La spécification OpenPS

1. **Introduction** :

De nombreuses organisations souhaitent passer à des architectures informatiques centrées sur les API et basées sur des micro services pour améliorer la vitesse, l'agilité et l'efficacité. Une gestion cohérente et efficace de ces interfaces et services est essentielle pour un déploiement et une utilisation réussie. Cependant, la plupart des organisations recherchent une solution de gestion des APIs ou une solution de conception des APIs REST avec une spécification offrant des services asynchrones. La plupart des services peuvent être l’interaction différent d’autres services fournis par des différents APIs. Sachant que la communication entre plusieurs applications peut être complexe, demande beaucoup de maintenance et augmente le temps de réponse total. L’uniformisations des normes de conception de ces derniers avec une telle spécification permet résoudre certaines contraintes la centralisation d’un serveur, la conversation des adresse IP en nom de domaine et le maintient d’une seule communication à la fois.

1. **Définition** :

La spécification OpenPS définie une interface standard pour les APIs web en se basant sur le modèle Publish/Subscribe. Elle assure l’interopérabilité et la lisibilité des code par les producteurs, consommateurs ainsi que pour les machines. Elle se réfère sur la spécification OpenAPI pour la description et la mise en place des APIs REST.

Avec les outils de documentation de génération de code une spécification OpenPS permet de générer le code du producteur et du consommateur dans divers langages de programmation. Le document OpenPS peut être un seul ou divisé en plusieurs partie connectées en référençant ces parties. Dans ce cas il est recommandé de nommer le document racine openps.json ou openps.yaml.

1. **Constitution de la spécification OpenPS**:

Tout comme la spécification OpenAPI, la spécification OpenPS est constituée d’un ensemble d’objet suivant une structure hiérarchique. Une spécification OpenPS est composé de sept (7) objet racine. Parmi ces objets la présence de quatre (4) sont exigé pour un minimum description d’un document OpenPS.

1. **openps** :

C’est une chaîne de caractère décrivant la version utilisée par les documents OpenPS permettant aussi leurs interprétations par les outils de cette spécification et fait parti des objets exigé par la spécification. Cette spécification est à la version « 1.0.0 »

1. **info** :

C’est un objet dont sa présence dans la spécification est obligatoire. Il fournit les informations sur l’entête de L’API, il est constitué d’autre objet dont les plus important sont :

* **title** : le nom de l’API.
* **version** : version du document OpenPS.

1. **broker** :

C’est un objet exigé par la spécification appelé aussi courtier qui offre les services Publish/Subscribe. Sa valeur fournie des informations des entités cibles. Il exige la présence de certains objets conteneurs tels que :

* **name** : est une unique chaîne de caractère qui représente le broker.
* **hostname** : représente le nom d’hôte du broker cible (ici localhost).
* **port** : un nombre entier qui désigne le port du broker cible.

1. **protocol** :

Il précise le protocole à implémenter pour la communication Publish/Subscribe par exemple le protocole AMQP.

1. **topics** :

Cet objet décrit le nom des topics disponible ainsi que les opérations de publication/abonnement pour l’API. Il représente l’objet Path de la spécification OpenAPI. Il exige que le nom des topics commence par un slash (/) sur lequel il n’y a que deux opérations disponibles et un objet message exprimant le format de l’information échangée sur ce topic. Il est le dernier objet exigé par cette spécification.

1. **publish** :

Il définit l’opération publish (de publication) d’un message sur topic. C’est un objet qui décrit les résumés sur un topic à publier, l’entités chargé de publier ce topic ainsi que les qualités de service pour la distribution de ce topic. Il définit ces informations à travers d’autres objets qu’il supporte. Ces objets sont présentés ci-dessous :

* **tags** : c’est un tableau de chaine de caractère dont l’item i correspond à la description de l’opération de catégorie i.
* **summary** : c’est une chaine de caractère désignant un petit résumé de ce que fait l’opération publish.
* **description** : c’est la description de l’opération publish en chaine de caractère.
* **entities** :
* **QoS** : c’est une chaine indiquant comment le message est distribué et les valeurs autorisées sont : *none, at-most-once, at-least-once, exactly-once*.

1. **subscribe** :

Il définit l’opération subscribe (abonnement ou récupération d’un message) sur un topic. C’est un objet qui décrit les résumés de l’abonnement sur un topic et l’entités chargé de récupérer les messages sur ce topic. Il fournit ces informations à travers les mêmes objets que publish mais ne prend pas en charge la qualité de services QoS.

1. **message** :

Il contient trois (3) composants permettant de définir le format et le contenu d’un message.

* ***description*** : un petit résumé en chaîne de caractère décrivant le message.
* ***content*** : c’est un tableau *map*(*string*, *object*) permettant de définir le contenu d’un message
* ***Required*** : un booléen qui indique si le corps du message est exigé et à la valeur false par défaut.

1. **components** : c’est un objet conteneur qui permet la description pour éviter la duplication du code des opérations qui ont des paramètre en commun et tout en le référençant dans l’opération en question avec la propriété $ref qui a un chemin comme valeur depuis l’objet racine. Il objet est le même comme dans la spécification OpenAPI.
2. **security** :

Il respectera toutes les règles de celle OpenAPI mais pas encore défini.

1. **tags** : le même que la spécification OpenAPI.
2. **externalsDocs** : le même que la spécification OpenAPI.

https://mina7.net/pin/