Case Study: Future Financial

- Cloud-Native Adoption

July 2022

Rahul Kumar Singh

Scenario

1. Future Financial is a multi-national organization specializing in online loans. They have traditionally been a Microsoft house, with most applications developed in-house using .net and deployed to VMware virtual machines.

2. Recently all existing applications have been lifted and shifted from an on-prem VMware environment to VMware on the public cloud. Public cloud is also now the default for all new development.

3. The team applications current team is 150 people, made up of architects, developers, and testers. A third party had managed on-prem infrastructure and operations, and there was a "throw it over the wall" relationship between the applications team and operations.

4. Although there were on-premises architecture skills in the team, these have not translated well to the public cloud.

5. There are few people in the group with public cloud experience.

6. Future financial are keen to learn how to make the best use of the public cloud both to reduce costs and speed up their application development.

7. They are also would like to explore and possibly leverage the big data tools available.

8. They want to create new cloud-native applications and modernize existing applications to be cloud-native.

9. Their operating model should make the most of the public cloud.

10. To date, there has been some early success in putting CI/CD pipelines in place for a few applications.

11. There is a limited cloud center of excellence focused on cost optimization.

12. Teams mainly interact by raising tickets to request new infrastructure. There is a frustration that working with the public cloud takes longer than when the applications were on-prem.

13. Future financial would like to move to feature teams where each team is self-sufficient enough to build and run applications and take care of cross-cutting concerns such as security.

14. They are keen to reskill both their existing people, make new hires, and seek advice on the most helpful skill profiles and structure the teams.

Table Of Contents

1. Challenges Highlights

2. Current Maturity Assessment
   - How would you access my current maturity?

3. Applications Modernization Options
   - What options do I have to modernize my applications?

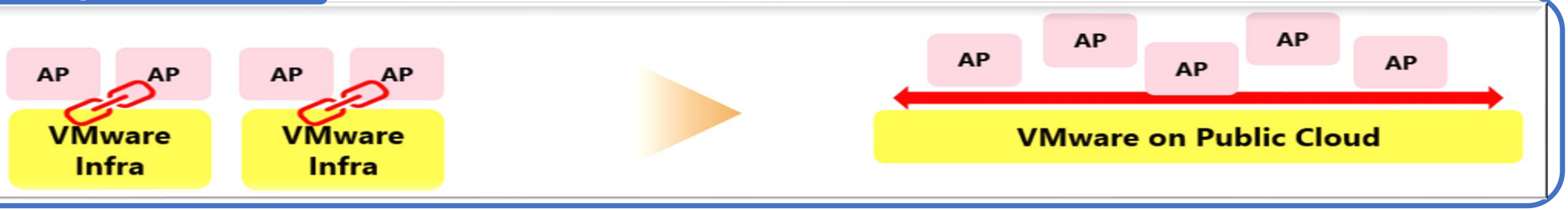
4. Operating Model
   - What options do I have to update my operating model?

5. Team Structure & Skills To Achieve
   - What are the options to structure my teams, and what skills should I build first?

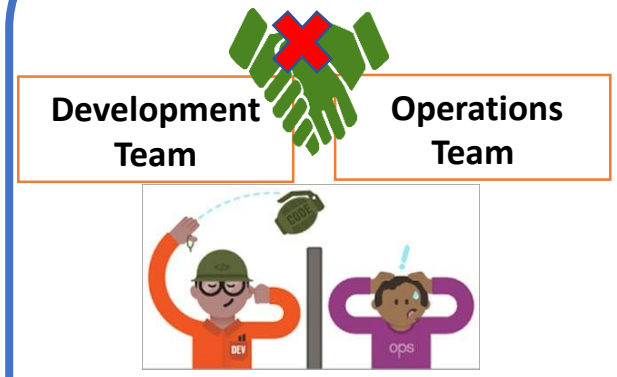
6. Current & Future Roadmap
   - What should I do first and what do you suggest are the milestones on the journey?

Future Financial – Challenges Highlights

Background Context



Work Model



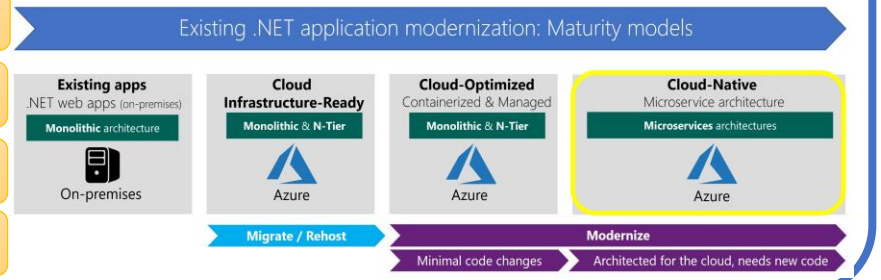
- Working in Silos, Lack of collaboration across teams
- Dev Team Lack of Ops Expertise
- Less Members having Existing Infra knowledge

Public Cloud Skills

- Absence Of Public Cloud CoE
- Lack of skills in Cloud Tech. Stack
- E2E DevOps Setup
- Reskill Existing Team
- Establish A Feature Team
- New Cloud People Hiring
- Team Restructuring
- Cloud Consultancy

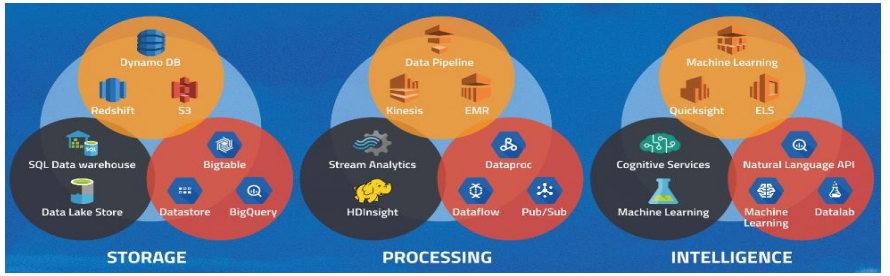
DX Modernization & DevOps Challenges

- DX Strategy
- Rearchitect
- Rehost(AP, Infra)
- Rewrite + Rebuild



- Data Exchange
- Data Analytics
- Cold & Hot Storage
- Reliability
- Data Security
- Competency

Big Data Challenge On Public Cloud



Current Maturity Assessment

Achieved

To Be Achieved

Maturity Level **Dimensions**	Level 0	Level 1	Level 2	Level 3	Level 4
Infrastructure	On Premise VMs on Private Data Centers	Cloud-ready	Move To Cloud VMC on Public Cloud	Cloud-optimized	Cloud-native
Public Cloud Skills	Basics Of IaaS	IaaS, CDN, IAM	IaaS, PaaS, Storage, Caches, Managed SVCs, DBs, SDN, VPC	ConfigMgt, APIs FaaS, Serverless Security, RBAC, DC|DR	IPaaS, IaC, Automation AI Compliance
Application Development	Monolithic Proprietary ACID transactions	N-tier Portable Virtualized	Monolithic on VMC on Public Cloud	Loosely coupled Relational 'Microservices' as Container	Cloud Native API Services Containers NoSQL BASE transactions
DevOps	Separate Operation Team	Basic CI-CD	E2E Pipelines Dev, Stage, Prod.	Security & Smart Ops	AI Driven Ops
Model	Iterative	Agile	DevOps	DevSecOps	Performance

Applications Modernization Options

Approach

Evaluate The Legacy System of Future Financial

Evaluation Drivers:

- Business Fit
- Business Value
- Security
- Complexity
- Scalability
- Availability

Evaluate The Modernization

Encapsulate As APIs → Rehosts → Replatform → Refactor → Rearchitect → Rebuild → Replace

Options

Public Cloud & Containers

aws, Amazon EKS, AKS, Azure, VMware Tanzu, OPENSHIFT

Public Cloud

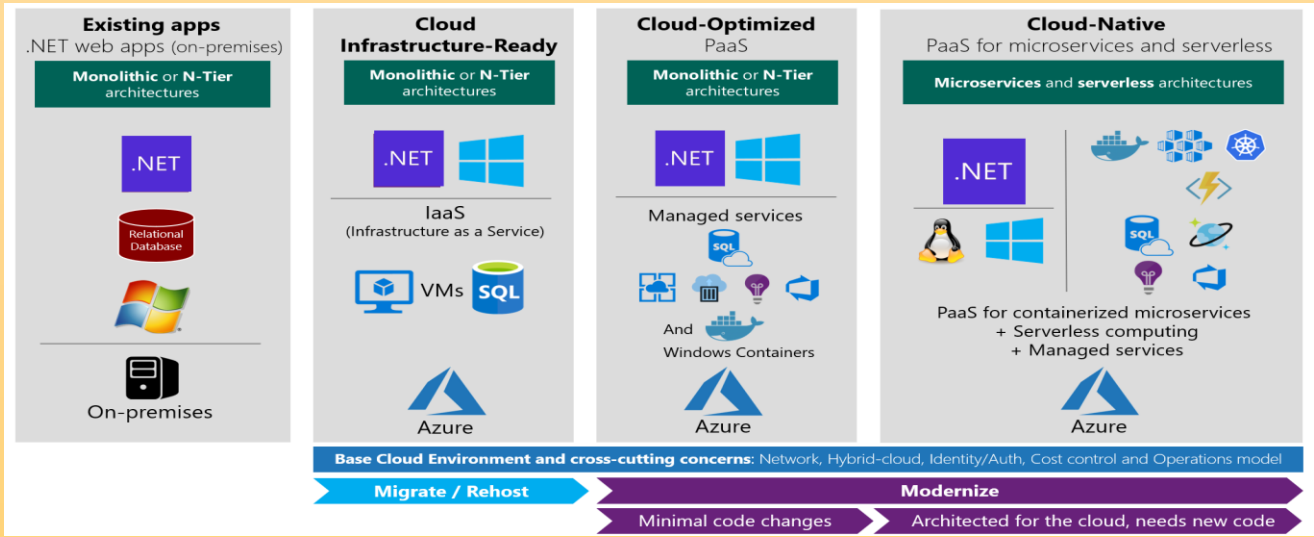
Managed Kubernetes

amazon RDS, Amazon EBS, Amazon S3

Data Storage & Big Data

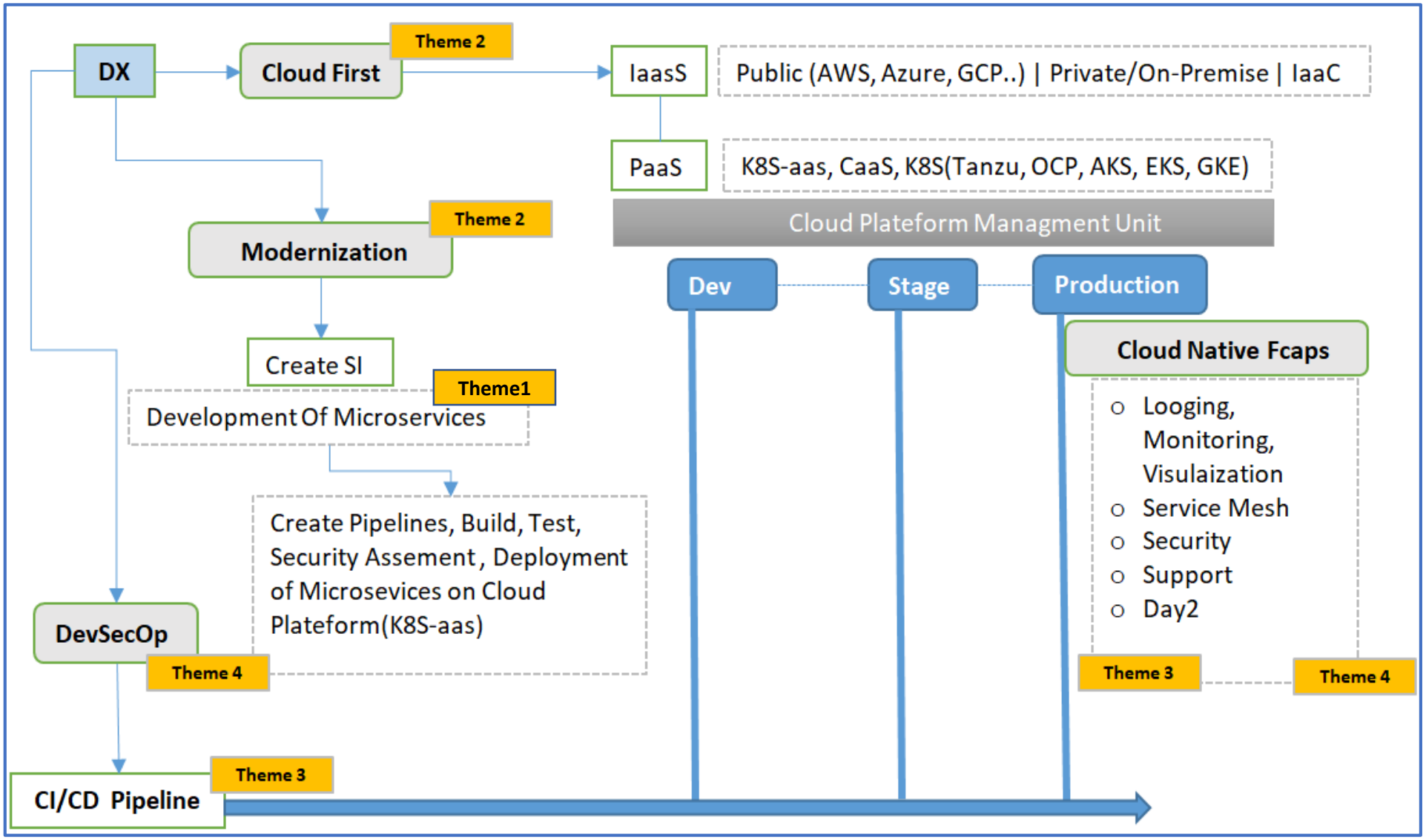
- Cloud Design Consultancy
- Concept Planning
- Re-Architecting
- Solution Design
- System Integration
- Cloud Migration
- Orchestration

Modernization (Example with MS Azure)

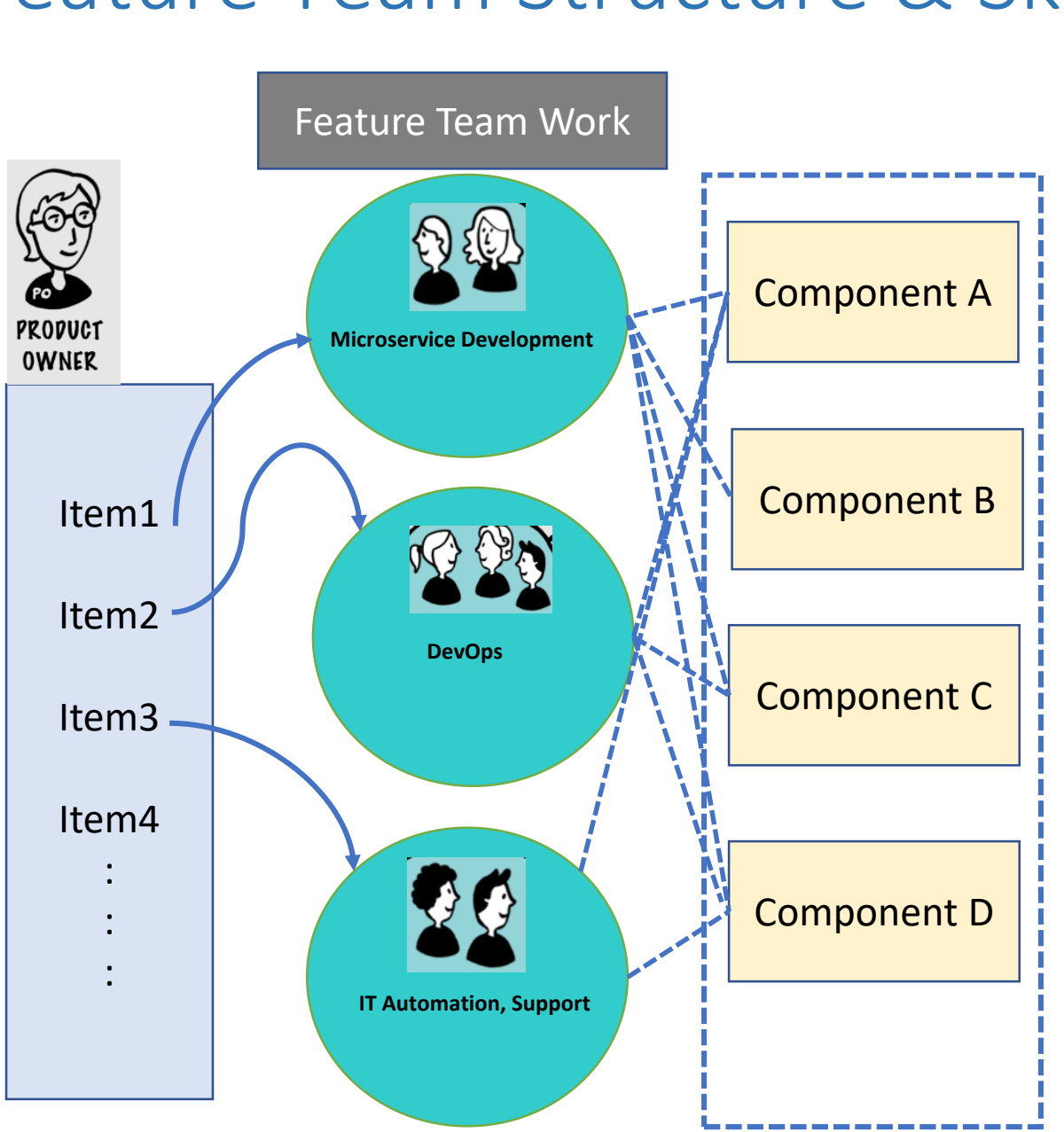


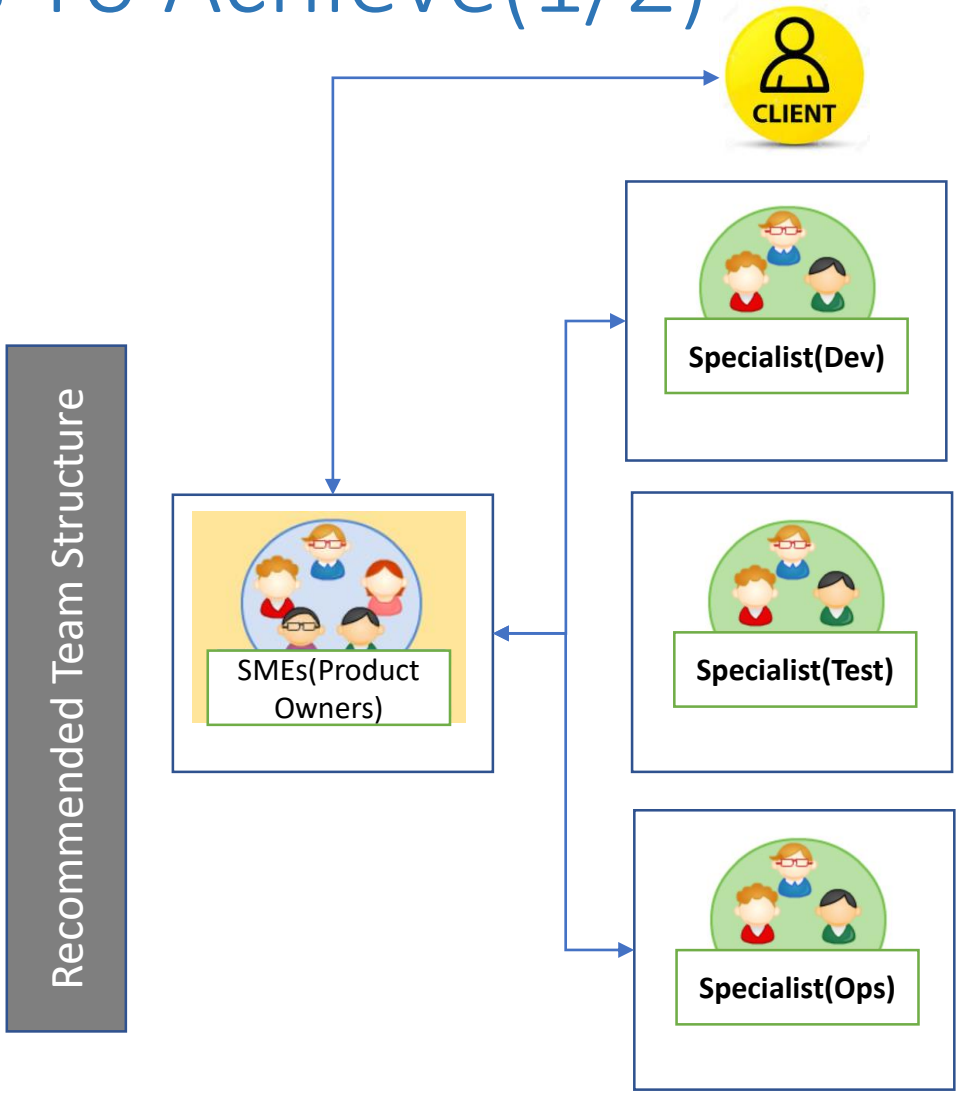
Operating Model

Theme	Areas
Theme1	Development
Theme2	Cloud CoE, Modernization Establishment
Theme3	OSS Bases Tools Frameworks for CICD Or Production Support
Theme4	DevOps, IaC, Infra Automation, L1,L3,L3 Support



Feature Team Structure & Skills To Achieve(1/2)

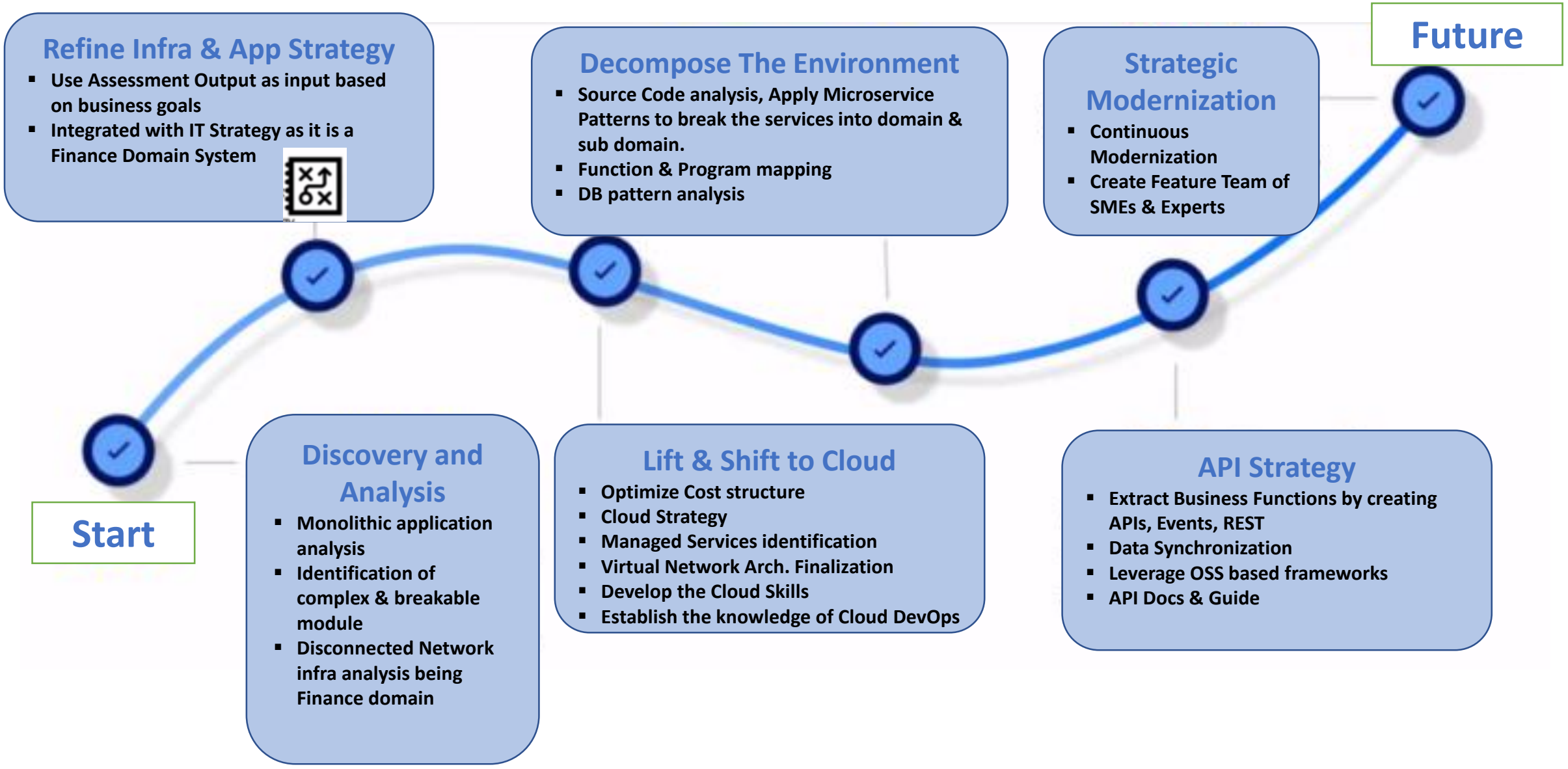




Feature Team Structure & Skills To Achieve(2/2)

Sl. No	Key Areas	Expertise
1	Public Cloud Platform (OS, PaaS, IaaS)	CoreOS, RHEL, CNCF Kubernetes, RedHat OpenShift, RedHat OpenStack, VMware vSphere, AWS, Azure, GCP
2	Container Platforms	Kubernetes, OpenShift, AKS, EKS, VMware Tanzu, Rancher, GKE
2	DevOps	▪ Repo Manager (GitLab, Git), Hook, Trigger ▪ Build (Maven, Ant, Fabricate) ▪ Image Build(Docker build, Docker Repo) ▪ Integration test(Junit, Mockito) ▪ Ansible ▪ Terraform* ▪ OCP Cluster deployment & provision via Jenkins
4	Testing, Evaluation, Support	▪ Feature Testing(User & Admin POV) ▪ Test Automation(UI, CLI) using Jenkins, Selenium, Robot framework. ▪ Performance Testing(CPU, Memory, Database Sizing activity) ▪ Production Support from SIer POV(L1, L2) ▪ Patch Development(L3 Support)
5	Coding & Scripting	▪ Programming: Go, Python, Java ▪ Scripting: Bash ▪ Configuration: Ansible
6	Logging & Monitoring	▪ Grafana-Loki Stack, ELK Stack ▪ Log Collector(Fluentd, Vector, FluentBit) ▪ Alerting(Tempo) ▪ Monitoring(Prometheus) ▪ Monolithic, Microservice Deployment ▪ Scalability Factors(Scale-in, Scale-out)
	Database, Storage	▪ MySQL, MongoDB, MariaDB, Cassandra, Bolt DB Shipper ▪ Object Storage(AWS-S3, MINIO), Azure Blob ▪ Block Storage: AWS-EBS ▪ Ceph

Current & Future Roadmap



Thank You