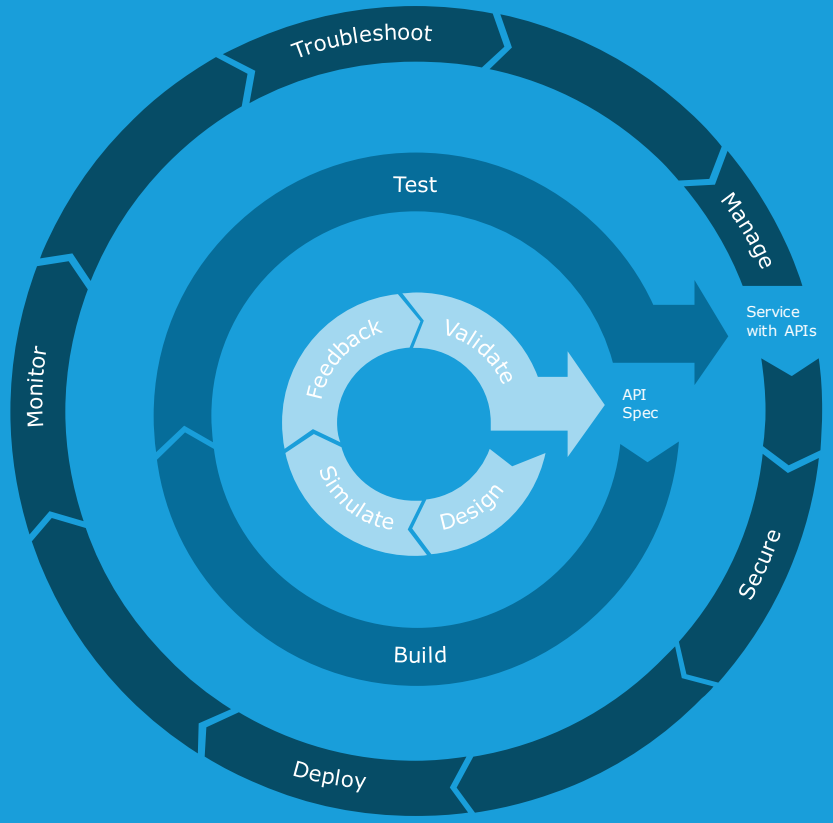
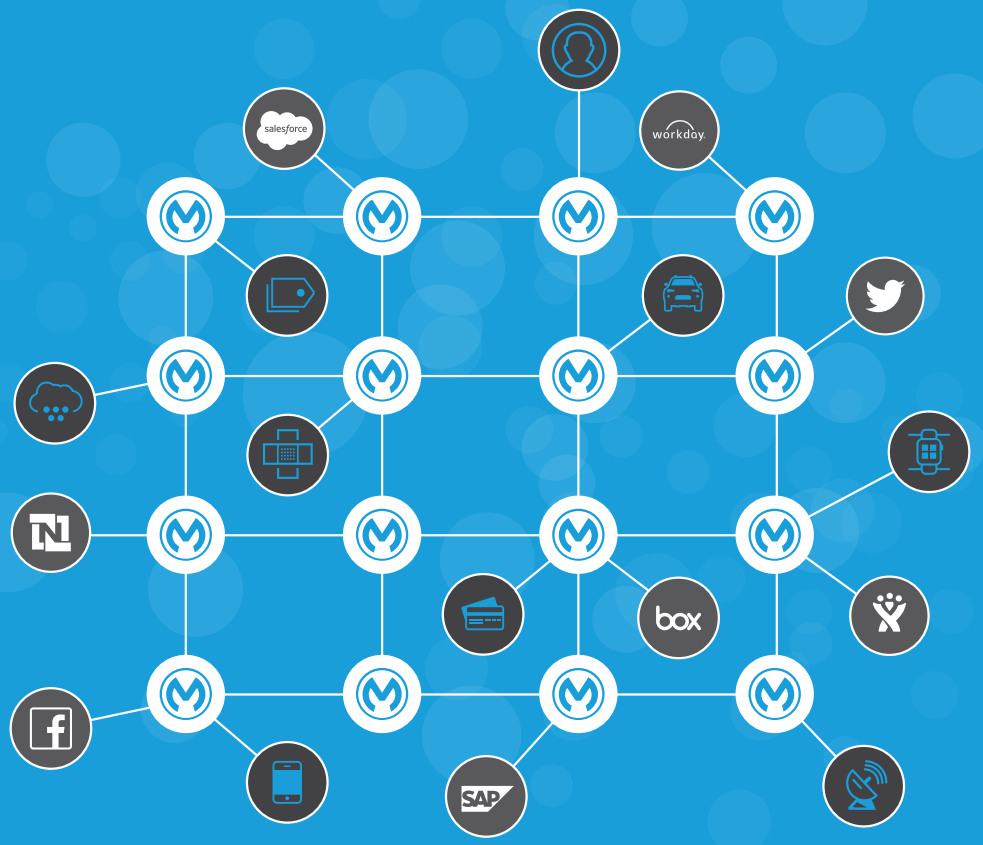
MuleSoft



Managing the full API lifecycle



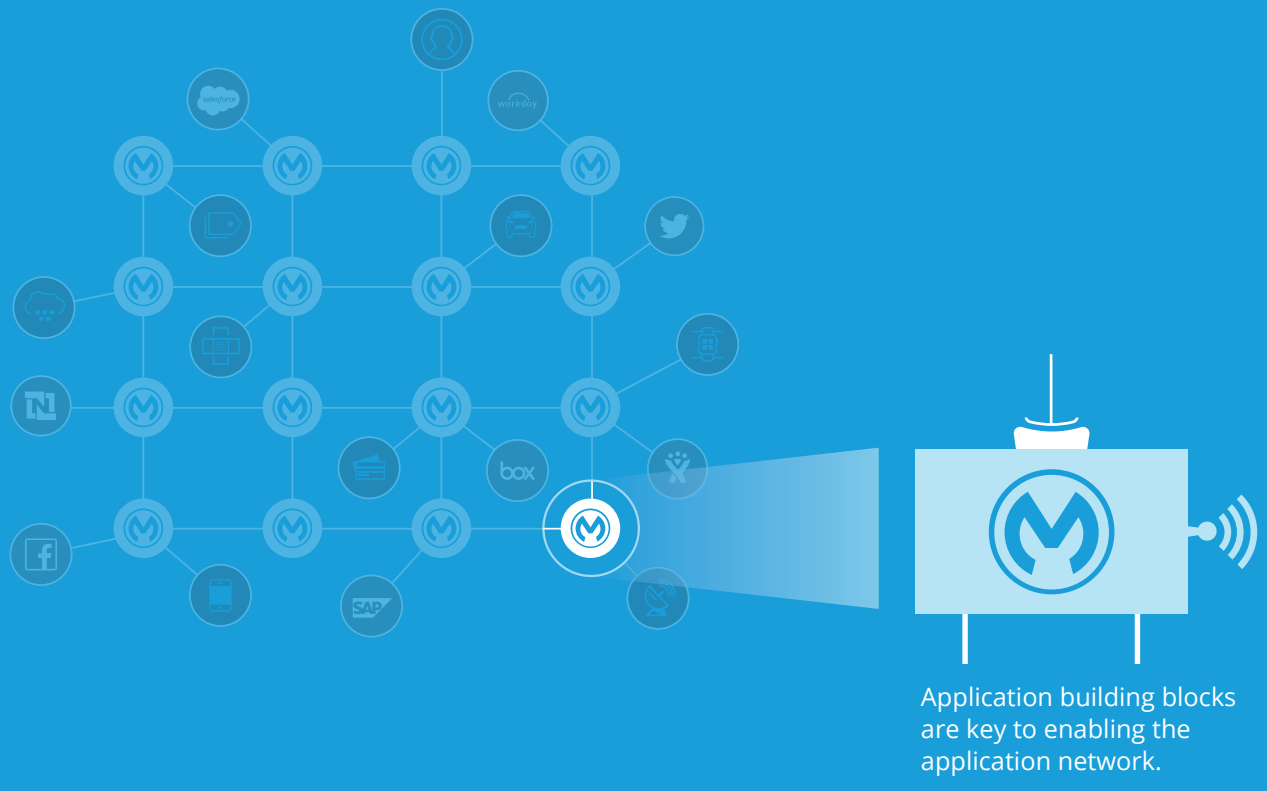
Introduction

In today's competitive landscape, businesses need to make decisions quickly; whether it's a new marketing campaign, a new product enhancement, a new partner portal or an employee productivity app, businesses are competing on speed.

What is needed is flexibility. And what better way to do this than to create purposeful, agile application building blocks that can be easily pieced together with well-designed, well-managed APIs to quickly create what is needed.

Connecting these building blocks results in a network of interchangeable functionality — a network of sub-functions, or applications. We call it the application network.

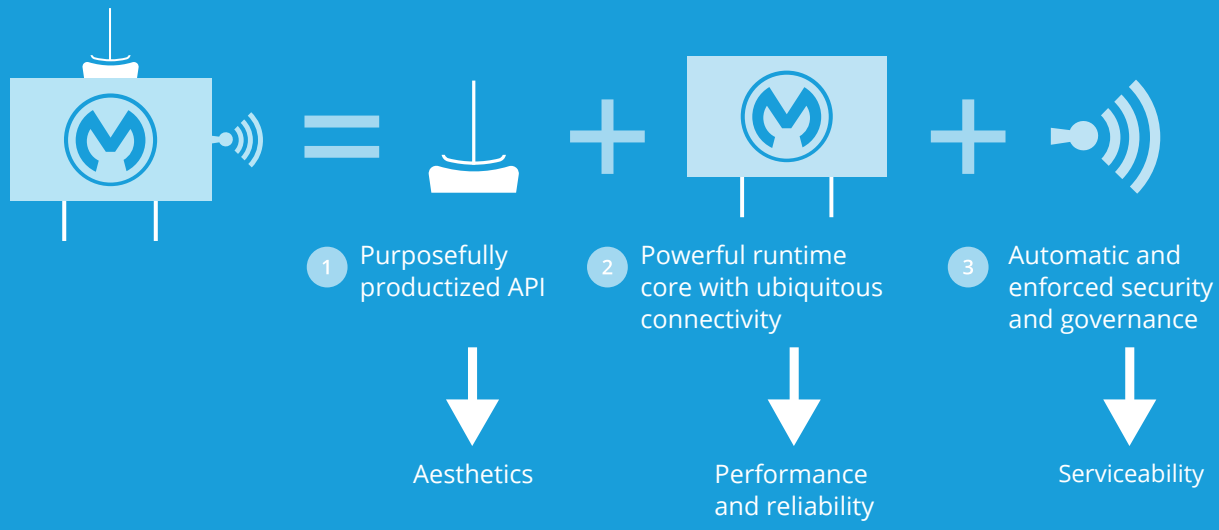
Let's future-proof ourselves and hedge against uncertainty by creating these building blocks, and then allow for the flexibility to rapidly piece them together on an on-demand basis.



Application building blocks are key to enabling the application network.

Welcome to the application network

The application network is the future. It emerges from the creation of multiple API-enabled microservices; connecting these microservices with a strategic API approach results in a composable network. The network allows the flexibility to rapidly piece together different services for multiple functionalities an on-demand basis, providing business agility and a robust platform for innovation.



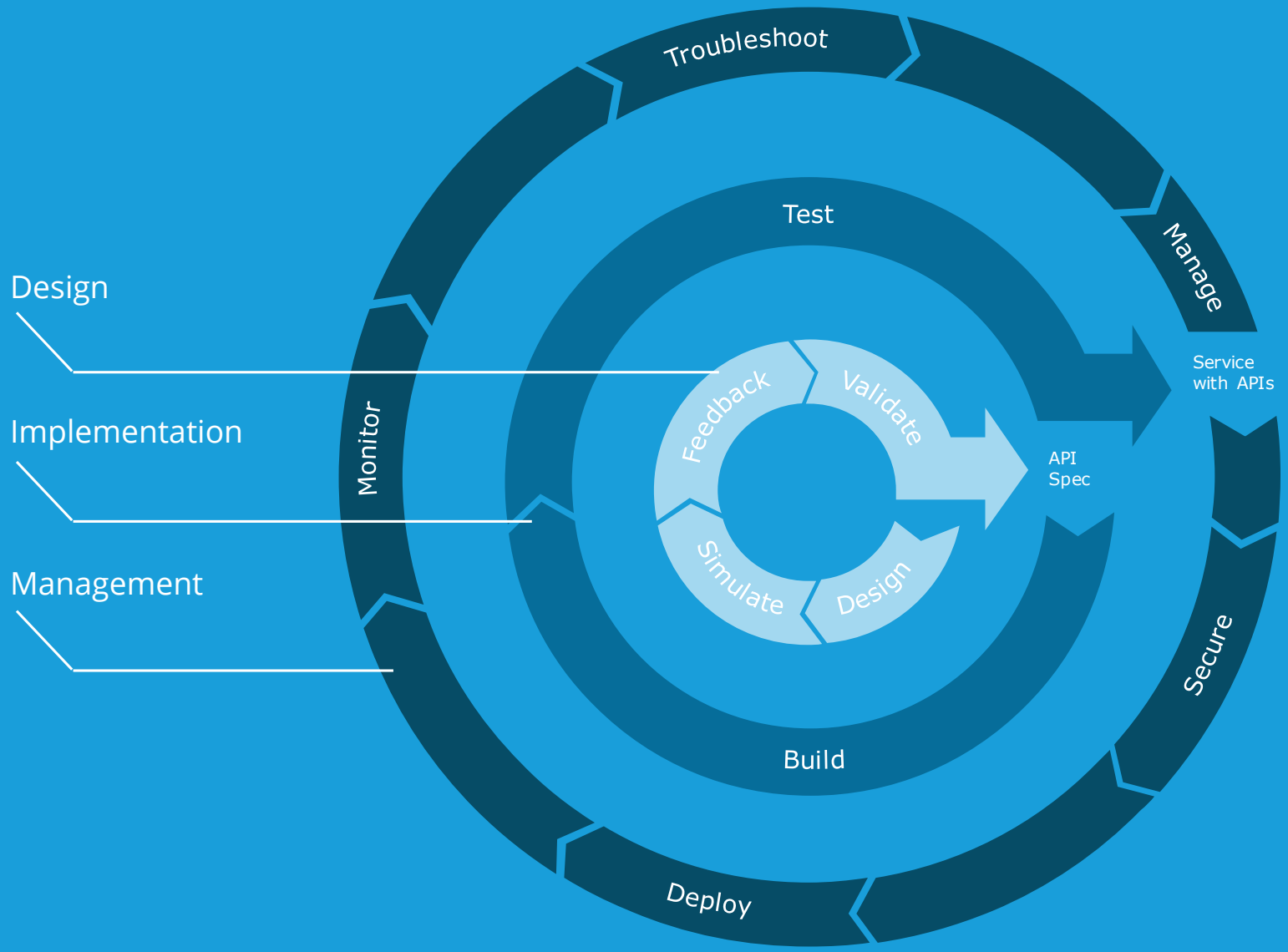
The anatomy of an application building block

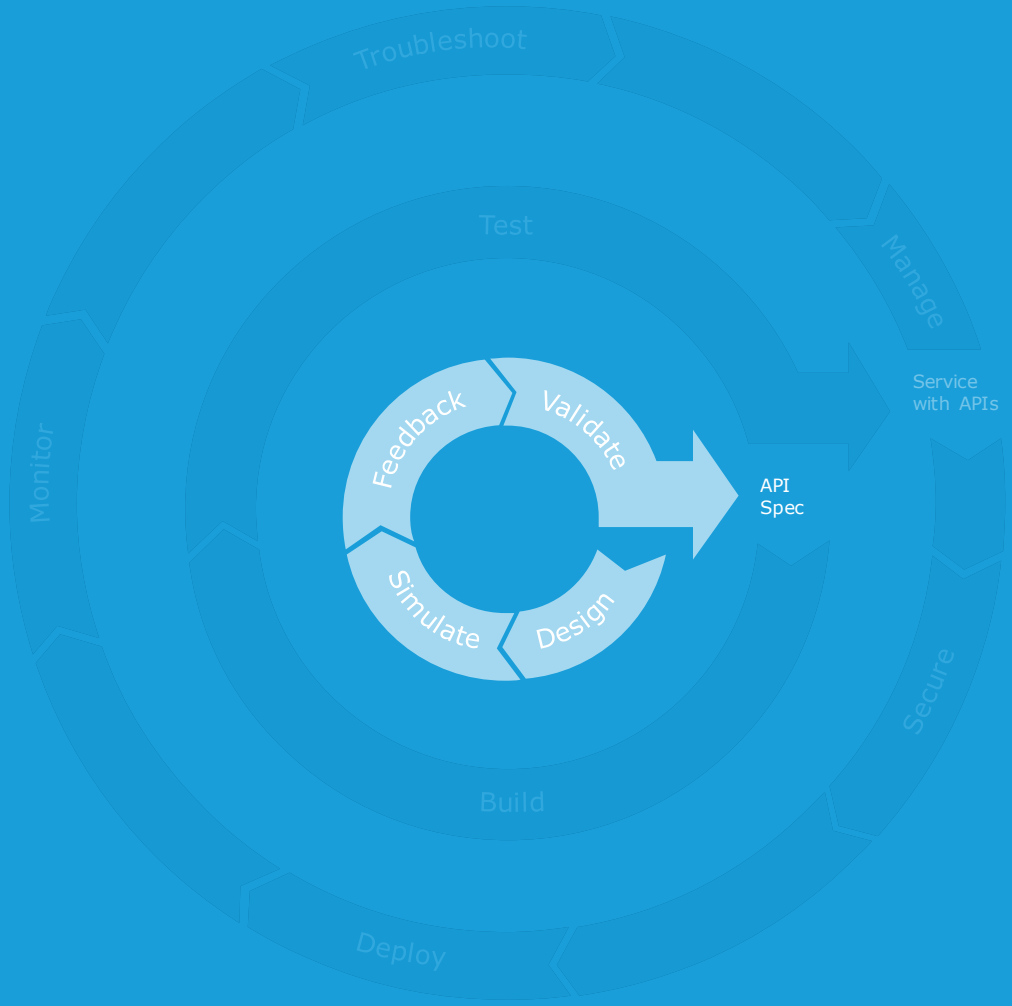
An application network is composed of application building blocks. These have multiple elements. and it's critical to separate the concerns between each. The API interface, the API implementation, and the API management aspects all have their own specific, unique lifecycles to follow. This building block should itself be treated as a product since these characteristics are common to what a good product should also have.

Therefore, it makes sense to treat a building block from a product-centric approach.

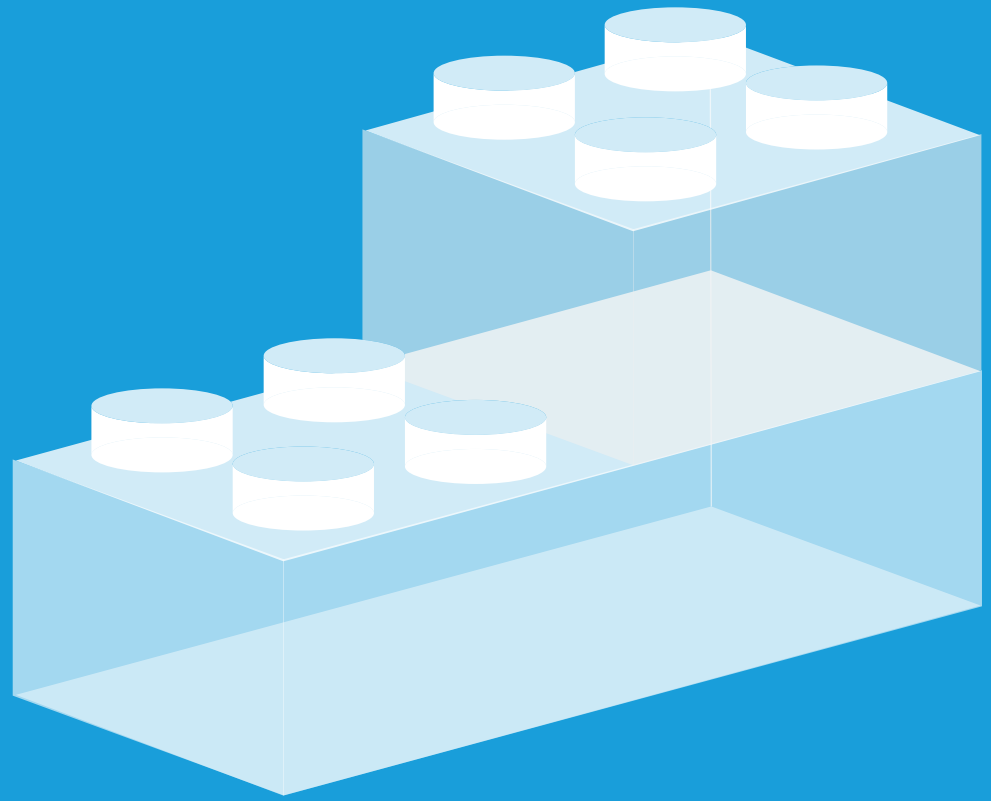
We see this product-centric lifecycle as having three distinct stages: design, implementation, and management.

The lifecycle of an application building block




Troubleshoot
Test
Manage
Monitor
Secure
Build
Deploy
Feedback
Validate
Simulate
Design
API
Spec
Service
with APIs

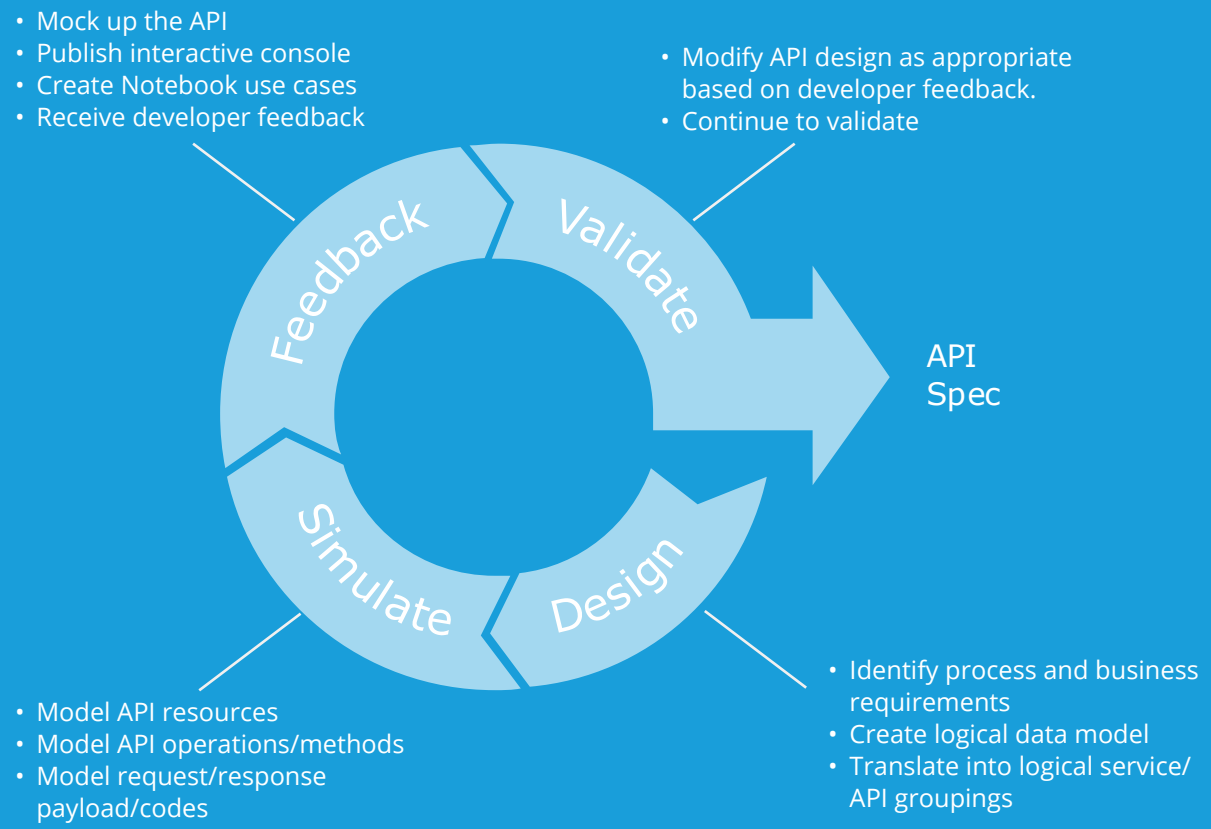
Design



How to design an API

Designing an API starts with an acknowledgement ahead of time that the API production process will start from an outside-in perspective, beginning with the “interface/contract” of the API. That is to say, first let us decide what the API will look and behave like before we actually begin to code the backend logic.

Often, developers build APIs without knowing the API has been validated and accepted. As a result, there is always this air of uncertainty: “is this what my consuming developers want?”



"Outside-in" done right

API developers must design the "user interface" of the API first – this is also known as the API contract. This approach is typically known as a "design-first" approach, and it should follow a deliberate API design lifecycle in order to optimize for the best API experience.

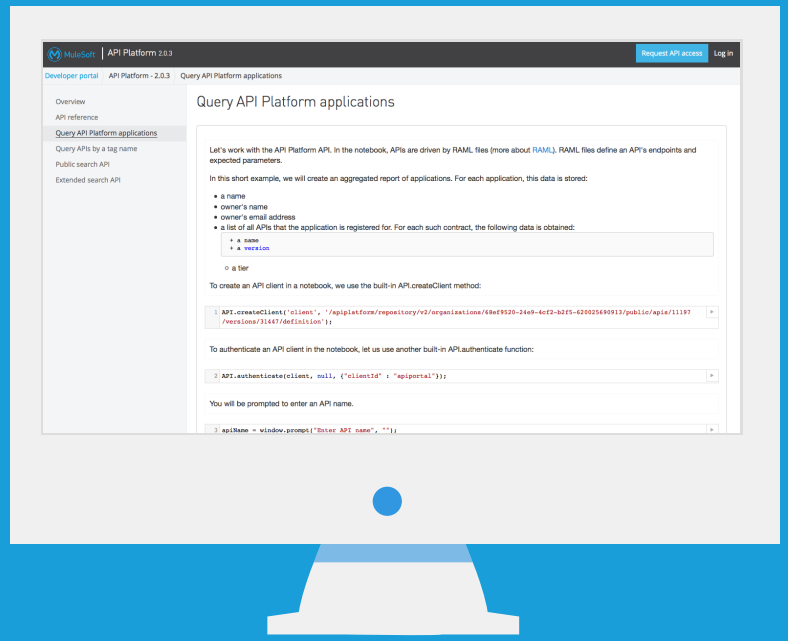
As a result, it is important to be able to do this in a human-readable fashion — to specify the contract in way that humans can easily digest.

| Tips | What is API Notebook and why is it important?

Think of API Notebook as the artifact to convey the inspiration for what is possible with an API.

It serves as a client application – it calls one or more APIs; it is therefore a live use case in action, where one can easily mash up multiple APIs, experiment, tinker and see what can be built.

Another benefit of having a Notebook is that it also serves as a sort of testing sandbox for an API.

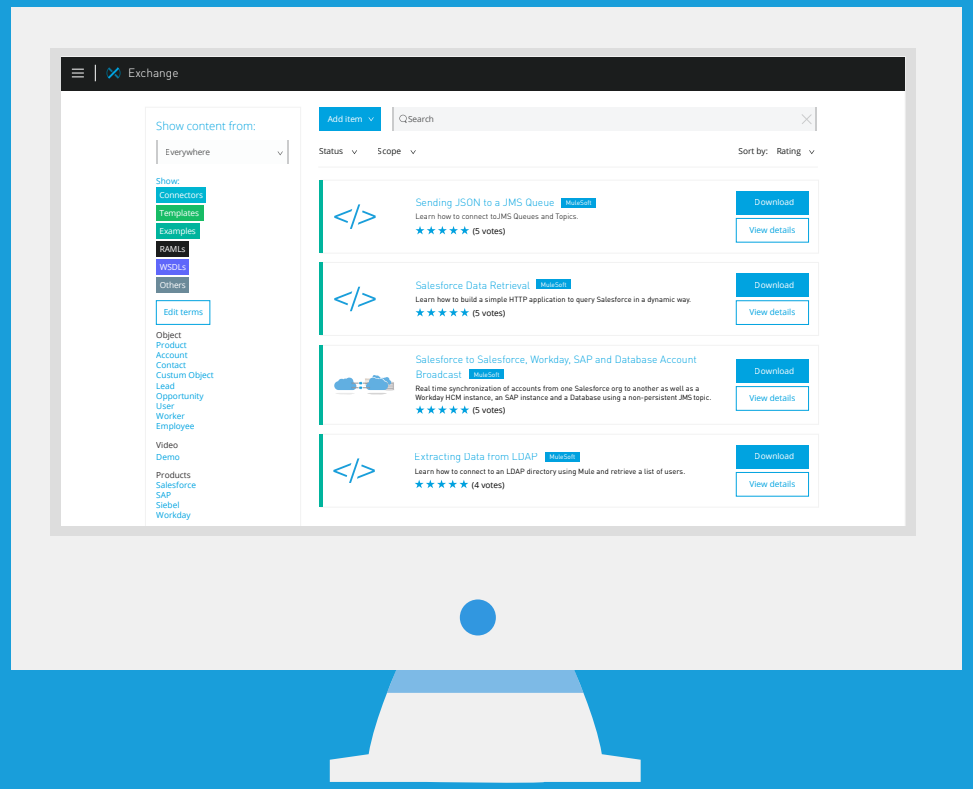


Soliciting feedback on the API design

At this point in the process, the API designer (i.e. the designer of the API contract) is ready to have the API be validated and tested by the API consumers (i.e. a client mobile app /website developer, or another API provider in some cases).

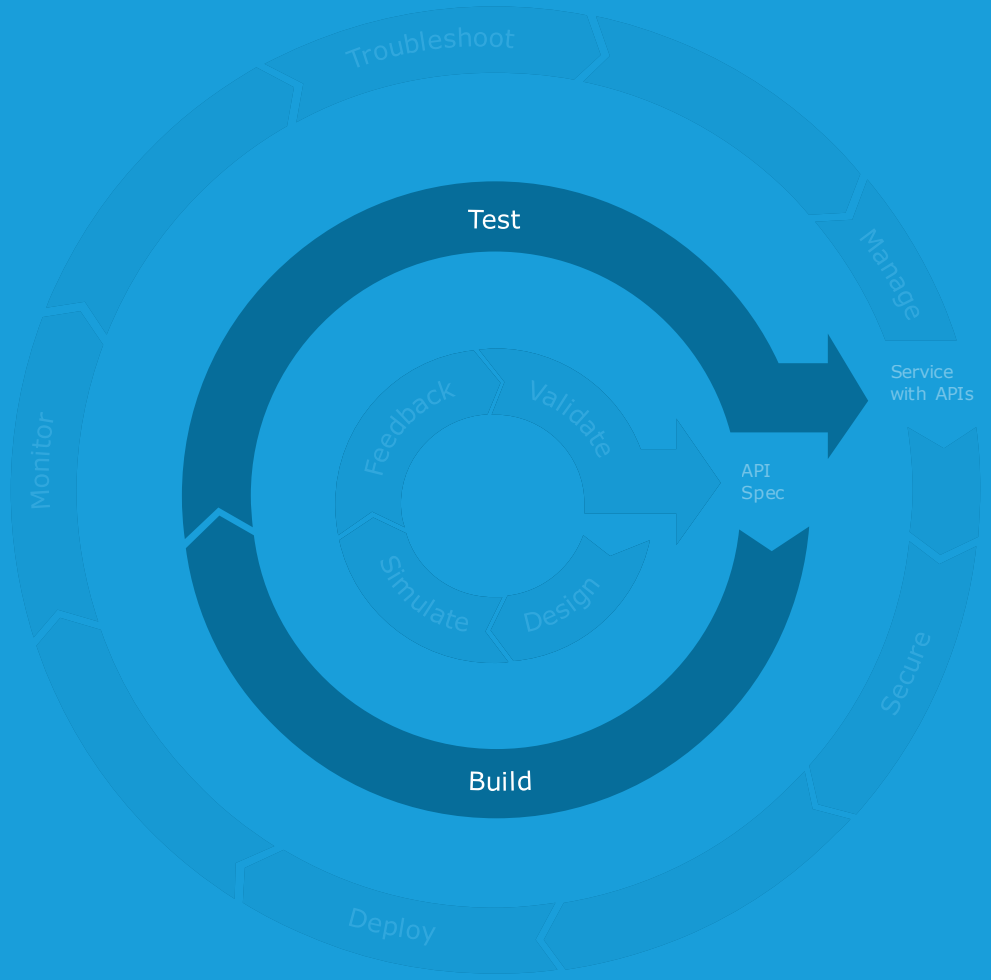
The currency of conversation between both parties will be via interactive tools such as the API Notebook and interactive documentation ,to name a few. There are many other tools that can be referenced from the raml.org site.

This process of validation may be brief or extended over numerous iterations.



Repeatable design

Any well-designed API will have repeatability in it as well as repeatability across other APIs. This can easily be encapsulated into best practice patterns both at the structural level of the API (nouns /resources), as well as at the method level (verbs) . So as API developers go about the design process, it is important to be able to discover and share repeatable patterns.



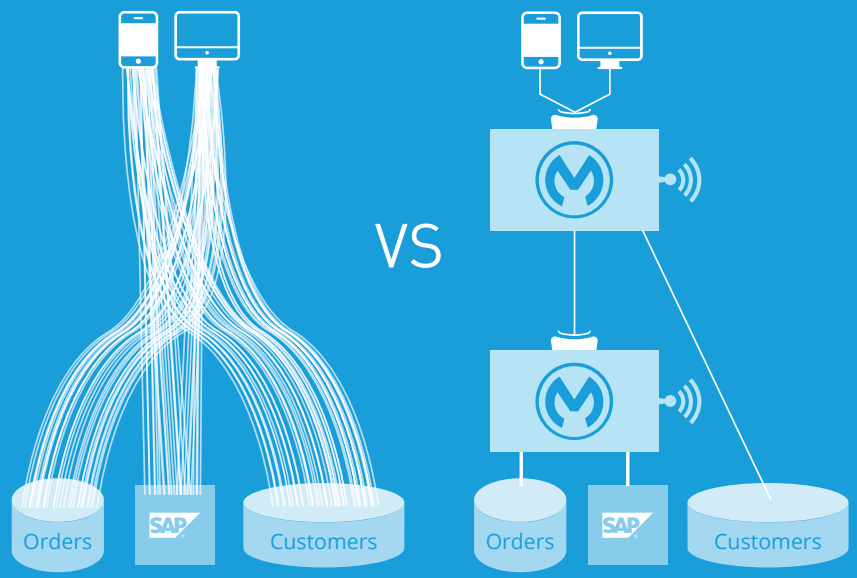
Implementation

| Tips | What do we mean by 'systematic' in this case? We see it as having the following architectural patterns easily available to the API developer:

- Orchestration
- Transformation
- Routing
- Data mapping
- Connectivity to popular SaaS, on-prem systems and data, file and other protocols

It also must be:

- Out of the box
- Pattern based
- Extendable and based on best practices



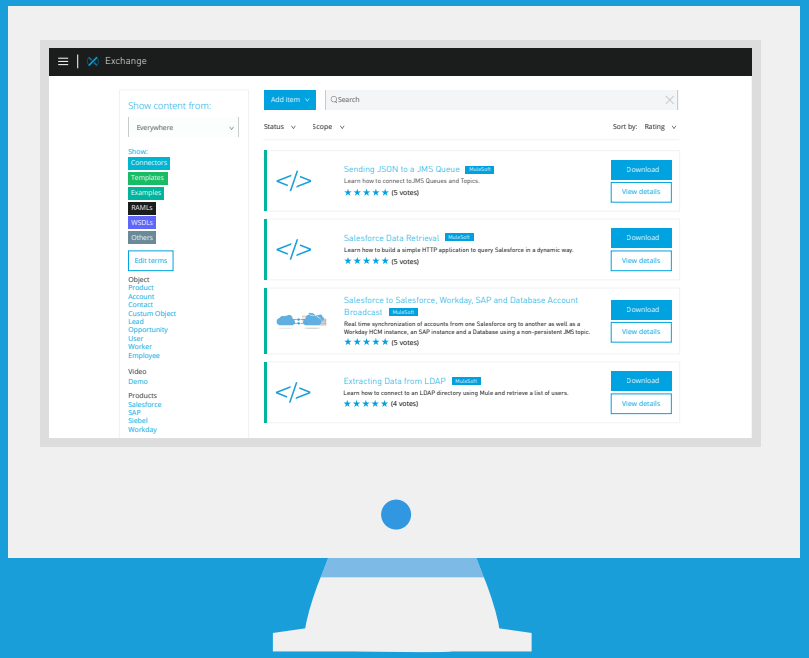
Connect systematically, not in an ad-hoc fashion

API implementation is a critical piece of enabling a next generation enterprise. Enabling for dozens, potentially even hundreds or thousands of APIs to be connected down to a backend and connected to each other will be key.

This must be done in a systematic manner (as opposed to point-to-point code).

| Tips | Best practice patterns to be leveraged when creating new building blocks.

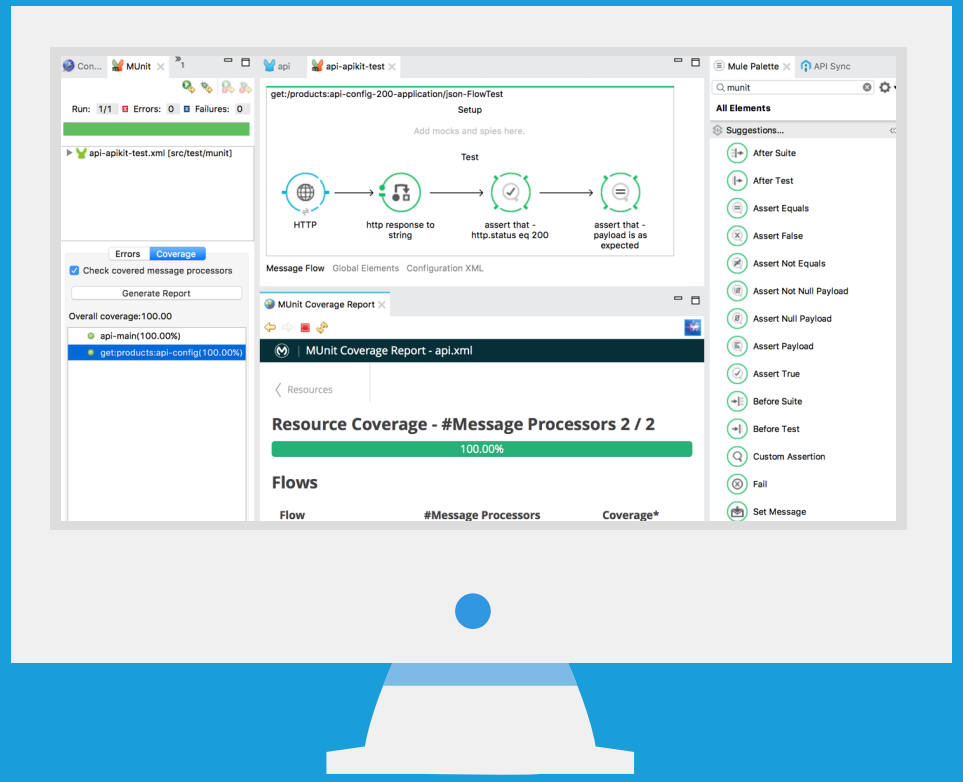
- popular database calls
- most commonly done transformations
- most popular workflows across multiple systems
- connections to popular SaaS and on-premise endpoints
- REST-SOAP transformations



Put API design principles and best practices in a common repository

Benefits of a best practices repository:

- Increase business agility
- Share best practices with reusable templates and logic
- Leverage best practice patterns
- Rapidly deploy APIs: fail fast, succeed faster
- Minimize point-to-point logic, and future proof for stability

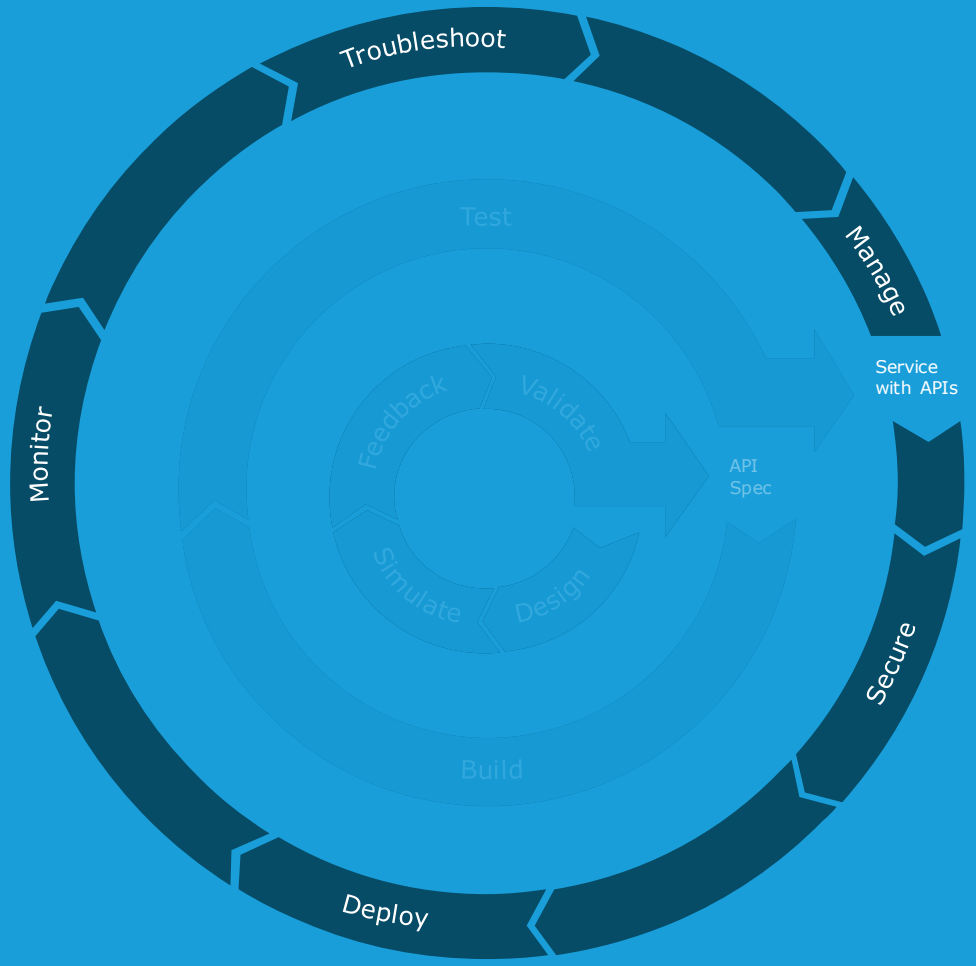


Testing the API implementation

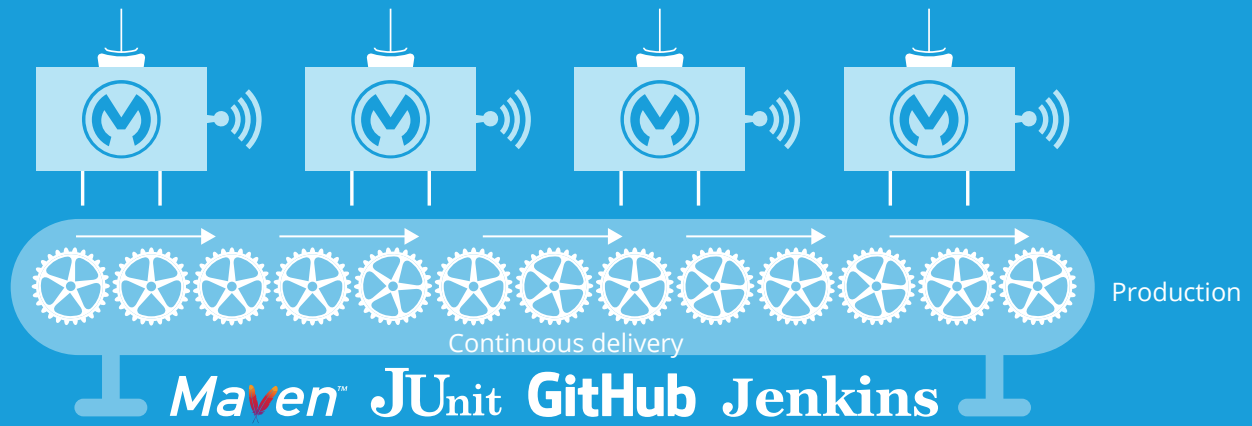
At this point in the process, the API provider (i.e. the developer behind the API implementation) is ready to have the 'guts' of the API tested.

MUnit is MuleSoft's testing solution, which is incorporated into the full application building block lifecycle.

Test automation tools are critical here, as this integrates into the DevOps processes of continuous delivery and deployment.


Troubleshoot
Manage
Service
with APIs
Secure
Deploy
Monitor
Test
Feedback
Validate
API
Spec
Simulate
Design
Build

Management



Embrace DevOps

Embracing modern DevOps-centric processes and tooling is critical to reduce mean time-to-production, and this should apply to your application building blocks as well.

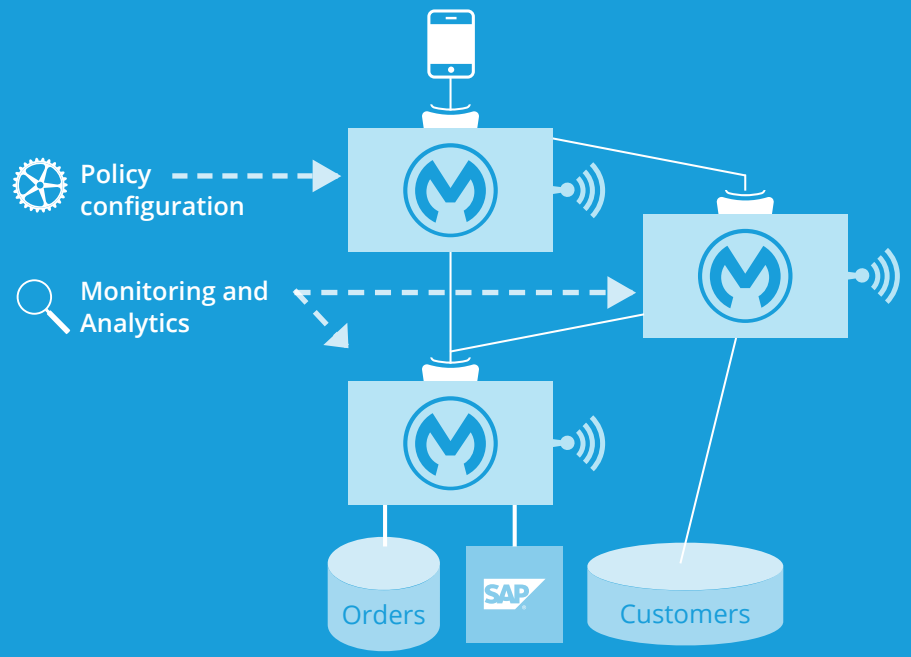
Once the application building block has been assembled and tested, deployment should be as easy as the click of a button.

The use of a hybrid integration platform that is lightweight, easy to install, and suitable for CICD workflows is key. The ability to have seamless support for dependency management, testing, version control, and automated deployment tooling should be an assumption.

| Tips |

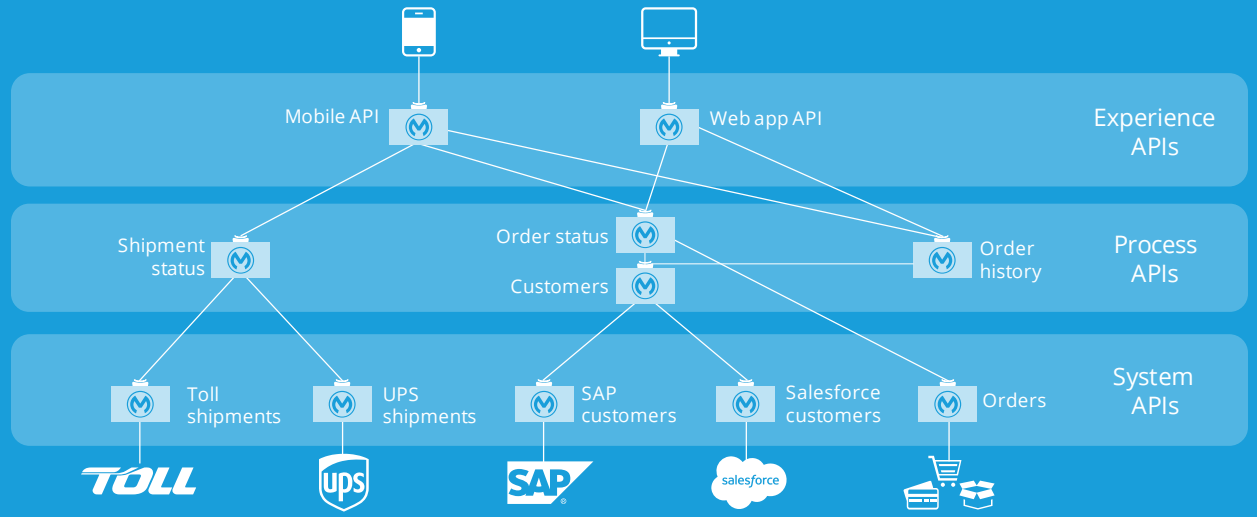
Policy configuration
Examples
- Traffic management (eg, rate limit)
- Access Policies (eg., OAuth2)
- Identity policies
- Custom policies

Monitoring and Analytics
Examples
- Infrastructure logs
- Service uptime analysis
- Client consumption data
- Provider analytics



Govern and secure all traffic

It is critical to ensure your application building blocks are following best practices in security and architectural governance by applying policies to them at runtime. Monitoring all traffic is equally critical because it just takes just one weak link to bring the ship down.

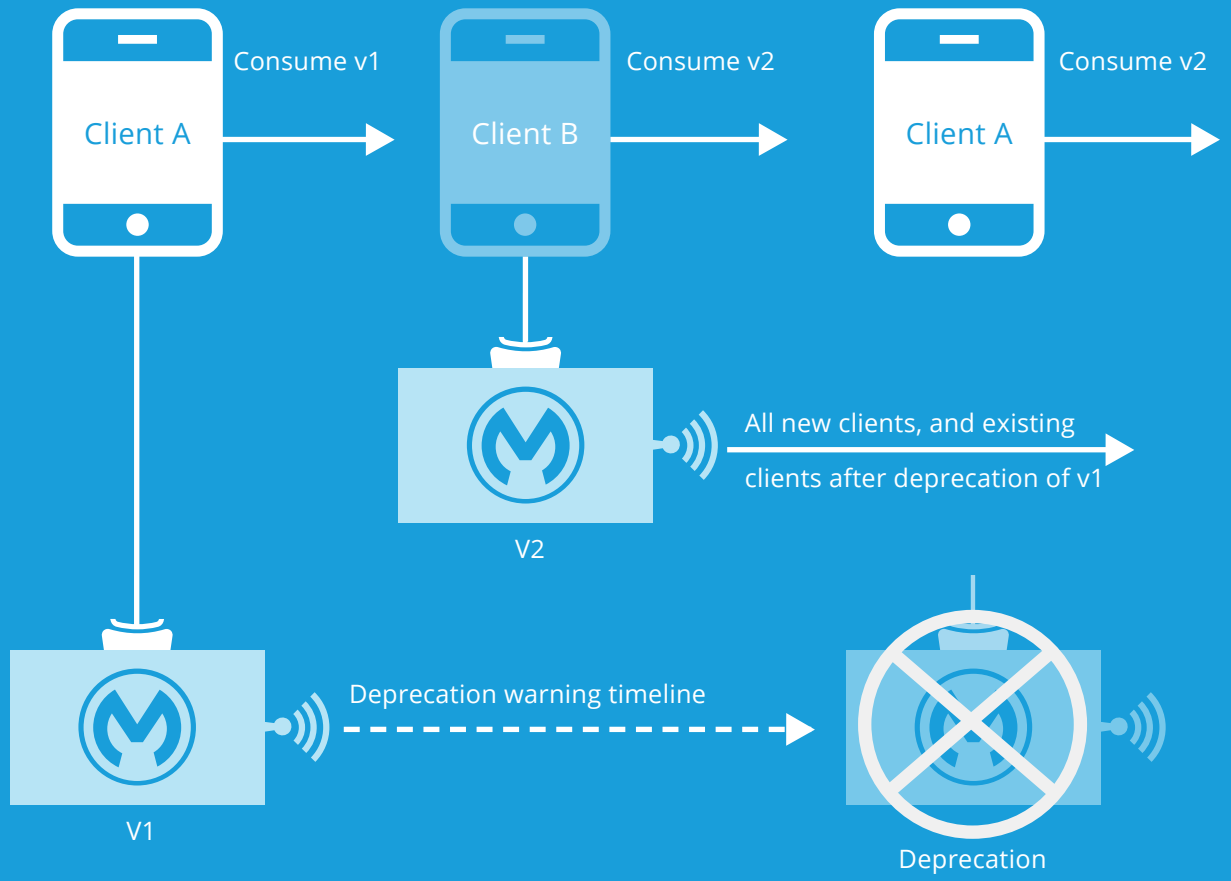


Don’t forget about the discoverability and onramp

Imagine your company with hundreds — if not thousands — of APIs in your expansive application network. Imagine you're adding several new ones every day.

Being able to appropriately publish them so the consuming developer can find, research, and understand them easily could make or break your entire program.

There is no point in building something that won’t ever be found, let alone used.

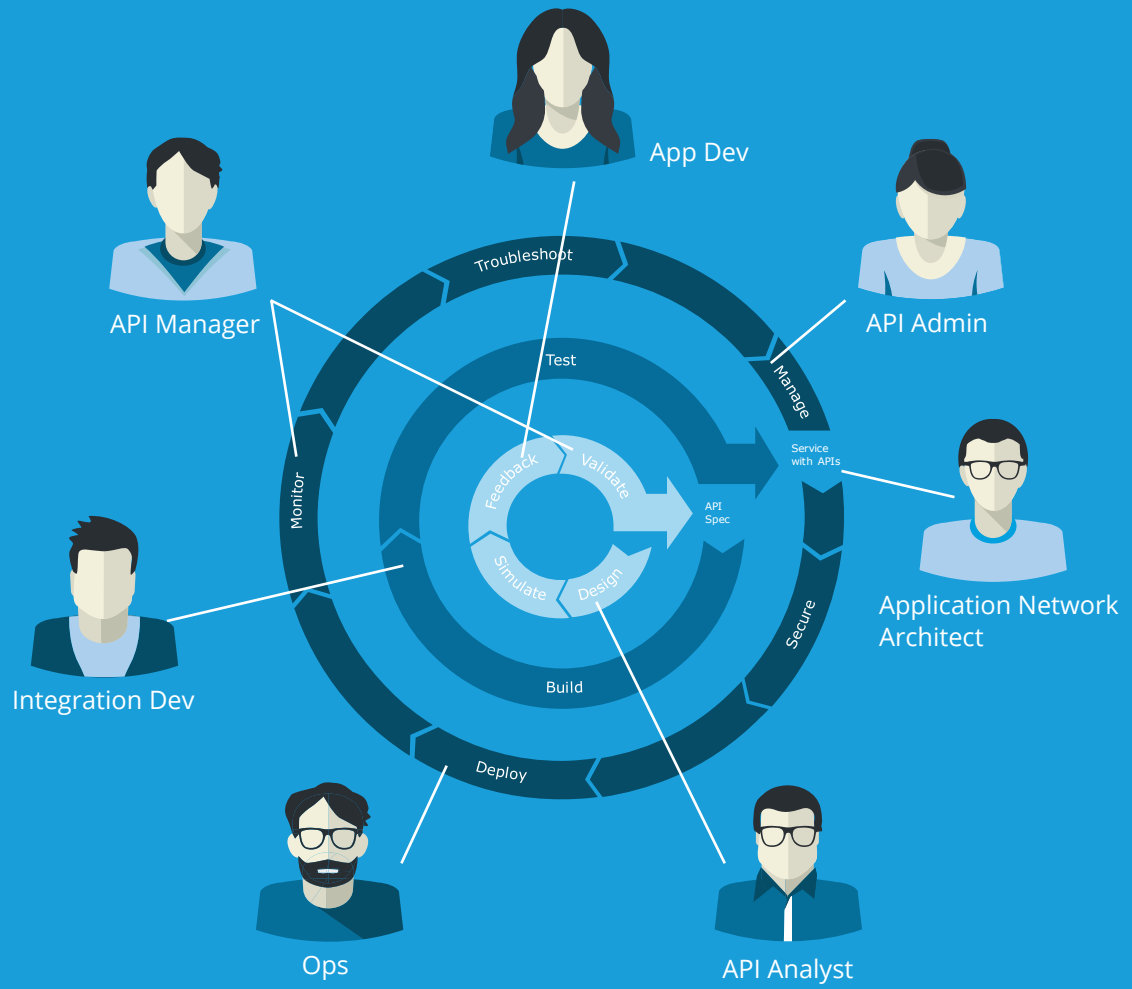


Just like any product, application building blocks change

Building blocks will change. It's a WHEN, not an IF.

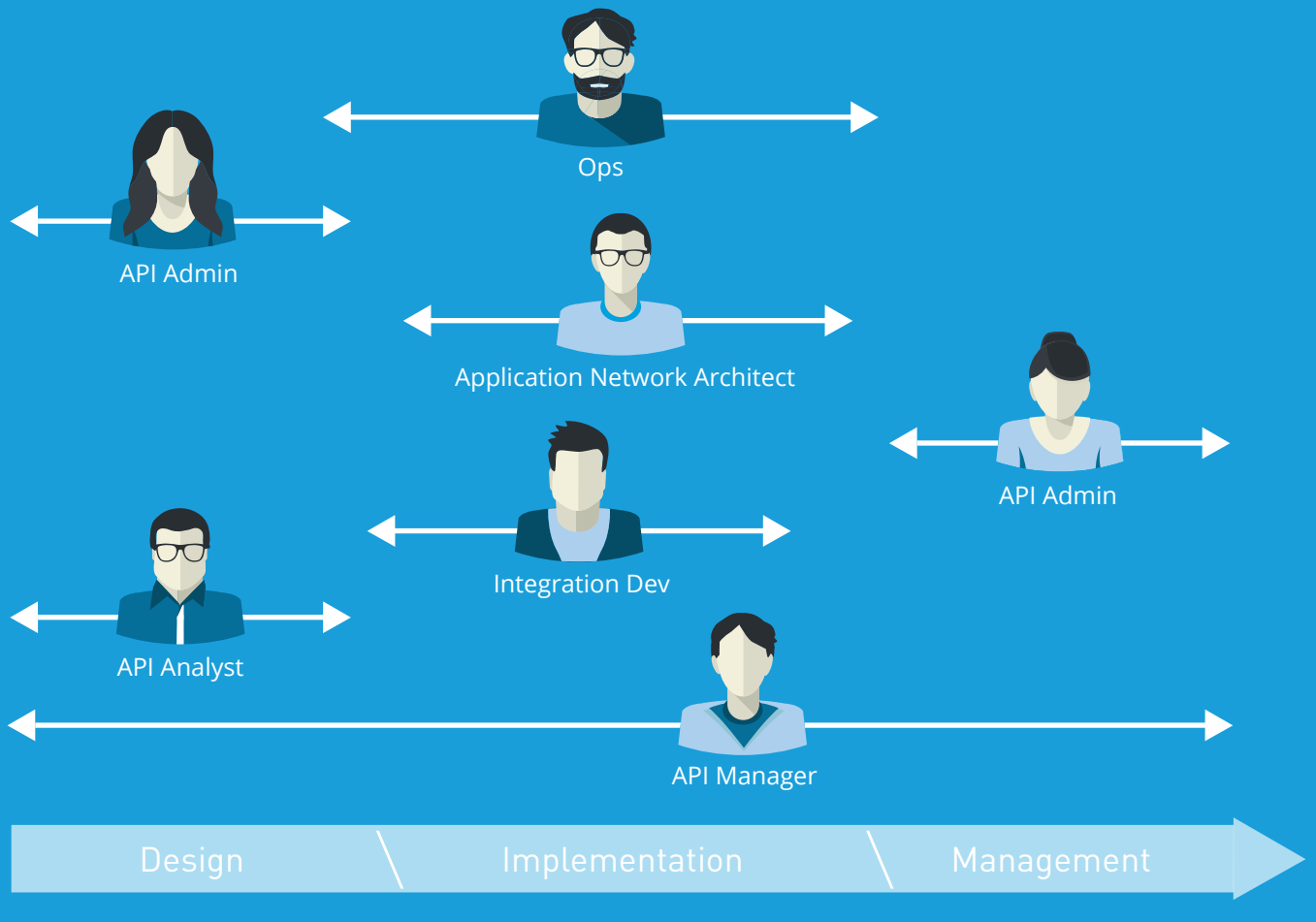
So be ready for it, which requires a carefully planned set of policies, procedures and the right platform to seamlessly migrate clients across new versions of APIs.

Getting this migration wrong will affect your customers. That's not a risk worth taking.



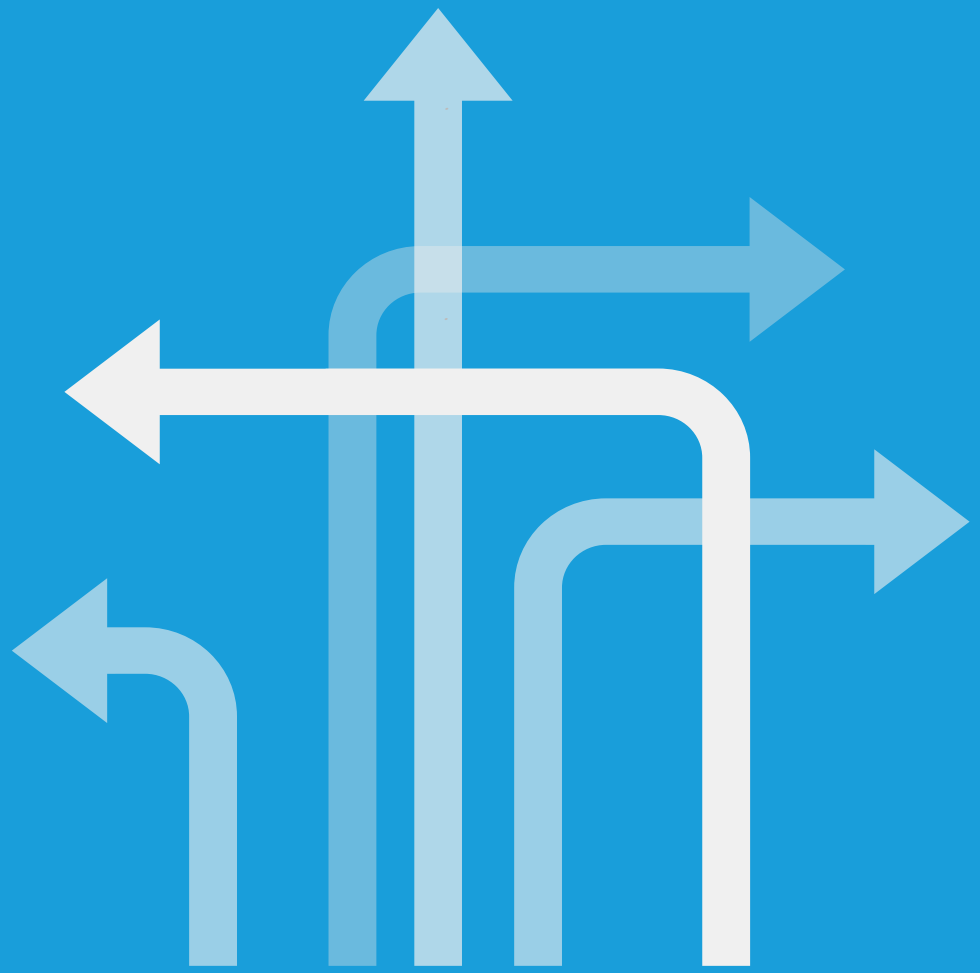
It takes a village to have an application network

It's key to have the ability to adapt towards the new operating model, one where DevOps and lean practices are adopted, as well as the creation of new roles and responsibilities to support this new operating model.



Where people play in the lifecycle

Depending on the maturity of your organization, there may be one person handling all these responsibilities or there may be multiple people doing so. The important thing to note is that each stage in the lifecycle provides specific value, which ensures application building blocks provide desired business outcomes.



So, now what?

For more background on full lifecycle API management, take a look at the Gartner Magic Quadrant, in which MuleSoft is a Leader. You can also take a look at further resources on API strategy and API development.

Contact us to find out more information about how full lifecycle API management can make a difference in your organization. **MuleSoft.com**

MuleSoft

MuleSoft's mission is to connect the world's applications, data and devices. MuleSoft makes connecting anything easy with Anypoint Platform™, the only complete integration platform for SaaS, SOA and APIs. Thousands of organizations in 60 countries, from emerging brands to Global 500 enterprises, use MuleSoft to innovate faster and gain competitive advantage.