Bring Your Use Case to Life Through Architectural Design

—

John Abbott – IBM Garage
Carlos Santana – STSM, Cloud Native
Sean Sundberg – Lead Engineer, Cloud Native Toolkit

IBM
Fast Start
2021

IBM

For the best digital learning experience

Please have your camera on when in session, if possible.

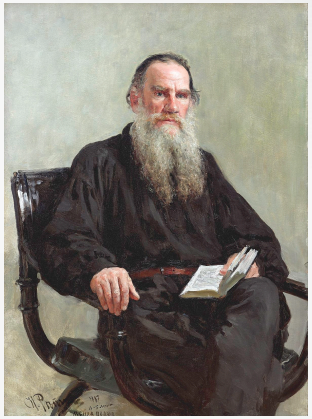
Focus and do not work on other tasks while in session.

Please participate in your session and ask questions. We encourage interactivity.

Topics

- Guiding principles
- Examples from the field
- Agile architecture, evolutionary architecture
- Cloud Paks and sizing for you story
- Measuring your success

You are not …



What this is not ….

Guiding Principles

Guiding Principles

“Do just enough design to get your project underway, make initial architectural decisions and to give you the opportunity to prove your hypothesis and mitigate risk”

Guiding Principles

“Do just enough design to get your project underway, make initial architectural decisions and to give you the opportunity to prove your hypothesis and mitigate risk”

“If you are successful, you will have the opportunity for additional architecture when it comes time to scale the solution in production”

Guiding Principles

“Do just enough design to get your project underway, make initial architectural decisions and to give you the opportunity to prove your hypothesis and mitigate risk”

“If you are successful, you will have the opportunity for additional architecture when it comes time to scale the solution in production”

“Fatigue is the invisible enemy of your project”

The approach we use to accelerate innovation enabled by Cloud Paks and OCP

Framing

We work with our sponsors within the client to identify the highest potential business value and desired business outcome.

Envision

Using Garage proven techniques, we work to align LOB and IT on initiatives that meet both the business and IT strategy agreeing upon modernization MVPs to pursue.

Architecture

With the technology owners, architects and solution developers we assessed and aligned the minimum viable architectures needed to build the MVP and test our hypotheses.

MVP Build

Together our teams test the hypotheses via learning and exploration-driven iterations to find the capabilities fit that aligns to business value and solution provider requirements.

Scale

Teams are aligned and prepared to power up their applications and scale capabilities to continue providing business value.

The approach we use to accelerate innovation enabled by Cloud Paks and OCP

Architecture Opportunities

Framing

We work with our sponsors within the client to identify the highest potential business value and desired business outcome.

Envision

Using Garage proven techniques, we work to align LOB and IT on initiatives that meet both the business and IT strategy agreeing upon modernization MVPs to pursue.

Architecture

With the technology owners, architects and solution developers we assessed and aligned the minimum viable architectures needed to build the MVP and test our hypotheses.

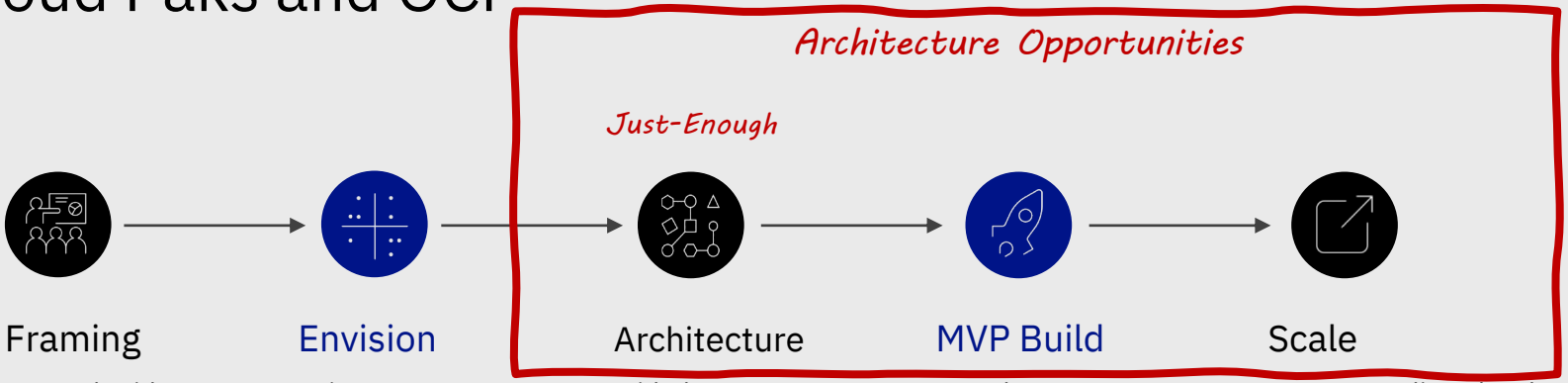
MVP Build

Together our teams test the hypotheses via learning and exploration-driven iterations to find the capabilities fit that aligns to business value and solution provider requirements.

Scale

Teams are aligned and prepared to power up their applications and scale capabilities to continue providing business value.

The approach we use to accelerate innovation enabled by Cloud Paks and OCP



Framing

We work with our sponsors within the client to identify the highest potential business value and desired business outcome.

Envision

Using Garage proven techniques, we work to align LOB and IT on initiatives that meet both the business and IT strategy agreeing upon modernization MVPs to pursue.

Architecture

With the technology owners, architects and solution developers we assessed and aligned the minimum viable architectures needed to build the MVP and test our hypotheses.

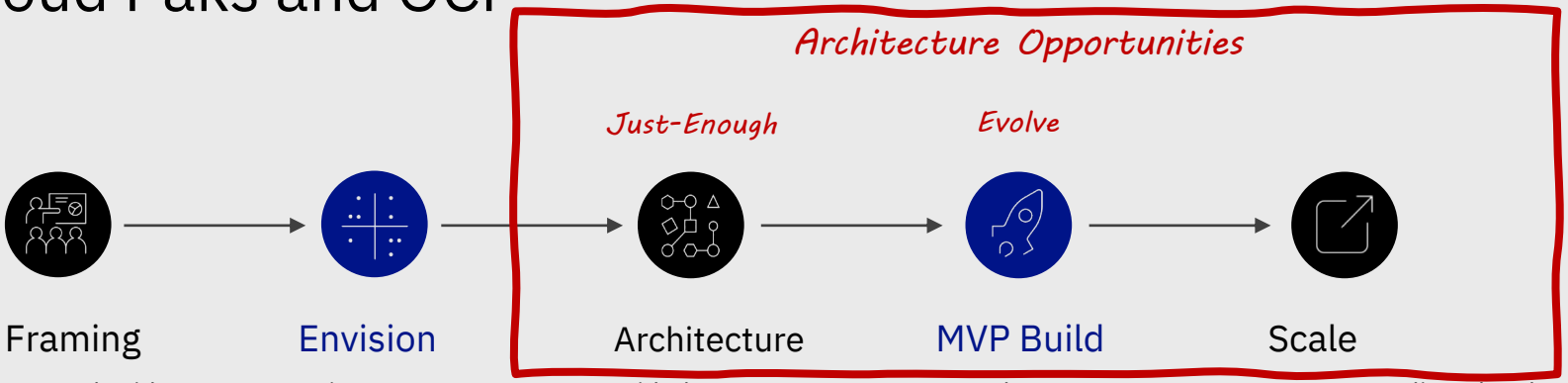
MVP Build

Together our teams test the hypotheses via learning and exploration-driven iterations to find the capabilities fit that aligns to business value and solution provider requirements.

Scale

Teams are aligned and prepared to power up their applications and scale capabilities to continue providing business value.

The approach we use to accelerate innovation enabled by Cloud Paks and OCP



Framing

We work with our sponsors within the client to identify the highest potential business value and desired business outcome.

Envision

Using Garage proven techniques, we work to align LOB and IT on initiatives that meet both the business and IT strategy agreeing upon modernization MVPs to pursue.

Architecture

With the technology owners, architects and solution developers we assessed and aligned the minimum viable architectures needed to build the MVP and test our hypotheses.

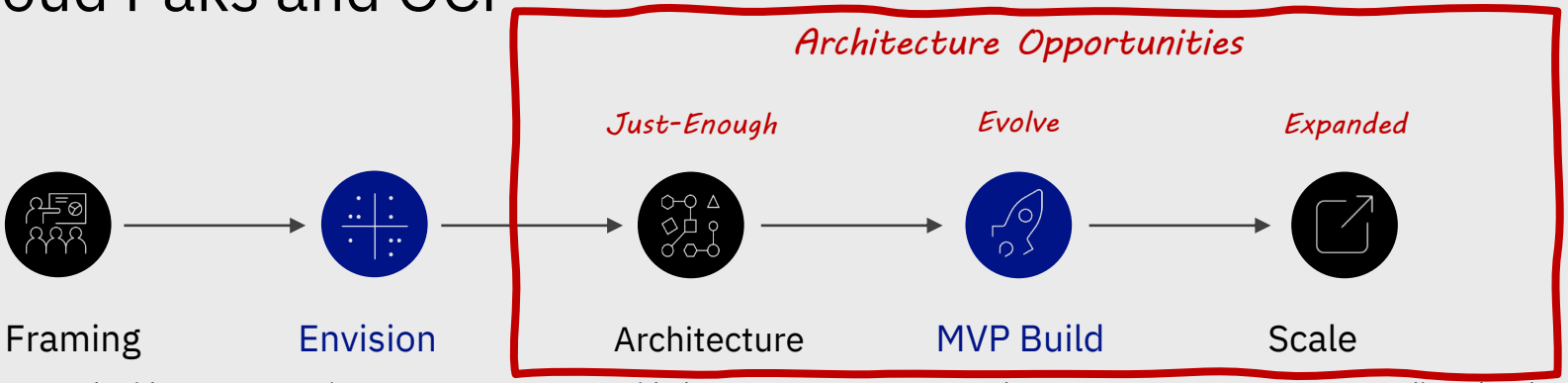
MVP Build

Together our teams test the hypotheses via learning and exploration-driven iterations to find the capabilities fit that aligns to business value and solution provider requirements.

Scale

Teams are aligned and prepared to power up their applications and scale capabilities to continue providing business value.

The approach we use to accelerate innovation enabled by Cloud Paks and OCP



Framing

We work with our sponsors within the client to identify the highest potential business value and desired business outcome.

Envision

Using Garage proven techniques, we work to align LOB and IT on initiatives that meet both the business and IT strategy agreeing upon modernization MVPs to pursue.

Architecture

With the technology owners, architects and solution developers we assessed and aligned the minimum viable architectures needed to build the MVP and test our hypotheses.

MVP Build

Together our teams test the hypotheses via learning and exploration-driven iterations to find the capabilities fit that aligns to business value and solution provider requirements.

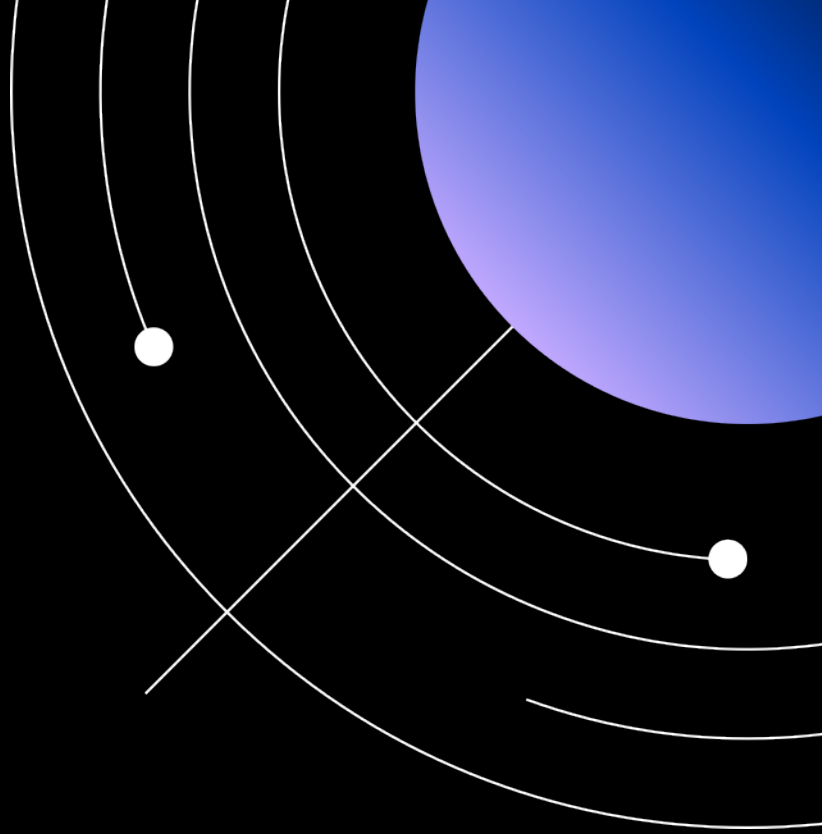
Scale

Teams are aligned and prepared to power up their applications and scale capabilities to continue providing business value.

Just Enough Architecture

(aka. Just Enough Software Architecture)

- Base your design on something that is proven
- Don’t architect beyond your MVP(s)
- The architecture develops commonality for making technical decisions
- Focus on retiring risks
- Define it early and keep it updated
- Focus on the architecturally significant aspects
- Understand size and scope
- How will you and your client measure the success of your architecture?



Book by
George Fairbanks

Notes Sizing for Cloud Paks

- Your must know more than just the Cloud Paks you are installing but also which capabilities / services / runtimes and how many instances
- Capability topology and reliability considerations
- Compute, memory and container storage for the capability containers
- Persistent storage sizes, providers, access modes, etc.
- Additional services for the solution (pipelines, service mesh, tools, utilities, monitoring, services, …)
- Load expectations (volume, size, performance expectations, …)
- Hosting platform(s)
- Head space (unused space required by OpenShift to schedule / reschedule workload)

Helpful blog with more description

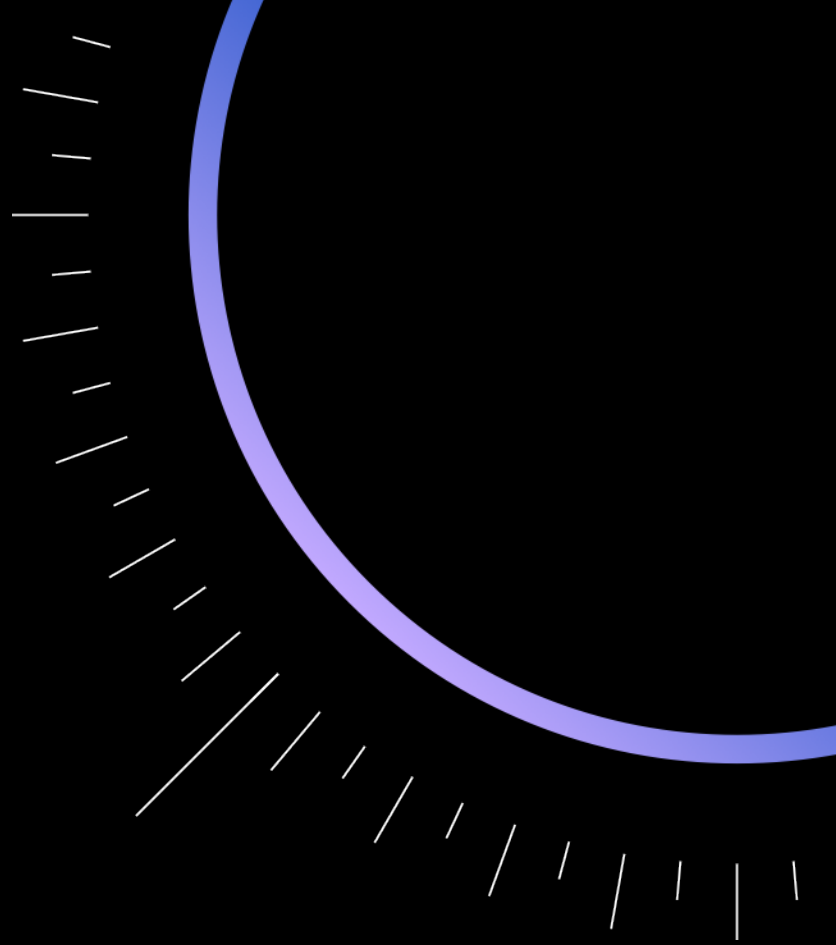
12 Principles of Agile Software

1. Our highest priority is to satisfy the customer through early and continuous delivery.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business staff and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity–the art of maximizing the amount of work not done–is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Your Design Will (and Should) Change

Architectures Must Provide for Evolution

- Learn to exploit change
- How long can you plan for?
- Build the ability to evolve into your architecture
- Easier said then done?
- Establish evolution as a conclusion



Architecting for Reliability

Architecting for Reliability
Blog by Ingo Averdunk
https://ibm.biz/reliable-arch

Reliability	Reliability describes the ability of a system or component to function under stated conditions for a specified period of time.
Availability	Availability is the proportion of time a system is in a functioning condition. Availability is one aspect to describe Reliability, but it is not limited to Availability. Examples of other aspects are Latency, Quality, Correctness, and Durability.
Resilience	Resilience is the ability of a system to recover from certain types of failure and yet remain functional from the user perspective. Disruptive events cannot always be anticipated, but when they occur they should result in learning and adaptation.
Antifragility	Antifragility is a property of systems that increase in capability to thrive as a result of stressors, shocks, volatility, noise, mistakes, faults, attacks, or failures.

Reliability cannot simply be delegated to the infrastructure and/or platform a service is running on. Every tier needs to provide their contribution towards the reliability goal of the entire system.

Reliability cannot be added after the fact on top of a built system, the system and its components need to be designed and implemented with reliability in mind.

It is a shared responsibility by everyone contributing across the Software Development Lifecycle, including the Architect, the Product Owner, the *conscientious* DevOps Engineer, and the SRE.



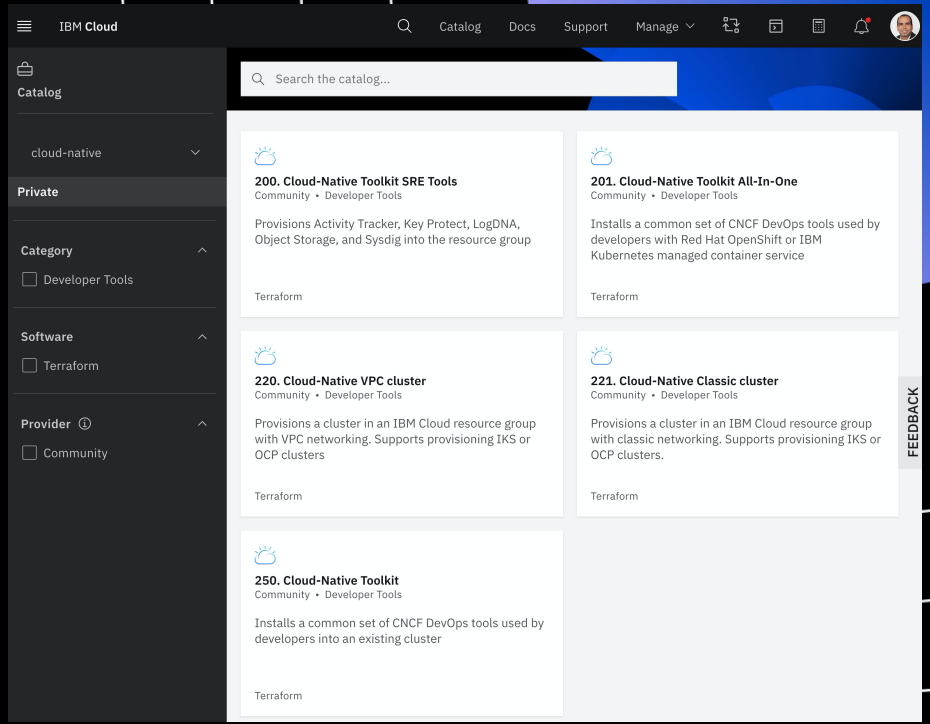

“
/rɪˌlaɪəˈbɪlɪti/ - Reliability describes the ability of a system or component to function under stated conditions for a specified period of time.
”

- Wikipedia

Use Cases

Use Case: Cloud Development Environment

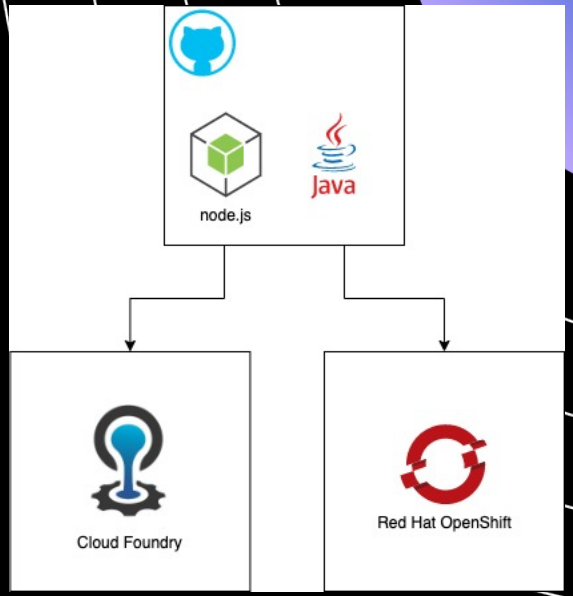
- Develop solutions built on top of Cloud Paks (Integration, Data, Automation)
- Quickly and repeatably set up new environments for development (and production)
- Introduce modern DevSecOps and GitOps best practices for on-going success
- Extend IBM Cloud with customizable Private Catalog offerings based on Terraform
- Simplifies the hand off to operations to move applications into production
- Supports the entire software development life-cycle from idea to production



- Client References:
  - Banking
  - Airline

Use Case: Cloud Foundry to OpenShift

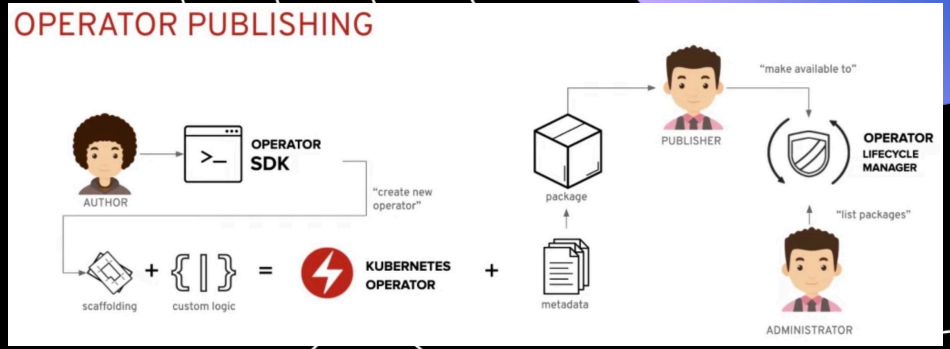
- Move from CF to containers using OpenShift
- Provide central governance of DevOps process and Config Management
- Minimize the impact on developers and their existing processes
- Support Websphere Liberty and Node.js as the Stack
- Introduce abstraction for service binding and support a mixed deployment environment for extended migration
- Run Spring Cloud Config Server on OpenShift
- Leverage IBM Cloud data services (i.e. Cloudant, Redis)



- Client References:
  - Health Insurance

Use Case: DevOps for OpenShift Operators

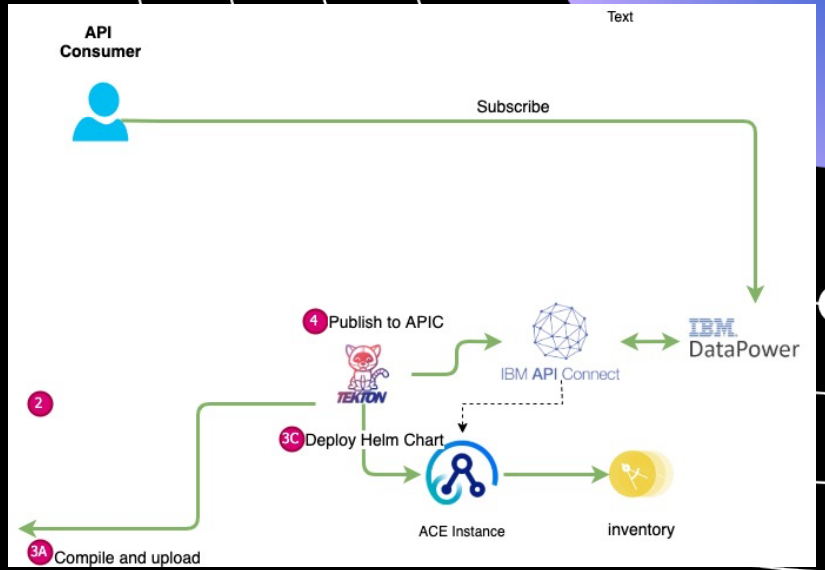
- Provide end-to-end development life-cycle for in-house Operators to extend the platform and Cloud Pak capabilities
- Support development environment on air-gapped OpenShift clusters
- Continuous Integration with OpenShift Pipelines
- Continuous Delivery with GitOps leveraging ArgoCD
- Package operators in Custom Catalog in OpenShift OperatorHub



- Client References:
  - Banking

Use Case: App and Ops Modernization

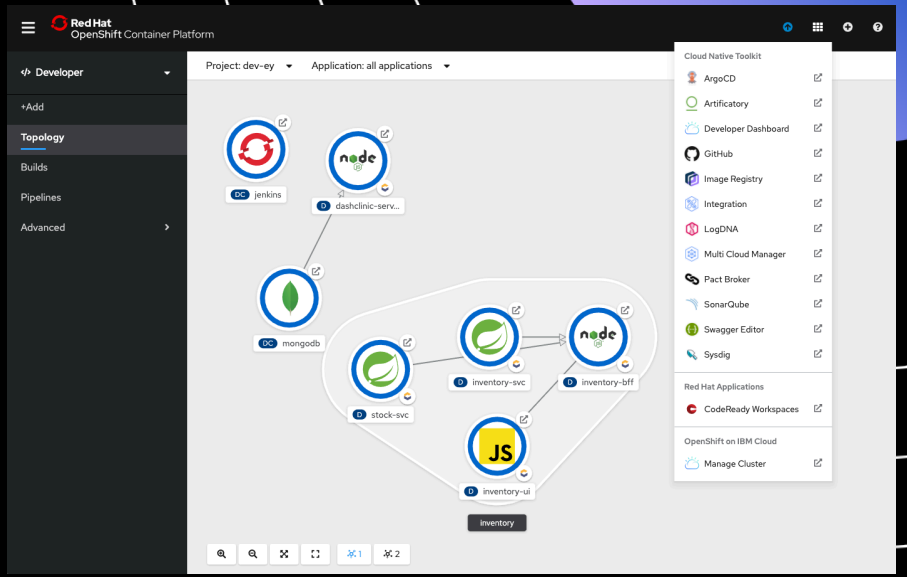
- Migrate from Monolithic application to Microservices leveraging Cloud Pak for Integration
- Continuous Integration with OpenShift Pipelines
- Customized Pipelines to build ACE bar
- Customized Pipeline for API Connect publishing
- Build once, deploy to multiple clouds
- GitOps leveraging Cloud Pak for MCM to deploy in multi-cloud environment



- Client References:
  - Home and Auto Insurance

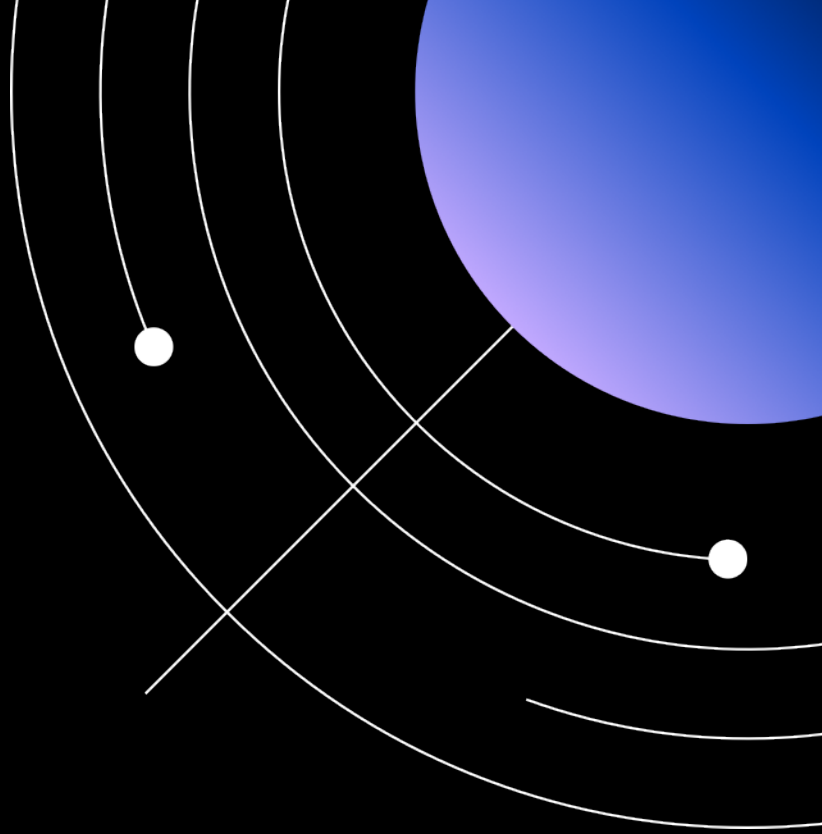
Use Case: Microservices Development

- Automated provisioning of Development environment
- Develop microservices that consume models built in Cloud Pak for Data and integrate using Event Streams in Cloud Pak for Integration
- Continuous Integration with OpenShift Pipelines
- Continuous Deployment with GitOps using ArgoCD
- GitOps process integrated with IBM Key Protect
- Extended Pipelines to add IBM Cloud Services
- Build and use customized Starter Kit templates
- Apply Build to Manage (Tracing, Logs, Metrics) techniques out of the box



- Client References:
  - Entertainment
  - Banking
  - Insurance

Resources



- Cloud Native Toolkit
  - https://cloudnativetoolkit.dev
- Cloud Native Toolkit Workshop
  - https://workshop.cloudnativetoolkit.dev
- Red Hat Open Labs
  - https://developer.ibm.com/openlabs/openshift
- DTE Infrastructure:
  - https://ccp-ui.csplab.intranet.ibm.com

Thank you