# DEVOPS: PEOPLE OVER PROCESS OVER TOOLS & DEPENDENCY INJECTION

Omar Lahkim
School of Science and *Engineering*
Al Akhawayn University in Ifrane
Ifrane, Morocco
o.lahkim@aui.ma

Houda Chakiri
School of Science and *Engineering*
Al Akhawayn University in Ifrane
Ifrane, Morocco
h.chakiri@aui.ma

***Abstract*— The Agile software development methodology kept attracting practitioners in the software engineering community, but it revealed that the agile mindset is still not enough to continuously deliver value to customers. DevOps solved this defect by extending the agile methodology and breaking the gaps between the development and operations teams to allow not only continuous delivery and continuous integration but also continuous deployment which improved the performance of the teams in terms of changes frequency, failure management, and lead time. However, some still define DevOps as a combination of development and operations, but it is not about combining the teams, it's about a cultural change that leverages people over processes over tools and gets the teams to understand each other and work together to deliver value faster and better. In this paper, we are trying to put in evidence DevOps culture, and one of the best practices which is dependency injection.**

***Keywords*— *DevOps, Agile, People, Process, Tools, Dependency Injection***

## I. INTRODUCTION

Over the past decades, software has evolved into a critical component of the industry. A new era of programming languages, frameworks, software development tools, architectures, and technologies have been introduced such as Swift Programming Language, IoT (Internet of Things), Big Data, Cloud Computing, REST services, etc...

The software development process has gained in maturity as well, as it relied on the traditional project management process which was revealed to be less practical, riskier, and slower. We can identify different traditional linear models such as Waterfall Model, Spiral Model, V-Model, etc... In 2001 a new methodology was officially introduced called Agile, it made the software development life cycle not only faster but also more valuable. This new model made the transition from the traditional linear model to an iterative model. Many frameworks and methodologies were derived from this mindset such as Scrum, Lean, and Kanban. Agile addresses the gaps in Customer and Developer communications, but it is not sufficient to deliver a fully valuable product to the end-user. DevOps was introduced few years before to extend the continuous development goals of the agile methodology to continuous integration and continuous release by improving the communication between Development and Operations teams. To leverage the continuous release, DevOps relies on automation of change, configuration, and release processes. Automation takes us the A of CAMS which are the four values of DevOps, C for Culture which is an important component of this paper, A for Automation, M for measurement, and S for sharing. DevOps culture is not about the entertainment facilities integrated in companies and organizations, it's about the behavior. It exists among people with mutual understand of each other and where they're coming from. DevOps suggests also practices reinforcing the DevOps Culture, in this paper, we will cover some of those practices but in more details the Dependency Injection practice which is inspired from the popular software engineering design pattern "Dependency injection", or "Inversion of Control". The organization of this paper is as follows: Section 2 discusses Agile Concepts, DevOps concepts, and their relationship. Section 3 explains the DevOps culture which is mainly about the "People over Process over Tools" principle. Section 4 introduces the practice of Dependency Injection suggested by DevOps. Section 5 summarizes and concludes the paper.

## II. BACKGROUND

### A. *Agile*

Unlike, the traditional project management approach which relies on a linear model that runs generally through five stages: Requirements, Design, Implementation, Verification, and Maintenance. Each phase results in an approved work that is then used in the next phase. Agile is a new software development approach for planning and managing projects which relies mainly on teamwork, collaboration, timeboxing tasks, and the flexibility to respond to change as quickly as possible. This methodology follows an iterative process to create a valuable product for the customer. We can distinguish different frameworks extracted from the agile mindset such as Scrum, XP, Lean, Kanban, and Scrumban. Each has its own particularities, for example in scrum iterations are called "sprints", the goal behind each sprint is to create a valuable product, before each sprint a sprint planning meeting is organized to plan the sprint, during the sprint stand-up meetings of no more than 15 minutes

are done to assure the progression, at the end of each sprint a review meeting is organized to test the product, and finally the last meeting is the retrospective meeting in which the teams discuss the just-finished sprint. But they all rely on the Agile Manifesto which was created by a group of software engineering. The Manifesto includes the four values of agile which are:·

- **Individuals and Interactions** over Processes and Tools·
- **Working Software** over Good Documentation
- **Customer Collaboration** over Negotiating Contracts
- **Respond to Change** over Following a Plan

The first value of agile which is Individuals and Interactions over Processes and Tools introduces the main subject of this paper, as it means having a coherent team working together effectively is more important than Processes and Tools, but that does not mean that Processes and Tools are not important.

### *B. DevOps*

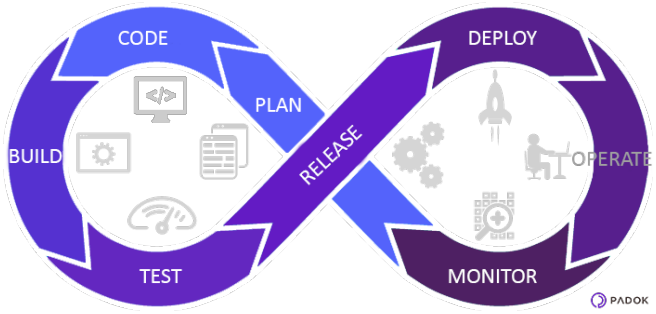


Figure 1: Shows DevOps Processes

DevOps is a combination of Dev which refers to Software Development and Ops that stands for IT Operations. The goal behind this combination is to stimulate a collaborative environment. The Development Team consists of developers, the front end, and Quality Assurance. They assure the development and testing of the software, as for the Operations team is made of the System administrator, Network Administrator, and Database Administrator, and takes care of the deployment and monitoring of the software. DevOps goes through eight different phases to deliver a valuable product:

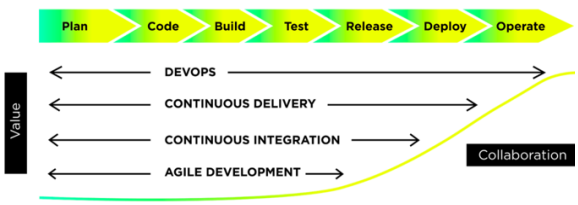


Figure 2: Shows the importance of DevOps

DevOps is situated in five levels:

- Values
- Principles
- Methods
- Practices
- Tools

DevOps values are summarized in the abbreviation "CAMS" which are Culture, Automation, Measurement, and Sharing. Culture is the interactions among groups and people and the capacity to understand each other, their goals, and responsibilities. This culture introduces the idea of "people over process over tools". Automation refers mainly to infrastructure as code which's the main purpose is to enable development or operations teams to automate the management, monitoring, and resources provisioning. Measurement insists on measuring the MTTP (Mean time to recover), Cycle time, cost, and revenue. Sharing is about collaboration and feedback.

DevOps principles are described as "The Three Ways" which are:

- The first way (Systems Thinking): Work always flows in one direction which means should done right the first time around.
- The second way (Amplify Feedback loop): Create, Shorten, and amplify feedback loops which encourages feedbacks to be given as early as possible.
- The third way (Culture of Continual Experimentation and Learning): Continued experimentation, in order to learn from mistakes, and achieve mastery.

Five methods are used in DevOps which are:

1. People Over Process Over Tools
2. Continuous Delivery
3. Lean Management
4. Visible ops-style change control
5. Infrastructure as Code

Many Practices have been found useful and practical to reinforce and improve the DevOps Culture are:

- Chaos Monkey
- Blue/Green Deployment
- Dependency Injection
- Andon Cords

- The Cloud Embedded Teams
- Blameless Postmortems
- Public Status Pages
- Developers on Call
- Incident Command System

DevOps Also relies on tools to assure the automations, the continuous integration, and continuous deployment. It is important to choose wisely the tools that compliment both each other and your strategy. There are different tools for each purpose such as "Chef" configuration management, "Jenkins" to support continuous integration, "Docker" for microservices, "Jira" for collaboration, "Ganglia" for Monitoring, and "Visual Studio" for development. But the choice of DevOps tools depends on many factors which are mainly the people and the strategy of the organization.

## III. PEOPLE OVER PEOCESS OVER TOOLS

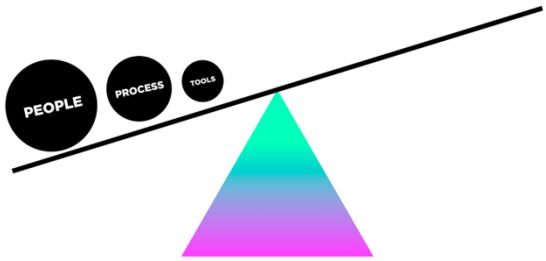


Figure 3: Shows the difference between people, process, and tools

DevOps is defined as a combination of people, processes, and tools that aims to continuously deliver value to customers or end-users. This combination follows a certain order according to their priorities, but that doesn't mean that any of them is not important. People come before process, and process comes before tools. What does that mean?

The worst common mistake is starting by looking for the right tools to fulfill the needs, It could cost the company a lot of time and money, as choosing, or developing a tool is an investment. Sometimes a specific tool is either expensive or inexistent, in that case, the company should decide to develop its own, but in case this tool doesn't fit the teams, that would cost the company either money or human resources.

People and their interactions are the most important, the DevOps culture focuses primarily on how people understand each other, and at what level they can collaborate and work together to continuously deliver, test, integrate, improve and deploy. Having conflicting goals, speaking different languages within a team, and moving fast and breaking things could only break the system. A mutual understanding is mandatory within a team in order to keep improving and moving forward. As Zetong Huang who is a Software engineer at ELMO said "The right DevOps culture requires communication and collaboration from both developers and Operations teams. To work effectively, with a shared vision, the team needs to be united and work together. As Sun Tzu says, "Harnessing the power of teams to achieve objectives is a wise use of energy.""

Processes have a big impact on breaking a system, they should be correct and assigned to the right person. Incorrect processes may not be well-documented, presented in the wrong order, or undocumented. Processes aim to achieve a continuous Integration, continuous Delivery, and continuous Deployment. Continuous Integration means integrating individual code with the overall development environment after building and testing. Continuous Integration which depends on the previous layer (Continuous Integration), as it involves building, testing, and improving the software code and user environments. Continuous Deployment consists of deploying code to production. All those processes could be done manually, or automatically. But to make them smoother, faster, and easier they should be automated using specific tools.

Tools might not be as important as people and processes, but they still are important in DevOps. Some processes are done manually which consumes much time and effort from the teams, that takes us to automation. Automation could be on different levels such as testing, and deployment. For that, the company should wisely choose suitable tools to support the automation while keeping the team productive and effective. They should fit the strategy, and also support each other.

To summarize, the first step is to look for the right persons to fulfill their roles, then make sure the process is correct and optimal, then finally choose the right tools to increase productivity and eliminate waste.

## IV. DEPENDENCY INJECTION

Dependency Injection also known as DI and inversion of control which is a technique used widely in software engineering. There are multiple ways to use an object inside a class, but some are less efficient than the others. The first common way is to instantiate the object inside the class, but this approach makes the class dependent on that object, so the class becomes limited in terms of reusability, flexibility, and extensibility. In other hand, another approach which is more practical and more advised, passing the object in runtime, some languages, or frameworks support the automatic object linking for example spring boot, the framework takes care of passing the object to the class on runtime, some languages or frameworks require a manual integration. This approach is called Dependency Injection.

Applications are composed of objects that collaborate and work together to fulfill the application's goal, instantiating for each class a different instance of an object is considered as a waste and complicated approach, this makes the test much difficult, and increases the dependencies in the code which is depreciated.

This applies on an organizational level; people also could be seen as dependencies. In the organizational level, Processes should not be dependent on a certain person within a team or another team, everyone should be aware of what the others do in case a person or a team is unreachable or absent when needed.

This also applies to software, the dependency between software should be minimized as much as possible, let's take an example of a company that restricts the use of a certain technology, while a developer uses that technology which also requires some other modifications in the infrastructure, that would create a compatibility issues and dependency problems. In this case, the advised approach is to use a technology that is compatible with all the cases possible.

## CONCLUSION

This paper aims to explain the idea called "People over process over tools", and how the dependency injection pattern completes the DevOps culture. DevOps is not only about using tools to create and maintain a product, its more than that, DevOps is a culture that should be applied implicitly while taking decisions, gathering teams, and working together to deliver a valuable product to the customer or the end-user. This philosophy invites us to consider certain values, principles, methods, practices, and tools.

This paper gives an overview over DevOps in general while focusing on the principle "People over Process over Tools" and the practice of dependency injection inside the teams.

## REFERENCES


[1] Mandi Walls, "Building a DevOps Culture", 2013

[2] Patrick Debois, "Just Enough Developed Infrastructure", 2017

[3] Sam Guckenheimer, "What is DevOps Culture?", 2017

[4] Alex Honor, "People over Process over Tools", 2010

[5] Ernest Mueller, and James Wickett, "Devops Foundations", 2020

[6] Sumit Singh, "DevOps Best Practices ", 2020

[7] Nick Kartman, "How to Implement DevOps with CAMS", 2019

[8] Aymeric Hemon-Hildgen, Barbara Lyonnet, Frantz Rowe, and Brian Fitzgerald, "From Agile to DevOps: Smart Skills and Collaborations", 2020

[9] Daniel Ståhl, Torvald Mårtensson, Jan Bosch, "Continuous practices and devops: beyond the buzz, what does it all mean?", 2017

[10] Clemance Plu, "Understanding the DevOps Process", 2019