EIP Anypoint Platform Architecture

**Prepared by:**  PwC

**Prepared for:** Platform Admins, Architects and Developers

**Last updated:** <>

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| 0.1 |  |  | Initial version |
| 1.0 |  |  | Delivered version |
|  |  |  |  |

Table of contents

[**Introduction**](#_jqt32y57udcr) **5**

[**Context**](#_1t3h5sf) **5**

[**Approach and Methodology**](#_4d34og8) **5**

[**Architecture Dimensions**](#_17dp8vu) **6**

[Business Architecture](#_3rdcrjn) 6

[Context Diagram](#_26in1rg) 7

[Platform Architecture](#_lnxbz9) 8

[Available Capabilities](#_35nkun2) 8

[Deployment Model](#_1ksv4uv) 8

[Components](#_44sinio) 10

[Environments](#_2jxsxqh) 10

[VPC (CloudHub Only)](#_z337ya) 10

[DNS (CloudHub Only)](#_3j2qqm3) 10

[VPN (CloudHub Only)](#_1y810tw) 11

[~~Shared Load Balancer (CloudHub Only)~~](#_4i7ojhp) 11

[Dedicated Load Balancer (CloudHub Only)](#_2xcytpi) 11

[~~Mule Clusters (Standalone Runtimes Only)~~](#_1ci93xb) ~~12~~

[~~Mule Server Groups (Standalone Runtimes Only)~~](#_3whwml4) ~~12~~

[~~TCP Load Balancer (RTF Only)~~](#_2bn6wsx) ~~12~~

[~~Workers (RTF Only)~~](#_qsh70q) ~~12~~

[~~Controllers (RTF Only)~~](#_3as4poj) ~~12~~

[~~Nodes (PCE Only)~~](#_1pxezwc) ~~13~~

[Backend Systems](#_49x2ik5) 13

[High Availability](#_2p2csry) 13

[Disaster Recovery](#_147n2zr) 14

[Gap Analysis](#_3o7alnk) 14

[Security Architecture](#_23ckvvd) 14

[Management Plane Security](#_ihv636) 14

[Business Groups](#_32hioqz) 15

[External Identity](#_1hmsyys) 16

[Client Management](#_41mghml) 16

[Roles and Permissions](#_vx1227) 16

[Users](#_3fwokq0) 17

[~~Special Users~~](#_1v1yuxt) 18

[Exchange Portals](#_4f1mdlm) 18

[Runtime Plane Security.](#_2u6wntf) 18

[DLB Security](#_19c6y18) 18

[VPC Firewall](#_3tbugp1) 18

[Application Security](#_28h4qwu) 19

[API Access](#_nmf14n) 19

[API Policies](#_37m2jsg) 19

[Mule application properties](#_46r0co2) 20

[Backend system users](#_2lwamvv) 20

[Gap Analysis](#_111kx3o) 20

[Data Architecture](#_3l18frh) 20

[Domains identified](#_206ipza) 21

[Gap Analysis](#_4k668n3) 21

[Application Architecture](#_2zbgiuw) 21

[Integration Patterns](#_1egqt2p) 21

[Layers](#_3ygebqi) 22

[Components View](#_2dlolyb) 23

[API Taxonomy](#_sqyw64) 24

[Development Best Practices](#_3cqmetx) 24

[Gap Analysis](#_1rvwp1q) 24

[**Common Mechanisms**](#_4bvk7pj) **24**

[Logging](#_2r0uhxc) 25

[Alerts and Notifications](#_1664s55) 25

[Monitoring](#_3q5sasy) 25

[Send Anypoint Platform Data/Metadata to External Systems](#_25b2l0r) 26

[**Common Services**](#_kgcv8k) **26**

[Examples/Templates](#_34g0dwd) 26

[RAML Fragments](#_1jlao46) 27

[Health Check Mechanism](#_43ky6rz) 27

[**SDLC**](#_2iq8gzs) **27**

[Process](#_xvir7l) 27

[Source Control and versioning strategy](#_3hv69ve) 29

[Deployment and sizing strategy](#_1x0gk37) 29

[Deployment Naming Conventions](#_4h042r0) 29

[CI/CD](#_2w5ecyt) 30

[SDLC Tooling Systems](#_1baon6m) 31

# Introduction

The following document is intended to provide a detailed view of multiple Architecture dimensions related to <XYZ project>; covering business, technology, application, security, cross mechanisms, SDLC phases/tooling and the high level solution design.

This Architecture document should be treated as a living document and tailored over time with Sandy Spring Bank team needs and experiences from each project.

# Context

Sandy Spring Bank selected MuleSoft and Anypoint Platform to serves as the companies Enterprise Integration Platform. The integration between Backbase and Fidelity (FIS) IBS will be piloted on this platform, as part of the transition of the digital banking experience from Apiture to Backbase.

The Anypoint Platform will serve as the services platform to decouple the Backbase implementation from the backend systems like IBS and Salesforce

# Approach and Methodology

<Describe the approach and methodologies to be used in the project, including Architecture guidelines to follow, quality assurance or any other formal process, as well the communication plan>

The approach to be used in this project will be based on multiple guidelines and best practices including mainly API-Led connectivity and Domain Driven Design.

The SDLC will be based on agile methodologies and tools.

References:

* [API-led connectivity: The next step in the evolution of SOA](https://www.mulesoft.com/lp/whitepaper/api/api-led-connectivity)
* [What is Domain-Driven Design?](https://dddcommunity.org/learning-ddd/what_is_ddd/)

# 

# Architecture Dimensions

The following sections cover multiple architecture dimensions at a detailed level

## Business Architecture

<Describe the goals of the project related to business processes, include key decisions and stakeholders involved in these processes>

Based on the discovery sessions, we identified some pain points that are directly related to the following business-related processes:

* Human Capital management (employee entry and data updates)
* Order Management

The main points identified across business processes are:

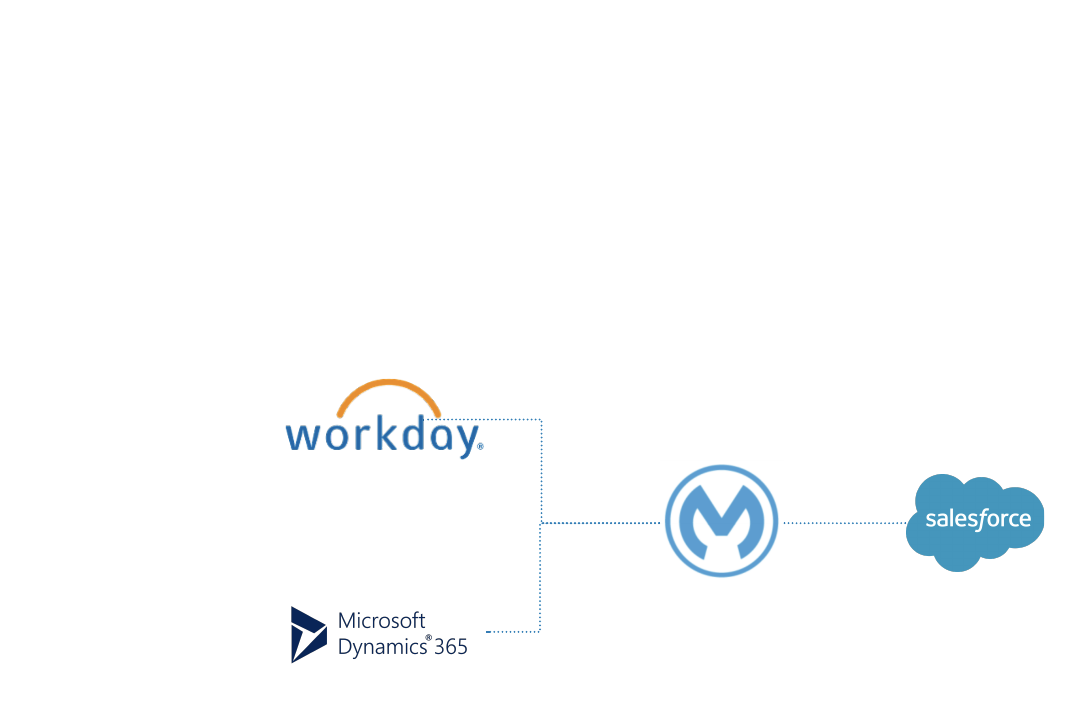
* Multiple systems involved
* Data synchronization

The specific data domains involved in the business are presented in the Data Architecture section and the solution to solve and improve the processes is covered as part of the Application Architecture section

### Context Diagram

<Describe the viewpoint of the context diagram, normally used as a high-level diagram showing actors and systems involved around Anypoint Platform>

The following diagram represents a high level view of the systems and actors involved in the scope of the project



<Catalogue of main systems involved>

|  |  |
| --- | --- |
| **System** | **Description** |
| Salesforce | *<Customer>* CRM |
| Workday | *<Customer>* Human capital management system to track Employee information |
| Microsoft Dynamics 365 | *<Customer>* main system for order management processes |

## Platform Architecture

The following section represents a drill down on the MuleSoft’s Anypoint Platform box presented in the context diagram

### Available Capabilities

This account has the following main capabilities/entitlements

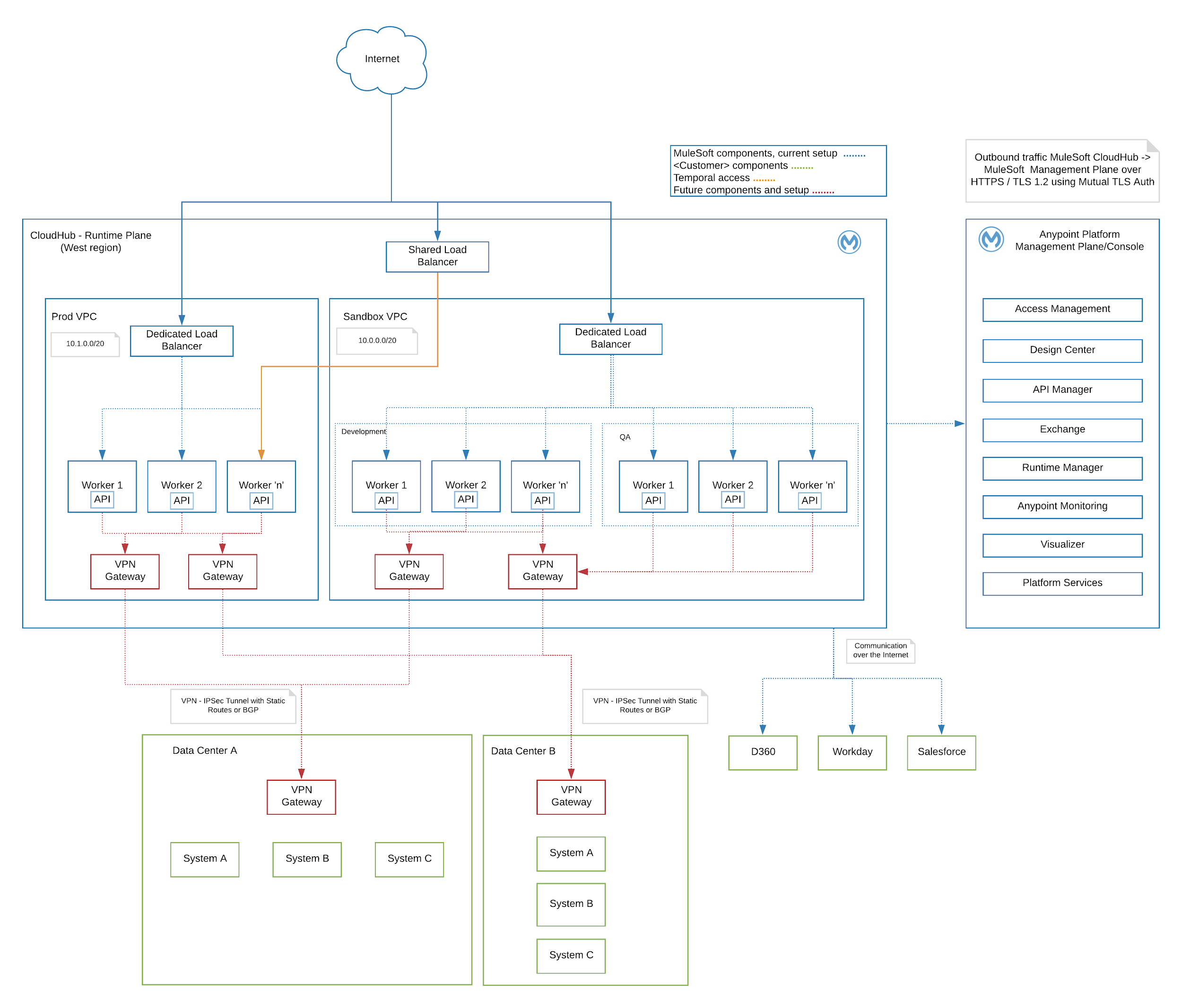
* Production vCores: <>
* Sandbox vCores: <>
* VPC: <>
* VPN: <>
* Dedicated Load Balancer: <>
* API Manager and Analytics
* Subscription: Titanium

Reference: [Product Subscription Plans](https://www.mulesoft.com/prod-subscription-plans)

### Deployment Model

The following diagram represents the Anypoint Platform’s deployment model selected for this project, **Management Plane/Console** in the cloud, hosted by MuleSoft and the **Runtime Plane** using CloudHub, as well hosted by MuleSoft.

Reference: [About CloudHub Architecture](https://docs.mulesoft.com/runtime-manager/cloudhub-architecture)



### Components

#### Environments

<Include the description of the environments to be used, and any relevant rationale behind the decision>

* 1 Production environment
* 2 Sandbox environments: Development and QA

#### VPC (CloudHub Only)

<Include the description of the VPCs to be used, and any relevant rationale behind the decision>

Two VPCs deployed in the West Region (N. California); One for Production environments and another one for Sandbox environments.

<Catalogue of VPCs example>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **VPC name** | **Location** | **CIDR** | **Environments** | **BGs** |
| <customer>-sandbox-vpc | US West (N. California) | 10.0.0.0/20 | Development  QA | Core Integrations |
| <customer>-production-vpc | US West (N. California) | 10.1.0.0/20 | Production | Core Integrations |

Reference: [Virtual Private Cloud](https://docs.mulesoft.com/runtime-manager/virtual-private-cloud)

#### DNS (CloudHub Only)

DNS to resolve *<Customer>* internal domains from CloudHub: To be configured in the future

Reference: [Resolve Private Domains in Your Internal Network](https://docs.mulesoft.com/runtime-manager/resolve-private-domains-vpc-task)

#### VPN (CloudHub Only)

<Include the description of the VPNs to be used, and any relevant rationale behind the decision, including the DR strategy for the VPNs>

**Future (possible)**: 2 or 4 VPNs deployed on top of the previously defined VPCs; Two for accessing *<Customer>* production systems and two for Sandbox connectivity.

<Catalogue of VPNs example>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **VPC name** | **VPC association** | **Type** | **Remote IP** | **Static Routes** |
| <customer>-sandbox-vpn | <customer>-sandbox-vpc | Static routing | TBD | TBD |
| <customer>-production-vpn | <customer>-sandbox-vpc | Static routing | TBD | TBD |

Reference: [Anypoint VPN](https://docs.mulesoft.com/runtime-manager/vpn-about)

#### Shared Load Balancer (CloudHub Only)

Shared Load balancer under the cloudhub.io domain

Reference: [CloudHub Networking Guide](https://docs.mulesoft.com/runtime-manager/cloudhub-networking-guide)

#### Dedicated Load Balancer (CloudHub Only)

<Include the description of the DLBs to be used, and any relevant rationale behind the decision>

<Include catalogues of TLS endpoints and mapping rules>

Dedicated Load balancer under the *<Customer>* vanity domain: to be defined in the future

Reference: [Dedicated Load Balancers](https://docs.mulesoft.com/runtime-manager/cloudhub-dedicated-load-balancer)

#### Mule Clusters (Standalone Runtimes Only)

<Include the description of the Mule clusters to be installed, and any relevant rationale behind the decision, include the DR strategy>

<Include the rationale of the cluster communication strategy: multicast/unicast>

<Include ant key decision related to O.S, JVM/wrapper tuning>

<Include catalogues mapping nodes to clusters, network segmentations, etc>

#### Mule Server Groups (Standalone Runtimes Only)

<Include the description of the Mule server groups to be defined, and any relevant rationale behind the decision, include the DR strategy>

<Include catalogues mapping nodes to server groups, network segmentations, etc>

<Include ant key decision related to O.S, JVM/wrapper tuning>

<Include catalogues mapping nodes to clusters, network segmentations, etc>

#### TCP Load Balancer (RTF Only)

<Include the description of the TCP Load balancers to be used in the RTF installation, and any relevant rationale behind the decision>

#### Workers (RTF Only)

<Include the description of the number of workers and topology to be used in the RTF installation, and any relevant rationale behind the decision>

#### Controllers (RTF Only)

<Include the description of the number of controllers and topology to be used in the RTF installation, and any relevant rationale behind the decision>

#### Nodes (PCE Only)

<Include the description of the number of nodes and topology to be used in the PCE installation, and any relevant rationale behind the decision>

### Backend Systems

<Backend systems catalogue example>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **System name** | **Location** | **Access Protocol** | **Host** | **Access** |
| Salesforce - sandbox - partial | Cloud | HTTPS | https://<customer>.my.salesforce.com | Integration user |
| Salesforce - sandbox - full | Cloud | HTTPS | TBD | Integration user |
| Salesforce - production | Cloud | HTTPS | TBD | Integration user |
| Workday - sandbox | Cloud | HTTPS | wd5-impl-services1.workday.com | Integration user |
| Workday - production | Cloud | HTTPS | TBD | Integration user |

### High Availability

<Include the description of the HA strategy for the Runtime Plane, and any relevant rationale behind the decision>

High Availability is covered out of the box by CloudHub fabric features by assigning more than one worker to each deployment

References:

* [Clustering](https://docs.mulesoft.com/runtime-manager/cloudhub-fabric)
* [High Availability and Disaster Recovery](https://docs.mulesoft.com/mule-runtime/4.1/hadr-guide)

### Disaster Recovery

<Include the description of the DR strategy for the Runtime Plane and any relevant rationale behind the decision>

Disaster Recovery covering CloudHub multi-region deployment not covered as part of this project

### Gap Analysis

<Include a gap analysis between the baseline/foundation Architecture and the target/future Architecture, specifically related to Platform components>

## Security Architecture

The following section is a drill down of the MuleSoft box presented on the Context diagram, it includes security related details of the solution, covering 3 different levels:

* Management Plane level
* Runtime Plane level
* Application level

### Management Plane Security

The following section contains a drill down on all security mechanisms related to the Management Plane (Anypoint Platform’s Management Plane/Console)

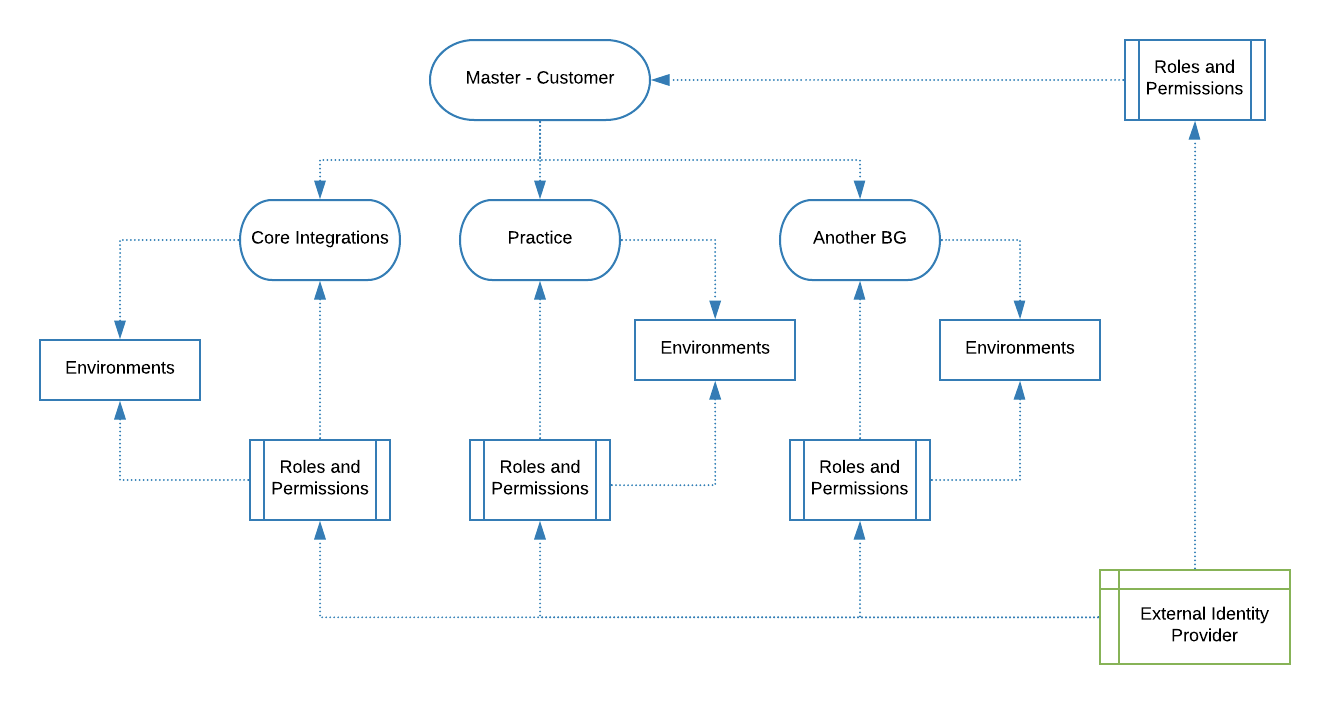
#### Business Groups

<Include the description of the Business Groups segregation and any relevant rationale behind the decision>

A business group defines an isolated group with its own objects (environments, applications, roles and permissions). Based on the current scope, the following Business Groups will be created in the Platform with a defined set of roles and permissions.

2 Business Groups are defined under the master org as part of the initial setup

* **Core Integrations**: The main BG where all the initial assets are going to be published
* **Practice**: For onboarding the team and to separate real assets from training-related ones



Reference: [Business Groups](https://docs.mulesoft.com/access-management/business-groups)

#### External Identity

Single Sign On to be configured using **One Login/Okta/etc**

Roles propagation: <Include details>

References:

* [About Identity Management](https://docs.mulesoft.com/access-management/external-identity)

#### ~~Client Management~~

~~<Include the strategy for Client Management, using OOTB Anypoint Platform Client Management or any external system or multiple IdPs>~~

#### 

#### Roles and Permissions

<Include the strategy for Roles and Permissions mapping, including OOTB roles, custom roles and the relationship with external IdPs>

The following roles and their assigned permissions are defined as part of the initial setup, including mappings with <XYZ> roles/groups

<Roles and Permissions catalogue>

|  |  |  |  |
| --- | --- | --- | --- |
| **Group** | **Anypoint Platform Role** | **Type** | **Permissions** |
| developers | CloudHub Admin  Exchange Contributors | Custom role  Default role | Runtime Manager Dev environments:   * CloudHub Network Viewer * Manage Alerts * Manage Application Data * Manage Queues * Manage Servers * Manage Schedules * Manage Settings * Manage Tenants * Read Alerts * Read Applications * Read Servers * Create Applications * Delete Applications * Download Applications   Anypoint Monitoring   * Anypoint Monitoring User   Design Center   * Design Center Developer |
| api-operations | Exchange Admin |  |  |
| platform-operations | Exchange Admin |  |  |
| platform-admin | Organization Administrator  Exchange Admin |  |  |

Reference: [Roles](https://docs.mulesoft.com/access-management/roles)

#### Users

The users have to be assigned to one or multiple roles previously showed

Reference: [Users](https://docs.mulesoft.com/access-management/users)

#### Special Users

Connected app to be defined for the CI/CD process

Reference: [Connected Apps](https://docs.mulesoft.com/access-management/connected-apps-overview)

#### Exchange Portals

**Private Portal:** Internal assets portal accessed by users with the right permissions in Anypoint Platform *<Customer>* account. (Exchange Viewers)

**~~Public Portal:~~** ~~Won’t be used as part of the scope of this project~~

References:

* [Anypoint Exchange](https://docs.mulesoft.com/exchange/)
* [Exchange Public Portals](https://docs.mulesoft.com/exchange/about-portals)

### Runtime Plane Security.

#### DLB Security

<Include relevant information related to the TLS termination at DLB level, certificates for the TLS endpoints, client certificates, whitelists and mapping rules related to security/access control>

#### VPC Firewall

<Include relevant information related to the VPC firewall rules to restrict access>

<VPC firewall rules catalogue example>

|  |  |  |
| --- | --- | --- |
| **CIDR** | **Destination** | **Scope** |
| Local VPC CIDR | 8091-8092 | Controls access:   * From any *<Customer>* applications within the VPC to other *<Customer>* applications within the VPC |
| *<Customer>* VPN remote subnets (in encryption domain) | 8091-8092 | Access from *<Customer>* Data Center subnets to Internal CloudHub url:   * mule-worker-internal-{app}.[cloudhub.io](http://cloudhub.io/):8091/8092   Optional in case any *<Customer>* (within *<Customer>* network) application needs to consume MuleSoft’s APIs |

Reference: [VPC Firewall Rules](https://docs.mulesoft.com/runtime-manager/vpc-firewall-rules-concept)

### Application Security

The following section contains a drill down on all security mechanisms related to applications/APIs

#### API Access

<Include relevant information related to the access to APIs>

* The access to APIs will be always using HTTPs and through the Dedicated Load Balancer only
* The authentication and authorization for accessing APIs will be managed by API Manager, policies: using clientId enforcement

#### API Policies

<Include a catalogue of API Policies to be used including decisions and rationales>

#### 

ClientId enforcement

Reference: [Policies for Mule 4](https://docs.mulesoft.com/api-manager/2.x/policies-mule4)

#### Mule application properties

All sensitive passwords and data on the properties files will be encrypted using the secure properties placeholder with a key that will be injected as part of the build process

Reference: [Secure Configuration Properties](https://docs.mulesoft.com/mule-runtime/4.2/secure-configuration-properties)

#### Backend system users

<Include decisions around the usage of integration credentials in Mule apps>

A system user per application per environment (sandbox/prod); e.g

* For Salesforce sandbox-partial: mule-integration-user
* For Salesforce sandbox-full: mule-integration-user
* For Salesforce prod: mule-integration-user
* For Workday sandbox: mule-integration-user
* For Workday prod: mule-integration-user

### Gap Analysis

<Include a gap analysis between the baseline/foundation Architecture and the target/future Architecture, specifically related to Security decisions>

## Data Architecture

<Include data domains identified, catalogues for data structures and any relevant view including the viewpoint>

The following section is a drill down of the data involved in the solution. Based on the use cases and the requisites provided, there are some clear domains identified, grouping information about Customers, Orders, etc. See the following sections to have a better understanding of the data involved in the solution.

### Domains identified

<Data domains and systems catalogue example>

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Domain** | **Source of truth (expected)** | **Current systems** |
| Employees | HCM | Salesforce | Salesforce, Workday |
| Orders | Order Management | D365 | Salesforce, D365 |

### Gap Analysis

<Include a gap analysis between the baseline/foundation Architecture and the target/future Architecture, specifically related to Data Architecture>

## Application Architecture

This section is a drill down of the Platform Architecture, covering the application related components and their interactions

### Integration Patterns

Based on the discussions and discovery sessions, the following integration patterns were identified:

<Integration patterns catalogue example>

|  |  |
| --- | --- |
| **Pattern** | **Description** |
| HTTP Synchronous APIs | Unlock information from different systems and expose the data as REST APIs |
| One way sync | Replicate data elements from one system to another  Workday to Salesforce |

### Layers

<Layer segmentation rationales>

**System Layer**

This layer represents multiple spheres of knowledge (ontologies) of Business objects (orders, accounts, etc), related to one or many business processes (e.g Order management, CRM, Sales, etc)

The system APIs will provide a means of accessing data of underlying/core systems (Salesforce, Workday, D360, etc) exposing that data in a domain-based canonical format while providing downstream isolation from any interface changes or rationalization of those systems.

**Process Layer**

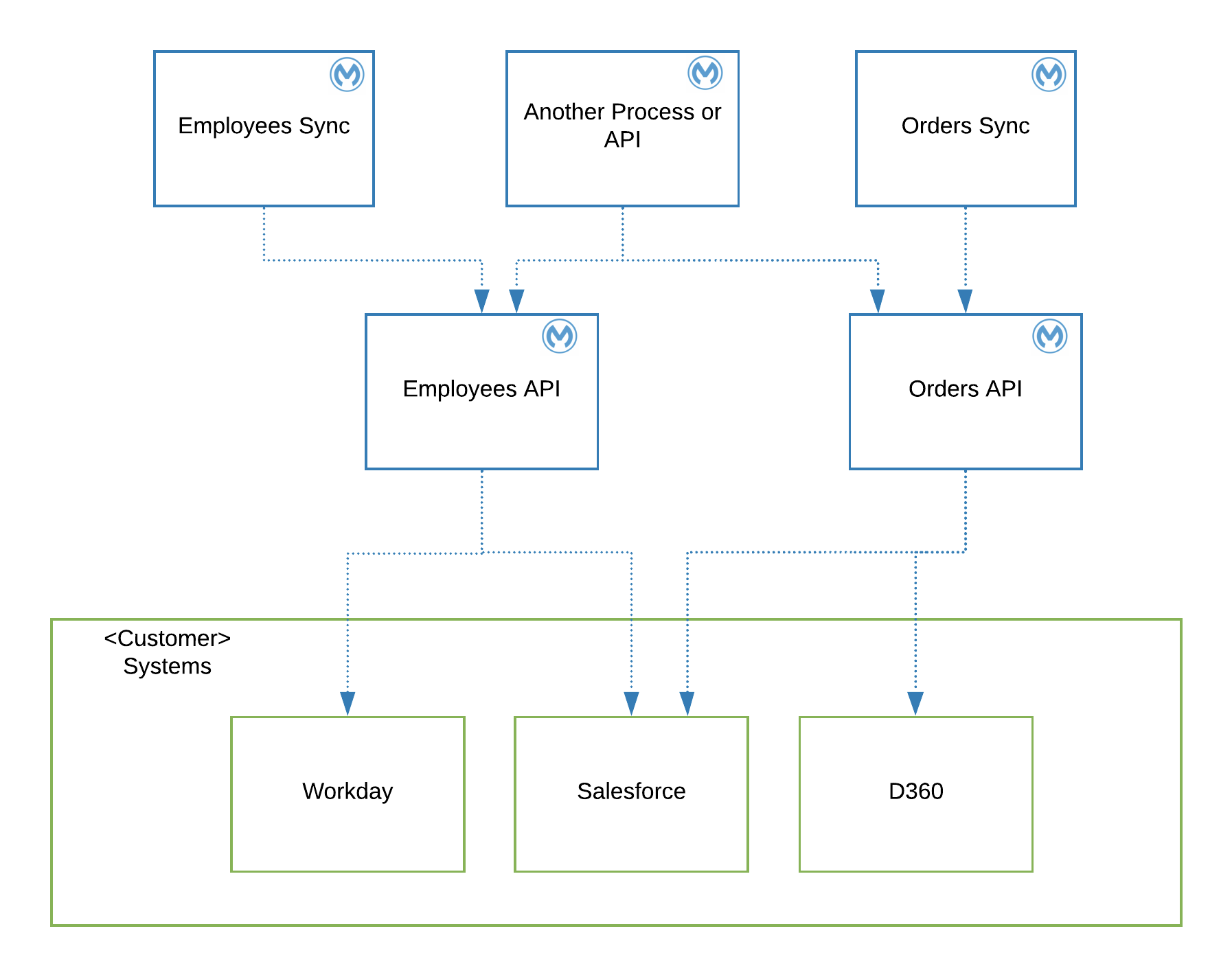
The underlying business processes (Human Capital Management, Order management, CRM, etc) that interact and shape this data should be strictly encapsulated independently of the source systems from which the data originates, as well as the target channels through which that data is to be delivered. These APIs perform specific business processes functions and provide access to non-central data

**Experience Layer**

Data is now consumed across a broad set of channels/teams, each of which wants access to the same data but in a variety of different forms. Experience APIs are the means by which data can be reconfigured so that it is most easily consumed by its intended audience, all from a common data source, rather than setting up separate point-to-point integrations for each channel.

Reference: [API-led connectivity: The next step in the evolution of SOA](https://www.mulesoft.com/lp/whitepaper/api/api-led-connectivity)

### Components View



<Mule applications catalogue example>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Layer** | **Business Domain** | **Components interaction** |
| Employees API | Mule API | Domain/System | HCM | Salesforce, Workday |
| Employees sync | Mule process | Process | HCM | Employees API |
| Orders API | Mule API | Domain/System | Order Management | D365, Salesforce |
| Orders sync | Mule process | Process | Order Management | Orders API |

Separate document to be provided with more details around each API: “Solution Design”

### API Taxonomy

The API Taxonomy should follow REST Design patterns. A separate document to be provided with details around API Taxonomy: “Solution Design”

### Development Best Practices

Separate document to be provided with details around Mule Development Best Practices: “Solution Design”

### Gap Analysis

<Include a gap analysis between the baseline/foundation Architecture and the target/future Architecture, specifically related to Application Architecture>

# Common Mechanisms

<Include the list of common mechanisms to be used in the Platform, including logging, monitoring and alerts>

## Logging

The logging mechanism will be implemented using the out of the box Logger component in applications

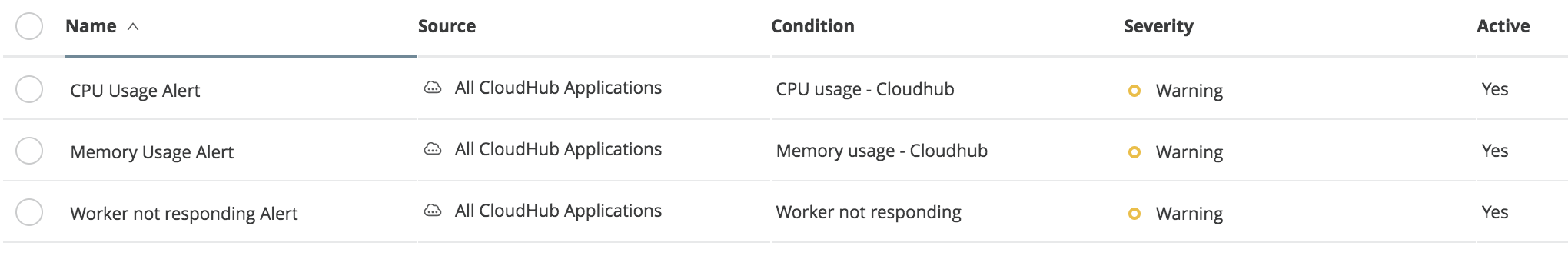
Reference: [Logger Component](https://docs.mulesoft.com/mule-runtime/4.2/logger-component-reference)

## Alerts and Notifications

The following alerts will be configured as part of the Platform setup

**API Manager, API Level alerts:** TBD

**Runtime Manager, Worker and Application level alerts:** CPU, memory and worker unresponsive; for all applications



Distribution List to be used: **team-dl@<customer>.com**

Reference: [Configure Alerts](https://docs.mulesoft.com/runtime-manager/alerts-on-runtime-manager)

## Monitoring

Use Anypoint Monitoring and Visualizer out of the box capabilities

Reference:[Anypoint Monitoring Overview](https://docs.mulesoft.com/monitoring/)

## Send Anypoint Platform Data/Metadata to External Systems

<Include details on how to send data from Anypoint Platform to external systems IF NEEDED. E.g Send Logs to Splunk/ELK, send API Analytics to Splunk/ELK>

# Common Services

<Include the list of common services to be used in the solution, including templates, archetypes, skeletons, RAML fragments, etc >

## Examples/Templates

APIs Skeletons: The skeletons will provide all the pieces needed for seamless development experience, covering the best practices described in the Application Architecture section

* Naming conventions
* Updated maven dependencies
* Properties per environment
  + users/passwords encrypted
* Dataweave scripts externalized
* APIKit including a default error handling flow
* Mule Maven Deploy plugin for CloudHub

## RAML Fragments

RAML Library containing the following fragments

* Authentication security scheme - Basic Authentication
* Server Error Codes trait: Default HTTP 5xx response codes definitions
* Client Error Codes trait: Default HTTP 4xx response codes definitions

## Health Check Mechanism

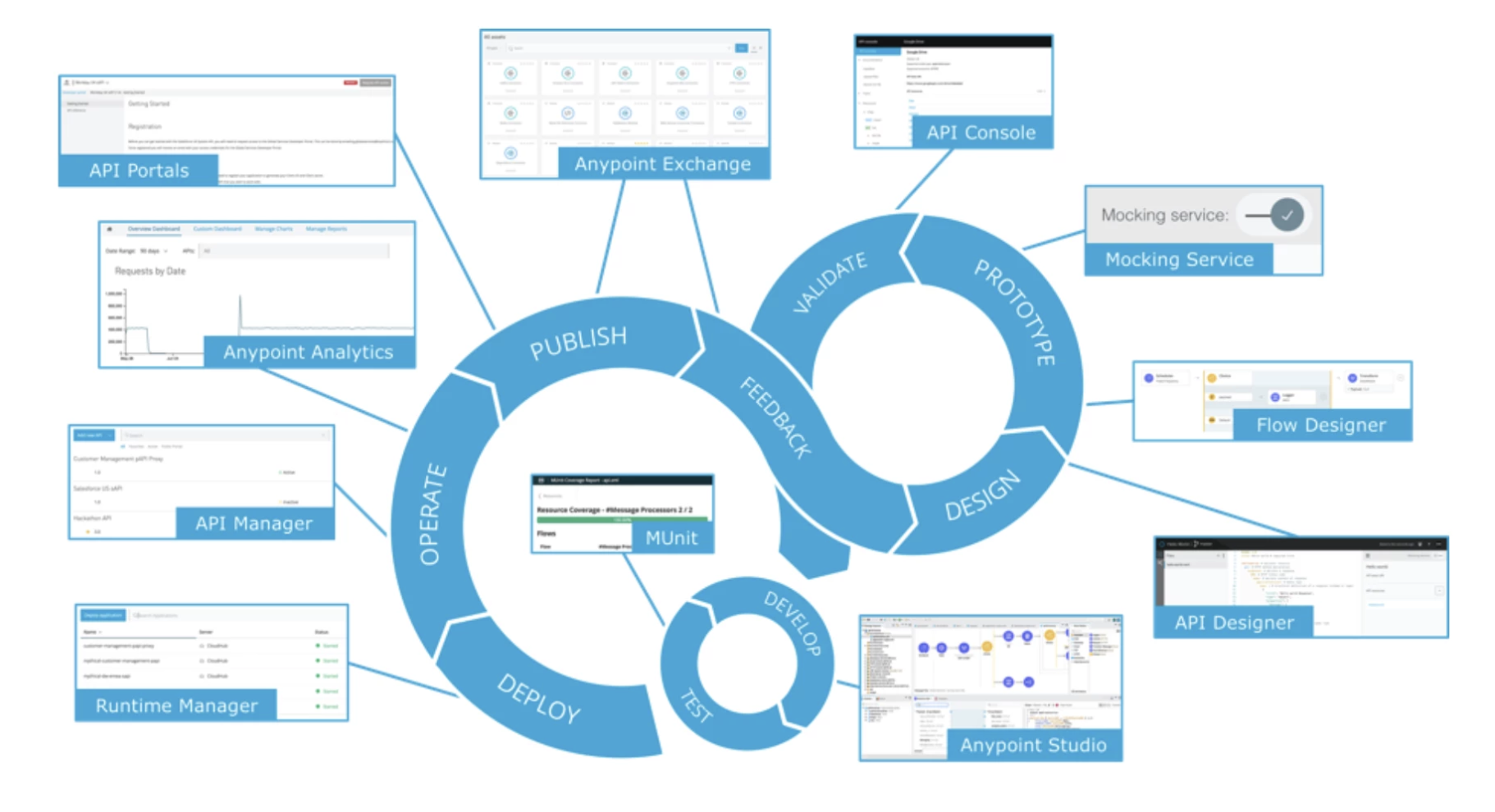
Each API/application has to provide a /health-check endpoint, to be used by external monitoring systems. It should return the self status and dependencies status

# SDLC

The following section describes the process and phases of the Software Development Life Cycle of a Mule application/API, the tooling and mechanisms used as part of it.

## Process

<Describe a high level of the process to be followed>



<SDLC Phases catalogue example>

|  |  |
| --- | --- |
| **Phase** | **Description** |
| **Discovery** | Gather information from <customer> related to specific needs of a defined story (JIRA). Research the available APIs in Exchange. Identify the domain (see Data Architecture section) |
| **Design (Prototype and Validate)** | Design the RAML in API Designer, publish the API Spec to Exchange. |
| **Development / Implementation** | Using Anypoint Studio and reusing any available common service (e.g auth-connector)   * Unit Testing (local): * Push source code to Bitbucket/GitHub |
| **Operate/Monitor** | Using out of the box features included as part of Anypoint Monitoring |
| **Functional Testing (Validation and Feedback)** | Example data set should be provided by <customer> |
| **Staging** | Once the integration is tested and approved by stakeholders, the promotion to higher environments should be executed |

## Source Control and versioning strategy

* Repository: Bitbucket/GitHub
* Repository Strategy: 1 repository per Mule application
* Branching strategy: As part of the scope of the project a simple branching strategy will be used: work on a “develop” branch, once the integration/features are ready, create a Pull Request for making a peer review, approve and merge the develop branch to the master. Tag the master once the version is approved by <customer> teams

Reference: [Git Workflow](https://www.atlassian.com/git/tutorials/comparing-workflows)

## Deployment and sizing strategy

Due to CloudHub’s deployment model, every application/API will be running in an isolated container; hence each of them has to be properly sized based on metrics from the business (if available) or results from performance testing. In the case of this project, there’s no available information about load/concurrency neither required throughput, hence it is needed to execute Performance testing to define the upper limits related to concurrency/load for each API’s operation.

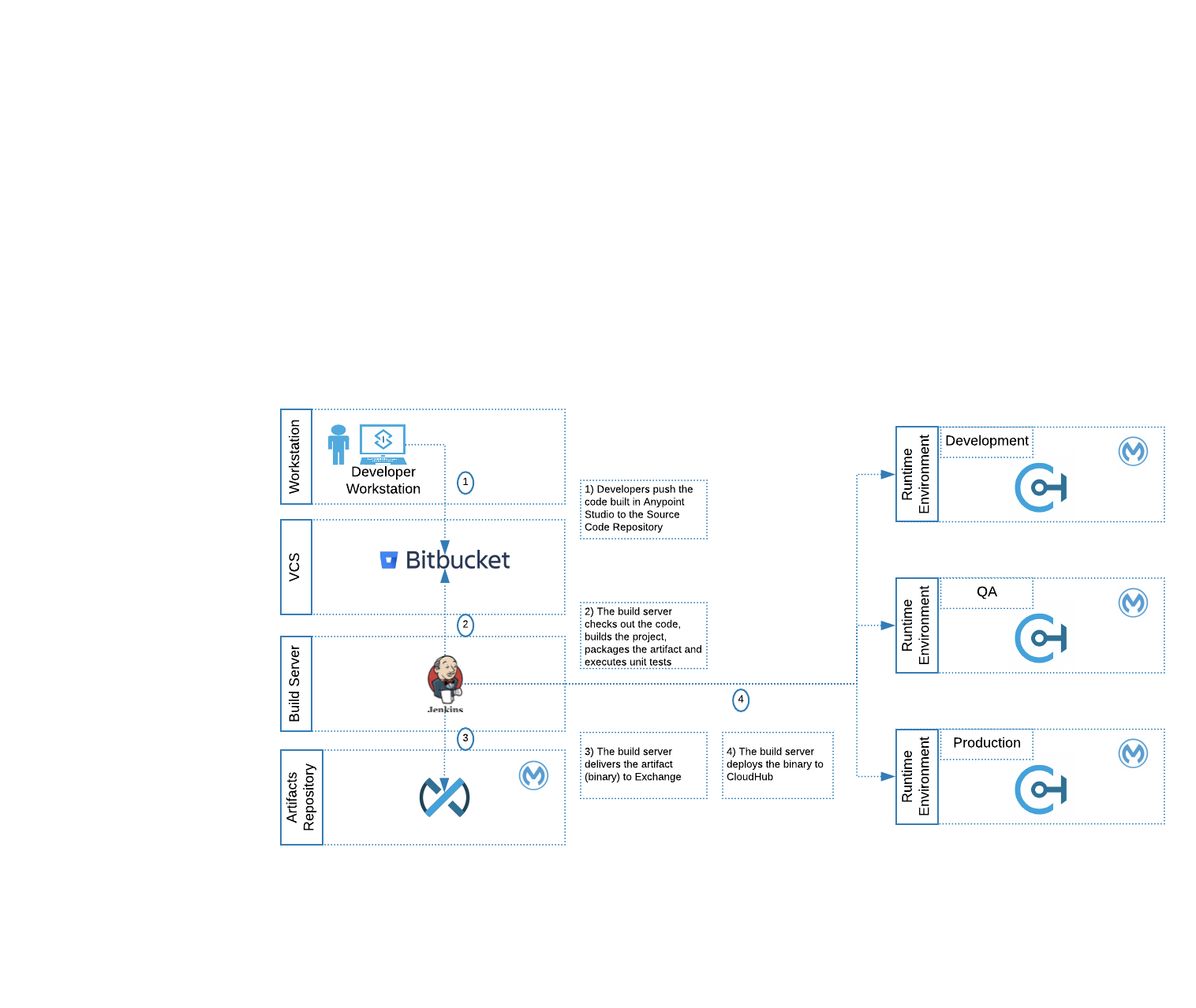
### Deployment Naming Conventions

Each API will be deployed using the following naming convention: <customer>-<env>-<name>-api.<region>.cloudhub.io e.g <customer>-dev-employees-api.us-w2clouhub.io

In the case 2 different versions of the same API need to be deployed as separate artifacts, the following convention should be used: <customer>-<env>-<name>-api-v<version\_number>.<region>.cloudhub.io e.g <customer>-dev-employees-api-v2.us-w2.cloudhub.io

## CI/CD

<Include rationales around the design of pipelines and how the CI/CD process will work for the solution>



## SDLC Tooling Systems

|  |  |  |
| --- | --- | --- |
| **Name** | **URL** | **Access** |
| **JIRA** | TBD | SSO |
| **Bitbucket/GitHub** | TBD | SSO |
| **Jenkins** | TBD | SSO |

About MuleSoft

MuleSoft, a Salesforce company

MuleSoft’s mission is to help organizations change and innovate faster by making it easy to connect the world’s applications, [data](https://www.mulesoft.com/integration-solutions/dataweave-integration), and [devices](https://www.mulesoft.com/integration-solutions/api/iot). With its API-led approach to connectivity, MuleSoft’s market-leading Anypoint Platform™ empowers over 1,600 organizations in approximately 60 countries to build application networks. By unlocking data across the enterprise with application networks, organizations can easily deliver new revenue channels, increase operational efficiency, and create differentiated customer experiences.

For more information, visit mulesoft.com

*MuleSoft is a registered trademark of MuleSoft, LLC, a Salesforce company. All other marks are those of respective owners.*