MindSpaces

Art-driven adaptive outdoors and indoors design

H2020- 825079

Dissemination level:	Public
Contractual date of delivery:	Month 19, 31/07/2020
Actual date of delivery:	Month 19, 31/07/2020
Workpackage:	WP5 - MindSpaces adaptive environment development
Tasks:	T5.3 Semantic representation and data integration T5.4 Semantic reasoning for emotion-based space adaptation T5.6 Development of semantically enhanced interactive 3D spaces
Type:	Report
Approval Status:	Approved
Version:	3.0
Number of pages:	80
Filename:	d5.3_mindspaces_Semantic_and_emotion_reasoning_tools_for_3DSpaces_adaptation_v1_20200731_v3.0.doc

Abstract

The present deliverable elaborates on the semantic representation framework for capturing the explicit relations of the relevant MindSpaces concepts as ontology structures. Moreover, it elaborates further the services and functions of the semantic integration and reasoning methods for the first version of MindSpaces. Apart from the essential technological background and related work being presented thoroughly, the modeling and reasoning requirements, both user and technical and already defined extensively in WP6 and WP7, are also described. Furthermore, the current version of the ontology-based constructs of MindSpaces are analysed along with pertinent pre-existing efforts in both the form of literature review and state-of-the-art technology adoption and adaption regarding formal knowledge representation and pre-defined ontologies. In addition, the WP5 emotion-based reasoning framework towards space adaptations is described in conjunction with the

semantically enhanced interactive 3D spaces. Finally, the document concludes with a simulation example.
The information in this document reflects only the author's views and the European Community is not liable for any use that may be made of the information contained therein. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

co-funded by the European Union

History

Version	Date	Reason	Revised by
0.0	20/5/2020	ToC	CERTH
0.1	12/6/2020	Requirements analysis	CERTH
0.11	19/6/2020	ORSD	CERTH
0.4	25/6/2020	State-of-the-art analysis	CERTH
0.6	02/07/2020	Initial ontologies and reasoning	CERTH
0.9	08/07/2020	Simulation example	CERTH
1.0	10/07/2020	Integration of T5.6 input form U2M and first draft circulated to WP5 for commentary	CERTH
2.01	17/07/2020	Second draft dispatched for internal review	CERTH
2.9	28/07/2020	Internal review comments	MU
3.0	30/7/2020	Finalization of the final draft	CERTH

Author list

Organization	Name	Contact Information
CERTH	Stefanos Vrochidis	stefanos@iti.gr
CERTH	Petros Alvanitopoulos	palvanitopoulos@iti.gr
CERTH	Sotiris Diplaris	diplaris@iti.gr
CERTH	Nefeli Georgakopoulou	nefeli.valeria@iti.gr
up2metric	Christos Stentoumis	christos@up2metric.com
up2metric	Ilias Kalisperakis	ilias@up2metric.com
up2metric	Pavlos Bakagiannis	pavlos.bakagiannis@up2metric.com
up2metric	Katerina Adam	katerina.adam@up2metric.com
up2metric	Vassilis Tsironis	tsironisbi@up2metric.com
up2metric	Lazaros Grammatikopoulos	lazaros@up2metric.com

Executive Summary

This deliverable summarizes the efforts conducted within the scope of T5.3, T5.4 and T5.6 tasks pertinent to the design and materialization of the MindSpaces Ontologies and vocabularies, the formal representation and correspondence of incoming data to ontological structures as an integration procedure, the initial implementation of the reasoning framework towards emotion-based space adaptations and the development of semantically enhanced interactive 3D spaces.

In more detail, this deliverable presents thoroughly the **current version** of the MindSpaces ontologies and the techniques which were followed to accomplish them. Based on both user and extracted technical requirements analysis, related to T5.3 and T5.4, an ontology requirement specification document was formed to identify dependencies, goals, users, competency questions and rest details. Sequently, a state-of-the-art analysis combined with the aforementioned resulted in the first version of the MindSpaces ontology describing concepts and content such as physiological, virtual, behavioral, textual, archival and so on. Upon integration of such information the MindSpaces knowledge graph is initially constructed and ever expanding either by novel data flowing from other components or by interlinking metadata for derived assets.

Furthermore, the initial version of the reasoning framework is presented, whose purpose is to boost, enhance and enrich the supported semantics and metadata by both defining automatically additional property and class axioms and by encompassing intricate custom inferences rules in the form of SPARQL queries or javascript manipulation scripts. A conceptualization of a rule authoring tool is also presented.

In addition, processes were developed using the Grasshopper tool inside Rhino Software to alter the geometry of the 3D reconstructed models of reality. In parallel, workflows were implemented, in order to generate 3D CAD models from the mesh models. Towards the automation of this process algorithms for semantic segmentation of pointclouds are implemented and presented.

Concluding, the efforts presented in this deliverable constitute the preliminary version, towards version 1, of MindSpaces semantic representation, data integration, semantic reasoning towards space adaptation and the space adaptation techniques and artefacts themselves. More advanced versions, techniques and algorithms will be addressed in future and upcoming deliverables.

Abbreviations and Acronyms

VR	Virtual Reality
WP	Work Package
EEG	Electroencephalogram
OWL	Web Ontology Language
API	Application Programming Interface
RDF	Resource Description Framework
3D	3 Dimensional
DL	Description Logic
W3C	World Wide Web Consortium
SWRL	Semantic Web Rule Language
SPARQL	SPARQL Protocol and RDF Query Language
SPIN	SPARQL Inferencing Notation
BCI-O	Brain Computer Interaction Ontology
PROV-O	Provenance Ontology
DCMI	Dublin Core Metadata Initiative
SSN	Semantic Sensor Network
ORSD	Ontology Requirements Specification Document
URL	Uniform Resource Location
PUC	Pilot Use Case
CCTV	Closed Circuit Television
ID	Identification
gRPC	gRPC Remote Procedure Call

Table of Tables

Table of Figures

Table of Contents

1 INTRODUCTION

Within various scopes in WP5, one essential target of affiliated tasks is the delivery of a customised framework for knowledge modeling and fusion through integrating heterogeneous multimodal information relevant to the MindSpaces domain (T5.3), custom logic procedures and algorithms based on emotions deriving from multiple sources in order to achieve pertinent adaptations of space (T5.4) where the generated 3D spaces follow a semantically augmented and interactive approach (T5.6). In particular to this document nonetheless, WP5 provides, amongst other technical solutions, the knowledge formations and hierarchies in pair with respective specialised vocabularies, known as ontologies, so as to suitably capture and describe the overall semantics of:

- A project, per artist and per use case, along with details and metadata for appropriate retrieval, archiving and management purposes.
- A human subject which experiences the virtual reality environments.
- A user of the design tool referred to as “an artist” or an “architect”.
- The emotional states of EEG subjects along with lower level details regarding the EEG raw signals.
- The color palettes of the aesthetics extraction module along with pertinent emotions and various useful metadata.
- The 3D objects and their types which will be utilised inside the virtual environment.
- The types of parameters and their range values regarding the system supported categories of suggested changes inside the virtual environment.
- The concept of a hotspot regarding stress points both inside virtual environments and real environment as captured from depth cameras.
- The results deriving from textual analysis such as “aspects” and “sentiments” and how they are perceived by the system.
- The results of the behavioral analysis component such as action classification tags, timestamps of observations and so on.
- Localization of every entity when applicable abiding by a global reference system both in virtual and physical environments.

The direct and indirect workflow reliance among WP5 and other WPs of MindSpaces is presented in Figure 1, where WP3 is entitled to sensor data collection, in turn forwarded to the WP4 analysis module. Figure 1 also includes the WP6 and WP7 which correlate with WP5 in the sense of system platform integration and user feedback respectively, following the perpetual evaluation and evolutionary requirements.

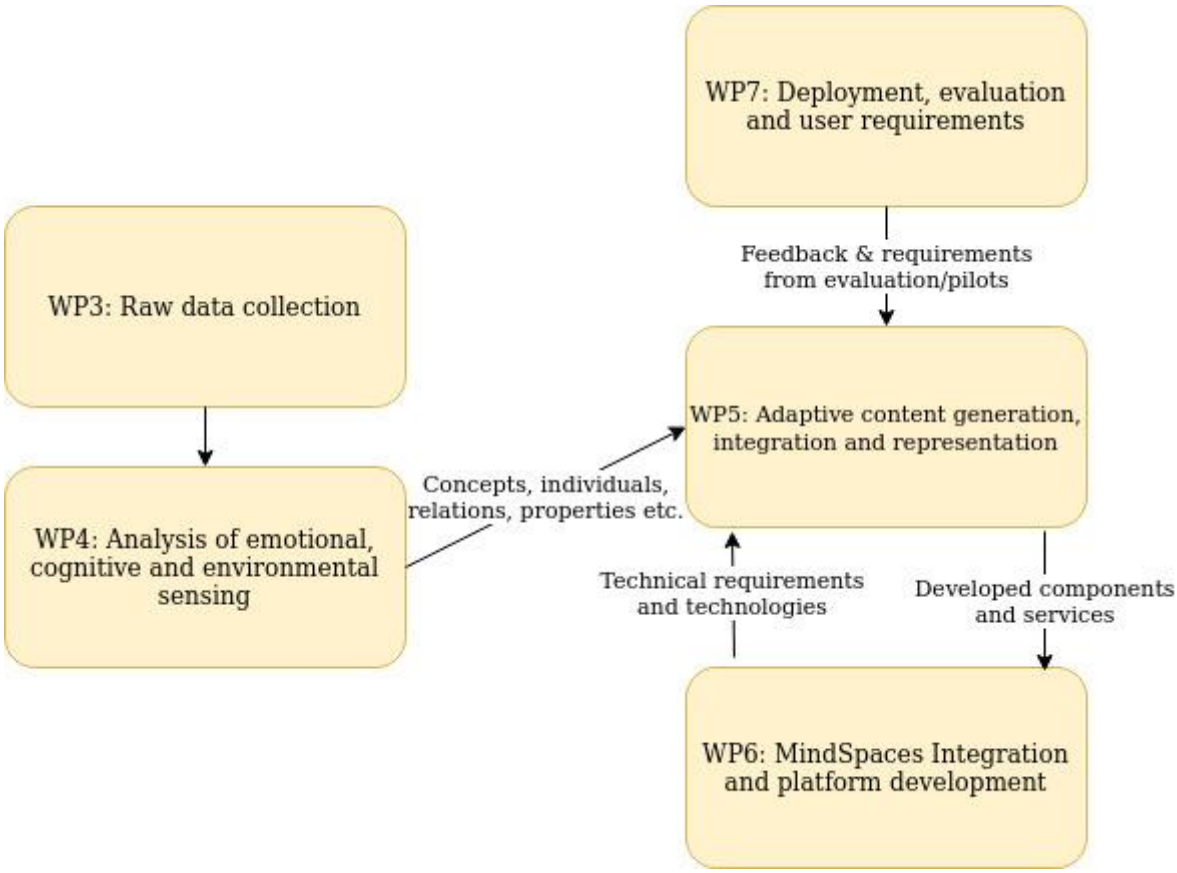


Figure 1: Logical flow of WP5-related WPs

One main goal of T5.3 is to import, reuse and potentially extend existing ontological frameworks, fusing relevant divergent vocabularies or part of such vocabularies and subjoining additional targeted custom-made concepts into one unified model that addresses the knowledge representation and management requirements of MindSpaces. The preferable semantic language utilised to define the aforementioned vocabularies is OWL 2 [1] (Web Ontology Language), the W3C recommendation for defining and sharing ontologies.

Apart from modeling the essential desired knowledge which fuels the semantics of the project, it deemed vital to populate the knowledge base (KB) in compliance with the customised ontological structures already defined and constantly iteratively updated. This is achieved automatically by mapping the multimodal information spawning by other modules of the platform in the form of analysis results; the semantic component does not occupy with raw data at all due to insufficient opportunities of meaningful semantic exploitation. As a result, WP5 implements both the automated algorithms and the Application Programming Interfaces (APIs) for the knowledge fusion of varying incoming information, thus building and expanding the initial RDF-based knowledge graph that describes the project. For instance, the emotional state tags extracted from the EEG algorithms are interlinked in WP5 with relative localization coordinates to enrich further the graph with additional individuals and relationships also.

WP5 tasks also include the reasoning tool, a framework responsible for satisfying all WP5-related, both user and technical, requirements regarding the reasoning procedures. For instance, classes, class hierarchies along with intricate descriptions and axiomatised properties, are valid examples of general axiomatizations relied on ontological structures and concepts. In addition to modeling the coherent knowledge, rules are applied on top of it, to infer and deduct further new knowledge and suggest according actions. The rules are either of automatic pre-made nature, imported directly from state-of-the-art technology stacks or custom-made, from scratch, to address specialised requirements. The overall methods, combining all the aforementioned entities, have the goal to discover underlying implicit knowledge from well-established knowledge in order to be infused into the knowledge graph and enhance it by further expanding it. Such a semantically dense and well-defined graph will enable a more semantically enriched retrieval approach of items beyond conventional methods, such as mere keyword matching, boosting the overall comprehension of concepts.

Figure 2 depicts the architectonics per relevant to this document task, part of WP5:

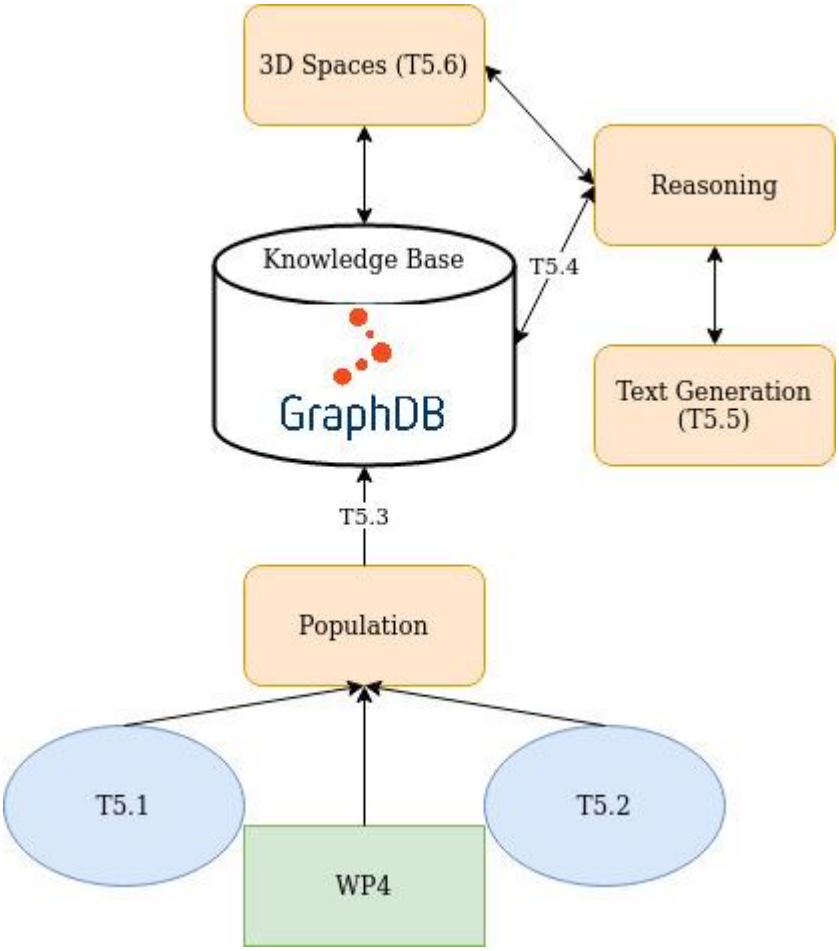


Figure 2: Diagram of affiliated tasks in WP5

The rest of the document is structured in the following sections: Section 2 provides the indispensable background knowledge along with related work. It also encapsulates the modeling and reasoning requirements that led to the MindSpaces ontology. Additionally, it refers to the population of the knowledge base and the

reasoning framework for emotion-based space adaptation. Section 3 describes the semantically enhanced interactive 3D spaces. Section 4 provides a simulation example showcasing an end-to-end example of the semantic pipeline, whereas Section 5 discusses the results, shows future perspectives and finalizes the document.

2 SEMANTIC REPRESENTATION & REASONING

This chapter encompasses an extensive literature review regarding the background knowledge and the state-of-the-art related work with respect to knowledge representation languages and frameworks, reasoning procedures and how multimodal heterogeneous data are integrated in the system, official existing vocabularies and ontologies relevant to the MindSpaces project. Moreover, the relevant modelling and reasoning requirements, along with competency questions, are presented which concluded in the current version of the ontology of the project which in turn enables the knowledge fusion upon the population of the knowledge base with automated techniques. Finally, the logic behind semantic reasoning is described thoroughly on how it enables emotion-based adaptation of 3D spaces.

2.1 Background & Related Work

Initially, it is only deemed appropriate to describe the fundamentals of Description Logic (DL) Languages [2] which are the basis for the official Word Wide Web Consortium (W3C) recommendation about the generation and distribution of ontologies throughout the Web. This W3C agreement back in 2004 established the foundations for the Web Ontology Language (OWL) [3] with an updated recommendation following in 2009 for a majorly updated version known as OWL 2 [4]. Beyond the standardization of OWL 2, there is a plethora of different OWL 2 genres along with rule-based languages.

Apart from concretely defining and hierarchizing the domain knowledge and relations pertinent to MindSpaces, reasoning services deriving from the DLs expressiveness capabilities are explicitly presented to demonstrate the potential of custom smart algorithms trying to infer further knowledge from existing knowledge and perform actions automatically when certain rules are triggered.

A challenging task, in pair with the design and implementation of ontological models and reasoning algorithms, is to investigate automatic methods to retrieve heterogeneous multimodal information efficiently and sufficiently semantify such pieces of information and populate them into a knowledge base with the utilization of an annotated unified data schema. The data integration process demands for semantically meaningful information instead of just raw data and is achieved by designing carefully the ontology mapping procedures. Consequently, commencing from harvesting raw data, the pipeline concludes into the generation of Resource Description Framework (RDF) [5] triplets, ever endingly forming in this way the expansive knowledge graph of the project. Many triplets, by explicitly following the subject, predicate, object principles, may implicitly interconnect with each other, thus densifying the knowledge graph. In such kinds of evolutionary graphs, nodes represent entities, edge labels represent types of relations and edges themselves correspond to existence of relationships among nodes.

Finally, the subchapter 2.1 concludes with a subsection referring to existent vocabularies, semantic formalities and reasoning frameworks from related works that are relevant to the MindSpaces semantic component and are either entirely or partially imported and extended to support and satisfy the needs of the project

without reinventing the wheel and following the desired principles of the reusability of open data and linked data.

2.1.1 Web Ontology Language

By investigating the related work of ontologies, it can be declared that their widespread utilization, spanning in multiple fields of research, derives from their efficacy in structuring various domain knowledges encompassing diversified nature within. An accurate modeling of each is capable of officially depicting complex concepts and fuse any modality of unstructured information, abstract context or interrelationships among entities in a manually organised manner. Furthermore, a significant advantage is their interusability by fusing, further developing, entirely importing or conglomerating segments of existent ontological models into novel semantic structures to address targeted needs.

An ontology is a solemn description of knowledge as a set of concepts within a domain and the relationships which exist between them [6]. For such descriptions to be formed, it is vital to formally establish components such as individuals, attributes, classes and relations as well as rules, axioms and constraints. Consequently, ontologies do not solely facilitate a shareable and reusable knowledge representation but can moreover add new knowledge about the domain by propagating well established facts.

The application of an ontology data model to a set of such individual facts results in the generation of a knowledge graph, a collection of entities where nodes and edges between nodes express the types and relationships between entities. By formally characterizing the structure of the knowledge in a domain, the ontology establishes the foundations for the knowledge graph to capture the data in it.

The competition about formal specifications for knowledge representation provides also other methods such as, vocabularies, thesauri, topic maps, taxonomies and logical models. Nevertheless, ontologies, for instance, and in contrast with taxonomies or relational database schemas, express relationships and enable the linking of multiple concepts to other concepts in a plethora of manners.

The shareability and reusability of the aforementioned approaches are the desired outcome of the Semantic Web in an effort to make internet data machine-readable. In the rest subchapter 2.1.1, the elementary principles of DL languages are showcased along with different categories of OWL and coherent rule-based languages.

OWL & OWL 2

The OWL is a language utilised mainly for knowledge representation purposes to address the Semantic Web needs regarding the authoring, design and generation of ontologies. The foundational elements of OWL have been strongly affected by the Description Logics. Currently, there are three different versions of varying scopes and levels of expressiveness: the OWL DL, the OWL Lite and the OWL Full. The OWL Lite initially was formed to satisfy the need of classification hierarchies and clearly expressed constraints. For instance, it only allows cardinality values ranging discretely from zero to one. Original expectations were that it would enable a quick migration path for thesauri and taxonomies systems. In contrast with OWL Lite, OWL

DL was designed with intentions to support maximum available expressiveness while preserving computational completeness, decidability and the availability of practical reasoning algorithms. OWL DL encompasses all OWL language constructs but withholds several constraints on them. OWL Full relies on totally different semantic schemas than the aforementioned frameworks and was designed to maintain as much compatibility as possible with RDF Schema. For instance, in this language a class can be regarded at the same time as a set of instantiated individuals and as an individual in its own right. In addition, ontological augmentations are also viable by expanding the vocabulary. A disadvantage of the latest relies in its undecidability, resulting in the incompetence of a reasoning framework to perform complete reasoning on it. Another one is that the OWL modeling follows a tree-like approach, thus disabling a more expressive way to support real world problems and applications. Those kinds of drawbacks and more is what set in motion the World Wide Web working group to assemble and produce and introduce OWL 2, bringing forward qualified cardinality constraints, property chains with regularity restrictions to avoid logical cycles and so on.

Regarding the different profiles of OWL 2, one can elaborate on three distinct sublanguages of the main language.

- OWL 2 QL: was formed to facilitate access and querying inside knowledge bases serving as data storage,
- OWL 2 EL: maintaining a polynomial time reasoning complexity,
- OWL 2 RL: a rule subset of OWL 2.

Finally, it is worth mentioning there are different kind of syntaxes, addressing different needs which can be summarised into two categories: The class including all high level syntaxes (OWL abstract syntax, OWL 2 functional syntax) targeting specification, and the class including all exchange syntaxes (RDF syntax, OWL 2 XML Syntax, Manchester Syntax) for facilitating data transferring and ease of use.

2.1.2 **Reasoning & Data Integration**

Description Logics Reasoning

Description Logics (DL) is a family of knowledge representation formalisms which represent the knowledge of a domain by first defining its terminology and then utilizing it so as to specify the properties of the domain and describe its world. They were formerly known as terminological knowledge representation languages, concept languages, term subsumption languages or KL-ONE-based knowledge representation languages. They are based on formal, logic-based semantics and provide reasoning as a central service, as their distinguished feature. Their core foundational elements are concepts, roles and individuals. For instance, a concept representing a set of objects with similar nature (e.g. Dog), a role which describes the semantic connection among objects (e.g. barksAt) and an individual which is basically an instantiated concept (e.g. Fluffy) can be combined via the utilization of constructors to form more intricate concepts. In this case, the notion $\exists \mathrm{hasOwner}.\mathrm{Dog}$ pertains the knowledge that an object from the class “Dog” is related in some semantic manner via the relationship “hasOwner”. The latest apples

to all instances of "Dogs" which have owners. Behind this kind of logic, lie two types of knowledges: the terminological knowledge (TBox T) and the assertional knowledge (ABox A). The former describes through axioms how the objects of a domain are related, whereas the latest contains axioms about instantaniated individuals and how they are related with each other. For the TBox axiom $man \sqsubseteq human$, it is firmly declared that every individual which belongs to the "man" concept" belongs also to the "human" concept. For the Abox axioms Man(Dimitri) and the relationship isEmployedIn(Dimitri, CERTH) it is asserted that Dimitri is a man who is employed in CERTH. The following table 1 includes all those types of axioms concerning the Tbox and Abox.

Name	Syntax	Semantics
Concept inclusion	$C \sqsubseteq D$	$C^I \subseteq D^I$
Concept equality	$C \equiv D$	$C^I = D^I$
Role Equality	$R \equiv S$	$R^I = S^I$
Role inclusion	$R \sqsubseteq S$	$R^I \subseteq S^I$
Concept assertion	$C(\alpha)$	$\alpha^I \in C^I$
Role assertion	$R(\alpha,b)$	$(\alpha^I, b^I) \in R^I$

Table 1: Tbox & Abox axioms

Description Logics Reasoning Services

Description Logics provide several useful services of reasoning, along with some algorithms that combined can fulfill a variety of requirements, such as the Pellet [7], the Hermit [8], the SHER [9], the RacerPro [10] and the FaCT++ [11].

- Pellet: is a free open-source Java-based reasoner for OWL 2 and SWRL. It supports the full expressivity of SROIQ Description Logic, user-defined datatypes and DL-safe rules. Pellet uses a tableau-based decision procedure to provide many reasoning services (subsumption, satisfiability, classification, instance retrieval, conjunctive query answering) along with the capability to generate explanations for the inferences it computes.
- Hermit: is an OWL 2 DL reasoner, one of the few such systems that attempts to fully and correctly support the OWL 2 DL specification.
- SHER: standing for "Scalable Highly Expressive Reasoner" is a technology that provides ontology analytics over very large and expressive OWL 2 knowledge bases.
- RacerPro: is a commercial but free for research OWL reasoner and inference server.
- FaCT++: is a free (LGPL) highly optimised open-source C++-based tableaux reasoner for OWL 2 DL.

Inside a knowledge base there can be several reasoning services [12] such as subsumption (Concept A subsumes concept b if the set of instances of B is at all

times a subset of the set of instances of A), equivalence (concepts A and B are equivalent if sets of their instances are at all times equal), disjoint (Concepts from A and concepts from B are disjoint only if sets of their instances are at all times disjoint), satisfiability (Concept A is satisfiable if it can withhold instances), consistency check (a knowledge base K is considered to be consistent if all named concepts from Tbox are satisfiable and Abox does not include any false individual), instance checking and retrieval (check whether a given individual is an instance of a given concept and retrieve all those individuals), realization (for a given individual y detect most specific named concepts C_i such that the instance (y,C_i) hold for every i).

Rules

The existence of rules in reasoning is to facilitate the concluding decision making of the Description Logics. In order to succeed in this, the Web Ontology Language exchanges some degrees of expressiveness for more effectual outcomes. A characteristic example of this is the decision-making tree structure. By initiating from the root until the leaves of such a structure, the desired finalization of decisions is guaranteed at the expense of applied constraints in how the variables and the quantifiers can be utilised. Consequently, it stands infeasible to depict a class whose instantiations are connected to an anonymous individual via distinct disparate property paths. To tackle those kinds of issues the research community of the domain dedicated a lot of effort towards integration strategies of Web Ontology Language with rules.

One such effort is the establishment of the Semantic Web Rule Language [13], where rules are expounded with the first order logic semantics. This approach extends the set of OWL axioms to include Horn-like rules, enabling Horn-like rules to be combined with an OWL knowledge base, in a form of implication between an antecedent and a consequent. This approach, although being promising, had a negative impact on decidability, thus several complementary works emerged [14], [15], [16], either by applying restrictions on rules, intersecting with Description Logic Programs or introducing Description Logic Safe rules. For instance, other approaches dictate the blend of rules and ontologies relying on mappings of an ontological subset of the semantics on rule engines [17].

Another more practical approach for information storing, retrieval and updating for Resource Description Framework graphs is the SPARQL Protocol and RDF Query Language [18]. It is a declarative and expressive language, founded and recommended by the World Wide Web Consortium, which enables delineation of intricate relationships among entities. It has been proved through mathematical studies that the SPARQL algebra has equal expressivity as the relational algebra [19]. A useful feature of SPARQL that transcends the role of just it being a query language is the CONSTRUCT graph pattern. That way one can build SPARQL rules which can generate novel RDF data by connecting nodes with edges resulting into larger and more dense graphs. This kind of query has the form of a CONSTRUCT and a WHERE clause, where in the CONSTRUCT field the triple pattern in the knowledge graph is defined and superinduced only when the criteria defined in the where clause are satisfied, meaning a successful pattern matching in the knowledge graph, in a way an ASK query returns true if a pattern is existent. In addition to this, there is a

supplementary framework, known as SPARQL Inferencing Notation [20] (SPIN), which facilitates the interpretation and execution of SPARQL rules on RDF graphs. By using SPIN, one can store SPARQL queries in the form of RDF triples along with any RDF model, interlinking in a graph both resources and queries and boosting interoperability and reusability.

Data Integration

Multiple components within the project are responsible for retrieving information of varying nature in an unstructured manner such as web content, physiological signals and video cameras feeds. On top of the raw data aggregation and their existence in the MindSpaces system, additional components are catering to the demands for further analysis. Consequently, the integration of such data and the extraction, storing and manipulation of knowledge is not a trivial task, thus precise ontological modelling and conceptualization of each existent entity are required. Intuitively, there is little to none knowledge residing inside raw data where on the contrary the analysis results of raw data seems extremely useful and highly exploitable in terms of reasoning. As a result, there is no need to integrate directly raw data as the added value of such a task is minimal.

These analysed data, deriving from multiple components, are unified via custom mapping algorithms and then populate into the knowledge base, resulting in building and expanding the knowledge graph. The mapping is based on the design of the ontology and how abstract concepts and properties are related whereas upon population RDF is used for representing such things and relate specific individuals, properties and relations to the abstract classes in a semantic way. Finally, the data integration is performed so as to facilitate the reasoning procedures which are based on knowledge fusion and custom inferences derived from irrelevant, uninterpreted raw data at first glance.

2.1.3 Ontologies Related to MindSpaces

This subsection discusses prior work elaborated regarding state-of-the-art ontologies that rely within the scope of MindSpaces and can be imported and reused for modeling core concepts of the project with minor to none changes. The purpose of this subsection is not to describe thoroughly a complete list of semantic frameworks relevant to MindSpaces, but rather to demonstrate the basic notions which have been followed during the review of the related work regarding the design and definition of conceptual models.

BCI-O

The brain-computer interaction ontology [21] describes a basic metadata model set for real-world multimodal Brain-Computer Interaction (BCI) data capture events. Its formation represents a conceptual framework that BCI applications can easily extend and utilize in their development, to define core concepts that depict a relevant and interoperable metadata vocabulary. BCI-O is aligned to the Semantic Sensor Network Ontology (SSN): a domain-independent and end-to-end model for sensor and actuators applications. Therefore, its structure has been normalised to assist its utilization in pair with other ontologies or linked data resources to describe in detail any particular definitions which applications in the BCI field might need, such as time

and time series, location, mobility, units of measurement and so on. Its spec provides general alignment data modeling guidelines for core concepts, to facilitate BCI applications during their design.

Prior to BCI-O there was not any complete and standardised formal semantic formation to model the BCI metadata annotations which are sensed signals so as to classify the analysis of brain states and dynamics in various occasions. In figure 3 the integration of a sense model (context to subject, relied on the SSO ODP and aligned to SOSA/SSN) and an actuation model (subject to context, relied on the AAE ODP and aligned to SOSA and SAN/IoT-O) for BCI data capture activities is depicted.

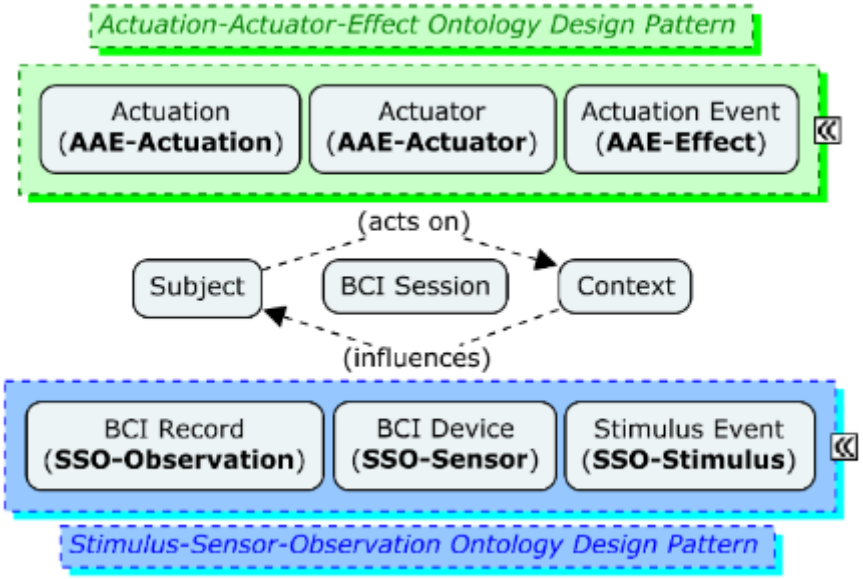


Figure 3: Core BCI Interaction Model

Event Model F

Event-Model-F [22] is a formal approach to model events and was designed to boost interoperability in distributed event-based systems. It incorporates the DOLCE+DnS Ultralite (DUL) [23] ontology and provides extensive support for representing space, objects, people, time and correlative, participation, documentation, causal, interpretation and mereological relations among events (see Figure 4 for examples and more). In addition, it provides a flexible way for event composition, modeling event causality and correlation and representing different aspects of the same event. It is developed abiding by the pattern-oriented philosophy of DUL and is modularised in disparate ontologies which can be extended by domain specific ontologies with minimal effort.

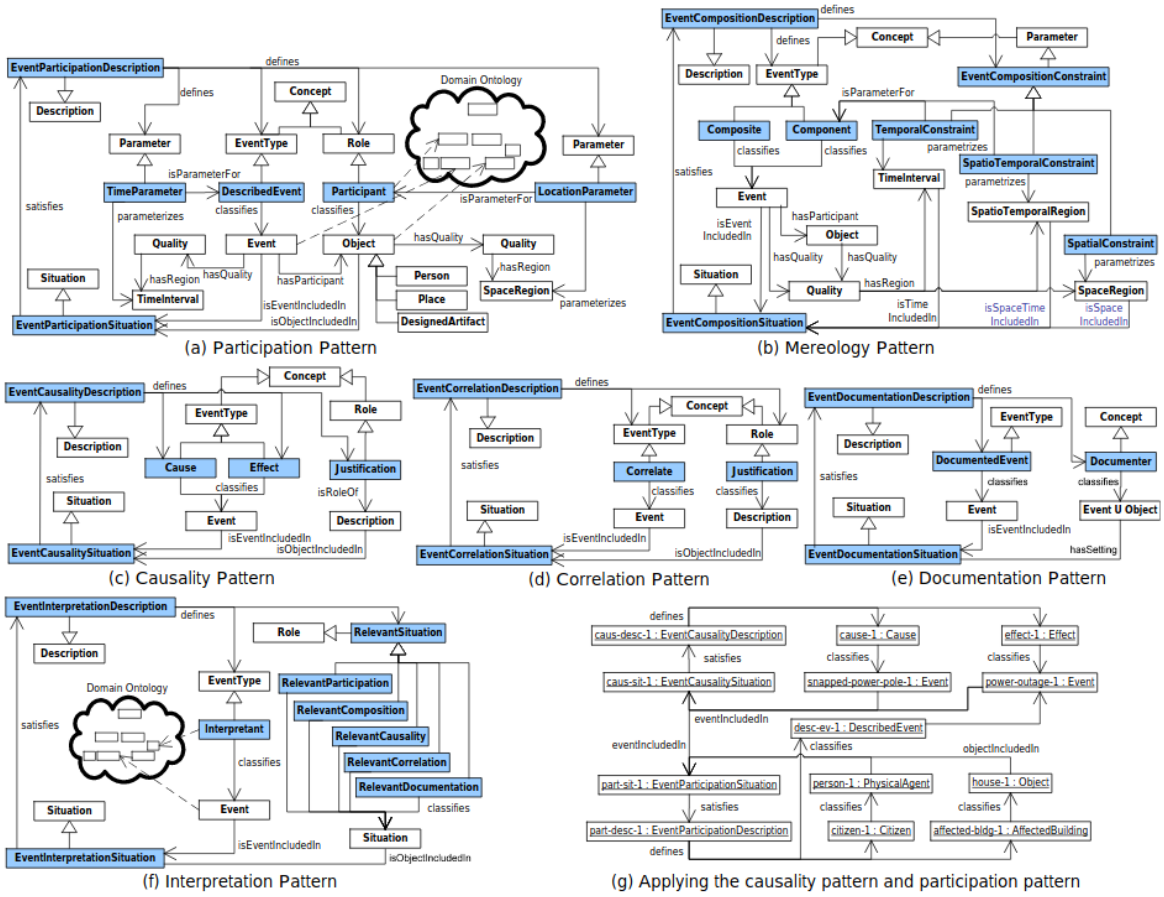


Figure 4: The patterns of F, (a) participation, (b) mereology, (c) causality, (d) correleation, (e) documentation, (f) interpretation, (g) example of applying the F ontology

PROV-O

The PROV Ontology defines the Web Ontology Language (OWL 2) encoding of the PROV Data Model [24]. It describes the entirety of classes, properties and restrictions which specify the basics to develop provenance applications in different domains, by representing, swapping and integrating provenance information created in distinct systems and under various contexts. Moreover, it can be specialised for modeling application-specific provenance details in a plethora of domains, meaning it can be utilised directly or be expanded according to demands facilitating interoperability. The core concepts of the PROV ontology are the agent which can be attributed to an entity or associated with an activity, the entity which can be created by an activity and the activity itself (check Figure 5 for more details).

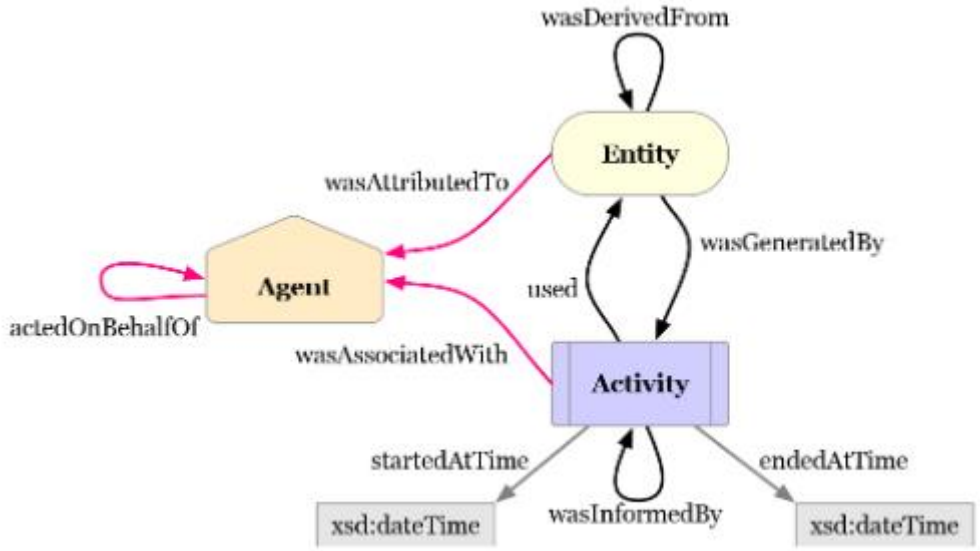


Figure 5: Core concepts of PROV-O

DCMI

The Dublin Core Metadata Initiative [25] consists of a set of 15 metadata elements (see Figure 6). Those elements are suitable for describing multimodal digital resources such as videos, images and so on, as well as physical resources. The vocabulary can be used to describe simplistic resources or even combine metadata vocabularies of different standards, thus ensuring interoperability in linked data and semantic web applications.

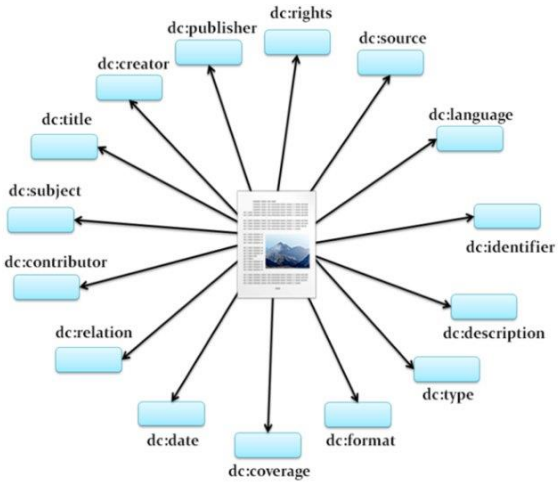


Figure 6: DCMI Core elements

2.2 Modelling & Reasoning Requirements

2.2.1 Ontology Development 101

There is a wide variety to represent knowledge about a domain and model it by utilizing ontological approaches and ontologies in general. It is a procedure which cannot be finalised at once and repetitions occur through iteratively evolving the initial approach to fit more suitably the requirements of the application. There are

various approaches and methodologies which provide guidelines regarding ontology engineering such as: United Process for Ontologies (UPON) [26], On-To-Knowledge [27], Methontology [28] and Ontology Development 101 [29].

During the scope of MindSpaces, we followed the Ontology Development 101 approach to implement the ontological framework of the project which includes 7 iterative stages [29]:

1) Decide and determine the knowledge domain and scope of the ontology (Competency questions),

2) Research the state-of-the-art and reuse existing ontologies,

3) Enumerate significant terms in the ontology,

4) Define the class hierarchy and the classes (top-down, bottom-up or a combinatory development process),

5) Define the slots (properties of classes),

6) Define the facets of the slots (cardinality, slot-value type, domain and range of a slot),

7) Create individuals.

For the first stage to be accomplished, in [30] they introduced the "Ontology Requirements Specification Document" (ORSD), which is a report answering questions regarding the necessity of the development of the new ontology, the intended uses and users of it and the requirements it should satisfy. More specifically the ORSD includes the purpose, the scope, the implementation language, the intended uses and end-users, the ontology requirements (functional and non-functional) and the Pre-Glossary of terms (from competency questions, answers and objects).

2.2.2 **Relevant User & Technical Requirements**

This subsection discusses about the relevant user requirements elaborated until this moment which guided the implementation of the MindSpaces ontological and reasoning frameworks. Thorough research regarding the each scenario has been conducted aligned with the user requirements which are presented in D7.1 "Use cases, requirements and evaluation plan" and which in turn are translated into technical requirements in D6.2 "Technical requirements and architecture". Table 2 shows the user requirements pertinent to the semantic tasks of WP5, shortly presenting the main functionalities and services that need to be addressed. In table 3, the translation between user requirements into technical requirements is presented.

UR	Detailed description	Functional or Non Functional (FR/N-FR)	Priority based on MoSCoW framework
UR_5	As an Architect I want to be able to geolocate the biometric and	Functional	M

	behavioural data in virtual space		
UR_10	As an architect I want to correlate material palettes and colours with the emotional state of users in designed workplace environments	Non Functional	M
UR_11	As an architect I want to correlate the amount / quality of light with the behaviour and/or emotional state of users in designed workplace environments	Non Functional	M
UR_12	As an architect I want to correlate the quantity and location of entry points, walls, tables, chairs, desks, and other architectural features with the behaviour and/or emotional state of users in designed workplace environments	Non Functional	M
UR_13	As an architect I want to correlate the ceiling / spatial height with the emotional state and/or behaviour of users in designed workplace environments	Non Functional	M
UR_14	As an architect I want to correlate the size/shape of a personal working desk/space with the emotional state and/or behaviour of users in designed workplace environments	Non Functional	M
UR_15	As an architect I want to correlate amenities with the emotional state and/or behaviour of users in designed workplace environments	Non Functional	M
UR_16	As an architect I want to correlate the location / type of green or exterior space with the emotional state and/or behaviour of users in designed workplace environments	Non Functional	M
UR_24	As a citizen I want to empower myself being conscious of my role of citizen shaping and giving sense to open public space	Functional	C
UR_41	As a senior I want to experience positive and empowering feelings/emotions	Non Functional	M

UR_42	As a senior I want to experience a space adapted to potential impairments (visual, hearing, mobility)	Non Functional	S
UR_44	As a senior I want to experience an aesthetical pleasant space	Non-Functional	M
UR_50	As a senior I want a space responding to my affective and intimacy needs	Non Functional	M
UR_51	As a senior I want to control the transmission of information (coming from my set and getting to my set)	Functional	S
UR_57	As a senior I want my living space to inspire/be adapted to/ physical activity practice	Non-Functional	C
UR_58	As a senior I want to my living space to be inspired/be adapted to/ physical activity practice	Non-Functional	C
UR_61	As an architect/Designer I want to have the capacity to incorporate data-driven behavioural and emotional analysis to workplace wellbeing.	Functional	M
UR_62	As an Architect/Designer I want to be able a process that can be easily pivoted to other domains where managed semi-public spaces are central.	Functional	M
UR_64	As an architect/designer I want to predict use of the new spaces by previous behavioural data	Functional	M

Table 2: Related user requirements

TRs	**Related URs**
TR 8. Create a knowledge base that will fuse multimodal data and unify them into one ontological model	UR_5, UR_10, UR_11, UR_12, UR_13, UR_14, UR_15, UR_16, UR_51, UR_61, UR_62
TR 9. Create a Reasoning Service providing the changes that are wanted to occur	UR_24, UR_41, UR_42, UR_44, UR_50, UR_57, UR_58, UR_64

Table 3: User requirements translated to technical requirements

2.2.3 **Ontology Requirements Specification Document**

This subsection presents the "Ontology Requirements Specification Document" (ORSD) which served as a foundational exercise to commence the ontological

modeling. It is an iterative process which can be further extended as the system services evolve.

MindSpaces ORSD	
1	**Purpose**
	The scope of the MindSpaces ontological framework is to describe and create the ontological formations and ontologies to represents the outcomes of the MindSpaces analysis modules in a way which follows the reusability and interoperability principles. Towards this, the representation framework will depict all the modeling aspects needed so as to support data modeling, integration and reasoning, including: Formations for representing and encapsulating metadata of varying resources, such as physiological data, behavioral analysis data, textual analysis data and so on. A well defined model and exchange format to support interoperability and reusability across different hardware and software platforms, enabling also assertions to be declared.
2	**Purpose**
	The Mindspaces ontology must formally represent: Video behavioral analysis data derived from the analysis of videos captured by the cameras. Textual analysis data derived from text analysis modules of public texts encapsulating sentiments and aspects. 3D objects along with their parameters based on which changes may be suggested to artists. Hotspots and localization of every entity present in the virtual environment. Emotional Analysis results derived from the emotional analysis module depicting physiological signals regarding what the subject is experiencing. Color Palettes data derived from the aesthetics module encapsulating also the sentiment dimension. Archiving parameters to monitor activity.
3	**Implementation Language**
	The language of implementation for the ontology of MindSpaces will be the OWL 2 [1]; it is the official recommendation by World Wide Web Consortium for knowledge representation in Semantic Web Applications.
4	**Intended End Users**
	The MindSpaces platform supports different types of end users, depending on each scenario, who will take into consideration and interact with the overall knowledge via different tools.

	PUC1: Outdoors urban environment Architects, artists and designers will reconstruct the surrounding urban environment of City De Hospitalet (add/move/remove/redesign/refurbish spatial elements) according to knowledge flowing through different modules, either in an online or offline manner, in order to emotionally affect active citizens. **PUC2: Inspiring workplaces** Architects, artists and designers will reconstruct a surrounding workplace environment (office) (add/move/remove/redesign/refurbish spatial elements) according to knowledge flowing through different modules, either in an online or offline manner, in order to emotionally inspire workers and boost productivity. **PUC3: Emotionally-sensitive functional interior design** Artists and designers will reconstruct an interior place of en elder person (home) (add/move/remove/redesign/refurbish spatial elements) according to knowledge flowing through different modules, either in an online or an offline manner, in order to make an emotionally and functionally senior-friendly environment.
5	**Ontology Requirements**
	Non-functional Requirements
	The MindSpaces ontology should follow proffered standardizations and the reusability principle by incorporating existing vocabularies (ontologies)
	Functional Requirements
	The functional requirements of the document consist of the competency questions which were extracted by analyzing each PUC scenario and relevant user requirements. CQ were also guided by the technical consortium discussions. In D1.2 “Data management and self-assessment plan V1” there is a register about all kinds of data involved in MindSpaces, where in this deliverable the WP4 data are of main concern, according to which the following competency questions are formed into groups. Competency Questions: **Visual Content:** <u>Color Palettes/Textures:</u> CQ1 Which is the unique identifier of the texture ? CQ2 Which is the map of the texture ? CQ3 Which is the material of the texture ? <u>Images:</u> CQ4 Which is the unique identifier of the image ? CQ5 Which is the title of the image?

	CQ6 Which is the description of the image ? CQ7 Which is the caption of the image ? CQ8 Which is the online source of the image ? CQ9 Who is the provisioner of the image ? CQ10 Who is the maker of the image ? CQ11 Who is the generator of the image? CQ12 Which is the Web resource the image was retrieved from ? CQ13 Which labels accompany the image ? CQ14 Which is the official license of the image ? CQ15 What is the type of file of the image ? CQ16 What is the size of the file of the image ? CQ17 What is the analysis of the image ? <u>Videos:</u> CQ18 Which is the name of the video ? CQ19 Which is the unique identifier of the video ? CQ20 Which is the online source of the video ? CQ21 Who is the generator of the video?? CQ22 Which are the labels that accompany the video ? CQ23 Which is the official license of the video ? CQ24 Which is the size/type of the file of the video ? CQ25 What is the duration of the video ? CQ26 What is the frames per second of the video ? CQ27 What are the dimensions of the video ? CQ28 What is the bits per second rate of the video ? CQ29 Which is the file with associated metadata ? <u>3D Models:</u> CQ30 Which is the unique identifier of the 3D model ? CQ31 Which is the name of the 3D model ? CQ32 Which is the description of the 3D model ? CQ33 Who is the maker of the 3D model ? CQ34 Which is the texture of the 3D model ? CQ35 Which is the location of the 3D model in VR ? CQ36 Which is the official license of the 3D model ?

	CQ37 Which is the online source of the 3D model ? CQ38 Which parameters are associated with the 3D model ? <u>Design Configurations:</u> CQ39 Which is the unique identifier of the design configuration CQ40 What kind of changes does the design configuration withholds ? CQ41 What kind of objects does the design configuration withholds ? CQ42 What kind of parameters does the design configuration withholds ? CQ43 What kind of transformations does the design configuration withholds ? CQ44 Which hotspots are associated with the design configuraiton ? CQ45 Who is the creator of the design configuration ? CQ46 For which project is the design configuration intended ? CQ47 For which PUC is the design configuration intended ? <u>Hotspots:</u> CQ48 Which is the unique identifier of the hotspot ? CQ49 Which is the localization of the hotpot ? CQ50 For which project is the hotspot intended ? CQ51 For which PUC is the hotspot intended ? CQ52 Who is the creator of the hotspot ? **Textual Content** CQ53 Which is the sentiment of the text ? CQ54 Which is the confidence of the sentiment of the text ? CQ55 Which is the aspect of the text ? CQ56 Which is the frequency of the aspect of the text ? CQ57 Which is the source of the text ? CQ58 Which is the URL of the text (to be retrieved) ? **Aesthetics** CQ59 In which image aesthetics extraction was conducted ? CQ60 Who is the artist of the artwork in the image ? CQ61 What is the style of the artwork in the image ? **Physiological Content** CQ62 Which is the arousal of the subject ? CQ63 Which is the valence of the subject ? CQ64 Which is the emotional state of the subject ?

	CQ65 Which is the location of the subject ?
	CQ66 Which is the timestamp of observation ?
	CQ67 To which subject do these signals refer to ?
	CQ68 To which PUC do these signals refer to?
	CQ69 To which project do these signals refer to?
	CQ70 Which is the URL of the saved signal ?
	CQ71 To which raw data do these signals refer to ?

The initial competency questions list spans widely to every pertinent perspective of the MindSpaces domain. This list contains more information from what is initially modeled in the ontology but offers the foundations for further extensions in the future.

2.3 **MindSpaces Ontologies**

In this subsection the first complete version of the ontology of MindSpaces is presented in detail, including the class hierarchy, the classes themselves as well as the object properties that bind the classes together and the data properties. The implementation was conducted with the utilization of the experimental platform of Protégé 5.5.0 [31], where apart from the indispensable tools for designing, a plethora of choices for validation of adequate logical coherence using pre-installed reasoners are provided as well as compatibility concerning the export file types. The development follows an iterative approach since the commencement of the tasks and until while their end, due to the model's evolutionary demands while the user and technical requirements of the project evolve in parallel.

The basic concepts that constitute the MindSpaces ontology with equivalent classes at the highest semantic level are: the Color Palette, the Emotional State, the hotspot, the location, the object, the parameter, the project, the subject, the text, the user and the video behavior. Each distinct class will be described thoroughly in the following subsection along with both the data and the object properties which interconnect classes with semantic relationships and enrich the individual's existence (see Fig. 7).

The concept clustering for further description in this document follows a per modality classification.

2.3.1 Class hierarchy & properties of MindSpaces

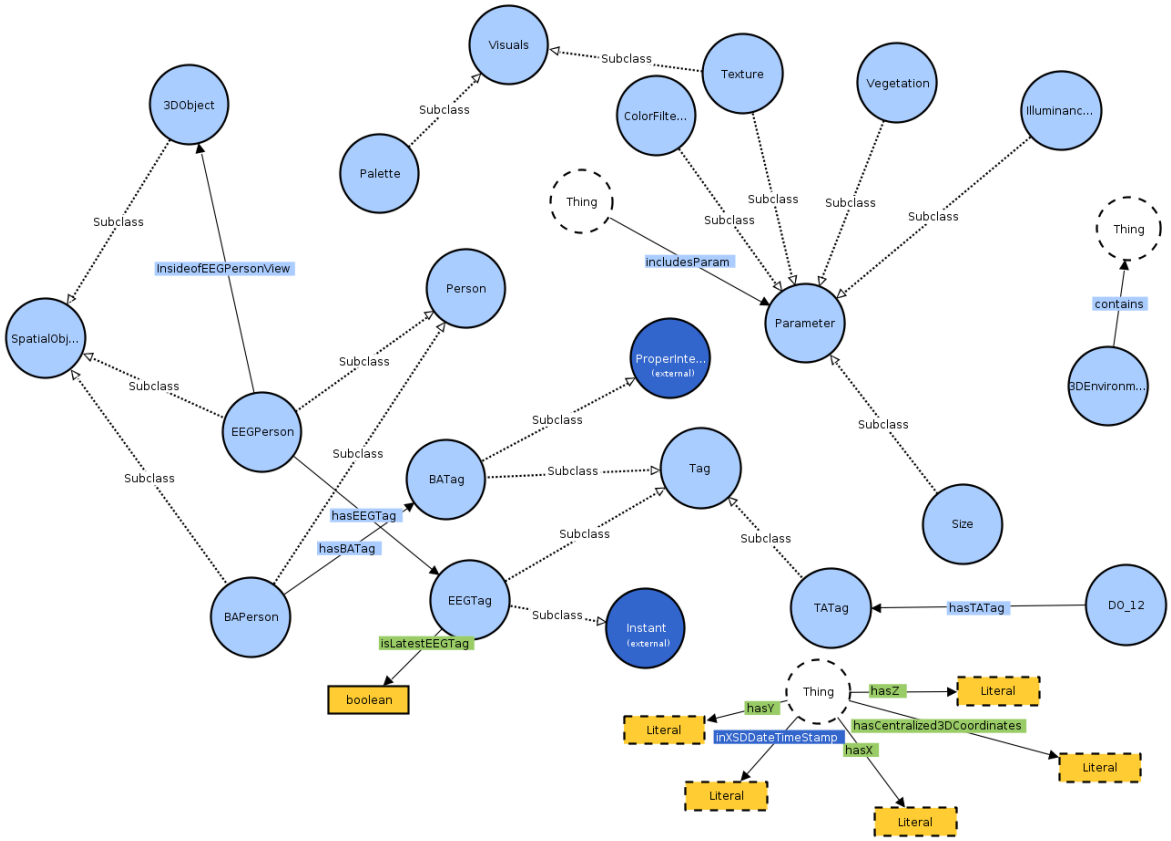


Figure 7: Higher level ontology of MindSpaces

Archiving Management

For purposes of archiving, storing, retrieval and monitoring of assets within the system and the knowledge graph, the concepts of the user, the project and the subject were materialised in the ontology of MindSpaces.

The class user conceptualizes the operator of the design tool (e.g. artist, architect, designer etc.) which is bound to a unique identifier to distinguish himself and his digital work artefacts in the design tool and the data storage from other users. However, the user interacts with the design tool and its incorporated tools, the outcomes of his utilization is encompassed archivally inside a unique project class instance. On the contrary, the subject class describes the bearer of the EEG device who is roaming inside the virtual environment and is spawning physiological signals.

The data properties, which mainly concern useful metadata, that reside inside the aforementioned classes are presented in table 4, whereas only the related, intuitively self-explainable object properties are depicted in table 5.

Domains	Data Property	Ranges
Project	ProjectDescription	xsd:string
Project	ProjectID	xsd:string
Project	ProjectLocation	xsd:string
Project	ProjectModel	xsd:string

Project	ProjectName	xsd:string
Project	ProjectType	xsd:unsignedShort
Project	ProjectUserID	xsd:string
Subject	SubjectID	xsd:string
User	UserID	xsd:string

Table 4: Data properties of archiving management classes

Domains	**Object Property**	**Ranges**
Project	assignedTo	User
VideoBehavior	belongsTo	Project
EmotionalState	hasParticipantUUID	Subject
Project	Includes	Object
Text	relatedTo	Project

Table 5: Object properties of archiving management classes

Localization

The localization of the project is mainly presented by the class location and its subclasses which correspond to coordinate “X” and “Y”, universal length and universal width. For MindSpaces at this phase the system will not handle any height properties as a third dimension. For the universal design, it was agreed that the virtual environment experienced inside the VR tool by the subject, the videos gathered during the data collection for behavioral analysis purposes and the emotional analysis component will share the same global relative coordinate system per PUC. Consequently, the same point at the physical world will correspond exactly in a one-to-one match with a point inside the virtual 3D environment. The concepts possessing physical extent being related through equivalent object properties to the class location and its subclasses are the hotspot, the video behavior and the object. In tables 6 and 7 the object properties and data properties are presented.

Domains	**Object Property**	**Ranges**
Hotspot or VideBehavior or Object	hasCoordX	CoordX
Hotspot or VideBehavior or Object	hasCoordY	CoordY
Hotspot	Inflicts	EmotionalState

Table 6: Localization related concept’s object properties

Domains	**Data Property**	**Ranges**
Hotspot	HotspothasActivity	xsd:int
Hotspot	HotspothasID	xsd:string
Hotspot	HotspothasName	xsd:string

Hotspot	HotspothasOrder	xsd:int

Table 7: Localization related concept's data properties

Textual Analysis Content

The results of textual analysis are incoming directly from a SOLR instance which indexes a mongoDB instance with all the textual data. Regarding the analysis of such entities, the resulting concept is the text class whose subclasses are the sentiment (happiness, sadness, displeasure and others for English and Spanish languages, positive/negative for the Catalan language), which comes along with a confidence estimation spanning from 0 to 100, paired with an aspect associated with a frequency number.

Video Behavioral Analysis content

The video behavior analysis component analyzes a finite set of videos of real-world establishments regarding each PUC through CCTV cameras. Per PUC the video behavior class encompasses several data properties, such as the VideoBehaviorhasPersonID where each unique Person detected inside the camera feed is assigned to a unique identifier, a VideoBehaviorTag regarding the activity detected that the person is performing in the camera feed and a VideoBehaviorStartTimeStamp which basically constitutes a unique time instant when a person is detected starting doing an activity. Additionally, as stated earlier there are spatial properties attributed to this class.

Color Palettes

The color palette class does not connect directly to any class of the ontology, meaning there is no object property present relating it to other instances. The separate subgraph it composes, consists of several data properties regarding archiving instances, providing metadata and declaring the associated emotion categories and subcategories (anger, anticipation, disgust, fear, happiness, optimism, pessimism, sadness, surprise, other). In table 8 the data properties of the color palette class are presented.

Domains	Data Property	Ranges
Color Palette	ColorPaletteArtist	xsd:string
Color Palette	ColorPaletteColors	xsd:string
Color Palette	ColorPaletteEmotion	xsd:unsignedShort
Color Palette	ColorPaletteEmotionCat	xsd:unsignedShort
Color Palette	ColorPaletteID	xsd:string

Table 8: Color palette related data properties

Physiological Analysis Content

The main class that represents the physiological analysis content is the Emotional State class. It is being related with the subject instances through unique object properties identifiers. The subject is always located at a hotspot towards version 1 of MindSpaces thus deriving its localization properties directly to the physiological signals' localization inside the VR tool. The subclasses that accompany the emotional

state class and their binary combination is what defines the state is the valence class and the arousal class, resulting in 4 distinct emotional states with different semantic meaning each (Happy, Angry, Sad, Relaxed). Additionally, a unique time instant is assigned to the emotional state class as a data property, depicting the time instant of observation of the emotional state during the experiment.

3D Virtual Environment

The 3D Virtual Environment inside the VR tool consists of 3D objects which each has several parameters assigned according to types and the flexible parametrised design results in all virtual concerns being covered. In table 9 the different types of objects along with descriptions and the associated parameters are presented.

Type	Description	Associated Parameters
Type 1 object	3D objects designed and imported by artist/architect	Freely Applied: Appear, Move, Scale, Orient, Morph, Texture, Illuminate, Animate
Type 2 object	3D objects designed by the consortium for the PUCs	Freely Applied: Appear, Move, Scale, Orient, Morph, Texture, Illuminate, Animate
Type 3 object	3D segments created by the consortium from the scans	Freely Applied: Texture, Illuminate
		Not allowed in some cases: Move, Scale, Orient, Morph, Animate
Type 4 object	Structural segments made from the scans	Freely Applied: Texture, Illuminate
		Not allowed in some cases: Animate

Table 9: Types of 3D objects

Object properties, equivalent to aforementioned parameters in table 9, bound subclasses of the class parameter with the relevant subclass of type of the object (Decorative, environmental, greenness, illumination, mobility, structural, topography, utilitarian). Finally, additional data properties are present which provide useful metadata such as unique identifier, name, description of the object, upload date and so on.

Ontology Metrics

Finally, in table 10 several general quantitative ontology metrics are presented regarding the current version of the MindSpaces ontology. Those are bound to be

changed in next iterations as the requirements evolve and need to be addressed appropriately.

Metric	Count
Axiom	178
Logical axiom count	106
Declaration axioms count	72
Class count	28
Object property count	13
Data property count	31
SubClassOf	16
Assymetric Object Property	2
Object Property Domain	13
Object Property Range	13
Data Property Domain	31
Data Property Range	31

Table 10: MindSpaces ontology metrics

2.4 KB Population

As defined in the overall architecture of MindSpaces and specifically in WP5 (see Fig. 2), one significant semantic task is the knowledge base population component. The service within this component is appropriate and capable of translating large chunks of (semi-)structured data (and metadata) in the format of JSON into Resource Description Framework triples and store them inside the appropriate GraphDB instantiated repository, thus further expanding the knowledge graph of the project.

More specifically, for each input modality (physiological analysis content, localisation, archiving management, textual analysis content, video behavioral analysis content, color palettes and 3D Virtual environment), custom mapping services are developed and implemented to receive very specific custom JSON formatted predefined structures of the incoming data as input (Fig. 8).

The mapping and storing services are implemented inside a node.js server which is an always online service. For version 1, every modality will be considered as live data incoming on demand, apart from the color palettes and the video behavioral analysis content, which constitute finite sets with no further processing inflicted after the initial dataset is formed. Inside the GraphDB instantiation, there is a provision of 2 different repositories each serving a different scope. The first repository acts as a graph-based rdf-based synchronised copy of the indexed SOLR subgraph which contains all useful analysed textual content (e.g. aspects and sentiments). The second repository will contain every other piece of information. This approach was

selected in order to be able to keep SOLR and GraphDB graphs always in sync with custom procedures and without the provision of a commercial SOLR-GraphDB connector.

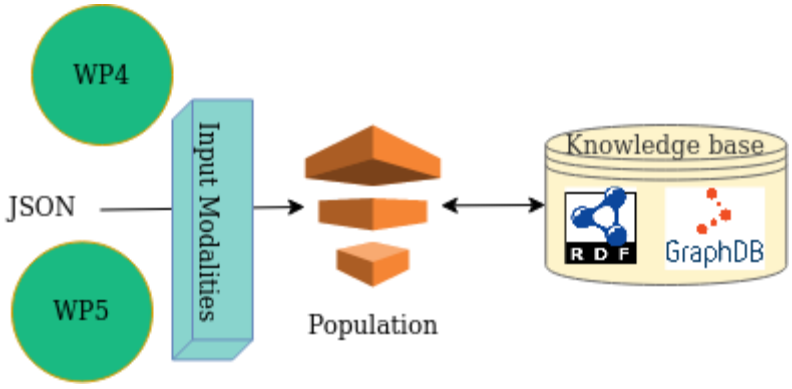


Figure 8: KB population service

2.5 Semantic Reasoning for Emotion-based Space Adaptation

Until this section, the present document mainly focalised on the firm establishment of key concepts pertinent to MindSpaces' scope and the implementation of equivalent ontologies based on relevant vocabularies to satisfy the design, the correspondence, the transformations and the representation needs of data towards semantic formations.

In this section, the initial semantic reasoning framework for emotion-based space adaptation (WP5) is presented (towards V1). Main goal is to smartly gather all useful data and metadata from other MindSpaces components by fusing their distilled knowledge and store it meaningfully inside the KB. The reasoning methodology followed in this task at this moment is the initial way to conform and suffice the user requirements, with additional features pending for subsequent milestones and prototypes. More concise and efficient reasoning development and knowledge management will be utilised in sequent iterations of the project, as MindSpaces modules ripe with continuous engagement, a fact that will enable more intricate implementations which will be included in sequent chronologically official documents.

2.5.1 Reasoning Architecture

The basic components of the reasoning framework are presented in Fig. 9 in an abstract design. It adds value on the ontologies and knowledge of MindSpaces with ruled-based reasoning applied on available data. It greatly relies on the semantics presented in previous chapters, in a way that the formal representation of concepts and context and relations among multimodal data interlink with each other efficiently, forming knowledge subgraphs. These subgraphs which usually are stored inside the KB are retrieved intelligently via exquisite SPARQL queries, fused and served as input into the reasoning algorithms. The retrieval from the KB is based on approximate parameter matching on localization and emotion (see fig. 10) and then custom logic is applied at this stage to showcase the true power of ruled-based reasoning and infer additional implicit relationships that the human eye could not perceive.

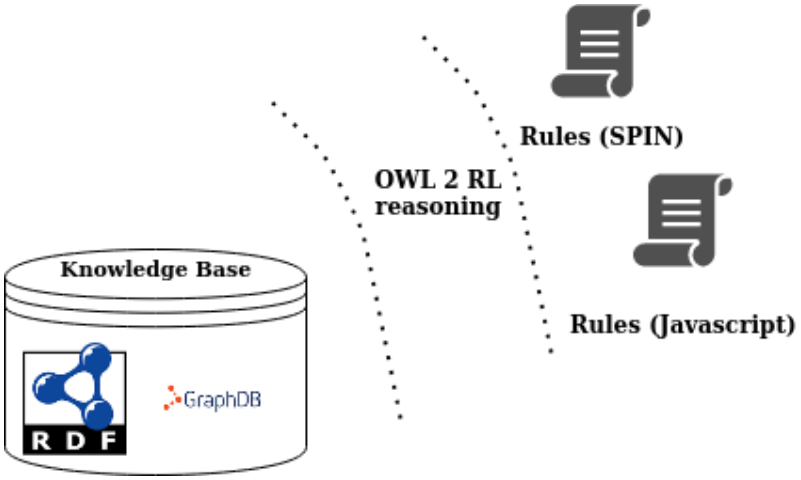


Figure 9: Abstract Reasoning Architecture

The reasoning procedures are conducted on top of the Knowledge Base where most knowledge of the MindSpaces pipelines is stored, in an always online service implemented with the node.js javascript library. For the time being, the initial selection of the triple store to serve as knowledge base is the GraphDB Free edition[1], due to its scalability towards native OWL 2 RL reasoning services [32] and interfaces that enable SPARQL queries [33]. The former guarantees that the Web Ontology Language is supported. Nevertheless, limitations of expressivity are present resulting in the inability to deal with intricate relations. For instance, the tree structure must be followed at all times when modeling properties thus anything not conforming to it cannot be modeled at all. Furthermore, the native DL semantics do not support and forbid the progressive creation of novel individuals on the fly.

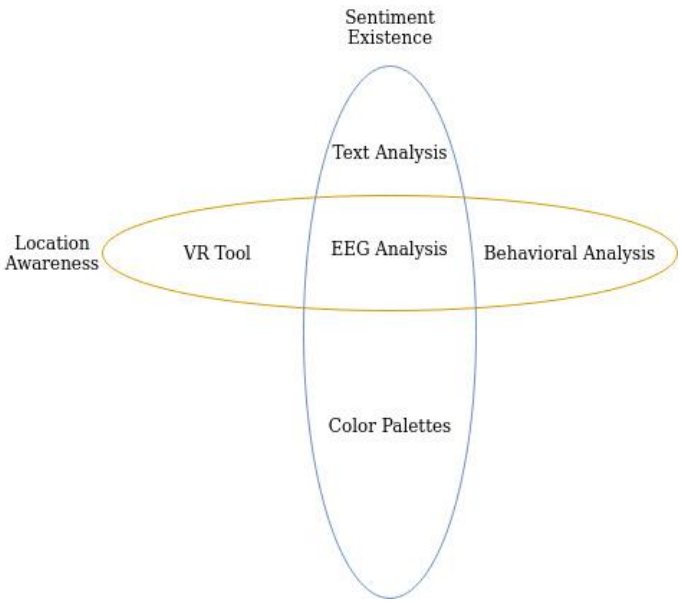


Figure 10: Venn Diagram on component-wise set theory

[1] http://graphdb.ontotext.com/

2.5.2 **Inference Rules**

A useful feature that has been preliminary exploited is the provision of SPIN rules. Basically, they constitute SPARQL queries with CONSTRUCT triples capabilities. That way the data can propagate, and individual creation can be achieved automatically when conditions are satisfied. Each SPIN rule corresponds to a particular reasoning exercise which tries to satisfy particular requirements. An example of it is shown in figure 11, where the knowledge of the 3D objects residing inside the field of vision of the EEG subject is propagated to the field of vision of hotspots of the behavioral analysis instances which share the exact same coordinates with the EEG subject.

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

CONSTRUCT
  {
    ?BASubject :insideofHotspotView ?3DObject .
  }
WHERE
  {   { ?EEGSubject  :hasEEGTag         ?EEGTag .
        ?EEGTag   :isLatestEEGTag       true .
        ?EEGSubject  :InsideofEEGPersionView  ?3DObject
      }
    UNION
      { ?BASubject  :hasX  ?BAX
        { SELECT  ?BAX
          WHERE
            { ?EEGSubject  :hasEEGTag    ?EEGTag .
              ?EEGTag   :isLatestEEGTag  true .
              ?EEGSubject  :hasX         ?BAX
            }
        }
        ?BASubject  :hasY  ?BAY
        { SELECT  ?BAY
          WHERE
            { ?EEGSubject  :hasEEGTag    ?EEGTag .
              ?EEGTag   :isLatestEEGTag  true .
              ?EEGSubject  :hasY         ?BAY
            }
        }
      }
  }
```

Figure 11: SPIN rule example

Apart from SPIN rules, custom ruled-based reasoning is implemented in a javascript service, relying on top of the knowledge base as a communication intermediate with the rest components of the project, where knowledge is intelligently fused with the retrieval of information with the appropriate SPARQL queries. When relevant data to the emotional state status and localization of the subject inside the virtual environment are retrieved (see Fig. 12), several predefined rules structured initially

in a decision tree approach (see Fig. 13), may be triggered or not accordingly and generate a desired suggestion inside a design configuration including all the pertinent parameters which need to be changed to achieve a certain goal of inflicting the desired emotional impact on the subject. Those rules are custom-made with a sole goal of showcasing the potential utilization of the rule authoring tool by users to create their own rules.

```
PREFIX  :     <http://www.semanticweb.org/MindSpaces.eu#>
PREFIX  rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX  owl:  <http://www.w3.org/2002/07/owl#>

SELECT  ?BATagF ?EEGTagF ?ParamF ?3DObjectF ?typeF
WHERE
  {   { ?EEGSubject  :hasEEGTag    ?EEGTag .
        ?EEGTag   :isLatestEEGTag  true
        BIND(strafter(str(?EEGTag), "#") AS ?EEGTagF)
      }
    UNION
      { ?BASubject  :hasX  ?BAX
        { SELECT  ?BAX
          WHERE
            { ?EEGSubject  :hasEEGTag    ?EEGTag .
              ?EEGTag   :isLatestEEGTag  true .
              ?EEGSubject  :hasX         ?BAX
            }
        }
        ?BASubject  :hasY  ?BAY
        { SELECT  ?BAY
          WHERE
            { ?EEGSubject  :hasEEGTag    ?EEGTag .
              ?EEGTag   :isLatestEEGTag  true .
              ?EEGSubject  :hasY         ?BAY
            }
        }
        ?BASubject  :hasBATag  ?BATag
        BIND(strafter(str(?BATag), "#") AS ?BATagF)
      }
    UNION
      { ?EEGSubject  :InsideofEEGPersonView  ?3DObject .
        ?3DObject  :includesParam       ?Param .
        ?Param    rdf:type              ?type
        FILTER ( ?type NOT IN (:Parameter) )
        FILTER ( ?type NOT IN (owl:NamedIndividual) )
        FILTER ( ?type NOT IN (:Visuals) )
        BIND(strafter(str(?Param), "#") AS ?ParamF)
        BIND(strafter(str(?3DObject), "#") AS ?3DObjectF)
        BIND(strafter(str(?type), "#") AS ?typeF)
      }
  }
```

Figure 12: SPARQL smart retrieval and fusion example

In Figure 12, the select clause returns the behavioral analysis tags, the latest emotional state stored, the parameters and the 3D objects. The overall logic is:

1) Get the latest emotional state of EEG analysis stored

2) And get the coordinates of the aforementioned EEG subject which are exactly the same of that of the Behavioral analysis subjects (hotspots)

3) And get all the Behavioral analysis tags from that behavioral analysis subjects

4) And get all the 3D objects which are inside of the field of vision of the EEG subject

5) And get all the parameters of those 3D objects

With the acquisition of all this relevant data one can decide at that moment how to manipulate the combinations, the matchings. The approximations and the range values in order to infer in an automatic manner which change is desired.

```
function getLatestChange(call, callback) {

 var triggered_rule = new String;
 switch (call.request['state']['eeg_tag']) {
  case 1:
   varconfiguration_id = 17
   varchange_configuration = true
   varchange_hotspot_id = 25
   triggered_rule = 'switch (call.request["state"]["eeg_tag"])@@@ case 1'
   break;
  case 2:
   varconfiguration_id = 22
   varchange_configuration = true
   varchange_hotspot_id = 26
   triggered_rule = 'switch (call.request["state"]["eeg_tag"])@@@ case 2'
   break;
  case 3:
   varconfiguration_id = 35
   varchange_configuration = true
   varchange_hotspot_id = 19
   triggered_rule = 'switch (call.request["state"]["eeg_tag"])@@@ case 3'
   break;
  case 4:
   varconfiguration_id = null
   varchange_configuration = false
   varchange_hotspot_id = null
 }
}
```

Figure 13: Custom Javascript Rule

In Figure 13, the gRPC function which is utilised in order to communicate with the VR Tool and exchange data withholds a decision tree based on a switch case javascript statement. The decisions made are entirely based on the live incoming EEG emotional states. More precisely, if the emotional state equals 1 it returns the design configuration with “ID=17” and “true” state of transformation of the proposed change along with the hotspot “ID=25” which the subject will be teleported to witness the change. Likewise, situations apply for emotional state 2 and 3. In case of emotional state equals 4 the function returns no ID at all and the false statement declaring that no change will take place at this moment. The whole transaction is live, with data exchange and execution of the algorithm happening every time there is an emotional state generated by the EEG analysis component and passed down through the entire pipeline. The “triggered rule” variable contains information about the rule that was activated and constitutes a string that will be passed down, as a part of a more complex gRPC message request, towards the Text Generation module.

2.5.3 **Rule Authoring Tool**

Towards version 1, the scenario of the reasoning scheme was to provide custom examples of the reasoning capabilities and showcase the potential logic that can be constructed in order to propose virtual changes. In order to enable the users to create their own custom rules and not rely exclusively on technical partners to translate high level demands into hardcoded rules or learn how to query through SPARQL and Javascript, the provision of a rule authoring tool was conceptualised.

The rule authoring tool will be a part of the design tool with its own exclusive interface, keeping in prospect all well-known approaches towards a better user experience and ease of use (UI/UX). The initial version of the tool will not provide the same expressivity and pluralism in rule generation as the precise custom hardcoded rules. It is a generic tool that will be connected with the pure backend reasoning framework, transmitting and translating the graphical user interactions into reasoning rules which will be triggered when occasions arise.

In more detail concerning the design, the user will be able to select from a drop down list classes of the ontology of MindSpaces and combine them along with logical and arithmetical operators and give specific values (e.g. strings or integers) or even range values to create his own custom rules that most likely will be translated into decision trees into the reasoning framework, infused inside the always online node.js service handling the overall communication of the semantics component. The rule authoring tool is planned to be fully integrated towards version 2.

3 SEMANTICALLY ENHANCED INTERACTIVE 3D SPACES

This section contains information about the solutions, workflows and algorithms used and developed in MindSpaces for the generation of semantically enhanced interactive models from the 3D reconstructions of urban and indoors spaces generated in T4.1.

One of the objectives of Mindspaces is to develop processes that will allow to transform, inside the Virtual Reality module, the 3D indoor and outdoor spaces according to the user responses. This includes the visualization of user responses on the 3D models and the modification of the 3D models' form according to user emotional input. In this context, the expected results are 3D models capable of adapting to Design Machine manipulation and users' interaction.

In the following we present, processes that were implemented using the Grasshopper tool inside Rhino Software to modify the geometry of the 3D reconstructed models of reality (from 3D scanning, image-based techniques, or the mobile mapping platform). In parallel, workflows were developed, in order to generate 3D CAD models from the mesh models. Details about the generation of 3D CAD models for each one of the three Pilot Use Cases (PUC 1, 2 & 3) are given. This is a process that requires 3D design skills and user effort to correct the models, simplify the geometry and textures and enhance the models with semantics. Towards the automation of this process algorithms for semantic segmentation of point clouds were implemented and presented.

According to DoA, the enhancement of 3D spaces is involved in RO4 (Semantic integration of emotion and environment (sensor) input) and RA4.2 (Adaptive 3D-models based on semantic reasoning) research objectives and activities of the MindSpaces project.

3.1 Modification of reconstructed 3D geometry in Rhino Grasshopper

Dedicated tools and procedures to collect data for 3D reconstruction of outdoors and indoors environments (T3.3 & T3.4 respectively) were implemented in Mindspaces. The recorded data is processed through the Outdoor and Indoor reconstruction services of the Mindspaces platform (T4.1). The output of these services is either coloured 3D point clouds, either 3D triangular mesh models draped with photo texture.

The first approach was to apply geometric modifications directly to the 3D mesh models in the environment of Rhino 3D CAD software using Grasshopper[1].

Grasshopper (Figure 14is a visual programming language and environment that runs within the Rhino 3D CAD software application. Programs are created by dragging components onto a canvas. The outputs of these components are then connected to the inputs of subsequent components. Grasshopper is used in a variety of application scenarios including parametric modelling for structural engineering, for architecture and fabrication. The feature of parametric design allows to assign descriptive parameters to the form of 3D objects and then to modify their shape by tuning these parameters. The values of each parameter can be defined through User Interface (UI) controllers such as sliders or text boxes or through external devices or programs that are linked to "listeners", which is convenient for the

[1] https://www.rhino3d.com/6/new/grasshopper

experiments with VR headsets and EEG capturing devices where the values of the descriptive parameters can be set in accordance to the users' emotional states.

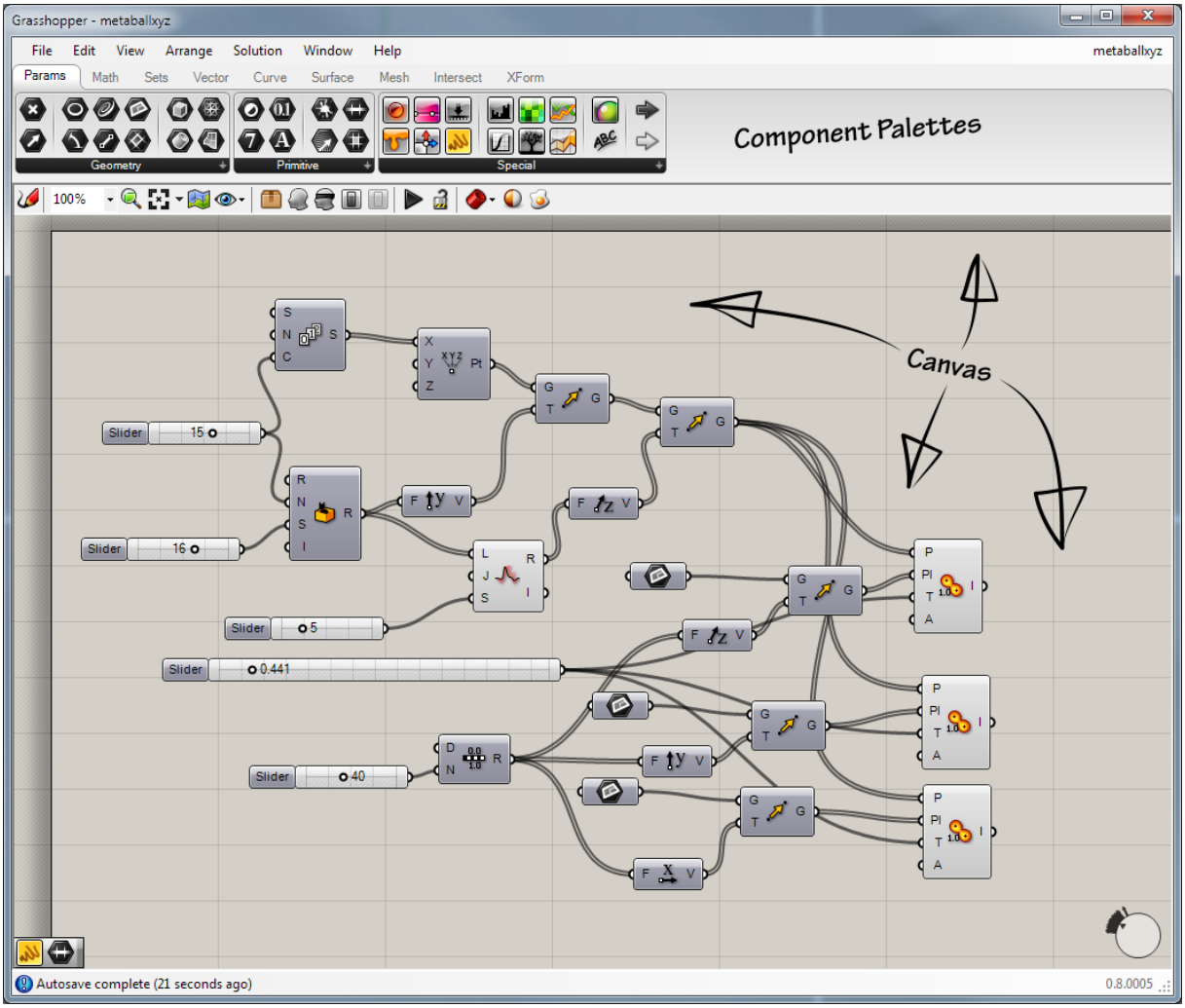


Figure 14. Grasshopper environment (source Wikipedia[2])

The expected functionality of the Grasshopper programs is the modification of certain predefined parts of the 3D model, such the position or the dimensions of a specific building or construction in an outdoors environment. The geometry of the rest of the environment should stay unchanged and the modified geometry should retain its photorealistic characteristics (materials or phototexture).

The high level workflow (Figure 15 & Figure 16of the implemented grasshopper program consists of the following steps:

1. The initial 3D mesh model is linked to grasshopper (Figure 17).

[2] https://en.wikipedia.org/wiki/Grasshopper_3D#/media/File:Grasshopper_MainWindow.png

2. The linked 3D mesh is connected to a custom Grasshopper Python Script (*GhPython Script*) that deconstructs the mesh into its component parts and more specifically into vertices, 3D faces and UV coordinates (Figure 17). The latter define a mapping between the geometry of the 3D mesh and the photorealistic texture, which is usually stored on one or multiple "texture atlas" image files.

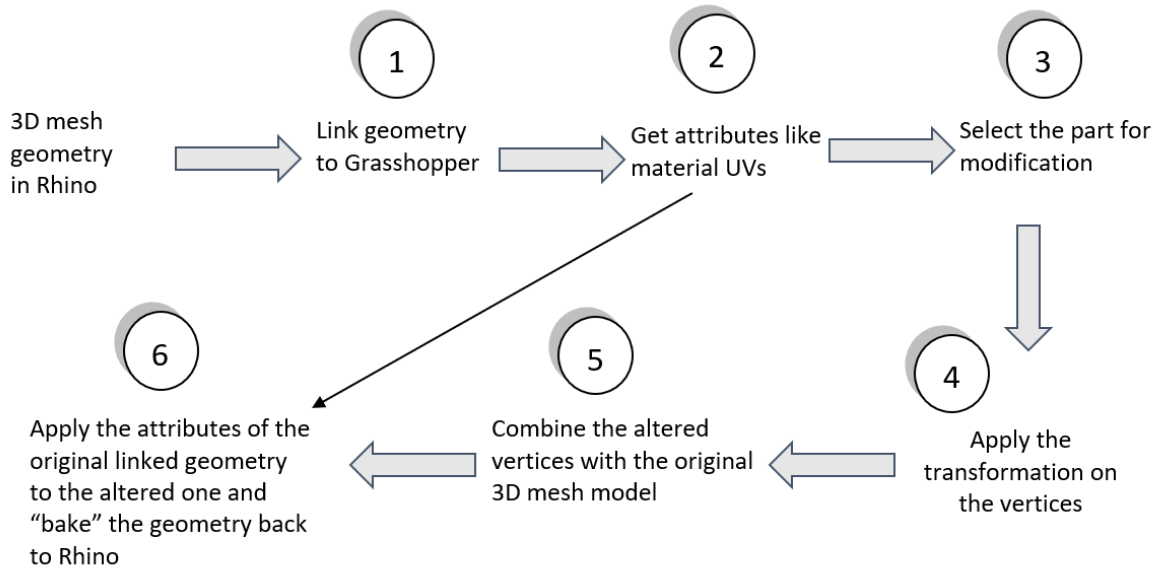


Figure 15. Workflow for modifying 3D mesh geometry via Grasshopper in Rhino

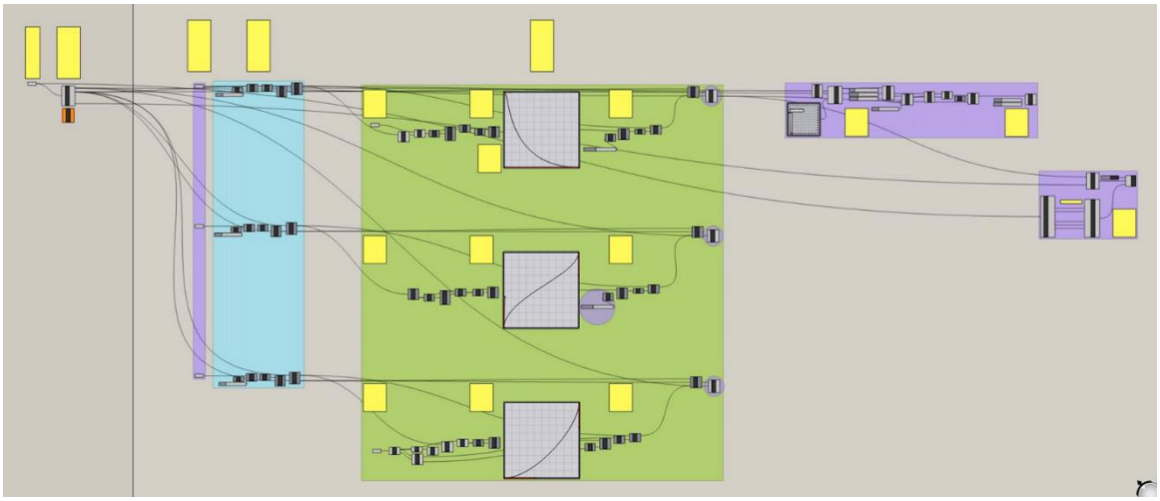


Figure 16. Overview of Grasshopper's canvas for modifying the geometry of 3D mesh models

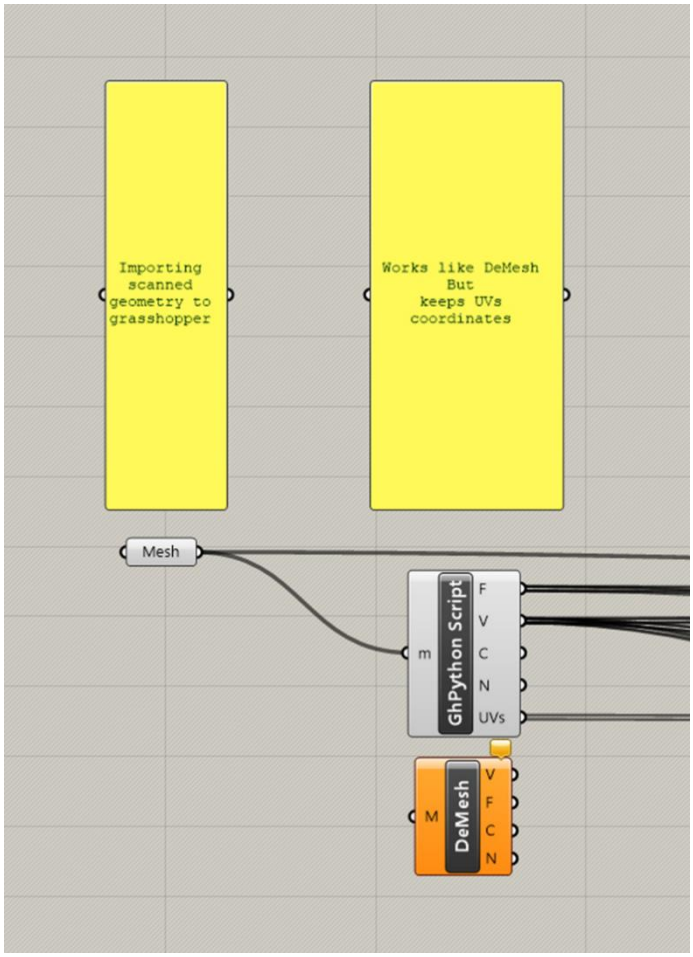


Figure 17. Grasshopper component for linking geometry and getting its' UV coordinates

3. Then the part of the original 3D model that is going to be altered is selected. For the selection of a subpart of the 3D mesh, two alternative functionalities were implemented (Figure 18):

a) The user draws a closed curve inside the environment of Rhino 3D. Then all vertices that lie inside the curve are identified and selected.
b) The user defines a point on the 3D mesh and sets a distance threshold via a slider. Then all vertices that are within the distance threshold are selected.

4. The next step applies the geometric modification on the selected vertices (Figure 19). Different types of linear and nonlinear transformations are supported, such as 3D movements on X, Y, Z directions, 3D and 2D rotations, isotropic or affine changes in scale, affine and projective transformations in 3D space etc. The mechanism for applying the different types of transformations is similar. New coordinates are estimated for each vertex according to certain algorithmic rules and parameters that are set by the user.

5. Once the geometry of the selected part is modified, the altered vertices are combined with the rest of the original 3D mesh model.

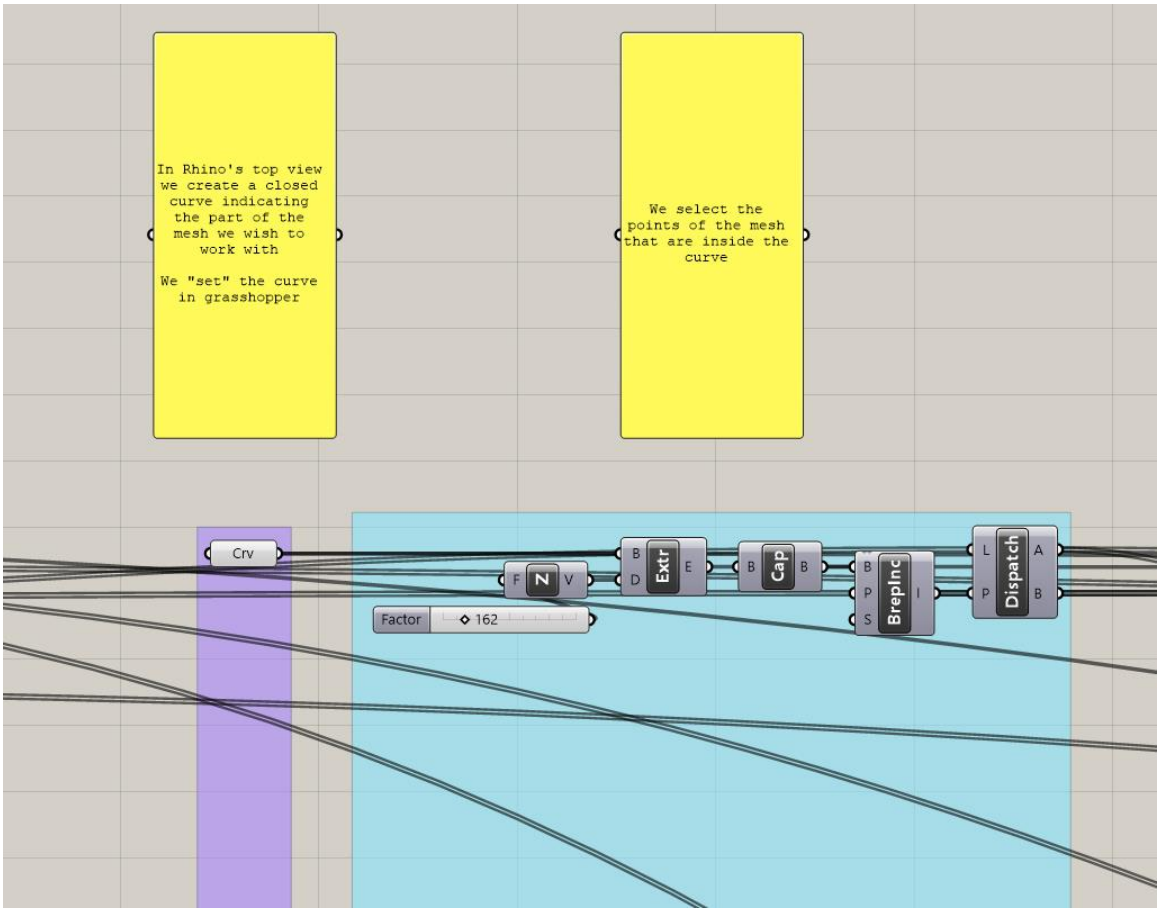


Figure 18. Grasshopper component for selecting the part for modification

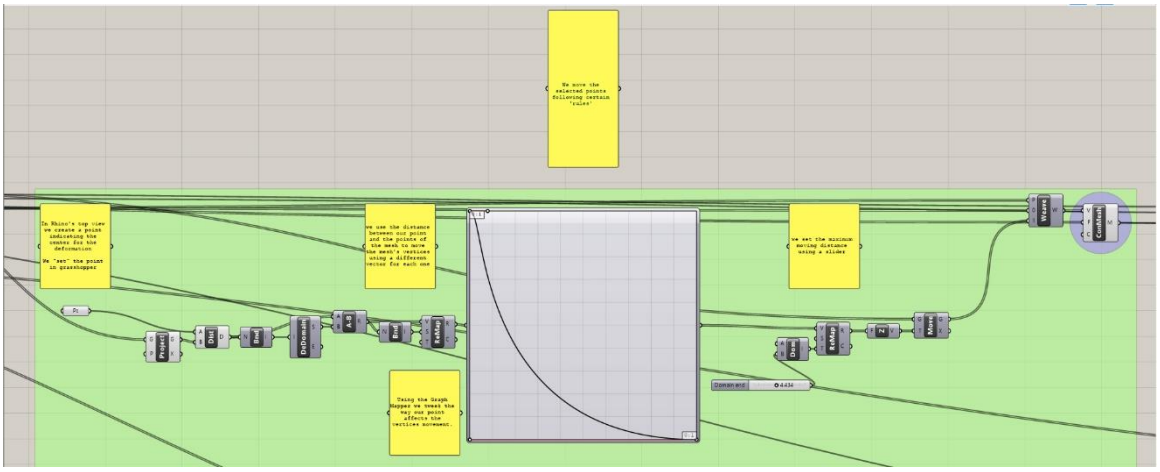


Figure 19. Grasshopper component for applying the transformation on the vertices and combining the altered vertices with the original 3D mesh model

6. In the final step, the UV coordinates of the original vertices of the 3D mesh model are assigned to the altered vertices (Figure 20). In this way the texture is transferred to the modified model and it can be rendered and visualised with photorealism, which is consistent with the rest of the 3D model.

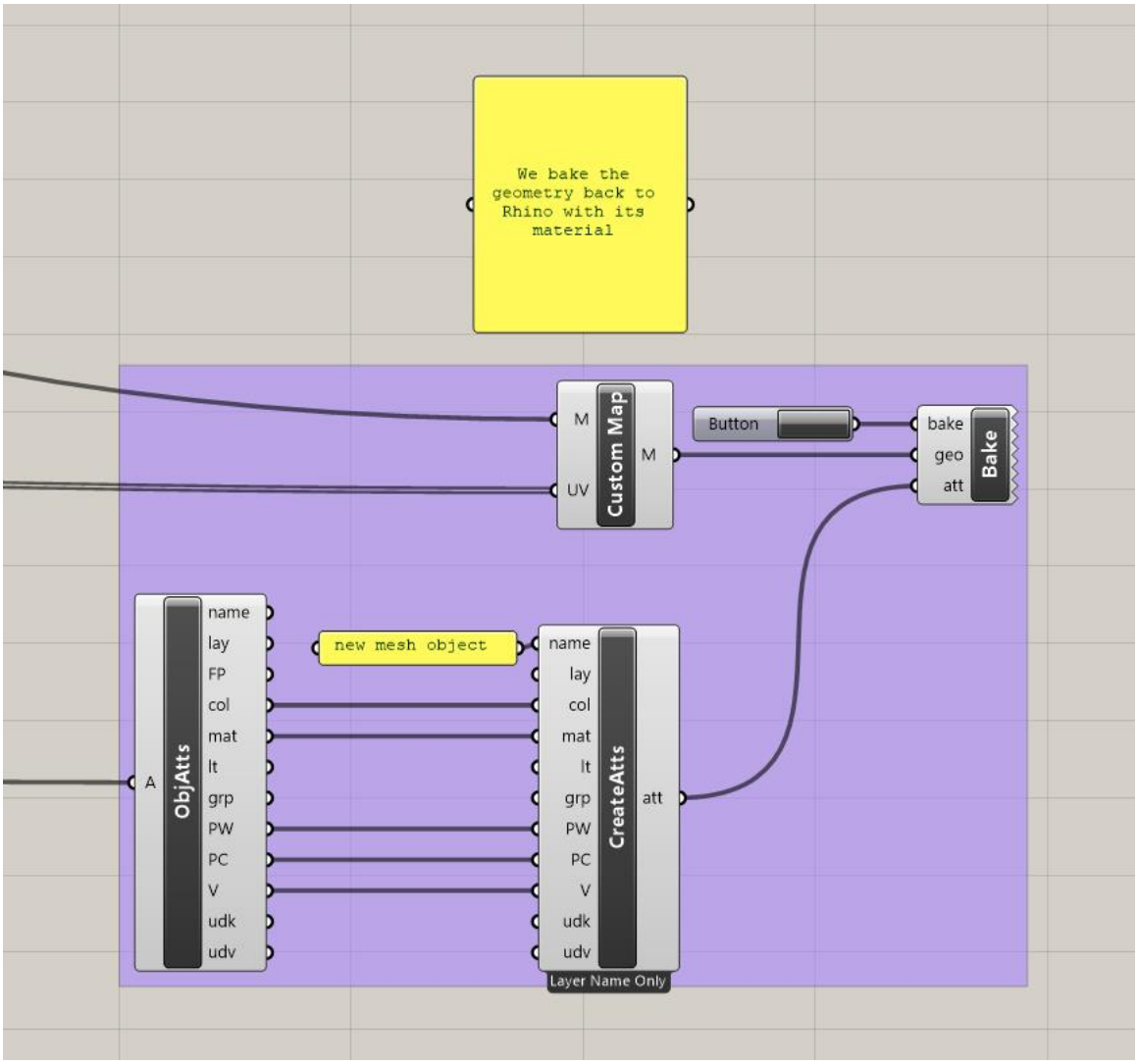


Figure 20. Grasshopper component for applying the attributes of the original geometry to the altered one

Figure 21presents an example of a change in the height of a specific building, which could correspond to an addition of a second floor. Since this transformation affects only the Z coordinates of the selected vertices, the deformation does not create overlapping triangles or other mesh problems and the result is rather smooth and convincing.

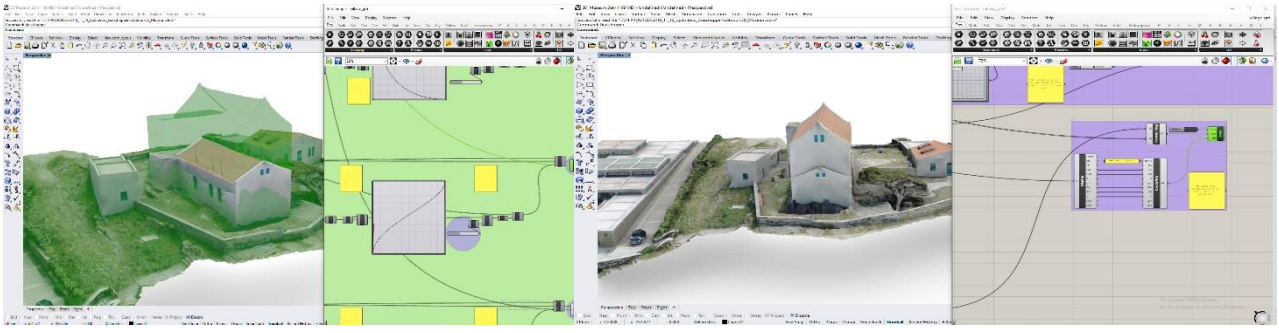


Figure 21. Example of a building geometry change (stretched in the z-axis)

Figure 22 presents a similar deformation on a different building. This time the length of the building is altered, by changing the scale across the y-axis. In this case the deformation does create overlapping triangles in the base of the building and the connection point between the altered geometry and the rest of the model can be visualised as a little off the ground or like crossing inside it, if there is significant difference in the elevation of the altered building triangles and the ground level.

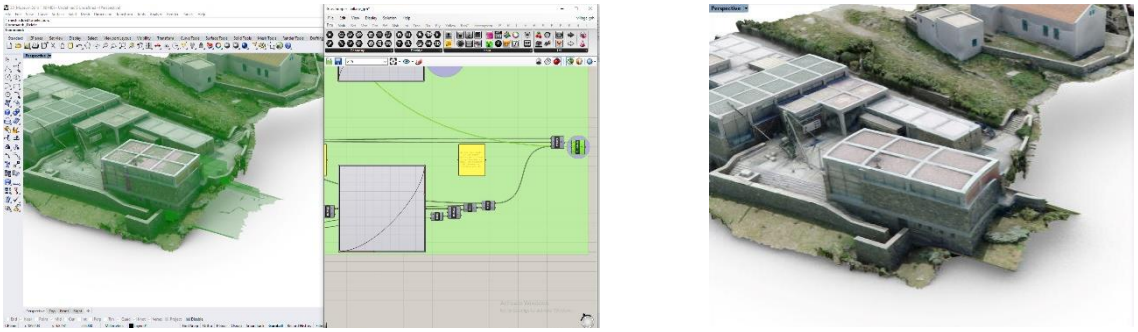


Figure 22. Example of a building geometry change (stretched in the y-axis)

3.2 **3D reconstructed models to 3D CAD models**

Although the approach of modifying parts of the original 3D mesh models that are reconstructed from reality capturing techniques supported by Mindspaces, such as photogrammetry, 3D laser scanning and mobile mapping surveys, seems to have potential, it has also some clear limitations. 3D mesh models are able to represent complex 3D geometries, but as a single, unified, rigid object. Structural components of buildings for example, such as walls, floors and ceilings, windows, furniture, and other objects are not modeled as independent 3D objects and thus cannot be modified independently. Since such modifications need to be supported by the Mindspaces Design Tool, in order to realize the idea of Adaptive 3D-models based on semantic reasoning, it was decided to generate 3D CAD models from the 3D triangular mesh models.

During this task in parallel to “transcribing” the reconstructed models into a more structured representation, the 3D models were also corrected, areas that were occluded in the original models were completed and in general an effort was given to optimize the quality of the obtained 3D reconstructions in areas where quality was inferior, due to limitations of the capturing technologies (highly reflective surfaces like glass, very complex object like vegetation etc).

It must be noted that the generation of semantically enhanced 3D CAD models from 3D meshes is not a fully automated procedure, although certain approaches exist to automate parts of it (such as semantic segmentation that was implemented and described in Section 3.3). It requires a lot of effort and domain specific knowledge and skills, such as 3D and architectural design. Here different approaches and workflows were followed to cover the specific needs of each Pilot Use Case. They are presented in the following sections.

3.2.1 **PUC 1 - Outdoors urban environments (Tecla Sala)**

For the area around the cultural center of Tecla Sala in the City of L' Hospitalet, a 3D mesh model was created in T4.1 by combining images from a DSLR camera, 3D scans and data from the Mindspaces mobile mapping platform (Figure 23). The quality of the 3D mesh was in general high, but the roof of the building was unmodeled, since it was impossible to collect relevant data. The windows of the second floor, parts of the circular bridge, as well as the vegetation were also incomplete.

Figure 23. Final textured 3D mesh model of PUC1

In order to obtain a complete 3D CAD model of the area, 3D Studio Max and Rhino 3D software tools were used in combination (Figure 24).

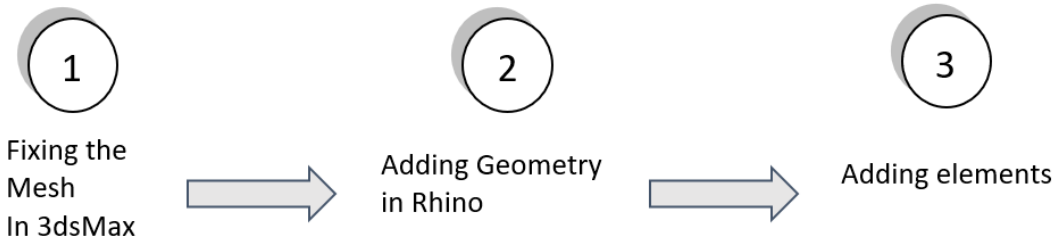


Figure 24. Workflow for the generation of a 3D CAD model for PUC 1

1. Initially the 3D mesh model was cleaned and fixed using 3D Studio Max. This process included:

A) The removal of stray triangles that were apparent on the 3D model, due to errors in the 3D data collection (Figure 25).
B) The removal of unwanted objects that should not be present on the final model (people, bikes, trees) (Figure 25).
C) The completion of small holes in the model, by simulating the missing geometry and replicating the texture from similar parts of the building.

Figure 25. Removal of stray triangles (noise) and unwanted objects (people, bikes, trees)

2. The cleaned mesh model was inserted in Rhino 3D software for further editing. The roof of the cultural center was designed based on the existing geometry of the building and available imagery from Internet sources, like Google Maps[3]. To complete the missing geometry and texture of the second floor windows, replicates of the lower ones were created and placed at the correct positions (Figure 27). The bridge and the surrounding environment were also remodeled based on the noisy data of the existing model (Figure 28).

[3] https://www.google.com/maps

Figure 26. Geometry completion in Rhino (occluded areas - roof)

Figure 27. Geometry completion in Rhino (windows)

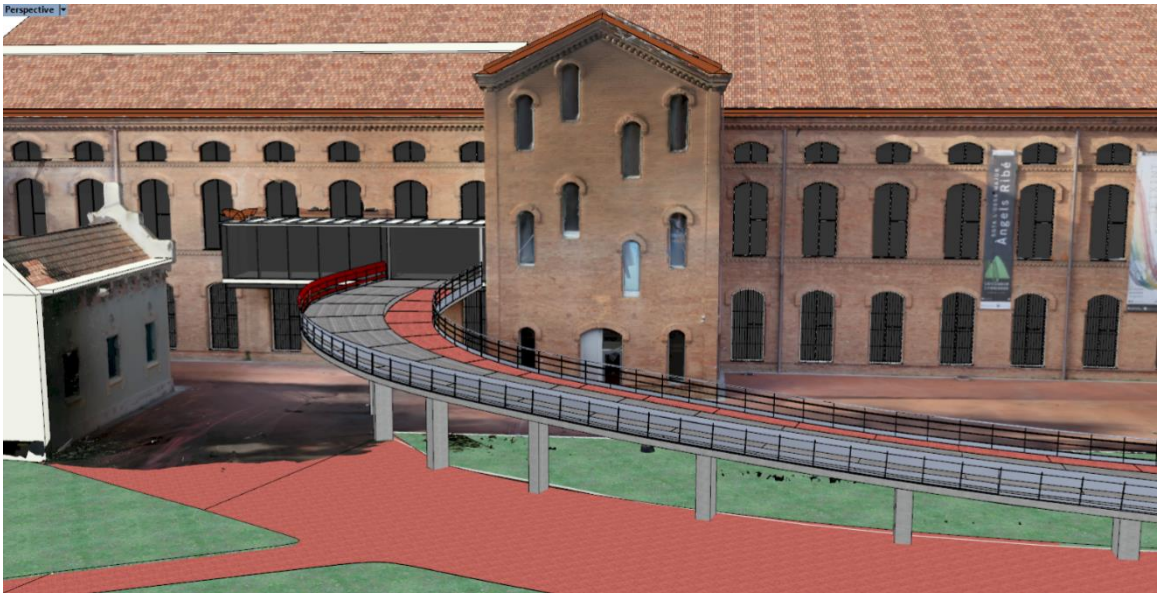


Figure 28. Geometry completion in Rhino (bridge)

3. Finally, elements like trees, benches and light fixtures were added to the 3D model. The corresponding noisy and incomplete triangles of the 3D mesh model were deleted and detailed 3D CAD models from an asset library were placed at the respective positions.

Figure 29. Geometry completion in Rhino. Addition of elements like trees, benches and light fixtures

3.2.2 **PUC 2 - Inspiring workplaces (McNeel Office Space)**

For the McNeel office space, a 3D mesh model was created in T4.1 based on data from a 3D scanning survey with a terrestrial laser scanner, a survey with the mobile mapping platform for indoor spaces and a photocapture survey with a DSLR camera mounted on a tripod dolly (Figure 30).

Figure 30. Initial 3D mesh model from T.4.1

The workflow for obtaining a semantically enhanced 3D CAD model of the office space consists of the following steps (Figure 31):

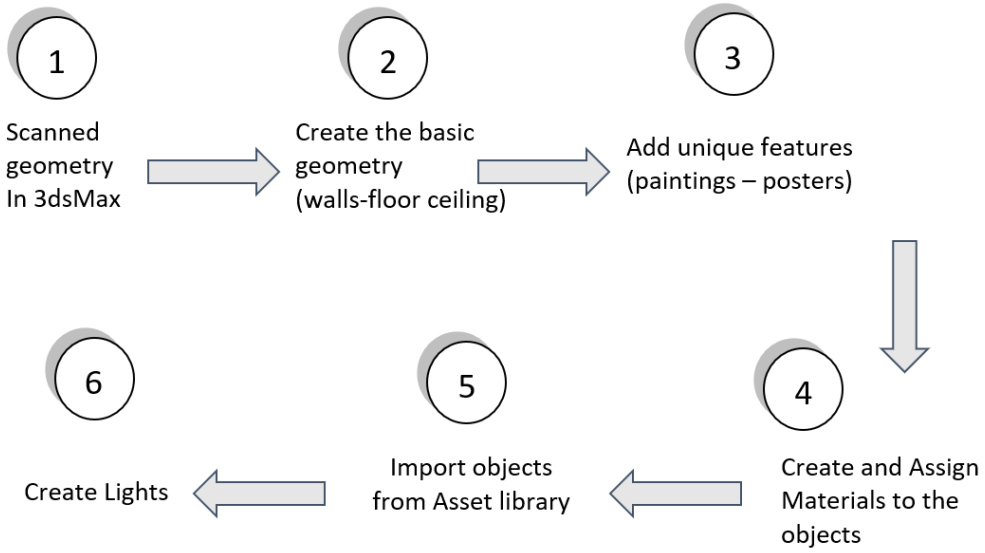


Figure 31. Workflow for the generation of a 3D CAD model for PUC 2

1. The 3D mesh model was inserted in 3D Studio Max software for editing.
2. The main geometry of the building's structural components (walls, columns, floor and ceiling) were traced and simple geometric objects that approximated the 3D mesh geometry were generated (Figure 32).

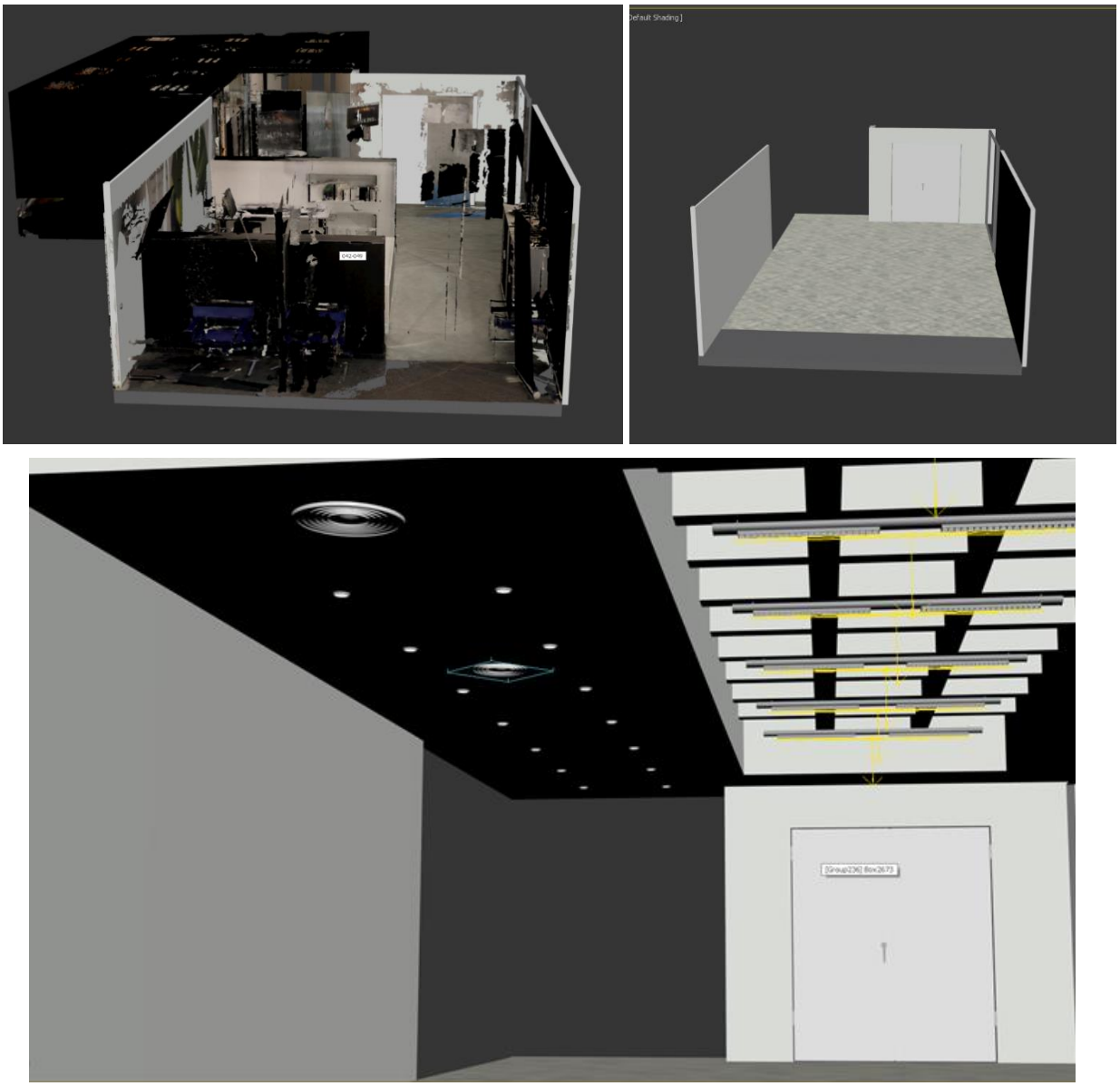


Figure 32. Tracing the base geometry (walls – floor - ceiling)

3. Features that were unique for the specific space, like posters and painting, were identified, as well, and modeled as independent geometric objects (Figure 33).

4. Materials and their properties, such as color, reflectivity, opacity and texture were assigned to the generated geometry. Regarding texture, maps were created in Adobe Photoshop, from suitably selected images from the data collection dataset. Depending on the quality of the initial reconstructed 3D mesh, certain features could be kept as textured triangular meshes and thus the last two steps could be avoided.

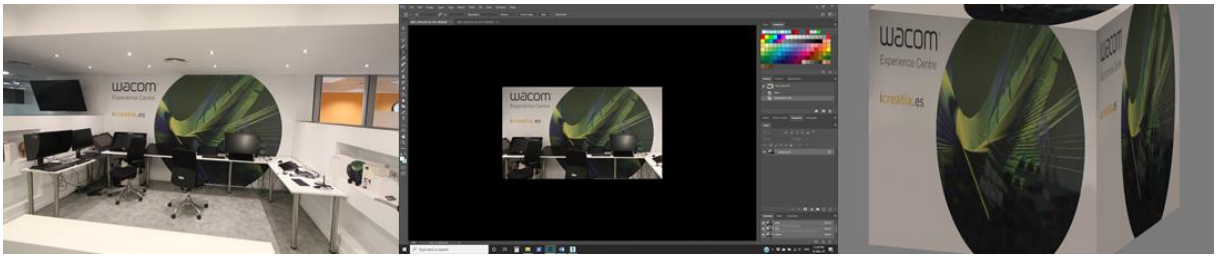


Figure 33. Materials creation from the photographs of the scanned space

5. Regarding furniture, models from an asset library were used to replicate the reconstructed ones. The furniture models were arranged according to the 3D mesh model and merged into the scene (Figure 34).

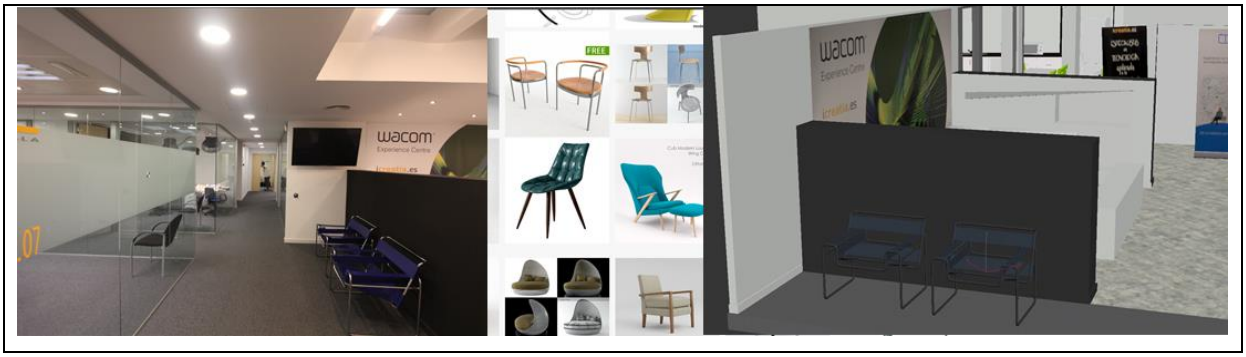


Figure 34. Using an Asset library to import furniture into the scene

6. Finally, lights were added to the scene, as 3D models, but also as light sources. The type of selected lights (omnidirectional, ambient, spot etc) affects how the scene is visualised during rendering and inside the VR tool.

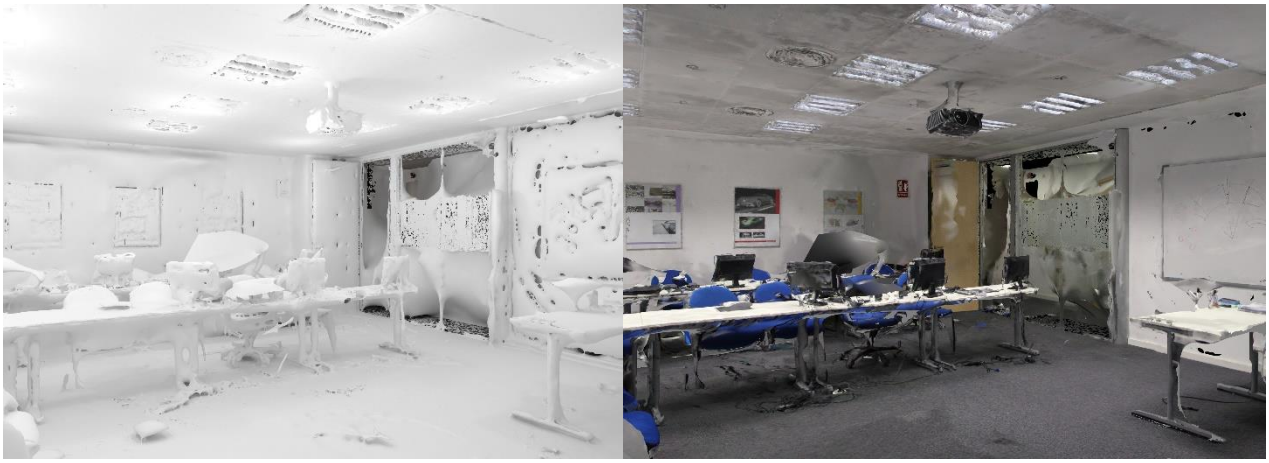
Figure 35. Addition of lights to the scene

Figure 36and Figure 37 show examples of the final generated 3D CAD model. The initial 3D mesh model, with and without texture are also shown for completeness.

Initial 3D mesh from point cloud | Textured 3D mesh

3D CAD model | Rendered 3D CAD model

Figure 36. Example of the generated 3D CAD model



Initial 3D mesh from point cloud | Textured 3D mesh

3D CAD model Rendered 3D CAD model

Figure 37. Example of the generated 3D CAD model

3.2.3 **PUC 3 - Emotionally-sensitive functional interior design (Senior's residence in Paris)**

For the generation of the 3D CAD model of PUC3 the 3D mesh model from T4.1 was used (Figure 39). A workflow similar to the one presented on the previous section (3.2.2) was followed in Rhino 3D software (Figure 38). The results of each processing step are shown in Figure 40- Figure 42.

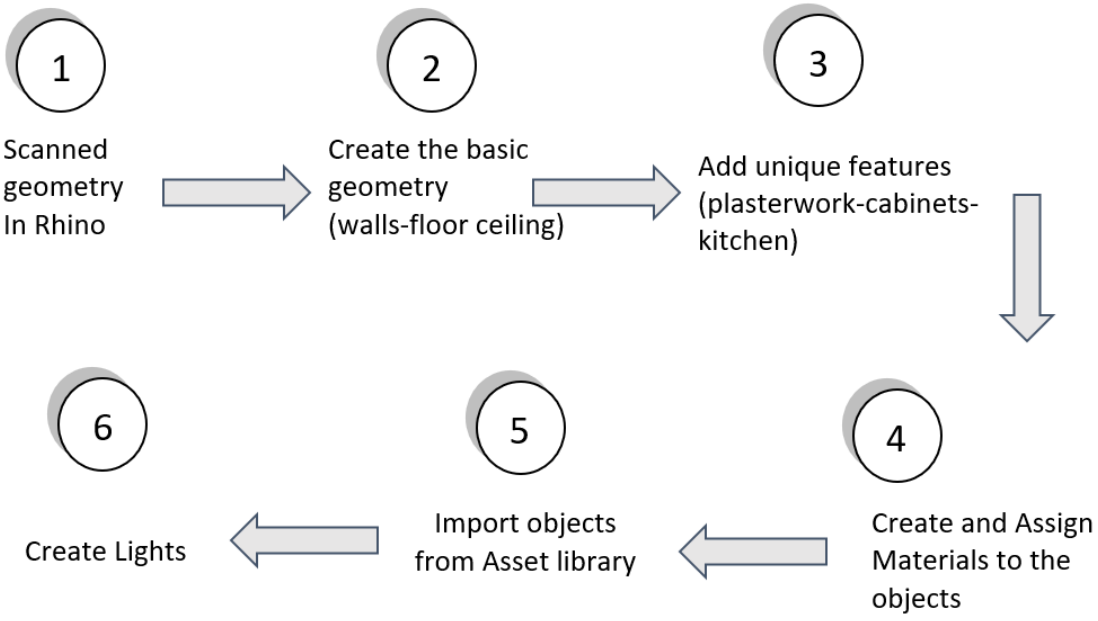


Figure 38. Workflow for the generation of a 3D CAD model for PUC 3

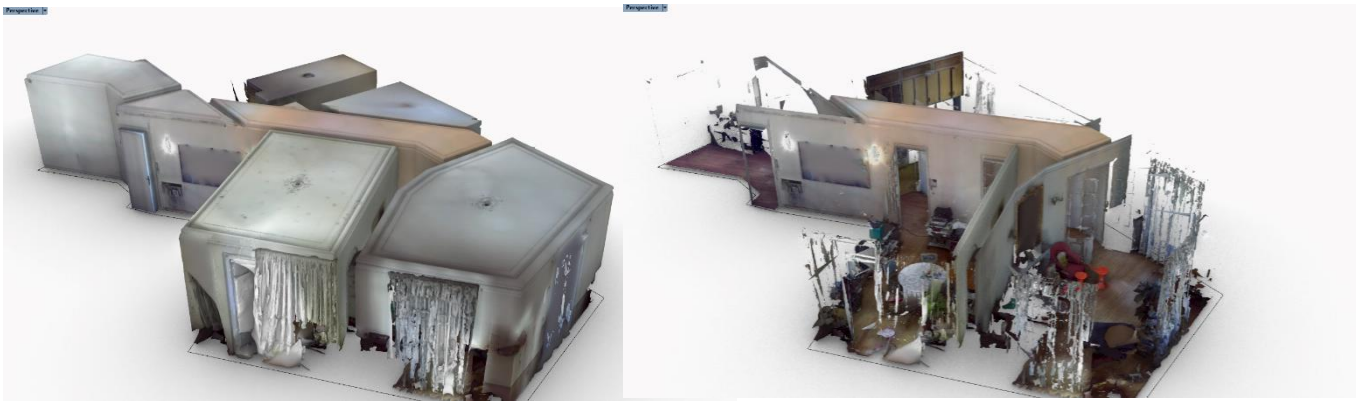


Figure 39. Initial 3D mesh model from T.4.3

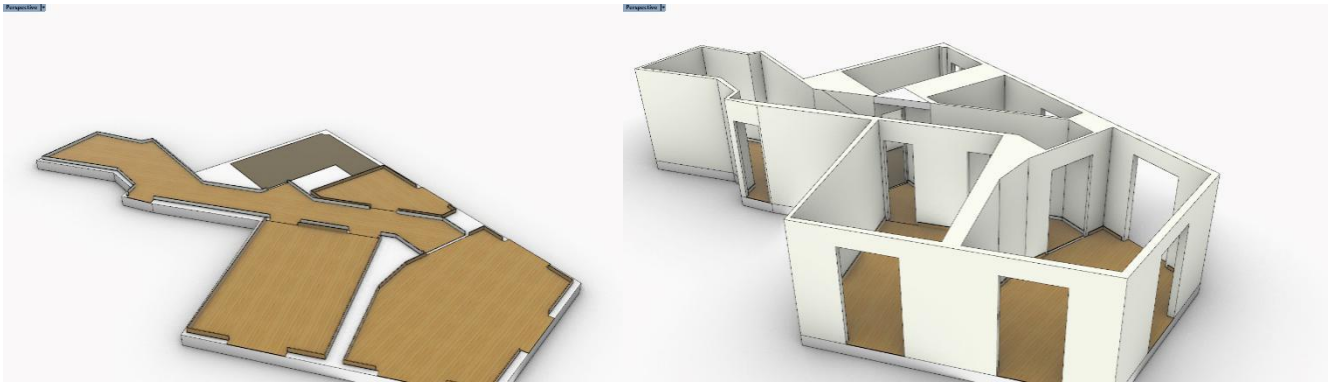


Figure 40. Building the floor and the walls in Rhino

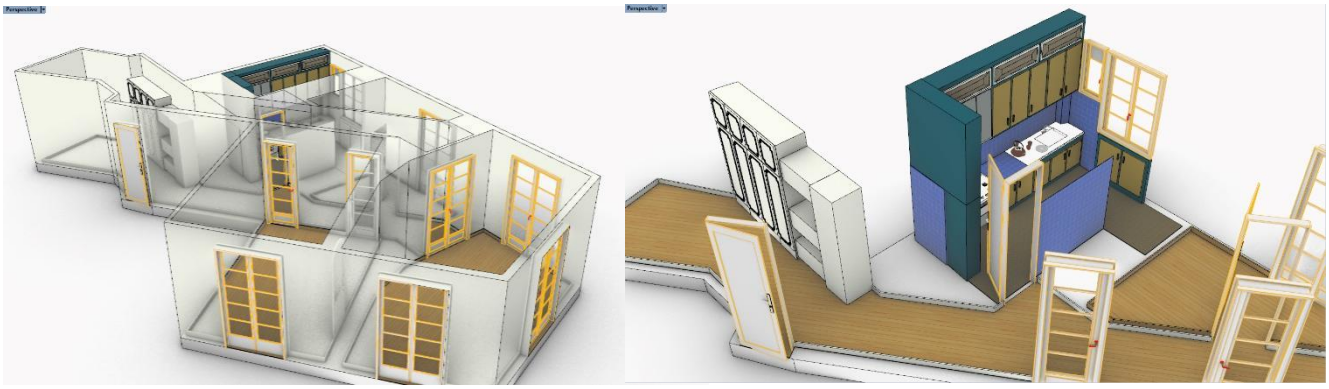


Figure 41. Building other unique features of the building in Rhino

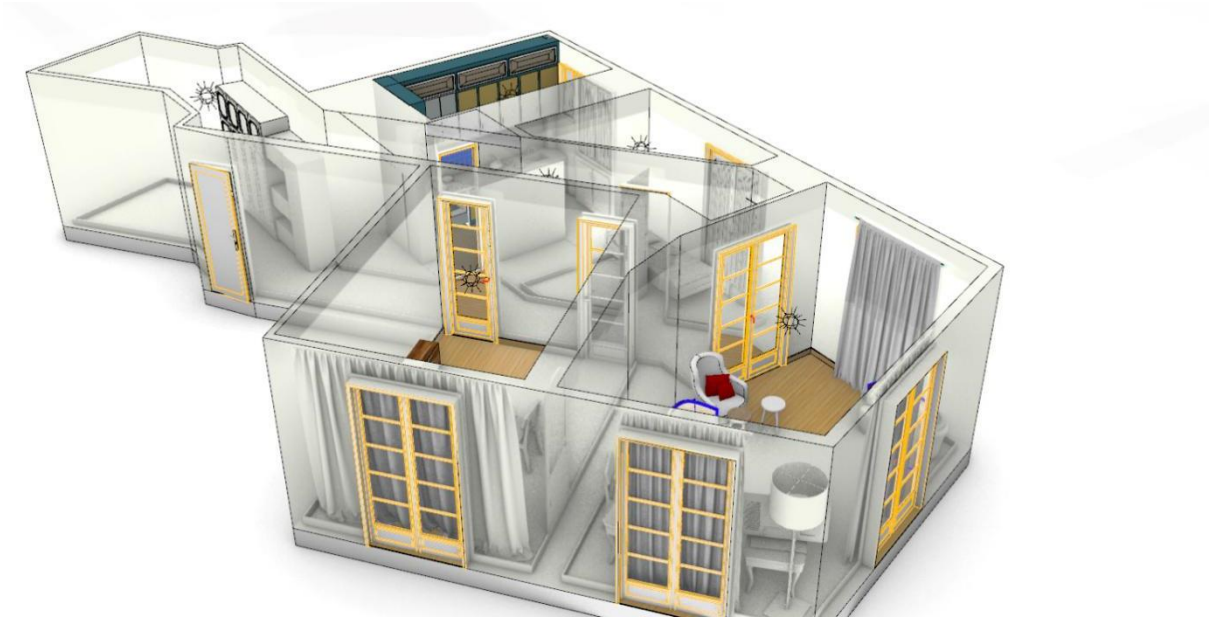


Figure 42. Addition of furniture and lighting in Rhino

3.2.4 **Application of Style Transfer to 3D CAD models**

In order to apply Style Transfer techniques on the 3D models, e.g. via the respective Mindspaces Service that was developed by CERTH, two approaches were implemented and evaluated.

The first approach was to apply style transfer on all images of the data collection missions and then use the altered images to generate the geometry and the texture of the 3D mesh models via image-based modelling techniques like Stricture from Motion (SfM). The results were not satisfactory since the edges of the object were dis-formed and blurred, as the style transfer technique modifies the edges in the images in a non-uniform way.

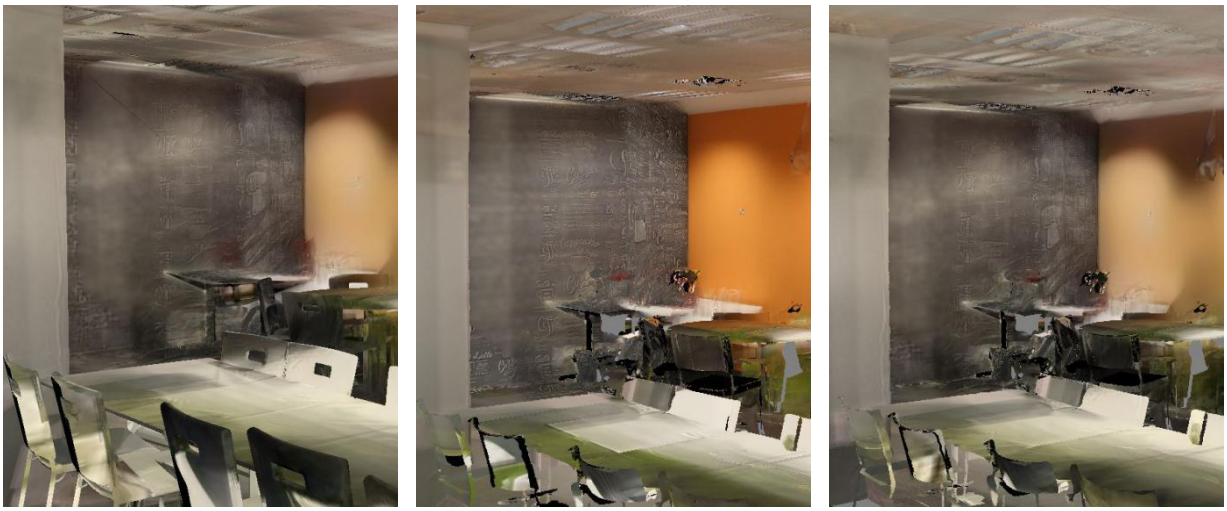


Figure 43. Application of style transfer on the original images of PUC2 dataset

The second approach, which gave results of higher quality, was to apply the style transfer techniques on the materials of certain objects of the generated 3D CAD models. Figure 44to Figure 46present the application of novel materials created through style transfer generation on furniture and decorative objects of a selected meeting room and a private office in PUC2 CAD model.

Figure 44. Application of style transfer on the materials of the 3D CAD model of PUC2

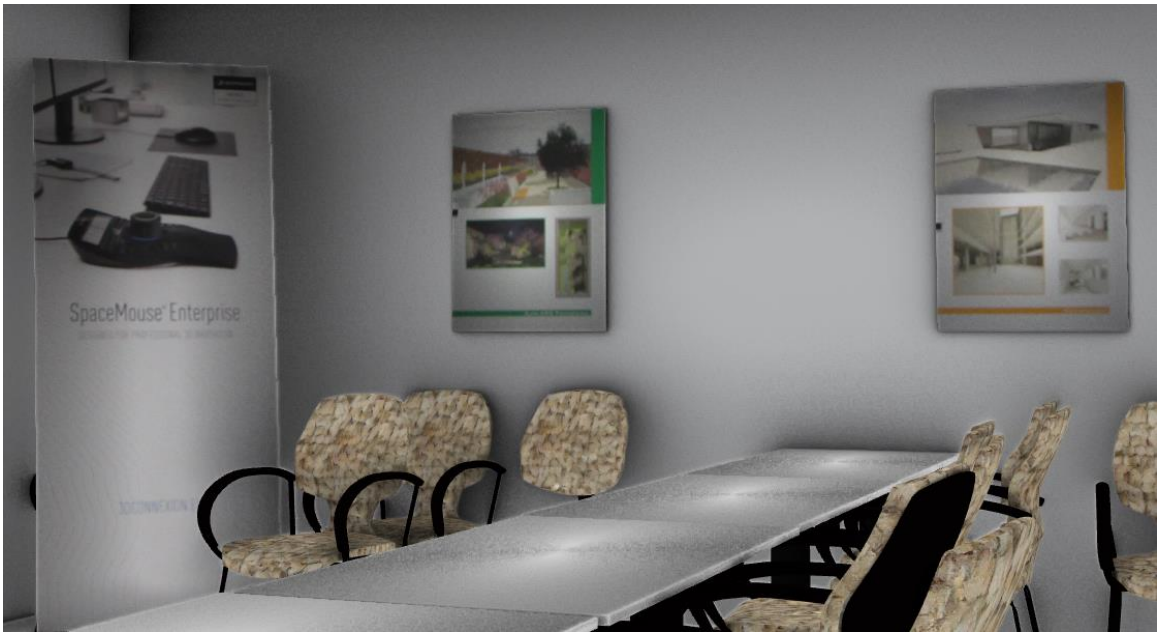


Figure 45. Application of style transfer on the materials of the 3D CAD model of PUC2

Figure 46. Application of style transfer on the materials of the 3D CAD model of PUC2

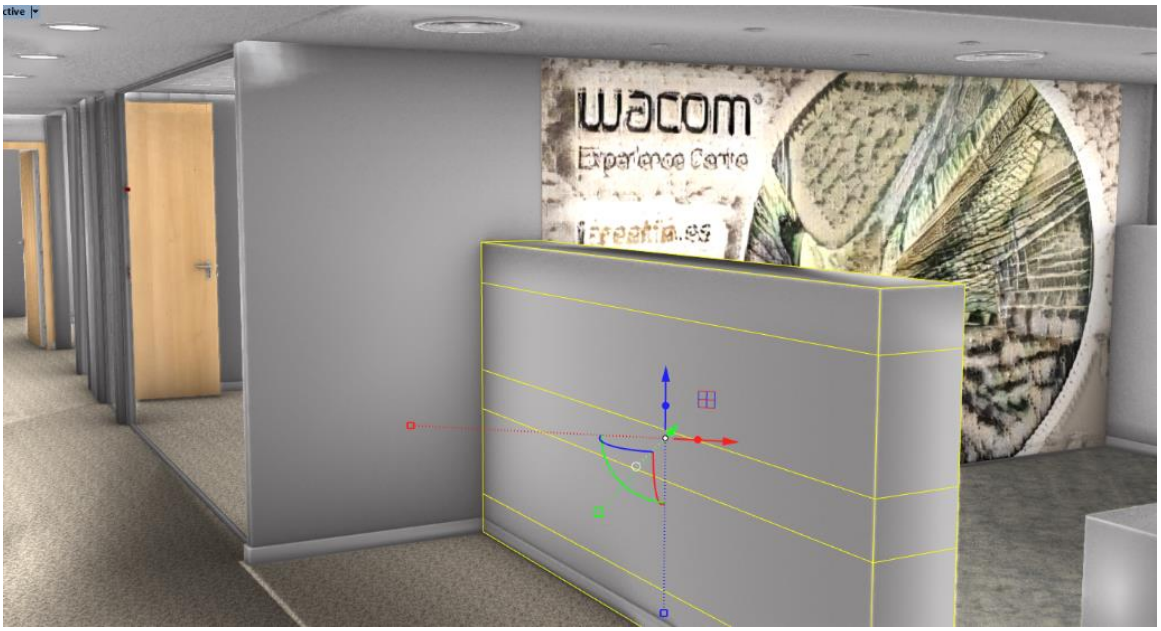


Figure 47. Application of style transfer on the materials of the 3D CAD model of PUC2

3.3 **3D point cloud Semantic Segmentation**

To automate the generation of enhanced 3D CAD models from the reconstructed 3D meshes or 3D point clouds, a necessary step is the automatic identification of structural elements like walls, floors, ceilings, columns etc. or objects like specific pieces of furniture, chairs, desks, boards etc. Semantic segmentation is a machine learning task that aims at segmenting

entire scenes into areas that correspond to specific categories. Although, it is a relative new field it shows a lot of potential and it seems to fit the needs of the Mindspaces platform. In this context, a preliminary study of state-of-the-art algorithm for semantic segmentation on point clouds of indoor spaces was carried out.

3.3.1 State of the Art Semantic Segmentation

Semantic segmentation of point clouds is the task of assigning a label to each 3D point. The last years, research in the field of deep learning has studied the 3D space, mainly focusing, however, on structured 3D formats, i.e. voxel grids. It was not until very recently, when the deep networks were capable of processing the unstructured point cloud data. PointNet is the most widely used network, that consumes directly point clouds and shall be adopted apart from semantic segmentation for object detection and part segmentation. Other solutions include hybrid networks, such as PVCNN, whose input data is points, in order to reduce the memory consumption, but performs the convolutions in voxels to reduce the irregular, sparse data access and improve the locality.

More specifically, PointNet [34] is a unified architecture that directly takes point clouds as input and outputs either class labels for the entire input or per point segment/part labels for each point of the input. The basic architecture of the network is relatively simple as in the initial stages each point is processed identically and independently. In the basic setting each point is represented by just its three coordinates (X, Y, Z). Additional dimensions may be added by computing normals and other local or global features. The key to the approach is the use of a single symmetric function, max pooling. The network learns a set of optimization functions/criteria that select interesting or informative points of the point cloud and encode the reason for their selection. The final fully connected layers of the network aggregate these learnt optimal values into the global descriptor for the entire shape (shape classification) or are used to predict per point labels (shape segmentation).

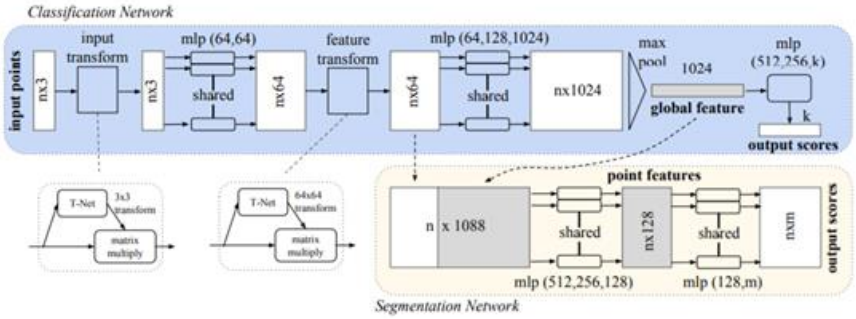


Figure 48. PointNet Architecture. The classification network takes n points as input, applies input and feature transformations, and then aggregates point features by max pooling. The output is classification scores for k classes. The segmentation network is an extension to the classification net. It concatenates global and local features and outputs per point scores. “MLP” stands for multi-layer perceptron, numbers in bracket are layer sizes. Batchnorm is used for all layers with ReLU. Dropout layers are used for the last MLP in classification net.

While examining the network, experiments were conducted on s3dis datasets. Results on the dataset are tabulated in the following table, using the metric of Intersection over Union (IoU).

Table 11. Results of PointNet on s3dis dataset, using the metric of IoU

Class	IoU(%)
ceiling	88.5
floor	96.8
wall	71.1
beam	0.0
column	0.0
window	49.8
door	9.3
table	61.9
chair	60.2
sofa	21.7
bookcase	57.8
board	33.5
clutter	35.9

Point-Voxel CNN (PVCNN) [35] represents the 3D input data as point clouds to take advantage of the sparsity to reduce the memory footprint. However, afterwards the network leverages the voxel-based convolution. In order to perform these actions, the algorithm normalizes the coordinates before converting the point cloud into the volumetric domain. Voxelization is then performed and feature aggregation takes place. After converting the points into voxel grids a stack of 3D volumetric convolutions to aggregate the features, is applied. Similar to conventional 3D models, the batch normalization is applied and the nonlinear activation function after each 3D convolution. Finally, devoxelization is performed. More specifically, the trilinear interpolation is used, to transform the voxel grids to points to ensure that the features mapped to each point are distinct.

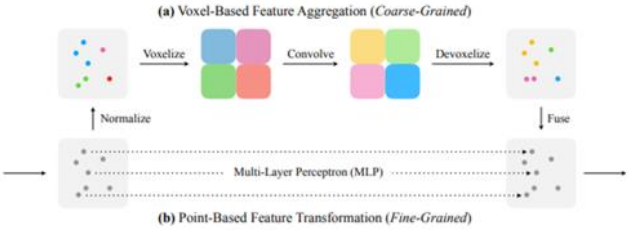


Figure 49. PVConv is composed of a low-resolution voxel-based branch and a high-resolution point based branch. The voxel-based branch extracts coarse-grained neighborhood information, which is supplemented by the fine-grained individual point features extracted from the point-based branch.

Some results on s3dis dataset are tabulated in the following table, using the metric of Intersection over Union (IoU).

Table 12. Results of PVCNN on s3dis dataset.

Class	**IoU(%)**
ceiling	51.05
floor	93.28
wall	98.11
column	78.51
beam	0.00
window	14.68
door	27.35
table	52.05
chair	72.04
bookcase	75.07
sofa	15.97
board	63.29
clutter	38.53

JSIS-3D [36] couples semantic and instance segmentation into a single task. Towards this goal, a network architecture namely multi-task pointwise network (MT-PNet) simultaneously performs two tasks: predicting the object categories of 3D points in a point cloud and embedding these 3D points into high-dimensional feature vectors that allow clustering the points into object instances. More specifically, given the 3D point cloud, it is first scanned entirely by overlapping 3D windows. Each window (with its associated 3D vertices) is passed to a neural network for predicting the semantic class labels of the vertices within the window and embedding the vertices into high-dimensional vectors. To enable such tasks, a multi-task pointwise network (MT-PNet) is developed, aiming to predict an object class for every 3D point in the scene and at the same time to embed the 3D point with its class label information into a vector. The network encourages 3D points belonging to the same object instance be pulled to each other while pushing those of different object instances as far away from each other as possible. Those class labels and embeddings are then fused into a

multi-value conditional random field (MVCRF) model. The semantic and instance segmentation are finally performed jointly using variational inference. Results on s3dis dataset are tabulated in the following table, using the metric of accuracy.

Table 13. Results of PointNet on s3dis dataset, using the metric of accuracy

Class	Accuracy (%)
ceiling	87.4
floor	98.4
wall	99.6
window	94.4
door	59.7
table	24.9
chair	80.6
sofa	84.9
bookcase	30.0
board	63.0
bookcase	52.5
clutter	70.5
beam	0.00

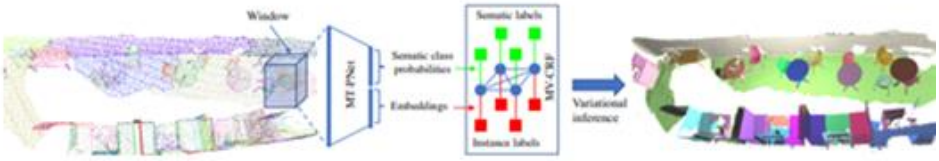


Figure 50. Pipeline of our proposed method. Given an input 3D point cloud, the point cloud is scanned by overlapping windows. 3D vertices are then extracted from a window and passed through the multi-task neural network to get the semantic labels and instance embeddings. Then, a multi-value conditional random field model is optimised to produce the final results.

Other SoA networks, working on point cloud semantic segmentation include a dynamic graph CNN [37], EdgeConv, which captures local geometric structure while maintaining permutation invariance. More specifically, EdgeConv generates edge features that describe the relationships between a point and its neighbors. EdgeConv is designed to be invariant to the ordering of neighbors, and thus is permutation invariant. Because EdgeConv explicitly constructs a local graph and learns the embeddings for the edges, the model is capable of

grouping points both in Euclidean space and in semantic space. More specifically, the model can learn to semantically group points by dynamically updating a graph of relationships from layer to layer.

SEGcloud [38], is also another approach for semantic segmentation of point clouds, using deep learning. SEGCloud is an end-to-end framework to obtain 3D point-level segmentation that combines the advantages of NNs, trilinear interpolation (TI) and fully connected Conditional Random Fields (FC-CRF). Coarse voxel predictions from a 3D Fully Convolutional NN are transferred back to the raw 3D points via trilinear interpolation. Then, the authors use a Fully Connected Conditional Random Field (FC-CRF), which is deployed to infer 3D point labels while ensuring spatial consistency. Transferring class probabilities to points before the CRF step, allows the CRF to use point level modalities (color, intensity, etc.) to learn a fine-grained labeling over the points, which can improve the initial coarse 3D-FCNN predictions. Finally, an efficient CRF implementation is used to perform the final inference.

3.3.2 **State of the art Datasets**

S3DIS Building Parser

One of the most frequently used datasets for assessing the performance of indoor semantic segmentation is Stanford s3dis dataset[1]. The dataset comprises of six large scale indoor areas, that originate from 3 different buildings of mainly educational and office use. The five areas are used for testing and one of them for training. The point cloud is semantically annotated and covers a total area of approximately 6000m2, with 695,878,620 points in total. Figure 50 depicts a raw point cloud of the dataset. The dataset includes 13 different semantic classes.

Figure 51. Area 6 of raw point cloud

ScanNet

ScanNet is an RGB-D video dataset containing 2.5 million views in more than 1500 scans, annotated with 3D camera poses, surface reconstructions, and instance-level semantic segmentations. To collect this data, the authors designed an easy-to-use and scalable RGB-D capture system that includes automated surface reconstruction and crowdsourced semantic annotation. Sample scenes of ScanNet are depicted in Figure 52.

Figure 52. Sample scenes of ScanNet dataset

SceneNN

SceneNN is a scene meshes dataset of indoor scenes with cluttered objects at room scale. SceneNN consists of RGB-D images depicting more than 100 indoor scenes. Scenes are captured at various places, e.g., offices, dormitory, classrooms, pantry, etc., from University of Massachusetts Boston and Singapore University of Technology and Design. All scenes are reconstructed into triangle meshes and have per-vertex and per-pixel annotation. Their semantic segmentation follows NYU-D v2 category set, which has 40 semantic classes. On this dataset, a train/test split also exists. Some groundtruth segmented meshes are depicted in Figure 53.

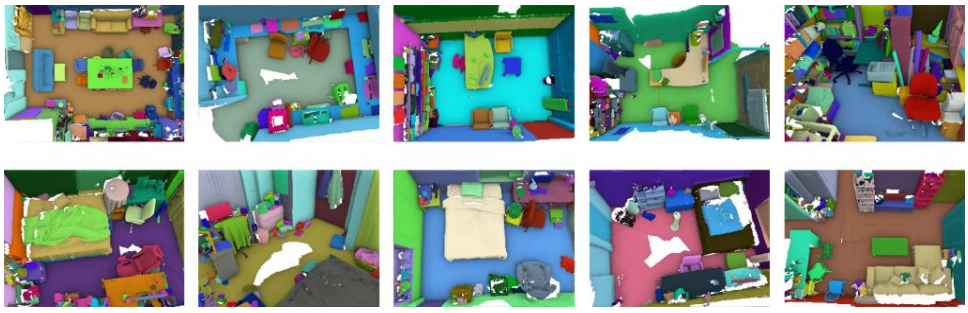


Figure 53. Sample groundtruth meshes of SceneNN

3.3.3 **The project's dataset**

Concerning the project's use case, we have scanned McNeel's offices in Barcelona and processed the dataset to align it with s3dis release. Towards this goal, we have documented the format of the input data of s3dis data, as depicted in Figure 54and followed a concrete workflow, which is described in Figure 55.

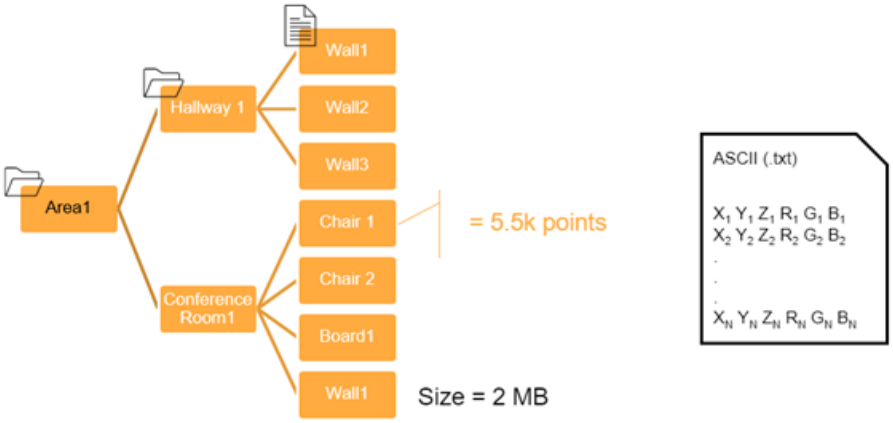


Figure 54. Format of S3dis dataset

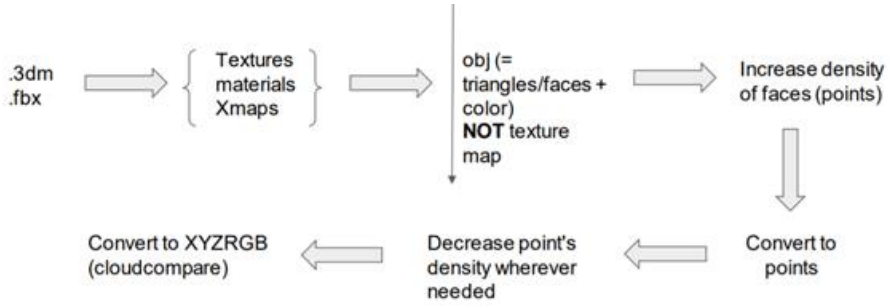
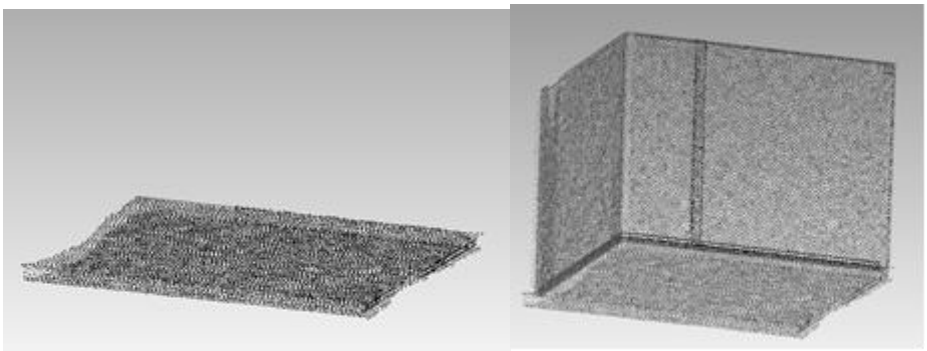
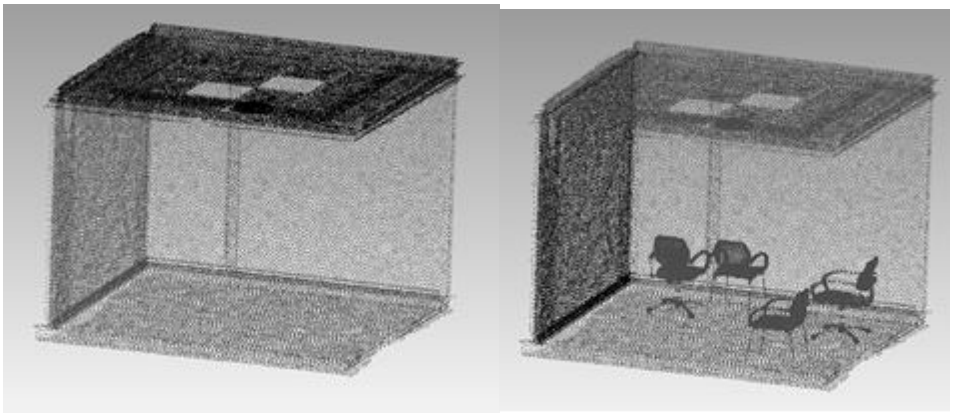


Figure 55. Processing of our dataset

Our dataset comprises of 13 different semantic classes, namely walls, doors, glass-walls, floor, ceiling, column, objects on the wall, plants, kitchen, sitting, bookcases, desks and lights. Some of the different semantic classes (i.e. floor, sitting, ceiling) are depicted in the following figure, for one conference room.



(a) (b)



(c) (d)

Figure 56. Semantic components of our dataset (a) floor, (b) floor and walls, (c) floor, walls and ceiling, (d) floor, walls, ceiling and sitting

3.3.4 **Results on McNeel Dataset**

The trained model of PVCNN (as trained from scratch) has been used in order to test its performance on the McNeel's dataset predictions. As the classes of s3dis dataset (on which PVCNN was built) slightly differ from the semantic classes of the project's dataset, we have decided to assign all the classes that are not consistent with the S3dis dataset to the clutter class. The accuracy of the predictions is shown on Table 14, while in Table 15, inference results are tabulated in the form of a confusion matrix, for a conference room of the McNeel dataset. Visual comparisons between ground truth data and prediction are also presented in Figure 57and Figure 58.

Table 14. Results of modified PVCNN t on McNeel room dataset, using the metric of accuracy

Category	**ceiling**	**floor**	**wall**	**chair**	**bookcase**	**board**	**clutter**
Accuracy (%)	35.38	14.36	59,83	41.34	0.01	0.00	48.2

Table 15. Confusion matrix of predicted (rows) and groundtruth (columns) labels

	clutter	ceiling	floor	wall	beam	column	door	window	table	chair	sofa	Bookcase	board
clutter	10190	8915	5007	5526	0	0	0	0	0	15659	0	19689	9
ceiling	5958	9955	0	31	0	0	0	0	0	0	0	0	0

floor	0	0	6442	1	0	0	0	0	0	0	0	0	0
wall	0	3218	1367	20991	0	0	0	0	0	0	0	873	1667
beam	0	62	0	10	0	0	0	0	0	0	0	0	0
column	0	0	0	0	528	0	0	0	0	0	0	4	6
door	0	0	83	810	0	0	0	0	0	0	0	0	26
window	0	0	0	4	0	0	0	0	0	0	0	0	4
table	0	0	3	0	0	0	0	0	0	11	0	23	0
chair	0	0	367	0	0	0	0	0	0	18348	0	11	0
sofa	0	0	91	0	0	0	0	0	0	9154	0	0	0
bookcase	0	0	3	0	0	0	0	0	0	836	0	2	0
board	0	0	0	0	0	0	0	0	0	0	0	0	0

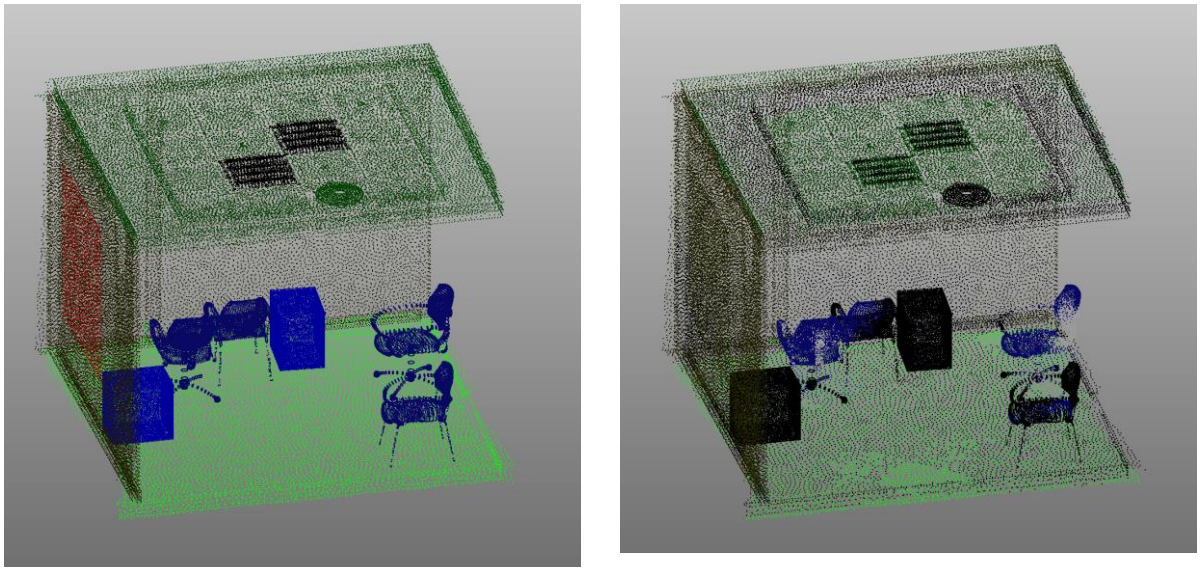


Figure 57. Preliminary pointcloud semantic segmentation on a McNeel room from modified PVCNN. Ground truth (left) vs inference results (right)

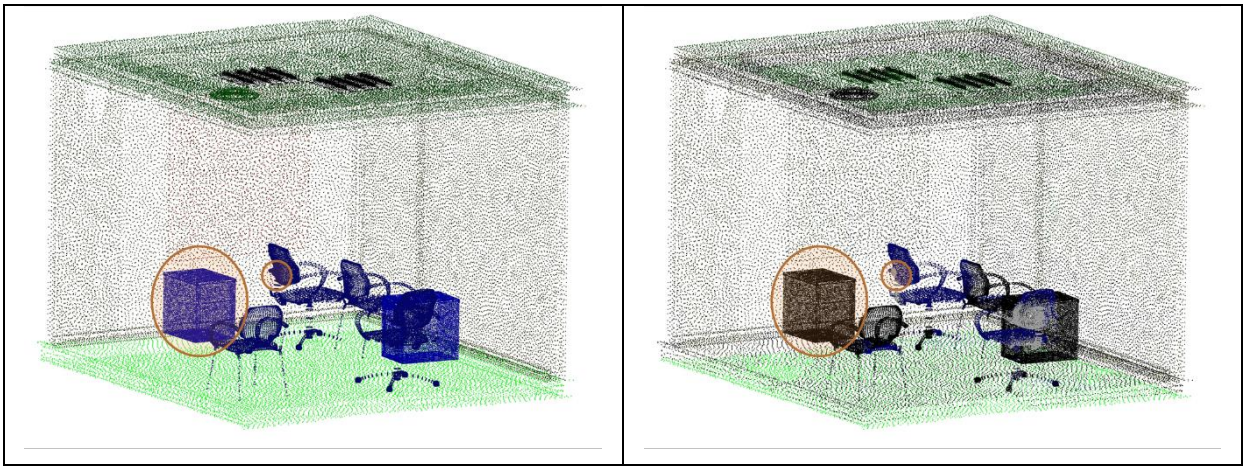


Figure 58. Failures of the inference results. Ground truth (left) vs inference results (right)

4 SIMULATION EXAMPLE

In order to validate the semantic development and implementations presented previously in this document and showcase the usefulness and the robustness of the frameworks, we conducted a simulation example which capitalizes on the semantic representation and data integration to perform semantic reasoning for emotion-based virtual adaptations. More specifically, we have generated and tested upon an initial virtual office environment which composes the first virtual 3D scenery prior to commencement of the emotional analysis (Fig. 59) .

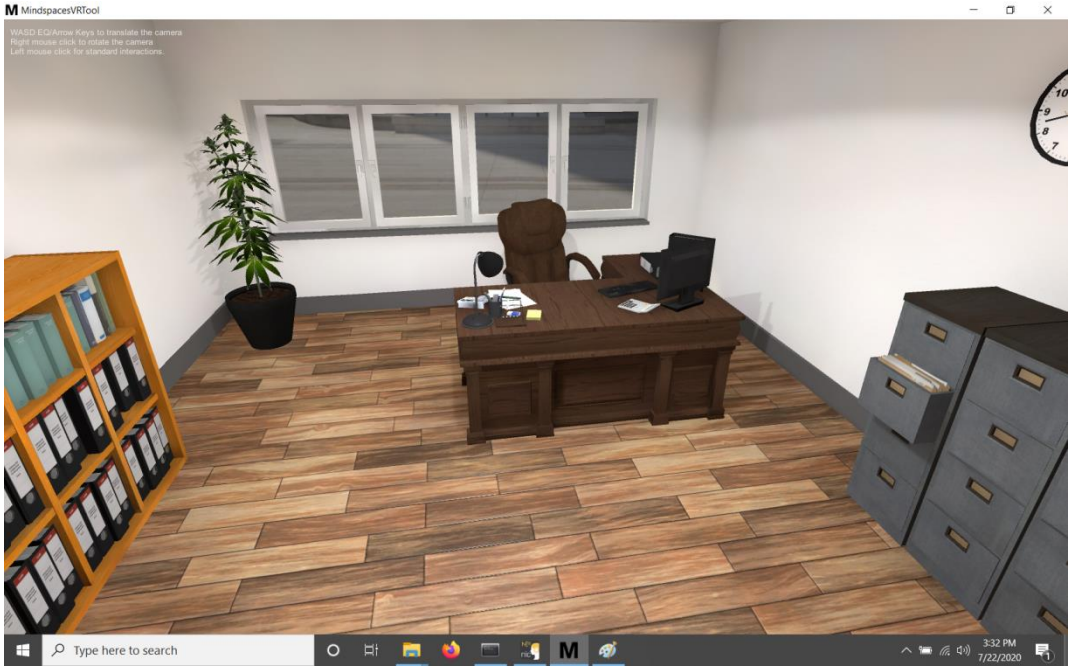


Figure 59: Initial testing virtual environment

As the emotional analysis service starts functioning is is generating emotional states which then are forwarded to the VR Tool, wrapped with additional metadata and being sent to the node.js online service; the intermediate layer among the knowledge base and the rest components. A viable example of JSON receival is depictioned in Fig. 60. There are currently 4 distinct emotional states and the logic behind choosing the suggested design configuration is to have a different set up for each distinct emotional state. For instance, when the first emotional state generated is happy, the suggested design configuration ID is 15 which corresponds to the “appear” functionality of 2 3D objects, 2 white chairs, in front of the desk (see. Fig. 62).

```
{
  "project_id": 20,
  "current_configuration_id": 20,
  "subject_id": 23,
  "experiment_id": 23,
  "emotional_state": {
    "timestamp": 1596025090,
    "eeg_tag": 1,
    "valence": 1,
    "arousal": 1
  },
  "location_info": {
    "hotspot_id": 15,
    "pos_x": 0,
    "pos_y": 24,
    "pos_z": 10
  }
}
```


Figure 60: JSON example from VR tool

The JSON presented in Fig. 60 is received and translated into the RDF syntax and stored inside the knowledge base. As can been seen in Fig. 61, every datum is represented in the form of a triplet (turtle syntax) except for the “pos_z” which is the height of the location which is not supported until the latest iteration.

```
@@prefix : <http://www.semanticweb.org/MindSpaces#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://www.semanticweb.org/MindSpaces> .

<http://www.semanticweb.org/MindSpaces> rdf:type owl:Ontology .

:emostate4981238 rdf:type owl:NamedIndividual ,
                          :EmotionalState ;
                 :EmotionalStatehasDate 1596025090 ;
                 :EmotionalStatehasState 1 ;
                 :hasArousal 1 ;
                 :hasValence 1 .

:expid592904 rdf:type owl:NamedIndividual ,
                       :getChange ;
             :getChangeProjectID 23 .


:locinfo1239823 rdf:type owl:NamedIndividual ,
                         :Location ;
                :hasCoordX <http://www.semanticweb.org/MindSpaces#0> ;
                :hasCoordY <http://www.semanticweb.org/MindSpaces#24> .

:prid2312423 rdf:type owl:NamedIndividual ,
                       :Project ;
             :ProjectID 20 .


:subj1623490 rdf:type owl:NamedIndividual ,
                       :Subject ;
             :SubjectID 23 .


<http://www.semanticweb.org/MindSpaces#0> rdf:type owl:NamedIndividual ,
                                                                        :CoordX .

<http://www.semanticweb.org/MindSpaces#24> rdf:type owl:NamedIndividual ,
                                                                         :CoordY .
```

Figure 61: RDF turtle-based syntax example

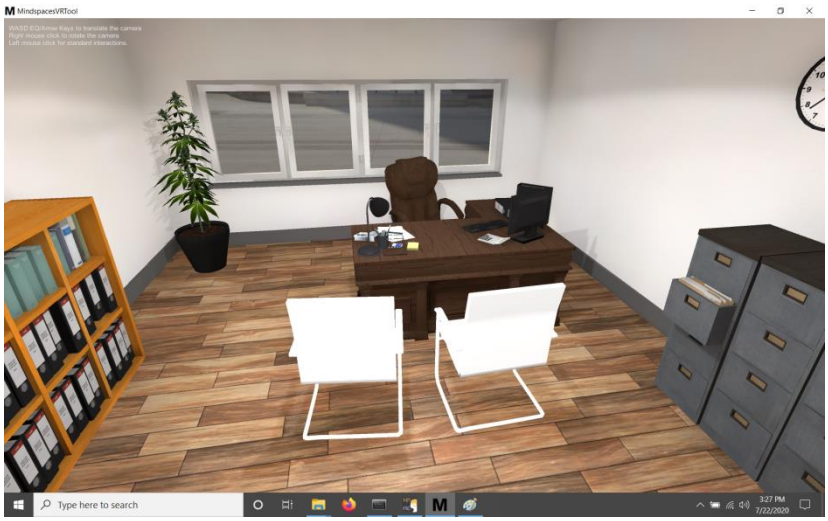


Figure 62: Virtual environment transformation after receiving happy emotional state

The Design Configuration IDs and the Hotspot IDs with additional metadata are retrieved each time with API calls directly from the data storage. Apart from multiple design configurations, there are also present different Hotspot IDs that can be suggested during this example, but for the sake of the simulation presentation and general demonstration to be more easily understandable, we chose to lock to a steady Hospot ID for the changes to be easily understood. In case of angry emotional state, the changes suggested and inflicted in the virtual environment are shown in Fig. 63, where a “white board” object appears in the wall on the right side.

Figure 63: Virtual environment transformation after receiving angry emotional state

For exposition purposes we also checked validity for the rest emotional states. Apart from the changes inflicted from the happiness emotional state (Fig. 62) and the anger emotional state (Fig. 63), the sadness emotional state and the relaxed emotional state are present. In the former, the change materialised in the virtual environment is the “move” functionality of the 3D object “plant” from the left corner to the right corner of the room (Fig. 64), which corresponds to design configuration with id 18 while simultaneously also maintaining the same Hotspot ID equal to 25.

Figure 64: Virtual environment transformation after receiving sad emotional state

Finally, the last emotional state, which corresponds to a relaxed mental state of the EEG subject, inflicts a change wall texture functionality from plain white texture to a brick texture while removing all other changes made so far during previous emotional states received (Fig. 65), in a manner where the design configuration, with ID, 20 is built directly on the initial base model designed, (Fig. 59), thus negating every design configuration materialised until now before applying the latest transformation.

Figure 65: Virtual environment transformation after receiving relaxed emotional state

5 CONCLUSIONS

In this deliverable, there are details about semantic-related requirements as well as concrete specifications of them and an up-to-date presentation of the state-of-the-art of the developed data-unifying ontology and semantic representation formats as elicited by T5.3 “Semantic representation and data integration”. Moreover, there is an extensive presentation regarding the status of the MindSpaces ontologies towards version 1 at the moment which contain in a graph-based approach the vocabularies that describe most suitably the MindSpaces context. Furthermore, we demonstrated the initial version of the reasoning framework towards version 1 as elicited by T5.4 “Semantic reasoning for emotion-based space adaptation” for integrating multimodal data and fusing their knowledge so as to construe and enhance the knowledge depicted in the graphs of the knowledge base. In addition, there was a detailed presentation about the generation of innovative texture mappings with the goal of visualizing on the surface of the acquired 3D models the data captured by the emotion sensors as educing by T5.6 “Development of semantically enhanced interactive 3D spaces”.

For future iterations, the related to this document WP5 frameworks will include further progress and enhancements through each phase. The ontological models will be enriched, as the project matures more, to conceptualize more intricate domain knowledge in complex semantic structures and relationships, as the analysis results become more concrete, precise and adequately enough. Reasoning will be enhanced in various approaches such as by enriching the semantics by defining additional axioms, both class and property axioms, and by incorporating additional inference rules. Specialised techniques are also going to be developed which will handle imperfect information, noise and general errors to achieve maximum robustness. Complementarily, the rule authoring tool will proceed from the conceptual stage to full materialization and implementation. Finally, for point cloud segmentation, the aim is the creation of novel datasets of actual and diverse cases to develop and evaluate 3D semantic segmentation algorithms resulting in building a robust and effective DL architecture for basic and commonly found categories and deploy a service for the MindSpaces platform.

6 REFERENCES

[1] Grau, B. C., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., & Sattler, U. (2008). OWL 2: The next step for OWL. Journal of Web Semantics, 6(4), 309-322.

[2] Baader, F., Calvanese, D., McGuinness, D., Patel-Schneider, P., & Nardi, D. (Eds.). (2003). The description logic handbook: Theory, implementation and applications. Cambridge university press.

[3] McGuinness, D. L., & Van Harmelen, F. (2004). OWL web ontology language overview. W3C recommendation, 10(10), 2004.

[4] OWL Working Group. (2009). OWL 2 Web Ontology Language Document Overview: W3C Recommendation 27 October 2009.

[5] Miller, E. (1998). An introduction to the resource description framework. D-lib Magazine.

[6] Gruber, T. R. (1993). A translation approach to portable ontology specifications. Knowledge acquisition, 5(2), 199-220.

[7] Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., & Katz, Y. (2007). Pellet: A practical owl-dl reasoner. Journal of Web Semantics, 5(2), 51-53.

[8] Glimm, B., Horrocks, I., Motik, B., Stoilos, G., & Wang, Z. (2014). HermiT: an OWL 2 reasoner. Journal of Automated Reasoning, 53(3), 245-269.

[9] Dolby, J., Fokoue, A., Kalyanpur, A., Schonberg, E., & Srinivas, K. (2009). Scalable highly expressive reasoner (SHER). Journal of Web Semantics, 7(4), 357-361

[10] Haarslev, V., Hidde, K., Möller, R., & Wessel, M. (2012). The RacerPro knowledge representation and reasoning system. Semantic Web, 3(3), 267-277.

[11] Tsarkov, D., & Horrocks, I. (2006, August). FaCT++ description logic reasoner: System description. In International joint conference on automated reasoning (pp. 292-297). Springer, Berlin, Heidelberg.

[12] Cotta, C., Reich, S., & Ligeza, A. (Eds.). (2008). Knowledge-Driven Computing: Knowledge Engineering and Intelligent Computations (Vol. 102). Springer.

[13] Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., & Dean, M. (2004). SWRL: A semantic web rule language combining OWL and RuleML. W3C Member submission, 21(79), 1-31.

[14] Motik, B., Horrocks, I., Rosati, R., & Sattler, U. (2006, November). Can OWL and logic programming live together happily ever after?. In International semantic web conference (pp. 501-514). Springer, Berlin, Heidelberg.

[15] Grosof, B. N., Horrocks, I., Volz, R., & Decker, S. (2003, May). Description logic programs: combining logic programs with description logic. In Proceedings of the 12th international conference on World Wide Web (pp. 48-57).

[16] Motik, B., Sattler, U., & Studer, R. (2005). Query answering for OWL-DL with rules. Journal of Web Semantics, 3(1), 41-60.

[17] ter Horst, H. J. (2005, November). Combining RDF and part of OWL with rules: Semantics, decidability, complexity. In International Semantic Web Conference (pp. 668-684). Springer, Berlin, Heidelberg.

[18] Harris, S., Seaborne, A., & Prud'hommeaux, E. (2013). SPARQL 1.1 query language. W3C recommendation, 21(10), 778.

[19] Pérez, J., Arenas, M., & Gutierrez, C. (2006, November). Semantics and Complexity of SPARQL. In International semantic web conference (pp. 30-43). Springer, Berlin, Heidelberg.

[20] Knublauch, H., Hendler, J. A., & Idehen, K. (2011). SPIN-overview and motivation. W3C Member Submission, 22, W3C.

[21] Méndez, S. J. R., & Zao, J. K. (2018). BCI Ontology: A Context-based Sense and Actuation Model for Brain-Computer Interactions. In SSN@ ISWC (pp. 32-47).

[22] Scherp, A., Franz, T., Saathoff, C., & Staab, S. (2009, September). F--a model of events based on the foundational ontology dolce+ DnS ultralight. In Proceedings of the fifth international conference on Knowledge capture (pp. 137-144).

[23] Gangemi, A., & Mika, P. (2003, November). Understanding the semantic web through descriptions and situations. In OTM Confederated International Conferences" On the Move to Meaningful Internet Systems" (pp. 689-706). Springer, Berlin, Heidelberg.

[24] Belhajjame, K., B'Far, R., Cheney, J., Coppens, S., Cresswell, S., Gil, Y., ... & Miles, S. (2013). Prov-dm: The prov data model. W3C Recommendation.

[25] Dublin Core Metadata Initiative. (2012). Dublin core metadata element set, version 1.1.

[26] De Nicola, A., Missikoff, M., & Navigli, R. (2005, August). A proposal for a unified process for ontology building: UPON. In International Conference on Database and Expert Systems Applications (pp. 655-664). Springer, Berlin, Heidelberg.

[27] Staab, S., Studer, R., Schnurr, H. P., & Sure, Y. (2001). Knowledge processes and ontologies. IEEE Intelligent systems, 16(1), 26-34.

[28] Fernández-López, M., Gómez-Pérez, A., & Juristo, N. (1997). Methontology: from ontological art towards ontological engineering.

[29] Noy, N. F., & McGuinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology.

[30] Suárez-Figueroa, M. C., Gómez-Pérez, A., & Villazón-Terrazas, B. (2009, November). How to write and use the ontology requirements specification document. In OTM Confederated International Conferences" On the Move to Meaningful Internet Systems" (pp. 966-982). Springer, Berlin, Heidelberg.

[31] Musen, M. A. (2015). The protégé project: a look back and a look forward. AI matters, 1(4), 4-12.

[32] Motik, B., Grau, B. C., Horrocks, I., Wu, Z., Fokoue, A., & Lutz, C. (2009). OWL 2 web ontology language profiles. W3C recommendation, 27, 61.

[33] Kifer, M., & Boley, H. (2013). RIF overview. W3C working draft, W3C,(October 2009). http://www. w3. org/TR/rif-overview.

[34] Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 652-660).

[35] Liu, Z., Tang, H., Lin, Y., & Han, S. (2019). Point-Voxel CNN for efficient 3D deep learning. In Advances in Neural Information Processing Systems (pp. 965-975).

[36] Pham, Q. H., Nguyen, T., Hua, B. S., Roig, G., & Yeung, S. K. (2019). JSIS3D: joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 8827-8836).

[37] Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., & Solomon, J. M. (2019). Dynamic graph cnn for learning on point clouds. Acm Transactions On Graphics (tog), 38(5), 1-12.

[38] Tchapmi, L., Choy, C., Armeni, I., Gwak, J., & Savarese, S. (2017, October). Segcloud: Semantic segmentation of 3d point clouds. In 2017 international conference on 3D vision (3DV) (pp. 537-547). IEEE.