Agile API Development Powered by API Connect

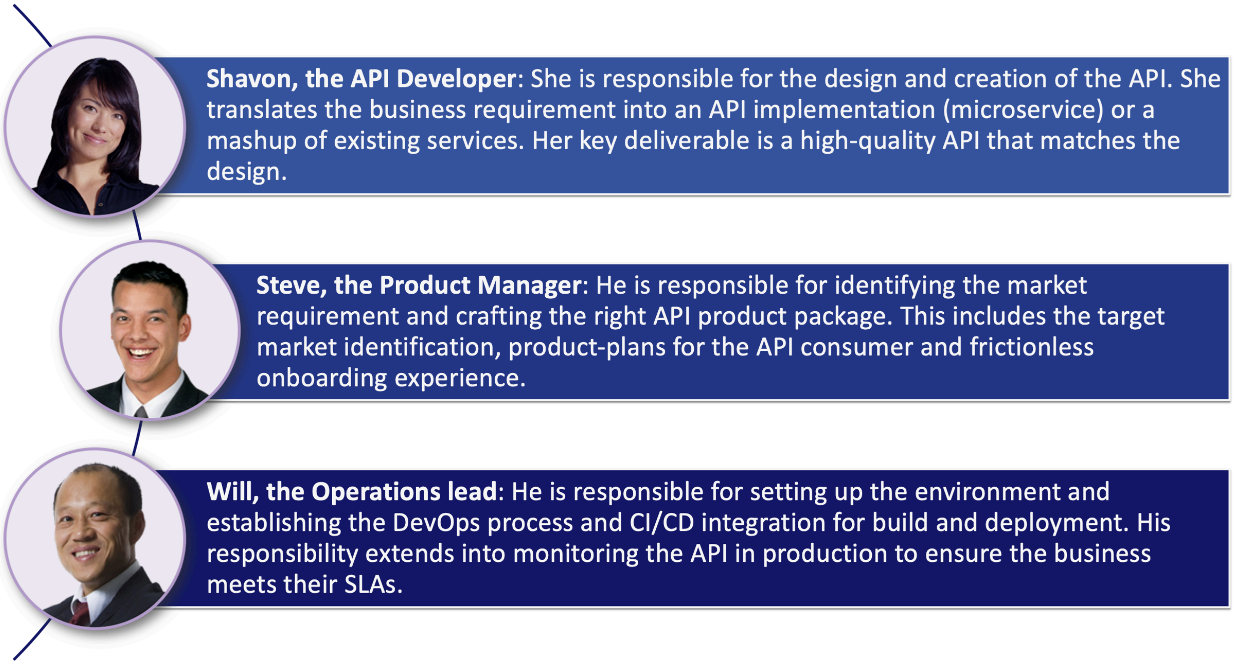
Agile API Development

Customer expectations and behavior are continuously changing. To deliver exceptional customer experience, a business must be nimble to adapt to these changing needs. Digital transformation projects are often the means of delivering this superior customer experience and technology (IT) has been the strongest enabler. Speed and agility have been the key tenets of a successful digital transformation project. Having said that, the IT organization has been seen as the long pole in the tent for delivering these projects on time. But over the last few years, CI/CD adoption and DevOps automation have assisted IT teams in becoming agile and adapting to changes quickly. This automation is manifested through a CI/CD pipeline. It helps automate steps in the software delivery process, such as initiating builds, running automated tests, and deploying to a staging or production environment.

API development and deployment are no exception and need to be simple and fast. API development should integrate into the existing CI/CD pipeline accelerating your build and deploy process leading to agile API development. A strong agile API development strategy needs to be complemented with the right set of tools, processes, automation, and control.

In this paper (blog), we address how to leverage the market-leading capabilities of API Connect to adopt agile API development practices.

Personas in API Lifecycle



Shavon, the API Developer: She is responsible for the design and creation of the API. She translates the business requirement into an API implementation (microservice) or a mashup of existing services. Her key deliverable is a high-quality API that matches the design.

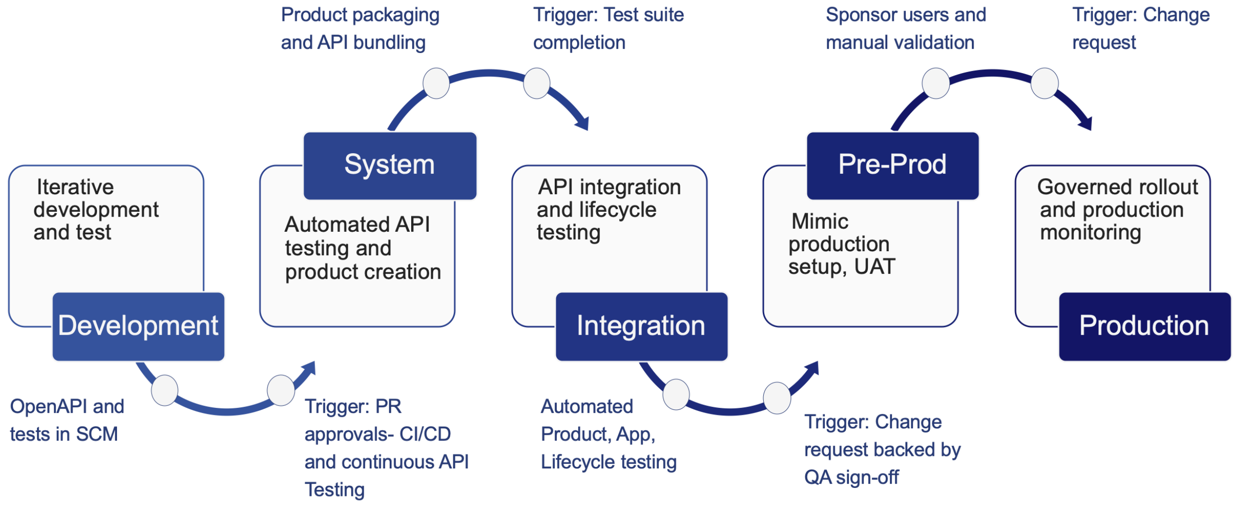
Steve, the Product Manager: He is responsible for identifying the market requirement and crafting the right API product package. This includes the target market identification, product-plans for the API consumer and frictionless onboarding experience.

Will, the Operations lead: He is responsible for setting up the environment and establishing the DevOps process and CI/CD integration for build and deployment. His responsibility extends into monitoring the API in production to ensure the business meets their SLAs.

Although there are more personas in an API lifecycle, the above list are the stakeholders that have the highest impact and participation in the CI/CD process.

API Promotion: From Development to Production

The diagram below is a typical representation of environments across which APIs are promoted as they progress through their lifecycle into production. Your setup may have more or fewer environments based on your business and governance needs. Regardless of your number and naming for the environments, you should be able to adopt the methodology.
Note: All other supporting artifacts in particular tests associated with the APIs are also promoted along this lifecycle.


Product packaging and API bundling
Trigger: Test suite completion
Sponsor users and manual validation
Trigger: Change request
Iterative development and test
Development
System
Automated API testing and product creation
API integration and lifecycle testing
Integration
Pre-Prod
Mimic production setup, UAT
Governed rollout and production monitoring
Production
OpenAPI and tests in SCM
Trigger: PR approvals- CI/CD and continuous API Testing
Automated Product, App, Lifecycle testing
Trigger: Change request backed by QA sign-off

Development

The focus here is to make the API developer, highly productive as she iteratively develops and tests the APIs. All of these can be installed on the developer's laptop providing a frictionless API Development Environment.

Here are the set of activities to be done at this stage:

1. Choose a bottom-up or a top-down approach for defining the API
    - Bottom-up: Import an existing OpenAPI definition. This may be provided by a microservice or other middleware or applications to address the business requirement. This could also be a SOAP service (WSDL definition).

- Top-down: Create a new OpenAPI definition based on the business requirements (resources, operations, etc.) and wire it to a microservice, existing middleware, or applications.

2. Define the API behavior by decorating the OpenAPI definition with the right set of policies for security, control, and transformation. For example, securing using OAuth, creating a mashup by mapping data elements, etc.

3. Test and debug the API including tracing the policy execution and optimizing it to meet the design requirements. To test the API, the API developer may be required to create mock backends.

4. Generate/create unit test cases to validate that the API is behaving as designed as the API gets updated through continuous development.

The above steps of Dev-Test are iteratively executed.

5. Once the API is ready, check-in and commit the API definition (OpenAPI) into the Source Code Management (SCM) system.

6. The pre-commit hook on the SCM would run a validation/linting check to ensure the API definition follows enterprise standards.

These steps are done by every API developer in the comfort and isolation of their laptop.

There are a few pitfalls to watch out for at this stage.

Common pitfalls	How to address them
API developers tend to focus on API products, packaging, plans, consumer subscription, etc. This slows down API development	Remove any friction for API development by bootstrapping any dependent resources
Setting up portal or analytics. This adds unnecessary resource burden and de-focuses on API development	Optimize resources so that only pre-requisites for API development are deployed
Working on a shared environment. This can lead to conflicts, resource constraints and hence can cause a delay in API development	During the early stages of development provide isolation by equipping the developers with the right set of tools

These concerns are important, but they need to be handled at a different stage and not interfere with the development of the API.

Agile API Development with API Connect

API Connect enables these imperatives with a wide range of capabilities to accelerate API development. These include

- ✓ API Connect offers a developer toolkit. Like any toolkit, this has multiple tools to make API development easy. The API Connect toolkit contains a user interface to design APIs (Designer), Local environment to test APIs (Local Test Environment) and a Command Line Interface (CLI) that developers love for quick access to resources. All of these can be installed on the developer's laptop providing a frictionless API Development Environment.

- ✓ API Connect also allows rapid microservice development using the Loopback framework. Loopback auto generates an OpenAPI definition for the microservice.

- ✓ Both bottom-up and top-down are supported in API Connect and accelerated through a wizard-driven flow.

- ✓ API Connect's intuitive drag and drop policy assembly reduces the learning curve and simplifies the API development.

- ✓ API Connect can automatically generate API test cases using the OpenAPI definition or from the data on the wire. This includes assertions for validating the payload, response codes, etc.

- ✓ API Connect makes the iterative process easy by bootstrapping the resources required to make the API operational. This removes any overhead from the API developers and allows them to focus just on the API.

You can read more on powering the API developer through an API Development Environment and the importance of developer isolation at early stages to accelerate development here

System Environment

In the system environment we want to enable automated API testing through continuous integration and deployment from the source code system. Also, at this stage product managers create products and bundle appropriate APIs based on market needs.

Here are the set of activities to be done at this stage.

For each successful Pull Request (PR) the CI/CD pipeline gets triggered and automates the following:

1. Creates a default product that references the API and publishes the product to an existing catalog
2. Runs a set of test cases against the API

3. Records the results, notifies the stakeholders of the results and of any failure
4. Removes the product and the API from the catalog

These steps are executed for every PR by every developer working with API Connect. The Provider organization (pOrg) and the Catalog remain the same. If there are multiple API teams, each team will have their own pOrg setup to which the respective APIs are made available and tested.

In parallel, the product manager

1. Creates API products by bundling the right set of APIs
2. Defines Plans (i.e. usage levels) through which this product can be accessed
3. Creates the target consumer groups and sets up the appropriate access control
4. Publishes the product and validate API and Product documentation on the developer portal
5. Pushes the product definition into the same repository alongside the APIs

These steps are done when a new product is created, or a product is updated.
Note: Products are offering packages separate from the API implementation and deserve the same treatment as that of the API.

There are a few pitfalls to watch out for at this stage.

Common pitfalls	How to address them
Focus on the API lifecycle and its implications like subscription maintenance etc. This translates into longer cycles of deployment and makes continuous testing very expensive	Remove any overheads that is not required for continuous API deployment and API testing
Manual publish and verification. This slows down API development and testing	Automation is key and validating the API behavior on every commit supports the shift-left strategy
Miss out on product packaging. Although this makes it easy for testing, skips a critical API strategy	Successful API programs have a strong API productizing strategy that goes beyond just a developer creating a product wrapper or using the default wrapper for the API

Agile API Development with API Connect

API Connect enables these imperatives with a wide range of capabilities around automation for continuous integration and deployment. These include

- ✓ To support CI/CD, both products and APIs are versioned resources within API Connect.

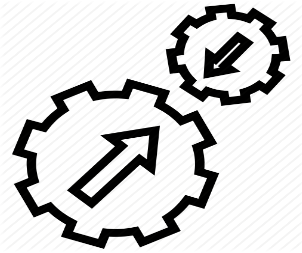
- ✓ For integration with the CI/CD pipeline, API Connect provides both a complete set of CLI and REST APIs for all functionality.

✓ API Connect is multi-tenant, separate environments for the product managers can be carved out if required on a single infrastructure at no extra cost.

✓ API Connect also provides a detailed dashboard of test results and quality progress.

Integration Environment

The Integration environment is a critical environment where different groups of APIs are deployed together, tested against governed backends, apps, consumers, subscriptions are simulated, and API lifecycle actions are validated. This environment also has spaces setup if they are used for catalog syndication. You can learn more about the Spaces here.



In preparation for this, the Quality Assurance (QA) team creates integration test cases by incorporating unit test cases from developers and enhancing them to include API sequencing, logical constructs, negative data sets, etc. You can learn about the importance of API Testing across the lifecycle here

Here are the set of activities to be done at this stage.

Typically, on a nightly build (ideally for every PR) or through a trigger from change request, the CI/CD pipeline automates the following:

1. The product definitions and the corresponding API versions are promoted to this environment
2. Apps, subscriptions are created to simulate consumers
3. Integration test cases are executed to validate all scenarios
4. API and Product updates are rolled out through appropriate API lifecycle stages
5. Perform load testing (You can conduct this on a separate environment if available)

If there are multiple API teams, for a single syndicated catalog, it should have spaces enabled in this environment (mirroring production).

There are few pitfalls to watch out for at this stage.

Common pitfalls	How to address them
Undersize the integration environment. Since all resources including APIs and Products are long-lived at this stage, the capacity could be a bottleneck	Even though it is a non-production environment, ensure it has the appropriate capacity based on automation workload and API resources
Ignoring versioning, Lifecycle transition and subscription maintenance. This could lead to unexpected downtime and loss of critical data	Govern the promotion and ensure the usage of lifecycle actions based on the stage of the rollout (new vs deprecation, etc.)

Common pitfalls	How to address them
Manually creating integration tests. This results in low test coverage due to lack of time and resources	Leverage test generation accelerators to reducing the time to create test cases

Agile API Development with API Connect

API Connect enables these imperatives with a wide range of capabilities for API testing and foundational API lifecycle capabilities. These include

✓ To aid in API testing, API Connect offers no code test creation through a drag and drop user interface.

✓ The intuitive user experience allows testers to easily build API sequencing, validate negative test cases with multiple data sets, etc.

✓ API Connect has the best API lifecycle capabilities in the industry with just enough governance. It supports hot-replace for bug fixes, supersedes for rolling out breaking changes, and co-publish for providing choice to consumers with multiple versions. You can read more on these lifecycle stages here

Pre-production Environment

The focus in the Pre-production stage is to prepare for production. The environment setup and configuration are like that of a production environment except for the backend that continues to remain directed to a test backend. This environment is also often used for user acceptance testing. In this environment, it is often a mixture of automated deployment and testing alongside manual verification.

Here are the set of activities to be done at this stage

Promotion to this environment is typically triggered when a change request for deployment is approved. The CI/CD pipeline automates the following:

1. The product definitions and the corresponding API versions are promoted to this environment
2. API lifecycle transitions are controlled and governed through approvals with integration to the change management system
3. Automated API testing and validation through a random test case execution.

In parallel, the Product managers and sponsor users manually (could be target consumers/ potential consumers):

1. Perform validation of the onboarding user experience
2. Build a sample app by leveraging the API by going through the documentation

The operations team leverages the API tests and extends the test suite for monitoring the API in production. This includes health checks and API behavior testing.

Agile API Development with API Connect

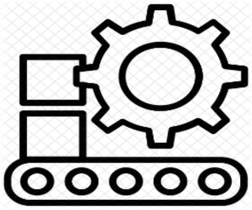
API Connect enables these imperatives with a wide range of capabilities for enterprise governance. These include

✓ API Connect provides a comprehensive set of hooks and plug points for integration with external systems such as change management for governance and control.

✓ The testing and monitoring capabilities of API Connect allow~~s~~ the operations team to collaborate with development and QA to identify the right set of test cases for production monitoring.

✓ Automate the promotion of portal customization across environments through a rich set of REST APIs and CLIs

Production Environment



The focus in the Production stage is to ensure controlled and governed promotion through automation while offering zero downtime to API consumers. As the APIs are deployed, the operations team should continuously monitor them to ensure the SLAs are met.

Here are the set of activities to be done at this stage:

1. On successful sign off from pre-production / UAT, the API is promoted to production through automated deployment
2. Conduct smoke tests to ensure the API is available
3. Operations team sets up synthetic monitoring across locations based on the geo-strategy
4. Operations team monitors for performance and API behavior anomalies through pro-active alerts

Agile API Development with API Connect

API Connect enables these imperatives with a wide range of capabilities for DevOps automation and monitoring. These include

✓ In line with modern DevOps approaches, API Connect supports blue-green deployment allowing API providers to test and roll out updates while providing maximum uptime to API consumers.

- ✓ Also, the operations team can schedule test runs from different locations and setup up alerts within API Connect.

- ✓ Autonomy with governance through syndication of multiple API providers while providing a single enterprise API portal for consumers

Conclusion

As you plan your agile API development strategy, for successful adoption ensure the following.

- Improve productivity by providing as much isolation as possible during the early stages of development
- Make API development frictionless by removing any distraction and allowing the developers to narrowly focus just on APIs
- Converge configurations, deployment as APIs are promoted into upper environments
- Accelerate go to market through continuous integration, testing, and deployment
- Provide the best consumer experience by not bypassing lifecycle stage transitions
- Just enough governance and control to provide the right check and balances during API promotion

For more information:

To understand more about IBM's thoughts on Digital Business and the API Economy visit the IBM API Economy website. IBM API Connect is IBM's complete foundation to Create, Secure, Manage, and Socialize APIs. You can find more information about IBM API Connect at the API Connect website. And you can also experience a trial version of API Connect