Swagger 2 est mort, vive OpenAPI 3

Speakers : Sébastien Lecacheur, Grégory Bloquel
Format : Conférence
Date : 20 avril 2018

La question de la documentation d'une API est importante.
Lors de sa mise en place, les speakers avaient pour exigences :

- Documentation pour les utilisateurs internes ou externes.
- Approche contract-first : rédiger la doc avant d'écrire le code
- Documentation gérée comme le code
- Documentation lisible par les humains

Word ne convenait pas. Ils ont opté pour Swagger.
Concurrents de Swagger : RAML et Blueprint

Swagger

Historique de Swagger :

- Swagger 1 est sorti en 2010.
- En 2014 est sorti la V2.
- En 2015 : démarrage de l'OpenAPI Initative au sein de la fondation Linux à partir de Swagger 2.
- En 2017, Swagger 3 est sorti et utilise l'OAI. Swagger 3 est une implémentation de l'API.

OpenAPI Initiative réunit de nombreux acteurs : WSO2, Mulesoft, Microsoft, ….
Tous les acteurs industriels majeurs se sont mis d'accord.

Swagger a été popularisé en 2014 par son approche code first.
Documentation générée à partir des annotations et donc du code.

Présentation de l'approche Contract First

On commence par des URI et des exemples. Cycle itératif avec le métier et les utilisateurs pour arriver à un contrat d'interface stable.
Outre la génération de documentation, on peut ajouter de la génération de tests, de code, de mocks.

Rappel sur les APIs REST :

- URI comme identifiant de ressource
- Verbe comme identifiant d'opération
- Réponse HTTP comme représentation de la ressource (en JSON par exemple)
- Liens comme relation entre les ressources (HATEOS)

Nouveautés de OAS 3.0.1

- De loin, pas de gros changements
- Outil OpenAPI Map pour naviguer dans la documentation
- Editeur en ligne editor.swagger.io ou outil Swagger editor

Nouveautés de Swagger 3

- Description migrée de GFM vers Common Mark
- Gestion des versions en Semantic Versionning : version sur 3 digits x.y.e
- Montée de version à JSON Schema Wright Draft 00
    - Nouveaux types supprotés : oneOf …

La 1ière ligne d'un fichier YAML de Swagger précise la version
Swagger : "2.0"
Openapi : "3.0.1"

Nouveautés serveur :

- URL permettant de préciser différents serveurs (production, staging)
- Varialibilsation de l'URL, par exemple avec le nom de l'environnement
- Possibilité de surcharger le server dans chaque endpoint

Nouveautés sécurité :

- Basic devient http
- Support de OpenID Connect
- OAuth 2 updated
- Ajout de « Scheme » et « bearer format »

Nouveautés composants :

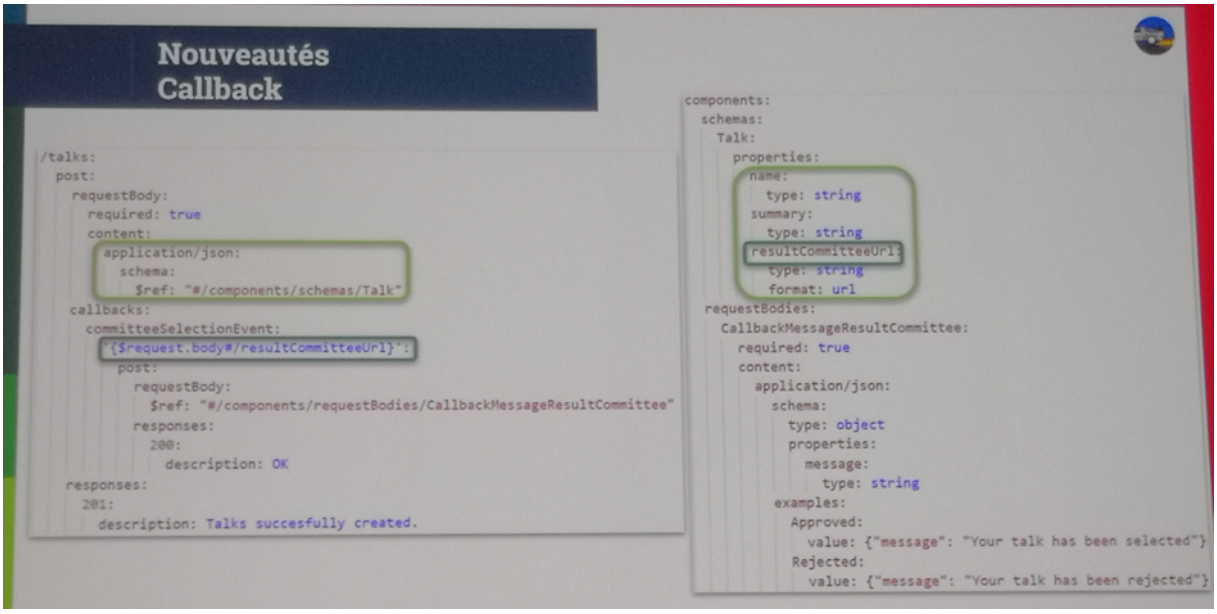
- Un composant peut être réutilisé dans la définition des API
- Beaucoup de déplacement / renommage

Nouveautés sur le format des requêtes :

- Wildcards pour les codes de retour (ex : 5XX) : on n'expose pas le code des erreurs serveurs internes au consommateur
- oneOf/anyOf pour du Polymorphsime/Composition

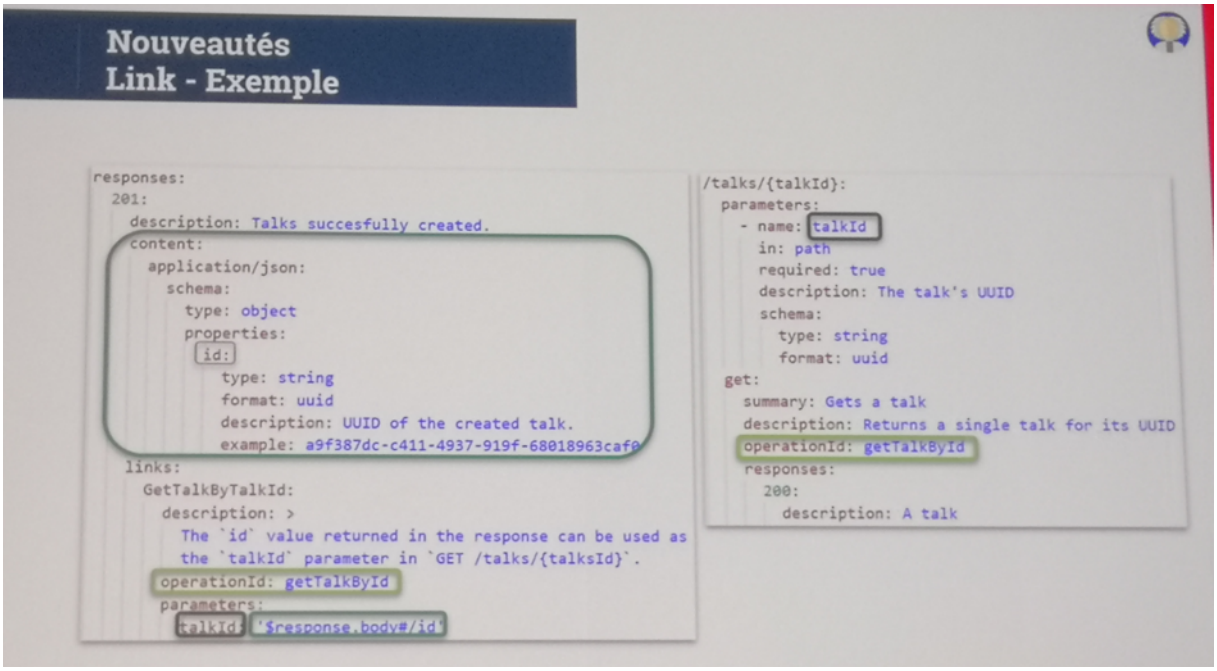
Nouveautés sur les callbacks :

- Support des webhook
    - Permet de notifier un utilisateur de l'API en temps réel d'un événement
- On peut ajouter des exemples



Nouveauté Link :

- Définir une relation entre une réponse et d'autres opérations
    - Opérations possibles
    - Pagination (cursor) : identifiant de curseur réutilisable pour appel suivant



Outils :

- Editeur swagger-editor
- Génération de code : swagger-codegen
    - Générateur de référence. Réécrit en Swagger 3.
    - Swagger-code-generators
    - Client/server Java, JAXS, Kotlin, HTML
    - Changement du moteur de templating : mustache => handlebar
    - Pull Request pour PHP et Scala

- Les autres générateurs dits compatibles OpenAPI 3 fonctionnent moins bien. Bien veiller à les qualifier.

Futur d'OpenAPI :

- Issues 1466
- Limitations
    - Pas de traits : décrire un comportement (filtrable, cachable …)
    - Pas de resources types : déclaration de type de ressources afin de mutualiser du comportement entre ressources (avec surcharge possible)
    - Pas de templating des composants : sur la réponse d'un POST, on ne peut pas mettre 'mon talk a été généré' mais 'ma ressource a été générée' => on doit rester générique