Fundamentals and Benefits

of CI/CD to Achieve, Build and Deploy Automation

Fundamentals

CI/CD

What is it?

A set of development practices aimed at automating the building, testing and deployment of applications

Automates much or all the manual processes traditionally required to get new code from a commit into production. A CI/CD pipeline encompasses the source, build, test and production stages. In each stage, the CI/CD pipelines provisions any infrastructure that is needed to deploy or test the code.

What are the costs and benefits?

Let’s see

Costs

The quality of the test suite will determine the quality of the releases, so the testing culture must be at it’s best,

We’ll need to keep up the documentation process with the pace of the deployments

The installation and maintenance of a CI server, but it can be significantly reduced if we use a cloud service like Bitbucket repository

Benefits

Develop faster: no need to pause development for new releases. Deployment pipelines are triggered automatically for every change.

Less developer time on issues from new code.
Less time to market.
Less time in testing.
Reduced downtime from a deploy-related crash or mayor bug.
Faster deployments.
Faster recovery from botched integrations

Releases are less risky and easier to fix in case of problem as we'll deploy small batches of changes.

Less bugs in production.
Less human error.
Version control everything.
Greater transparency and communication.
Singular, unified, consistent build process.
Merging all developers working copies to a shared mainline.
Automatically packaged and downloadable binaries.

Customers see a continuous stream of improvements and quality increases every day instead of every moth, quarter or year

Every change that passes all stages of production pipeline is released to the customer.
Accelerate the feedback loop with our customers.
Take pressure off the team.
Developers can focus on building software.

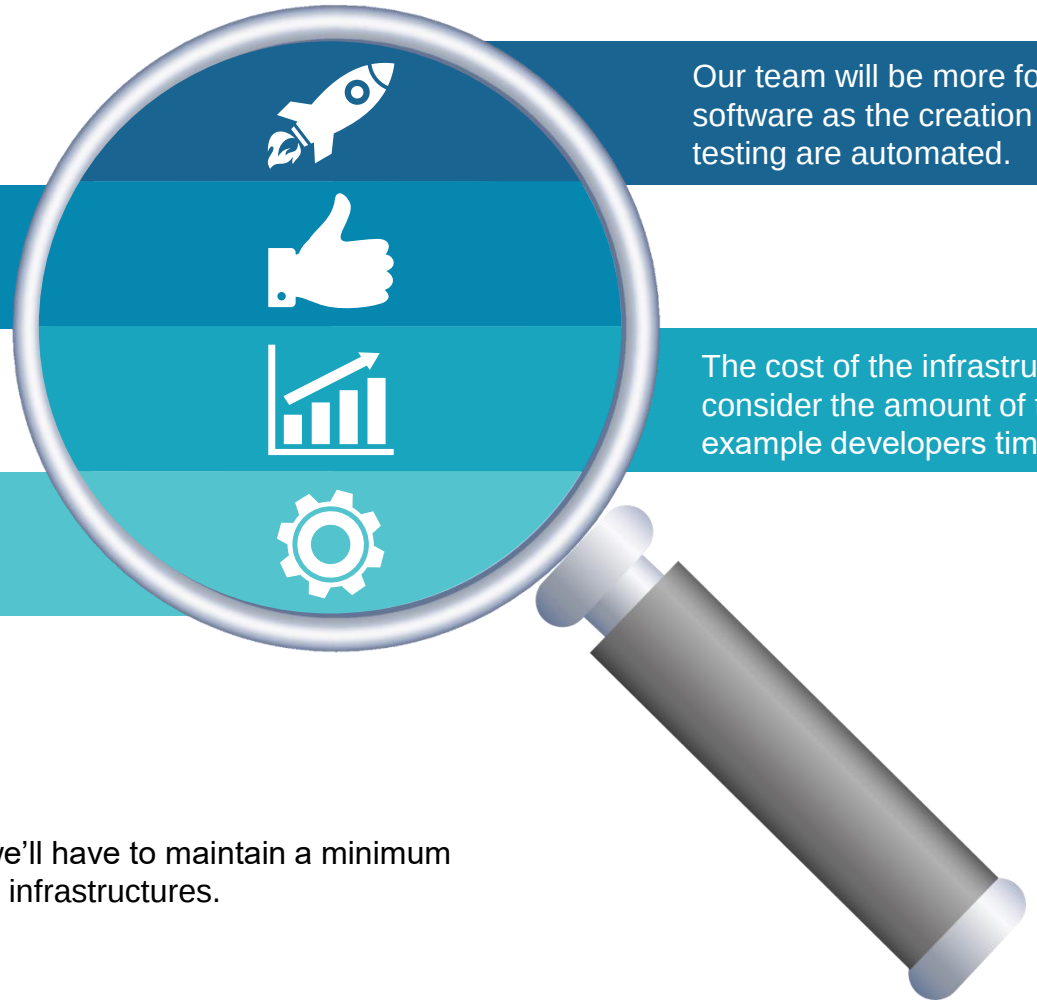
Conclusion

Our team will be more focused on developing software as the creation of infrastructure and testing are automated.

Customer will perceive the improvements faster, which will translate in a faster feedback.

The cost of the infrastructure are minimal if we consider the amount of time and resources (for example developers time) we'll save.

The maintenance and constant deployment can be taken care by just adding a configuration file at the rot of the repo.



In summary

Everything will be simpler, faster and less risky, but we'll have to maintain a minimum level of quality on our testing and maintenance of the infrastructures.


THANK YOU