**WSO2 APIM to APIGEE- Migration**

Contents

[**1.** **Preamble:** 4](#_Toc9344930)

[**2.** **WSO2 Migration:** 4](#_Toc9344931)

[Existing Platform Analysis: 4](#_Toc9344932)

[Consumer Mapping: 4](#_Toc9344933)

[Zero Transaction APIs 5](#_Toc9344934)

[Provider Mapping: 6](#_Toc9344935)

[**3.** **Migration Model and Approach** 6](#_Toc9344936)

[**4.** **Migration Key Activities & Deliverables** 7](#_Toc9344937)

[WSO2 APIM Platform Analysis 7](#_Toc9344938)

[Product Migration 7](#_Toc9344939)

[Common Component Design in APIGEE 8](#_Toc9344940)

[Firewall opening and DNS routing 8](#_Toc9344941)

[Proxy Migration 8](#_Toc9344942)

[**5.** **APIGEE Migration Steps** 8](#_Toc9344943)

[Phase 1 8](#_Toc9344944)

[Phase 2 8](#_Toc9344945)

[**6.** **Pre-Migration Activities** 9](#_Toc9344946)

[Network Connectivity: 9](#_Toc9344947)

[Common Components: 9](#_Toc9344948)

[Redefinition/Refactoring: 9](#_Toc9344949)

[Collect Reusable Artifacts: 10](#_Toc9344950)

[Testing and Production deployment 10](#_Toc9344951)

[**7.** **Waves and Iteration** 10](#_Toc9344952)

[Wave 1 10](#_Toc9344953)

[Wave 2 10](#_Toc9344954)

[Wave 3 10](#_Toc9344955)

[Pilot 10](#_Toc9344956)

[**8.** **Shared Flows and Common Components** 11](#_Toc9344957)

[Logging Components: 11](#_Toc9344958)

[Security Framework: 11](#_Toc9344959)

[Shared Flow: 11](#_Toc9344960)

[**9.** **APIGEE API Product Strategy** 12](#_Toc9344961)

[Phase 2 – Product Strategy 12](#_Toc9344962)

[Phase 2 – Long Vision 12](#_Toc9344963)

[**10.** **Testing, Production Deployment and Rollback** 12](#_Toc9344964)

[Testing: 12](#_Toc9344965)

[Rollback Plan: 13](#_Toc9344966)

[**11.** **Developer Portal Migration** 13](#_Toc9344967)

[APIGEE integrated portal 13](#_Toc9344968)

[Drupal Portal 13](#_Toc9344969)

[**12.** **Consumer Communications** 13](#_Toc9344970)

[**13.** **Disaster and Recovery** 14](#_Toc9344971)

[**14.** **Migration Team – Daily Activities** 14](#_Toc9344972)

[**15.** **Execution Plan** 14](#_Toc9344973)

[Efforts Estimation 14](#_Toc9344974)

[Execution Plan 16](#_Toc9344975)

[Resource Loading 17](#_Toc9344976)

[Roles and Responsibilities 18](#_Toc9344977)

[**16.** **Assumptions and Risks** 19](#_Toc9344978)

[**17.** **Tools for Migration** 20](#_Toc9344979)

[**18.** **Sunset WSO2 Manager** 20](#_Toc9344980)

**Author: Mani Manavalan**

**Digital Architecture**

|  |  |  |
| --- | --- | --- |
| **Version & Date** | **Created/Changed by** | **Change details** |
| **1.0/ May 17, 2019** | **Mani Manavalan** | **Initial version** |
|  |  |  |

# **Preamble:**

SE has signed an enterprise license agreement with Google for the Apigee API Management solution effective from Jan 31st for global API solution**. Consolidate and migrate** legacy APIs existing in different platforms such as WSO2, MuleSoft and Exchange today.

# **WSO2 Migration:**

## Existing Platform Analysis:

There are 299+ APIs are operations in WAO2 APIM and around 25+ APIs are under development. Operational APIs are mainly grouped into 3 major categories based on their complexity level.

|  |  |  |
| --- | --- | --- |
| **Technical Complexity** | **Description** | **Count** |
| **Simple (Group 1)** | Pass-through APIs. No IFW ESB involved | 11 |
| **Medium (Group 2)** | APIs with payload/header transformation in API layer. No IFW ESB involved | 81 |
| **Complex (Group 3 & 4)** | Pass-through transaction in API layer. IFW ESB has logic involved. Candidates for refactoring. | 39 |
| Custom logic in API layer as well as IFW ESB layer. Candidates for redesigning APIs | 168 |

### Consumer Mapping:

Easy sales, Go-Digital applications are consuming a greater number of APIs.

10+ APIs consumed

### Zero Transaction APIs

Below APIs are doesn’t have any transaction for last 12 months in WSO2 APIM platform. API migration team check with platform owner before migrating these APIs into APIGEE. If required, selected APIs will be decommissioned.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **sl.no** | **API\_Name\_Version** | **Iteration** | **Wave** | **Zero Transaction** |
| 8 | BDO-Repository:v2.0 | 2.4 | 2 | Yes |
| 10 | BEM-SEAdvProposal:v1.0 | 6 | 1 | Yes |
| 16 | BFO-Approval:v1.0 | 2.2 | 2 | Yes |
| 23 | BFO-Case:v2.1 | 2.1 | 2 | Yes |
| 31 | BFO-ContactRelatedList:v1.0 | 2.2 | 2 | Yes |
| 42 | BFO-GetZQuoteAttachmentDetails:v1.0 | 2.1 | 2 | Yes |
| 43 | BFO-GlobalSearch:v1.0 | 2.2 | 2 | Yes |
| 59 | BFO-SalesEntity:v1.0 | 2.2 | 2 | Yes |
| 66 | BFO-UserRecordAccess:v1.0 | 2.2 | 2 | Yes |
| 67 | BFO-WorkOrderNotification:v1.0 | 2.1 | 2 | Yes |
| 76 | BOX-Search:v1.0 | 4 | 1 | Yes |
| 83 | DFSERP-DOMFinanceDocument:v1.0 | 3 | 1 | Yes |
| 88 | EBILL-Accounting:v1.0 | 2.3 | 2 | Yes |
| 100 | EBILL-Product:v1.0 | 2.3 | 2 | Yes |
| 102 | EBILL-Staging:v1.1 | 2.3 | 2 | Yes |
| 107 | EBILL-System-File:v1.0 | 3 | 1 | Yes |
| 122 | ERP-PSProject:v1.0 | 2.4 | 2 | Yes |
| 126 | ERP-SalesOrderBOM:v1.0 | 2.4 | 2 | Yes |
| 182 | PRM-Cockpit:v1.0 | 3 | 1 | Yes |
| 185 | PRM-Notification:v1.1 | 3 | 1 | Yes |
| 207 | SOAP-BFO-GetZQuoteAttachmentDetails:v1.0 | 2.1 | 2 | Yes |
| 213 | SOAP-BFO-UpdateZQuote:v1.0 | 2.1 | 2 | Yes |
| 227 | SOAP-ERP-UpdatePlannedCostsAndRevenues:v1.0 | 2.6 | 2 | Yes |
| 241 | SOAP-OneClick-Satelite-UpdatePSProject:v1.0 | 6 | 1 | Yes |
| 252 | SR-Bookmark:v1.0 | 3 | 1 | Yes |
| 253 | SR-Community:v1.0 | 3 | 1 | Yes |
| 255 | SR-Preference:v1.0 | 1 | 1 | Yes |
| 257 | SR-Product:v1.1 | 1 | 1 | Yes |
| 258 | SR-Repository:v1.0 | 1 | 1 | Yes |
| 259 | SR-Repository:v1.1 | 1 | 1 | Yes |
| 260 | SR-Repository:v1.2 | 1 | 1 | Yes |
| 261 | SR-SharedCommunity:v1.0 | 1 | 1 | Yes |
| 264 | SR-Switchboard:v1.1 | 1 | 1 | Yes |
| 276 | BFO-CreateWorkOrderNotification:v1.0 | 3 | 1 | Yes |
| 277 | EBILL-System-Account:v1.0 | 1 | 1 | Yes |
| 278 | EBILL-System-Quote:v1.0 | 1 | 1 | Yes |
| 279 | Email:v1.0 | 4 | 1 | Yes |
| 280 | ERP-SalesOrder:v1.0 | 2.6 | 2 | Yes |
| 281 | PRM-PartnerLocator:v1.0 | 1 | 1 | Yes |
| 282 | SOAP-GSE-ManageDevice:v1.0 | 1 | 1 | Yes |
| 283 | SOAP-GSE-UserOrgHierarchy:v2.0 | 1 | 1 | Yes |
| 284 | SOAP-LicenseService:v1.0 | 5 | 1 | Yes |
| 293 | BFO-ProjectCategorizationQuestions:v1.0 |  |  | Yes |

### Provider Mapping:

BFO and EBILL applications are providing higher number of APIs.

# **Migration Model and Approach**

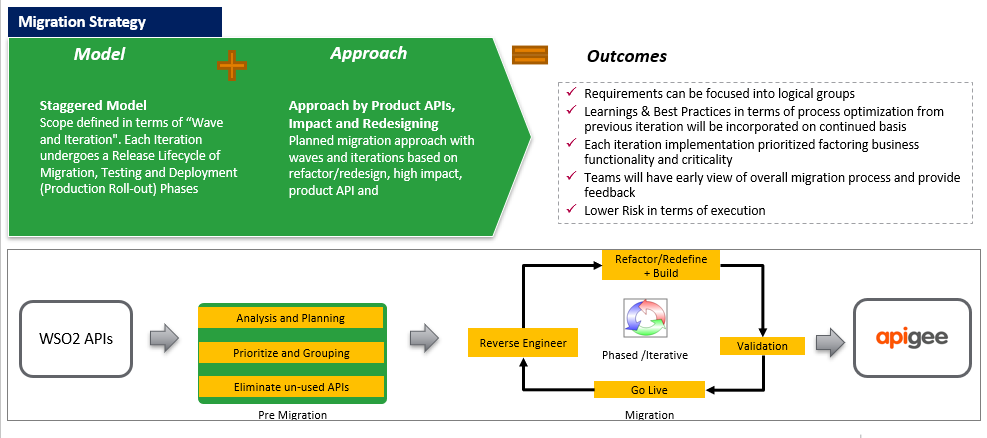
Analyzed the inventory in detail and prioritized the APIs based on product API candidate and high impact for consumers. Migrating APIs are categorized into three different waves and each wave will have collection of iterations.

Each iterations delivery will be discussed in the respective sprints and API migration process starts. There are 4 major actions will be done in each iteration.

1. Reverse engineer the code from WSO2 APIM
2. Decide to refactor/redefine the code. API Migration team communicate with consumers for redefinition of the API required or not.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Definition** | **Consumer Impact** | **API Impact** |
| **Rehost** | No major changes required in API. Apply best practices and move to APIGEE platform | All | 300+ |
| **Refactor** | WSO2 custom policy conversion to APIGEE out-of-box policies, ~~WSO2 custom logic refactoring, IFW Token validation logic in APIGEE, ESB light orchestrations to APIGEE, Routing logics from ESB to APIGEE~~ | Depends | 160+ |
| **Redesign** | Additional functionality and changes in request and response payload | Some | 2% |

1. API development in DEV environment and promote the QA by API team. THBS team will involve in the functional testing of the API. 20% of refined test scripts will be executed by THBS team and testing sign off document will be prepared.
2. API Ops team will push the API into production based on input from platform owner.



# **Migration Key Activities & Deliverables**

### WSO2 APIM Platform Analysis

* Deep dive discussion with WSO2 APIM team and group the APIs based on the complexity.
* Capture API technical complexity, security patterns, custom logic which will be reused, Custom logics mapped to APIGEE inbuilt policy, consumer & provider details, etc.
* Authentication/Authorization framework.
* Logging framework, Error handling framework & Rate Limits (spike/quota)
* Request/Response Validation, Request/Response transformation components.

### Product Migration

* WSO2 APIM doesn’t have API product concept. APIGEE API team defines the products in two phases. Phase 1: Products based on provider.
* Phase 2 – Products based on consumers. Future goal is to create products based on capability (capability criteria has to be finalized).

### Common Component Design in APIGEE

* Analyze configuration requirements such as KVM and Target servers for APIGEE.
* Throttling will be applied in each API and thread protection will be applied in shared flow.
* Consumer API Quota will be applied in the product level.
* Internal/External consumers authorized using Oauth security framework. Analyze to move ESB token validation & long-lived access token to APIGEE.
* Migrate WSO2 Oauth credentials to APIGEE to make sure consumer keys not affected.
* Logging framework (Logz.io?) will be implemented across all APIs.

### Firewall opening and DNS routing

* APIGEE cascade layer added in front of WSO2 and all the traffic go-through APIGEE server.
* DNS level setting adjusted to make sure no consumer will impact for this switch.

### Proxy Migration

* Consumers contact collected from internal portal(?) will be notified at-least 15 days(?) before production.
* API proxy creation with standards and best practices applied. Leverage partner’s tools and scripts to accelerate the migration and CI/CD process.
* Testing Sign-Off: THBS will share the 20% refined test script. Result will be compared on both platforms
* After testing sign off, separate CI/CD pipeline will push the proxy into production environment

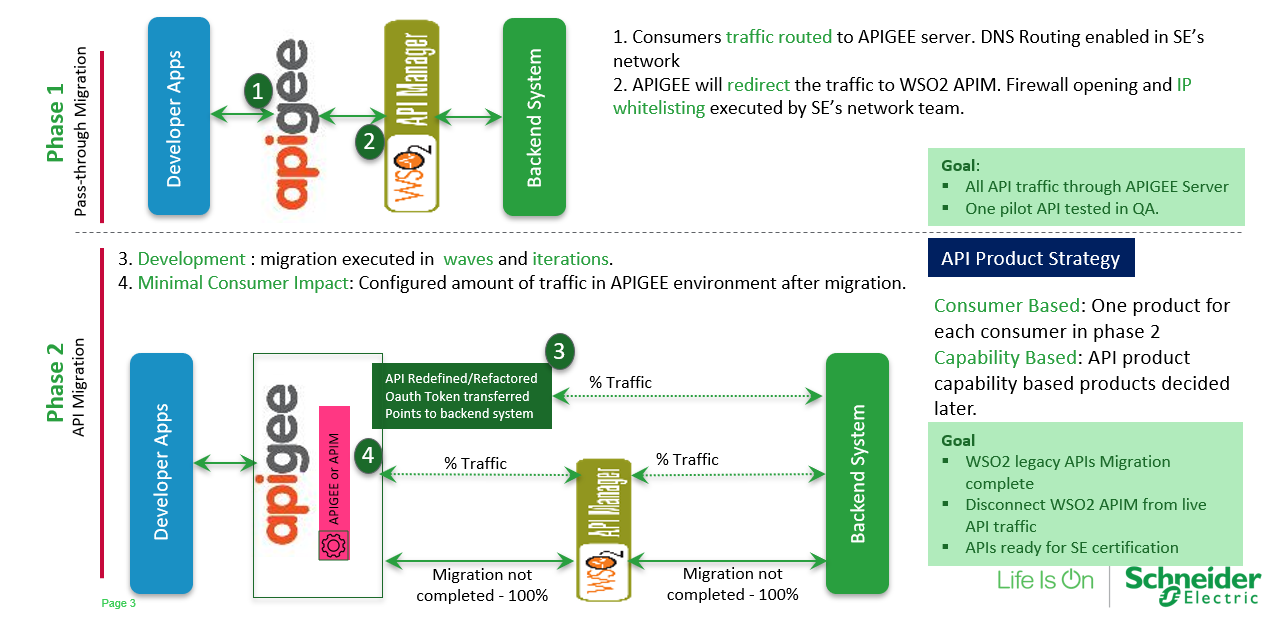
# **APIGEE Migration Steps**

Migration phase 1 and phase 2 are targeted to minimal/no impact to the consumers.

### Phase 1

1. APIGEE server will be added between developer apps and WSO2 APIM
2. APIM endpoint mapped to APIGEE so that consumers don’t need to change the endpoint.
3. APIM consumers key, secret and token will be migrated to APIGEE
4. All traffic flows through APIGEE as pass-through and APIGEE points to WSO2 APIM.
5. Oauth token validation and logic happens in WSO2 APIM as happens today.

### Phase 2

1. API logic and token validation migrated to APIGEE.
2. APIGEE sets the configuration to route the traffic through APIGEE or APIM.
3. Example: We may set only 5% traffic to APIGEE migrated proxy and rest of the traffic still using APIM flow. Un-migrated pass-through proxies still flows 100% in APIM.

# **Pre-Migration Activities**

### Network Connectivity:

* Security Approval for proposed cascaded Phase 1 architecture. Developer Apps 🡪 APIGEE (SaaS) 🡪 WSO2 APIM (On-Premise) 🡪 Backend (IFW ESB or System of Records)
* Security Approval for proposed Phase 2 architecture. Developer Apps 🡪 APIGEE (SaaS) 🡪 WSO2 APIM (On-Premise) 🡪 Backend and Developer Apps 🡪 APIGEE (SaaS) 🡪 Backend. If backend is in SE on-premise then APIGEE(SaaS) should have access.
* DNS Routing configured and pointing to APIGEE(SaaS) environment.

### Common Components:

* Common shared flow approved, Logging framework finalized and approved. CI/CD pipeline usage for each iteration agreed.
* Oauth grant type and token expiry setting are approved for migration APIs.

### Redefinition/Refactoring:

* Redefining/Refactoring logic finalized before starting any wave iterations. Late requests may require addition time to complete.
* There are 20+ APIs are under development in WSO2 APIM and needs to be planned for APIGEE migration.

### Collect Reusable Artifacts:

* WSO2 API Open API Specification, Detailed custom logic in agreed excel format, WSO2 API credentials and XSLT files.
* Valid Backend certificates which has to be imported in APIGEE SaaS environment

### Testing and Production deployment

* 20% refined test scripts (Postman files) to be shared. Performance testing executed only based on demand from consumers.
* Consumers will be notified before starting any iteration and feedback collected. Two reminders will be sent to consumers and if there is no response then considered as accepted for production deployment

# **Waves and Iteration**

300+ APIs are grouped into 3 waves and multiple iterations. Waves are created based on APIs impact and candidates for product API.

### Wave 1

Wave 1 primarily focus on simple and medium APIs with combination of Products APIs, high impact APIs and vice versa.

### Wave 2

Wave 2 APIs are complex in its implementation and potential candidates for refactor and redefinition.

**Refactoring:**

* Custom logics are optimized and migrated to APIGEE
* API policy naming standard and best practices will be applied.
* Dynamic routing implemented wherever needed.
* Token validation implemented in APIGEE.
* Migrate IFW light weight routing logic into APIGEE. Assumed 5% of APIs in its category will fall under this category.

**Redefinition:**

* Based on requirement, API Migration team will implement the changes in the request and response payload structure.
* Assumed 5% of APIs in its category will fall under this category.

### Wave 3

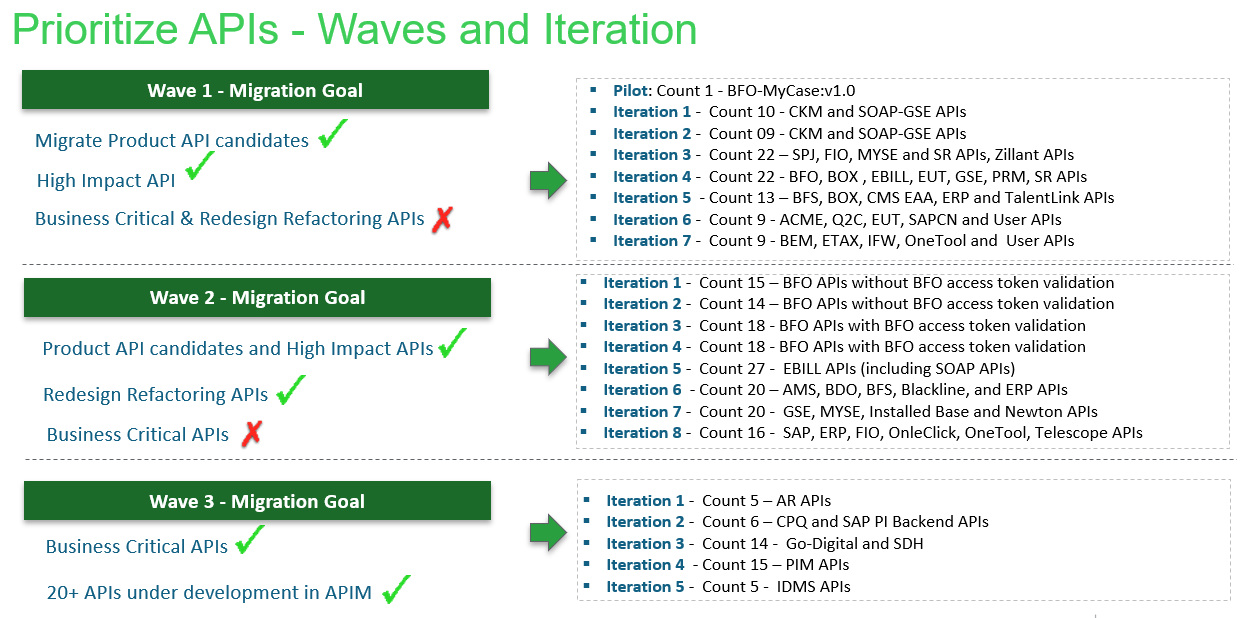
All business-critical APIs are migrated as part of this wave implementation. PIM, IDMS APIs are used by across SE and it needs more testing before moved to production.

### Pilot

BFO-Case:v1.0 API is considered for Pilot. Following activities will be completed as part of pilot process.

1. Common shared flow will be validated.
2. Security token migration will be checked in APIGEE environment.
3. Consumer Key/Secret migration will be validated.
4. API traffic and metrics validated between APIGEE and WSO2 APIM.
5. No consumer impact during the migration.

Below are the categorized waves and iterations.



# **Shared Flows and Common Components**

### Logging Components:

* API context and error logging framework will be developed and used across all migrated APIs.
* Logz.io might be used for logging and assuming that decision will be in place before starting the migration. If any delay in the decision may lead delay in production movement of migrated APIs.

### Security Framework:

* Oauth security token validation across all APIs.

### Shared Flow:

* Throttling will be applied in each API level using Spike Arrest Policy
* Quota will be applied in API product level.
* Shared flow contains following policies. Oauth Policy, Assign Message Oauth Header remove, JSON threat protection/XML threat protection, Raise fault Policy.
* Proxy dial and access token shared flow for migration. This shared flow required only for migration.

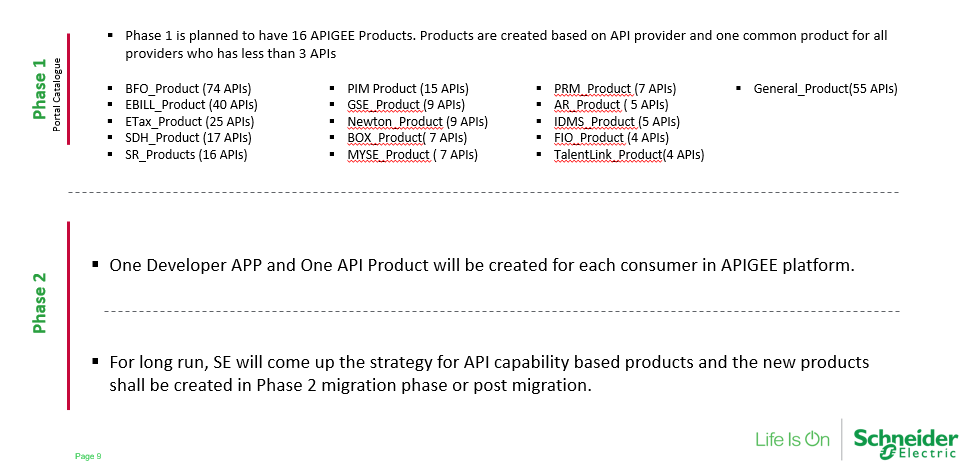
# **APIGEE API Product Strategy**

### Phase 2 – Product Strategy

Products will be created for each consumer. Consumer key and Consumer secret will be migrated from WSO2 APIM to APIGEE.

### Phase 2 – Long Vision

SE will up a scenario where products will be created based on APIs capability. Today we don’t have this information and it will be decided later.



# **Testing, Production Deployment and Rollback**

### Testing:

* Unit Test: Custom logic unit test will be performed by CI/CD pipeline or manual testing.
* System Integration Test: Executed in QA environment. API Migration team is responsible to execute the integration test.
* Functional Test: Tests executed in APIGEE QA environment by THBS team. THBS team will identify and extract the 20% functional unit test cases and execute the test in APIGEE QA and WSO2 APIM UAT environment. Test results are documented in word/excel document and sent to APIGEE platform owner for testing sign off.
* Performance Test: Executed to only selected APIs where consumers interested to execute the test. API team will only assist for the testing and test data and scripts preparing activities from consumer side.
* Environment Mapping: WSO2 UAT 🡺 APIGEE QA and WSO2 APIM Pre-prod 🡺 APIGEE pre-prod.

Production Deployment:

* Cut-Over Plan: API Migration team is responsible to preparing the cut-over plan at-least week before(?) the production deployment. Platform owner approves the cut-over plan.
* Production Deployment: Production CI/CD pipeline will be triggered for selected APIs.
* Production Smoke Test: Smoke test only executed when consumer has test customers to execute the smoke test in production.

### Rollback Plan:

* KVM Update: Configuration in APIGEE decides how much percentage traffic can flow in APIGEE and WSO2. In case of failure, configuration will set as zero traffic to APIGEE platform.
* Deploy Previous Revision: If the API running in APIGEE and additional functionality deployment failed in APIGEE then roll-back to previous working APIGEE revision.

# **Developer Portal Migration**

### APIGEE integrated portal

There will be multiple developer portal created for Schneider Electric internal/external users and partners.

**SE Internal Portal:**

Anyone from Schneider electric internal team can register themselves and access the API catalogue. There is no restriction will be applied in the portal and resources are open to anyone in the internal.

**SE Partner Poral:**

Only partner APIs are available in this portal and access is restricted only to approved partner developers.

Partner developer request for an access through developer portal and API portal admin will approve the access request. API and API product will be restricted based on the requirement.

### Drupal Portal

One Drupal portal will be created for SE internal/external and partners. Multiple UI pages and developer onboarding restriction will be applied in the Drupal portal.

Actual requirements will be captured from the community members based on the suggestions/requirements from APIGEE integrated portal.

# **Consumer Communications**

* It is challenging to get the latest contact consumer details and today we don’t have (don’t aware) updated consumer contacts stored in one place.
* Even-though APIGEE migration is happening without consumer impact, it is mandatory to give the platform change information to consumers at-least 15(?) days before.

**Yammer Community**

* Search the consumer groups in Yammer and inform about the migration activity and request them to participate in testing. If consumer wants re-modeling the APIs then capture the request in demand pipeline(?) and implement in agreed timeline.

# **Disaster and Recovery**

* Schneider Electric procured APIGEE SaaS for API platform which is hosted in Google Cloud Platform.
* Google provides 99.99% availability for APIGEE platform which is hosted in Google Cloud.
* APIGEE API proxy bundle is maintained by SE’s GIT Hub Repository.
* There is no infrastructure disaster and recovery in scope for Schneider Electric Migration team.

# **Migration Team – Daily Activities**

* Consumer communications for capturing specific requirement for migrated proxy
* Reverse Engineer the logic from WSO2 APIM to APIGEE with the help of THBS team.
* Map the reverse engineered logic to APIGEE inbuilt policies and create custom scripts in APIGEE wherever needed.
* Create a proxy with standard policies (using tool) or manual.
* Create shared flow if required.
* Create product and import API key/secret from APIM to APIGEE environment.
* XSLT transformation and integrate light weight custom logic in API proxy.
* Execute unit test and integration test.
* Communicate to backend team for challenges in connecting backend systems and resolve connectivity issues.
* Support with THBS team for FUT completion and test result document preparation
* Assist performance test-based requirement from consumers.
* Consumer communications for Production deployment

# **Execution Plan**

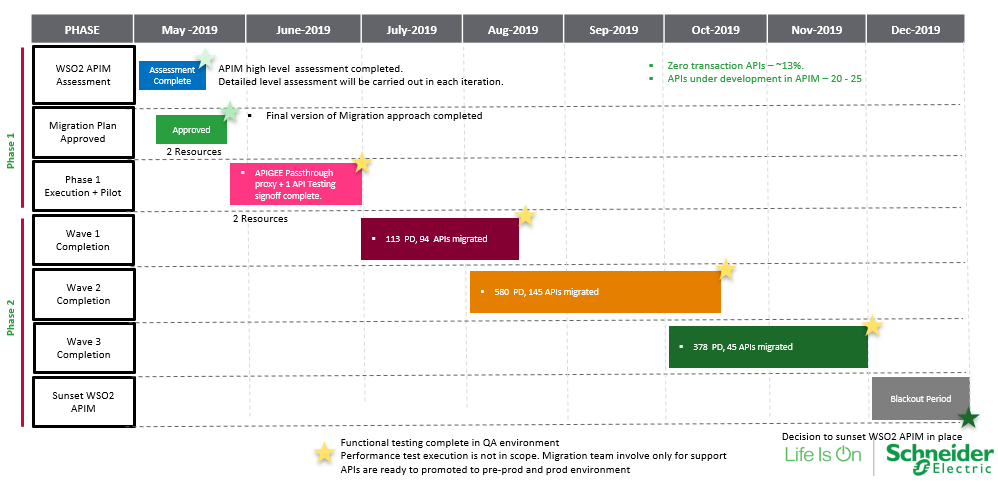
### Efforts Estimation

Efforts estimation calculated based on the technical complexity for migration, testing efforts involved and consumer specific requirements.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Activity** | **Simple** | **Medium** | **Complex** | **Comments** |
| **Reverse Engineering** |  |  |  |  |  |
|  | Document APIM logic | 1 | 3 | 4 | THBS Team effort. Not included |
|  | APIM logic KT | 1 | 3 | 4 |  |
| **Design** |  |  |  |  |  |
|  | Reusable Artifact Mapping | 0 | 2 | 4 | Proxy chaining Basic auth flow conversion Dynamic Routing Analysis XSLT mapping conversion Custom JAR Fault handling analysis |
|  | IFW Orchestration design to APIGEE | 0 | 0.5 | 1 | Refactoring effort - Not applicable for all APIs. Actual code refactoring/redesigning takes more time based on complexity and assumed 2 hours for complex interface. |
| **Development** |  |  |  |  |  |
|  | Pass-through | 2 | 2 | 2 | Proxy and policy creation |
|  | Header & Token Mapping | 1 | 1 | 1 | Token mapping logic implementation. |
|  | Payload Mapping | 0 | 4 | 8 | Medium : One or two mapping involved Complex: Multiple mapping involved. |
|  | Routing & Mash up | 0 | 2 | 4 | Less number of APIs has the Mash up logic. |
|  | Scripts(JS/Node) | 0 | 2 | 4 | Actual number of script file not known. |
|  | Security | 1 | 2 | 3 |  |
|  | Logging |  |
|  | Fault Handling |  |
|  | Threat Protection |  |
|  | Proxy Chaining / Aggregation |  |
|  | IFW logic migration to APIGEE | 0 | 1 | 2 | Simple Mash ups and routing to be migrated from IFW ESB to APIGEE. Madhu and Tejas to confirm the 5% APIs falls under this category. 8 hours to migrate for one API? |
| **Testing** |  |  |  |  |  |
|  | Unit Test | 0 | 4 | 4 |  |
|  | Integration Test | 1 | 4 | 6 |  |
|  | Functional Test | 2 | 4 | 6 | Effort is proportional to number of  operation in the resource |
|  | Performance Test | 0 | 1 | 1 | THBS team to confirm |
| **Other Activities** |  |  |  |  |  |
|  | Consumer Communications | 2 | 3 | 4 |  |
|  | APP and Product Creation |  |
|  | Key/Secret Migration for Products |  |
|  | Publish API to Portal |  |
|  | Total in PD | 1.375 | 4.8125 | 7.25 |  |
|  | Average effort per API |  |  |  |  |
|  | 10% effort reduced using APIMakr tool | 1.2 | 4.3 | 6.5 |  |

### Execution Plan

Requirement is to deliver all 300 APIs in before end of December 2019. This execution plan is prepared to complete all the migration work to be completed within 4 months from July-2019



### Resource Loading

Migration will be executed in onsite(USA) SE office and offshore(India) SE office. Resources will be onboarded in both the location.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Roles** | **19-Jun** | **19-Jul** | **19-Aug** | **19-Sep** | **19-Oct** | **19-Nov** | **19-Dec** |
| Onsite Lead | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Offshore Lead | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Onsite Sr.Developer | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Offshore Sr.Developer | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Onsite Developer |  |  | 2 | 2 | 2 | 2 |  |
| OffshoreDeveloper |  |  | 4 | 4 | 4 | 4 |  |

### Roles and Responsibilities

#### Onsite Lead

* Responsible for migration approaches and execution plan.
* Create common proxies and reusable components for migration activities.
* Collaborate with WSO2 APIM team for reverse engineering and get the knowledge transfer and document.
* Build migration proxies in APIGEE environment
* Execute integration test and unit test.
* Assist the functional test with THBS team.
* Test case sign off document preparation
* Promote proxies and higher environments
* Create products, app in APIGEE environment
* Consumer communications and align for performance test and delivery
* Publish the API to portal
* Developer management.

#### Offshore Lead

* Create common proxies and reusable components for migration activities.
* Collaborate with WSO2 APIM team for reverse engineering and get the knowledge transfer and document.
* Build migration proxies in APIGEE environment
* Execute integration test and unit test.
* Assist the functional test with THBS team.
* Test case sign off document preparation
* Promote proxies and higher environments
* Create products, app in APIGEE environment
* Consumer communications and align for performance test and delivery
* Publish the API to portal
* Developer management.

#### Onsite Sr.Developer

* Collaborate with WSO2 APIM team for reverse engineering and get the knowledge transfer and document.
* Build migration proxies in APIGEE environment
* Execute integration test and unit test.
* Assist the functional test with THBS team.
* Test case sign off document preparation
* Promote proxies and higher environments

#### Offshore Sr.Developer

* Collaborate with WSO2 APIM team for reverse engineering and get the knowledge transfer and document.
* Build migration proxies in APIGEE environment
* Execute integration test and unit test.
* Assist the functional test with THBS team.
* Test case sign off document preparation
* Promote proxies and higher environments

#### Onsite Developer

* Collaborate with WSO2 APIM team for reverse engineering and get the knowledge transfer and document.
* Build migration proxies in APIGEE environment
* Execute integration test and unit test.

#### OffshoreDeveloper

* Collaborate with WSO2 APIM team for reverse engineering and get the knowledge transfer and document.
* Build migration proxies in APIGEE environment
* Execute integration test and unit test.

# **Assumptions and Risks**

* We assume that security approval along with firewall opening, IP Whitelisting and DNS routing for proposed architecture in place on or before May 31st, 2019. Delay in securing approval may impact the migration timeline.
* Assume that latest consumer contact details provided on or before May 31st, 2019. Migration team inform to consumers before starting phase 1 migration in Dev/QA, pre-prod and production environments.
* THBS team provides reverse engineering logic in agreed excel format. Assume that documents not change once agreed.
* Valid consumer key/secrets exported from WSO2 APIM to APIGEE.
* APIs which is required for re-definition may require addition implementation time based on requirements from consumers.
* APIGEE lead and senior developers will be working for this
* Assume that test data and test user will be ready in the functional testing phase.
* THBS team’s support is secured throughout the migration. Amount of support will be agreed based on the waves and iteration plan.
* Blackout Period – December 2019 is considered blackout period for migration and assuming there is no other blackout period on 2019.
* THBS team executes the functional test in QA environment and THBS resource is secured for through-out the migration.
* Changes in the migration execution approach/model will impact overall timeline.
* Assume that valid SSL certificates provided to APIGEE migration team to connect backend system.
* 5% APIs are considered for refactoring and 5% APIs are considered for redefinition.
* Resource estimation for APIGEE integrated portal and Drupal portal is not included in this document.

# **Tools for Migration**

CTSAPIMakr tool will be used to migrate SE’s template-based APIs into APIGEE environment.

# **Sunset WSO2 Manager**

Once all the APIs are migrated to APIGEE, WSO2 APIM will remain isolated for some agreed duration before we sunset the WSO2 APIM environment.