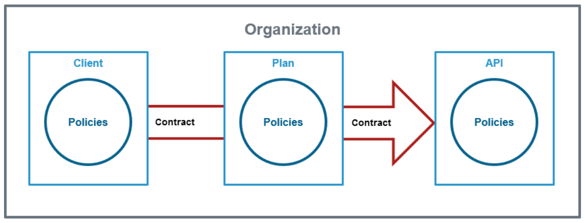
The Concepts of API Management

API Management uses a hierarchical data model that consists of five primary elements:

Figure: Overview on the API Management Data Model



The main elements of API Management are **Clients**, **Plans** and **APIs**. All of them can contain **Policy** definitions. To be used, they need to be grouped by an **Organization** and related by a **Contract**.

Element	Description
Organization	Almost everything in the API Management data model exists in the context of an organization: • An organization is a logical unit within API Management. This can be a company, department, etc. • An organization is a container of other elements: plans, APIs, and clients are defined per organization. • Every user must be associated with at least one organization to be able to manage elements in the application. • API Management implements role-based access control for users. You can give organization members different roles to restrict the actions he is able to perform and the elements he can manage within the organization. • Membership for each organization can be easily managed in the **Organization** tab.
Client	The client is the consumer of the API: • The client consumes managed APIs offered through API Management. • Each client can consume multiple APIs within API Management. The relation between client and API is defined via a contract and a plan. • As with an API or a plan, you can also add policies to a client. When creating a contract, an API-Key to invoke the API will be assigned.
Plan	A plan is a set of policies that defines the level of service API Management provides for an API. • Plans enable users to define multiple different levels of service for their APIs. • Plans specify the contract between a client and an API. • It is common to define multiple plans with divergent configuration options for the same API. **Example:** An organization offers two plans for the same API: Plan A is more expensive than plan B, but it offers a higher level of API requests in a given (and configurable) period of time.

On this Page:

Related Pages:

- The API Management User Interface
- The Concepts of API Management

- Organizations
- Clients
- Plans
- APIs
- Policies
- Developer Portal

API	APIs in API Management represent real back-end APIs (Application Programming Interfaces). An API is also known as a **service**, meaning anything that can be invoked remotely by some sort of client. API Management provides a way to turn unmanaged (raw) back-end APIs into **managed** APIs by attaching policies to them. Every managed API can be published as **Public** API or **Private** API or both: • **Public** APIs are available to consumers without a key. Only policies defined on the API apply to public APIs. • **Private** APIs are only accessible for known consumers, called clients. Every client has an individual key to access the API. Policies defined on the client, the selected plan in the contract and the API apply. In API Management, users can create new APIs manually or easily import them from the API Catalog.
Policy	Policies are at the lowest level of the data model, but they are the most important concept: A policy is a rule or a set of rules API Management uses to manage access to your APIs. • Policies are applied to all API requests and represent a unit of work applied at runtime to the request by API Management. • You can define a policy chain, a defined order in which the policies will be applied to API requests.
Contract	A contract relates a client to an API, using a plan.

Versioning

API Management supports versioning for APIs, plans and clients. All three elements share one behavior: They have to be determined to be available for use in the gateway.

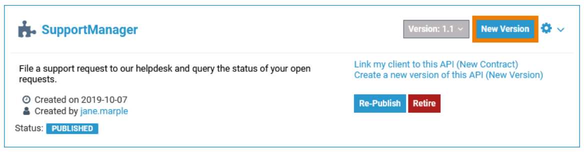
- APIs must be PUBLISHED
- Plans have to be LOCKED
- Clients have to be REGISTERED

While it is still possible to modify a published API and a registered client, a locked plan cannot be revised.

If modifications to the API Management configuration are necessary, you can do this by:

- Creating a new element.
- Modifying an existing element (only APIs and clients).
- Create a new version of an existing element.

Versioning allows you to modify the configuration of an existing element, but retain the previous version. To create a new version of an element, open its details page and click **New Version** in the basic settings:



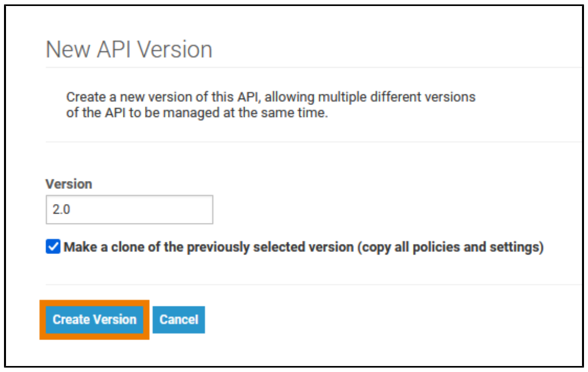
File a support request to our helpdesk and query the status of your open requests.

Created on 2019-10-07
Created by jane.marple

Status: PUBLISHED

Link my client to this API (New Contract)
Create a new version of this API (New Version)

Then, configure your new version:


New API Version

Create a new version of this API, allowing multiple different versions of the API to be managed at the same time.

Version: 2.0

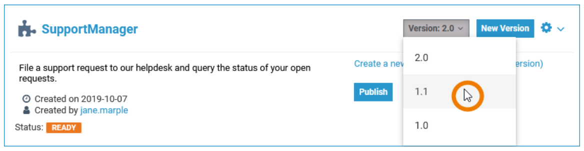
[✓] Make a clone of the previously selected version (copy all policies and settings)

Field	Description
Version	Enter a version name or number.
Make a clone	Enable this checkbox if you want copy the policies and settings of the previous version to your new version. Typically cloning is more convenient when making incremental changes.

Click **Create Version** to add the new version. It will be saved to the element.

You can enter numbers and text in the **Version** field which allows the use of version numbers (e.g. 1.0, 2.1 ...) as well as version descriptions (e.g. Gold, Super etc.).

Use drop-down **Version** in the basic settings of the element to switch between the different versions:


SupportManager

File a support request to our helpdesk and query the status of your open requests.

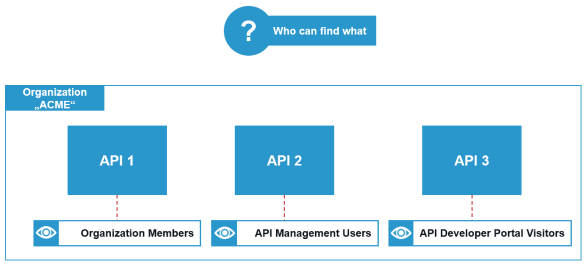
Created on 2019-10-07
Created by jane.marple
Status: READY

Visibility

The visibility concept of API Management enables a fine-grained access to APIs and defines which user groups can find them. The visibility resides on top of the permission system as another security layer. Visibilities are applicable to Public APIs and plans for Private APIs. The chosen visibility affects both UIs: The content of API Management as well as the content of the API Developer Portal from where API consumers can access the APIs. Relevant for the visibility is the identity management (IDM) group a user belongs to: A user who is part of the **API-Management-User** group can browse and find the same APIs in API Management and in the Developer Portal once logged in.

You can choose between three different visibilities:

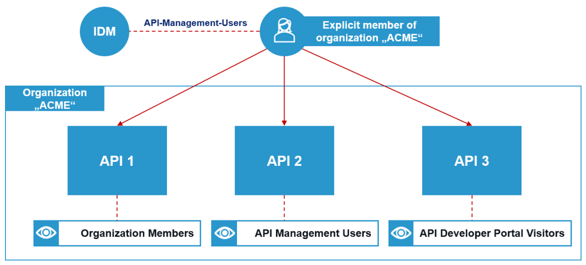
- **Organization Members (default)**
- **API Management Users**
- **API Developer Portal Visitors**

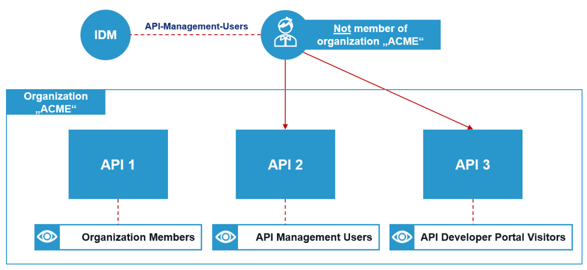
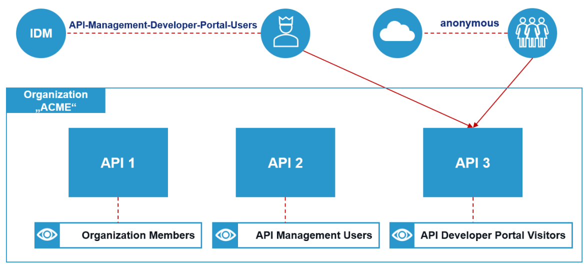


Example:

Three APIs have been created in organization ACME. Each API is assigned a different visibility:

- **API 1: Organization Members**
- **API 2: API Management Users**
- **API 3: API Developer Portal Visitors**

 IDM API-Management-Users Explicit member of organization „ACME“ Organization „ACME“ API 1 API 2 API 3 Organization Members API Management Users API Developer Portal Visitors	**Organization Members (default)** APIs are secured by default: If you create a new API, the default visibility setting is **Organization Members**. Only members of the organization the API has been created in are allowed to see it. **Example:** API Management users who are explicit member of organization ACME can see API 1, API 2 and API 3.

 IDM API-Management-Users Not member of organization „ACME“ Organization „ACME“ API 1 API 2 API 3 Organization Members API Management Users API Developer Portal Visitors	**API Management Users** Select the visibility **API Management Users** to allow all API Management users to see your API. **Example:** All API Management users or administrators can see API 2 and API 3, they do not need to be member of organization ACME.
 IDM API-Management-Developer-Portal-Users anonymous Organization „ACME“ API 1 API 2 API 3 Organization Members API Management Users API Developer Portal Visitors	**API Developer Portal Visitors** Choose the visibility **API Developer Portal Visitors** to allow API developer Portal users and visitors to see your API. **Example:** Developer Portal users and anonymous portal visitors without a PAS login can see API 3 only.