Sweden, Denmark, the Netherlands & Germany. We create the best software organizations in the world.