

# Cours SOA

**Hazem Feki**

Ingenieur développement logiciel | devops



+216 52 86 13 20



hazemfeki087@gmail.com

**01**

# **Introduction à SOA & Microservices**

# 1. Rappel des architectures logicielles

## 1.1 Architecture monolithique

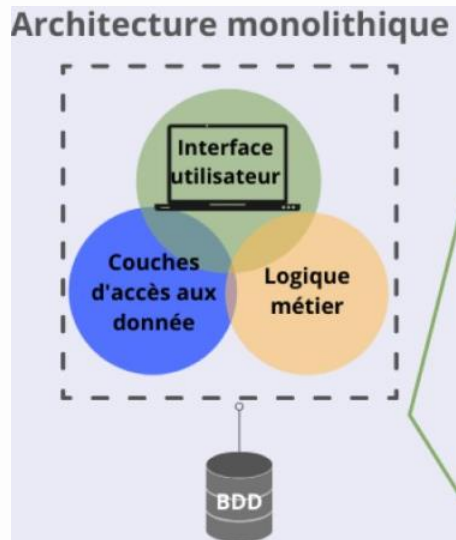
**Architecture** : un modèle de développement logiciel où tous les composants d'une application (interface utilisateur, logique métier, accès aux données) sont regroupés dans une seule et unique unité unifiée

### Avantages :

- Simple à concevoir au début.
- Facile à tester et à déployer (une seule application).
- Bonne performance interne (pas de communication réseau entre modules).

### Inconvénients :

- Difficile à maintenir quand le projet devient grand.
- Une petite modification peut casser tout le système.
- Difficile à faire évoluer : on ne peut pas mettre à jour un seul module.
- Le déploiement prend du temps (il faut tout redéployer).



## 1.2 Architecture SOA (Service Oriented Architecture)

**Architecture :** L'application est divisée en services indépendants (appelés *services métiers*), chacun offrant une fonction bien définie

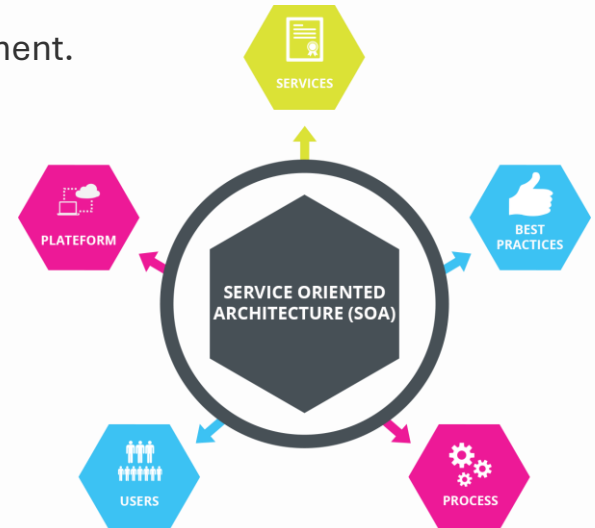
Ces services communiquent entre eux via un bus de service (ESB) ou des web services (SOAP / REST).

### Avantages :

- Meilleure modularité : chaque service peut être développé séparément.
- Réutilisation des services dans d'autres projets.
- Communication standardisée entre les services.

### Inconvénients :

- Besoin d'un bus de communication (ESB), souvent complexe à gérer.
- Performance plus faible (les services échangent par le réseau).
- Plus de complexité pour la sécurité, la gestion des erreurs, etc.



## 1.3 Architecture microservices

### Architecture :

C'est une évolution moderne du concept SOA.

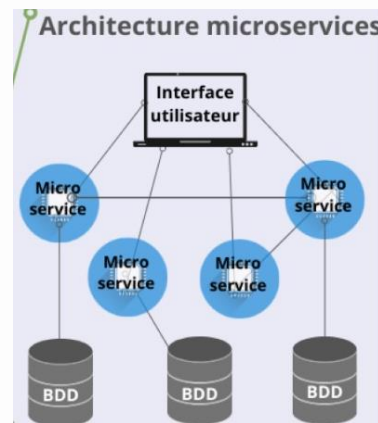
L'application est composée de petits services indépendants appelés *microservices*, chacun ayant sa propre base de données, son propre déploiement et souvent même sa propre technologie.

### Avantages

- Très flexible : chaque équipe gère son microservice.
- Mise à jour indépendante (on peut déployer un seul service sans toucher aux autres).
- Résilience : si un service tombe, les autres continuent de fonctionner.
- Scalabilité facile : on peut agrandir seulement le service le plus utilisé.

### Inconvénients

- Plus complexe à concevoir et à maintenir.
- Nécessite une infrastructure moderne (Docker, Kubernetes, CI/CD...).
- Plus de communications réseau → plus de surveillance nécessaire.

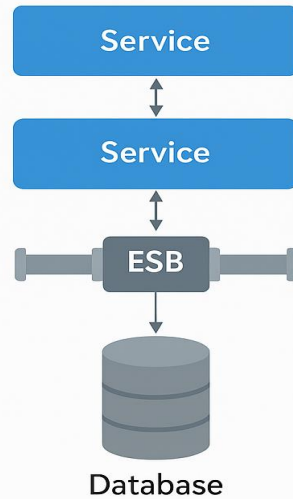


## Monolithic Architecture

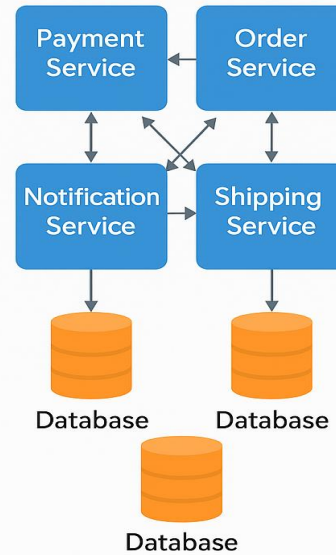


Database

## SOA



## Microservices



1S,

## 2. Comparatif global

Critère	Monolithique	SOA	Microservices
Structure	Un seul bloc	Services connectés via un bus	Services indépendants et légers
Communication	Interne	Via ESB (SOAP)	Via API REST
Déploiement	Unique	Centralisé	Indépendant pour chaque service
Base de données	Unique	Partagée	Séparée par service
Évolution	Difficile	Moyenne	Très flexible
Outils modernes	Non nécessaires	Partiellement	Oui (Docker, Kubernetes, CI/CD)

**02**

# **SOA, SOAP, WSDL**



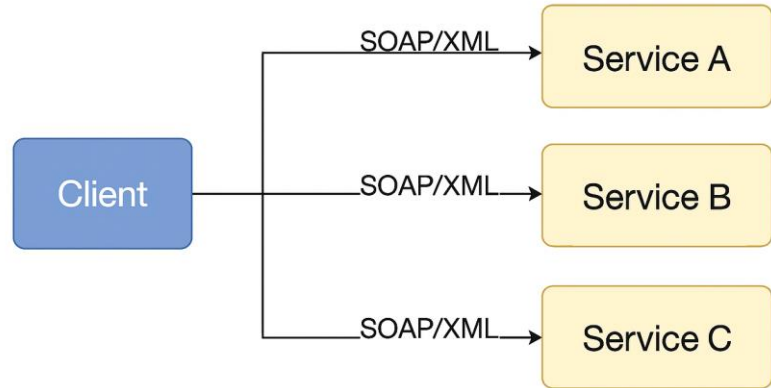
# Objectifs

- Expliquer ce qu'est une architecture SOA et pourquoi on utilise **SOAP** et **WSDL**.
- Lire et interpréter un **fichier WSDL**.
- Créer et tester une **requête SOAP** avec **SoapUI**.
- Et comprendre comment **les services communiquent via un contrat formel**.

# 1. Introduction à SOA

## Définition de SOA

- SOA (Architecture Orientée Services) est une approche de conception logicielle où les fonctionnalités sont décomposées en services indépendants et réutilisables.
- Ces services communiquent via des interfaces standardisées, souvent sur un réseau.





## **Avantages de SOA**

- + Réutilisabilité
- + Flexibilité et évolutivité
- + Interopérabilité entre différentes plateformes
- + Maintenance facilitée

## Composants principaux

- **Services** : unités fonctionnelles autonomes
- **Consommateurs** : applications qui utilisent ces services
- **Registre de services** : annuaire où les services sont décrits et découverts



## 2. Introduction à SOAP

### Qu'est-ce que SOAP ?

SOAP (Simple Object Access Protocol) est un protocole basé sur XML pour l'échange d'informations structurées dans un environnement décentralisé et distribué.

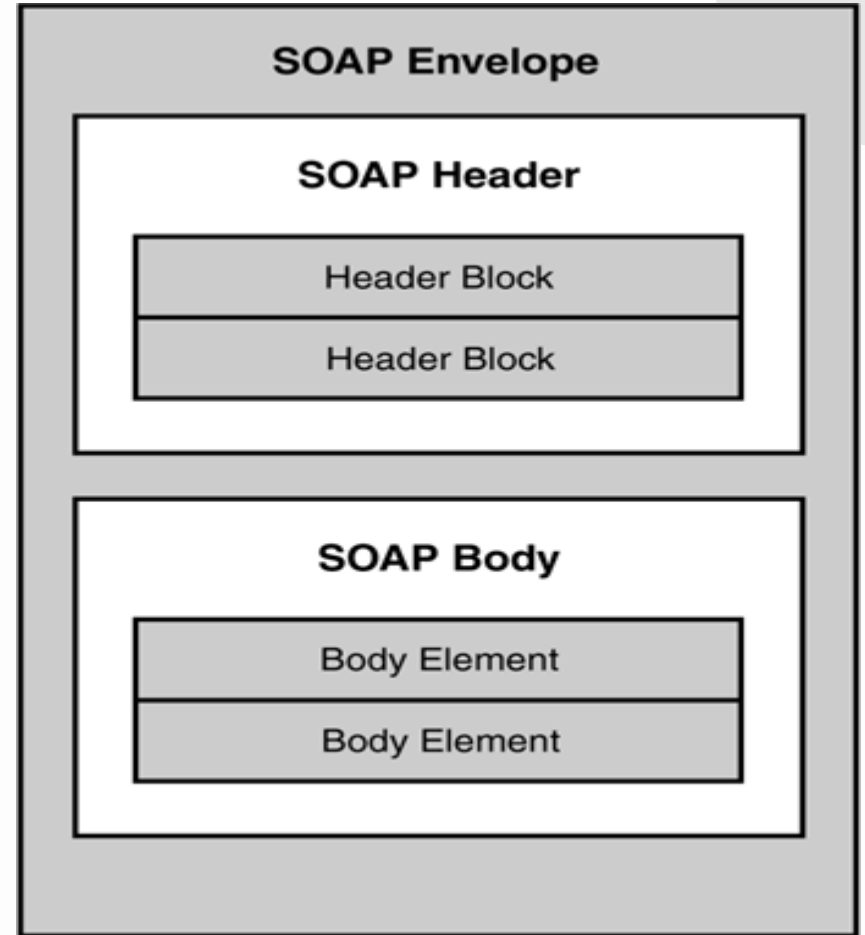
### Fonctionnement de SOAP

- Le client envoie une requête SOAP au service web
- Le service traite la requête et renvoie une réponse SOAP
- Les messages SOAP sont encapsulés dans des enveloppes XML



# Structure d'un message SOAP

- Enveloppe SOAP: `<soap:Envelope>`
- En-tête optionnel: `<soap:Header>`
- Corps: `<soap:Body>` qui contient l'opération et les paramètres



## Exemple de Requête Soap:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <tem:Add>
      <!--type: int-->
      <tem:intA>3</tem:intA>
      <!--type: int-->
      <tem:intB>3</tem:intB>
    </tem:Add>
  </soapenv:Body>
</soapenv:Envelope>
```

## Réponse attendue:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/ http://schemas.xmlsoap.org/soap/envelope.xsd">
  <soap:Body>
    <AddResponse xmlns="http://tempuri.org/">
      <AddResult>6</AddResult>
    </AddResponse>
  </soap:Body>
</soap:Envelope>
```



## 2. WSDL (Web Services Description Language)

### Qu'est-ce que WSDL ?

Le **WSDL** (Web Services Description Language) est au cœur de l'architecture SOA.

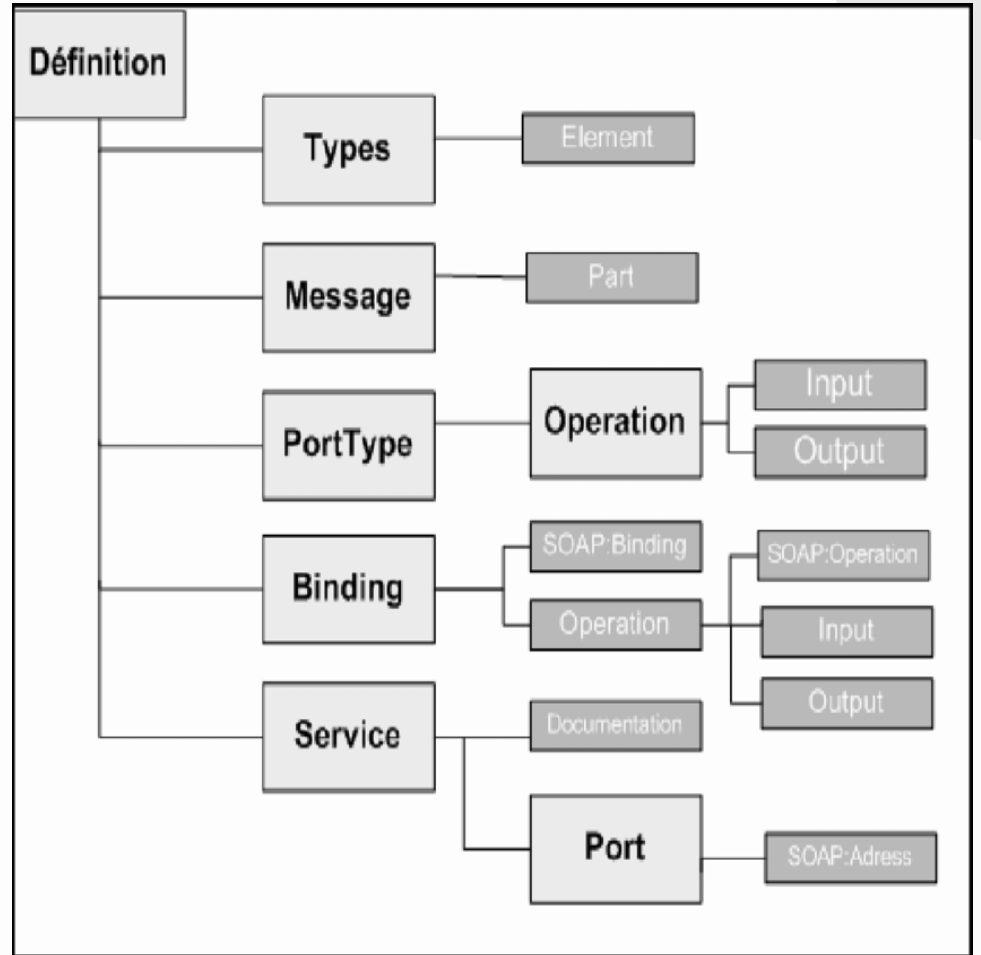
Il joue le rôle de **contrat** entre le fournisseur de service et le consommateur.

C'est un **document XML** qui décrit tout ce qu'il faut savoir pour consommer un service SOAP :

- les **opérations** disponibles (ex : Add, Subtract...),
- les **types de données** échangés,
- les **adresses réseau** (endpoint),
- et la **structure exacte** des messages d'entrée et de sortie.

## Structure générale d'un WSDL

- definitions (racine)
- types (définitions XSD)
- message (description des messages d'entrée et de sortie)
- portType (ensemble d'opérations)
- binding (protocole et style de message)
- service (points d'accès et ports)



# Résumé de la relation SOAP ↔ WSDL

Élément

Rôle

SOAP

Protocole qui transporte le message XML

WSDL

Contrat qui décrit comment parler avec le service SOAP


XML

Langage utilisé pour structurer les messages

SOAPUI

Outil pour tester les services SOAP à partir d'un WSDL





**LAB-SOA**