COMPÉTENCE B2.1 – SLAM
SÉQUENCE 7 – CRÉER ET EXPLOITER UN SERVICE WEB

SYNTHÈSE

Contenu

1. Les composants logiciels

1A. Qu'est-ce qu'un composant logiciel ?

Un composant logiciel est un élément logiciel indépendant, qui peut être exploité par une application cliente.

EXEMPLES DE COMPOSANTS

- Plug-in : composant intégrable, qui n'a pas de vie propre.
- Widget : composant graphique
- Pilote : composant permettant de gérer un périphérique
- Web Service : application exploitable par une autre application
- ...

1B. Étapes de mise en place d'un composant (dans le cas de composants complexes)

Analyse du besoin : analyse de l'existant, du besoin et des solutions techniques possibles (choix du composant le plus adapté au besoin).

Impact de l'intégration du composant : financier, technique et organisationnel.

Intégration : dans l'application cliente en vue de son utilisation.

Construction d'un composant : il est parfois nécessaire de construire un composant qui réponde exactement aux attentes. Il faut alors le coder, le mettre à disposition (déploiement), le documenter et éventuellement l'adapter.

2. Les web services (protocole SOAP)

2A. Qu'est-ce qu'un web service SOAP ?

Un web service SOAP (Simple Object Access Protocole) est un composant logiciel, de type applicatif, accessible via une interface. Il peut être écrit dans n'importe quel langage.
Pour être exploité, il doit mettre à disposition un fichier au format WSDL (Web Service Description Language) qui contient toutes les informations pour accéder à ses méthodes.

2B. Les standards utilisés

XML (Extensible Markup Language) : langage de balisage, proche du HTML au niveau structure, avec la possibilité de créer ses propres balises. Il permet de mémoriser des données au format texte.

SOAP (Simple Object Access Protocole) : protocole d'échange d'informations via le web, au format XML.

WSDL (Web Service Description Language) : format du fichier d'interface du web service, permettant de décrire comment accéder au web service (syntaxe utilisant le XML).

2C. Exploitation d'un web service SOAP

L'application cliente peut être écrite dans n'importe quel langage.

Pour exploiter un web service SOAP, il faut :

- connaitre l'adresse de son fichier WSDL ;
- y accéder dans l'application (comme on accède à une BDD) ;
- utiliser les méthodes proposées.

La syntaxe pour accéder et exploiter un web service SOAP dépend du langage client utilisé.

La plupart des IDE offrent des facilités pour intégrer et exploiter un web service SOAP.

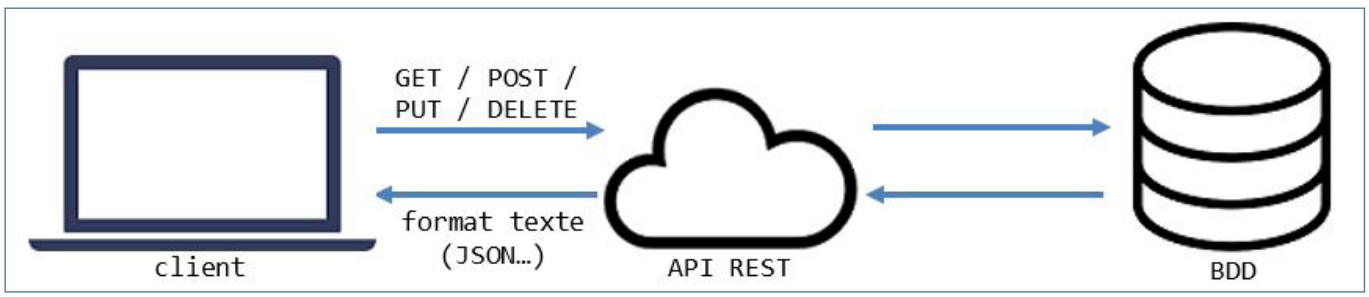
3. L'API REST

3A. Qu'est-ce qu'une API REST ?

Explication des termes :

- **API** : (Application Programming Interface) application exploitable par une autre application (cliente) et qui rend disponible des données et fonctionnalités ;
- **REST** : (REpresentational State Transfert) style d'architecture proposant un ensemble de principes et de bonnes pratiques pour coder une API.

Une API REST est un service web respectant juste un standard et un ensemble de bonnes pratiques.
Elle offre une solution plus légère et rapide que le web service SOAP, entre autres en évitant l'interface WSDL.



L'API REST reçoit les requêtes HTTP du client dans différents modes, exploite une BDD (ou autre) pour répondre à la requête, renvoie au client une réponse dans un format léger (texte), classiquement le format JSON.

3B. Les règles de l'API REST

Pour qu'une API soit considérée comme RESTFull, elle doit respecter les 6 règles suivantes (si elle en respecte certaines, elle est juste REST).

- Séparation du client et du serveur.
- Sans état (stateless) : pas de conservation de session.
- Uniformité de l'interface (actions clairement définies).
- Mise en cache possible des réponses.
- Architecture en couches.
- (optionnel) Possibilité de retourner du code exécutable (javascript...).

3C. Les bonnes pratiques de l'API REST

REST propose aussi un ensemble de bonnes pratiques, dont :

- utiliser les verbes HTTP (GET pour récupérer, POST pour insérer, PUT pour modifier, DELETE pour supprimer) ;
- utiliser l'URL rewriting pour que l'URL représente la requête ;
- retourner au format JSON 3 informations : "code" (avec codes officiels 200, 404...), "message" (correspondant au code), «result» (résultat de la demande).

3D. Création d'une API REST

Il suffit de créer une application qui accepte les requêtes HTTP et retourne un résultat dans un format léger (JSON fortement conseillé). L'application peut être écrite dans n'importe quel langage.

Il faut ensuite rendre accessible sa documentation qui présente le but de l'API, les chemins possibles (endpoints) avec les paramètres attendus et ce qui est obtenu.

3E. Exploitation d'une API REST

N'importe quelle application qui peut envoyer des requêtes HTTP, peut invoquer une API REST. Il suffit de construire l'url, de l'envoyer et de récupérer la réponse. Celle-ci étant dans un format précis (normalement JSON), il faut ensuite la décoder pour l'exploiter.

CNED, BP 60200, 86980 Futuroscope Chasseneuil Cedex, France

87D22FSWB1321