Rendu du TP : Conception d'un système d'évaluation de demande de prêt immobilier

Architecture orientée services (SOA) – SOAP / SPYNE

Nom : Joly **Prénom :** Théo

a. Élaboration d'une architecture décrivant le fonctionnement du système

L'objectif est de concevoir un **système distribué** qui évalue une demande de prêt immobilier à partir d'une **phrase en langage naturel**. Le système repose sur une **architecture orientée services (SOA)** composée de services indépendants communiquant via **SOAP**.

Schéma conceptuel de l'architecture

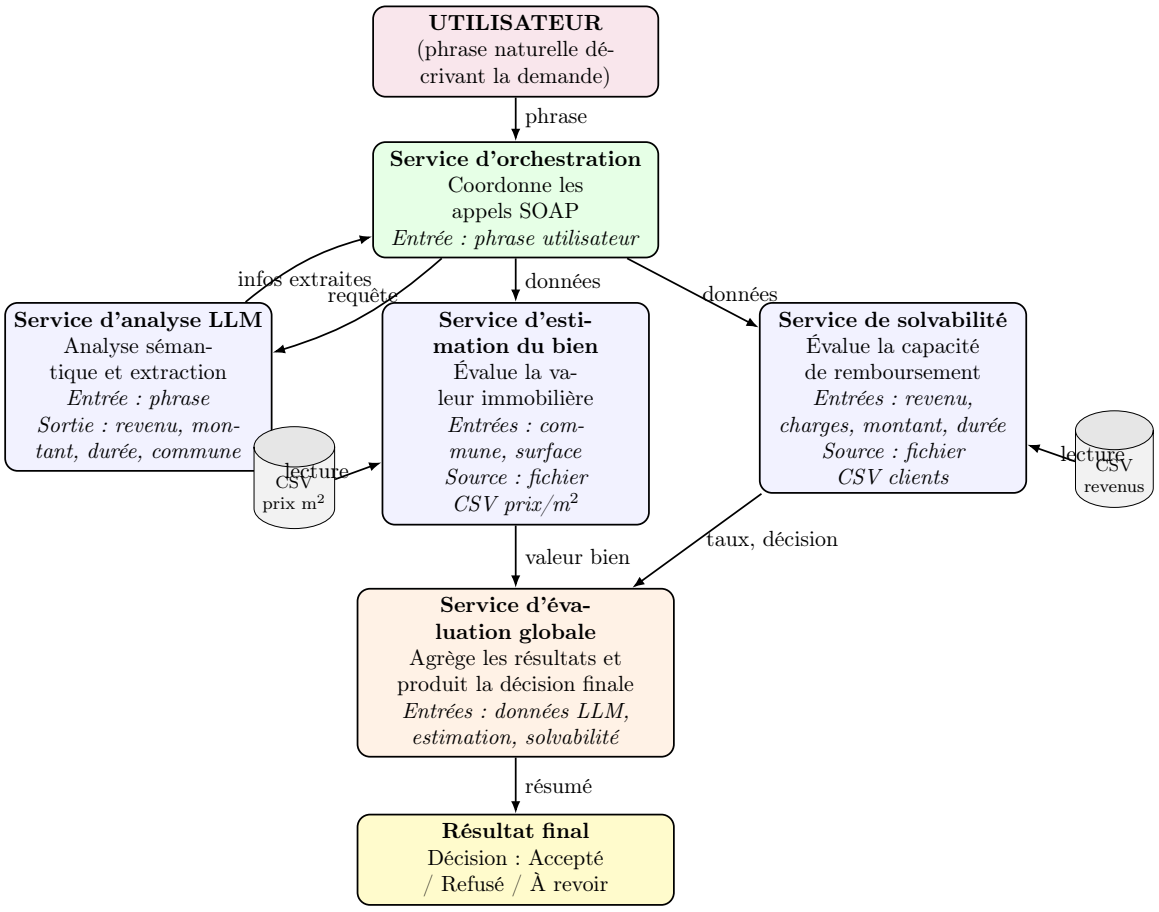

Figure 1 – Architecture SOA du système d'évaluation de demande de prêt immobilier.

b. Identification des services et architecture à base de services

Service	Rôle principal	Variables d'entrée	Source / Communication
Analyse LLM	Extraction automatique d'informations à partir d'une phrase utilisateur en langage naturel (revenu, montant, durée, commune).	phrase utilisateur	SOAP
Estimation du bien	Calcule la valeur d'un bien immobilier selon la commune et la surface, en se basant sur les prix moyens au m^2.	commune, surface	Fichier CSV (prix moyen/m^2) / SOAP
Solvabilité	Évalue la capacité de remboursement du client en calculant son taux d'endettement à partir de ses revenus et charges.	revenu, charges, montant, durée	Fichier CSV (revenus/charges) / SOAP
Évaluation globale	Agrège les résultats des autres services et applique des règles de décision (Accepté / Refusé / À revoir).	résultats des services (LLM, estimation, solvabilité)	SOAP
Orchestration	Coordonne les appels aux différents services et restitue la décision finale à l'utilisateur.	phrase utilisateur	SOAP / Client

c. Modélisation et spécification des services

Service d'analyse LLM

Entrée : phrase naturelle.

Sortie :

```
{"revenu": 3500, "montant": 200000, "duree": 20, "commune": "Lyon"}
```

Service d'estimation du bien

Entrées : commune, surface. **Source :** fichier CSV (prix moyen/m^2).

Sortie :

```
{"valeur_bien": 300000}
```

Service de solvabilité

Entrées : revenu, charges, montant, durée. **Source :** fichier CSV (revenus/charges).

Sortie :

```
{"taux_endettement": 0.29, "decision": "Accepté"}
```

Service d'évaluation globale

Entrées : résultats des services précédents.

Sortie :

```
{"decision_finale": "Accepté", "score_global": 0.91}
```

d. Implémentation et déploiement (sous SPYNE)

Chaque service est exposé via le framework **SPYNE** sous le protocole **SOAP**. Exemple de structure générique :

```
from spyne import Application, rpc, ServiceBase, Float, Unicode
from spyne.protocol.soap import Soap11
from spyne.server.wsgi import WsgiApplication

class EstimationBienService(ServiceBase):
    @rpc(Unicode, Float, _returns=Float)
    def estimation(ctx, commune, surface):
        prix_m2 = get_prix_m2(commune)  # lecture CSV
        return prix_m2 * surface

application = Application(
    [EstimationBienService],
    tns='banque.services.estimation',
    in_protocol=Soap11(),
    out_protocol=Soap11()
)
wsgi_app = WsgiApplication(application)
```

e. Démonstration du processus

Phrase utilisateur :

> « Je gagne 4200 € par mois et je souhaite un prêt de 250 000 € sur 25 ans pour un appartement à Lyon. »

Étapes du processus :

1. Service d'analyse LLM → extrait les données.

2. Service d'estimation du bien → calcule la valeur estimée.
3. Service de solvabilité → calcule le taux d'endettement.
4. Service d'évaluation globale → produit la décision finale.

Résultat final :

```
Commune : Lyon
Valeur estimée du bien : 310 000
Revenu mensuel : 4 200
Taux d'endettement : 29 %
Décision finale : Accepté
```

Conclusion

Cette architecture SOA est **modulaire**, **évolutive** et **interopérable**. Chaque service est clairement défini avec ses entrées et sorties, et la présence du service LLM permet une interaction en langage naturel entre l'utilisateur et le système.

Fin du compte rendu du TP