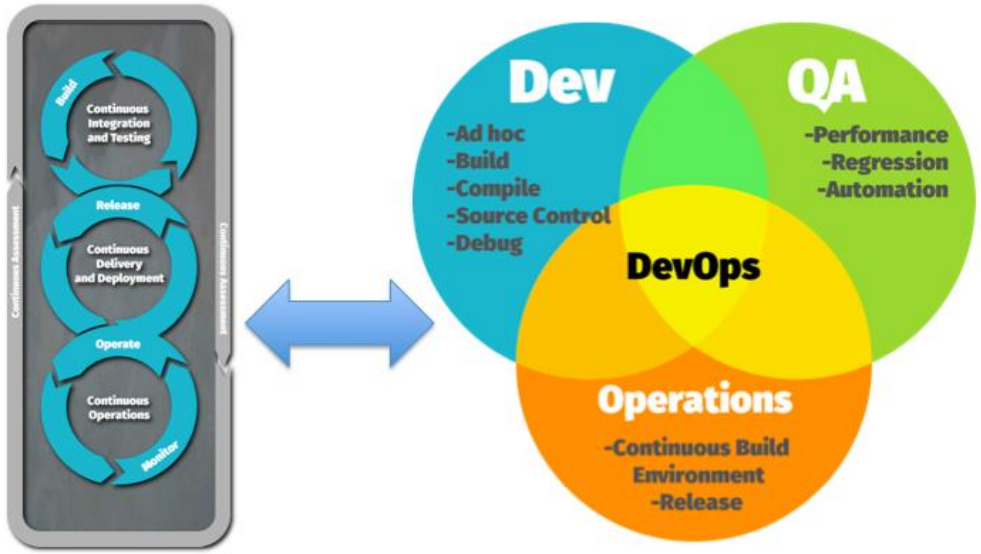
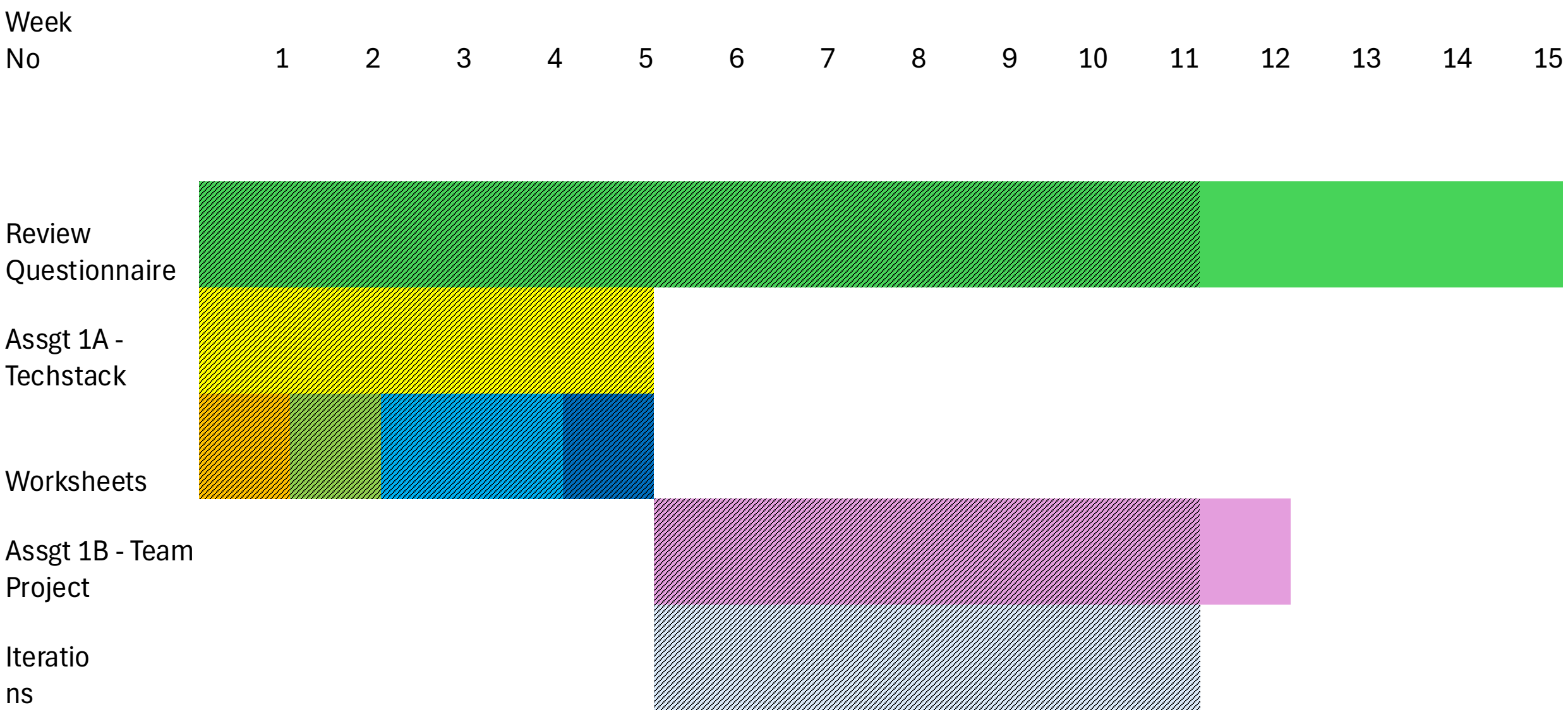
Week 12
Course Review
Empirical Software Engineering
The DevOps Way of Working



Taking Stock



Briefly Reviewing the Course

- Check the course schedule and modules for topics and activities traversed
- See the next slide for assessment
- Review the example test (under assessments on Canvas) to get some sense of the style of the final test (although topics have evolved since then)
- Review Jim's mind map – but technology elements based on the earlier MERN platform used - and consider the roadmaps for the current tech stacks and elements – two slides on
- Review the key events in the development process and practices adopted
- Consider the insights from your collaborative software engineering experience and lectures and digest them for the final test

Assignments Drive your Learning

Ass 1A preparing for Software Development (20%)
(*Individual*)

- Set up the tools an individual needs to support coding, good code craft, version control and unit testing
- Set up the tools needed to collaborate with a team to achieve product goals together

Sharing code – integrate code, review code,
Setup the tools needed to work with the selected Tech Stack (front-end/backend)
Set up tools to assure quality of product
Set up tools to deploy the product to the cloud
Set up tools to monitor and alert issues post deploy
Learn how to use the tools
Learn how to use the Tech Stack
Understand the product goals -> Product Backlog
Sprint 1 Goals -> Sprint Backlog

Submission in Tutorials weeks 1-5 (sign off by TA)
Evidence portfolio and demo

Ass1B Full SDLC full stack product Dev (50%)
(*small team - 4 Including QA*)
Capability building by Developing a Product in a small team
Apply a new tech stack and tool set
Practice DevOps and Scrum WoW
Collaborate with a Product Owner and team
Three sprints to learn fast – fast feedback
Submit – reviews weeks 7,9,11 (tutorials)
Capability and learning Portfolio with Evidence
Product increments

Sprint 1 weeks 5 and 6
Sprint 2 weeks 7 and 8
Sprint 3 weeks 9 and 10

Ass 2 Knowledge Check (30%)
(*Individual, online questions*)
A set of questions about scenarios to confirm you have understood main language and principles
Sometime in Revision weeks (Faculty schedules)

Roadmap Resources - Topics

Skill Based

https://roadmap.sh/react

https://roadmap.sh/javascript

https://roadmap.sh/typescript

Role Based

https://roadmap.sh/frontend

https://roadmap.sh/backend

e.g. Architectural Patterns - 12 Factor Apps

https://www.youtube.com/watch?v=FryJt0Tbt9Q

roadmap.sh is a community effort to create roadmaps, guides and other educational content to help guide the developers in picking up the path and guide their learnings.

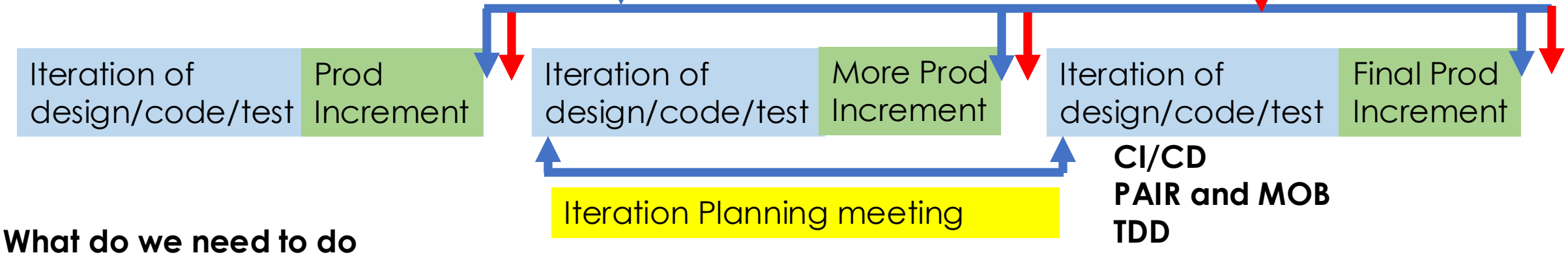
https://roadmap.sh/

How will we get feedback on product from users/client?

Regular review of product increment

How will we keep improving our process

Regular review of team process



CI/CD
PAIR and MOB
TDD

What do we need to do before we start coding?

How will we decide what is in each Iteration?

-Initial product backlog and story map (some uncertainty)
-User stories with Acceptance criteria
-Detail understanding and design of features for next iteration only
-Architecture and tech stack and deployment
-Dev Environment set up
-Plan for iteration 1
  Goal and Iteration Backlog

Regular team meeting during iteration…
Is there anything stopping us from reaching the goal?

How will we coordinate work with each other and keep on the same page?

How will we manage changes to requirements?

How will we manage risks?

How will we assure quality?

Some Empirical Research in SERL: What works under what circumstances and why?

When is it better to use mob, pair or solo programming?

How can technical debt be measured?

What are the expectations of the PO?

In an Agile team, how does shared understanding of requirements occur?

What are the benefits and challenges of planning poker for estimation? What can be changed to improve the benefits?

How is a new team member onboarded and how can this be improved?

Project or feature triage – how to feed the agile delivery engine?

How are requirements changes managed in globally distributed teams?

What can we learn from failed projects such as Novapay? Dilemma analysis.

Organizational structure and Agile – managing organizational interfaces with agile teams

Which code should be refactored and how? Semi-automated refactoring tool.

Behaviour patterns of developers from mining software repositories?

How can programming/SE/SRE/Testing be taught/learned more effectively?

How is DevOps implemented and what are the benefits and challenges?

What potential requirements are embedded in user reviews? Machine learning for text processing.

Measuring testability with dynamic metrics?

What are the emerging roles in an agile team and how is work divided among them?

What are the benefits and challenges of implementing TDD and why?

How can coordination of distributed teams be improved?

Requirements tracing for cyber-physical systems

How can team "health" be monitored and diagnosed?

What are the success factors post adoption of Agile?

Some Empirical Research in SERL: What works under what circumstances and why?

When is it better to use mob, pair or solo programming?

What are the expectations of the PO?

an technical debt asured?

What are the benefits and challenges of planning poker for estimation? What can be changed to improve the benefits?

How is a new team member onboarded and how can this be improved? Interview Survey

When is best to do mob, pair or solo programming? Interview and Observation.

Organizational structure and Agile – managing organizational interfaces with agile teams

bred and oring tool.

Behaviour patterns of developers from mining software repositories?

How can programming/SE/SRE/Testing be taught/learned more effectively?

How is DevOps implemented and what are the benefits and challenges?

What potential requirements are embedded in user reviews? Machine learning and text processing.

nation of ns be impr

cyber-physical systems

How can team "health" be monitored and diagnosed?

How is DevOps implemented and what are the benefits and challenges in practice? Interview-based Case Studies

The plan…

There are a lot of claims about the benefits of DevOps and descriptions of how to implement it, but very little empirical evidence?

What are the benefits, enablers and challenges in implementing DevOps ***in practice***?

The plan…

There are a lot of claims about the benefits of DevOps and descriptions of how to implement it?

What are the benefits, enablers and challenges in implementing DevOps in practice?

Interviews survey

Multiple organisations

Different stages of

Views of different roles

Deep insights in real

Content Analysis of Transcripts

The plan…

What are the enablers and challenges in implementing DevOps in practice?

Interviews survey

Multiple organisations

Different stages of

Views of different roles

Dev (one of 3 back-end, 2 front-end, two floaters), Tester, Ops,
QA release manager, Tech Lead Infrastructure, Training manager

Medium Sized SaaS Product company. Fast growth.

Agile way of working (scrum)
2 years into the DevOps journey

Examples of titles of other research papers…

Relationship of DevOps to Agile, Lean and Continuous Deployment
A Multivocal Literature Review Study

Lucy Ellen Lwakatare(✉), Pasi Kuvaja, and Markku Oivo

Faculty of Information Technology and Electrical Engineering,
University of Oulu, Oulu, Finland
{lucy.lwakatare,pasi.kuvaja,markku.oivo}@oulu.fi

Abstract. In recent years, the DevOps phenomenon has attaracted interest amongst practitioners and researchers in software engineering, reflecting the greater emphasis on collaboration between development and IT operations. However, despite this growing interest, DevOps is often conflated with agile and continuous deployment approaches of software development. This study compares DevOps with agile, lean and continuous deployment approaches in software development from four perspectives: origin, adoption, implementation and goals. The study also reports on the claimed effects and on the metrics of DevOps used to asses those effects. The research is based on an interpretative analysis of qualitative data from documents describing DevOps and practitioner's responses in a DevOps workshop. Our findings indicate that the DevOps phenomenon originated from continuous deployment as an evolution of agile software development, informed by a lean principles background. It was also concluded that successful adoption of DevOps requires agile software development.

Literature Review: Promises and Challenges of DevOps

Shubhangi Nagpal
University of Waterloo
s6nagpal@uwaterloo.ca

Anam Shadab
University of Waterloo
a2shadab@uwaterloo.ca

DevOps Adoption Benefits and Challenges in Practice: A Case Study

Leah Riungu-Kalliosaari[1(✉)], Simo Mäkinen[1], Lucy Ellen Lwakatare[2], Juha Tiihonen[1], and Tomi Männistö[1]

[1] Department of Computer Science, University of Helsinki,
Gustaf Hällströmin katu 2b, P.O. Box 68, 00014 Helsinki, Finland
riungu@cs.helsinki.fi
[2] Department of Information Processing Science,
University of Oulu, P.O. Box 3000, 90014 Oulu, Finland

Abstract. DevOps is an approach in which traditional software engineering roles are merged and communication is enhanced to improve the production release frequency and maintain software quality. There seem to be benefits in adopting DevOps but practical industry experiences have seldom been reported. We conducted a qualitative multiple-case study and interviewed the representatives of three software development organizations in Finland. The responses indicate that with DevOps, practitioners can increase the frequency of releases and improve test automation practices. DevOps was seen to encourage collaboration between departments which boosts communication and employee welfare. Continuous releases enable a more experimental approach and rapid feedback collection. The challenges include communication structures that hinder cross-department collaboration and having to address the cultural shift. Dissimilar development and production environments were mentioned as some of the technical barriers. DevOps might not also be suitable for all industries. Ambiguity in the definition of DevOps makes adoption difficult since organizations might not know which practices they should implement for DevOps.

Examples of titles of other research papers…

DevOps Capabilities, Practices, and Challenges: Insights from a Case Study

Mali Senapathi
Auckland University of Technology
Auckland, New Zealand
mali.senapathi@aut.ac.nz

Jim Buchan
Auckland University of Technology
Auckland, New Zealand
jim.buchan@aut.ac.nz

Hady Osman
Auckland, New Zealand
hadyos@gmail.com

ABSTRACT

DevOps is a set of principles and practices to improve collaboration between development and IT Operations. Against the backdrop of the growing adoption of DevOps in a variety of software development domains, this paper describes empirical research into factors influencing its implementation. It presents findings of an in-depth exploratory case study that explored DevOps implementation in a New Zealand product development organisation. The study involved interviewing six experienced software engineers who continuously monitored and reflected on the gradual implementation of DevOps principles and practices. For this case study the use of DevOps practices led to significant benefits, including increase in deployment frequency from about 30 releases a month to an average of 120 releases per month, as well as improved natural communication and collaboration between IT development and operations personnel. We found that the support of a number of technological enablers, such as implementing an automation pipeline and cross functional organisational structures, were critical to delivering the expected benefits of DevOps.

1 INTRODUCTION

The DevOps concept [1] emerged to bridge the disconnect between the development of software and the deployment of that software into production within large software companies [2]. The main purpose of DevOps is to employ continuous software development processes such as continuous delivery, continuous deployment, and microservices to support an agile software development lifecycle. Other trends in this context are that software is increasingly delivered through the internet, either server-side (e.g. Software-as-a-Service) or as a channel to deliver directly to the customer, and the increasingly pervasive mobile platforms and technologies on which this software runs [3]. These emerging trends support fast and short delivery cycles of delivering software in the fast-paced dynamic world of the Internet. As such DevOps has been well received in the software engineering community and has received significant attention particularly in the practitioner literature [4]. Annual 'State of DevOps' reports show that the number of DevOps teams has increased from 19% in 2015 to 22% in 2016 to 27% in 2017 [5].

Examples of titles of other research papers…

Emerging Trends for Global DevOps: A New Zealand Perspective

Waqar Hussain*, Tony Clear*, Stephen MacDonell*
*Software Engineering Research Lab (SERL)
School of Engineering, Computer and Mathematical Sciences (SECMS), Auckland University of Technology (AUT)
Auckland, New Zealand
whussain,tclear,smacdonell@aut.ac.nz

***Abstract*—The DevOps phenomenon is gaining popularity through its ability to support continuous value delivery and ready accommodation of change. However, given the relative immaturity and general confusion about DevOps, a common view of expectations from a DevOps role is lacking. Through investigation of online job advertisements, combined with interviews, we identified key Knowledge Areas, Skills and Capabilities for a DevOps role and their relative importance in New Zealand's job market. Our analysis also revealed the global dimensions and the emerging nature of the DevOps role in GSE projects. This research adds a small advanced economy (New Zealand) perspective to the literature on DevOps job advertisements and should be of value to employers, job seekers, researchers as well educators and policy makers.**

Keywords*— *GSD; GSE; DevOps; Continuous Integration; Continuous Deployment; Education; Empirical, Analysis; Online Job Postings Analysis; Content Analysis' Cloud; AWS.

I. Introduction

DevOps has recently gained popularity as a philosophy that synergizes the operational silos of Software Development (Dev) and IT Operations (Ops) [1]. The three main catalysts that propel its rapid adoption include: a) higher quality expectations from software as it is increasingly offered as a service in the cloud b) demands for rapid delivery of change with growing acceptance of agile and its change embracing attitude, and c) the availability of on-demand powerful and plentiful hardware on the cloud [2]. The uptake of the DevOps trend has been global [1]. Some large organizations claim to have successfully applied its practices in their distributed teams and achieved smooth team collaboration, shortened feedback loops and better customer collaboration [3].

A DevOps strategy supports a globally scalable, rapid and incremental service delivery strategy within a cloud computing infrastructure. Thus it offers potential for software and services companies to operate and compete successfully beyond the traditional centres of technology innovation. For many 'Small Advanced Economies[1] (SAEs) such as New Zealand [4], this service model has attractions. Governments and the IT sector see opportunities to move themselves up the global value chain through delivery of high value products and services. Underpinned by a skilled population base, they believe this will result in more high paying jobs and greater export income [5].

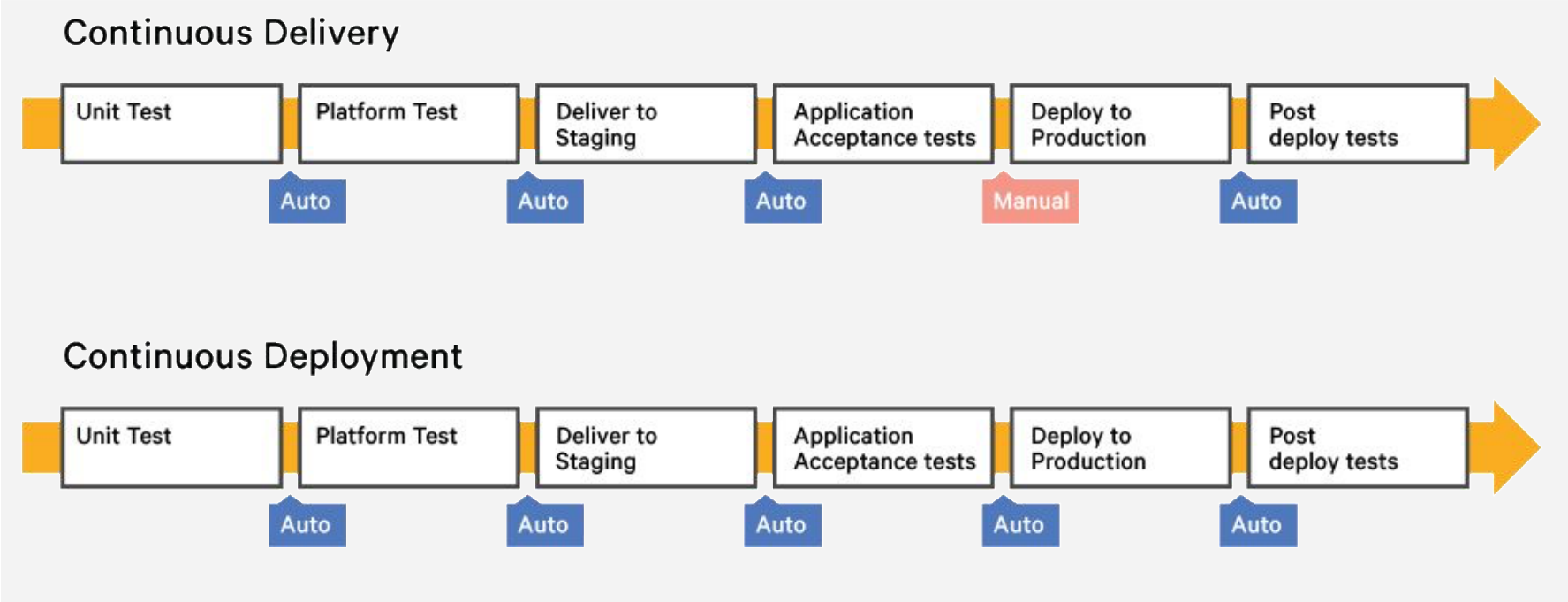
Many believe that DevOps is here to stay, at least in many IT sectors to help organizations deliver quality service with efficiency [2]. However, given the relative recency and emergent nature of the DevOps phenomenon, an inadequate body of knowledge, and general confusion surround DevOps concepts and definitions [6]. The software industry therefore, does not share a common view of its meaning. This lack of clarity fuels several misperceptions. Employers are unable to set and describe the right expectations (Knowledge Skills and Capabilities (KSCs)) for DevOps roles. Job seekers are thus unsure of the commonly assigned responsibilities and expectations [7]. Educators on the other hand, find it challenging to train students and impart the desired skillsets and capabilities for their smooth transition into these roles [8].

A recent study has compared responsibilities of a DevOps engineer role in three countries (USA, UK and Canada) and has shown that the responsibilities and skills expected from DevOps roles significantly vary from country to country [8]. Motivated by country specific differences in their study, this paper analyzes what New Zealand employers actually state they require in DevOps job listings and the extent to which Global Software Engineering (GSE) aspects are included, whether explicitly or implied. The aim is to better understand the growing phenomenon of DevOps in the New Zealand job market, (as one example of an SAE), identify major KSCs and understand how some of those relate to GSE. We draw on interviews and insights from practitioners, to complement the job description data. The questions this research tries to address are:

1. *What knowledge areas, skills and capabilities (KSCs) are in demand for a DevOps role in New Zealand's IT market?*

Case Study: The Software Development process

- Cross functional Scrum teams
- Product Owner
- Teams organised by product functional module
- Sprints (2-3 weeks)
- Product Backlog
- Daily standups
- Sprint planning
- Sprint review
- Retrospectives



it is continuous delivery of small features or feature sets
May not suit an iterative approach like Scrum
May suit more of a a Kanban Approach

Diagram Used with permission of Hady Osman

Case Study: Team structures

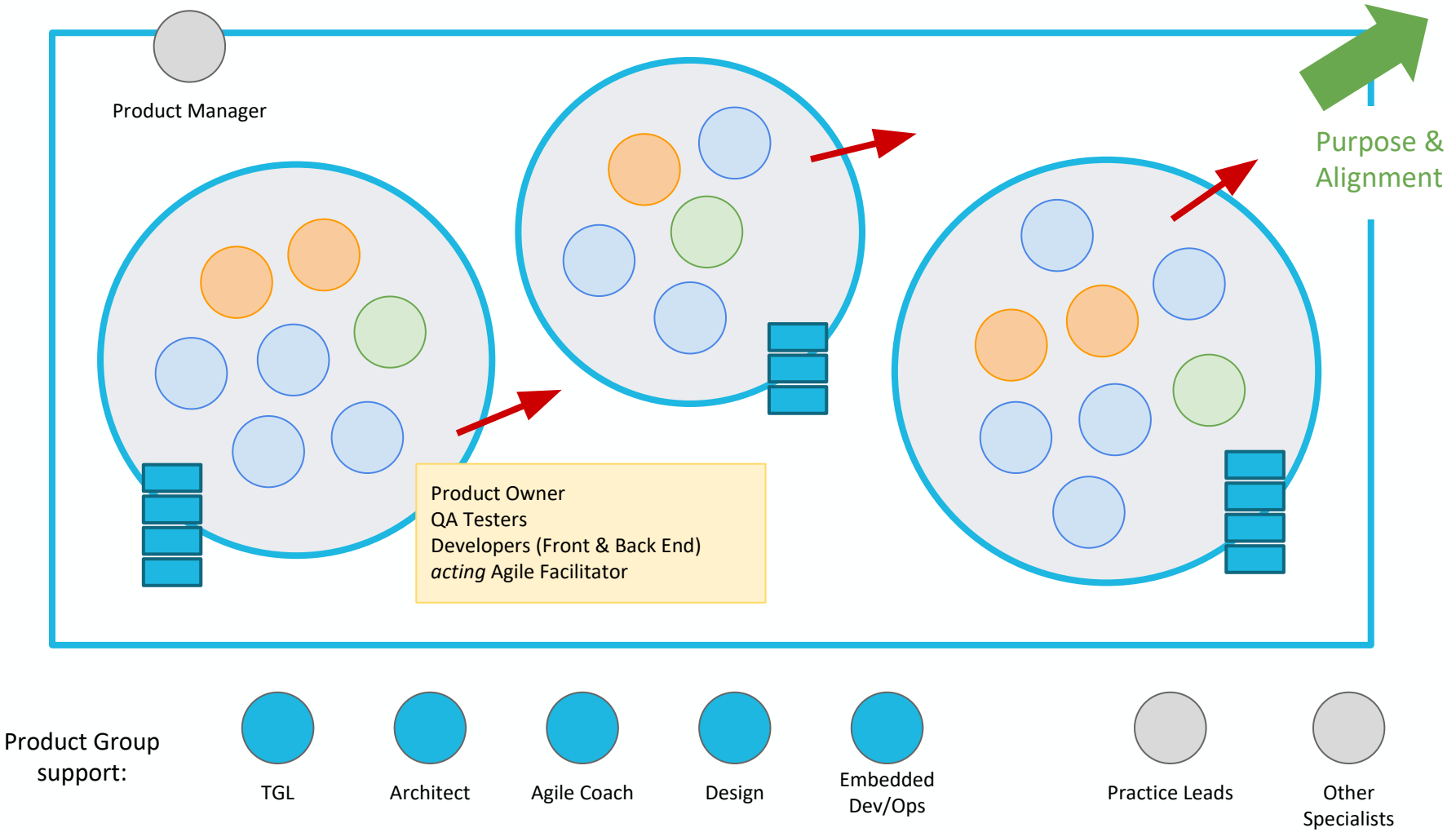
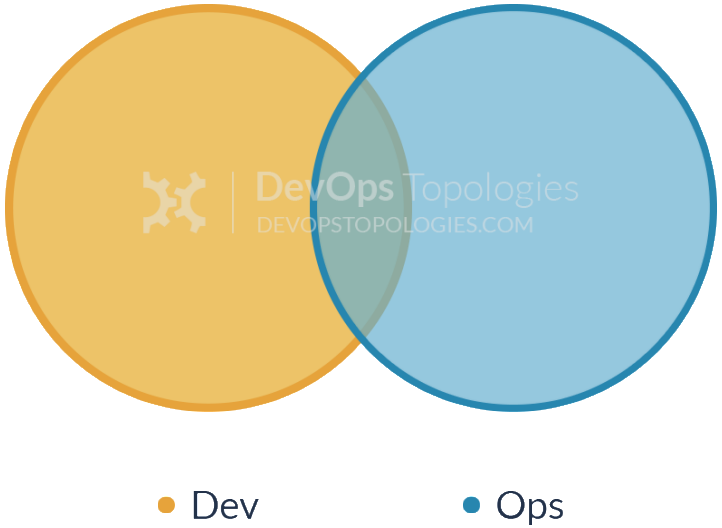


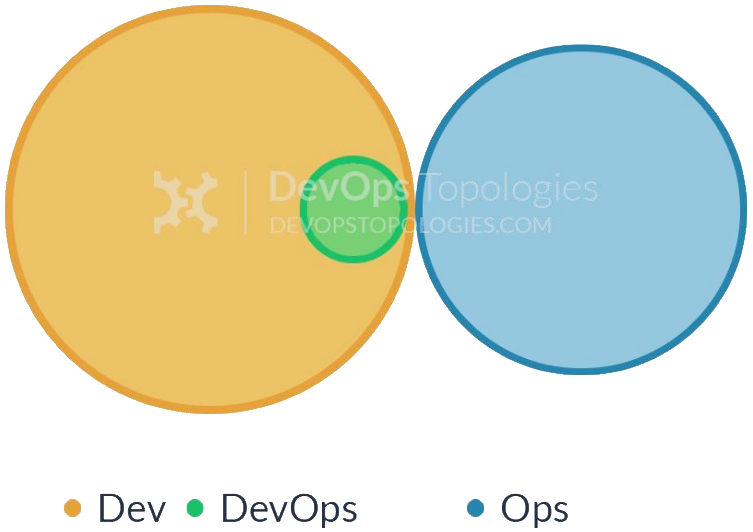
Diagram Used with permission of Hady Osman

Team structures – patterns and anti-patterns



Type 1 suitability: an organisation with strong technical leadership.

Potential effectiveness: HIGH



Type 3 suitability: organisations with several different products and services, with a traditional Ops department, or whose applications run entirely in the public cloud.

Potential effectiveness: MEDIUM

Devopstopologies.com

The Shared meaning of DevOps

What are the main concepts associated with the meaning of DevOps?

User feedback of new feature in production- try small things out and learn fast!

Is there a shared meaning of DevOps – Case Study

Embedded ops. Ops, Dev, QA work together. Ops is considered from the start and in a longer term (*QA Release manager*)

It just means you are not relying on other teams to do the infrastructure. You have control over it – choice of tool to use for example (*Developer*)

Bringing [ops and dev] skill sets together ….and bringing people together… with more natural communication…. Also looking at automation as a rule-of-thumb (*Team lead Infrastructure*)

Deployment [of a feature] is in the control of the team that develops it…..We write code, review it, test it, merge it and deploy it. (*Ops in Team)*

I think it's a name for the period of time where software developers transition from just giving their stuff to system admins when it is finished, to actually caring and looking after it themselves. (*Training manager*)

Every team owns its own infrastructure rather than …someone else is doing the job when we have problems. (*Tester/QA*)

Is there shared understanding of the meaning of DevOps? - Literature

- The combination of people, culture, process, tools and methodologies that reduce risk and cost, enable technology to change at the speed of the business and improve overall quality.[13]
- DevOps describes the practices that streamline the software delivery process, emphasizing the learning by streaming feedback from production to development and improving the cycle time. [11]
- DevOps is a movement intended to reduce the time between code complete and code in production as well as share responsibility and increase collaboration between developers and system administrators [2]
- DevOps is a job title

The meaning of DevOps – main concepts

Bringing Dev and Ops closer

- **collaborate** (rather than blaming)– planning, design, code.
- **Communicate** early and frequently and face to face
- Extend team **capability**, influence each other's design of code and infrastructure,
- Team **responsibility** extended to **deployment**, infrastructure, as well as **post-deployment** monitoring, debugging and fixing issues in deployed features (on-call is shared)
- what the team **cares** about is changed

Get **team capability** to deliver features frequently to:

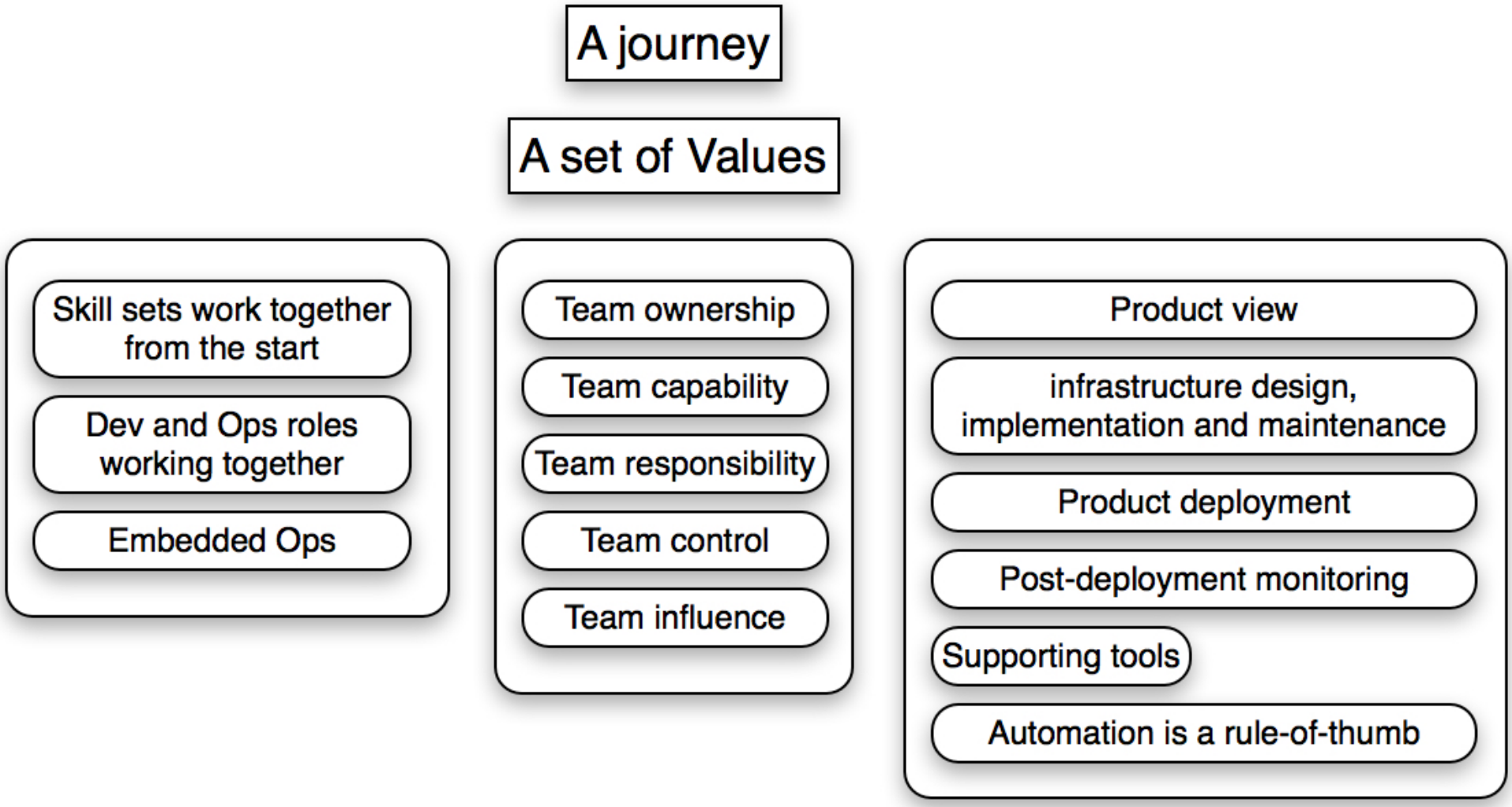
- create value for customers quickly and
- get fast feedback on usefulness to users
- Learn from mistakes
- Reduce risk

A movement – **a set of values** – a way of thinking. A way of working - tools and practices that support the realisation of these values and assumptions.

A **journey**…stages of adoption

A **job title** – with Dev and Ops (and QA?) capabilities

What does DevOps mean…



What DevOps is NOT

- It isn't just a job title prefix
- It isn't a team (but "Dev" and "Ops" are)
- It isn't a technology
- It doesn't mean "a system administrator who can code"
- It doesn't mean "automating stuff"
- It doesn't mean "there's no operations team now"

https://thenextweb.com/syndication/2020/06/05/get-the-fundamentals-of-devops-right-then-worry-about-tools/

Triggers and Motivation for Adopting DevOps

What events and reasoning trigger/drive an organisation to start/continue) adopting DevOps?

What motivates a team to be willing to adopt DevOps?

What are the expected benefits/value to the customer/organisation/team?

Who (what roles) drive the move to DevOps?

Triggers and goals of adopting DevOps

The end goal is that we can institute change sort of fast but with integrity. We accept more risk for this speed. (*Embedded Ops*)

Cadence and speed. To be able to deliver quality software at speed you need to have fewer points along the journey. (*Training Manager*)

Continuous deployment I think. The ability to make a change and have it reflected in the real word instantly.(*Team lead Infrastructure*)

It just means you are not relying on other teams to do the infrastructure. You have control over it – choice of tool to use for example. To get the feel of small startups in a big organisation. (*Developer*)

The goal is for the production team to own the infrastructure (*Tester*)

Development team wanted more hands-on with infrastructure (*QA Release manager*)

As the teams grew there was a bottleneck to get stuff into production because we had to give it to the Ops team. DevOps was to overcome this bottleneck (*Training Manager*)

Avoid the double ups and start-stops in communications between ops and devs dealing with an issue ticket.(*Team lead Infrastructure*)

Avoid the outages needed for large releases (*Team lead Infrastructure*)

Triggers and Motivation for Adopting DevOps

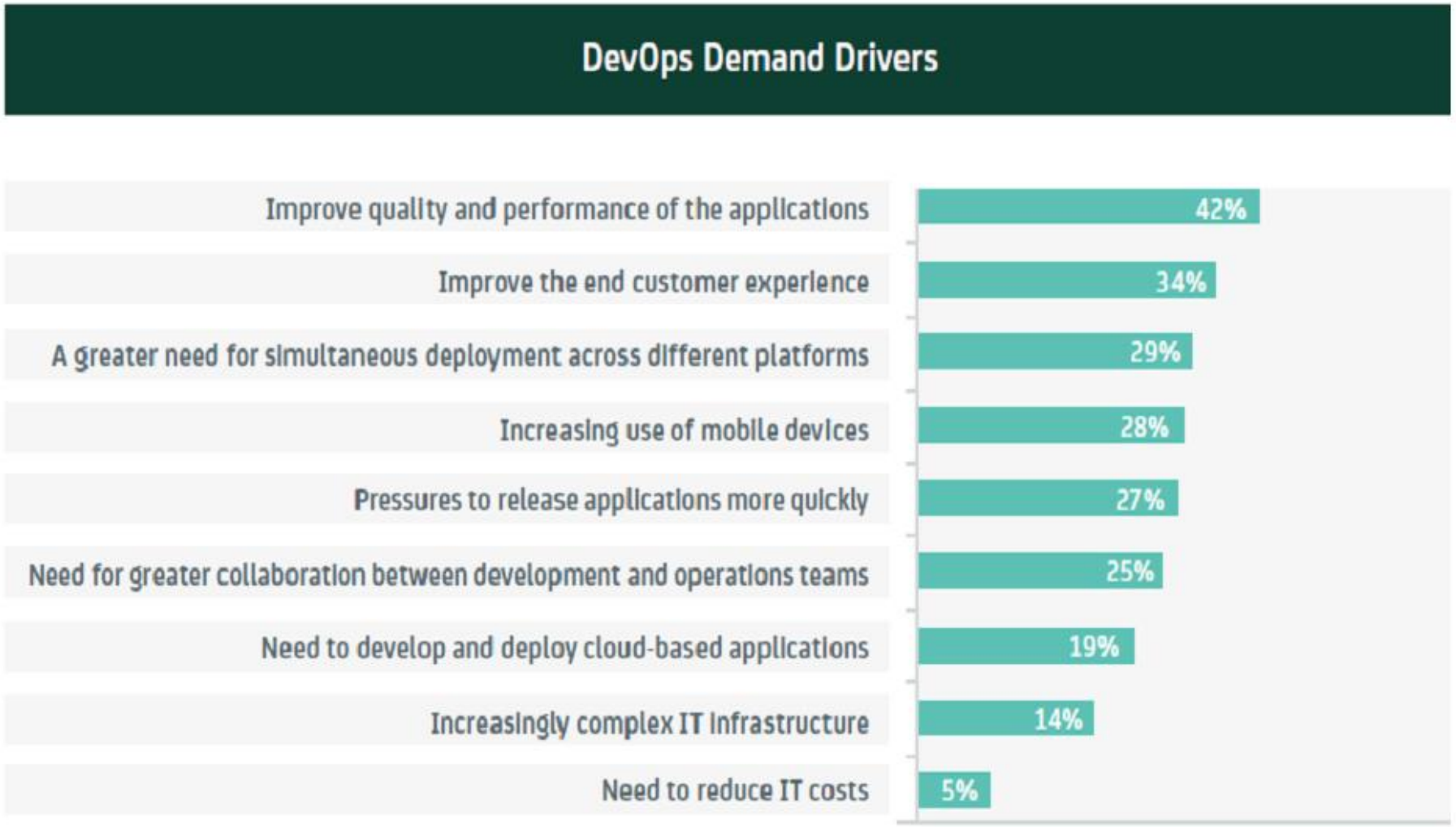


Figure 2what is driving need for DevOps?

[13] (literature review)

Triggers and Motivation for Adopting DevOps

Developers driving/willing to avoid pain of Ops bottleneck

Managers – driving for faster feature release to respond to users needs quicker. Also fewer outages (planned and unplanned) – improved customer experience.

Ops – increase speed of feature releases.

Drivers for Adopting DevOps

Conversation with pre-DevOps champions Chief Platform Officer and Chief Product Officer

frequent frustration between the company's operation and product teams who have had **competing priorities**

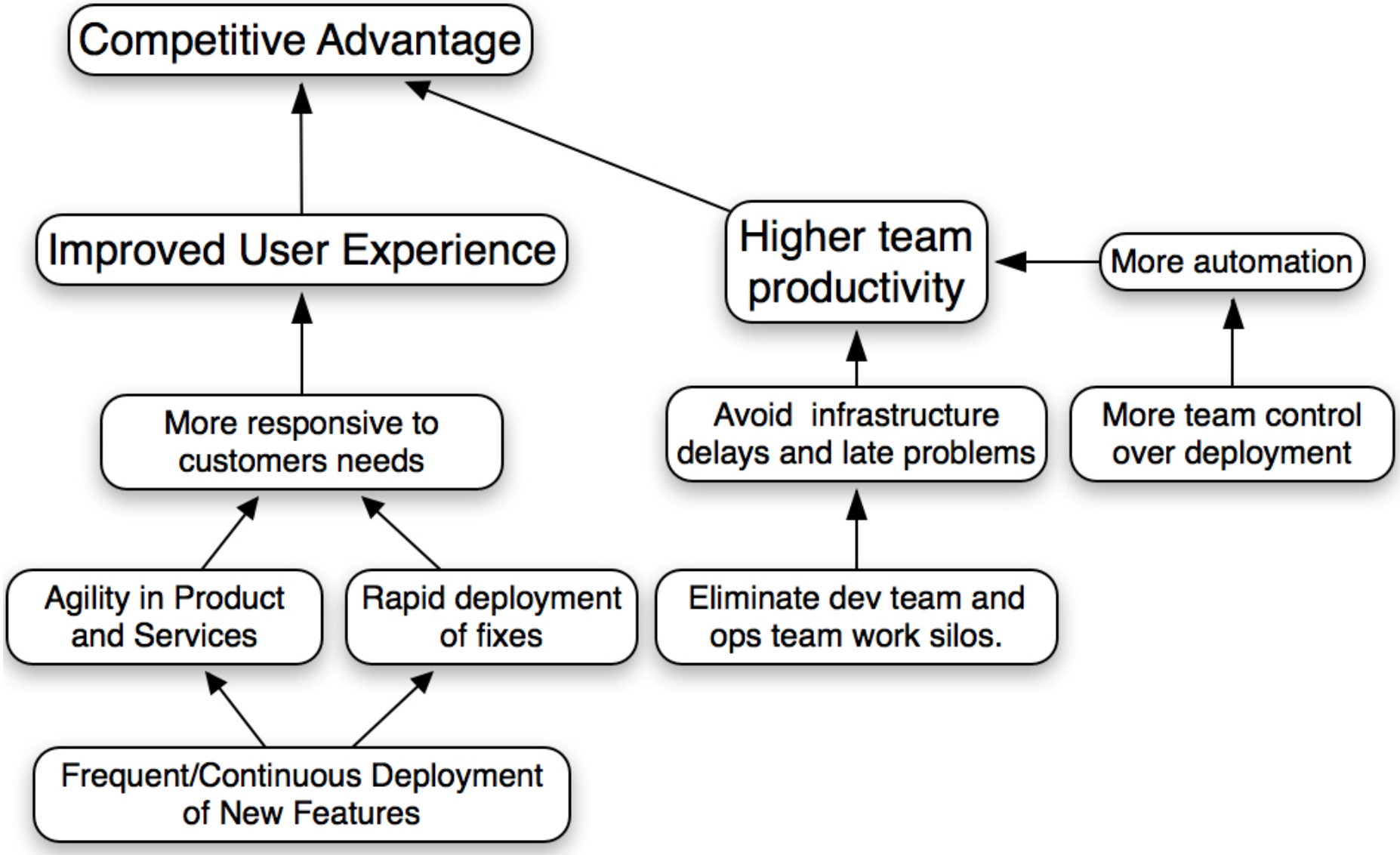
Operation and Product teams operated under what was identified as a **mismatch of incentives and control**.

- Operation teams were accountable for performance and uptime, development teams were in a better position to improve it
- development teams were accountable for shipping product with great agility and velocity, operation teams were in controls of major portions of the SDLC

Strategic tech changes required **infrastructure as code capability**

- skill set is dev and ops

The drivers to adopt DevOps



Benefits of DevOps Adoption

What are the expected/claimed benefits of adopting DevOps?

What are the actual organisational/customer/team benefits realised as a result of adopting a DevOps way of working?

What evidence is there that these benefits ar realised?

Case Study - Benefits of DevOps Adoption

Team ownership and responsibility is huge, the devs and QAs have loved it. More frequent releases – because more deployers and smaller releases. Easier to contain a release. More features for end users. (*QA Release manager*)

You can build so much better integrity. You build your own solutions that work for your own environment. *(Embedded Ops)*

You write better code because you know what's gong to happen to it. (*Developer*)

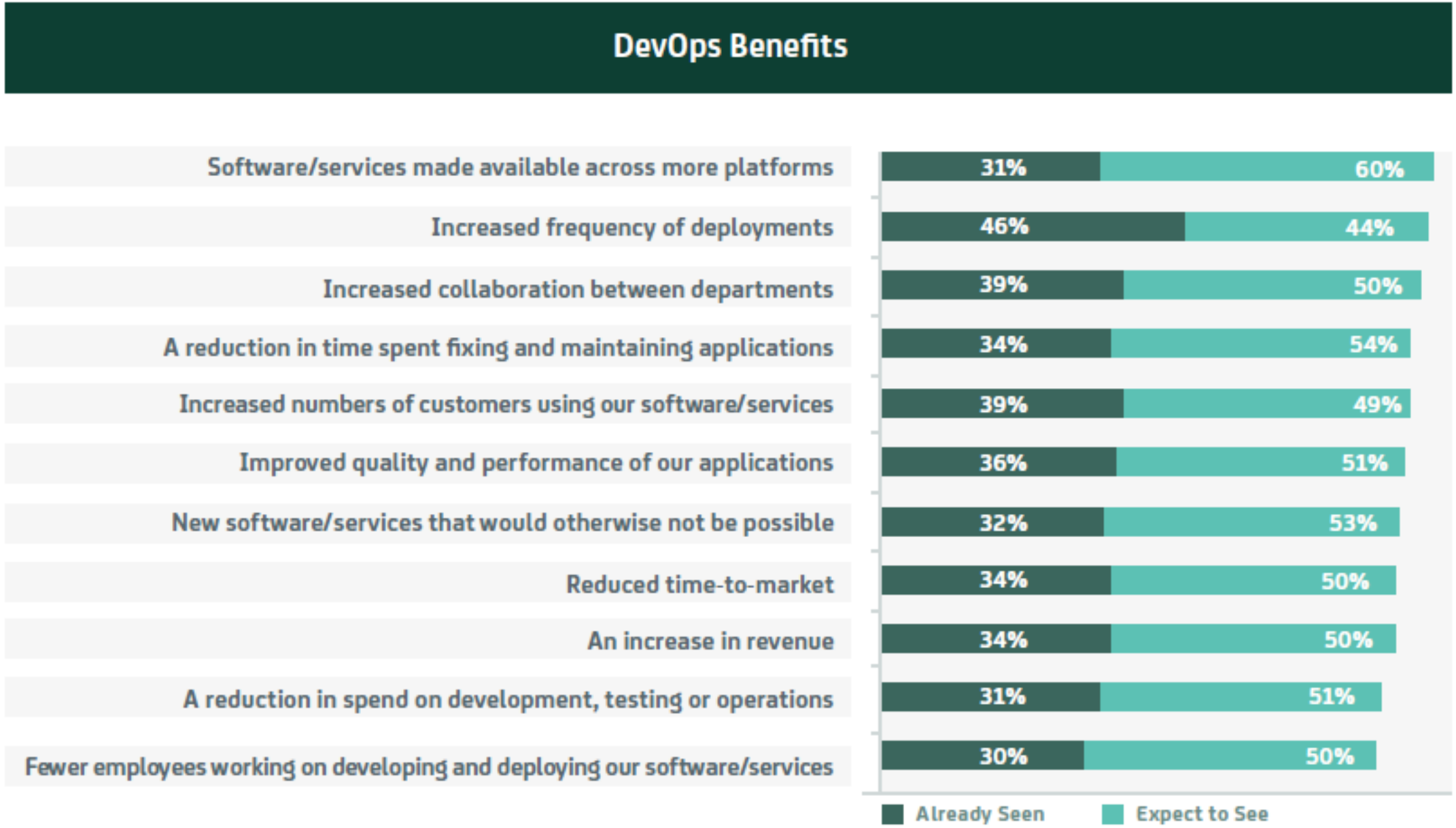
Thinking about infrastructure before the end – getting more rapid development. Releases smaller and less risky – fewer outages. (*Training manager*)

Reduced the dependencies and reliance on Ops team for infrastructure related tasks needed for a new feature (eg a new DNS entry could take a week – the Dev team wouldn't even know what to ask sometimes). (*Team lead Infrastructure*)

Speed up changes to infrastructure needed for a release (fewer checkpoints, and in our control) (*Tester/QA*)

Better shared knowledge – we can diagnose and fix problems quicker. (*Tester/QA*)

Benefits of DevOps Adoption – literature [13]



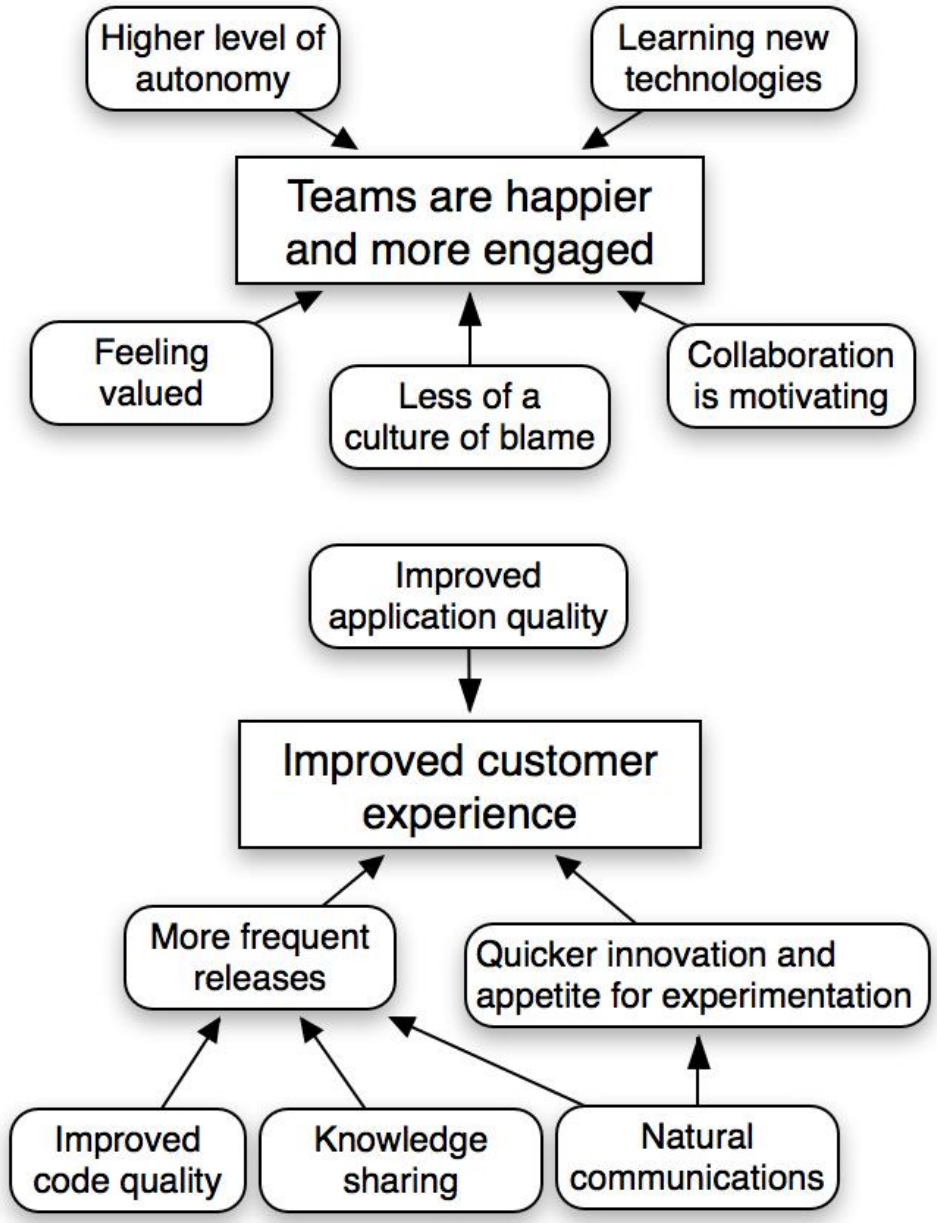
Tony (

Benefits of DevOps Adoption

More deploys means **faster time-to-market** and an **enhanced feedback loop**. With an enhanced feedback loop, organizations have a better idea of customer reactions to features and consequently, better means of **continuous improvement**.[5]

DevOps also contributes to **stability enhancements**. Stability in this context stands for error-free operation of a system and the ability to keep processing requests. **Enhanced change success rate and (60 x) deployment success rate[5]**

Perceived Benefits of Working the DevOps way



Enablers and success factors of DevOps Adoption

What are some **success** factors of DevOps adoption?

What capabilities, culture, practices, technologies and tools are significant enablers of successful DevOps adoption?

Agile and Lean

Automation

Service Frameworks

Enablers and success factors

Trusting the production team to deploy was difficult (*QA Release manager*)

The team can set up their own dashboard for monitoring whatever they think is important post-deployment of a feature. (*Ops in Team)*

Lots of communications – you can never have enough. (*Developer*)

Cross training the development team on ops and having ops take part in the production team meetings. It's hard just getting the right people, though. (*Training manager*)

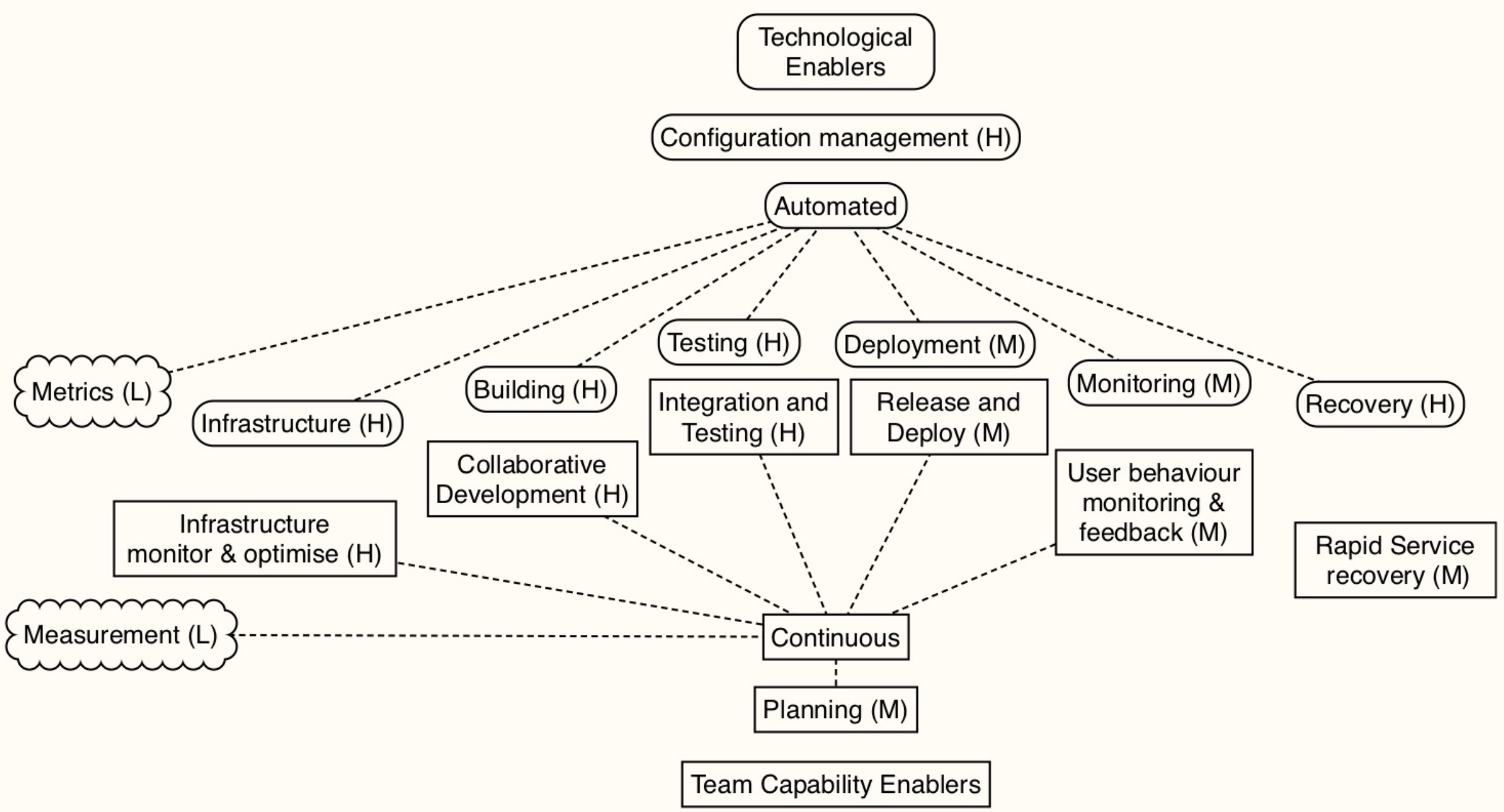
The switch from a monolithic application to microservices was a big thing.(*Team lead Infrastructure*)

Having a set of tools to automate testing and builds and continuous integration(*Tester/QA*)

Enablers of Value of DevOps Adoption

Capabilities	Continuous planning Collaborative and continuous development Continuous integration and testing Continuous release and deployment Continuous infrastructure monitoring and optimization Continuous user behavior monitoring and feedback Service failure recovery without delay
Cultural Enablers	Shared goals, definition of success, incentives Shared ways of working, responsibility, collective ownership Shared values, respect and trust Constant, effortless communication Continuous experimentation and learning
Technological Enablers	Build automation Test automation Deployment automation Monitoring automation Recovery automation Infrastructure automation Configuration management for code and infrastructure

Technical and team capability Enablers of success in DevOps



Challenges and Barriers of DevOps

What are some inhibitors/barriers to successfully adopting and continuing with DevOps

What are significant challenges in implementing DevOps in practice?

Challenges?

Staffing is probably our biggest challenge. (*QA Release manager*)

Windows is a big challenge. It's hard to get the right tools together for automation.(*Ops in Team)*

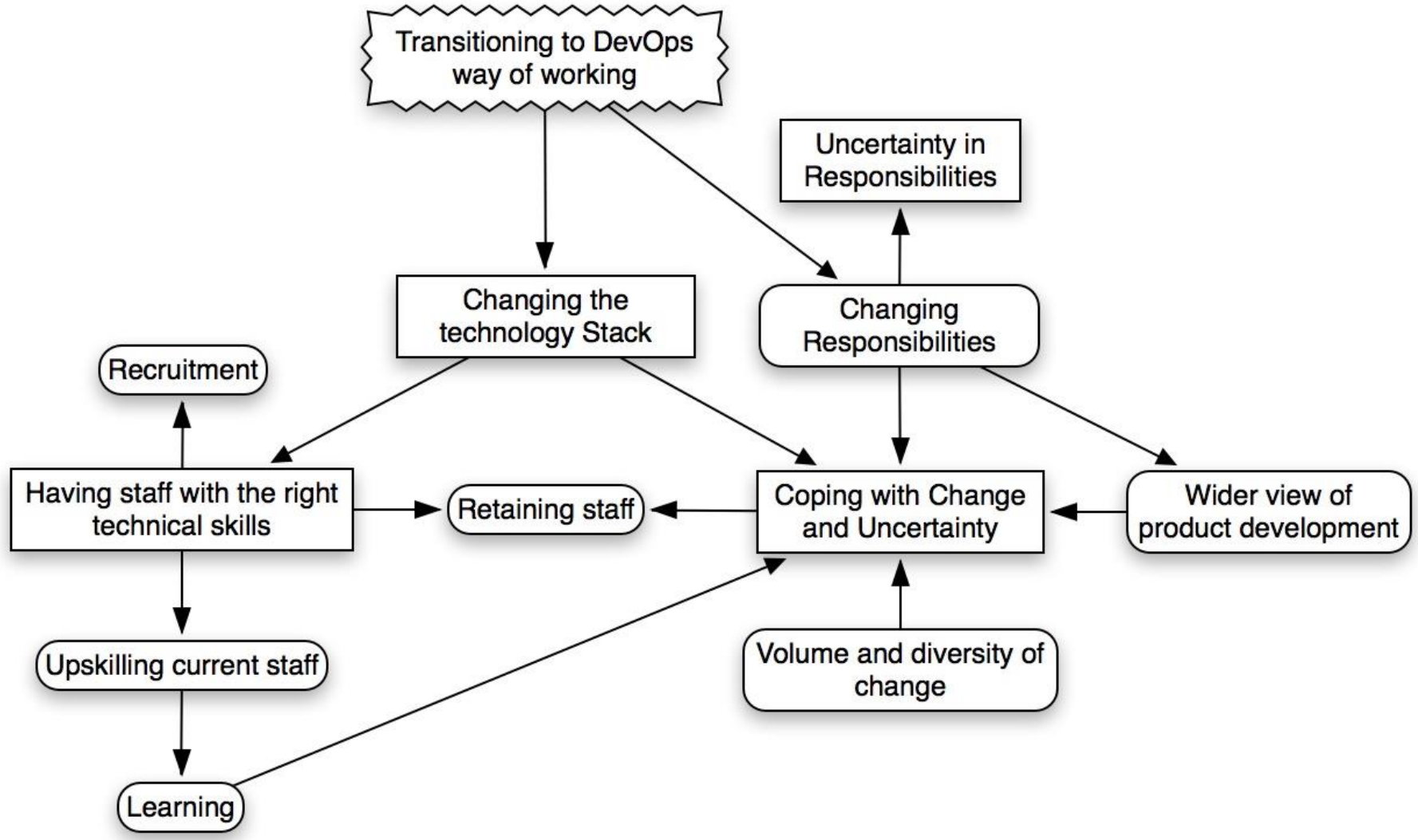
The big mental shift was huge (*Developer*)

Upskilling the entire team so anyone has the capability to be on-call. (*Training manager*)

Changing the technology stack to the cloud and micro-services was a big enabler, but it was incredible complex. (*Team lead Infrastructure*)

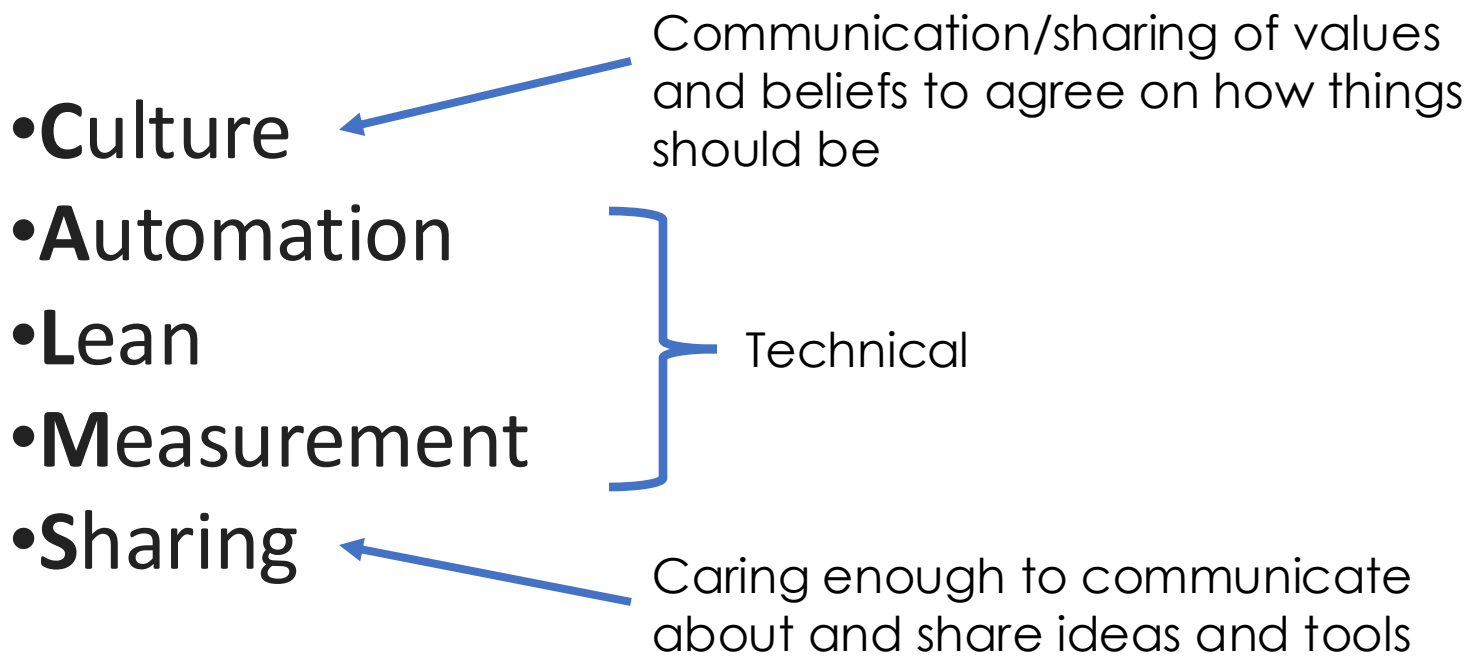
There are so many new tools and ideas, it's hard just keeping up. (*Tester/QA*)

Challenges in working the DevOps way



DevOps principles: Keep your CALM(S) first (then select some tools to support this…)

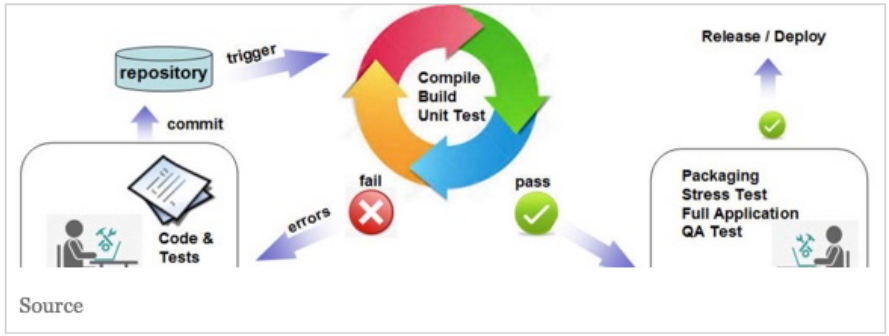
CALMS is a very accurate description of how to practice DevOps:



https://thenextweb.com/syndication/2020/06/05/get-the-fundamentals-of-devops-right-then-worry-about-tools/

Technology & DevOps

Automated Testing Tools



Source

We must start an evaluation of automated testing tools by first fitting them into the testing pyramid. The testing pyramid has 4 layers:

- **Unit** — This is your base of all automated testing. As far as volume is concerned, you should have the most unit tests compared to other types. These tests should be written and run by software developers to ensure that a section of an application (known as the "unit") meets its design and behaves as intended.
- **Component** — The main objective of component testing is to verify the input/output behavior of the test object. This ensures that the test object's functionality is working correctly, as per the desired specification.
- **Integration** — This is the phase in testing in which individual software modules are combined and tested as a group.
- **End-to-End** — This layer is self-explanatory. We're looking at the flow of an application, right from the start to the finish, and making that it's behaving as expected.

#1 Integration testing tool of 2020: Cucumber

#1 End-To-End Testing Tool of 2020 — Functional: SoapUI Pro

#1 End-To-End Testing Tool of 2020 — Load Testing: LoadRunner

- Development and Build Tooling
- **Automated Testing Tools**
- Deployment Tooling
- Runtime DevOps Tooling
- Collaboration DevOps Tooling

https://medium.com/better-programming/must-learn-devops-tools-for-2020-1a8a2675e88f

Technology & DevOps

- **The Periodic Table of DevOps Tools**

  https://digital.ai/learn/devsecops-periodic-table/

Summary

DevOps is a way of working that combines culture and values with practices and tools to provide the development team with capability to continuously develop, deploy and monitor new features

Drivers – some context dependent. Expectation is that more frequent delivery will result in increased agility and better customer experience.

Benefits – some context dependent. Frequent delivery resulted in increased collaboration, communications, agility and better customer experience. Value all round!

Enablers – clear understanding of value, responsibilities. Upskilling support. Supporting automation tools. Architecture.

Challenges – change management and supporting it. Finding staff, upskilling and retaining staff.

DevOps Capabilities, Practices, and Challenges: Insights from a Case Study

M. Senapathi, J. Buchan and H. Osman

In Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018 (EASE'18). ACM, New York, NY, USA, 57-67. DOI: https://doi.org/10.1145/3210459.3210465

Emerging Trends for Global DevOps: A New Zealand Perspective

Hussain, W., Clear, T., & MacDonell, S.

In D. Cruzes & A. Sharma (Eds.), *Proceedings 2017 IEEE 12th International Conference on Global Software Engineering* (pp. 21-30). IEEE. https://doi.org/10.1109/ICGSE.2017.16

References

[1] B. Fitzgerald, N. Forsgren, K.-J. Stol, J. Humble, and B. Doody, "Infrastructure Is Software Too!," 2015.

[2] T. A. Limoncelli and D. Hughes, "LISA'11 Theme—"DevOps: New Challenges, Proven Values"," *USENIX; login: Magazine*, vol. 36, 2011.

[3] L. Riungu-Kalliosaari, S. Mäkinen, L. E. Lwakatare, J. Tiihonen, and T. Männistö, "DevOps Adoption Benefits and Challenges in Practice: A Case Study," in *Product-Focused Software Process Improvement: 17th International Conference, PROFES 2016, Trondheim, Norway, November 22-24, 2016, Proceedings 17*, 2016, pp. 590-597.

[4] S. Elliot, "DevOps and the cost of downtime: Fortune 1000 best practice metrics quantified," *International Data Corporation (IDC)*, 2014.

[5] J. Hamunen, "Challenges in adopting a Devops approach to software development and operations," 2016.

[6] J. Smeds, K. Nybom, and I. Porres, "DevOps: a definition and perceived adoption impediments," in *International Conference on Agile Software Development*, 2015, pp. 166-177.

[7] L. Bass, I. Weber, and L. Zhu, *DevOps: A Software Architect's Perspective*: Addison-Wesley Professional, 2015.

[8] J. Wettinger, U. Breitenbücher, and F. Leymann, "DevOpSlang–bridging the gap between development and operations," in *European Conference on Service-Oriented and Cloud Computing*, 2014, pp. 108-122.

[9] L. Baresi, S. Guinea, A. Leva, and G. Quattrocchi, "A discrete-time feedback controller for containerized cloud applications," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2016, pp. 217-228.

[10] K. Nybom, J. Smeds, and I. Porres, "On the Impact of Mixing Responsibilities Between Devs and Ops," in *International Conference on Agile Software Development*, 2016, pp. 131-143.

[11] M. Hüttermann, "Gain Fast Feedback," in *DevOps for Developers*, ed Berkeley, CA: Apress, 2012, pp. 81-94.

[12] J. Sussna, "Cloud and DevOps: A Marriage Made in Heaven," *Introducing DevOps to the Traditional Enterprise/eMag*, vol. 14, 2014.

[13] S. Nagpal and A. Shadab, "Literature Review: Promises and Challenges of DevOps."

[14] V. P. R. Kumar and V. Babu, "Devops-a review," *Image and Video Processing*, p. 32, 2012.

[15] P. Duvall, "Breaking down barriers and reducing cycle times with DevOps and continuous delivery," *Retrieved from GigaOM Pro website: http://try. newrelic. com/rs/newrelic/images/GigaOm-Pro-Report-Breaking-down-barriersand-reducing-cycle-times-with-devops-and-continuous-delivery. pdf*, 2012.

[16] J. Humble and J. Molesky, "Why enterprises must adopt devops to enable continuous delivery," *Cutter IT Journal*, vol. 24, p. 6, 2011.

[17] S. W. Hussaini, "Strengthening harmonization of development (dev) and operations (ops) silos in it environment through systems approach," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, 2014, pp. 178-183.

[18] J. Kim, C. Meirosu, I. Papafili, R. Steinert, S. Sharma, F.-J. Westphal, *et al.*, "Service provider DevOps for large scale modern network services," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, 2015, pp. 1391-1397.

[19] M. Goodwell. (2016, 2016-03-27). DevOps in a Microservices World. Available: https://dzone.com/articles/devops-microservices-world

[20] L. E. Lwakatare, T. Karvonen, T. Sauvola, P. Kuvaja, H. H. Olsson, J. Bosch, *et al.*, "Towards DevOps in the Embedded Systems Domain: Why is It So Hard?," in *System Sciences (HICSS), 2016 49th Hawaii International Conference on*, 2016, pp. 5437-5446.

[21] J. Greenough, "How the 'Internet of Things' will impact consumers, businesses, and governments in 2016 and beyond," *Business Insider*, 2016.

[22] A. Storms, "How security can be the next force multiplier in devops," *RSAConference,(San Francisco, USA)*, 2015.

[23] H. Karl, S. Dräxler, M. Peuster, A. Galis, M. Bredel, A. Ramos, *et al.*, "DevOps for network function virtualisation: an architectural approach," *Transactions on Emerging Telecommunications Technologies*, vol. 27, pp. 1206-1215, 2016.

[24] ISACA, "Title," unpublished|.

[25] P. Labs, "State of DevOps Report," 2016.

[26] A. Balalaie, A. Heydarnoori, and P. Jamshidi, "Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture," *IEEE Software*, vol. 33, pp. 42-52, 2016.

[27] L. E. Lwakatare, P. Kuvaja, and M. Oivo, "Dimensions of DevOps," in *International Conference on Agile Software Development*, 2015, pp. 212-217.

[28] R. Jabbari, N. bin Ali, K. Petersen, and B. Tanveer, "What is DevOps?: A Systematic Mapping Study on Definitions and Practices," in *Proceedings of the Scientific Workshop Proceedings of XP2016*, 2016, p. 12.

[29] W. J. Geurts, "Faster is Better and Cheaper," in *INCOSE International Symposium*, 2016, pp. 1002-1015.

[30] J. Clapham, "DevOps – The Reluctant Change Agent's Guide," *Introducing DevOps to the Traditional Enterprise/eMag*, vol. 14, 2014.

[31] F. Colavita, "DevOps Movement of Enterprise Agile Breakdown Silos, Create Collaboration, Increase Quality, and Application Speed," in*Proceedings of 4th International Conference in Software Engineering for Defence Applications: SEDA 2015*, P. Ciancarini, A. Sillitti, G. Succi, and A. Messina, Eds., ed Cham: Springer International Publishing, 2016, pp. 203-213.

Later Developments and Framings - References

Dennehy, D., & Conboy, K. (2017). Going with the flow: An activity theory analysis of flow techniques in software development. *Journal of Systems and Software, 133*, 160-173.

Zhao, G. (2021-TBD). *Modelling Strategies for DevSecOps in a Global Setting: A Delphi Study* [Doctoral Thesis, Auckland University of Technology]. Auckland.

Thank you!

Questions and Comments….