How

did we arrive here?

Agile, Kanban, Lean, DevOps and now **DevSecOps**, it seems that we have not finished adopting a philosophy when another one is already underway. And it is actually so. All of these methodologies, movements and cultures, respond to business needs and their adoption is accelerated or not based on how pressing they are in the needs of an organization.

Initially, IT departments supported and streamlined business processes. They were considered internally as one more expense within the big organizations (some have even qualified them as necessary evils). Within the IT departments each had a well defined, specfic and perfectly delimited responsibility.

As large organizations opened new channels of business through the new channels, IT departments simply stopped supporting business processes to bring a differential value that would boost business. The time-to-market of the new functionalities became critical and the non-availabilities of IT services did not simply lead to low productivity but rather produced direct losses in the core business (for example, an online sale) . Business analysts continued to mark requirements that were interpreted by development analysts and the availability time of new functionality, far from shortened, was lengthened (increased time-to-market).

In order to provide solutions to these problems, agile methodologies were precisely designed. Business analysts were no longer external agents to development and became part of these teams through the figure of the product owner. This was not the only change, development teams were transformed into multidisciplinary cells that, along with business, were able to launch a new functionality independently and autonomously.

The product became (and I hope it continues to be) at the center on which everything revolved. The user was present, through the product owner, to express the needs and perform the validation of the functionalities, the development teams enjoyed the technical autonomy necessary to implement all the requirements. Quality was not debatable in the process, it was present and inherent in it. You

could not be agile if you didn't develop with quality. The whole process ended with the deployment (or not) in production because the purpose of the methodology was to produce versions that could be deployed.

But the process had forgotten a fundamental part of the functionalities: its operation in production. That's why DevOps was born, which introduces the operations and monitoring teams into the development process. At this time the applications were not only developed in an agile way and according to the business needs, but also they were deployed and operated in accordance with the needs of production. The processes are further accelerated thanks to automation and, based on technology, we have moved from continuous integration to continuous deployment.

By opening up the possibility of end-users interacting directly with business actions without an intermediary, system vulnerabilities are being exposed directly, therefore:

- Users, without previous training, can take actions on the systems and affect their functionality

- Security vulnerabilities are open to a much wider range of people who may discover leaks and perform security attacks.

With all these needs the development of secure systems becomes essential nowadays.

Resistance

from teams against DevSecOps

So by adopting DevSecOps we have the problem solved, right? In fact, as we pointed out earlier, DevSecOps is a cultural movement. This movement has to be landed in a practical way in the organizations taking into account the peculiarities of each of them.

We must remember that organizations, large or small, are made up of people and it is precisely those who are more resistant to cultural change.

The most common drawbacks when implementing DevSecOps in organizations are the following.

Security cultura & Knowledge

Knowledge of security vulnerabilities and ways to remedy it is concentrated in specialized teams in organizations that (historically) have not had much contact with the development side but they have with systems. Development teams are more focused on translating functional needs and are more aware (to date) of the requirements that go into performance issues. While on the one hand it is an advantage, these organizations would be closer to SecOps than to DevSecOps.

It is necessary to ensure that the knowledge in application security and, above all, how to develop those functionalities in a safe way is transferred from the security team to the development teams. All this without forgetting that every day new vulnerabilities are discovered as well as new ways of avoiding them.

Several teams, several tools

The culture that has tended to be imposed in the organizations has been departmental. In it, each department had autonomy, authority and responsibility for a particular task. Within their autonomy they selected both the procedures and the tools to be used. The communication between departments was carried out through reports which diminished in an exorbitant way both the productivity and the communication.