Shaun Norris

Pivotal

Field CIO for Asia Pacific & Japan

Dapeng Liu

Head of Credit Risk Development team at DBS Bank in Singapore.

DBS + Pivotal - DevOps Experience Report

- Key Outcomes
- DevOps focus
- Culture

Business Outcomes


EUROMONEY

World's best bank 2019: DBS

Wednesday, July 10, 2019

The Singapore-based bank has developed into a business fit for purpose in the modern world. Is this the model for the bank of the future?

https://www.euromoney.com/article/b1fmmkjyhws0h9/world39s-best-bank-2019-dbs

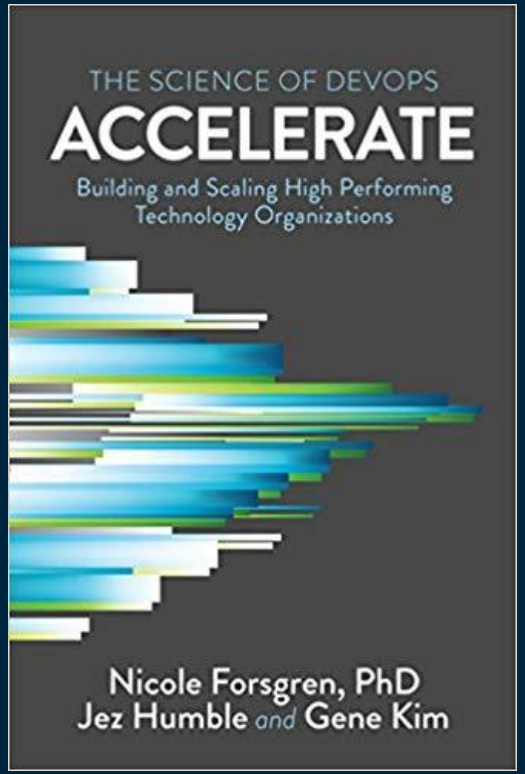
Technical Outcomes

Lead time:	**32 hours**	→	**3 hours**
Release Effort:	**12 hours**	→	**2 Hours**
Infra Provision:	**1 week**	→	**Intra Day**

Source: https://content.pivotal.io/springone-platform-2018/day-3-keynote-siew-choo-soh

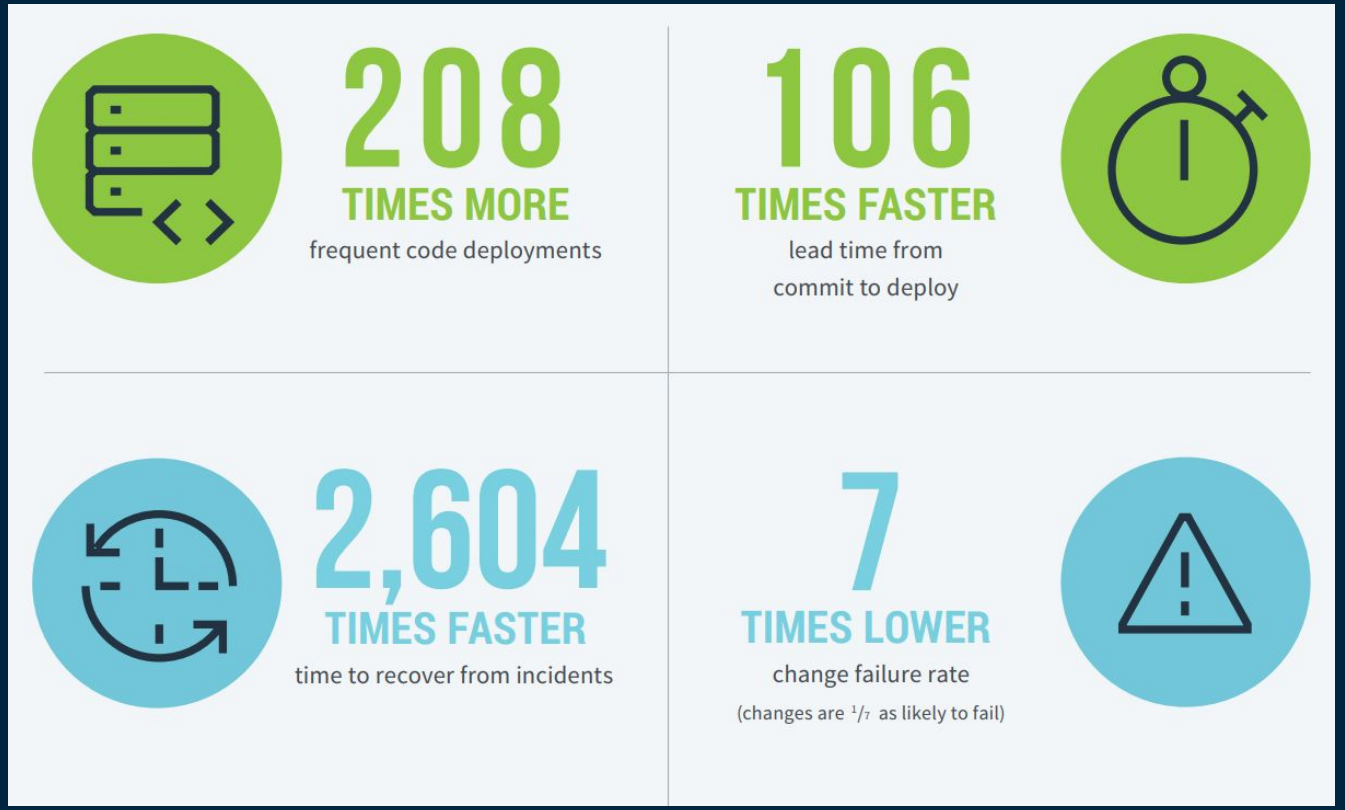
The 4 key DevOps metrics

Throughput	
Deployment Frequency	Lead Time for Changes
Change Failure Rate	Time to Recover
Stability	



https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling

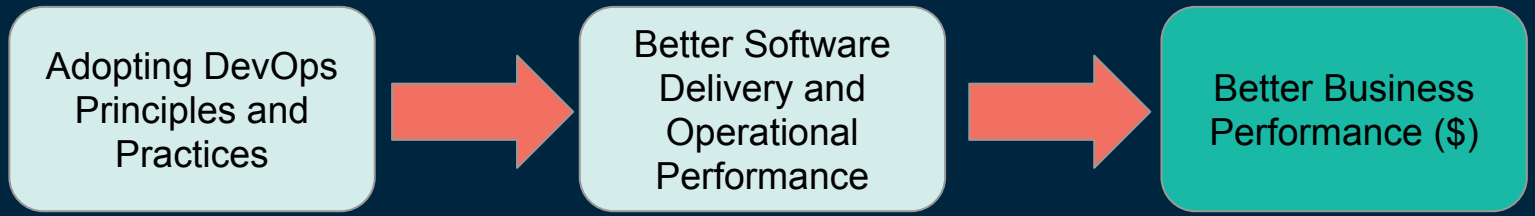
Elite Performers vs Low Performers



Pivotal

https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling

The Accelerate punchline

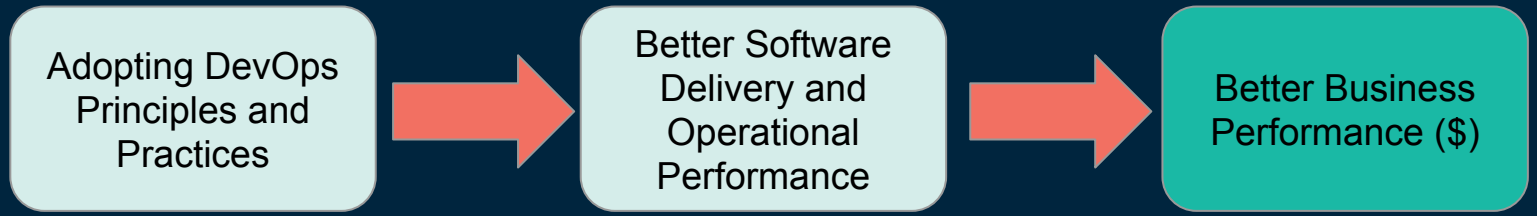


Why does this matter to every business?

"[the] highest performers are twice as likely to meet or exceed their organizational performance goals. "

https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling

in a DBS context



- Cloud native architecture
- Using a PaaS to abstract infra
- Using private cloud
- Automated delivery pipeline
- Self service infra provisioning

- 10x release cadence improvement
- Release prep time dramatically lower
- Deployments now in office hours, not on weekends
- Improvement in employee engagement

- World's Best Bank as voted by EuroMoney in July 2019

Accelerate suggests 7 technical practices that are key to improving software delivery.

Seven Key Themes from “Accelerate”

Two areas we want to focus on:

- **Platform as a Service**
- **Infrastructure as Code**
- Lean and Agile Adoption
- Open Source
- Cloud Computing
- Cloud Native Architecture
- Continuous Delivery

Infrastructure as Code

- Environments provisioned automatically via automation rather than hand-crafted
- Declarative, version-controlled environments

Teams using infrastructure as code are 1.8x more likely to be elite performers

https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling

How do you use infra as code at DBS?

- Self service infra on-demand for developers - same day delivery
- Pipeline automation
  - Automation to provision a git-compatible repo
  - Pipeline for build-deploy
  - Ticket only required for prod release
- 1:1 mapping of repo <> microservice
  - 260+ repos currently managed in these pipelines
  - Around 60 developers working on these
- Code-commit to UAT can easily be < 5 mins
  - Did this more than 2000x in Sep 2019

DevOps at DBS - Pipeline Automation

- Used to be that with every new service, would have to setup pipeline manually - now you fill in name of repo, where you want to deploy, pipeline auto-generated for you - No more manual pipeline / build tool definition file building
- In 2017 only 13 “micro” services - bigger ones had 100 Controllers so they were more of a macro-service - this was because pipeline setup took so long for each new microservice
    - moral of the story? Low-level infra friction can lead to high-level architecture challenges
- You don’t reach excellent state in one step - in retrospect, this was a reasonable transition - don’t expect first iteration to be perfect
- Iteration > revolution!

Platform as a Service

Teams who use a PaaS are 1.5x more likely to be elite performers (2018 SODR)

Why? Likely because their developers are spending more time above the value line and less time wrangling infrastructure.

What has DBS experience been using PaaS in your team?

- As a development team, using a platform to abstract infra has allowed devs to focus more on business logic
- We don’t use persistent storage - forces right design decisions
- Routing setup by default when we deploy
- Logging and log aggregation are automatic, no need to wire up for every application
- We use buildpacks to build containers, no need to manually write YAML files for docker containers
- 140+ applications running in production
- 600+ container instances running to support these apps

https://cloud.google.com/blog/products/devops-sre/the-2019-accelerate-state-of-devops-elite-performance-productivity-and-scaling

How has Operations improved?

- Incident Management?
- Coping with legacy process like change management?

DevOps at DBS - Incident management

- Uptime and availability - ran into one reporting system with a badly written query - app ended up crashing 5-10x in one day
- platform would automatically restart instances, and crashes were largely invisible to end business users - bought enough time to roll out a release fix
- “We largely don’t roll back any more, only roll forward with fixes in event of bugfixes”
- Previously, this type of issue on legacy platform would have caused significant downtime and pain
    - Crashes would not have easily been able to auto heal
    - Business would have been impacted
    - Time pressure to roll out urgent fix would have been greater

DevOps at DBS - Coping with legacy processes

- For change release process, we still deal with lots of manual approvals, we are still a small team in a large organisation
- One workaround is that we now autofill manual forms - writing code to generate excel forms rather than manually preparing and filling every time
- In this way, we still follows bank processes strictly but it is now far easier for teams to get it right the first time and for request reviewers to see all required info the first time

Culture Changes?

- Team Topology
- Sustainable pace
- Employee Engagement

Team Topology

- Previously we had a siloed org with lots of layers
  - now we do full-stack teams
- Used to have distinct dev and QA teams
  - we merged them
  - we still test code, but this isn’t a dedicated role anymore
- Used to be once we deployed to prod, was ops problem
  - now prod errors are fed back quickly to dev
  - people now want to be proud of what they’ve built and have a happily running app without lots of exceptions etc.

Working to a sustainable pace

- Pre 2017 overtime and weekend deployments were normal
  - no longer normal to work overtime and virtually no more Saturday deployments
- “If you look after people well, they will look after your applications well”
- every time we see people hanging around in office after 630 - we find out why and try to help them with better planning
- Now have official policy that if you work 1/2 day out of normal hours, you get 2x time off during normal business hours

Employee Engagement

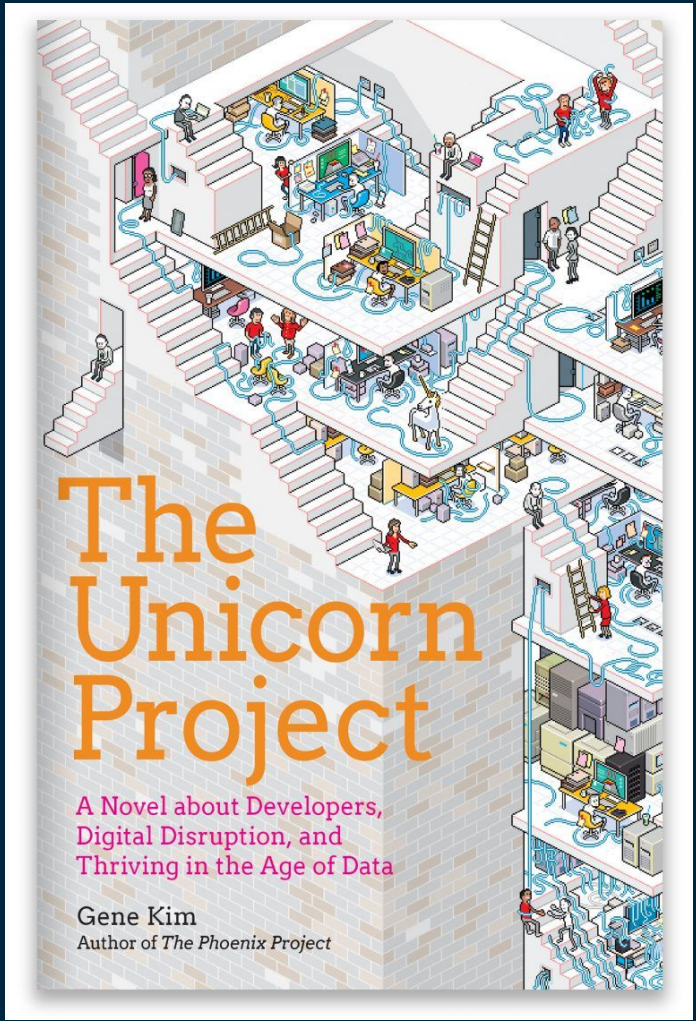
- First year team engagement score was ~ 66% vs dept ~ 80%
- By the next year, team score improved to 89%
- This corresponds to when we changed the ways of working in the team - got into the flow of continuous delivery
- cohesion within team now much higher than 2 yrs ago
    - E.g. weekly lunch and learn sessions run by the team

Advice for others starting out on this journey?

- The right processes will cultivate the right behaviour
  - e.g. why didn’t we split up microservices earlier
- Aim for small (~1% increments) over a long journey. As long as you are improving every day by 1% - big changes will happen over the long run
  - $1 with 1% daily interest gives almost 40x return after 1 year
- Look at the 4 Accelerate metrics as a way track progress

Challenges that remain

- automated change requests?



@RealGeneKim

Expected to ship
November 28th 2019
from IT Revolution Press