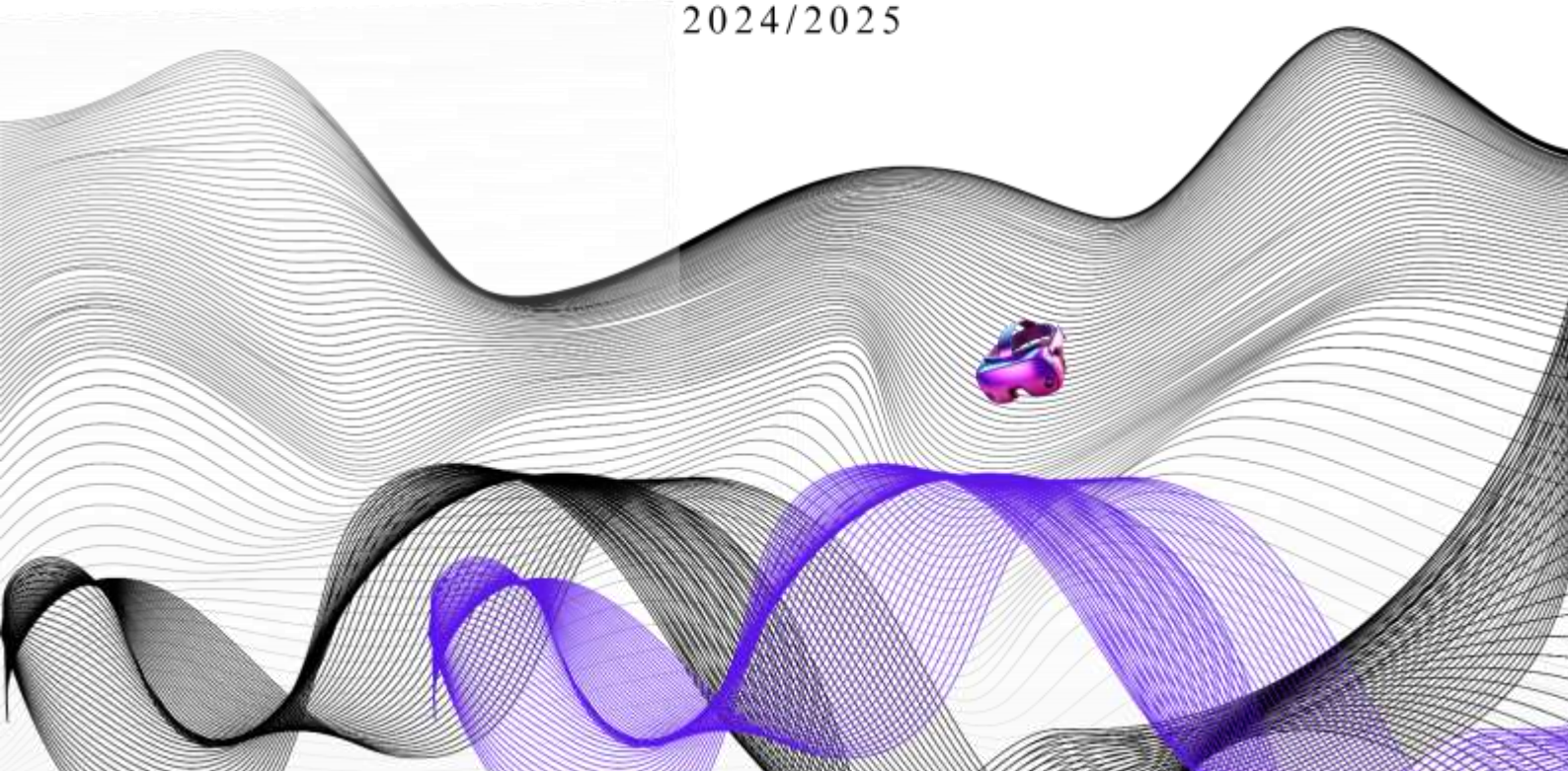
Higher Institute of Computer Science at Mahdia
Master's Degree in Data Science
M2 Level

VIRTUAL REALITY ONTOLOGY

An Academic Project for the Subject: Semantic Web

REALIZED BY:
JEBAI WAFA
BELANEZ HAMDI
SUPERVISED BY:
LEÏLA BAYOUDHI
ACADEMIC YEAR:
2024/2025



“When we get lost between the existing and unexisting,

Ontologies come to reconnect these two universes,

bringing a structure to the unstructured World.”

By Jebali Wafa.

Table of Contents

Table of Figures

1. Overview

1.1 Introduction

Virtual reality, or VR, is a reproduction of the real world or a completely imaginary universe. The experience is both visual and auditory, and in some cases, haptic with the production of feedback. When the person is equipped with the appropriate interfaces, such as gloves or clothing, they can then experience certain sensations related to touch or specific actions. (coup, impact...).

Virtual reality has several essential features that make it an immersive and interactive medium. Unlike traditional interfaces, VR positions the user inside the virtual environment, delivering a truly immersive experience. It is applied in several fields, although it is known in the Gaming field, today VR is present in the Education, training, Job Collaboration and HealthCare fields.

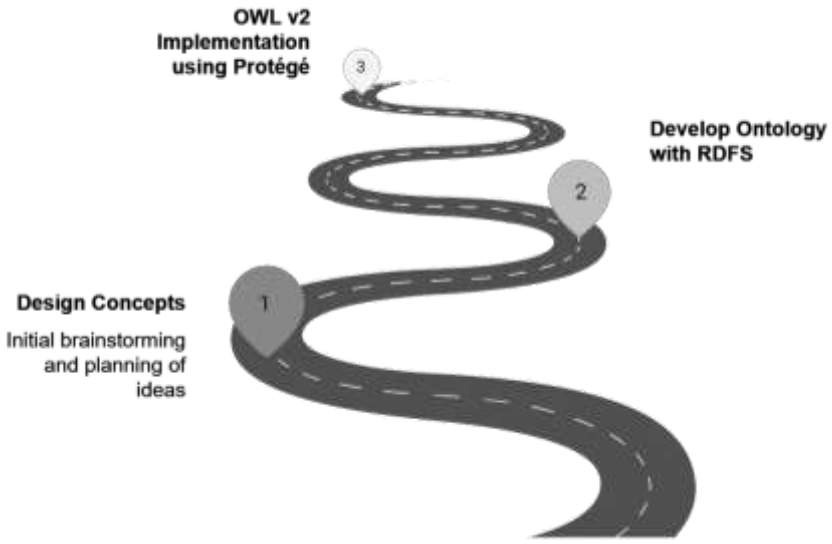
1 Virtual Reality Ontology Image Generated By Dall-E based on our promt

1.2 Aims and Objectives

This project aims to design and create an Ontology based on a semantic structure for the Immersive Virtual Reality Concept to represent the existing and non-existing entities and relationships provided by this concept, using RDFS, leveraging it to OWL v2, with the help of Protégé.

2. Methodology

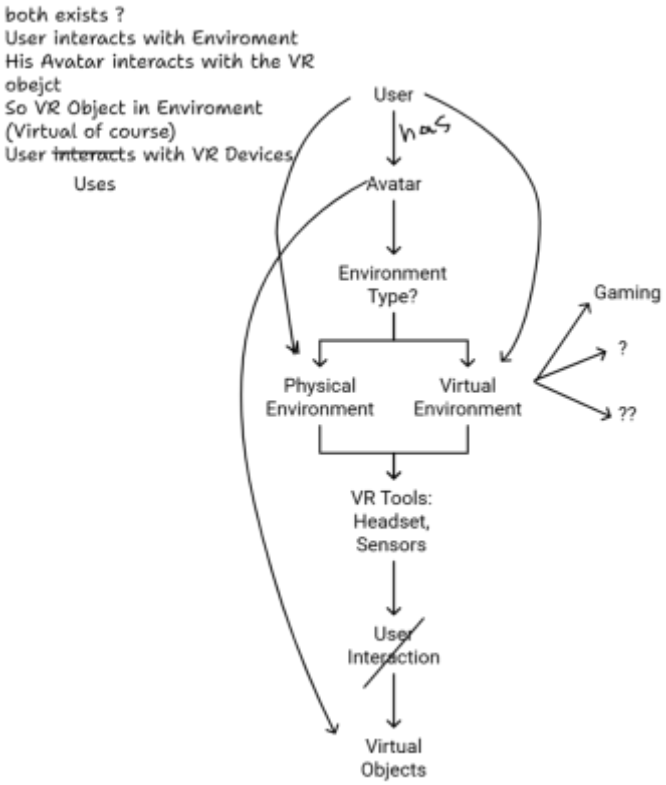
We created our Ontology through multiple steps, starting with brainstorming, then developing the structure with RDFS and then finish the converting the whole process with OWLv2 to bring the semantic concept of the Ontology to life.



2 Ontology Development Process

2.1 Conceptual Design

As IT enthusiasts, we still not experts in the VR domain, we have made some researches to capture the key concepts and have made various Brainstorming sessions and made several ideas plans to make the first basic structure of the VR Knowledge model, find below the initial brainstorming plan we came up with.



3 Initial Brainstorming Plan

2.2 Ontology Development in RDFS

RDFS (RDF-Schema), an extension of RDF, allows for the construction of lightweight ontologies based on RDF.

It provides mechanisms for describing groups of related resources and the relationships between these resources. RDF Schema is written in RDF using the terms described in this document. These resources are used to determine characteristics of other resources, such as the domains and ranges of properties.[1]

2.2.1 Classes

[1] W3C RDF Schema 1.1(Dan Brickley and R.V. Guha, Google 2014)

Class	SubClass	Description
User	MultiUserParticipant	A user that interacts with other users
	Player	User that usually interacts within Gaming or Training Environment
	Spectator	User that usually interacts within a Social Environment
Avatar	AbstractAvatar	User representation in a geometrical shape...
	HumanAvatar	User representation in a human shape.
	NonHumanAvatar	User representation in animal shape...
VirtualRealityDevice	Headset	VR device used to render the Graphic Environment.
	HandController	Controllers used to interact with VR environment.
	AudioSystem	VR device that renders the Audio effects.
	MotionTrackingSystem	VR device sensor that tracks the motion of the User to converted into the Virtual Environment.
Physical-Environment	Indoor	Closed space could be the Lab,Home..
	Outdoor	Open space, could be Park,Garden...
Virtual-Environment	GameWorld	Graphic Environment within a Game software.
	SocialEnvironment	Graphic Environment within a Software for Job meetings or Job simulations...
	TrainingEnvironment	Graphic Environment within a software for Training or Education ...
VirtualObject	Dynamic	A moving object within the Virtual Environment .
	Static	A static object within the Virtual Environment, coudl be Table .....
Feedback	Auditory	an Audio feedback rendered by a VR Device.
	Visual	a graphic visuals rendered by a VR Device.

4 Classes and Subcalasses Table

To create the Classes and subclasses hierarchy listed above we used RDFS to define the structure of ontology, find below an example of developing a Class and subclass using **rdfs:Class** and **rdfs:subClassOf**.

```
:User rdf:type rdfs:Class .
:MultiUserParticipent rdf:type rdfs:Class ;
      rdfs:subClassOf :User .
:Player rdf:type rdfs:Class ;
           rdfs:subClassOf :User .
:Spectator rdf:type rdfs:Class ;
                rdfs:subClassOf :User .
```

5 RDFS Class Script

2.2.2 Properties

Property	SubProperty	Description
belongsTo		Links a VirtualObject to a VirtualEnvironment
tracksMotion		Links a VirtualRealityDevice to a User
isUsedBy		Links a VirtualRealityDevice to a User
uses		Links a User to a VirtualRealityDevice
existsIn	UserExistsInPhysicalEnvironment	Links Avatar or User to PhysicalEnvironment or VirtualEnvironment
	AvatarExistsInVirtualEnvironment	
interactsWith	UserInteractsWithVEnv	Links User or Avatar to VirtualEnvironment or VirtualObject
	AvatarInteractsWithVOb	
enhancesExperienceOf		Links a VirtualRealityDevice to a VirtualEnvironment
contains	PhysicalEnvironmentContains	Links a VirtualEnvironment to a VirtualObject
	VirtualEnvironmentContains	
providesFeedback		Links a VirtualRealityDevice to Feedback
represents		Links Avatar to User
hasAvatar		Links User to Avatar

6 Properties and SubProperties Table

In order to represent those relationships using RDFS we proceeded like this :

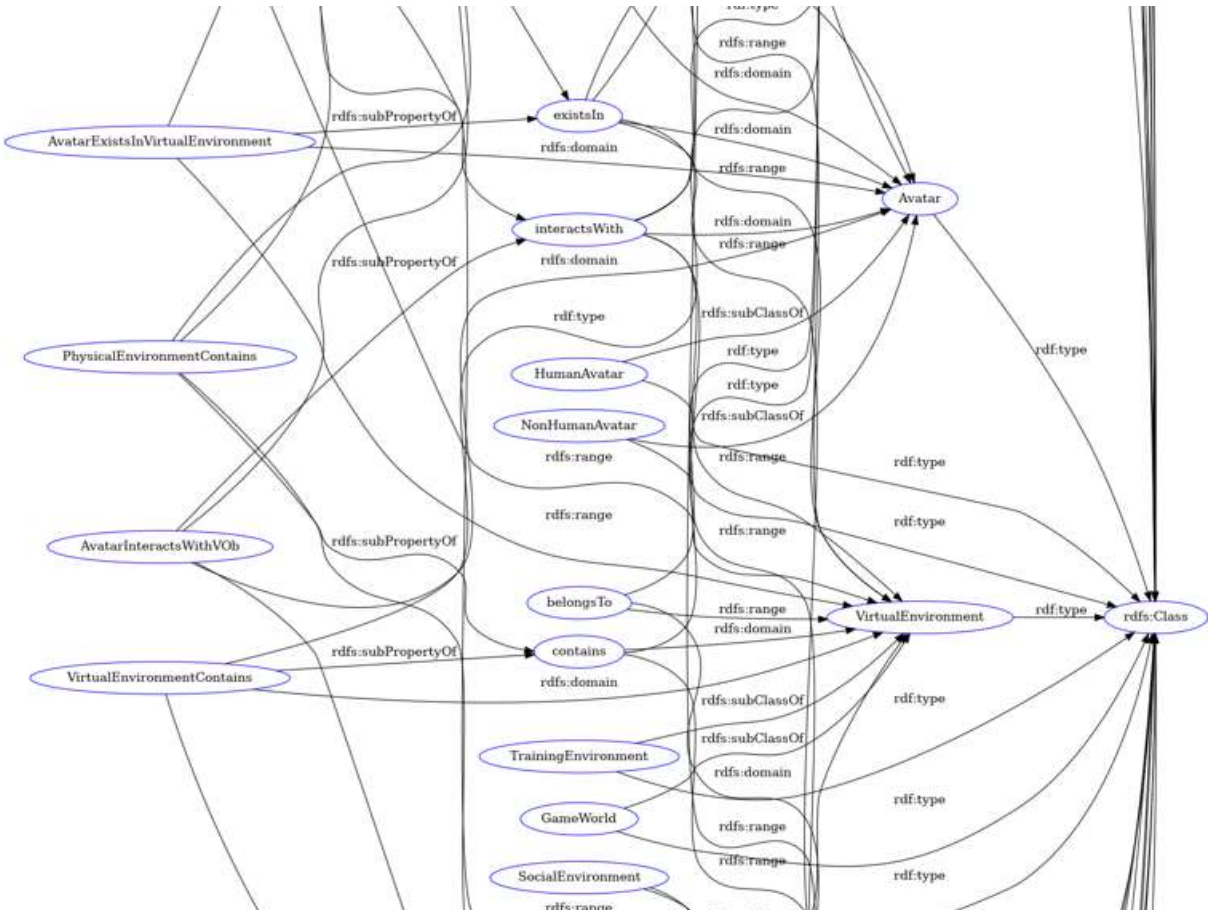
```
# Base property
:existsIn rdf:type rdf:Property ;
    rdfs:domain :Avatar, :User ;
    rdfs:range :PhysicalEnvironment, :VirtualEnvironment .

# Subproperties of existsIn
:UserExistsInPhysicalEnvironment rdf:type rdf:Property ;
    rdfs:subPropertyOf :existsIn ;
    rdfs:domain :User ;
    rdfs:range :PhysicalEnvironment .

:AvatarExistsInVirtualEnvironment rdf:type rdf:Property ;
    rdfs:subPropertyOf :existsIn ;
    rdfs:domain :Avatar;
    rdfs:range :VirtualEnvironment .
```

7 RDFS Property script

We have accessed to RDF Grapher[2] website to create our graph using the RDFS TURTLE script, find below an overviex of the RDFS Graph:



8 RDFS Graph Overview By RDF Grapher

[2] (https://www.ldf.fi/service/rdf-grapher n.d.)

2.3 Conversion to OWL V2:

OWL (Web Ontology Language) is an extension of RDFS. It allows for greater expressiveness in the definition of ontologies.

It notably introduces the concepts of class or property equivalence, or the identity of two resources. In general, it allows for the introduction of logical relationships between elements of the same vocabulary, but also between distinct ontologies: this is referred to as alignment.

Although RDFS is used to create Semantic ontologies , we seems to be far from semantic reasoning even with the Properties and subProperties created, a machine still unable to infer relationships correctly for example:

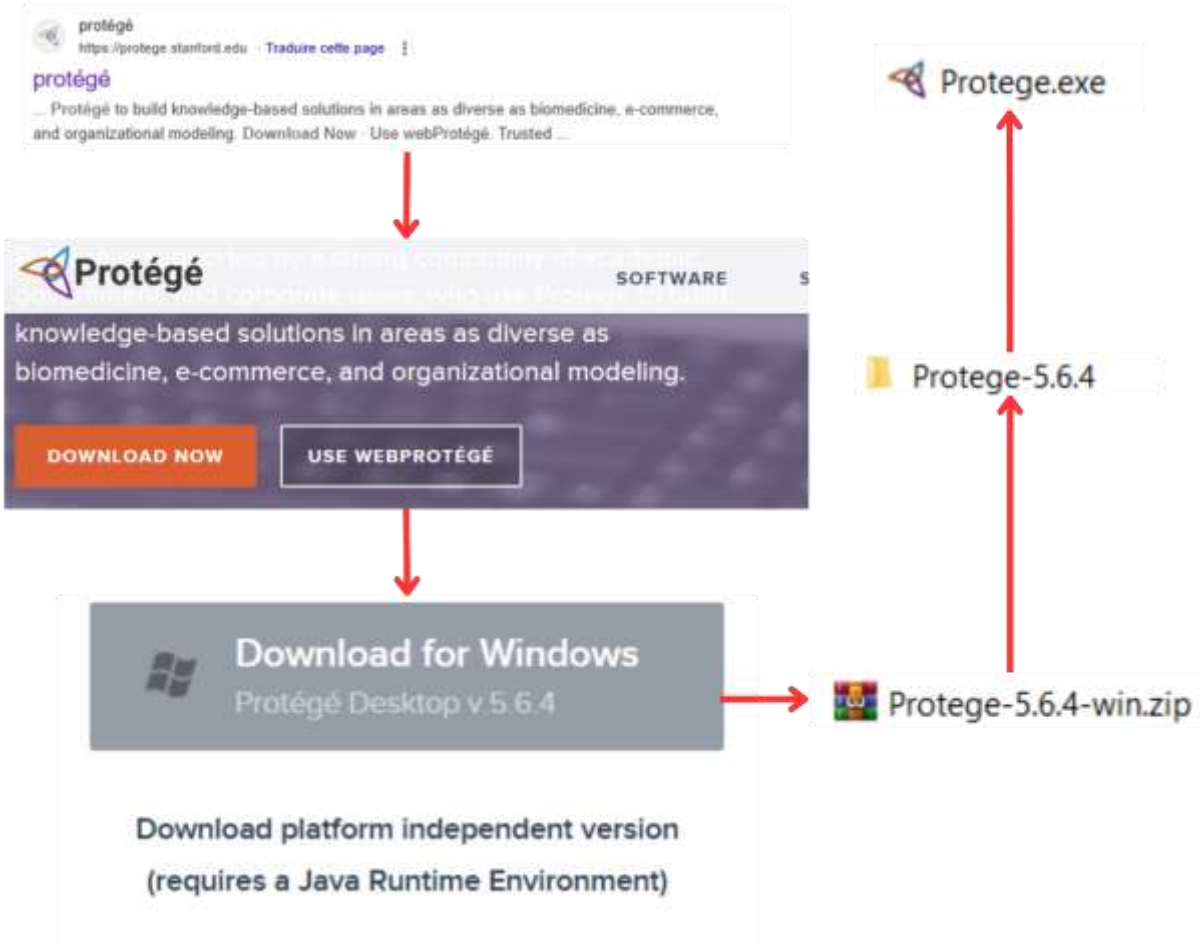
- If we say Avatar **existsIn** Virtual Environment what does guarantee that Avatar is not a **VirtualObejct** ?
- Or if w instantiate **WafaAvatar** as **HumanAvatar** that represents **Wafa** (a **User**) what will guarantee that **Wafa** dosen't have two avatars in that same Environment ? Or what will guarantee that **WafaAvatar** isn't the same **HamdiAvatar** ?

That's why we chose to use Owl2 that will make these complex relationships and rules; that seems simple to the human mind but trust us , it is way complex to the machines; more clear.

2.4 Implementation in Protégé

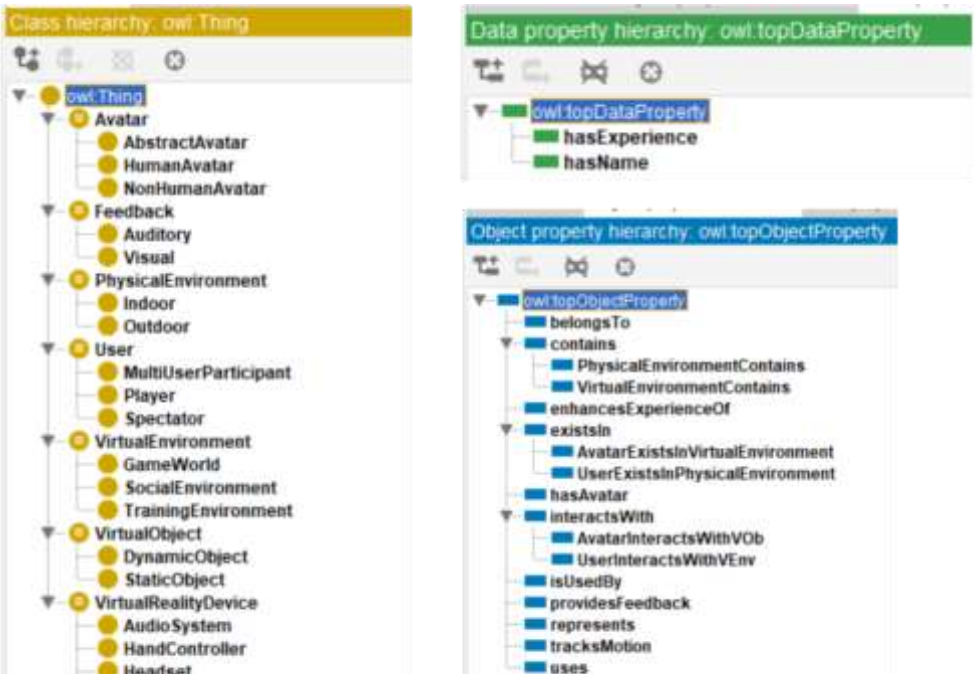
2.4.1 Protégé Installation setup:

We proceeded to the official Protégé website and followed as the figure below states.



10 Protégé Installation Steps

2.4.2 Ontology Creation:



9 Ontology Hiearchy Overview

Then we Created our entities : Classes, Subclasess,Obejct Properties, DataTypeProperties…

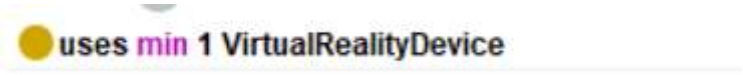
We instantiated some individuals too :

11 Annotation example

12 Individuals Overview

Then we added some axioms to our ontology, for example

- A User could use at minimum one VirtualRealityDevice (if we don't have VirtualRealityDevice we are not talking then about VR) :



13 Cardinality Example

- Also a User have a **Target Key** which is **Unique** using the **DataTypeProperty hasName:**

14 Target key Example

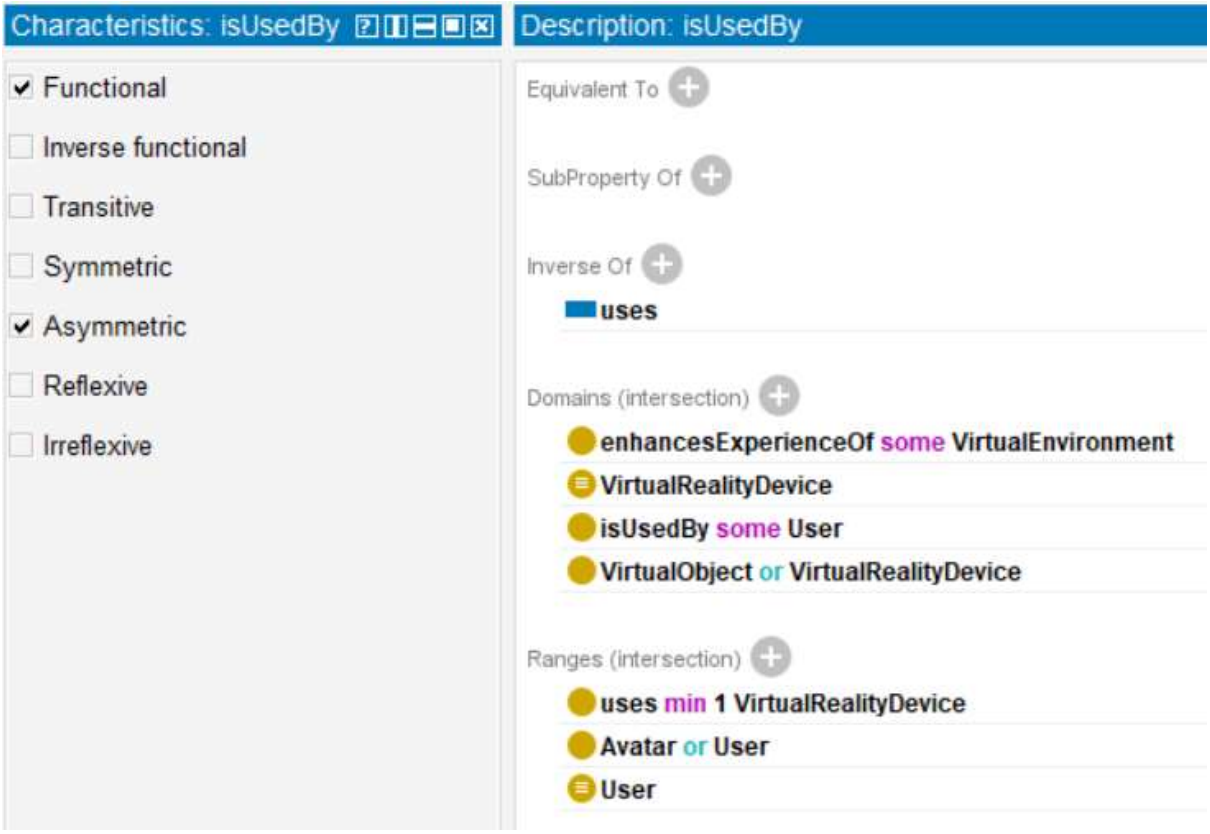
- Now if a User is and Individual of “User” Class what will guarantee that this same instance is not an Aavatar right ?? In order to guarantee this Disjunction we used this axiom:

15 Disjunction Example

- A User could be either a MultiUserPlayer or Player or Spectator , to represent this axiom we used :

16 Disjunction Union Example

- For an Object poperty **isUsedBy** is a :
  - Functional Property because a device can be used only by one User.
  - InverseOf the **uses** property.
  - Asymmetric : a User canno’t be **isUsedBy** a device .



17 Property Types Example

Using those Logical and non-logical axioms (Properties types, Cardinalities..) we made our ontology more consistent, coherent and decidable.

Metrics

Axiom	458
Logical axiom count	389
Declaration axioms count	63
Class count	26
Object property count	17
Data property count	2
Individual count	19
Annotation Property count	3

18 Statistics Ontology Overview

However, there are still several gaps that need to be addressed by establishing rules and constraints.

3. Ontology Rules and Constraints

3.1 SWRL Rules

In this section, we define a set of Semantic Web Rule Language (SWRL) rules that govern the interactions within our virtual environment system. These rules help to formalize the relationships and behaviors between users, devices, avatars, and virtual objects. Below are the key rules implemented:

- If a user interacts with a device that enhances the experience of a virtual environment, then the user is considered to be interacting with that virtual environment.

Name
R1
Comment
User Interacts with Virtual Environments through Devices
Status
Ok
User(?u) ^ uses(?u, ?device) ^ enhancesExperienceOf(?device, ?env) -> UserInteractsWithVEnv(?u, ?env)

19 SWRL Rule 1 Example

- If an avatar exists in a virtual environment and a virtual object belongs to that environment, then the avatar interacts with the virtual object.

Name
R2
Comment
Avatar with vobj in same ENv
Status
Ok
AvatarExistsInVirtualEnvironment(?avatar, ?env) ^ belongsTo(?vobj, ?env) -> AvatarInteractsWithVOb(?avatar, ?vobj)

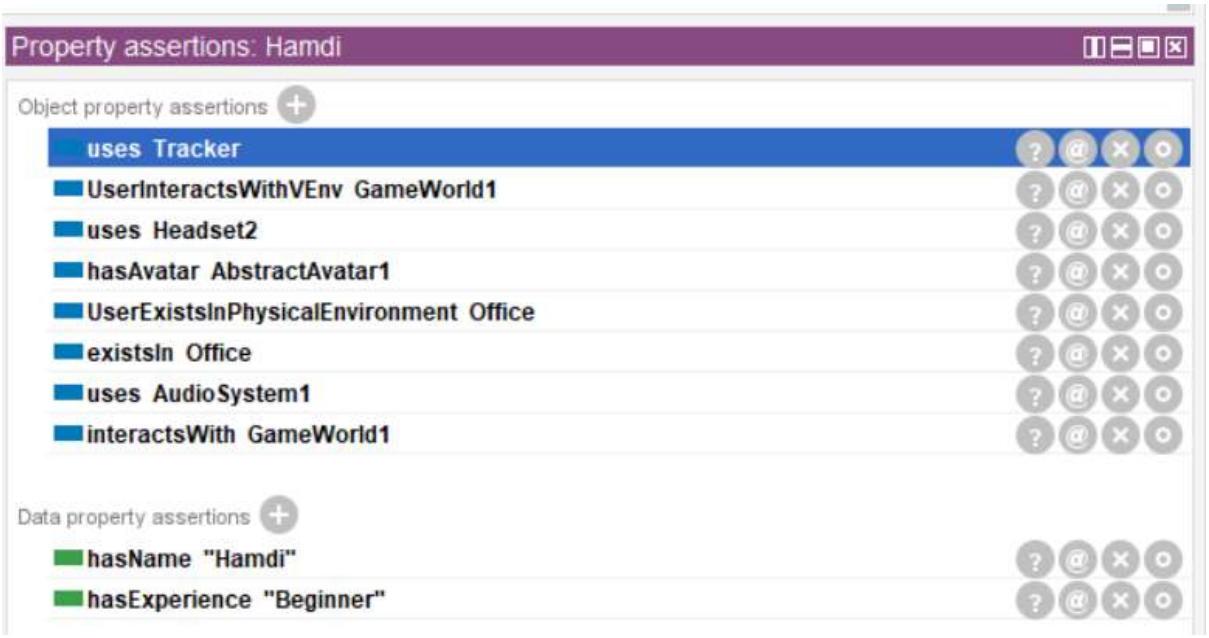
20 SWRL Rule 2 Example

- If a user interacts with a virtual environment and the environment is enhanced by a virtual reality device, then the user uses that device.

Name
R6
Comment
If a User interacts with a Virtual Environment, then the User uses a Virtual Reality Device:
Status
Ok
User(?u) ^ UserInteractsWithVEnv(?u, ?ve) ^ VirtualEnvironment(?ve) ^ VirtualRealityDevice(?vrd) ^ enhancesExperienceOf(?vrd, ?ve) -> uses(?u, ?vrd)

21 SWRL Rule 3 Example

To showcase the use of the last mentioned rule in the ontology, we deleted the uses property between Hamdi (a User individual) and Tracker (a VirtualRealityDevice individual). After synchronizing the reasoner and running our rules on the ontology, we found that the uses Tracker relationship was inferred. Upon checking the explanation of this inference, we discovered that one of the reasons was the SWRL rule.



22 Property Assertions of User Individual Hamdi

23 Explanation of Inference

3.2 SHACL Constraints

In this section, we define the SHACL (Shapes Constraint Language) constraints that are applied to our ontology to ensure data integrity and consistency. SHACL constraints help validate the structure and relationships within the ontology, ensuring that it adheres to the specified rules and requirements. Below are the key constraints implemented:

- Cardinality Constraint: This constraint ensures that a User must have at least one device and can have at most four devices. This helps in managing the relationships between users and devices, ensuring that each user has a reasonable number of associated devices.

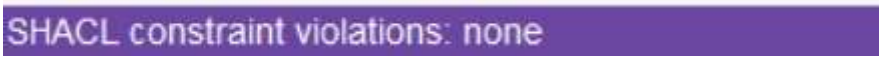
```
# Cardinality Constraints
ex:UserDeviceCardinalityShape
  a sh:NodeShape ;
  sh:targetClass ex:User ;
  sh:property [
    sh:path ex:uses ;
    sh:minCount 1 ;  # User must have at least one device
    sh:maxCount 4 ;  # User can have at most four devices
  ] .
```

24 Shacl Constraint 1

- Property Presence and Type Constraints: This constraint ensures that an Avatar can interact with at most one VirtualObject, and if it does, the VirtualObject must be of the correct type.

```
# Property Presence and Type Constraints
ex:AvatarPresenceAndTypeShape
  a sh:NodeShape ;
  sh:targetClass ex:Avatar ;
  sh:property [
    sh:path ex:AvatarInteractsWithVOb ;
    sh:minCount 0 ;
    sh:maxCount 1 ;  # Avatar must interact with 0 or 1 VirtualObject
    sh:node ex:VirtualObject ;  # The VirtualObject must be of the correct type
  ] .
```

25 Shacl Constraint 2



26 Shacl Constraint Violations

After setting all the Shacl Constraints we checked for any violations, and it seems that our Ontology is well structured and follows those constraints.

4. Querying the Ontology using SPARQL

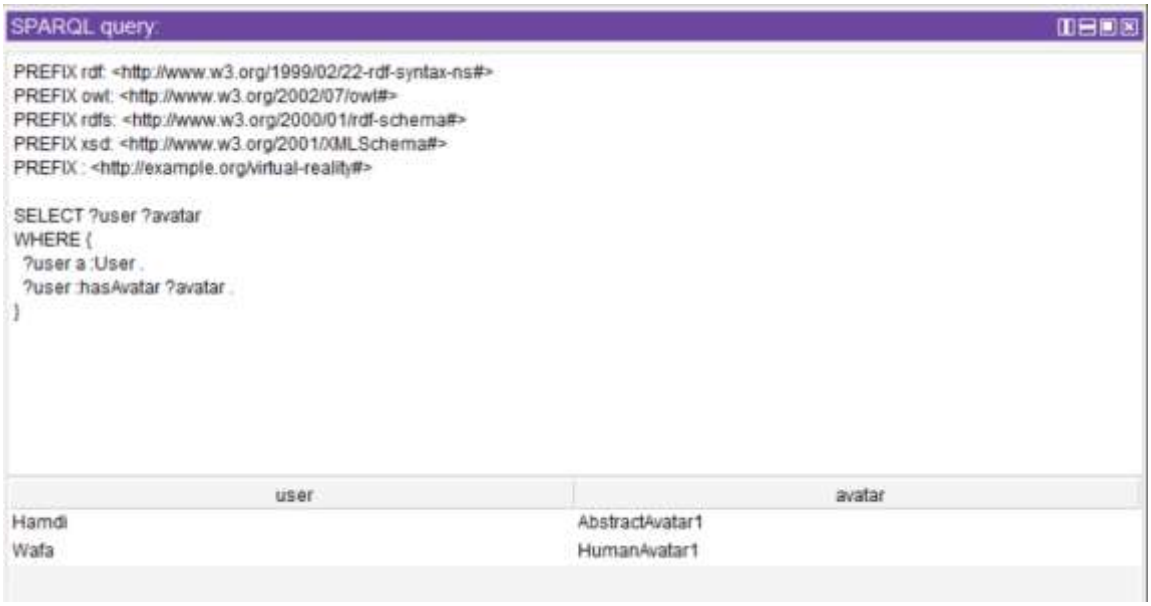
Now that we have designed a decidable ontology and created individuals, it's time to start querying our data using SPARQL. SPARQL (SPARQL Protocol and RDF Query Language) is a powerful query language and protocol used for

retrieving and manipulating data stored in Resource Description Framework (RDF) format. It allows us to perform complex queries to extract meaningful information from our ontology.

SPARQL queries consist of a set of triple patterns, similar to RDF triples, which are used to match the data in the ontology. These queries can be used to retrieve specific data, filter results based on certain criteria, and even perform aggregations and calculations.

Here ae some SPARQL queries we used to question the Ontology :

- Find Users and their corresponding Avatars:



27 SPARQL Query 1

- Retrieve Users, Their Avatars, and Devices if exist(Optional)

28 SPARQL Qyery 2

- Count for each user the devices used :

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX : <http://example.org/virtual-reality#>


SELECT ?user (COUNT(?device) AS ?DeviceCount)
WHERE {
 ?device rdf:type :VirtualRealityDevice .
 ?user :uses ?device .
}
GROUP BY ?user
```

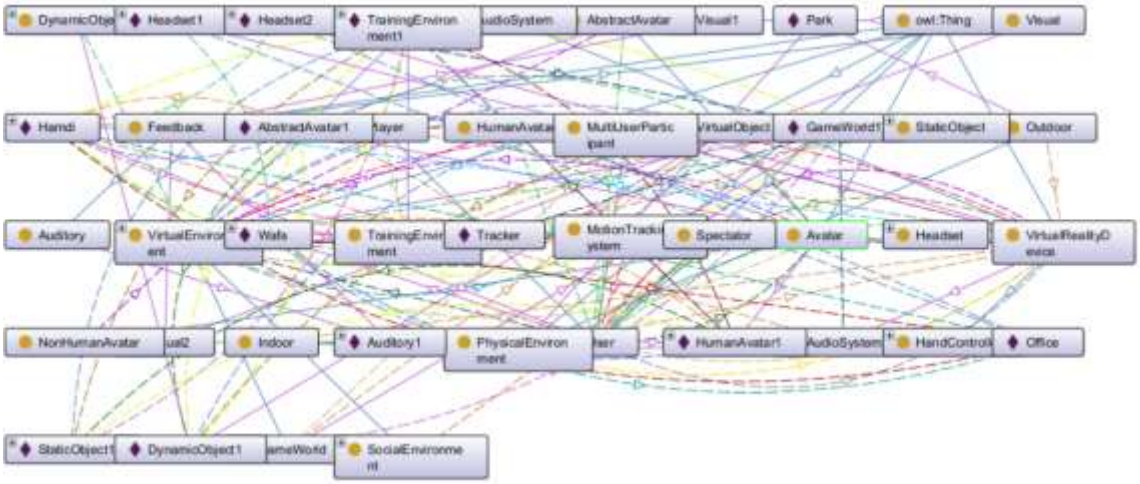
user	DeviceCount
Hamdi	"2"^^<http://www.w3.org/2001/XMLSchema#integer>
Wafa	"1"^^<http://www.w3.org/2001/XMLSchema#integer>

29 SPARQL Query 3

5. Graph By OntoGraph:

OntoGraph is a visualization tool that works alongside the Protégé ontology editor. It provides interactive browsing of OWL ontologies, allowing users to investigate connections such as subclass, individual, domain/range object attributes, and equivalence. OntoGraph provides a variety of layout choices that automatically arrange the ontology's structure, making complicated links easier to grasp.

(!) Warning: This Graph may make you feel confused or feel the need to rearrange your life 😊 .



30 OWL Graph By OntoGraph

6. Conclusion:

This VirtualReality Ontology project marks a significant step forward in the structured representation and administration of virtual environments. We used Semantic Web technologies to construct a decidable ontology that properly describes the intricate interactions between people, devices, avatars, and virtual items. This ontology not only improves our knowledge of virtual reality systems, but also provides a solid foundation for future advances and applications.

To ensure data integrity and consistency, we set explicit and precise SWRL rules as well as SHACL limitations throughout the project. These rules and constraints contribute to the ontology's logical structure by avoiding inconsistencies and maintaining the validity of all interactions and links. The use of SPARQL queries has further demonstrated the power and flexibility of our ontology.

To summarize, this project helped creating a structured existing of a Complex Concept, and As **John F. Sowa** said, "*An ontology is a catalog of the types of things that are assumed to exist in a domain of interest from the perspective of a person who uses a language to talk about the domain*[3]." As we continue to research and create in this field, ontology will surely play an important role in molding the future of all Concepts.

[3] (https://www.jfsowa.com/talks/ontology.htm n.d.)