**Explication présentation mémoire**

**Introduction** :

Pour commencer, Lorsque nous parlons du monde du développement des logiciels, des sites web, des APIs, une spécification est un ensemble d’exigence qu’un système, logiciel ou API doit satisfaire.

Elle décrit les fonctionnalités et les attentes généralement exprimées par les utilisateurs.

Pour les APIs ou Application Programming Interface qui permettent à deux logiciels indépendants de communiquer l’un avec l’autre. Par exemple le propriétaire d’une base de données peut décider de partager les informations de base de données à travers une API. Une spécification est une interface permettant aux utilisateurs ou développeurs d’applications de comprendre les fonctionnalités de l’API. Ce qui leurs permet d’intégrer des services à distance offerts par l’API dans leurs applications ou sites web.

Cependant, il existe une spécification pour les APIs d’un modèle particulier qui peut être différent du modèle publication/abonnement.

**C’est quoi le modèle publication/abonnement** :

Le modèle publication/abonnement ou (publish/subscribe ou pub/sub en anglais) est un modèle qui permet à une en entité de publier ses informations et à une autre de s’aboner et communiquer ces informations publier.

Ce modèle est constitué d’un ou plusieurs producteurs, d’un intermédiaire appelé broker et d’un ou plusieurs consommateurs.

* Les producteurs : produisent les informations et les publient
* Le Broker : per
* Les consommateurs : s’abonnent aux informations et les consomment

**Le fonctionnement** :

font partie des composants des différents programmes. Utile pour permettre à un produit ou un service d’être en interaction avec d’autres éléments, l’API sert à simplifier le processus de développement d’une application.

Pour les utilisateurs, l’interface de programmation d’application rend les données ou les fonctionnalités d’une application disponible. D’une manière plus concrète, une Application Programming Interface joue le rôle d’intermédiaire entre deux systèmes informatiques indépendants pour pouvoir interagir de façon automatique, en minimisant ou sans demander l’intervention d’un utilisateur.

Désormais, la construction de logiciels ne nécessite plus une équipe d’ingénieurs ou de coûteux serveurs. Une clé API et sa documentation sont suffisantes pour intégrer aisément une fonctionnalité externe.

Pour avoir une documentation et une clé pour accéder aux services partagés par les API web, La spécification OpenAPI actuelle découle d’un projet antérieur, Swagger. L’entreprise de développement SmartBear a placé les spécifications Swagger existantes sous licence libre et confié leur maintenance et leur développement ultérieur à l’OpenAPI Initiative. Outre SmartBear, l’OpenAPI Initiative rassemble des géants du secteur comme Google, IBM et Microsoft. Le projet est également sous la gouvernance de la fondation Linux.

Grace a une documentation OpenAPI l’utilisateur peut ajouter des fonctionnalités sur son site ou dans son application pour les rendre de plus en plus dynamiques, sans que l’internaute ne connaisse les détails du programme. Il est important de souligner que le fournisseur qui décide de se servir d’une Application Programming Interface peut intégrer quelques modifications dans son programme sans avoir à déranger ses utilisateurs.

OpenAPI est limité à HTTP(S) en tant que protocole (d'application) et, par conséquent, ne peut pas décrire les API basées sur un autre protocole tel que WebSockets, MQTT ou CoAP.

La restriction aux API RESTful (avec les opérations GET, PUT, POST et DELETE) signifie qu'elle ne peut pas du tout décrire les services Web HTTP/SOAP (qui sont généralement décrits dans WSDL) et d'autres services et API non RESTful.

Le travail effectué pour la mise en place de cette nouvelle spécification comprend quatre (4) parties:

Plan du travail :

* La spécification open source existante pour les APIs qui est ***OpenAPI***
* Le modèle de communication publication/abonnement sur lequel notre nouvelle spécification est basé
* La nouvelle spécification pour les APIs du modèle publication/abonnement appelé ***OpenPS***
* Le générateur de code ***OPENAPI-PS*** et un cas d’utilisation pour un producteur (qui publie) et un consommateur (qui s’abonne).

**OpenAPI**:

OpenAPI définie une interface standard, indépendante du langage de programmation pour les API REST. Cette interface permet aux consommateurs et aux machines de découvrir et comprendre les capacités des services sans avoir accès au code.

OpenAPI est constituée de huit (8) objets qui sont : openapi, info, serveur, paths, components, security, tags et externalsDocs.

Openapi : version de la spécification OpenAPI utilisé

Into : décrit l’entête de l’api

Serveur : description et url du serveur qui contient les services auxquels l’API est supposée partager

Paths : les différentes méthodes ou opérations d’accès aux services par le protocole http. Il décrit les manières d’accès aux services autorisées par l’API

C’est la sécurité de l’API décrit par l’objet security. Ces manières d’accès sont :

Query : pour la sécurité des requêtes ou sécurité API\_KEY

Header : dans l’entête des paquets

Components : il décrit les parties réutilisables de l’API pour éviter la duplication. Une fois ces parties définit dans component on la fait référence avec l’objet $ref fourni par OpenAPI.

Security : il décrit la sécurité utilisée par l’API pour avoir accès aux donnés. OpenAPI utilise quatre types de sécurité qui sont : API\_key, Aouth2 :

Tags : c’est array tableau de description d’un ensemble d’opération

ExternalsDocs : pour faire référence a un document externe qui peut guider l’utilisateurs de l’API.

Un document de cette spécification est utilisé par l’outil Swagger qui a l’origine était cette spécification Mais qui est renommé à partir de la version 3.0.0 en OpenAPI. Maintenant OpenAPI est la spécification et Swagger l’outil utilisant cette spécification. C’est avec Swagger qu’on décrit les documents APIs et qu’on génère les codes d’implémentations en différents langages. Il permet aussi de visualiser et de tester les APIs comme le fait Postman.

Sur le plan technique, YAML et JSON présentent très peu de différences, ce qui permet d’automatiser la conversion d’une définition API existante d’un langage à l’autre

Avantage :

Définir des API HTTP sans dépendre d’un langage de programmation particulier ;

Générer du code serveur pour une API définie en OpenAPI ;

Générer des bibliothèques client pour une API conforme à OpenAPI en plus de 40 langages de programmation ;

Manipuler une définition OpenAPI avec des outils appropriés ;

Créer une documentation API interactive ;

Permettre aux hommes et aux machines de découvrir et comprendre les capacités d’un service sans consulter le code source ni avoir recours à une documentation supplémentaire ;

Accéder aux services API avec une charge de travail minime à l’implémentation.

Limite de OpenAPI :

Modèle publication/abonnement :