Système distribué

Définition

Un système distribué est un ensemble d'entités communicantes qui apparaissent à l'utilisateur comme étant un système cohérent

Caractéristiques

- Hétérogénéité
- Imperfection
- Redondance
- Goulot d'étranglement

Problématique

Interopérabilité : Il faut prévoir (n × (n −1) / 2 connexions pour n systèmes)

Solution : bus magique

(EAI : Entreprise Application Integrator) qui est une solution magique pour réduire le nombre de connexions et d'interfaces dans un système, il nécessite une seule connexion et une seule interface pour chaque système

Problématique

- Il est très facile de connecter les systèmes mais personne ne peut comprendre leurs dépendances au niveau du bus
- Si ce même système tombe en panne, le bus magique ne peut pas résoudre le problème

ESB

Entreprise Service Bus : son but est avant tout de permettre la communication des applications qui n'ont pas été conçues pour fonctionner ensemble,

Considéré comme une nouvelle génération de (EAI)

La différence majeure avec l'EAI réside dans le fait que l'ESB propose une intégration complètement distribuée.

SOA (Service Oriented Architecture)

Définition

SOA est un paradigme, un style, un concept, une approche et un système de valeurs qui conduit à certaines décisions concrètes lors de la conception d'une architecture logicielle

Concepts techniques de SOA

- **Prestation de Services** : Un service est une fonctionnalité métier autonome
- **Interopérabilité** : avec des systèmes hétérogènes, l'objectif principal est de pouvoir mettre en place ces systèmes facilement : haute interopérabilité
- **Couplage faible** : C'est le concept de réduction de dépendances du système

L'approche SOA se compose de 3 éléments :

- **Services** : représente des fonctionnalités métier autonomes, peuvent être implémentées par n'importe quelle technologie et n'importe quelle plateforme
- **Infrastructure spécifique** : permettant de combiner ces services de manière simple et flexible
- **Politiques et processus** : traitent l'hétérogénéité des systèmes distribués et le fait d'avoir des propriétaires différentes

SOA : Terminologie

Provider : fournisseur, le système qui implémente le service afin qu'un autre système puisse l'utiliser

Consumer : consommateur ou demandeur de service : le système qui appelle un service

Les Services

Définition

Un service est une fonctionnalité métier autonome qui accepte une ou plusieurs demandes et renvoie une ou plusieurs réponses via une interface standard bien définie.

Un service ne doit pas dépendre de l'état d'autres fonctionnalités

Interface

- Tout ce qui peut être utilisé comme interface et qui représente une fonctionnalité autonome peut être un service
- Une interface doit être bien définie sans ambiguïté

Contrat

Au départ, un service est décrit avec une interface, ensuite quand un consommateur spécifique souhaite utiliser le service avec des exigences spécifiques, on doit alors définir un contrat basé sur ces exigences.

Interface Besoins fonctionnels

Contrat Besoins non fonctionnels

Caractéristiques

- **Autonome** : Un service doit être indépendant
- **Coarse-grained** : est une abstraction qui cache la mise en œuvre au consommateur
- **Visible** : Pour appeler un service, on doit pouvoir vérifier s'il existe
- **Stateless** (Sans état) : L'état des objets ne doit pas être gardée dans le service pour une meilleur performance
- **Idempotent** : elle produit le même résultat lorsqu'elle est appelée à plusieurs reprises
- **Réutilisable** : Chaque fonctionnalité ne doit être implémentée qu'un seule fois
- **Composable** : Les services peut utiliser/appeler d'autres services
- **Interopérable** : un service peut être appelé par n'importe quel autre système

Classification de services

Service de base

- Chaque service fournit une fonctionnalité métier de base, et qu'il n'est pas logique de la diviser
- Stateful
- Un service de base doit avoir les propriétés ACID (Atomique, Cohérent, Isolé, Durable)

Deux types de services de base existent :

- **Service de données** (Lit ou écrit des données depuis ou vers un système backend) *Exemple* : Créer un nouveau client, Changer l'adresse d'un client
- **Service logique** Représente des règles métier fondamentales - *Exemple* : Indiquer si une année est bissextile ou pas

Service compose

Dans SOA, la composition de nouveaux services à partir de services existants est appelée orchestration

- Service composé pour plusieurs backends
- Service composé pour un seul backend

Service de Process

Représentent des workflows ou des process métier à long terme

Contrairement aux autres types, celui-ci a généralement un état qui reste stable sur plusieurs appels

Exemple : service de panier

Etat de Service

Stateless Un service est sans état lorsque toutes les données de l'instance du service, qui fait l'appel, sont toutes supprimées après l'appel

Stateful C'est un service qui maintient l'état sur plusieurs appels de service

Pour garder l'état dans le service, il faut s'assurer que les appels de service sont acheminés vers la même instance : il faut donc établir des sessions

L'état peut être gardé dans :

- Un service
- Le backend
- Le consommateur (sous forme de cookies)

Les Services Web

Les services Web sont considérés comme la manière dont SOA doit être réalisée dans la pratique

Normes fondamentales

Il existe 5 normes fondamentales de services Web :

Standards généraux

XML : utilisé comme format général pour décrire les modèles, les formats et les types de données

HTTP : protocole de bas niveau utilisé par Internet

Standards spécifiques aux services Web

WSDL : utilisé pour définir des interfaces de services

UDDI : norme pour l'enregistrement et la recherche des Services Web

SOAP : protocole d'échange des messages entre le consommateur et le service Web

WSDL (Web Services Description Language)

Est un langage d'interface permettant de définir l'interface d'un service web offert par un fournisseur.

Il permet d'identifier le service et l'ensemble d'opérations qui le constitue

Un fichier WSDL est un document XML qui permet de décrire :

- Ce que fait le service : la liste des opérations, leur sens, leur signature
- Où le trouver : le nom du service, et le point d'accès ("Endpoint")
- Comment l'invoquer : les règles d'encodage, le protocole de transport utilisé

Partie abstraite

Les types de données : Cette partie est optionnelle, elle permet de définir les structures de données spécifiques à l'application

Les messages : Un message décrit l'ensemble de données transmises au cours d'une opération

Les Types de Ports (portType) : Un type de port est composé de l'ensemble des opérations abstraites applicables au service(in-out/in-only/out-only)

Les Liaisons (Binding) : Une liaison décrit la façon dont un ensemble d'opérations abstraites (appelées types de port) est lié à un port selon un protocole réel (HTTP, SNMP). Cette liaison peut être une liaison de style RPC ou document.

Les Services : Un service est décrit comme un ensemble de points finaux (Endpoint) du réseau, appelés ports. Un port spécifie une adresse qui est associée à une liaison,

UDDI

Constitue une norme pour les annuaires de services web : les registres UDDI.

Un annuaire contient des informations techniques et administratives sur les entreprises et les services qu'elles publient.

Il permet alors de :

- Publier des informations sur une entreprise et ses services
- Chercher les informations sur une entreprise et ses services

Chaque registre UDDI stocke trois sortes de données dans trois pages différentes :

Les pages blanches : contiennent la liste des entreprises ainsi que des informations associées à ces dernières

Les pages jaunes : contiennent les services Web, au format WSDL, la description des services déployés par l'entreprise.

Les pages vertes : fournissent des informations techniques précises sur les services

SOAP (Simple Object Access Protocol)

SOAP est un protocole de communication basé sur XML pour permettre aux applications d'échanger des informations via HTTP, SMTP ou FTP

- Son objectif est de définir la structure générale des messages échangés entre les composants Web Services
- SOAP utilise XML (il est indépendant de tout langage de programmation)
- SOAP normalise la gestion des erreurs

Structure d'un message SOAP

Message SOAP =En-tête http+ enveloppe

Une **enveloppe** contenante :

Header : c'est un bloc optionnel, contenant des informations d'en-têtes sur le message

Body : c'est le bloc obligatoire, contenant le corps du message

Fault : définie dans le bloc Body. Il sert à reporter des erreurs Sa présence n'est pas obligatoire

REST

REST est un modèle architectural pour la création des services Web, adapté alors pour les systèmes distribués

Avec REST, il vous suffit de remplacer un appel SOAP par l'URL

Les services RESTful sont :

- Faciles à développer et à déployer
- Ils sont légers
- Leur hébergement et leur maintenance sont économiques

Éléments architecturaux

- **Client/serveur** : les applications REST ont un serveur qui gère les données et l'état des applications.
  - Le serveur communique avec un client qui gère les interactions des utilisateurs
- **Stateless :** HTTP contient toutes les informations nécessaires pour l'exécuter, ce qui signifie que ni le client ni le serveur n'ont besoin de se souvenir d'un état antérieur
  - Les clients gèrent l'état de leur application
- **Cacheable :** Les serveurs, les infrastructures et les clients peuvent les mettre en cache lorsque cela est possible pour améliorer les performances
- **URI (Uniform Ressource Identifier)** : C'est le seul identifiant de chaque ressource dans ce système REST.
- **Interface uniforme** est traitée par la combinaison d'URI et de verbes http
- **Système en couches** : REST est un style en couche entre les composants, Chaque couche a une fonctionnalité dans le système REST
- **Utilisation de l'hypermédia** : dans le cas d'une API REST, le concept d'hypermédia explique la capacité d'une interface de développement à fournir au client et à l'utilisateur les liens adéquats pour exécuter des actions spécifiques sur les données

Ressource : la représentation des données primaires est appelée Ressource

Une ressource peut être : une collection, un singleton, ressource de sous-collection

Solution hybride : SOA/REST

L'approche d'un mutualisme REST/SOA consiste à ajouter un serveur Web frontal à une application SOA.

ESB

Problématique

- **Communication entre les services** : si un système d'information est composé de plusieurs applications et services, leurs interactions devient difficile
- **Intégration** : un changement dans une application d'un SI rend son intégration délicate
- **Couplage entre consommateur et fournisseur** :
    - Couplage technique : le consommateur doit connaître le protocole d'échange
    - Couplage fonctionnel : le consommateur doit connaître le format d'échange

Définition

ESB agit comme un middleware intelligent pour permettre la communication entre différents systèmes / applications.

Un ESB permet aux applications d'accéder aux services disponibles de façon standardisée.

Solutions d'intégrations

- **ETL (Extract Transform Load)**
- **EAI (Entreprise Application Integration)**
- **ESB (Entreprise Service Bus)**

Pourquoi un ESB ?

- Un standard d'interopérabilité : le rôle principal de l'ESB est d'assurer
- Solution qui découple le consommateur du fournisseur
- Solution de traçabilité des messages
- Solution pour sécuriser les services

Responsabilités d'un ESB

- **Routage** est la capacité à canaliser une demande vers un service particulier
- **Transformation** : Convertir le format de données d'une requête par rapport au format de données approprié au service
- **Échange de messages**
- **Gestion la sécurité** : Protéger les services de l'entreprise contre les accès non autorisés
- **Gestion de transactions**

Processus

1. Le consommateur ne connait que l'ESB il invoque le service exposé par l'ESB
2. L'ESB détermine le fournisseur du service à invoquer par un mécanisme de routage
3. L'ESB assure la transformation de format de données reçu par celui qui pris en charge par le fournisseur
4. L'ESB invoque le fournisseur

Produits d'ESB

Les produits d'intégration diffèrent selon les fonctionnalités qu'ils offrent

Integration Framework : Permet d'intégrer différentes applications, avec différents protocoles et technologies, de manière standardisée

Exemple : Apache Camel

Entreprise Service Bus : Un ESB offre un support d'outils solide pour le déploiement et l'administration en plus des fonctionnalités de base

Exemple : Oracle Service Bus, Mule ESB, Apache Synapse

Integration Suite : Une suite qui offre toutes les fonctionnalités d'une ESB tout en ajoutant d'autres fonctionnalités telles que : BPM (Business Process Management), BAM (Business Activity Monitoring), MDM (Master Data Management)

Exemple : Talend Suite

Critères de choix

- **Convivialité** : Quelle est la complexité de l'installation ? Combien d'outils sont nécessaires
- **Maintenabilité** : comment administre-t-on le produit ? Existe-t-il une interface graphique ?
- **Communauté** : Existe-t-il des forums publics ou des listes de diffusion actifs ?
- **Fonctionnalité** : toutes les fonctionnalités requises sont-elles proposées ?
- **Flexibilité** : peut-on personnaliser les fonctionnalités du produit en fonction des besoins ?
- **Extensibilité** : est-il possible d'étendre le produit ?
- **Coût**
- **Licence** : quel modèle de licence ou d'abonnement est utilisé ?