DevOps Tales from the Front-Line
Who am I?

Originally from the USA

Background in Software Development

Moved to NZ in 2010

With ClearPoint for 4 years

Chief of Product and Platform
Who Are We?

New Zealand-owned digital experience & engineering company
Founded by Phil Pietersen and Bain Hollister in 2007 in Auckland, NZ

ClearPoint is engineering focused. We like to tackle big, enterprise-heavy engineering problems.

DevOps Culture
Our culture adheres to DevOps principles and practices and we help organisations transform their DevOps capabilities.
Dev-Ops

“A technology transformation that drives value to organizations through an ability to deliver code with both speed and stability. Its essential components are technology, process, and culture.”

-Nicole Forsgren
I expect a great experience

I expect flawless performance

I expect direct & intelligent channels

I expect you to listen to my feedback
Architecture - modern, decoupled (e.g. Micro-services)

Feature toggling and trunk based development

Test automation against production-like datasets

Elastic, dynamic infrastructure, elastic build infrastructure

Immutable Infrastructure & Containerisation, Container Management

Infrastructure as code with full automation, including the pipeline itself
CULTURE
The Hard Stuff
Fear of Failure

Fear of Exposure

Fear of Redundancy

Lack of Change Management

Exhaustion & Cynicism

Predisposition
Creating a Fear Friendly Culture

When we fail, we recover quickly and learn from our failures. We do not create a culture of avoidance.

We trust our employees and we expect them to fail.

We create a culture of innovation.

We value transparency.

We value a culture of rapid recovery over prevention.

We learn from past mistakes.

Set approaching goals, not avoidance goals.

“Failure is a part of [the] process. You just learn to pick yourself up. And the quicker and more resilient you become, the better you are.”—Michelle Obama
Change Management Strategies

Assess the Change
- What will the specific changes include?
- Who will the changes impact?
- How will the changes impact them?
- Why would people resist the changes?

Manage Communications
- Build a community based culture
- Be transparent and open
- Set an example
- Be comfortable changing your plans
- Be proactive

Understand what you are getting into…

“We need to create a culture that reinforces the value of taking risks and learning from failure and the need for repetition and practice to create mastery.” – Gene Kim
Exhaustion, Cynicism and Predisposition

Too much change can be a bad thing.

Show results regularly.

Communicate success as well as failure.

Be inclusive.

Get staff alignment.

Make data-driven decisions.

Encourage self-organising teams.

Acknowledge that some team members may have a disposition against change.

“Scratch any cynic and you will find a disappointed idealist”
– George Carlin
PROCESS
Finding the right balance
Process

1. Principles
2. Too Much Process
3. Too Little Process
4. Team Agility
5. Organisational Agility
6. Lean
Principles

• Uncertainty is inherent and inevitable in software development processes and products. (**Ziv’s Uncertainty Principle**) 

• For a new software system, the requirements will not be completely known until after users have used it. (**Humphries Requirements Principle**) 

• It is not humanly possible to completely specify an interactive system. (**Wegner’s Principle**)
Agile Principles

In February 2001 a group of software strategists and pundits got together in the US to address growing concerns around Waterfall delivery. Out of these discussions came the Agile Manifesto.

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- We welcome changing requirements, even late in development.
- We deliver working software frequently.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals.
- The most efficient and effective method of conveying information is face to face conversation.
- Working software is the primary measure of success.
- Agile processes promote sustainable development and consistent long-term pace.
- Continual attention to technical excellence and good design enhances agility.
- Simplicity – the art of maximising the amount of work not done – is essential.
- The best architecture, design and requirements emerge from self organising teams.
- At regular intervals the team reflects on how to become more effective and then tunes and adjusts its behavior accordingly.
Agile Principles

In February 2001 a group of software strategists and pundits got together in the US to address growing concerns around Waterfall delivery. Out of these discussions came the Agile Manifesto.
Agile Principles

In February 2001 a group of software strategists and pundits got together in the US to address growing concerns around Waterfall delivery. Out of these discussions came the Agile Manifesto.
Agile Principles

In February 2001 a group of software strategists and pundits got together in the US to address growing concerns around Waterfall delivery. Out of these discussions came the Agile Manifesto.

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- We welcome changing requirements, even late in development.
- We deliver working software frequently.
- **Business people and developers must work together daily throughout the project**
- Build projects around motivated individuals.
- The most efficient and effective method of conveying information is face to face conversation.
- Working software is the primary measure of success.
- Agile processes promote sustainable development and consistent long-term pace.
- Continual attention to technical excellence and good design enhances agility.
- Simplicity – the art of maximising the amount of work not done – is essential.
- The best architecture, design and requirements emerge from self organising teams.
- At regular intervals the team reflects on how to become more effective and then tunes and adjusts its behavior accordingly.
Scaling Agility

• Agile alone, while very effective as an SDM, does not scale well at large organisations

• Scaled Agile methodologies were created to address this challenge.

• Many implementations of Scaled Agile are not very successful:
  o The Spotify Model is a product development model.
  o SAFe – easy to bake in non-agile processes
  o Agile alone will not change culture
  o Lack of understanding of principles

"My advice is to stop emphasizing the Agile Transformation process frameworks and start focusing on the company culture and mindsets" - Selena Delesie
Too Much Process vs. Too Little Process

“Be light on process” does not mean “have no process.”

Some process is needed to scale.

Process is needed in mission critical activities.

Avoid Proxies.

Processes should never be used to enforce business rules that do not result in customer satisfaction.

Processes should not be created to support org structures or positions.

Too much process is dangerous for innovation.

“Everyone has a plan until they get punched in the face” – Mike Tyson
“Traditional approach is prevention is better than recovery. Mostly though, rapid recovery is the right model. We’re in a creative-inventive market, not a safety-critical market, like medicine or nuclear power. You may have heard preventing error is cheaper than fixing it. Yes in manufacturing or medicine, but not so in creative environments. Good process helps talented people get more done. Bad process tries to prevent easily recoverable mistakes. **Rule Creep** – Bad process tends to creep in (preventing errors just sounds good) we try to get rid of rules when we can, to reinforce the point.”
Lean

- Originates in manufacturing at Toyota, with W.E. Deming in post-WW2 Japan.
- Clear Focus on Quality and Continual Improvement which aligns well with Agile and Dev Ops Culture.
- There is a difference between Lean Manufacturing and Lean Product Development.
- There is a difference between deployment and release
  - Deployment is IT, software related
  - Releasing is business – customer related

“Agile is about deciding how to build things right. Lean is about deciding the right things to build” – Sven Kräuter
**Lead Time:** The latency between the initiation and execution of a process.

**Cycle Time:** The total time from the beginning to the end of your process, as defined by you and your customer. Cycle time includes process time and delay time.

**Value Added Time:** The time spent that improves the outcome of a process. This is typically just the processing time associated with production.

VAT is used to identify non-value added activities and eliminate them from a process, so that the total time required to complete a process is reduced.
Enabling High Trust

High deployment frequency of new features and enhancements via CD.

Low overhead and lead time for lead up to releases via CI.

Low support overheads via containerisation, test automation, CD and CI.

Low change failure rate via test automation.

High scaleability via microservices and automation.
Technology is an enabler
Technology Enablers

1. Decoupled Architecture
2. Immutable Infrastructure
3. Infrastructure Automation
4. Test Automation
5. Deployment Automation
6. Observability
Decoupled Architecture

Pre SOA (monolithic)
Tight coupling

Traditional SOA
Looser coupling

Microservices
Decoupled
Immutable Infrastructure

Once you instantiate something, you never change it. Instead, you replace it with another instance to make changes or ensure proper behavior.

Benefits

• Simplifying operations.
• Continuous deployments, fewer failures.
• Reduces errors and threats.
• Easy recovery
Infrastructure, Test, & Deployment Automation

AUTOMATED ACCEPTANCE TESTS
- E2E ACCEPTANCE TESTS
- KEY BUSINESS PROCESSES COVERED
- MANUAL EXPLORATORY TESTING ONLY

AUTOMATED INFRASTRUCTURE
- INFRASTRUCTURE AS CODE
- ENVIRONMENT CONSISTENCY
- NO MANUAL ACCESS TO INFRASTRUCTURE
- ALL CHANGES GO THROUGH PIPELINE

AUTOMATED DEPLOYMENTS
- FULLY AUTOMATED BLUE/GREEN DEPLOYS
- DATABASE SCHEMA MIGRATION AS CODE
- FEATURE TOGGLES
Observability is a measure of how well internal states of a system can be inferred from knowledge of its external outputs.

Monitoring is an action, Observability is a property of a system.

If your systems and applications don’t adequately externalize their state, then even the best monitoring can fall short.
Summary

- The essential components of Dev-Ops are **technology, process, and culture**. Focusing on just one component will not bring success to your Enterprise. It’s a holistic, cultural, mindset.

- **Culture is the hardest thing to change.** Each culture is different and there is no “out of the box” solution. A change management plan is essential for success.

- Agility is comprised of a set of principles that need to be understood and accepted at a Leadership Level. Controls can be put in place, that still align with Agility. **Agility does not mean YOLO.**

- Agility is designed to make things quickly, safely and with high quality.

- Lean Product Development is designed to ensure that you are making the right things. Separate releases from deployments. **Be Lean in your releases.**

- **Technology is an enabler.** All of the above needs to be in place to use it. There is no point in buying a Tesla if you do not now how to drive and don’t know where you are driving.
Digital Transformation & Acceleration

- Culture
- Process
- Technology
Remember…

“It is not the strongest of the species that survive, nor the most intelligent, but the one most responsive to change.”

- Charles Darwin
Thank you!

gorge.putnam@clearpoint.co.nz | putnam.george@gmail.com

https://www.linkedin.com/in/george-putnam-6323914/