LIMEPOINT

DevOps and Continuous Delivery for Oracle

Industry Publication

DEVOPS AND CONTINUOUS DELIVERY FOR ORACLE

By Goran Stankovski

Operations of the delivered capability often is not on the priority list of project teams. The project teams objective is to deliver the solution in the timelines communicated by the project management team. Timelines that are often very tight, and usually defined with little technical input and/or made with large assumptions, that remain invalidated until so late in the life of the project that remediation is impractical, if not impossible.

This puts enormous pressure on project teams to deliver and transition the system to Operations for the on-going management. Meanwhile Operations are often left with an un-manageable solution, with which they have had little to no involvement. From the Operations perspective they've been thrown a "ticking time bomb" in that while it delivers an essential capability to the business, Operations cannot effectively manage, modify, remediate, or enhance it.

As a side effect in a traditional enterprise delivery model, Operations teams are incentivised to dissuade and even actively impede delivery of new, improved or streamlined business function.

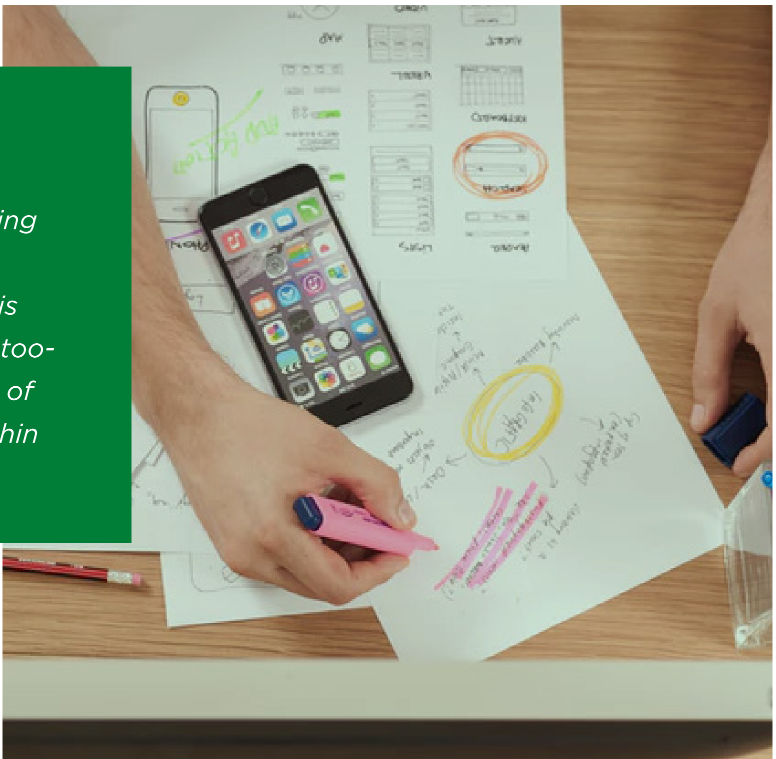
After all, completely remaking "the world" will always carry significant operational and procedural risks associated with any system that is untried and untested in the production environment. Additionally as any new system or capability can often represent a year or more of change, it is virtually impossible to roll back if things go wrong – if not technically, often politically.

DevOps (as the name suggests) is the union of Development and Operations teams, typically demonstrated through their increased collaborationand cooperation.

Implementing DevOps and Continuous Delivery methodologies in a traditional enterprise environment has many challenges.

One such challenge is that DevOps can mean many different things to different people - there is no DevOps governing or certifying body, nor is there one clear and concise definition. In many respects DevOps is about attitudes and relationships, more than tooling and procedures - it is the bringing closer of Development and Operations capabilities within an organisation.

> “
> *DevOps can mean many different things to different people - there is no DevOps governing or certifying body, nor is there one clear and concise definition. In many respects DevOps is about attitudes and relationships, more than too-ling and procedures - it is the bringing closer of Development and Operations capabilities within an organisation'.*



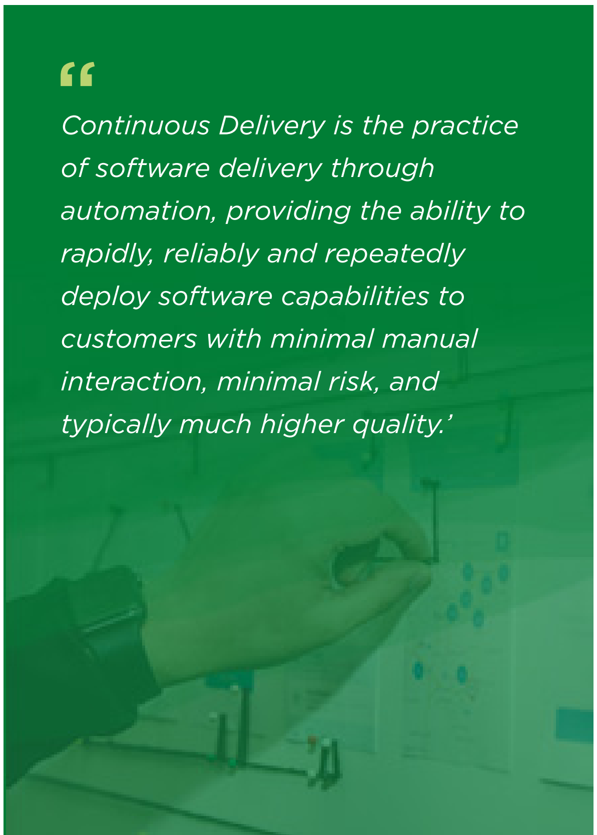
One thing is clear; the greater the cohesion of Development and Operations, the greater the business value of delivered solutions and capabilities.

Common objectives of DevOps include delivering capabilities with a faster time to market (or speed-to-value), reducing the failure rate of new releases, minimising the lead time between fixes, improving system quality, and achieving a faster mean time to recovery in the event of a new releas impacting the current system. Continuous Delivery is the method by which such objectives can be met.

Continuous Delivery is the practice of software delivery through automation, providing the ability to rapidly, reliably and repeatedly deploy software capabilities to customers with minimal manual interaction, minimal risk, and typically much higher quality.

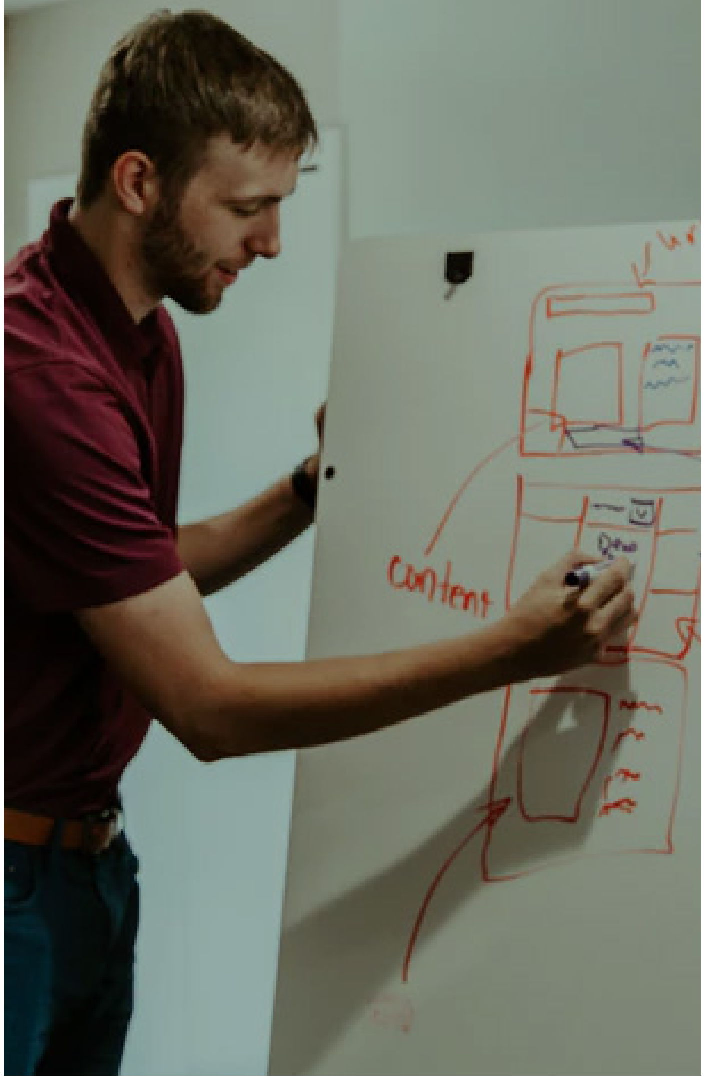
With Continuous Delivery, we continuously deliver small, understandable and reversible changes over time. Of course there is an overhead to this– you can no longer delay all testing until the end of the project; you are now doing a deployment every week (even every day) instead of every year. Therefore automated testing and deployment systems tend to be inextricably linked to these methodologies.

There are a number of key DevOps and Continuous Delivery principles that we believe are imperative in ensuring that the benefits of DevOps and Continuous Delivery can be realised.



“

Continuous Delivery is the practice of software delivery through automation, providing the ability to rapidly, reliably and repeatedly deploy software capabilities to customers with minimal manual interaction, minimal risk, and typically much higher quality.’



EMBRACE **CHANGE**

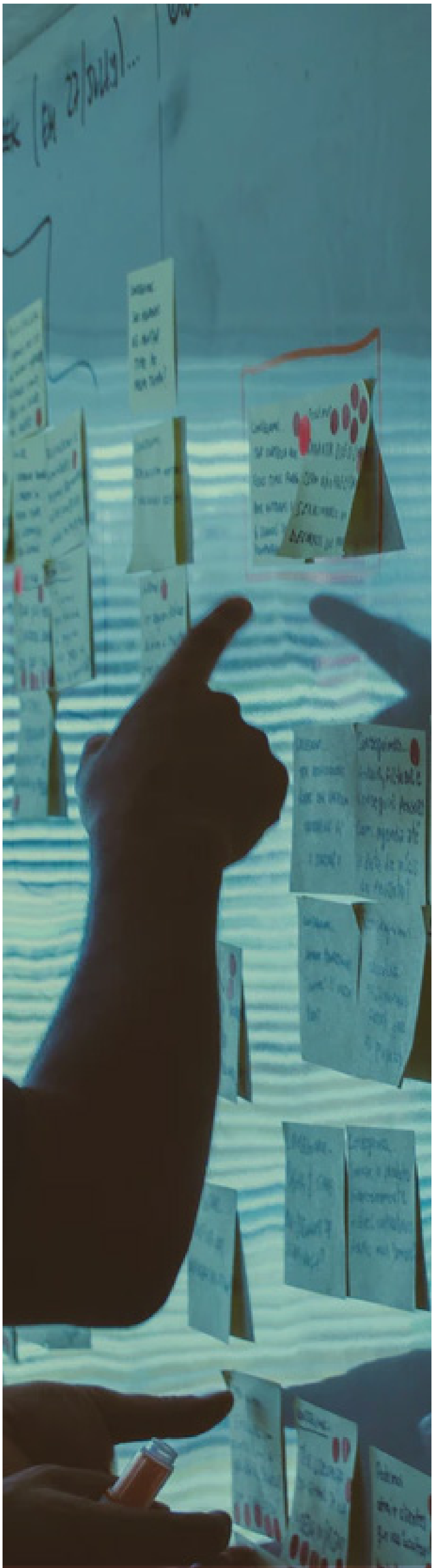
Change is good. Delivering capability to the business increases business value. Being able to do this faster and better than your competition provides a business ifferentiator that can be used to capitalise on opportunity.

AUTOMATE **EVERYTHING!**

Automation is a foundational building block in DevOps and Continuous Delivery, providing a repeatable, reliable, and higher quality outcome. The more manual interaction you have in your system development and operation, the slower you will be able to adapt to change.

DO NOT AVOID THE **“TOO HARD” BASKET**

If something is “difficult” or “too hard”, then do it more often – it will become easier over time. If something is hard to automate, this only increases the necessity to automate it.



KEEP EVERYTHING IN **SOURCE CONTROL**

Adopting an “Infrastructure-as-code” approach will allow you to define your infrastructure and platform through code constructs and deliver it through automation. Storing your entire platform and application configuration in source control systems, such as GIT, will allow you to deliver consistent platforms and applications to support your business initiatives. Large numbers of small changes are superior to small numbers of large changes. Smaller, digestible changes are easier managed, and easily rolled back.

ADOPT THE **CONTINUOUS** IMPROVEMENT APPROACH

Improving the Continuous Delivery and Automation capability over time will allow you to incorporate feedback into the process to deliver prolonged value. So what about DevOps and Continuous Delivery in an oracle context? Oracle is considered an “Enterprise” class software. In Continuous Delivery and DevOps circles, enterprise software is viewed by many to be diametrically opposed.

To its key principles and tenets. It is inflexible, “hard to work with”, delivered on a long software release cycle, and as a result is slow to catch up with industry trends and emerging standards.

This misalignment in worldview tends to have many DevOps and Continuous Delivery initiatives place enterprise software, such a Oracle, in the “too hard” basket and leave it for a later project to worry about.

Many enterprises and businesses, however, rely on such enterprise software for a majority of their business capability, and as a result, the business value of DevOps or Continuous Delivery initiatives delivered excluding such enterprise software is significantly diminished.

Adopting Continuous Delivery and DevOps within your enterprise requires a fundamental change to your technical project delivery capability, which will require maturity to progress it within the organisation.

In our opinion, DevOps and Continuous Delivery capability maturity can be viewed in the following three distinct phases:

1.

Destructible Development **Enviroments**

The foundational capability in DevOps and Continuous Delivery initiatives is the ability to deliver destructible development environments. How do you deliver consistent environments through a repeatable and predictable process? How do you deliver this across Oracle Infrastructure, Virtualisation, Database, Middleware, and Applications? This will be the topic of our next article in the series.

2.

DevOps Lifecycle **Management**

As your maturity progresses, the next step in the evolution is bringing lifecycle and release management capability through your DevOps and Continuous Delivery framework. How do you adopt DevOps principles in your organisation's Dev/Test lifecycle? What key considerations must youtake into account in order to deliver this capability? What does this mean with respect to Oracle Infrastructure, Virtualisation,Database, Middleware, and Applications technologies? Subsequent articles in the series will focus on this aspect.

3.

DevOps and BAU

The final phase in the DevOps maturity is brin-ging the capability into business as usual (BAU). Considerations such as configuration change management and drift detection are critical in this phase. This is the most difficult phase to adapt, as there is typically a huge separation and disconnect between the Development and Operations BAU teams in many organisations. Most of these issues are typically organisational and not technical in nature. Trust plays a key part. How can BAU trust what is delivered? How can we bridge this divide?

In forthcoming articles, we will focus on each of these three phases and share our experience and recommendations.

“

Adopting Continuous Delivery and DevOps within your enterprise requires a fundamental change to your technical project delivery capability, which will require maturit to progress it within the organisation. ‘

ABOUT THE AUTHOR

Goran Stankovski is a Director and Co-founder of LimePoint. He's a recognised thought leader in Enterprise and Technical Architecture, Technology Enablement, Business Process Modeling, SOA Governance, SOA Implementation Methodology, Identity and Access Management, and has presented at Oracle Open World. This article was first published in Oracle's O-Tech Magazine in Winter 2014.

CONTACT US

www.limepoint.com

info@limepoint.com

LIMEPOINT