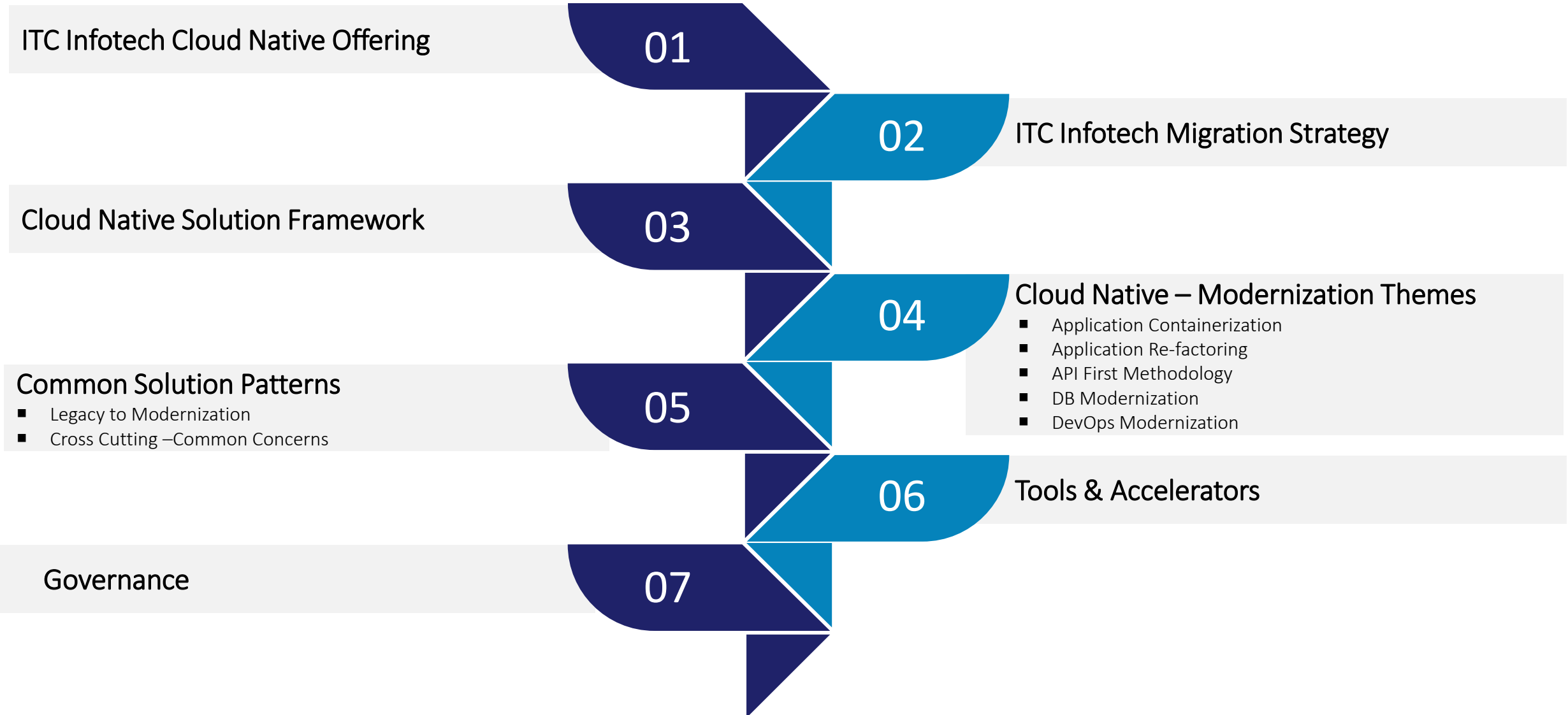


ITC Infotech Cloud Native Solution Framework

APRIL 2022

Topics





ITC Infotech Cloud Native Offerings

ITC Infotech - Service Offerings and Frameworks

Strategy & Consulting



- Cloud Native application strategy, planning and implementation approach

Maturity Assessment & Roadmap



- Cloud and Cloud native capabilities assessment
- Adoption Plan, Reference architecture blueprints

Migration Services



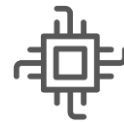
- Application migration on Cloud native capabilities
- Enhance with Cloud containerization and DevOps

Engineering



- Application Modernization with cloud native technologies
- API and Microservices based design and development on cloud
- Serverless architecture services development

Technology



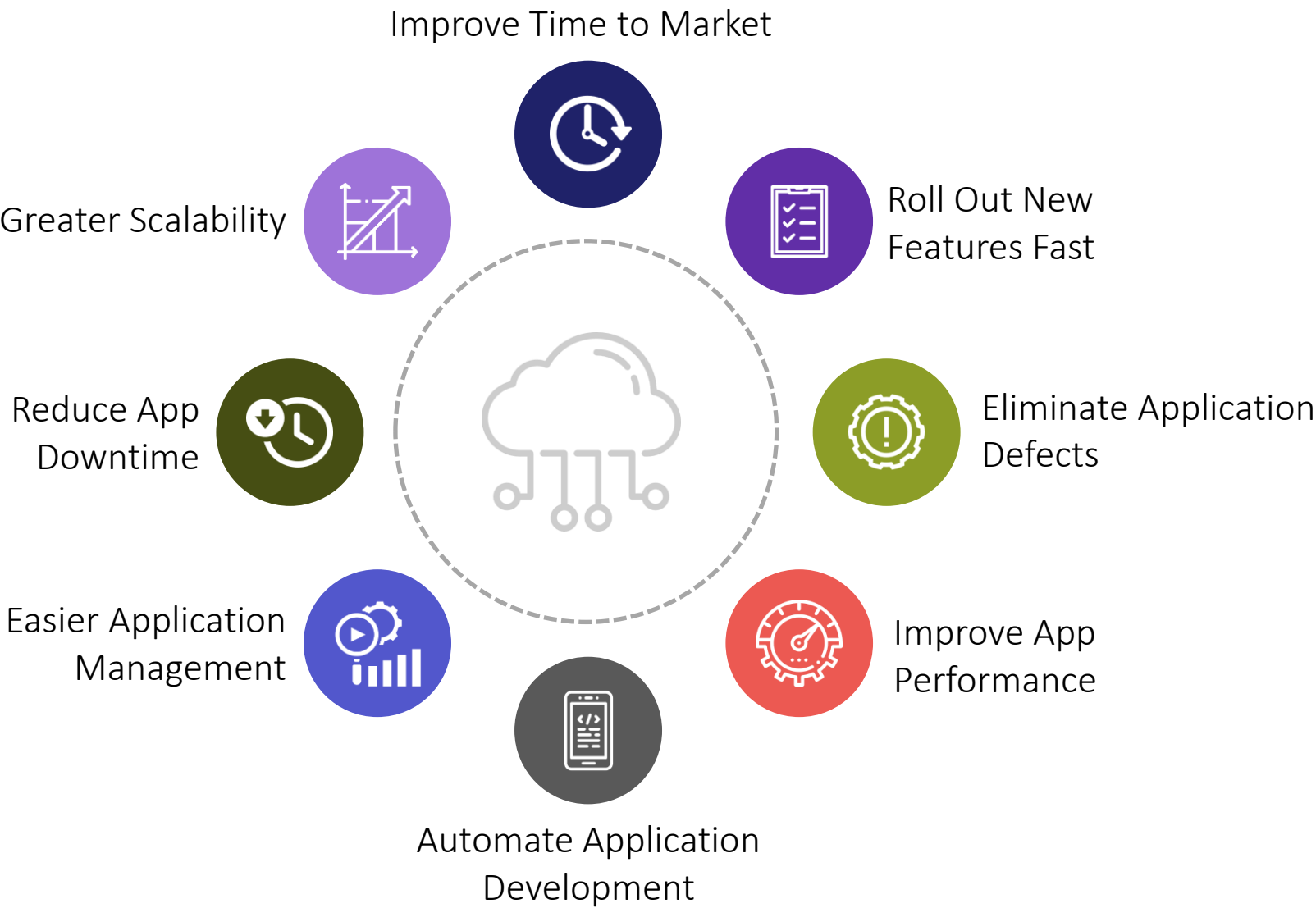
- Azure Cloud native experience
- Large Scale Implementation experience
- API Led Digital transformation and Digital Integration Experience

Value Adds & Accelerators



- Implementation Accelerators reduce overall Implementation Effort
 - Microservices Assessment framework
 - Integrated Development & Deployment framework

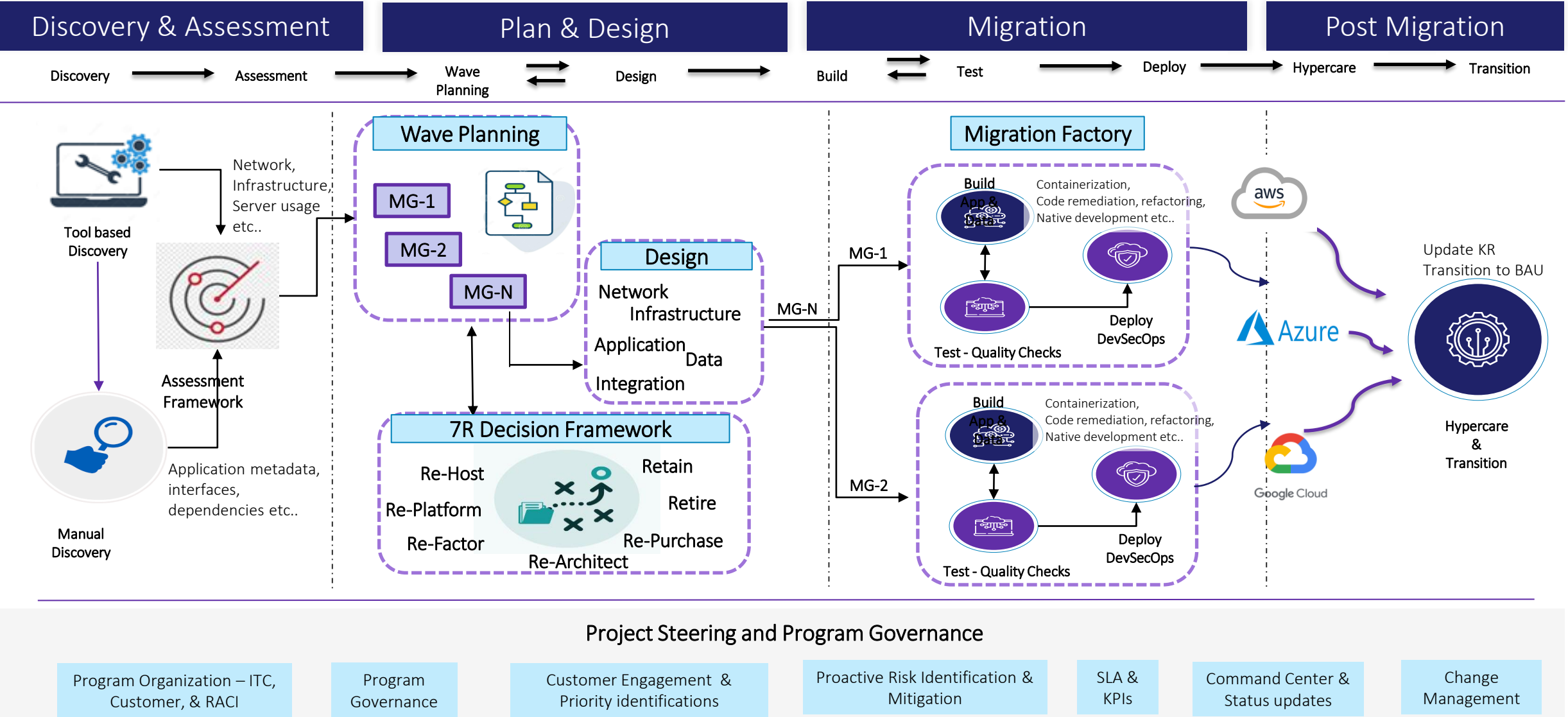
Our Cloud Native Solution - Benefits





ITC Infotech Migration Strategy

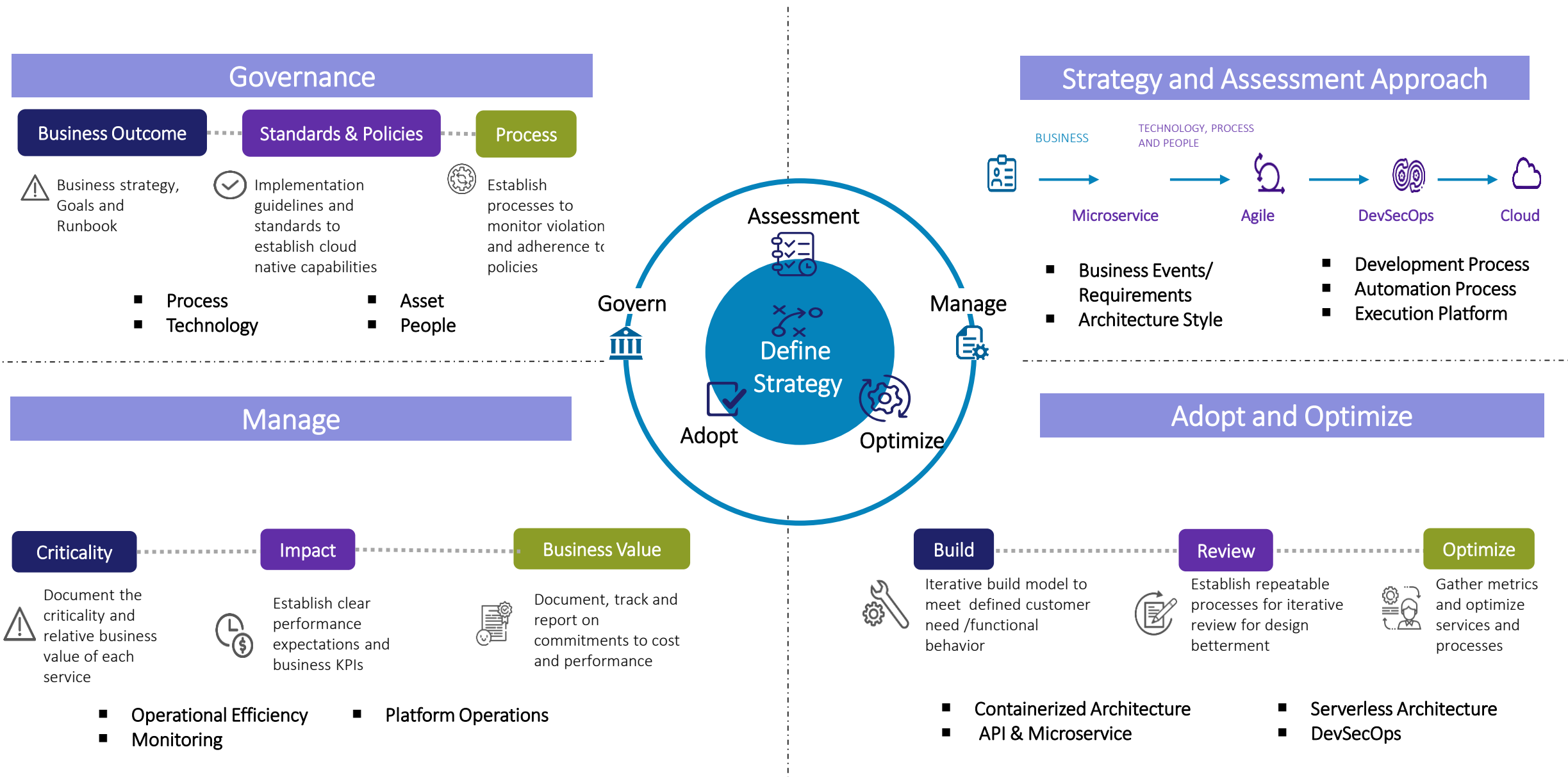
Migration Methodology - Phased Approach





Our Cloud Native Solution Framework

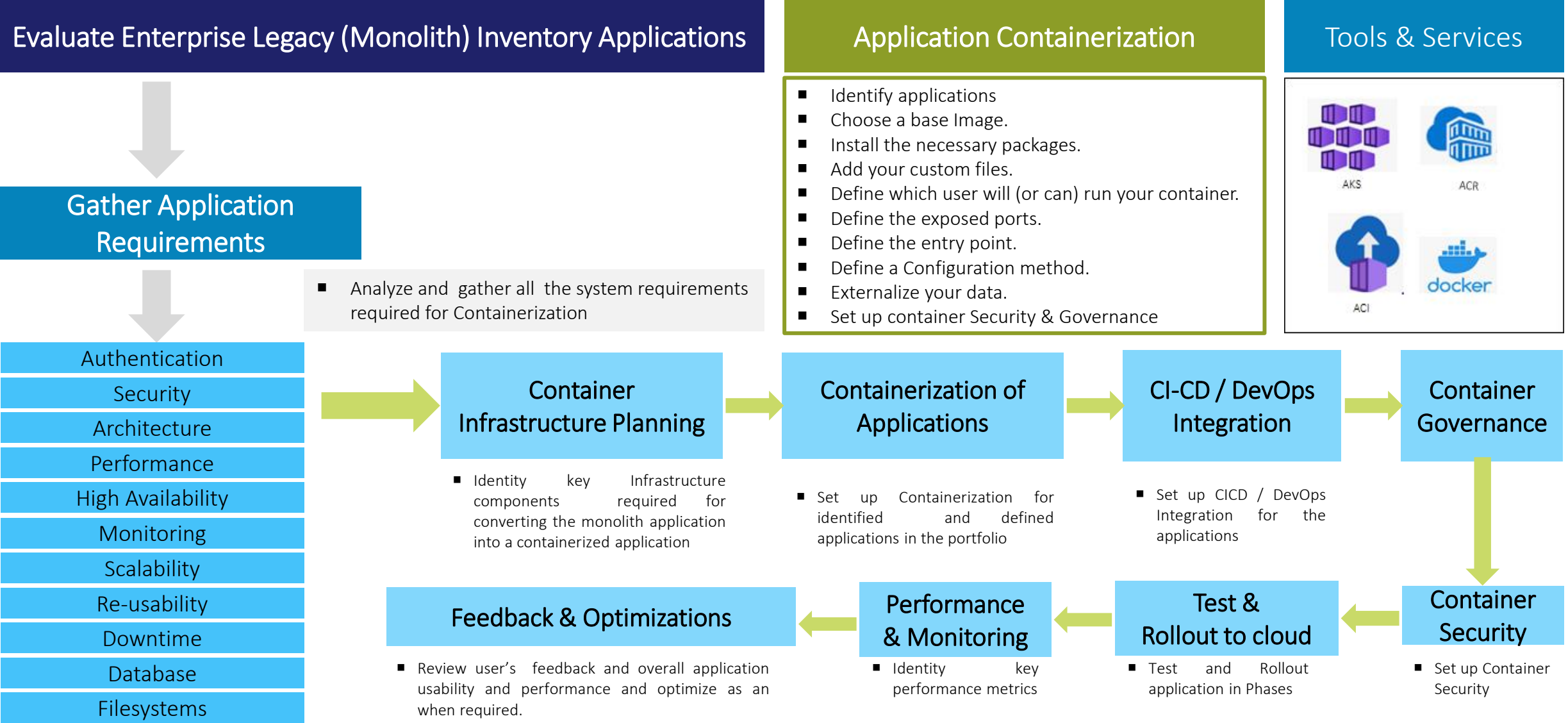
Cloud Native Solution Framework



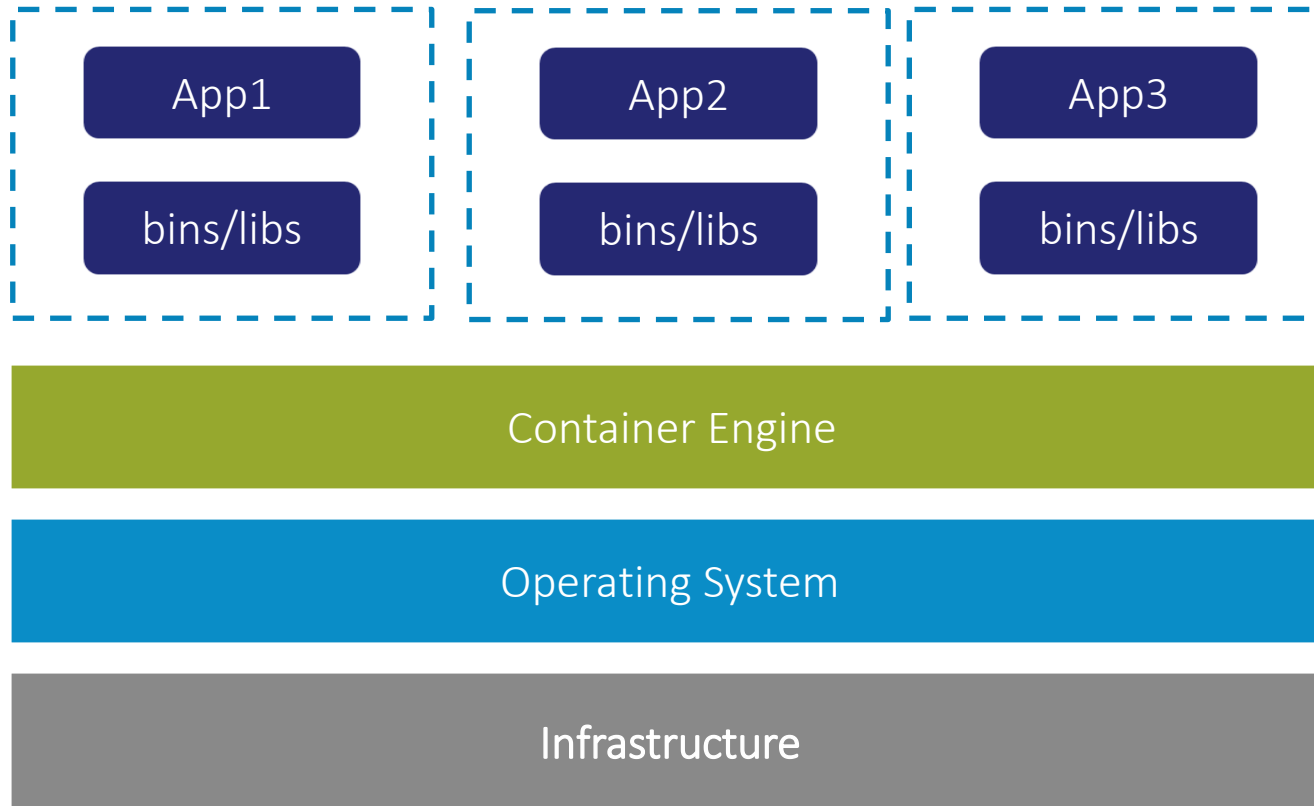


Cloud Native - Modernization Themes

Modernization Themes – Application Containerization

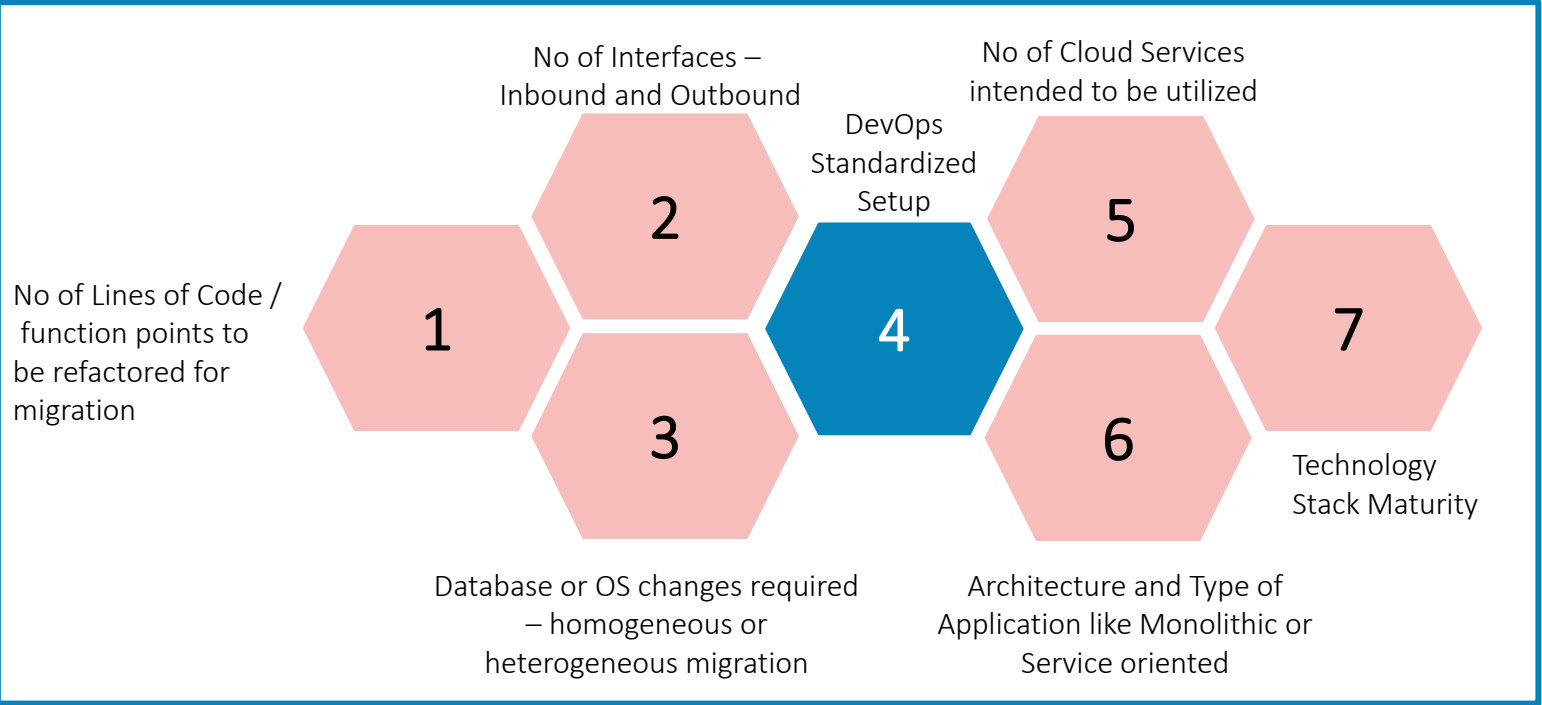


Modernization Themes – Application Containerization Contd.



- Package once , deploy everywhere with less risk.
- Portability
- Supports Infrastructure-as-a-Code principles
- DEV/PROD environment parity from desktop to production environments
- Better community support including Docker
- Supports scalable workload
- Provides application and process isolation
- Better Performance
- Ease of deployment
- More control on the technology stack
- Prefer more serverless architecture over infrastructure

Modernization Themes – Application Refactoring



Application Refactoring

- Identify key enterprise apps
- Define Application parameters like authentication , security , performance etc.
- Identify the lines of code and touch points for refactoring
- Define no of Interfaces – Inbound and Outbound
- Setup CI/CD process
- Define well defined Architecture
- Leverage serverless cloud platform
- Continuous Health Monitoring
- Automated Alerts and Notifications

Tools & Services



Azure WebApp



Functions



Service Bus

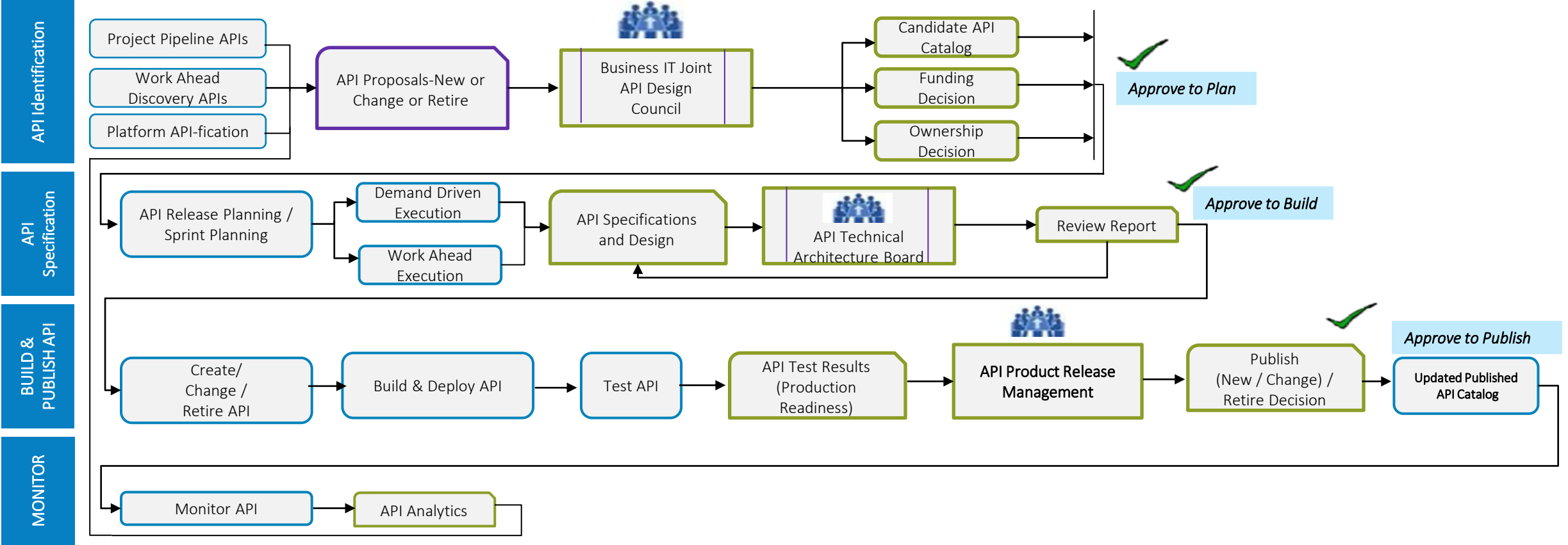


Logic Apps

Considerations

Application Architecture ■ No Existing Webservices ■ Existing Macro Services ■ REST or JMS ■ SOAP or EJB ■ Servlet / JSP ■ JSF / Spring Portlet	Data Architecture ■ Master Data Management ■ Reference Data ■ Flat Object Structure ■ Independent Tables ■ Blob Storage	Deployment ■ EARs Packaging ■ Containerization ■ Container Orchestration ■ Infra Pipeline – IaC ■ Application Pipeline	Re-factoring ■ Macro Service to Micro Services ■ Serverless ■ App Component Dependencies ■ Flatten & Refactor Components ■ Component Grouping ■ API Proxy & Facade	Cross-cutting Concerns ■ AuthN & AuthZ ■ Service Mesh ■ Shared services/domain model ■ Log Aggregation & Tracing ■ Service Monitoring ■ NFRs
--	--	---	---	---

Modernization Themes - API First Strategy



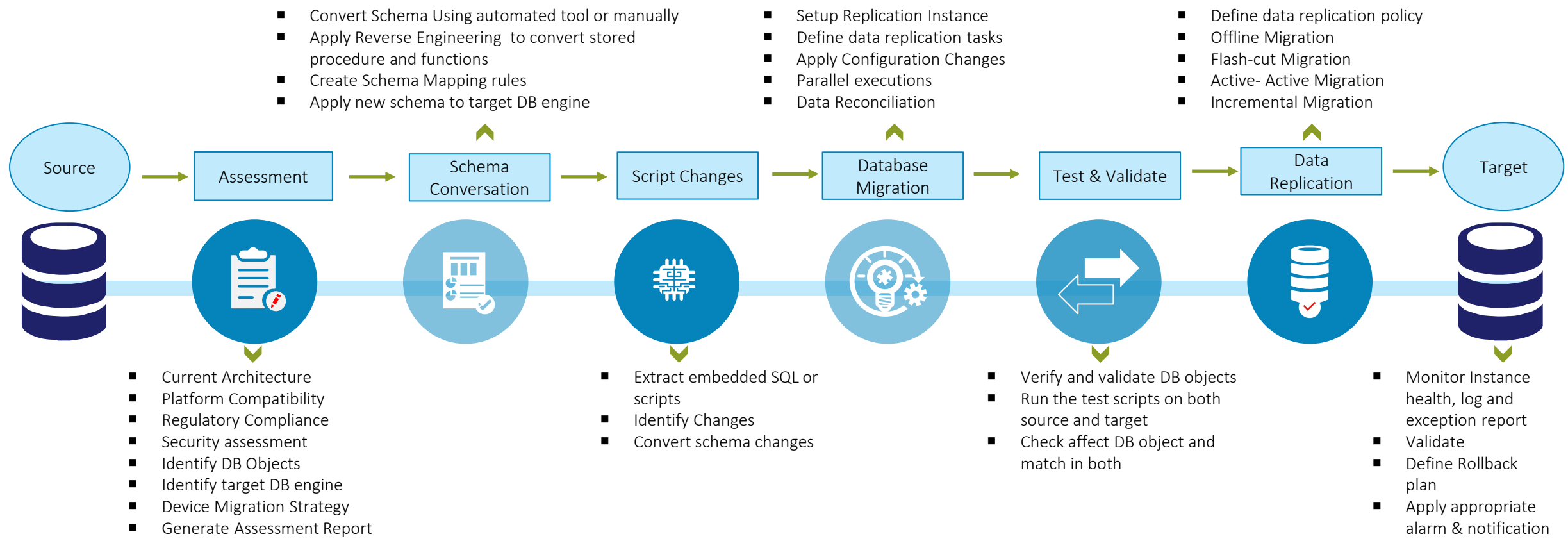
Considerations

- | | | | |
|--|--|--|--|
| <ul style="list-style-type: none">■ Unlocking legacy assets■ API-centric app integration■ Multi-channel user experiences■ B2B integration | <ul style="list-style-type: none">■ API Developer Portal■ API Discovery■ API Consumption■ Scaling | <ul style="list-style-type: none">■ API Gateway■ Subscription & Token Auth■ Caching■ CI/CD - DevOps | <ul style="list-style-type: none">■ Integrations■ Security■ Monitoring■ Analytics |
|--|--|--|--|

Tools & Services

 API Management	 Azure Functions	 API Gateway
---	--	--

Modernization Themes – DB Modernization



Examples

Homogenous Oracle 8x/9x/11x → Oracle 19c MySQL 5.x → MySQL 8	Heterogenous Oracle 12 → MSSQL MySQL → Postgres	Cloud PaaS Oracle → Azure SQL MS SQL → Azure SQL
---	--	---

Tools & Services

Azure DMS

ADF

Azure SQL

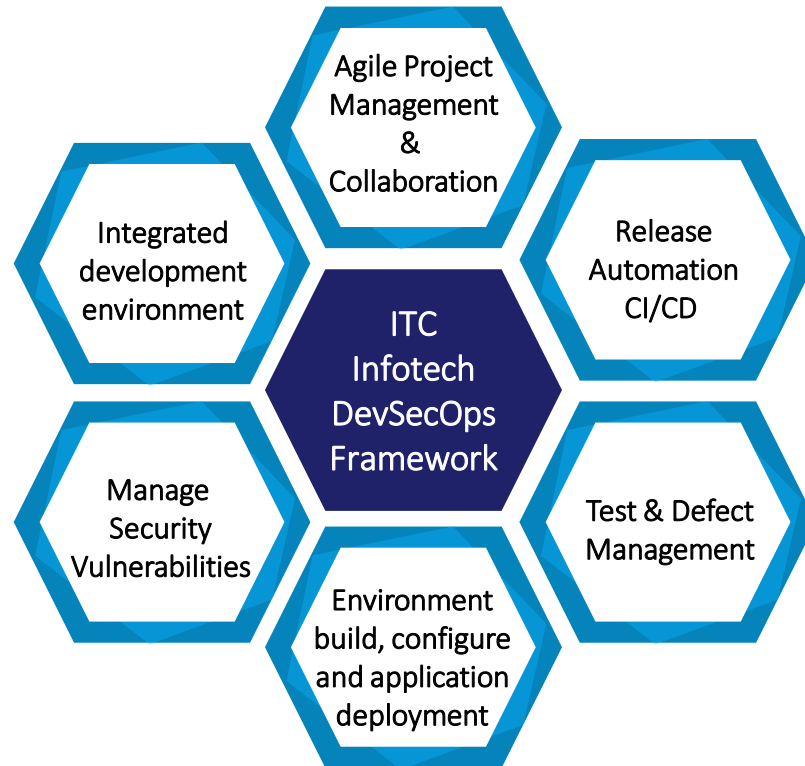
Modernization Themes – DevSecOps

Guiding Principles

- No code DevSecOps - GUI based, automation
- Cloud First - scalable across hybrid Cloud ecosystem
- Secure by Design – RBAC and Policy control
- DevSecOps compliant templates
- Value Stream – AI/ML based analytics

Potential Benefits

- Cloud agnostic DevSecOps platform - GitLab based reusable, scalable Core
- Automating the deployment workflow and change approval process
- Infra-as-code pipeline architecture for reusable library development
- Implementing AI/ML Ops based on need and requirements



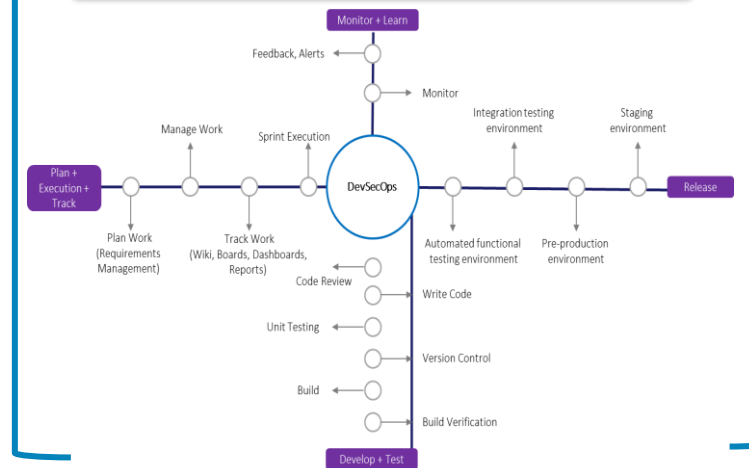
Potential tools that can be leveraged as appropriate



Key Considerations

- Customized process as per application type (Custom/COTS)
- Automated provision of infra resource (Iac)
- Use pre-built templated architecture blueprint library
- Constantly vulnerability scan
- Real time monitoring to track system performance
- Scalable across the application landscape
- Promote reuse
- Rationalization of toolsets
- Future ready for any cloud changes

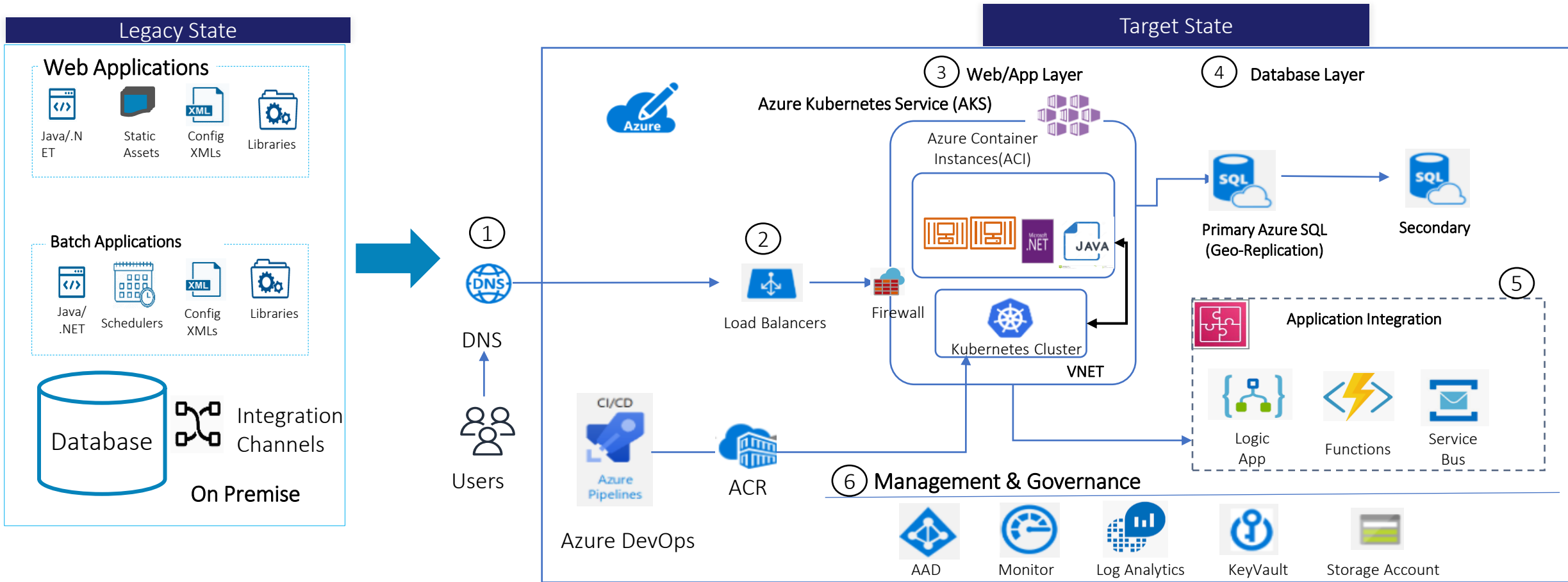
Course of Actions





Solution Patterns

Solution Pattern - Monolith Java/.NET App to Containerization App



- 1

Azure DNS will be routing internet traffic to application domain name.
- 2

Azure Firewall inspects requests coming to CloudFront and perform allow or block as per defined rules. Allowed request passes to Azure Load balancer.
- 3

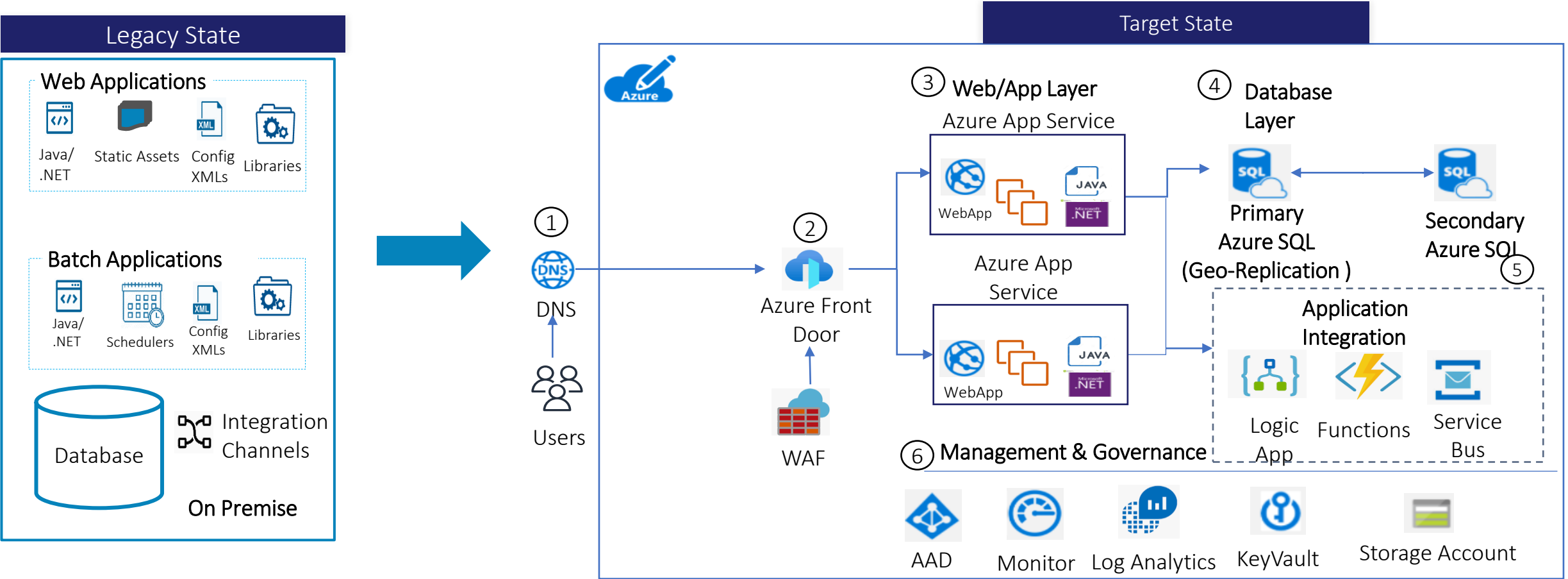
The containerized web/app layer will be deployed using AKS service via the ACR
- 4

Azure SQL is used for application database with Replication option
- 5

Application Integration can be achieved with multiple Azure offerings. Some of the examples are Logic App , Functions, Event Hub and Service Bus
- 6

Application resource access managed by IAM roles and policies. Monitor and Log Analytics OMS is used to do logging. Key Vault is used to store Keys and Secrets of Applications. Storage Account is used to store application data to maintain sources of truth and applying life cycle rules.

Solution Pattern - Monolith Java/.NET App to Azure PaaS



- 1

Azure DNS will be routing internet traffic to application domain name
- 2

Azure Load Balancer is used in AKS and After creating an AKS cluster, the cluster is ready to use the load balancer
- 3

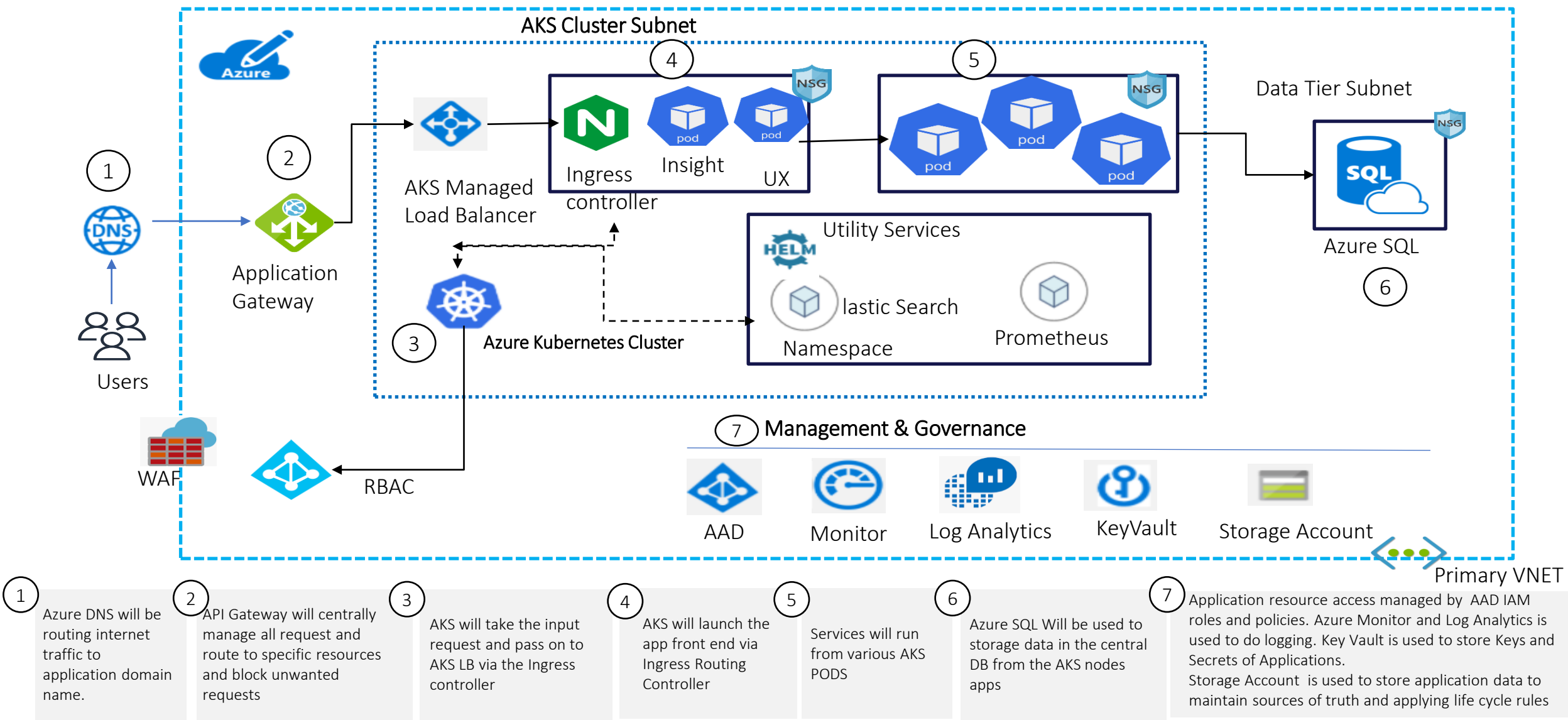
The containerized web/app layer will be deployed using AKS with ACI
- 4

Azure SQL is used for application database with geo-replication
- 5

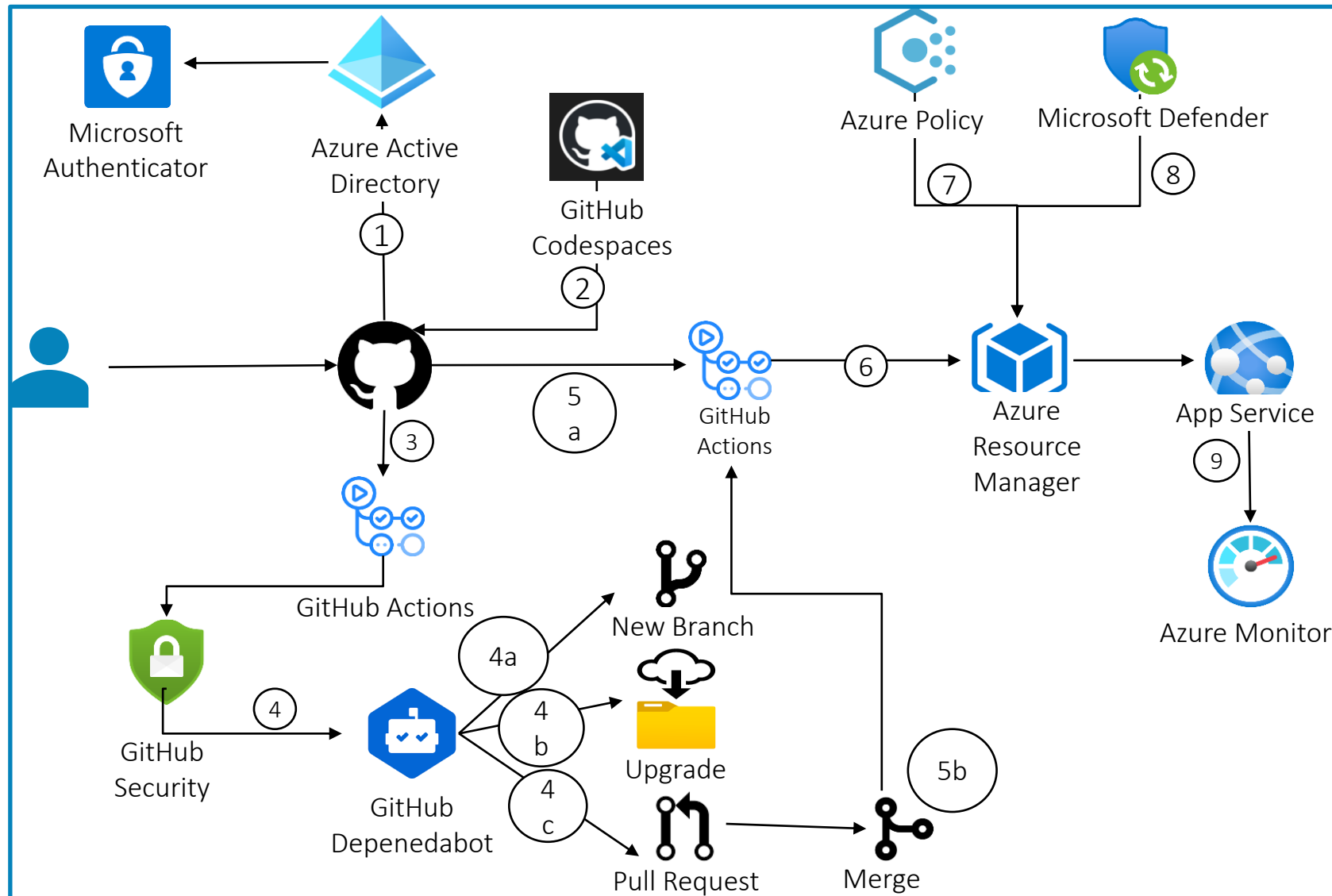
Azure Container Registry. Use Container Registry to store private Docker images, which are deployed to the cluster. AKS can authenticate with Container Registry using its Azure AD identity. Note that AKS does not require Azure Container Registry. You can use other container registries, such as Docker Hub.
- 6

Application resource access managed by AAD IAM roles and policies. Azure Monitor and Log Analytics is used to do logging. Key Vault is used to store Keys and Secrets of Applications. Storage Account is used to store application data to maintain sources of truth and applying life cycle rules.

Solution Pattern - Microservices Architecture on Azure Kubernetes Service

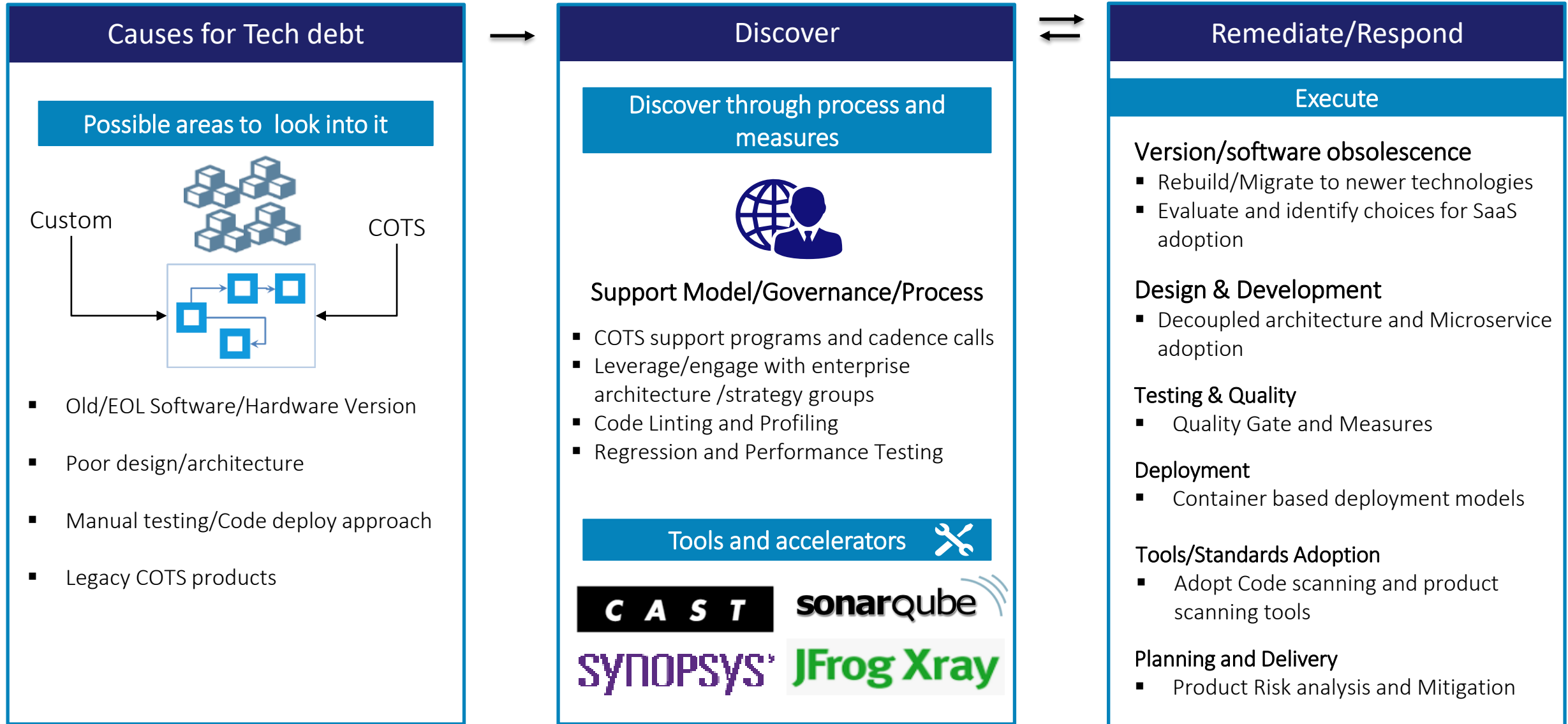


Solution Pattern - DevSecOps with GitHub



- ① GitHub do developer SAML authentication using MS authenticator
- ② Developers use pre-built dev env with security extensions
- ③ On new code commit - GitHub Actions automatically scan the code to quickly find vulnerabilities and coding errors
- ④ GitHub Dependabot, a DevOps bot agent, automatically detects vulnerability in dependencies
- ⑤ Trigger code builds and automated testing through GitHub Actions
- ⑥ Deploy build artifacts to Azure App Service while making changes to other cloud resources, such as service endpoints
- ⑦ Azure Policy evaluates Azure resources that are in deployment
- ⑧ Microsoft Defender for Cloud identifies attacks targeting applications that are running in deployed projects
- ⑨ Azure Monitor continuously tracks and evaluates app behavior. When threats materialize, this service sends alerts to start the process of rolling code back to previous commits

Cross cutting Solution Pattern – Handling Technical Debt

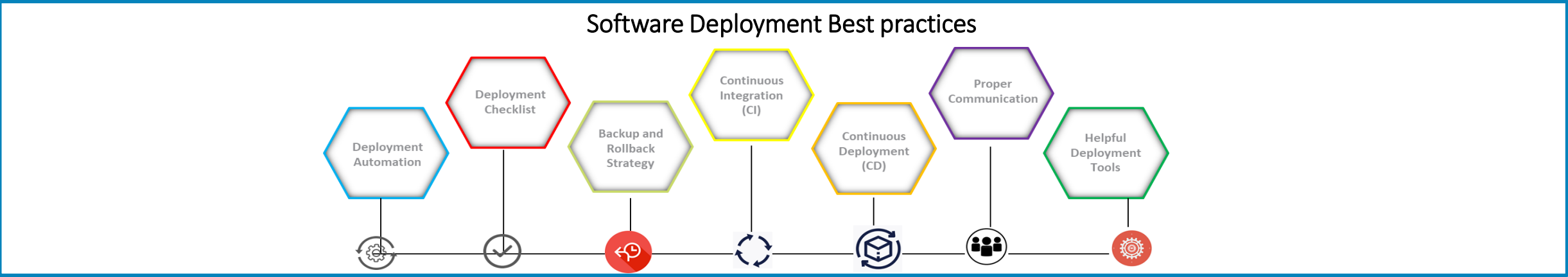


Cross cutting Solution Pattern – Version Upgrade

Key Considerations
▪ Downtime and Application Availability ▪ Reduced Deployment risks ▪ Slow Rollback (vs) Instant Rollback ▪ Sticky sessions ▪ Cost & Operational Overhead ▪ Observability ▪ Ability to test live production traffic ▪ Testing new backend features by using the production load

Best Practices
▪ Version Lifecycle management ▪ Backward compatibility checks & scripts - for UI, App layer, DB schema changes ▪ Continuous integration/continuous deployment/Continuous testing (CI/CD/CT) ▪ Automated pipelines- Managing Environment ▪ Operating & Configuration Management ▪ Rollback strategy ▪ Release Automation Tools ▪ Post Deployment Monitoring ▪ Infrastructure as Code (IaC) ▪ Monitoring tools

Patterns
Deployment Pattern ▪ Recreate ▪ Rolling Update ▪ Blue/green ▪ Canary
Testing Pattern ▪ A/B ▪ Shadow





TOOLS AND ACCELERATORS

Cloud Native Modernization - Tools & Accelerators

ITC Infotech Tools

**Monolith 2 Microservices
Assessment tool**

**Intelligent Monitoring
Platform**



CSP PaaS offerings

**Azure**

**Azure AD**

**App Service**

**Function**

**Service Mesh**

**Blob Storage**

**Service Bus**

**API management**

**AKS**

**ACR**

**DevOps**

**DMS**

**ADF**

**Azure SQL**

**Storage Account**

**KeyVault**

**Monitor**

**Log Analytics**

3rd Party Tools

Version Control


Secure Code Review


API Management


IaC and Configuration Management


Web Application Pen Testing


Infra VAPT and Vulnerability Management

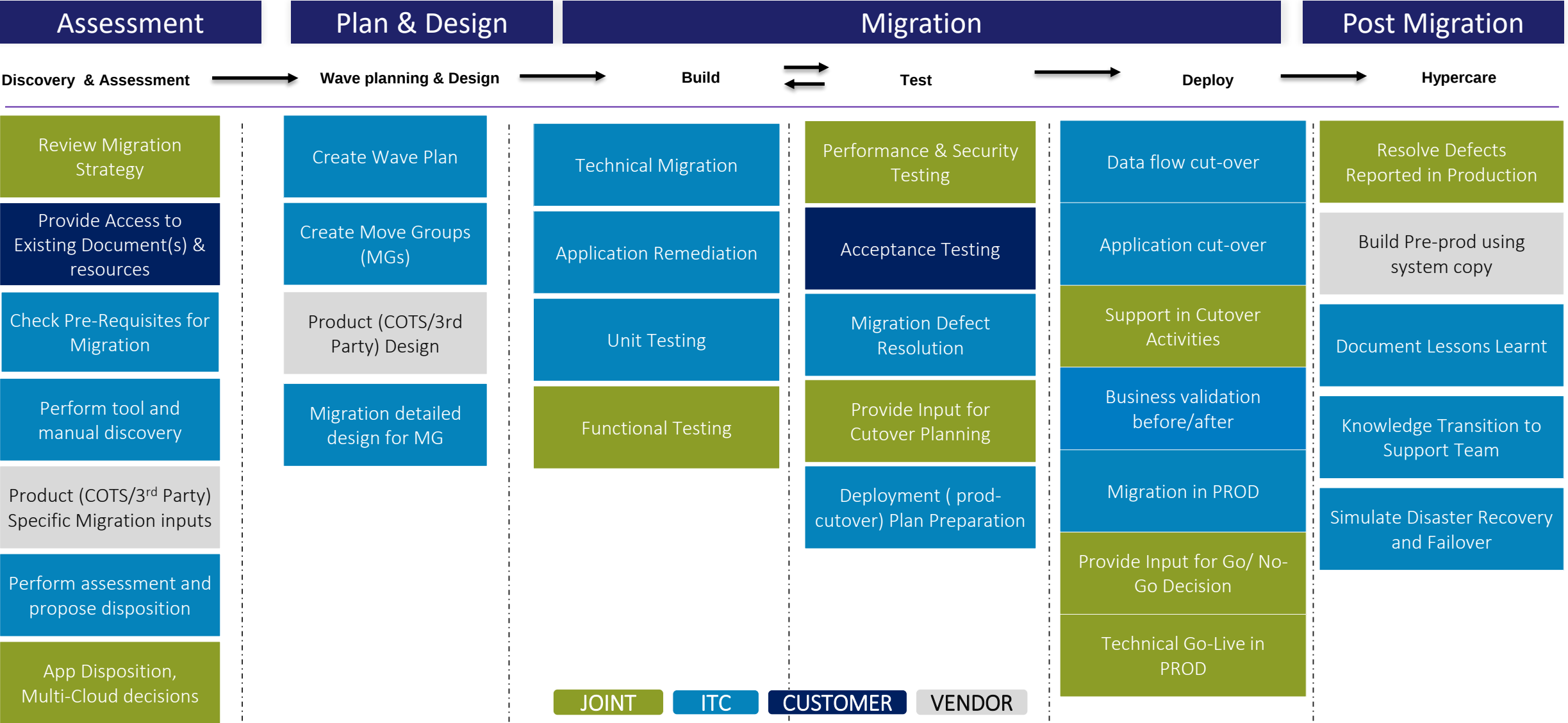

DevSecOps


Standards




GOVERNANCE

Migration Governance – Collaboration Model





THANK YOU



Email: contact.us@itcinfotech.com



Web: www.itcinfotech.com