Dx and Deployment Strategies

Key Takeaways

- Business context is key
- Product categorization is key
- Taking a Platform view is important
- and … we will look at a few Deployment strategies

Shivagami Gugan

- **Technology Transformation Leader** who envisions and implements Business Transformation using disruptive Technologies in a manner that makes pure Business value. Envisioning and building the Transformation of Software engineering functions to DevOps and SRE culture in large-scale, mission-critical IT environments. Head of Software Engineering, Agile Transformation, DevOps, SRE , Cloud, Data and Analytics Architect.
- My primary passion is to demystify the mumbo-jumbo around Dx and execute the transformation within a Business context.
- Currently driving Tech transformation for an enterprise in UAE.

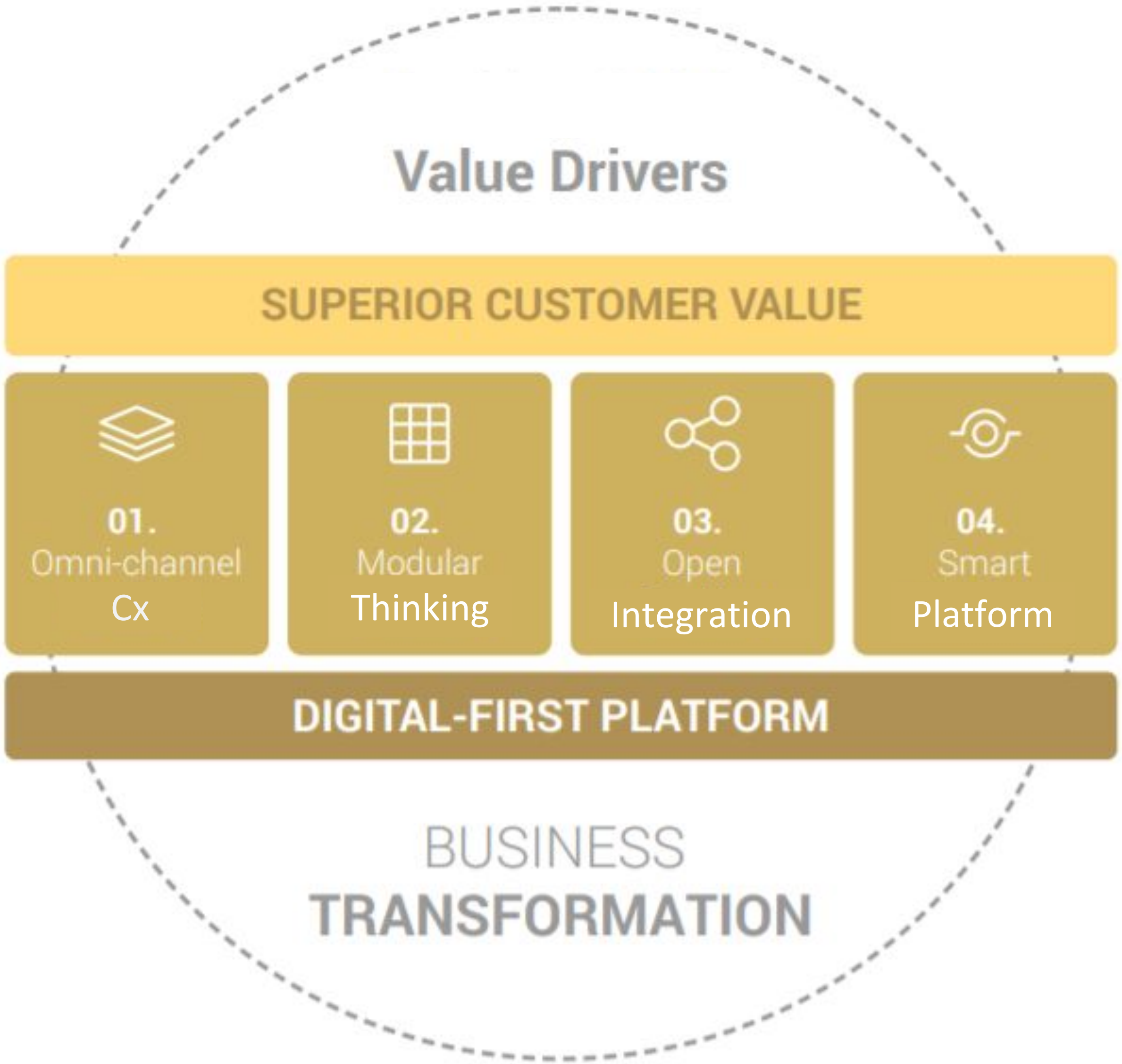
AMBASSADOR

DevOps

INSTITUTE

Human of DevOps

Dx



Value Drivers

SUPERIOR CUSTOMER VALUE

01. Omni-channel Cx

02. Modular Thinking

03. Open Integration

04. Smart Platform

DIGITAL-FIRST PLATFORM

BUSINESS TRANSFORMATION

DX Leaders

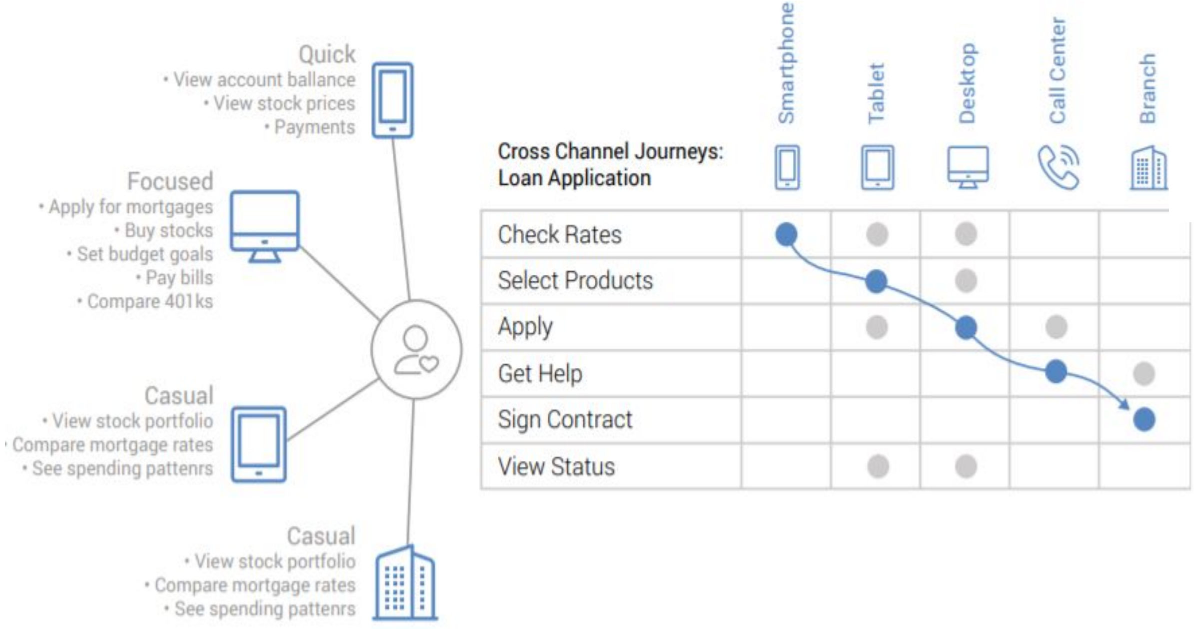
"Digital transformation is the shift from organization-centric to customer-centric culture."

— GERRY MCGOVERN

DXSUMMIT

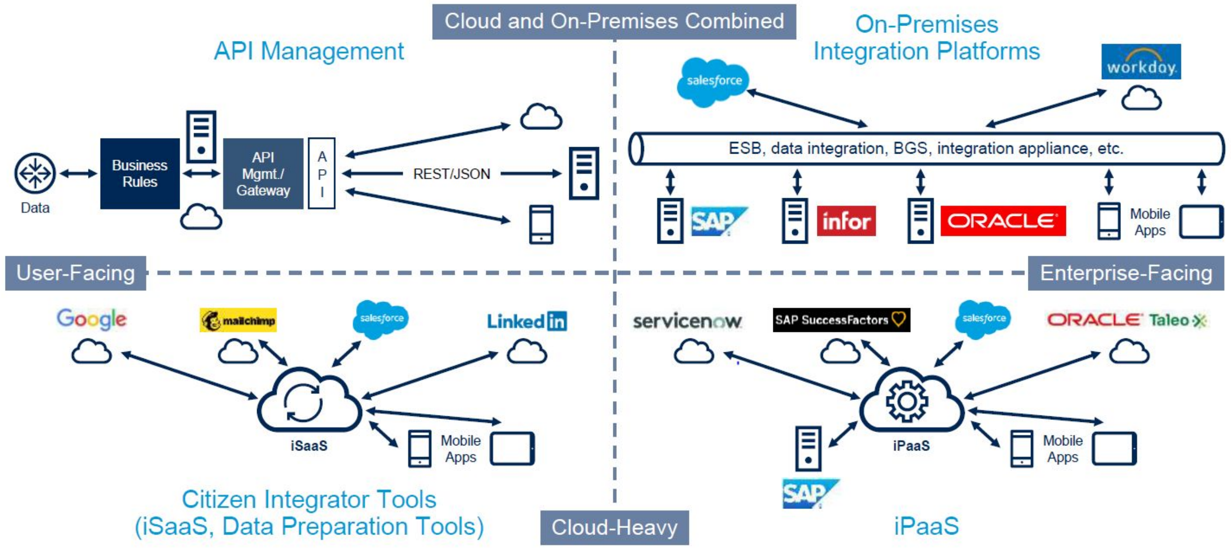
Dx

Ubiquitous, Cross-Channel – Personlised Cx enables Customers **choose your Business** and **stick with your Business**

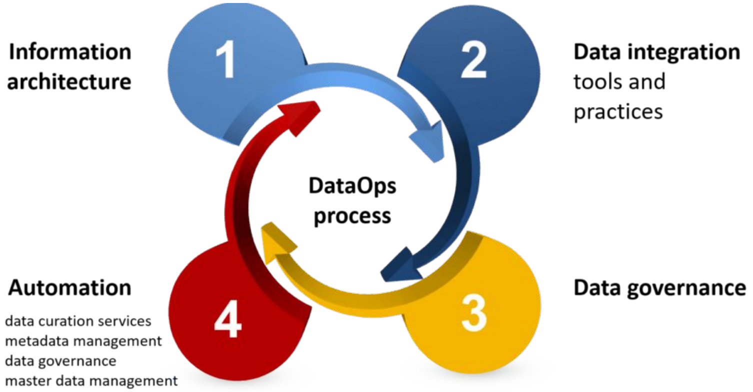


Cross Channel Journeys: Loan Application	Smartphone	Tablet	Desktop	Call Center	Branch
Check Rates	●	●	●		
Select Products		●	●		
Apply		●	●	●	
Get Help				●	●
Sign Contract					●
View Status		●	●		

Diversified Integration Technology, One size no longer fits all

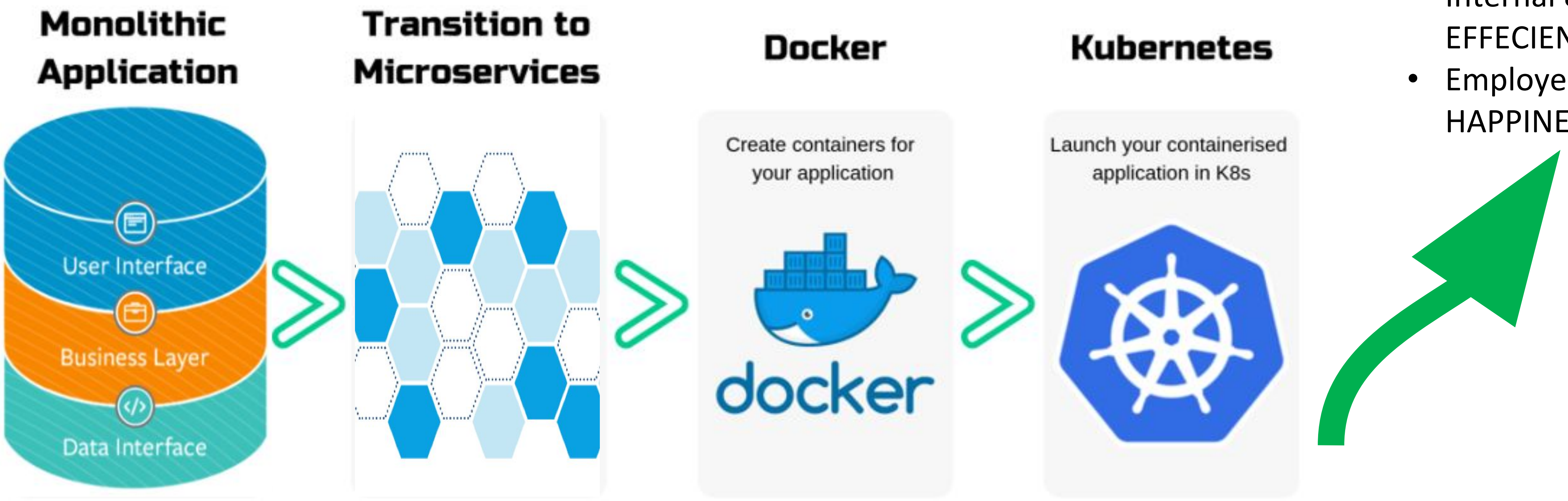


Data Insights



- data curation services
- metadata management
- data governance
- master data management
- self-service interaction

Granular Thinking



Monolithic Application

Transition to Microservices

Docker

Create containers for your application

Kubernetes

Launch your containerised application in K8s

- AGILITY whilst SCALING
- Superior Product PERFORMANCE
- Internal & External EFFECIENCIES
- Employee & Customer HAPPINESS

Key Manifestations of Dx (from an Engg perspective)

OMNI – MODULAR – OPEN – SMART

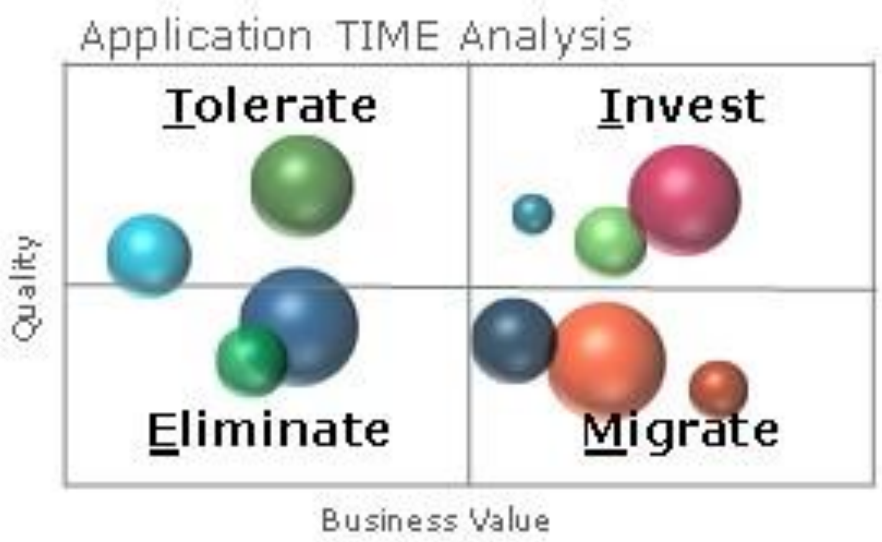
Project Lifecycle	→	**Product Lifecycle**
Software Development, Ops	→	**Agile & DevOps & SRE**
Data from being Diagnostic	→	**Predictive & Prescriptive**
IT Architecture	→	**Microservices**
Infrastructure	→	**Elastic**
Services	→	**Cloud-Native, hybrid**
VMs	→	**Container Orchestration**
Manual	→	**Manual is Evil**
Siloed Engineering efforts	→	**Platform Engineering**

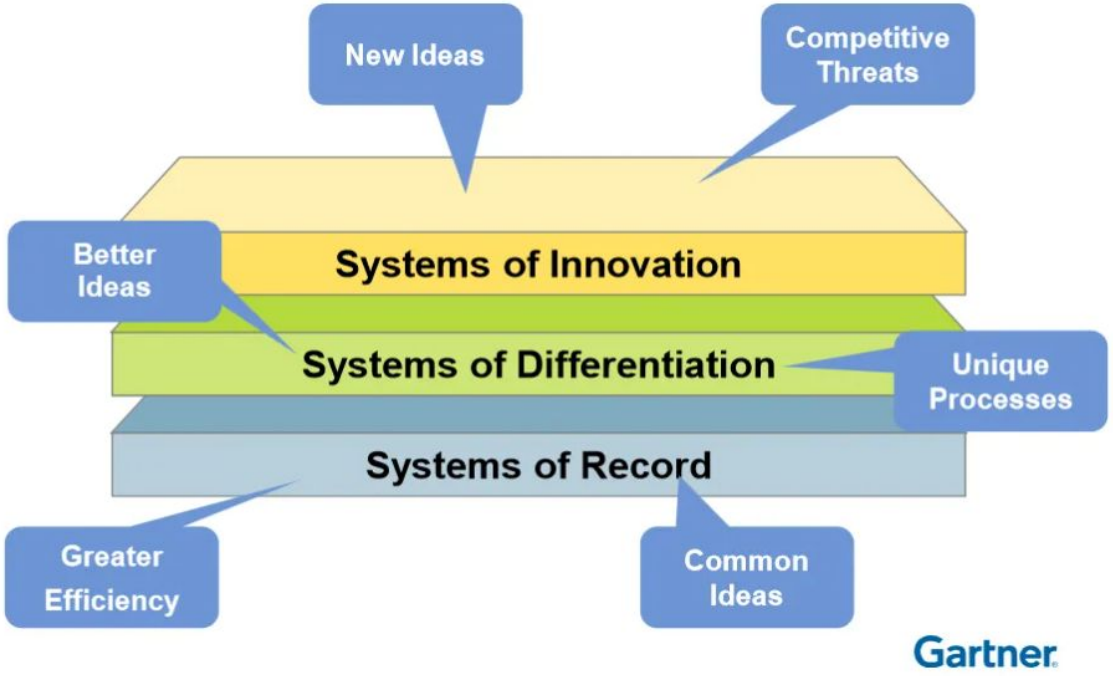
Where to start?

Amazon’s ‘Every 11 sec’

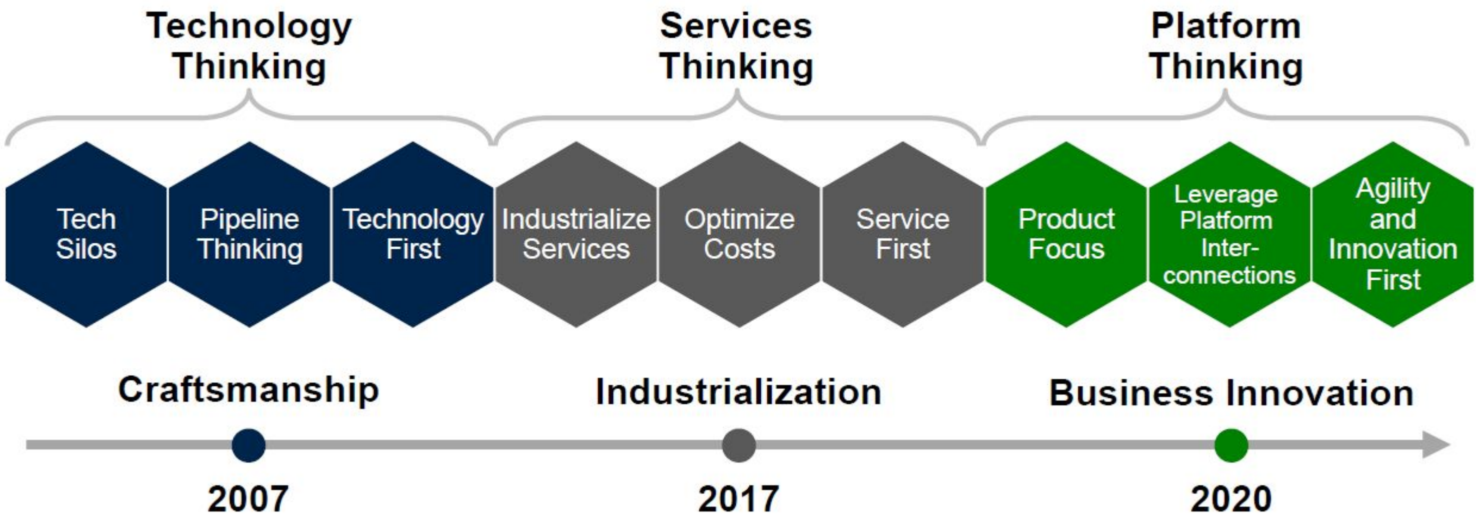
You are small and beautiful.. (Tamil proverb)

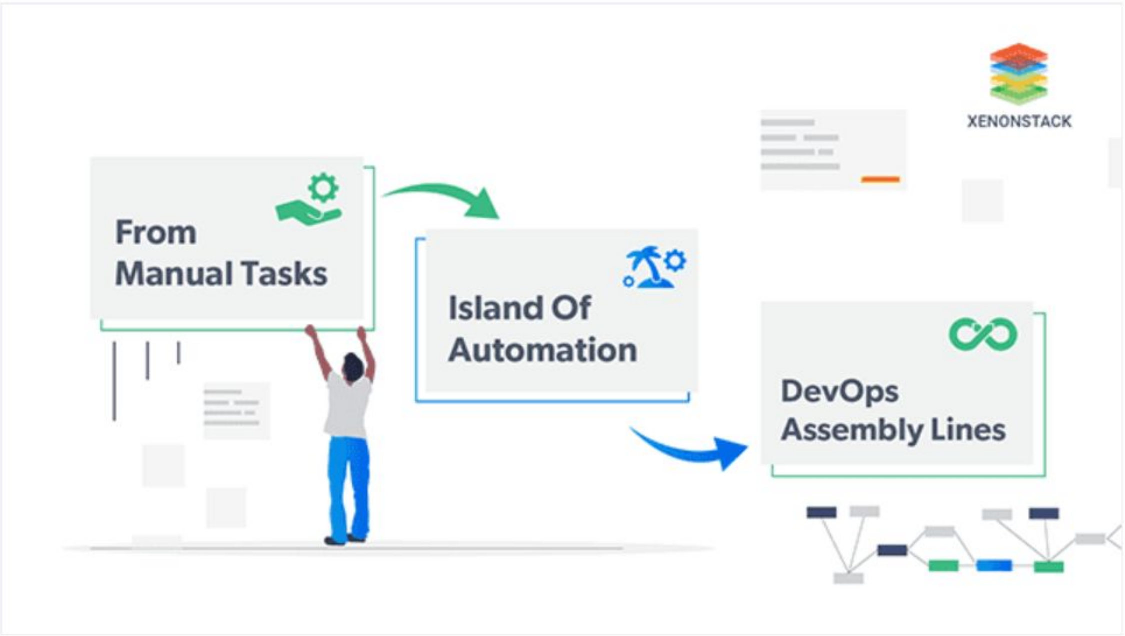
Review Product Portfolios





Have a Platform View on your Portfolio





- Forming a platform engineering team is one way an organization could begin **modernizing their engineering culture**

Criteria for your Deployment Strategy

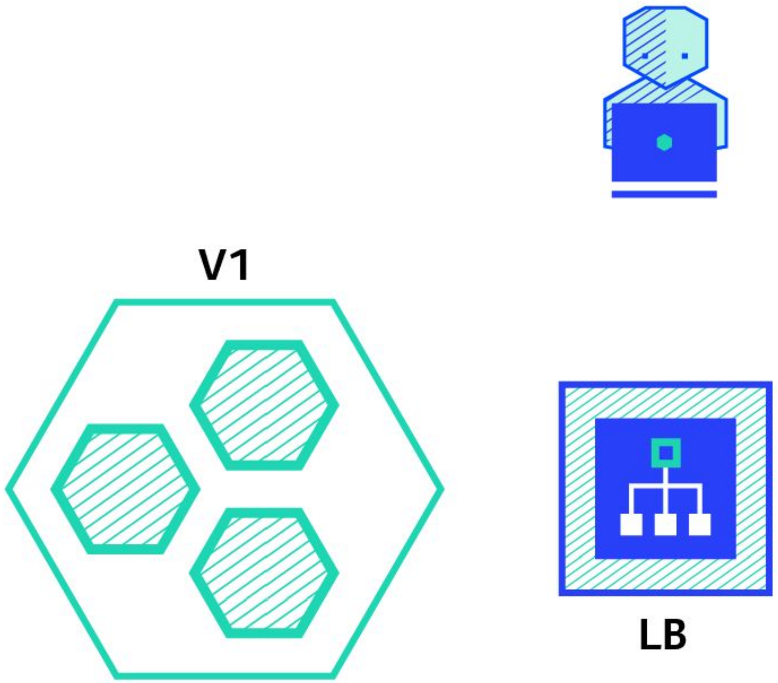
- Fault Tolerance
- High Availability
- Responsiveness/ Scalability
- Risk of Deployment
- Cost Effectiveness

✓ Long-running connections must be handled gracefully.
✓ Database conversions can be complex and must be done and rolled back along with the application.
✓ If the application is a hybrid of microservices and traditional components, additional care must be taken to achieve Zero-down time deployment

Deployment Strategies are always @Product level

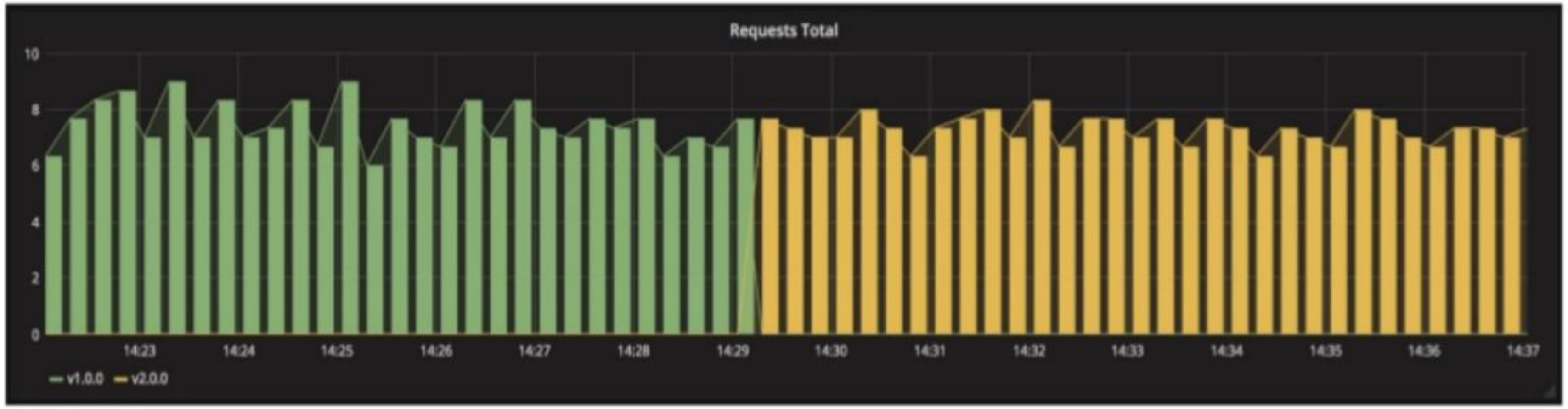
- **Blue/Green**: version B is released alongside version A, then the traffic is switched to version B
- **Canary**: version B is released to a subset of users, then proceed to a full rollout based on a % traffic
- **Rolling Update**: version B is rolled out replacing version A
- **Recreate**: version A is terminated then version B is rolled out
- **A/B Recreate**: version B is released to a subset of users under specific condition
- **Serverless**: application comes alive based on an event triggered by traffic

Blue-Green



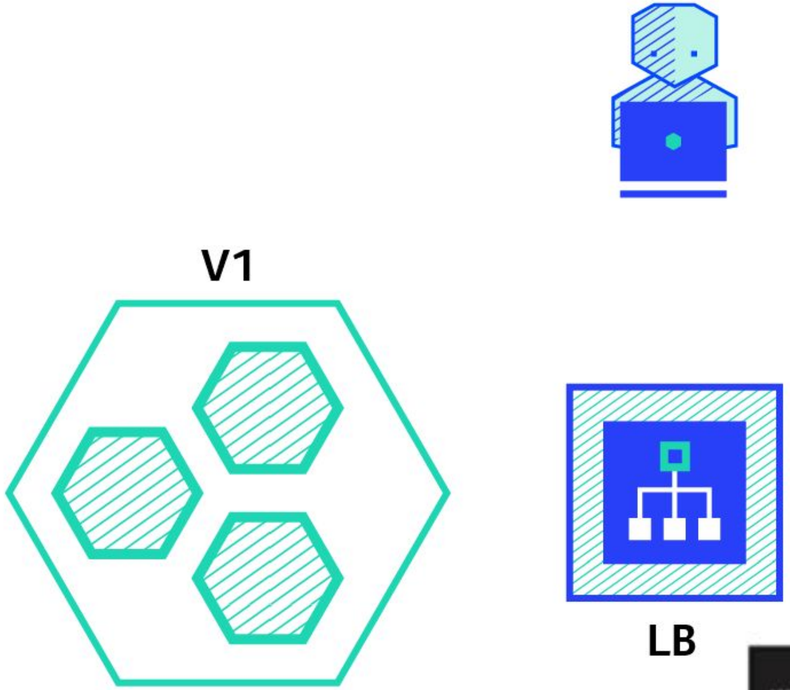
- The new version of an application is deployed in an identical environment and a router is used to select the application version which will be exposed to the users.

- Instant rollout/rollback.
- Zero downtime approach, because the switch is almost instantaneous (which is close to ideal), causing users not to notice when their request was served by the new environment.
- Expensive as it requires double the resource

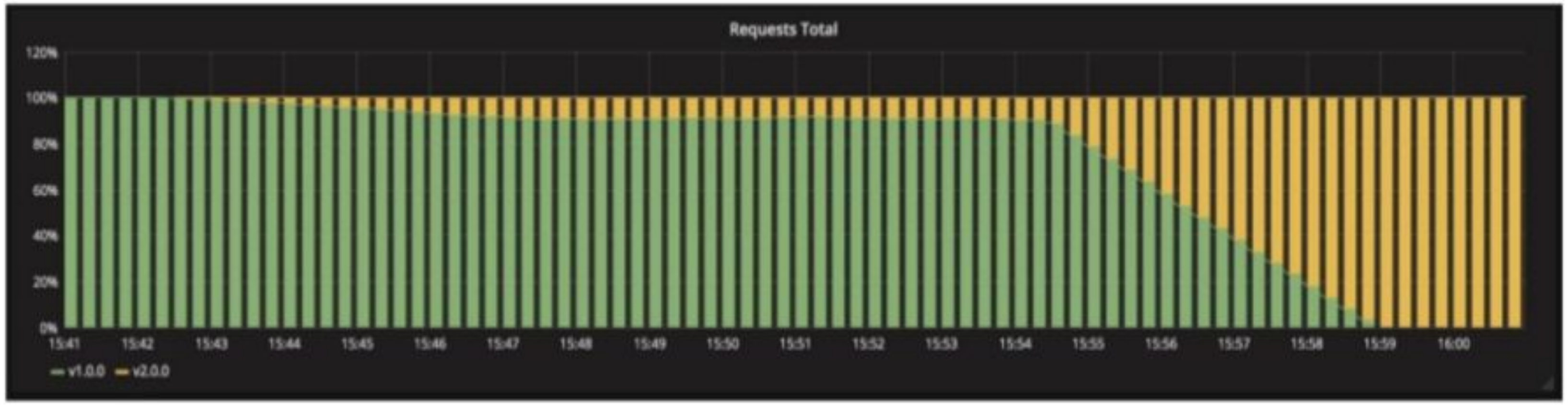


Canary

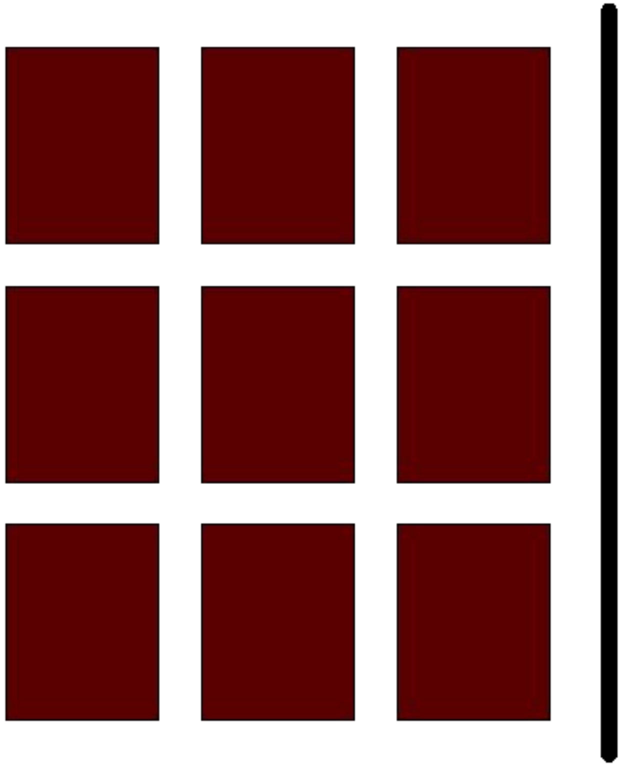
- A canary deployment consists of gradually shifting production traffic from version A to version B.
- The newer are rolled out to a smaller group of users initially to minimize risk, detect problems, or weed out regression issues.



- Version is released for a subset of users.
- Convenient for error rate and performance monitoring.
- Fast rollback
- N-1 compatibility is required

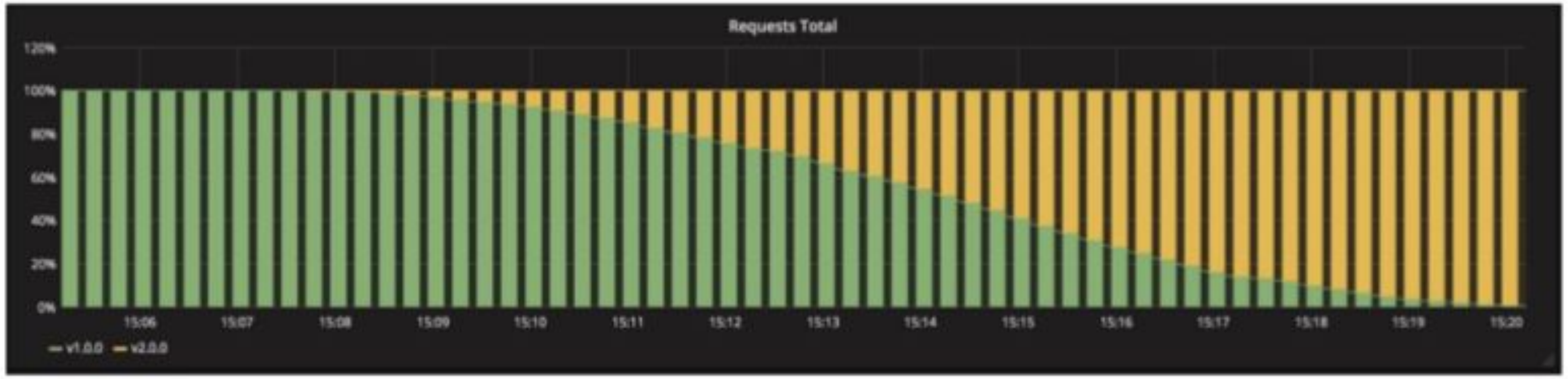


Rolling upgrade

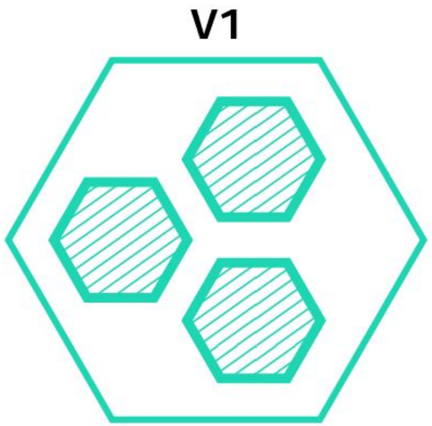


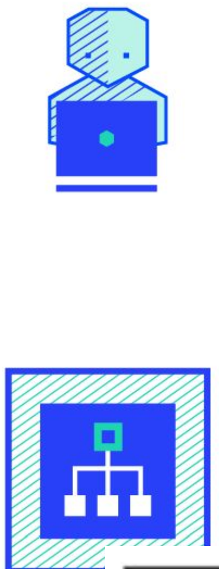
- Both old and new versions of your code run for some time, hence requires that your application handle **N-1 compatibility**
- Waits for new pods to become ready before scaling down the production pods.
- At any point in time we have exactly N+1 instance running.

- Old pod is removed only when the new pod passes health checks
- Max surge: How many instances to add in addition of the current amount.
- Max unavailable: Number of unavailable instances during the rolling update procedure.



Recreate

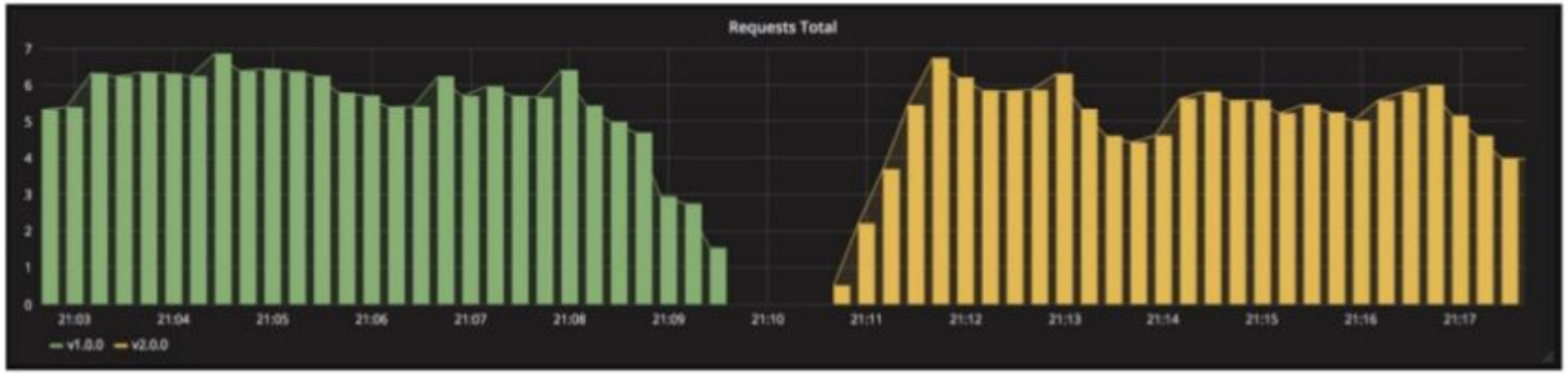




- When you do not support having new and old versions of your application code running at the same time.

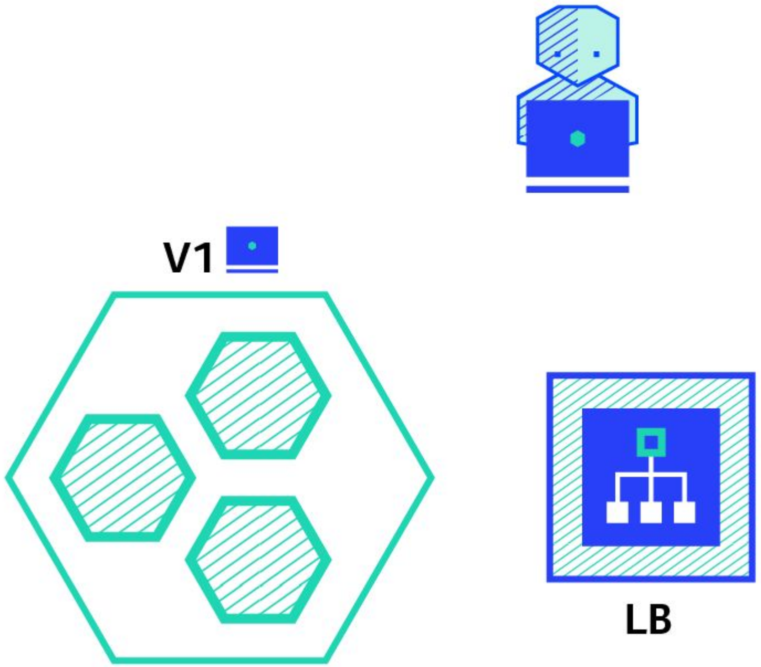
- When you want to use a RWO volume, which is not supported being shared between multiple replicas.

- Downtime impact on users arising from shutdown time + promote time + boot time



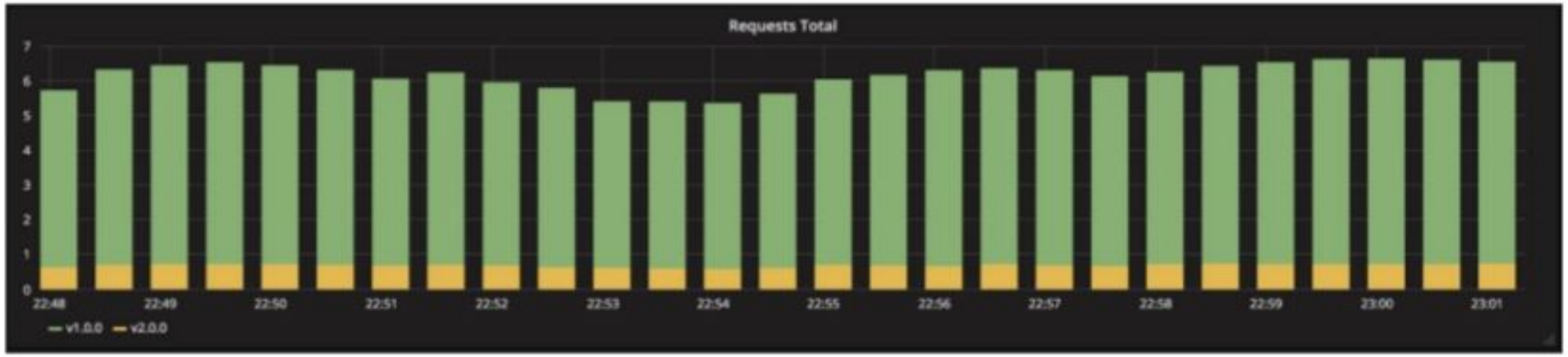
Service unavailable

A/B upgrades

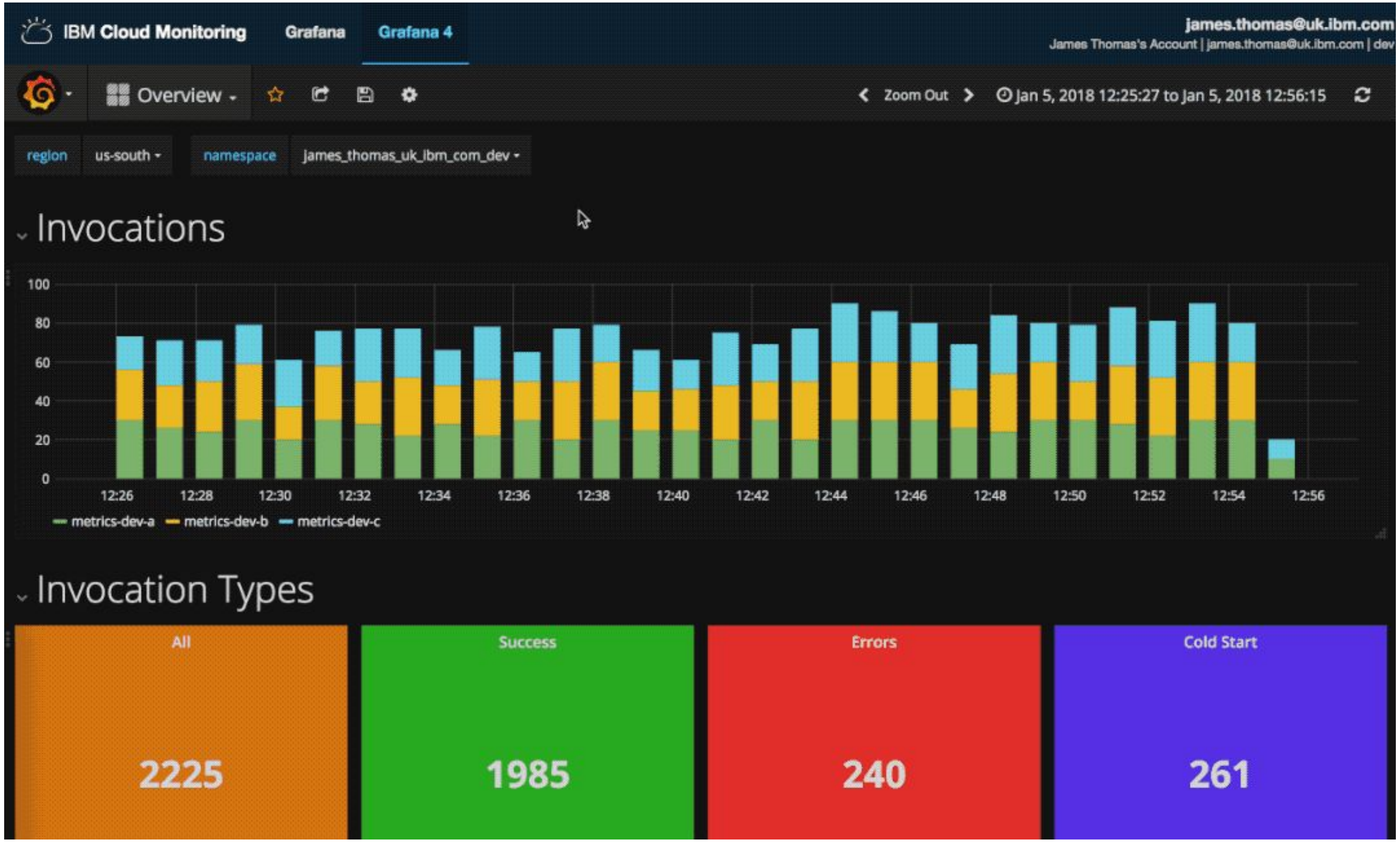


- A/B testing deployments consists of routing a subset of users to a new functionality under specific condition based on a Business criteria

- The switch of traffic could be based upon Geolocation, Language, a certain query criteria, based on Technology support such as browser version or OS



Serverless



region: us-south

namespace: james_thomas_uk_ibm_com_dev

All	Success	Errors	Cold Start
2225	1985	240	261

- Developers focus on solving core business problems with independently built and deployed functions that react to an event
- Cost effective
- Automatically scale up based on event-triggers in response to incoming demand, and is then able to scale to zero after use.

- Triggers that start and scale containers, and scale them back to zero when not in use.
- Initial set-up time (latency) per invocation needs to be considered

Summary

Deployment strategies which would have been difficult to set up and implement on-premises configurations and application setups have been made easier using the **new container** technologies.

No one-size fits all approach. Understanding the approach and exploring alternative options is good.

Developers and operations teams work closely together when picking the right approach for the application.

Getting a handle on **deployment strategies** will ensure continuous delivery and manage the risks of introducing new features in a controlled manner