



IA pour les Opérations de Cockpits Civils

PIE 034

16 Mars 2020

Lorène Authier
Elisa Caudin
Clément Maitre
Mathis Potel



Sommaire

Introduction

I. Organisation des Activités

II. Besoins et Architecture

III. Création de la Maquette

- Implémentation de l'Ontologie
- Les Requêtes
- Jeu de Données Test
- Interface
- Vérification et Validation

IV. Apports et Approfondissements

Conclusion

Introduction

Introduction

Projet et son Contexte



Cockpit d'un Falcon 8X

Introduction

Objectifs du Projet

- Identifier les principaux **scénarii**.
- Définir l'architecture d'une **ontologie** permettant de répondre en priorité aux scénarii identifiés.
- Implémenter en langage **OWL** l'ontologie permettant de modéliser l'avion civil, ses systèmes et son environnement.
- Créer un ensemble d'une **vingtaine de requêtes** standards pour interroger l'ontologie et les implémenter en **SPARQL**.
- Créer un **jeu de données test** cohérent et réaliste pour vérifier le fonctionnement de l'ontologie. Le jeu de données doit simuler un court vol d'une heure.

Organisation des Activités

Organisation des Activités

Organisation de l'Équipe et des Travaux

Organisation de l'Équipe

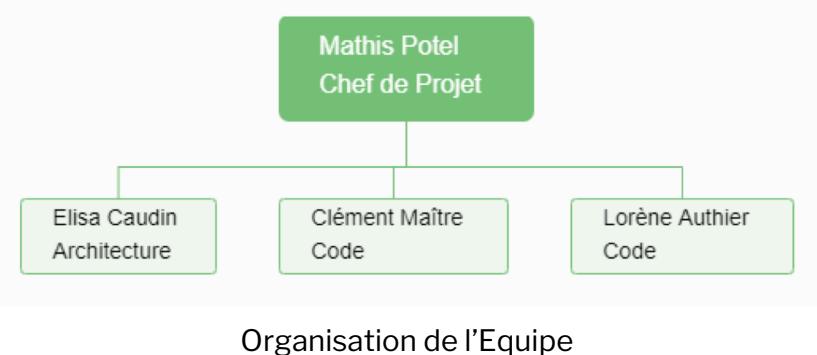
Deux équipes :

- ✈ Architecture
- ✈ Code

Organisation des Travaux

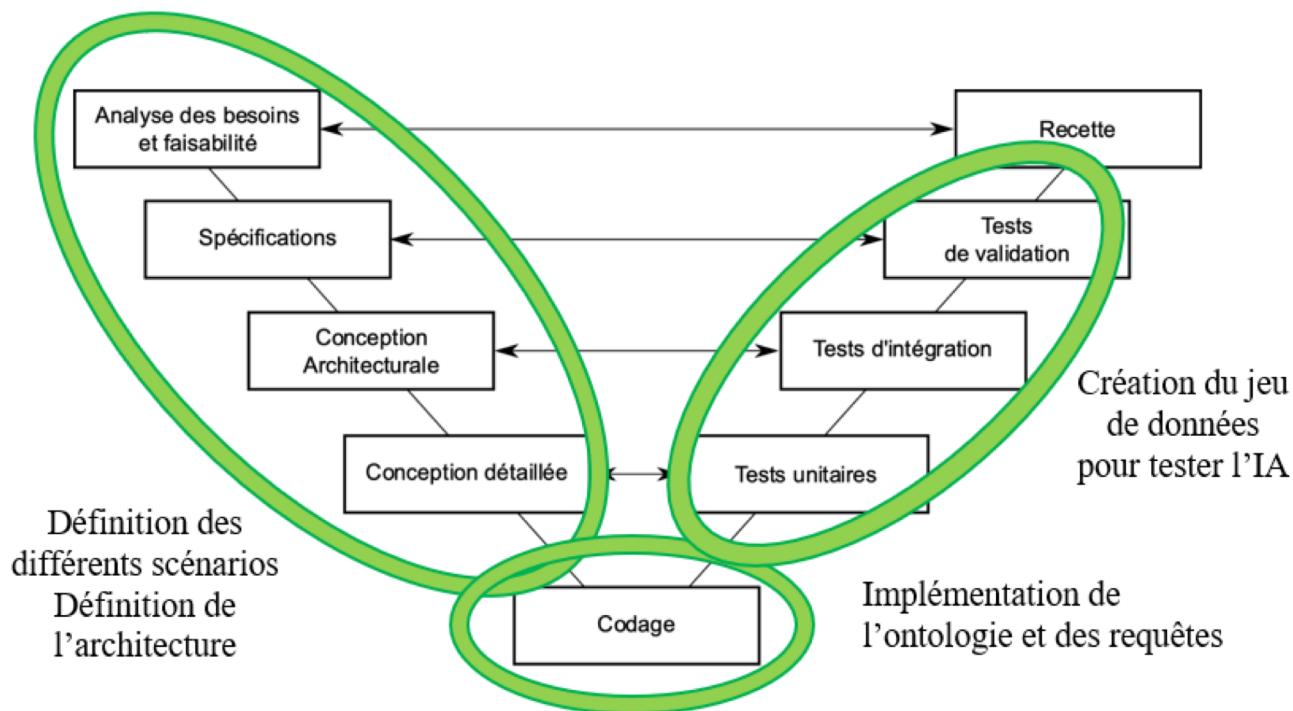
Séance de travail à 4 :

- ✈ Réunions de suivi de l'avancement du projet
- ✈ Entretiens toutes les 2 semaines avec le client



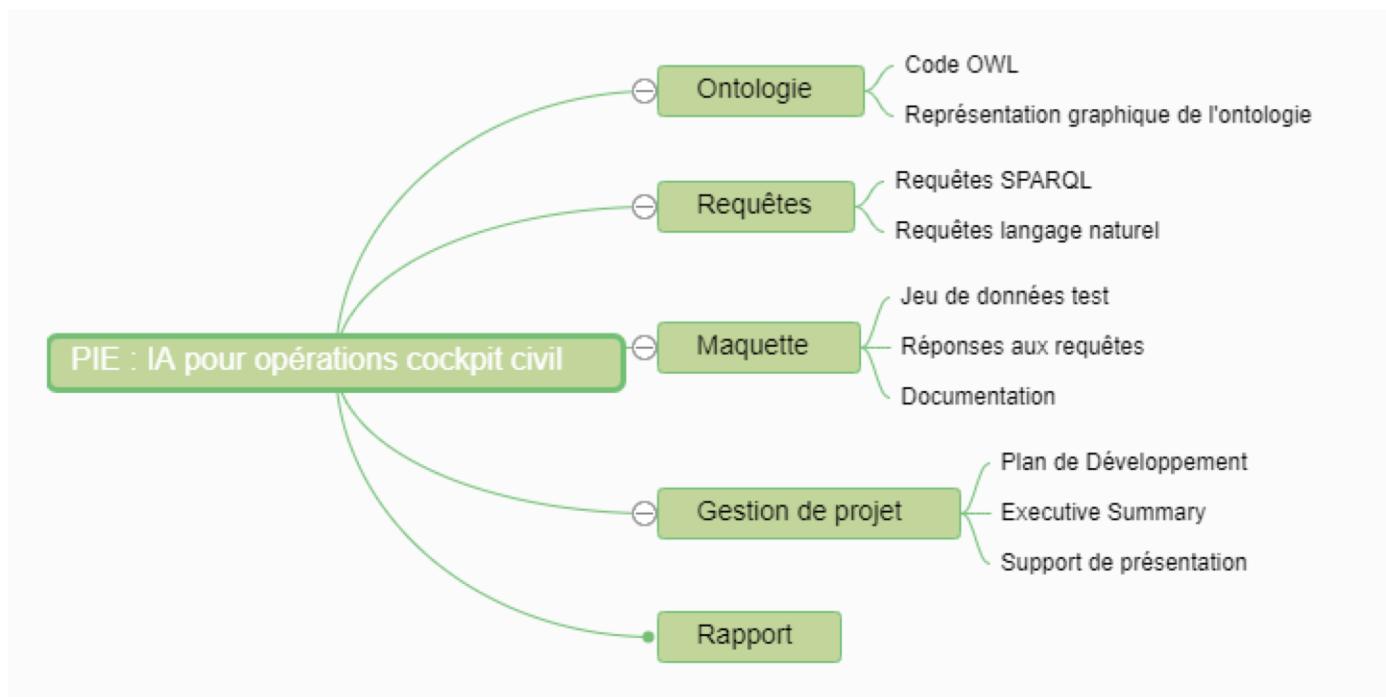
Organisation des Activités

Processus de Développement



Organisation des Activités

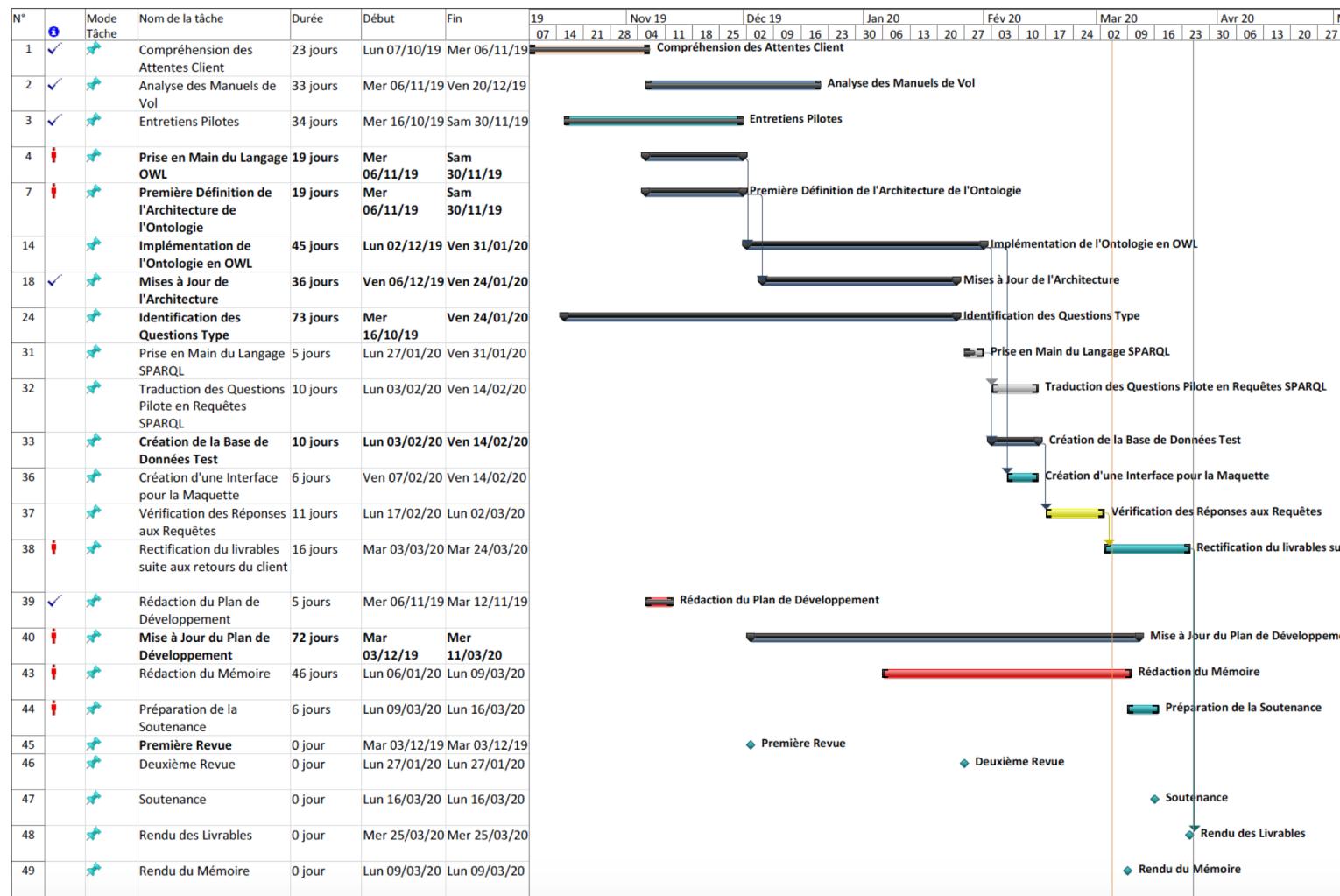
Définition Détailé du Projet - PBS



Organisation des Activités

Planning du Projet (Diagramme de Gantt)





Organisation des Activités

Spécifications du Projet

Contraintes

- Langages utilisés :
 - ◆ OWL → Ontologie
 - ◆ SPARQL → Requêtes
- OS : MacOS, Windows
- Réponses immédiates aux questions
- Réponses non redondantes face aux données déjà disponibles sur le tableau de bord

Hypothèses

- La modélisation reposera sur les données d'un avion de ligne dont les caractéristiques sont plus facilement accessibles
- Nous supposons que notre ontologie a accès à toutes les informations nécessaires grâce aux capteurs ou aux systèmes embarqués de l'avion, notamment le *FMS*

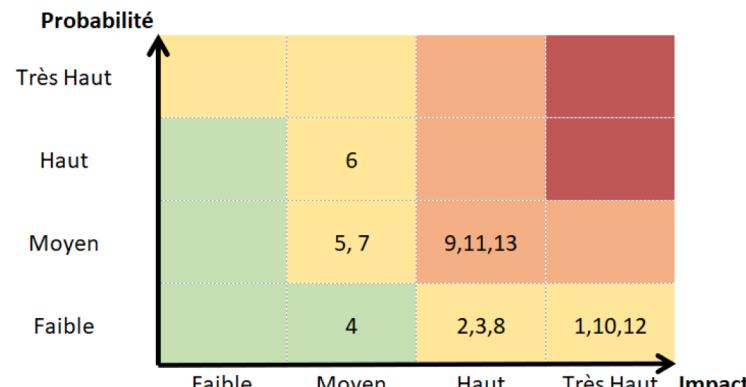
Spécifications

- Ontologie :
 - ◆ Code ontologie → OWL 2.0.0 via l'éditeur Protégé
- Maquette :
 - ◆ OS : Windows ou MacOS,
 - ◆ Python 3.7 / connexion permettant l'installation de packages Python
 - ◆ Le code source : en Python 3.7
 - ◆ La maquette pourra être téléchargée en intégralité via un dossier GitHub
 - ◆ Un manuel d'utilisation de la maquette devra être accessible
 - ◆ La maquette → simulation un vol d'avion civil d'une durée d'une heure.
 - ◆ La maquette → une interface permettant à l'utilisateur de taper une question parmi les requêtes possibles.
 - ◆ La maquette permettra à l'utilisateur de poser plusieurs questions au cours du vol simulé.

Organisation des Activités

Maitrise des Risques

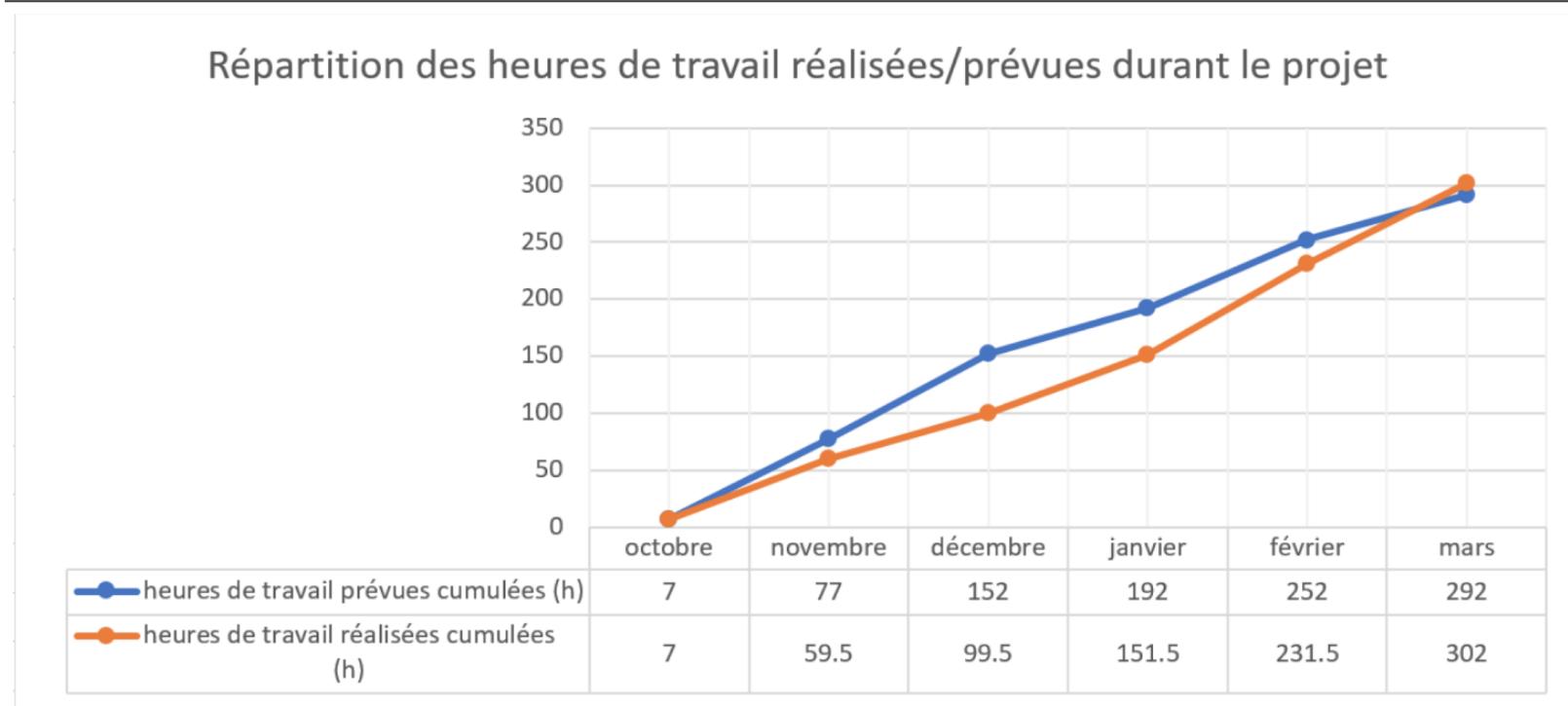
Exemple :



Catégorie	Description			Evaluation pré-risque	
	Causes	Événements à risques	Conséquence	Probabilité	Impact
Technique	Manque de compétence en OWL	Impossibilité d'implémenter le code	Impossibilité d'implémenter l'ontologie	Moyenne	Grave
Impact	Description du risque	Actions préventives	Actions curatives	Responsable de l'action	
Grave	Impossibilité d'implémenter le code	Suivre des formations et tutoriels	Demander de l'aide à une personne compétente pour ce type de langage	Equipe code	

Organisation des Activités

Suivi et Contrôle – Suivi de l'Avancement



Besoins et Architecture

Besoins et Architecture

Identification des Besoins

Objectif Client : Identifier plusieurs cas d'utilisation de l'IA

- Conduite de 3 entretiens à questions ouvertes avec des pilotes
- Etude de manuels de vol

Scénarii identifiés :

- Aide à la navigation
- Conditions météorologiques
- Performances en vol

Besoins et Architecture

Questions et Identification des Données

Scénario	Question	Input Data	Source of the Input Data
Navigation	What is my current achievable range?	Specific Range, Fuel volume	FMS
Navigation	Give me the nearest airport.	A/C live coordinates Airport coordinates	Navigation Database
Weather	Give me the weather at airport ' <i>ICAO_CODE</i> '.	METAR message	FMS
Weather	Do I need to activate the anti-icing system?	Icing Probes	Anti-icing system
Performances	What is my optimal altitude?	Calculated Optimal Altitude	FMS
Performances	Under this configuration of flight, (slats, flaps, spoilers) what is the landing speed to consider?	Position of slats, flaps, spoilers	FMS



Création de la Maquette

Création de la Maquette

Les Ontologies

Ontologie : Ensemble structuré de concepts permettant de modéliser un champ d'informations

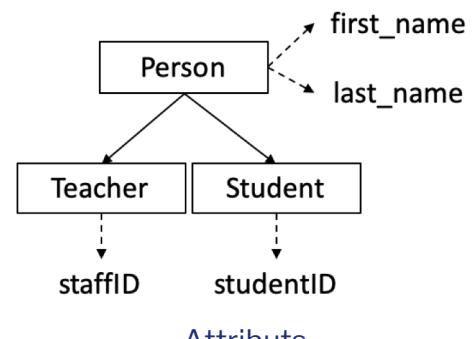
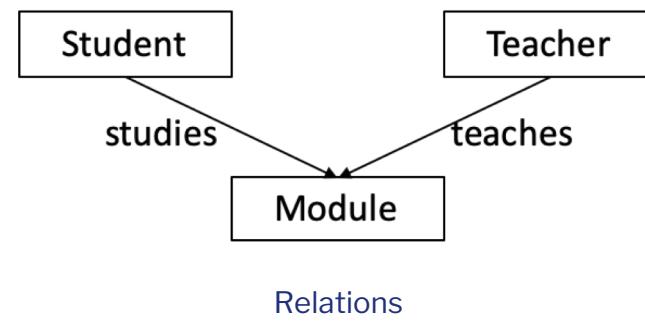
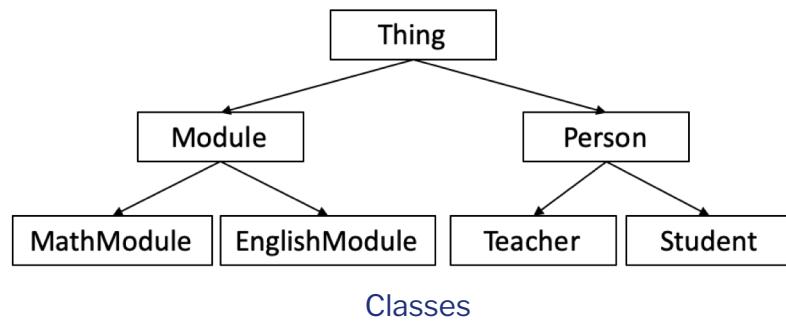
Une ontologie se forme de :

- *Classes* : ensembles d'objets
- *Attributs* : propriétés des objets
- *Relations* : liens entre objets
- *Individus* : objets de base (représentants d'une classe)

Peut s'implémenter grâce au langage **OWL**

Création de la Maquette

Exemple Illustratif d'une Ontologie : Modélisation d'une Université



Teacher1
 Type : Teacher
 teaches : Math101
 last_name : Dupont
 first_name : Jeanne
 staffID : 1234

Student1
 Type : Student
 studies : Math101
 studies : English101
 last_name : Dupond
 first_name : Jean
 studentID : abcd

Math101
 Type : MathModule

English101
 Type : EnglishModule

Création de la Maquette

Implémentation de l'Ontologie (1 / 2)

- À partir de l'architecture, identification des classes, relations et attributs pertinents

Classes :

- | | |
|------------|-------------|
| → Aircraft | → Waypoint |
| → Airport | → Weather |
| → Runway | → Checklist |

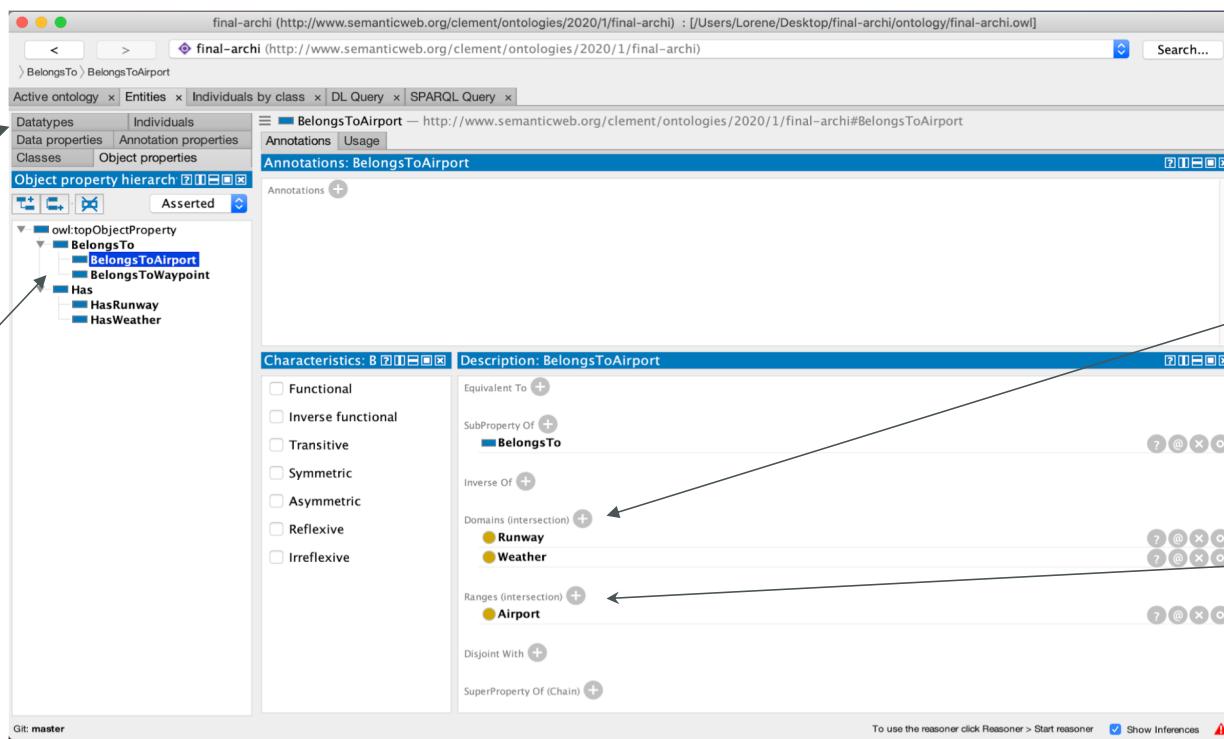
- L'éditeur Protégé permet de définir les éléments de l'ontologie et de générer le code OWL de façon simple, claire et intuitive
- Les individus sont générés automatiquement par un code Python dans la maquette finale

Création de la Maquette

Implémentation de l'Ontologie (2/2)

Navigation entre les classes, relations (Object properties) et attributs (Data properties)

Relations implémentées



Domaine de la relation sélectionnée

Range de la relation sélectionnée

Création de la Maquette

Les Requêtes - Présentation

Implémentation en SparQL des requêtes

En Python : rdflib pour l'ontologie et le requêtage

Utilisation mixte Python et SparQL

Réponse en moins 2 secondes

```
PREFIX pie:<http://www.semanticweb.org/clement/ontologies/2020/1/final-archi#>
SELECT ?windspeed
WHERE {
    ?arrival_airport pie:AirportIsArrival 1 .
    ?weather_at_arrival pie:BelongsToAirport ?arrival_airport .
    ?weather_at_arrival pie:WeatherSpeed ?windspeed .}
```

Requête SparQL pour obtenir la vitesse du vent à l'arrivée

Création de la Maquette

Les Requêtes - Un Exemple de Syntaxe

Un exemple : la météo à Paris-Orly (ICAO : LFPO)

L'utilisateur doit indiquer les informations pertinentes pour sa requête :



Objectif : utiliser le minimum de mots

```
PREFIX pie:<http://www.semanticweb.org/clement/ontologies/2020/1/final-archi#>
SELECT ?Ceiling ?Clouds ?QNH ?Hail ?Orientation ?Speed ?Temp ?Rain ?Snow
WHERE {
    ?Airport pie:AirportICAOCode ?ICAO .
    ?Airport pie:HasWeather ?Weather .
    ?Weather pie:WeatherCeiling ?Ceiling .
    ?Weather pie:WeatherCloudsType ?Clouds .
    ?Weather pie:WeatherQNH ?QNH .
    ?Weather pie:WeatherHail ?Hail .
    ?Weather pie:WeatherOrientation ?Orientation .
    ?Weather pie:WeatherSpeed ?Speed .
    ?Weather pie:WeatherTemperature ?Temp .
    ?Weather pie:WeatherRain ?Rain .
    ?Weather pie:WeatherSnow ?Snow .
    FILTER regex(?ICAO, """ + 'stest' + """ , "i")}
```

La requête correspondante en SparQL

Création de la Maquette

Les Requêtes - Comprendre le Pilote

Défi : traduire la requête du pilote en requête SparQL

- ↗ 20 requêtes possibles dans la maquette finale
- ↗ Instructions au pilote pour rédiger requêtes
- ↗ Reconstruction requête SparQL avec une suite de conditions

Exemple : Météo à Paris-Orly

weather airport LFPO

Structure des conditions :

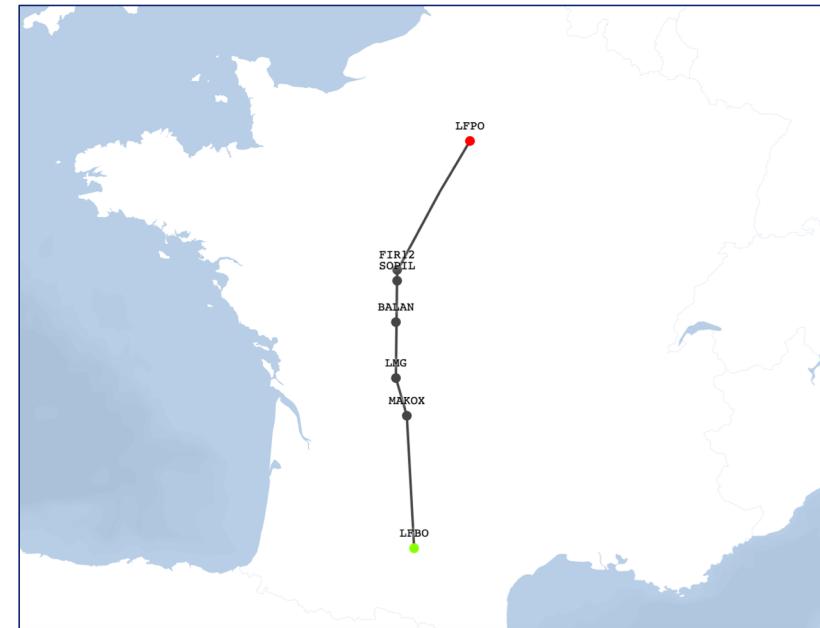
```
if "weather" in user_input:  
    if "airport" in user_input:  
        icao = user_input.split(' ') [-1]  
        request_airport_weather(icao)
```

Création de la Maquette

Jeu de Données Test

Simulation d'un vol Toulouse-Orly en A320neo

- Création du plan de vol
- Simulation du vol et enregistrement des données à l'aide du logiciel *FlightGear*
- Post-traitement et formatage automatique des données grâce à un script Python



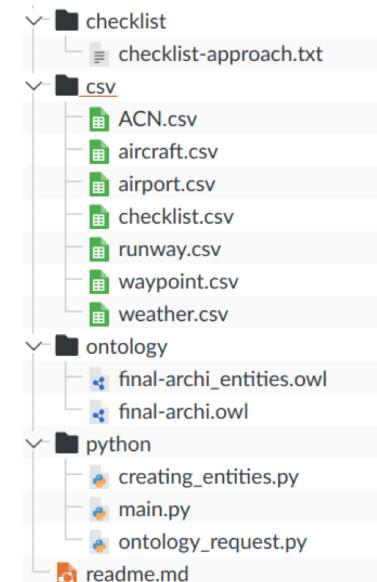
Plan de vol Toulouse-Orly simulé

Création de la Maquette

Interface - Présentation

Objectif : réunir l'ontologie, le requêtage, la lecture des données de vol et l'interface dans un même script Python

- Interface simple utilisable rapidement par un pilote
- Affichage de la réponse aux requêtes immédiat
- Portable sur différentes machines sans modification du code



Structure des fichiers de la maquette

Création de la Maquette

Interface - Fonctionnement

- Pour lancer la maquette, exécuter le script `main.py`. L'interface apparaît.
- Un script Python crée les objets (aéroports, avion, ...) dans l'ontologie à partir des données en `.csv`
- À chaque pas de temps, les objets sont mis-à-jour dans l'ontologie
- À chaque requête utilisateur, un script effectue la requête SparQL et affiche le résultat

Modules Python utilisés :

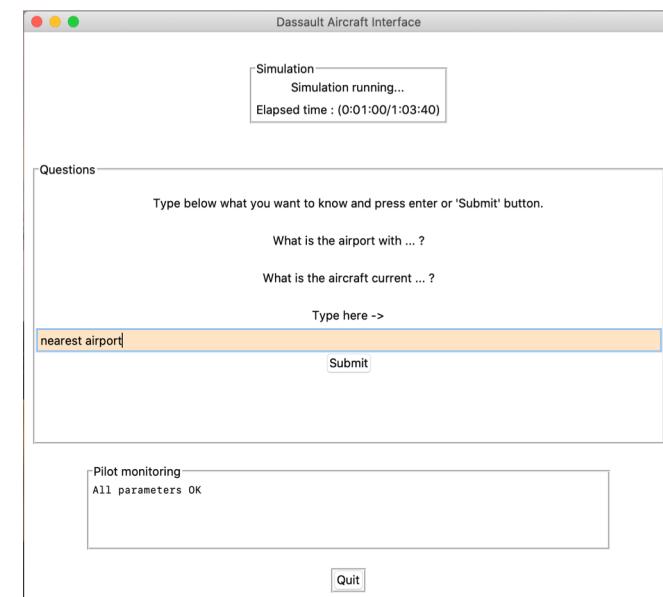
- `pandas` : lecture `.csv`
- `owlready2` : lecture et écriture ontologie
- `rdflib` : requêtes sur ontologie
- `tkinter` : interface graphique

→ Open-source et maintenus, compatibles toutes plateformes

Création de la Maquette

Interface - Utilisation

- Lancer maquette : `python main.py`
- L'utilisateur appuie sur **Start Simulation** pour lancer la simulation de vol
- Il appuie sur **Start Questions** pour lancer l'interface de requêtes
- Il tape une requête et presse Entrée pour avoir la réponse



Exemple d'une requête dans l'interface

Création de la Maquette

Vérification et Validation

- Exécuter l'ensemble des requêtes à intervalles réguliers tout au long du vol simulé
- Vérifier l'exactitude des réponses en les comparant avec les réponses calculées directement à partir des données du tableau de données

Mesure n°	temps	Tableau de Données		Réponse Maquette		Tableau de Données		Réponse Maquette	
		19.02	16.08	9	nearest airport	nearest airport landing	Toulouse	Bordeaux	Paris
1	00:20	131,77	132	Toulouse	Toulouse	Toulouse	Toulouse	Bordeaux	Paris
2	06:40	186,03	186	Toulouse	Toulouse	Toulouse	Toulouse	Bordeaux	Paris
3	13:00	185,48	185	Toulouse	Toulouse	Toulouse	Toulouse	Bordeaux	Paris
4	19:20	185,08	185	Agen	Agen	Bordeaux	Bordeaux	Bordeaux	Paris
5	25:40	184,65	185	Bordeaux	Bordeaux	Bordeaux	Bordeaux	Bordeaux	Paris
6	32:00	184,24	184	Paris	Paris	Paris	Paris	Paris	Paris
7	38:20	184,07	184	Paris	Paris	Paris	Paris	Paris	Paris
8	44:40	183,99	184	Paris	Paris	Paris	Paris	Paris	Paris
9	51:00	183,83	184	Paris	Paris	Paris	Paris	Paris	Paris
10	57:20	128,62	129	Paris	Paris	Paris	Paris	Paris	Paris

Création de la Maquette

Démonstration



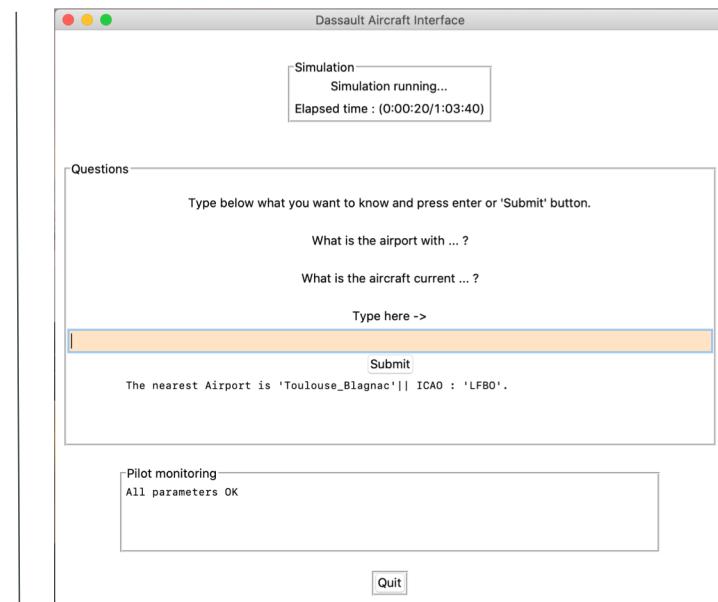
Apports et Approfondissements

Appports et approfondissement

Pilot Monitoring

Objectif : alléger la charge mentale du pilote

- ❖ À chaque pas de temps vérification des paramètres avions (IAS, altitude, ...)
- ❖ Si pas de problème, on affiche All parameters OK
- ❖ Si problème, on affiche les paramètres à vérifier
- ❖ Affichage intégré à l'interface



On voit dans le cadre inférieur que tous les paramètres sont bons

Conclusion

Références

Diapositive de titre :

[https://en.wikipedia.org/wiki/Dassault_Aviation#/media/File:Dassault_Falcon_7X_OY-VIK_\(7416828394\).jpg](https://en.wikipedia.org/wiki/Dassault_Aviation#/media/File:Dassault_Falcon_7X_OY-VIK_(7416828394).jpg)

Cockpit d'un Falcon 8X: <https://samchui.com/2018/03/04/flying-dassault-falcon-8x-private-jet/>