

**Semantic and emotion reasoning tools for 3D Spaces adaptation v2**

H2020- 825079

|  |  |
| --- | --- |
| **Dissemination level:** | Public |
| **Contractual date of delivery:** | Month 39, 31/3/2022 |
| **Actual date of delivery:** | Month 43, 18/7/2022 |
| **Workpackage:** | WP5 - MindSpaces adaptive environment development |
| **Tasks:** | T5.3 Semantic representation and data integration  T5.4 Semantic reasoning for emotion-based space adaptation  Τ5.6 Development of semantically enhanced interactive 3D spaces |
| **Type:** | Report |
| **Approval Status:** | Approved |
| **Version:** | 1.0 |
| **Number of pages:** | 89 |
| **Filename:** | d5.4\_mindspaces\_v1.0 |
| **Abstract**  The current deliverable includes the latest updates on the semantic representation framework for capturing the relations of the MindSpaces concepts as ontological models. Furthermore, it analyses the functions and services of the semantic integration and reasoning services for the last version of the project. The updated technological background, as well as the updated related work, are presented along with the relevant technical and user requirements. Moreover, the latest version of the ontology is analysed and presented along with literature and state-of-the-art review. Finally, the latest reasoning mechanisms are presented along with the latest updates on the semantically enhanced interactive 3D spaces. | |
| 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. | |

|  |
| --- |
| eu-flag |

co-funded by the European Union

History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Reason** | **Revised by** |
| 0.1 | 25/2/2022 | ToC | CERTH |
| 0.2 | 30/4/2022 | Updates on the Semantic Framework | CERTH |
| 0.8 | 8/7/2022 | Integration of updated T5.6 input and circulation of draft for comments | CERTH |
| 0.9 | 15/7/2022 | Internal review comments | McNeel |
| 1.0 | 18/7/2022 | Finalisation of the final draft | CERTH |

**Author list**

|  |  |  |
| --- | --- | --- |
| **Organization** | **Name** | **Contact Information** |
| CERTH | Stefanos Vrochidis | stefanos@iti.gr |
| CERTH | Evangelos Stathopoulos | estathop@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 |
| up2metric | Anastasios Papanagnou | anastasis.papanagnou@up2metric.com |

**Executive Summary**

This deliverable concludes the final efforts during the MindSpaces project conducted within tasks T5.3, T5.4 and T5.6. Such efforts include the latest updates and final form of the MindSpaces Ontologies and vocabularies, the formal representation and interlinkage of incoming data to ontological structures upon integration, the final implementation of the reasoning mechanisms towards space adaptation, and the latest updates on semantically enhanced interactive 3D spaces.

In more detail, this document includes the **final version** of the MindSpaces ontologies, and the methodologies followed to complete this task. An updated ontology requirement specification document is also included and presented. An updated state-of-the-art analysis combined with other analysis concluded in the final version of the ontology, describing concepts and content such as physiological, virtual, behavioural, textual, VR experiments and so on. With integration at a complete stage, the information incoming concluded into the final state of the knowledge graph of the project.

Furthermore, the **final version** of the reasoning mechanisms are 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 to address special user requirements.

In addition, processes were finalized using the Grasshopper tool inside Rhino Software to alter the geometry of the 3D reconstructed models of reality. In parallel, workflows were also finalized, to generate 3D CAD models from the mesh models. Towards the latest automation of this process algorithms for semantic segmentation of point clouds are updated and presented.

Concluding, the efforts presented in this deliverable are the final version, version 3, of MindSpaces semantic representation, data integration, semantic reasoning towards space adaptation, and the space adaptation techniques and assets themselves.

**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 1** | Tbox & Abox axioms |
| **Table 2** | User requirements translated to technical requirements |
| **Table 3** | Ontology Requirements |
| **Table 4** | Classes of the ontology scheme for PUC1 |
| **Table 5** | Object properties of the ontology scheme of PUC1 |
| **Table 6** | Datatype properties of the ontology scheme for PUC1 |
| **Table 7** | Classes of the ontology scheme for PUC2 |
| **Table 8** | Object properties of the ontology scheme for PUC2 |
| **Table 9** | Datatype properties of the ontology scheme for PUC2 |
| **Table 10** | Results of PointNet on s3dis dataset, using the metric of IoU |
| **Table 11** | Results of PVCNN on s3dis dataset. |
| **Table 12** | Results of PointNet on s3dis dataset, using the metric of accuracy |
| **Table 13** | Results of modified PVCNN on McNeel room dataset, using the metric of accuracy |
| **Table 14** | Confusion matrix of predicted (rows) and ground truth (columns) labels |
| **Table 15** | 2D image semantic segmentation results to automatically identify the sky in images and mask it when using images for 3D reconstruction. |
| **Table 16** | Example of application of semantic segmentation for sky removal during depth prediction. |

**Table of Figures**

|  |  |
| --- | --- |
| **Figure 1** | Logical flow of WP5-related WPs |
| **Figure 2** | Diagram of affiliated tasks in WP5 |
| **Figure 3** | Core BCI Interaction Model |
| **Figure 4** | The patterns of F, (a) participation, (b) mereology, (c) causality, (d) correlation, (e) documentation, (f) interpretation, (g) example of applying the F ontology |
| **Figure 5** | Core concepts of PROV-O |
| **Figure 6** | DCMI Core elements |
| **Figure 7** | Part of the GoodRelations Ontology Scheme |
| **Figure 8** | Part of the VirtualHome Ontology Scheme |
| **Figure 9** | A subset of ATOMIC |
| **Figure 10** | ConceptNet browsable, shows facts about the English word “bicycle |
| **Figure 11** | Urban redesign of Karlsruhe square |
| **Figure 12** | Ontology Scheme for PUC1 |
| **Figure 13** | Ontology scheme with object and datatype properties |
| **Figure 14** | Workspace redesign based on people behaviour patterns and spatial needs |
| **Figure 15** | Ontology Scheme for PUC2 |
| **Figure 16** | Ontology scheme with object and datatype properties |
| **Figure 17** | Overview of Grasshopper’s canvas for modifying the geometry of 3D mesh models |
| **Figure 18** | Workflow for modifying 3D mesh geometry via Grasshopper in Rhino |
| **Figure 19** | Final textured 3D mesh model of PUC1 (Tecla Sala) |
| **Figure 20** | Final rich 3D CAD model of PUC1 (Tecla Sala) |
| **Figure 21** | Original photogrammetric 3D mesh model of Thessaloniki Music Hall generated in T4.1 (surface) |
| **Figure 22** | Model improvement and completion in Rhino 3D |
| **Figure 23** | Final 3D CAD Model as seen in Virtual Reality in Unity 3D |
| **Figure 24** | Initial 3D mesh model from T.4.1 (left). Final rendered 3D CAD Model (right) |
| **Figure 25** | Initial 3D mesh model from T.4.1 (top). Final rendered 3D CAD Model (bottom) |
| **Figure 26** | 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 |
| **Figure 27** | PVConv is composed of a low-resolution voxel-based branch and a high-resolution point based branch. The voxel-based branch extracts coarse-grained neighbourhood information, which is supplemented by the fine-grained individual point features extracted from the point-based branch |
| **Figure 28** | 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 |
| **Figure 29** | Sample scenes of s3dis dataset |
| **Figure 30** | Sample scenes of ScanNet dataset |
| **Figure 31** | Sample groundtruth meshes of SceneNΝ |
| **Figure 32** | Format of S3dis dataset |
| **Figure 33** | Processing of our dataset |
| **Figure 34** | Semantic components of our dataset (a) floor, (b) floor and walls, (c) floor, walls and ceiling, (d) floor, walls, ceiling and sitting |
| **Figure 35** | Synthetic views of the enhanced 3D CAD model of McNeel office (left). Semantic classes automatically assigned by the properties of the 3D components of the 3D CAD model |
| **Figure 36** | Preliminary point cloud semantic segmentation on a McNeel room from modified PVCNN. Ground truth (left) vs inference results (right) |
| **Figure 37** | Failures of the inference results. Ground truth vs inference results |

**Table of Contents**

[**INTRODUCTION**](#_heading=h.73evllaihgjh)12

[**SEMANTIC REPRESENTATION & REASONING**](#_heading=h.2q5vjuwlrv6n)15

[Background and Related Work](#_heading=h.y921qm21ggk8) 15

[Reasoning and Data Integration](#_heading=h.a0jpzhh3frdr) 17

[Description Logics Reasoning](#_heading=h.ykwwqk8ioow6) 17

[Description Logics Reasoning Services](#_heading=h.k7my5ase5vr0) 18

[Rules](#_heading=h.7ecp50e26kk0) 18

[Data Integration](#_heading=h.upspr2v2ql2k) 19

[Ontologies Related to MindSpaces](#_heading=h.lrtnb8zg9l7o) 20

[BCI-O](#_heading=h.reme4avr1o7w) 20

[Event Model F](#_heading=h.f7z8e494rbtm) 21

[PROV-O](#_heading=h.966c8qtuzbhb) 22

[DCMI](#_heading=h.e51unr6aizm8) 23

[GoodRelations](#_heading=h.duejw05dkcg4) 23

[O-PrO](#_heading=h.bzodg9gl00ms) 24

[VirtualHome Ontology](#_heading=h.d64k25iqlqs0) 24

[DBpedia Ontology](#_heading=h.comd80pr6rkz) 26

[Knowledge Graphs Related to MindSpaces](#_heading=h.7gv45enrjsge) 25

[Atomic](#_heading=h.comd80pr6rkz) 26

[BabelNet](#_heading=h.fd752rneuy5f) 28

[WordNet](#_heading=h.fd752rneuy5f) 28

[ConceptNet](#_heading=h.fd752rneuy5f) 28

[YAGO](#_heading=h.nug3rcu6xqg0) 29

[WebChild](#_heading=h.nug3rcu6xqg0) 29

[Modelling and Reasoning Requirements](#_heading=h.c3ezptm4sz9c) 29

[Ontology Modelling](#_heading=h.1wzdh1sh1nt) 29

[Relevant User & Technical Requirements](#_heading=h.qzjau8emnwh) 30

[Ontology Requirements](#_heading=h.1vvz86gnu9al) 30

[**OUTDOORS URBAN ENVIRONMENT (PUC 1)**](#_heading=h.7xd4le1p271n)35

[Motivation and Description of PUC1](#_heading=h.i2xyt6915bi5) 35

[Knowledge Representation for Outdoor Urban Environments](#_heading=h.er8v26ibabhf) 36

[Multimodal Fusion for Outdoor Urban Environments Use Case](#_heading=h.r9xsc2dibq4y) 44

[Reasoning for Outdoor Urban Environments Use Case](#_heading=h.c9qqdf9u1e8i) 47

[Discussion and Future Work](#_heading=h.yci6enabx8wu) 51

[**INSPIRING WORKSPACES (PUC 2)**](#_heading=h.fb9ml4ipqg5z)52

[Motivation and Description of PUC2](#_heading=h.mk8mcmf1ondm) 52

[Knowledge Representation for Inspiring Workspaces Use Case](#_heading=h.9c3dae81ekpi) 53

[Multimodal Fusion for Inspiring Workspaces Use Case](#_heading=h.mbz8zko14opb) 58

[Reasoning for Inspiring Workspaces Use Case](#_heading=h.wn8sjadhgzmf) 60

[Discussion and Future Work](#_heading=h.gkzy9ucif28x) 61

[**SEMANTICALLY ENHANCED INTERACTIVE 3D SPACES**](#_heading=h.1w0sdc2pmd4o)62

[Modification of reconstructed 3D geometry in Rhino Grasshopper](#_heading=h.yxxh7pe66g2s) 62

[3D reconstructed models to 3D CAD models](#_heading=h.j7hlaxz6m8vn) 64

[PUC 1 - Outdoors urban environments (Tecla Sala & Thessaloniki Music Hall)](#_heading=h.z2svh1an9ag) 65

[PUC 2 - Inspiring workplaces (McNeel Office Space)](#_heading=h.ly0dee423rxa) 68

[PUC 3 - Emotionally-sensitive functional interior design (Senior’s residence in Paris)](#_heading=h.9yamyptiphpi) 69

[3D point cloud Semantic Segmentation](#_heading=h.7zgz2q8nrmm5) 70

[State of the Art Semantic Segmentation](#_heading=h.hsdf0tjnujmo) 70

[State of the Art Datasets](#_heading=h.k34kydcxxqf3) 76

[S3DIS Building Parser](#_heading=h.z27phxqni16s) 76

[ScanNet](#_heading=h.b6rjzv86skwh) 78

[SceneNN](#_heading=h.m569wz8ev8uf) 77

[The project’s dataset](#_heading=h.6cibqwnvkj8w) 78

[Results on McNeel Dataset](#_heading=h.g7oy8fl2abeh) 80

[2D Image Semantic Segmentation to assist Single Image Depth Prediction](#_heading=h.n0wzahecgha) 82

[**REFERENCES**](#_heading=h.yfql1pz8aaq6)86

# INTRODUCTION

Within the various scopes of WP5, one important goal of affiliated tasks is to deliver a customised framework for knowledge modelling and fusion by integrating heterogeneous multimodal information relevant to the MindSpaces domain (T5.3), custom logic procedures, and algorithms based on emotions derived from multiple sources to achieve pertinent spatial adaptations (T5.4), where the generated 3D spaces follow a semantically augmented and interactive approach (T5.6). On the other hand, one of the core purposes of WP5 is to construct a reasoning mechanism (T5.4) that will return the desired information, with respect to the use case (i.e., PUC1, PUC2), to the end-user and will provide the necessary information to the text generation component, to produce textual explanations (T5.5). Moreover, WP5 includes, among other technical solutions, knowledge forms and hierarchies in conjunction with corresponding specialised vocabularies, known as ontologies, to appropriately capture and express the overall semantics of:

⦁ A project, per artist, and peruse case, along with details and metadata for appropriate retrieval, archiving and management purposes.

⦁ A human subject that experiences the virtual reality environment.

⦁ A user of the design tool is 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 colour palettes of the aesthetics extraction module along with pertinent emotions and various useful metadata.

⦁ The 3D objects and their types 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.

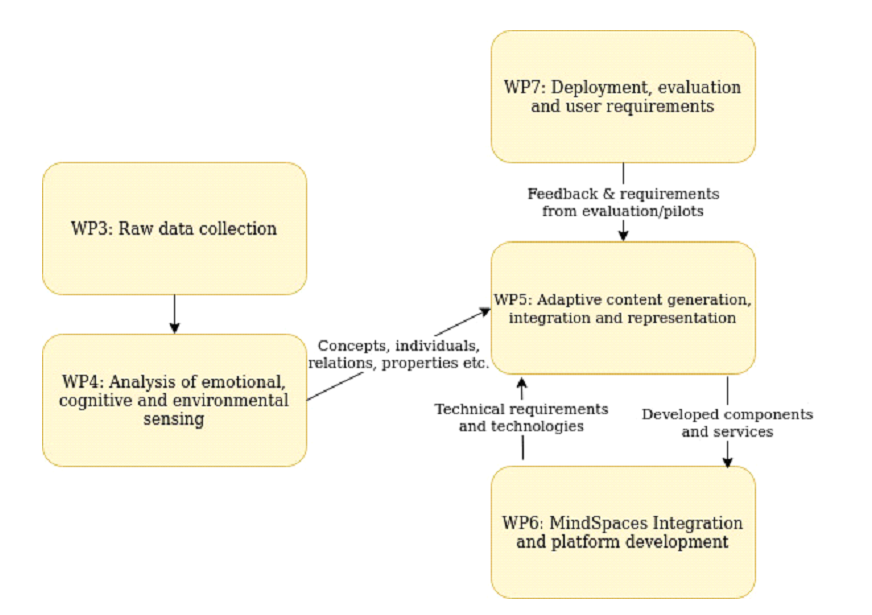
⦁ The concept of a hotspot regarding stress points both inside virtual environments and real environments is captured from depth cameras.

⦁ The results derive from textual analysis such as “aspects” and “sentiments” and how they are perceived by the system.

⦁ The results of the behavioural 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.

Figure 1 depicts the direct and indirect workflow dependency across MindSpaces' WP5 and other WPs, where WP3 oversees sensor data gathering, which is then transmitted to the WP4 analysis module. Following the perpetual evaluation and evolutionary requirements, Figure 1 also includes WP6 and WP7, which correlate with WP5 in terms of system platform integration and user feedback, respectively.

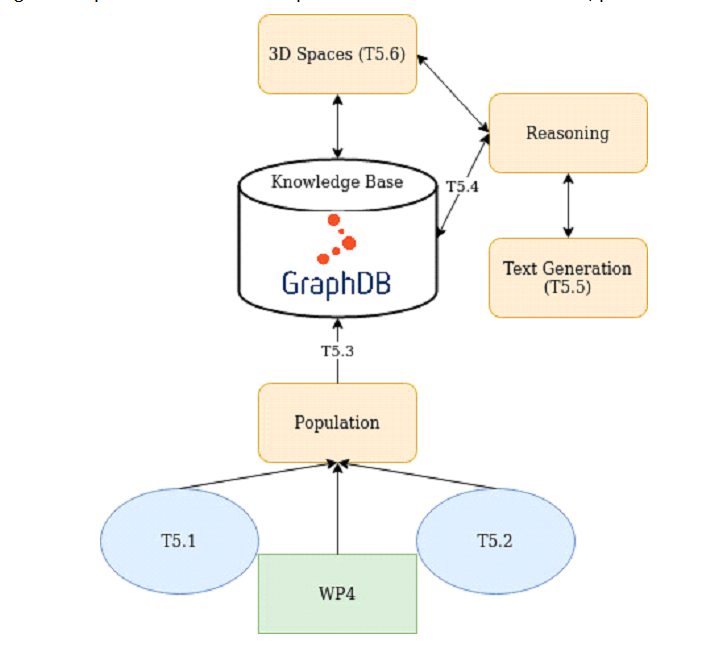


**Figure 1:** Logical flow of WP5-related WPs

One of T5.3's main goals is to import, reuse, and potentially extend existing ontological frameworks by fusing relevant divergent vocabularies (or parts of such vocabularies) and sub joining additional targeted custom-made concepts into a single unified model that addresses MindSpaces' knowledge representation and management needs. OWL 2 [1], the W3C recommendation for establishing and exchanging ontologies, is the preferred semantic language for defining and sharing the aforementioned vocabularies.

Apart from modelling the project's fundamental desired knowledge, it was regarded as critical to populate the knowledge base (KB) in accordance with the customised ontological structures that had already been created and were being iteratively updated. This is accomplished automatically by mapping multimodal information spawned by other platform modules in the form of analysis findings; the semantic component does not interact with raw data at all due to a lack of relevant semantic exploitation options. As a result, WP5 builds and expands the basic RDF-based knowledge graph that characterises the project by implementing both automated methods and Application Programming Interfaces (APIs) for knowledge fusion of changing incoming information. The emotional state tags generated from EEG techniques, for example, are coupled with relative localization coordinates in WP5 to expand the network with more individuals and relationships.

WP5 tasks also include the reasoning tool, a framework responsible for meeting all WP5-related requirements for reasoning procedures, both user and technical. Classes, class hierarchies, elaborate descriptions, and axiomatized properties, for example, are all viable examples of general axiomatizations that rely on ontological structures and concepts. Rules are implemented on top of the coherent knowledge to infer and deduce new knowledge and recommend appropriate actions. The rules are either pre-programmed and imported directly from cutting-edge technology stacks, or they are custom-made to meet specific requirements. In addition, the purpose of the reasoning mechanism is to provide the necessary information to the text generation, in order for the text generation mechanism to provide explanations to the end-user about a decision. The overall approaches, which combine all of the above components, aim to uncover underlying implicit information from well-established knowledge so that it can be incorporated into the knowledge graph and enhanced by future expansion. Beyond standard methods, such as keyword matching, a semantically dense and well-defined network will enable a more semantically enriched retrieval strategy of items, enhancing overall comprehension of concepts. Figure 2 depicts the architectonics relevant to this document task, part of WP5.



**Figure 2:** Diagram of affiliated tasks in WP5

The rest of this document is organised as follows. In Section 2, we include essential background information as well as related work. The section also contains the modelling and reasoning needs that drove the development of the MindSpaces ontology. Next in Section 3, we describe the ontology schema, the multimodal fusion mechanism, the reasoning module all needed for PUC1, and a discussion over the results. Section 4 contains the description of the ontology schema and the multimodal fusion for PUC2, and a discussion over the results. Section 5 contains the results of task T5.6 which include solutions, workflows and algorithms for the generation of semantically enhanced interactive models from the 3D reconstructions of urban and indoors spaces that are generated in MindSpaces in the context of WP4.

# SEMANTIC REPRESENTATION & REASONING

This chapter includes a thorough review of the background knowledge and current state-of-the-artwork in the areas of knowledge representation languages and frameworks, reasoning procedures, and how multimodal heterogeneous data is integrated into the system, as well as official existing vocabularies and ontologies relevant to the MindSpaces project (this is presented in subsection 2.1). Furthermore, the essential modelling and reasoning requirements, as well as competency questions, are described. The current version of the project's ontology, which enables knowledge fusion during the generation of the knowledge base using automated methodologies is also presented in subsection 2.2.

## Background and Related Work

Initially, it seems appropriate to begin our Background Section by outlining the principles of Description Logic (DL) Languages [2], which serve as the foundation for the official World Wide Web Consortium (W3C) proposal for ontology production and distribution on the Web. The foundations for the Web Ontology Language (OWL) [3] were laid by this W3C agreement in 2004, with an updated recommendation for a considerably improved version known as OWL 2 [4] followed in 2009. Aside from the standards of OWL 2, there is a variety of OWL 2 genres and rule-based languages.

Apart from explicitly defining and hierarchizing the domain knowledge and relations relevant to MindSpaces, reasoning services derived from the DL's expressiveness capabilities, are explicitly presented to show the potential of custom smart algorithms attempting to infer additional knowledge, from existing knowledge and performing actions automatically when certain rules are triggered.

Investigating automatic methods to retrieve heterogeneous multimodal information efficiently and sufficiently semantify such pieces of information and populating them into a knowledge base using an annotated unified data schema is a difficult task that goes hand in hand with the design and implementation of ontological models and reasoning algorithms. The data integration process necessitates the use of semantically significant information rather than just raw data, which is accomplished by carefully structuring ontology mapping techniques. As a result, the pipeline begins with raw data collecting and ends with the development of Resource Description Framework (RDF) [5] triplets. Many triplets may implicitly interconnect with each other by explicitly following the subject, predicate, and object rules, thus densifying the knowledge graph. Nodes represent entities, edge labels reflect types of relations, and edges themselves correspond to the existence of relationships among nodes in such evolutionary networks.

**Web Ontology Language**: By looking into the related work in the area of knowledge representation, it can be concluded that ontologies have very broad use, which stems from their effectiveness in structuring various domains of knowledge. Ontologies can represent the knowledge about a domain in a manually organised fashion, with accurate modelling of each concept, and combine any modality of unstructured information, abstract context, or interrelationships across concepts. Furthermore, by fusing, pieces of existing ontological models into an innovative semantic structure to satisfy specified demands, brings a more fine-grained representation of the domain.

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.

When an ontology data model is applied to a set of such individual facts, a knowledge graph is created, which is a collection of entities with nodes and edges between them expressing the types and relationships between them. The ontology lays the foundations for the knowledge graph to capture the data in a domain by formally specifying the structure of the knowledge in that domain.

Other methods such as vocabularies, thesauri, topic maps, taxonomies, and logical models are also available in the competition about formal specifications for knowledge representation. In contrast to taxonomies or relational database schemas, ontologies, for example, express relationships and enable the linking of many concepts to other concepts in a variety of ways.

The Semantic Web's goal is to make internet data machine-readable, therefore the shareability and reusability of the aforementioned methodologies are desired outcomes. The fundamental ideas of DL languages, as well as several categories of OWL and coherent rule-based languages, are discussed in subsection 2.1.

**OWL & OWL 2**: The OWL language is a knowledge representation language designed to meet the needs of the Semantic Web in terms of ontology authoring, design, and generation. The Description Logics have had a significant impact on the core aspects of OWL. 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 was created to fill a gap in the market for classification hierarchies and explicitly stated restrictions. For example, it only accepts cardinality values that are discretely between zero and one. All OWL language constructs are included in OWL DL, however, they are subject to numerous limitations. OWL Full uses entirely different semantic schemas than the aforementioned frameworks, and it was created with RDF Schema compatibility in mind. For example, in this language, a class can be thought of as both a collection of instantiated individuals and an independent entity. Additionally, increasing the lexicon allows for ontological augmentations. One of the disadvantages of the latter is that it is undecidable, making it impossible for a reasoning framework to execute thorough reasoning on it. Another is that OWL modelling is based on a tree-like approach, which prevents a more expressive approach to supporting real-world problems and applications. The World Wide Web working group was formed to gather, generate, and introduce OWL 2, which included qualified cardinality requirements, property chains with regularity limits to avoid logical cycles, and other features.

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

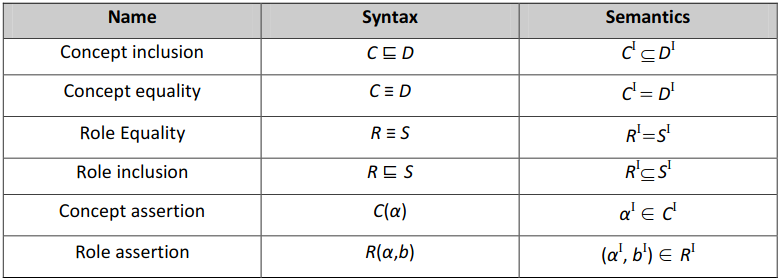
* OWL 2 QL: was formed to facilitate access and querying inside knowledge bases serving as data storage,
* OWL 2 EL: maintaining a polynomial-time reasoning complexity,
* OWL 2 RL: a rule subset of OWL 2.

Finally, different kinds of syntaxes, addressing different needs, can be grouped into two categories: the class that includes all high-level syntaxes (OWL abstract syntax, OWL 2 functional syntax) that target specifications, and the class that includes all exchange syntaxes (RDF syntax, OWL 2 XML Syntax, Manchester Syntax) that facilitate data transfer and ease of use.

### Reasoning and Data Integration

#### Description Logics Reasoning

Description Logics (DL) are a group of knowledge representation formalisms that represent domain knowledge by first defining its language and then using it to specify the domain's features and describe its environment. Terminological knowledge representation languages, concept languages, term subsumption languages, and KL-ONE-based knowledge representation languages were their previous names. They are distinguished by the fact that they are built on formal, logic-based semantics and provide reasoning as a fundamental function. Concepts, roles, and individuals are the fundamental parts of their foundation. For example, constructors can be used to combine a concept representing a set of objects of similar nature (e.g. Dog), a role that represents the semantic link among objects (e.g. barksAt), and a person that is essentially an instantiated concept (e.g. Fluffy). In this case, the notion pertains to the knowledge that an object from the class “Dog” is related in some semantic manner via the relationship “hasOwner”. The latest applies to all instances of “Dogs” which have owners. Behind this kind of logic, lie two types of knowledge: terminological knowledge (TBox) and assertional knowledge (ABox). The former describes through axioms how the objects of a domain are related, whereas the latter contains axioms about instantiated individuals and how they are related to each other. For the TBox axiom  , 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.

****

**Table 1:** Tbox & Abox axioms

#### Description Logics Reasoning Services

Description Logics provide several useful services of reasoning, along with some algorithms that combined can fulfil 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 data types, 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 attempt to support the OWL 2 DL specification fully and correctly.
* 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 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), realisation (for a given individual y detect most specific named concepts  such that the instance (y,) hold for every i).

#### Rules

The existence of rules in reasoning facilitates the Description Logics' concluding decision-making. In order to achieve this, the Web Ontology Language sacrifices some levels of expressiveness in favour of more effective results. The decision-making tree structure is a good example of this. By working from the root to the leaves of such a structure, the desired finalisation of decisions is ensured at the cost of applied constraints on the variables and quantifiers' use. As a result, depicting a class whose instantiations are linked to an anonymous individual via many property pathways is impossible. To address such challenges, the domain's academic community has put a lot of effort into developing Web Ontology Language integration solutions with rules.

The construction of the Semantic Web Rule Language [13], where rules are articulated with first-order logic semantics, is one such endeavour. This method includes Horn-like rules in the set of OWL axioms, allowing Horn-like rules to be integrated with an OWL knowledge base in the form of implication between an antecedent and a consequent. Although promising, this technique had a negative influence on decidability, prompting much complementary research [14], [15], [16], which either applied limitations to rules, intersected with Description Logic Programs, or introduced Description Logic Safe rules. Other approaches, for example, mandate a mix of rules and ontologies based on mappings of an ontological subset of semantics on rule engines [17].

The SPARQL Protocol and RDF Query Language [18] are another more practical solution for storing, retrieving, and updating information for Resource Description Framework graphs. The World Wide Web Consortium created and recommends it as a declarative and expressive language for defining complex relationships between entities. Mathematical investigations have shown that the SPARQL algebra has the same expressivity as the relational algebra [19]. The CONSTRUCT graph pattern is a helpful element of SPARQL that goes beyond its role as a query language. By connecting nodes with edges, one can create SPARQL rules that generate unique RDF data, resulting in larger and more dense graphs. This type of query has a CONSTRUCT and a WHERE clause, where the triple pattern in the knowledge graph is defined in the CONSTRUCT field and superinduced only when the criteria defined in the where clause are met, implying successful pattern matching in the knowledge graph; in a sense, an ASK query returns true if a pattern exists. In addition, there is an additional framework called SPARQL Inferencing Notation [20] (SPIN) that makes it easier to comprehend and execute SPARQL rules on RDF graphs. SPIN allows you to store SPARQL queries as RDF triples alongside any RDF model, interlinking resources and queries in a graph and increasing interoperability and reusability.

#### Data Integration

Multiple components in the project are in charge of extracting unstructured data of various types, such as online content, physiological signals, and video camera feed. Additional components cater to the requests for further analysis on top of the raw data aggregation and their presence in the MindSpaces system. As a result, integrating such data and extracting, storing, and manipulating information is a difficult undertaking, necessitating exact ontological modelling and conceptualization of each existing entity. Intuitively, raw data appears to have little to no information, yet the analytical findings of raw data appear to be incredibly helpful and exploitable in terms of reasoning. As a result, there is no need to integrate raw data directly because the extra value of doing so is negligible.

These analysed data are united using specific mapping algorithms and then populated into the knowledge base, resulting in the creation and expansion of the knowledge graph. The mapping is based on the ontology design and how abstract concepts and qualities are related, whereas RDF is used to describe such things and connect concrete individuals, properties, and relationships to abstract classes in a semantic fashion. Finally, data integration is undertaken to aid reasoning techniques that are based on knowledge fusion and custom inferences drawn from uninterpreted raw data that appears to be unimportant.

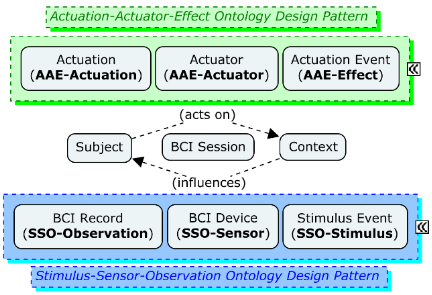
### Ontologies Related to MindSpaces

This section reviews previous work on state-of-the-art ontologies that are compatible with MindSpaces and can be imported and reused for modelling the project's basic concepts with few to no changes. The objective of this subsection is to demonstrate the essential principles that have been followed during the examination of related work involving the design and development of conceptual models, rather than to provide a comprehensive list of semantic frameworks pertinent to MindSpaces.

#### BCI-O

The brain-computer interaction ontology [21] describes a fundamental metadata model set for multimodal BCI data capture events in the real world. Its creation represents a conceptual framework that BCI applications can use to construct basic concepts that reflect a relevant and compatible metadata language. The Semantic Sensor Network Ontology (SSN) is a domain-independent and end-to-end paradigm for sensor and actuator applications, and BCI-O is aligned to it. As a result, its structure has been normalised to make it easier to use in conjunction with other ontologies of linked data resources to explain in detail any specific definitions that BCI applications would require, such as time and time series, position, mobility, units of measurement, and so on. Its specification includes generic alignment data modelling recommendations for essential concepts to aid in the development of BCI applications.

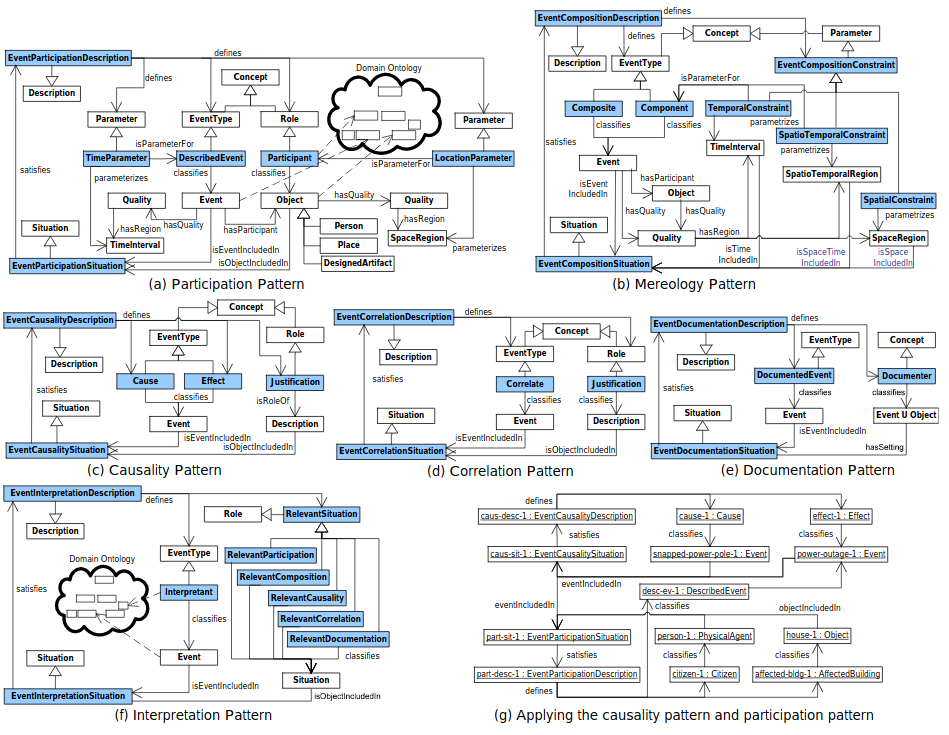
There was no complete and standardised formal semantic construction prior to BCI-O to model the BCI metadata annotations, which are detected signals, in order to classify the study of brain states and dynamics in diverse situations. For BCI data capture activities, Figure 3 depicts the integration of a sense model (context to the subject, based on the SSO ODP and aligned to SOSA/SSN) and an actuation model (subject to context, based on the AAE ODP and aligned to SOSA and SAN/IoT-O).



**Figure 3:** Core BCI Interaction Model

#### Event Model F

The Event-Model-F [22] approach to modelling events is a formal approach that was created to improve interoperability in distributed event-based systems. It incorporates the DOLCE+DnS Ultralite (DUL) ontology [23] and offers significant support for modelling space, objects, people, time and correlative, participation, documentation, causal, interpretation, and mereological links between events (see Figure 4 for examples and more). It also gives you a versatile approach to composing events, describing event causality and connection, and representing distinct features of the same event. It is built in accordance with DUL's pattern-oriented philosophy and is modularized into separate ontologies that may be easily extended by domain-specific ontologies.



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

#### PROV-O

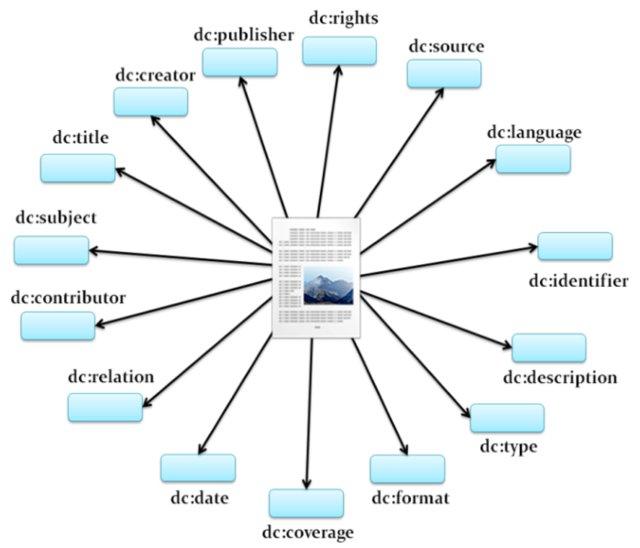
The PROV Ontology specifies the PROV Data Model Web Ontology Language (OWL 2) encoding [24]. It encompasses all of the classes, features, and constraints that define the foundations for developing provenance applications in diverse domains by describing, swapping, and integrating provenance information created in multiple systems and situations. Furthermore, it can be specialised for modelling application-specific provenance details in a variety of domains, allowing it to be used directly or expanded to meet demand, allowing for interoperability. The agent that can be assigned to an entity or linked with an activity, the entity that can be formed by an activity, and the activity itself are the key notions of the PROV ontology (check Figure 5 for more details).



**Figure 5:** Core concepts of PROV-O

#### DCMI

There are 15 metadata items in the Dublin Core Metadata Initiative [25]. (see Figure 6). Those elements can be used to describe both multimodal digital and physical resources, such as movies, photos, and so on. The vocabulary can be used to define basic resources or to mix metadata vocabularies from several standards, ensuring interoperability in linked data and semantic web applications.



**Figure 6:** DCMI Core elements

#### GoodRelations

The GoodRelations ontology [26] is a lightweight ontology for exchanging e-commerce information, namely data about products, offers, points of sale, prices, terms, and conditions, on the Web. It can be used in all RDF syntaxes (like RDF/XML, Turtle, RDFa, JSON-LD, among others), Microdata, and any syntax that supports an Entity-Attribute-Value pattern. GoodRelations started as an independent Web Ontology in 2007, and since 2012, it has been almost fully integrated into schema.org and is now the official e-commerce data model of this initiative. GoodRelations remains an independent project and will remain the official version of the conceptual model. While a majority of GoodRelations data published on the Web will be in the schema.org namespace, the original version will be the default way of handling respective data in a full RDF/SPARQL/OWL environment.

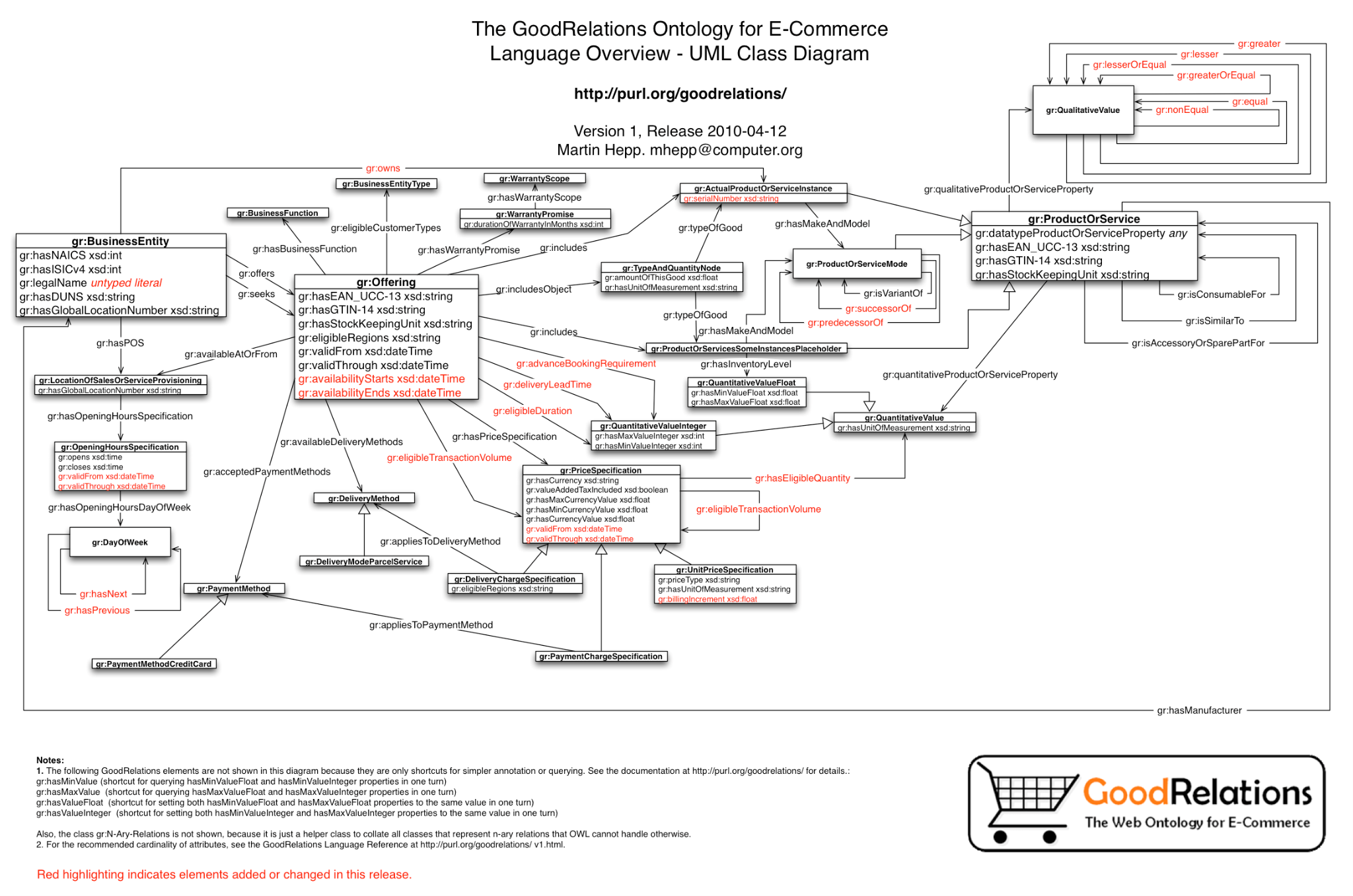
GoodRelations provides a standard vocabulary for expressing things like

• that a particular Website describes an offer to sell cell phones of a certain make and model at a certain price,

• that a piano house offers maintenance for pianos that weigh less than 150 kg,

• or that a car rental company leases out cars of a certain make and model from a particular set of branches across the country.

Also, most if not all commercial and functional details of e-commerce scenarios can be expressed, e.g. eligible countries, payment and delivery options, quantity discounts, opening hours, etc. Figure 7 shows part of the Good Relations scheme. Other product ontologies are [27,28], and [29].



**Figure 7:** Part of the GoodRelations Ontology Scheme

#### O-PrO

Object affordances, namely actions that an object allows to be performed on/with it (e.g. cut can be performed by knife), provide useful information related to the understanding of human activities. The aim of the O-PrO (Object Property Ontology) ontology [30], is to provide reasoning for object affordances that can be shared across different assistive robots operating within the household domain. O-PrO is a novel ontology consisting of 61 household objects. The ontology can be used for computing cognitive and semantic object affordances.

#### VirtualHome Ontology

VirtualHome Ontology [31], can be used by any cognitive robotic system that acts in a household environment, in order to give information and instructions about the environment and the objects in it, as well as on how to perform activities (e.g. how to make a sandwich). The most common queries that a user can address to a robotic system have been selected through an extensive literature overview on the topic of household cognitive robotics [32]. The VirtualHome Ontology was constructed based on the VirtualHome dataset [33,34]. For instance, a human user (target users can be elderly people in the early stages of dementia) can address the cognitive robotic system queries such as “What can I do with object X?”, “Which other objects are related to object X?”, or “Can I do the activity Y with the objects X and Z?”, among others, in order to help them in a household environment.



**Figure 8:** Part of the VirtualHome Ontology scheme.

#### DBpedia Ontology

DBpedia Ontology [35] is a community effort to extract structured information from Wikipedia and to make this information available on the Web. DBpedia allows sophisticated queries against datasets derived from Wikipedia and to link other datasets on the Web to Wikipedia data. DBpedia contains various datasets, with the resulting information published on the Web for human- and machine consumption. The DBpedia Ontology is constructed and enhanced with the information provided by the DBpedia community. Moreover, website authors can facilitate DBpedia content within their sites. Finally, the DBpedia Ontology has an interlinking status with other open datasets on the Web.

### Knowledge Graphs Related to MindSpaces

In mathematics, graph theory is the study of graphs, where a graph is considered a set of nodes and edges which connect the nodes. There are numerous theorems, definitions, and properties that can be applied to graphs, with maybe the most well-known separation of directed and undirected graphs. In an undirected graph, edges link two nodes symmetrically, in directed graphs, edges link two nodes asymmetrically. Based on graph theory, KGs emerged. KGs are graphs that contain commonsense knowledge and constitute one of the most promising areas of AI [36]. KGs have found applications in many areas such as in cognitive robotics, knowledge representation, argumentation, and helping data-driven models to solve various tasks, among others.

#### Atomic

ATOMIC [37] is a KG containing generic causal relations. More specifically, ATOMIC is an atlas of everyday commonsense reasoning, organised through 877k textual descriptions of inferential knowledge. Compared to existing resources that centre around taxonomic knowledge, ATOMIC focuses on inferential knowledge organised as typed if-then relations with variables (e.g., “*Person cut the cucumber with a knife* → *the cucumber is in pieces*”). The ATOMIC KG proposes nine if-then relation types to distinguish causes vs. effects, agents vs. themes, voluntary vs. involuntary events, and actions vs. mental states (see Figure 8). Moreover, ATOMIC can show that neural models can acquire simple commonsense capabilities and reason about previously unseen events. By generativity training on the rich inferential knowledge described in ATOMIC. Experimental results demonstrate that multi-task models that incorporate the hierarchical structure of if-then relation types lead to more accurate inference compared to models trained in isolation, as measured by both automatic and human evaluation.

#### BabelNet

BabelNet [38] is an innovative multilingual encyclopaedic dictionary, with wide lexicographic and encyclopaedic coverage of terms, and a semantic network/ontology which connects concepts and named entities in a very large network of semantic relations, made up of about 20 million entries. BabelNet follows the WordNet [39] model based on the notion of synset (for synonym set) but extends it to contain multilingual lexicalizations. Each BabelNet synset represents a given meaning and contains all the synonyms which express that meaning in a range of different languages.



**Figure 9:** A subset of ATOMIC

#### WordNet

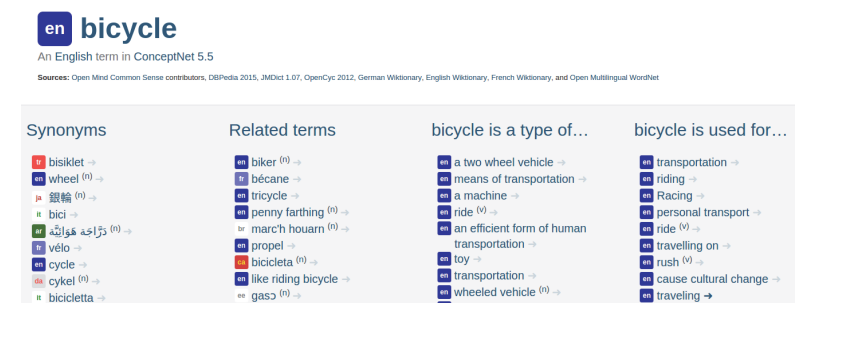
WordNet [39] is a large lexical database of English. Nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. WordNet's structure makes it a useful tool for computational linguistics and natural language processing.

WordNet superficially resembles a thesaurus, in that it groups words together based on their meanings. However, there are some important distinctions. First, WordNet interlinks not just word forms—strings of letters—but specific senses of words. As a result, words that are found near one another in the network are semantically disambiguated. Second, WordNet labels the semantic relations among words, whereas the groupings of words in a thesaurus do not follow any explicit pattern other than meaning similarity.

#### ConceptNet

ConceptNet [40] is a knowledge graph that connects words and phrases of natural language with labelled edges. Its knowledge is collected from many sources that include expert-created resources, crowdsourcing, and games with a purpose. It is designed to represent the general knowledge involved in understanding language and improving natural language applications by allowing the application to better understand the meanings behind the words people use.

When ConceptNet is combined with word embeddings acquired from distributional semantics, it provides applications with an understanding that they would not acquire from distributional semantics alone, nor from narrower resources such as WordNet or DBpedia. This with SotA results is demonstrated on intrinsic evaluations of word relatedness that translate into improvements on applications of word vectors, including solving satisfaction style analogies. Figure 9 shows an instance of the ConceptNet KG.



**Figure 10:** ConceptNet’s browsable [**interface**](https://conceptnet.io/), shows facts about the English word “bicycle”

#### YAGO

The YAGO knowledge base [41], was the first academic project to build a KB from Wikipedia, closely followed by the DBpedia project. The particular focus in YAGO has been on precision, i.e., on the correctness of the extracted facts. By sending the extracted facts through a sequence of filters, YAGO achieves a precision of 95%. Today, YAGO is a larger project at the [**Max Planck Institute**](https://www.mpg.de/institutes) and [**Tcom ParisTech University**](https://www.telecom-paris.fr/en/home). The knowledge graph draws on several sources by now, including WordNet and [**Geonames**](http://www.geonames.org/), and has grown to 16 million entities and more than 100 million facts. It is part of the Linked Open Data cloud.

#### WebChild

WebChild 2.0 [42] is another KG that contains commonsense knowledge. WebChild is built with a series of algorithms to distil fine-grained disambiguated commonsense knowledge from massive amounts of text. The WebChild 2.0 knowledge base is one of the largest commonsense knowledge bases available, describing over 2 million disambiguated concepts and activities, connected by over 18 million assertions. Among this data, there are 964.758 unique activity instances, grouped into 505.788 activity synsets (i.e., group of synonyms), obtain 581.438 location synsets, 71.346 time synsets, and 5.196.156 participant attribute entries over all activities.

One can argue that there are other important KGs such as Freebase [43] or OpenCyc [44] which is true, but we choose to present the most well-known and currently commonly used KGs.

## Modelling and Reasoning Requirements

In this subsection we describe how we constructed the ontologies for PUC1 and PUC2, and the multimodal fusion and reasoning mechanisms. For this reason, we present methods for the ontology development (subsection 2.2.1), user and Technical Requirements (TRs) (subsection 2.2.2), and the ontology requirements for the MindSpaces project.

### Ontology Modelling

There are several ways to express and model knowledge about a topic using ontological techniques and ontologies in general. It is a technique that cannot be completed all at once, and repetitions occur when the initial approach is iteratively evolved to better suit the application's requirements. United Process for Ontologies (UPON) [45], On-To-Knowledge [46], Methontology [47], and Ontology Development [48] are some of the techniques and processes that provide guidance for ontology engineering.

During the MindSpaces project, we used the Ontology Development approach to implement the project's ontological framework, which contains seven iterative steps [48]:

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.

In order to complete the first stage, they introduced the "Ontology Requirements Specification Document" (ORSD) in [49], which is a report that answers questions about the need for the new ontology creation, its intended purposes and users, and the requirements it must meet. The objective, scope, implementation language, intended uses and end-users, ontology requirements (functional and non-functional), and the Pre-Glossary of terms are all included in the ORSD (from competency questions, answers and objects).

### Relevant User & Technical Requirements

This section goes over the important user needs that have been developed up to this point and have led the development of the MindSpaces ontological and reasoning frameworks. Each scenario has been thoroughly researched in accordance with the User Requirements (URs) outlined in D7.2, which are then converted into technical requirements in D6.2 "Technical requirements and architecture." One can find a detailed analysis of the URs in D7.2, here in Table 2 we only present the translation of URs into TRs.

|  |  |
| --- | --- |
| **Technical Requirements** | **Related User Requirements** |
| Create a knowledge base that will fuse multimodal data and unify them into one ontological model | UR\_5, UR\_6, UR\_9, UR\_10, UR\_11, UR\_12, UR\_13, UR\_14, UR\_15, UR\_16, UR\_51, UR\_61, UR\_62, UR\_66 |
| 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, UR\_70, UR\_72, UR\_73, UR\_83 |

**Table 2:** User requirements translated to technical requirements

### Ontology Requirements

The "Ontology Requirements Specification Document" (ORSD), which served as a basic exercise to begin the ontological modelling, is presented in this subsection. It's an iterative process that can be expanded as the system's functions improve.

|  |  |
| --- | --- |
|  | **MindSpaces ORSD** |
| **1** | **Purpose** |
|  | The scope of the MindSpaces ontological framework is to describe and create the ontological formations and ontologies to represent 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 modelling aspects needed so as to support data modelling, integration and reasoning, including:  Formations for representing and encapsulating metadata of varying resources, such as physiological data, behavioural 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 behavioural 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 Palette's data derived from the aesthetics module encapsulate 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 (W3C) 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. |
| **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 analysing each PUC scenario and relevant user requirements. Competency Questions (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**  **Images:**  CQ1 What is the total imageability of the image?  CQ2 Which are the top 3 segmentation labels?  CQ3 Which is the top 1 verge label?  CQ4 Which is the unique identifier of the image?  CQ5 Which is the textual description of the image?  CQ6 Which is the caption of the image?  CQ7 What is the valence of the image?  CQ8 What is the arousal of the image?  CQ8 Which is the online source for a segmented label?  CQ9 What is the location of the image?  CQ10 Which are the verge label(s) in an image?  CQ11 Which are the segmentation label(s) in an image?  CQ12 Does the image contain a landmark?  CQ13 Which are the named entities contained in the image?  CQ14 Which is the category of the named entities contained in the image?  **Videos:**  CQ15 Which is the unique identifier of the video ?  CQ17 Which are the labels that accompany the video?  CQ18 What is the duration of the video?  CQ19 What are the frames per second of the video?  CQ20 Which is the file with associated metadata?  CQ21 Which are the verge label(s) in the video?  CQ22 Which are the segmentation label(s) in the video?  CQ23 Does the video contain a landmark?  CQ24 Which are the named entities contained in the video?  CQ25 What is the location of the image?  CQ26 Which are the top 3 segmentation labels?  CQ27 Which is the top 1 verge label?  **3D Models:**  CQ28 Which is the unique identifier of the 3D model?  CQ29 Which is the name of the 3D model?  CQ30 Which is the textual description of the 3D model?  CQ31 Which is the location of the 3D model in VR?  CQ32 Which is the official licence of the 3D model?  CQ33 Which is the online source of the 3D model?  CQ34 Which parameters are associated with the 3D model?  CQ35 Which is the file with associated metadata?  CQ36 Which are the verge label(s) in the 3D model?  CQ37 Which are the segmentation label(s) in the 3D model?  **Design Configurations:**  CQ38 Which is the unique identifier of the design configuration?  CQ39 What kind of changes does the design configuration withhold?  CQ40 What kind of objects does the design configuration withhold?  CQ41 What kind of parameters does the design configuration withhold?  CQ42 What kind of transformations does the design configuration withhold?  CQ43 Which hotspots are associated with the design configuration?  CQ44 Who is the creator of the design configuration?  CQ45 For which project is the design configuration intended?  CQ46 For which PUC is the design configuration intended?  **Hotspots:**  CQ47 Which is the unique identifier of the hotspot?  CQ48 Which is the localization of the hotpot?  CQ49 For which project is the hotspot intended?  CQ50 For which PUC is the hotspot intended?  CQ51 Who is the creator of the hotspot?  CQ52 Which hotspot provokes less stress?  CQ53 Which is the favourite hotspot of a subject?  CQ54 Which is the favourite hotspot of a subject, based on the type of subject?  **Textual Content:**  CQ55 Which is the sentiment of the text?  CQ56 Which is the confidence of the sentiment of the text?  CQ57 Which are the concepts of the text?  CQ58 Which is the frequency of the aspects of the text?  CQ59 Which is the source of the text?  CQ60 Which is the URL of the text (to be retrieved )?  CQ61 How many sentences does the text have?  CQ62 What is the text’s imageability?  **Physiological Content:**  CQ63 Which is the arousal of the subject?  CQ64 Which is the valence of the subject?  CQ65 Which is the emotional tag of the textual description?  CQ66 Which is the emotional state of the subject ?  CQ67 Which is the location of the subject?  CQ68 Which is the timestamp of observation?  CQ69 To which subject do these signals refer to?  CQ70 To which raw data do these signals refer to ?  CQ71 Which location provokes less stress?  CQ72 Which is the favourite location of a subject?  CQ73 Which is the favourite location of a subject, based on the type of subject? |

**Table 3:** Ontology Requirements

The first set of competency questions covers virtually every aspect of the MindSpaces domain. This list contains more information than what was originally modelled in the ontology, but it also lays the groundwork for future extensions.

# OUTDOORS URBAN ENVIRONMENT (PUC 1)

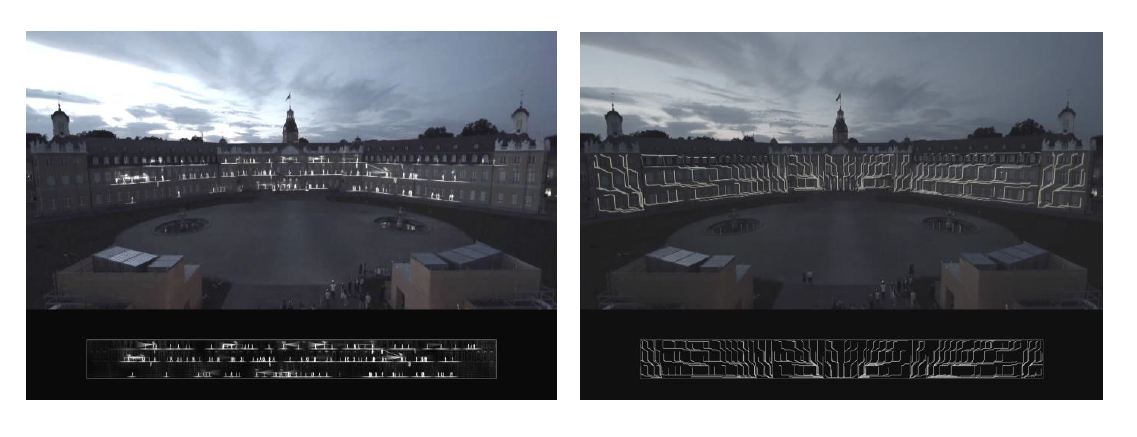
In this section we will describe the knowledge representation we constructed for PUC1, the multimodal fusion mechanism, which inserts the information from the visual and textual analysis component into the ontology, and the reasoning mechanism that extracts information from the ontology for the needs of PUC1. Moreover, we give a discussion over the results and some future work directions. But first we will give a brief description of PUC1 and the motivation.

## Motivation and Description of PUC1

PUC 1 aims to improve urban design in a fast growing city by tackling new difficulties that may occur in terms of functionality, mobility, attractiveness, cultural preservation, and environmental protection. MindSpaces will use unique art pieces in strategic locations to raise sensitivity and understanding about a city's historic relevance and current challenges, such as the environment and mobility. As a result, it has the potential to raise awareness of the city's cultural significance as well as challenges associated with its expansion, including environmental and mobility difficulties. It can also create spaces that allow for new forms of social interaction and new levels of social integration with the urban fabric. As a result, the area's touristic potential, citizen well-being, quality of life, and total economic activity will all improve.

Targeted organisations of PUC1 are City councils and municipalities that want to renovate outdoors urban spaces, architecture offices that want to democratise the design process and improve outdoors urban design, architecture academic units studying trends and innovations in outdoors urban design, virtual reality and augmented reality companies that want to deploy realistic city scenarios in their games.

For the implementation, first, off-the-shelf tools will provide 2D and 3D documentation of the urban environment, based on data from a variety of sources, including drone footage, 3D scans, and mobile mapping surveys, on which architects and artists will collaborate. Additional input is available through citizen participation and social media data. Citizens of L'Hospitalet (which is the city that the experiments will take place) will be able to stroll through this "latent" city space and select abstract spatial occurrences of their choice, in response to difficulties and challenges relating to the city's socioeconomic demands as proposed by city authorities (L'Hospitalet). Then with a computational design tool that can facilitate the translation of these abstract city forms to urban geometry. Architects and creatives will use these urban design alternatives to generate more detailed spatial interaction models to be experienced in virtual reality. Feedback from the general public will be collected, analysed, and unified using multimodal fusion techniques under the knowledge base, where aggregate knowledge can indicate changes in the virtual environment's parameters based on intelligent decision making, in order to arrive at a proposal generated by the collective behaviour of participants. Figure 10 depicts a typical goal of urban redevelopment.

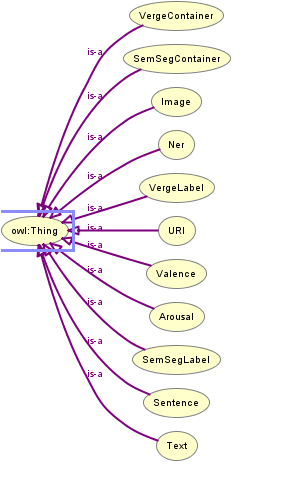


**Figure 11:** Urban redesign of Karlsruhe square[[1]](#footnote-1)

## Knowledge Representation for Outdoor Urban Environments

The ontology scheme was constructed based on the requirements of the MindSpaces project. In more detail, we analysed the messages that we received from the textual and visual analysis components, and we developed an ontology scheme based on those. In other words, we represent the important concepts of these messages through classes and relations between them. Moreover, we analysed the requirements of the reasoning mechanism, meaning that we took into consideration the competency questions from Table 3, and defined some classes and properties in such a way that would help the reasoning mechanism return the crucial information.

In Figure 11 one can see the whole ontology scheme for PUC1, as can be noticed the hierarchy relations between the classes are trivial, as there was no need for a sophisticated hierarchical model. Next, Table 4 shows the classes of the ontology along with a small description for them (i.e., their purpose). Table 5 describes the object type relations of the ontology (i.e., the properties that have domain and range a class of the ontology). Finally, Table 6 describes the data type properties of the ontology (i.e., the properties that have domain a class of the ontology and range a data type).



**Figure 12:** Ontology Scheme for PUC1

|  |  |  |
| --- | --- | --- |
| **Class** | **Usage Note** | **URI** |
| Arousal | A class that indicates the arousal of the user for an image | mind1:Arousal |
| Image | A class that contains the IDs of the images | mind1:Image |
| Ner | A class that contains information about the named entities found in an image | mind1:Ner |
| SemSegContainer | A class that contains information about the objects segmented into an image | mind1:SemSegContainer |
| SemSegLabel | A class that contains the IDs of the objects that are segmented into an image | mind1:SemSegLabel |
| Sentence | A class that contains information about the sentences of the textual description of an image from a user | mind1:Sentence |
| Text | A class that contains information about the textual description of an image from a user | mind1:Text |
| URI | A class that contains information about the DBpedia[[2]](#footnote-2) URIs of the named entities from an image | mind1:URI |
| Valence | A class with information about the valence of the user for an image | mind1:Valence |
| VergeContainer | A class with information with the verge classification of an image | mind1:VergeContainer |
| VergeLabel | A class with the verge labels of an image | mind1:VergeLabel |

**Table 4:** Classes of the ontology scheme for PUC1

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Description** | **Domain** | **Range** |
| hasArousal | A property that relates an image with its arousal scores | mind1:Image | mind1:Arousal |
| hasNer | A property that relates a textual description of an image with its named entities | mind1:Text | mind1:Ner |
| hasSentence | A property that relates a textual description of an image with its sentences | mind1:Text | mind1:Sentence |
| hasText | A property that relates an image with its textual description | mind1:Image | mind1:Text |
| hasURI | A property that relates the named entities of an image with their URIs | mind1:Ner | mind1:URI |
| hasValence | A property that relates an image with its valence scores | mind1:Image | mind1:Valence |
| isSemSegContainer | A property that relates an image with its segmented objects | mind1:Image | mind1:SemSegContainer |
| isSemSegLabel | A property that relates a segmented object with its label | mind1:SemSegContainer | mind1:SemSegLabel |
| isVergeContainer | A property that relates an image with its verge classification | mind1:Image | mind1:VergeContainer |
| isVergeLabel | A property that relates a verge classification of an image with its labels | mind1:VergeContainer | mind1:VergeLabel |

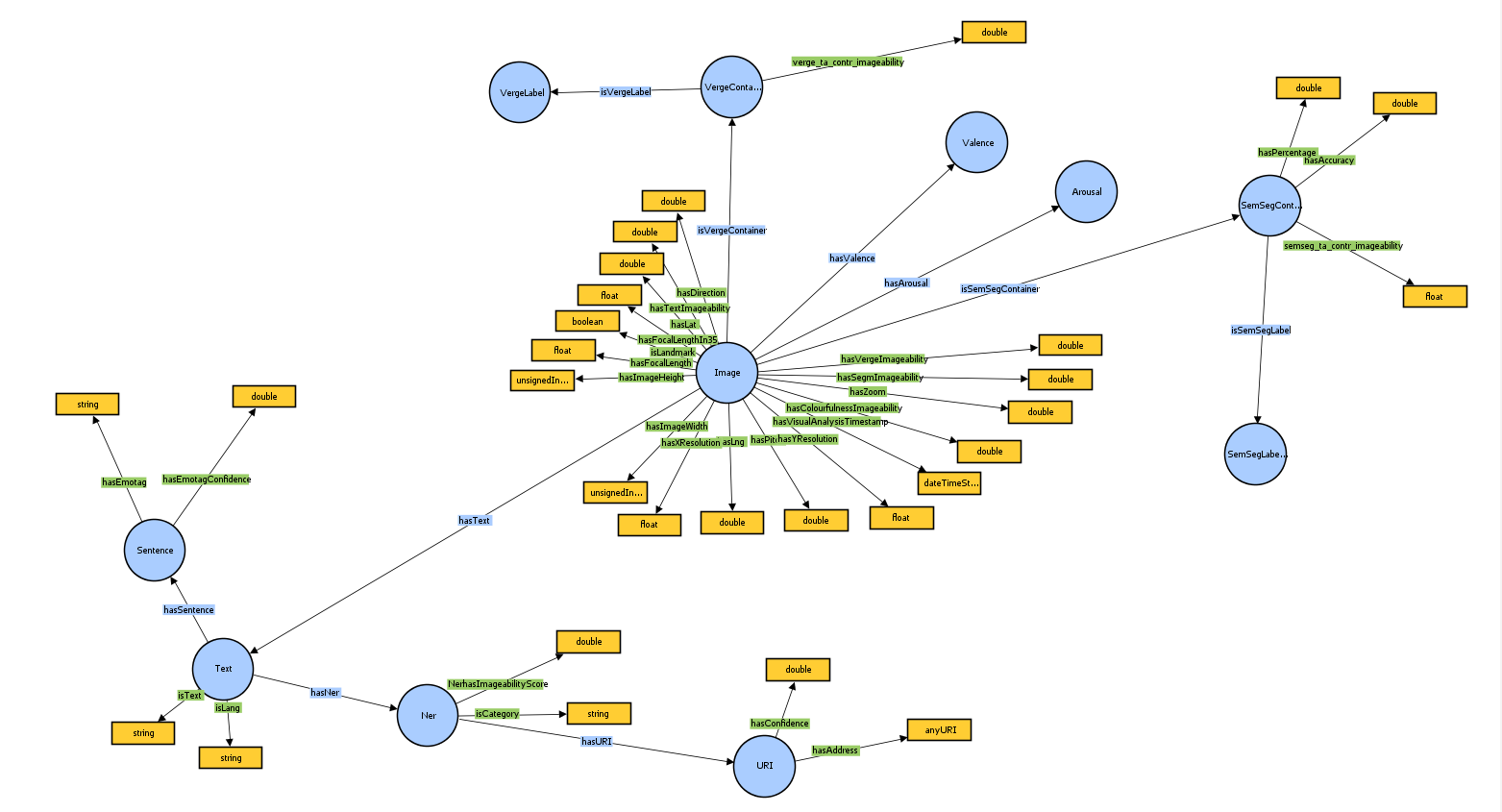
**Table 5:** Object properties of the ontology scheme of PUC1

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Description** | **Domain** | **Range** |
| hasAccuracy | A property that indicates the accuracy of a segmented object in an image | mind1:SemSegContainer | xsd:double |
| hasAddress | A property that indicates the URI address of a named entity from an image | mind1:URI | xsd:anyURI |
| hasBegin | A property that indicates the beginning of a timeframe | mind1:Ner OR mind1:Sentence OR mind1:URI | xsd:int |
| hasColourfulnessImageability | A property that indicates the colourfulness imageability of an image | mind1:Image | xsd:double |
| hasConcept | A property that indicates the concepts in a sentence | mind1:Sentence OR mind1:URI | xsd:string |
| hasConfidence | A property that indicates the confidence of the asserted URI to a named entity | mind1:URI | xsd:double |
| hasCoveredText | A property that indicates the covered text of an sentence from an image | mind1:Ner OR mind1:Sentence | xsd:string |
| hasDirection | A property that indicates the direction of an image | mind1:Image | xsd:double |
| hasEmotag | A property that indicates the emotional tag of an image | mind1:Sentence | xsd:string |
| hasEmotagConfidence | A property that indicates the confidence of the emotag for an image | mind1:Sentence | xsd:double |
| hasEnd | A property that indicates the ending of a timeframe | mind1:Ner OR mind1:Sentence OR mind1:URI | xsd:int |
| hasFocalLength | A property that indicates the focal length of an image | mind1:Image | xsd:float |
| hasFocalLength35 | A property that indicates the focal length 35 of an image | mind1:Image | xsd:float |
| hasImageHeight | A property that indicates the height of an image | mind1:Image | xsd:int |
| hasImageWidth | A property that indicates the width of an image | mind1:Image | xsd:int |
| hasLabel | A property that indicates the label of a named entity, a sentence, or an URI | mind1:Ner OR mind1:Sentence OR mind1:URI | xsd:string |
| hasLat | A property that indicates the latitude of an image | mind1:Image | xsd:double |
| hasLng | A property that indicates the longitude of an image | mind1:Image | xsd:double |
| hasPercentage | A property that indicates the percentage of a segmented object in an image | mind1:SemSegContainer | xsd:double |
| hasPitch | A property that indicates the pitch of an image | mind1:Image | xsd:double |
| hasProbability | A property that indicates the probability of the arousal of an image, the valence of an image, or the verge contained in an image | mind1:Arousal OR mind1:Valence OR mind1:VergeContainer | xsd:double |
| hasSegmImageability | A property that indicates the probability of the segmented in an image | mind1:Image | xsd:double |
| hasTextImageability | A property that indicates the text imageability | mind1:Image | xsd:double |
| hasType | A property that indicates the type of a sentence of an URI | mind1:Sentence OR mind1:URI | xsd:string |
| hasValue | A property that indicates the value of the arousal or valence in an image | mind1:Arousal OR mind1:Valence | xsd:double |
| hasVergeImageability | A property that indicates the imageability of the verges in an image | mind1:Image | xsd:double |
| hasVisualAnalysisTimestamp | A property that indicates the timestamp of the verges in an image | mind1:Image | xsd:dateTimeStamp |
| hasXResolution | A property that indicates the X resolution of an image | mind1:Image | xsd:float |
| hasYResolution | A property that indicates the Y resolution of an image | mind1:Image | xsd:float |
| hasZoom | A property that indicates the zoom of an image | mind1:Image | xsd:double |
| isCategory | A property that indicates the category of an named entity | mind1:Ner | xsd:string |
| isLandmark | A property that indicates if an image contains a landmark or not | mind1:Image | xsd:boolean |
| isLang | A property that indicates the language of an image’s text | mind1:Text | xsd:string |
| isText | A property that indicates the text of an image | mind1:Text | xsd:string |
| NerhasImageabilityScore | A property that indicates the imageability score of the named entities | mind1:Ner | xsd:double |
| semseg\_ta\_contr\_imageability | A property that indicates the contribution imageability score for the segmented objects in an image from the textual analysis | mind1:SemSegContainer | xsd:float |
| verge\_ta\_contr\_imageability | A property that indicates the contribution imageability score for the verge category in an image from the textual analysis | mind1:VergeContainer | xsd:double |

**Table 6:** Datatype properties of the ontology scheme for PUC1

As one can notice the namespace **mind1** was given to indicate ontology classes and properties. Next, we will give a detailed analysis of the relations between the classes and their purpose. The whole ontology scheme with the object and data type properties can be found in Figure 12. In Figure 12 the blue circles are the classes, the edges which have a domain and range classes (i.e., start and end at a blue circle) are the object type properties, and the edges which have domain a class and range a data type (i.e., start from a blue circle and end at yellow rectangle) are the data type properties.

* The **Sentence** class contains information about the sentences that compose a textual description of an image. For each sentence it has information about the emotional tag that was given by the user, and the confidence of each emotional tag.
* The **Text** class contains information about the textual description of an image. The Text class is connected through the property **hasSentence** with the class Sentence, in order to give further information about the sentences that compose the textual description. Moreover, it has information for the language of the textual description, and the textual description itself. Finally, the Text class is connected through the property **hasNer** with the **Ner** class, which has information about named entities found in the textual description. Named entities can be words that refer to real life objects, actions, or activities.
* The **Ner** class contains information about the named entities found in the textual description of an image. The Ner class gives information for the category of the named entity relation (i.e., if it is a concept or something else), the imageability score of the named entity, and is connected through the property **hasURI** with the class URI.
* The **URI** class contains information about the URIs of the named entities. Currently, the URIs point only to DBpedia entities. The URI class has information about the URI link and the confidence that a named entity should be related with a specific URI.
* The **VergeLabel** class contains information about the labels of the verge classifications found in an image.
* The **VergeContainer** class contains information about the imageability scores of the verge classifications found in an image. Moreover, it is connected though the property **hasVergeLabel** with the class **VergeLabel**, in order to indicate the label of a verge classification.
* The **SemSegLabel** class contains information about the labels of the segmented objects found in an image.
* The **SemSegContainer** class contains information about the imageability scores of the segmented objects found in an image, the percentage of space they capture in the image, and the confidence that they are part of the image. Moreover, it is connected though the property **hasSemSegLabel** with the class **SemSegLabel** , in order to indicate the label of a segmented object.
* The **Arousal** class contains information about the arousal score that was given by a user for an image.
* The **Valence** class contains information about the valence score that was given by a user for an image.



**Figure 13:** Ontology scheme with object and datatype properties

* The **Image** class is the most important class, as it contains a lot of metadata information about the characteristics of the image, such as the latitude, longitude, the pitch, the zoom, and others. But apart from these it is connected: (i) with the **Arousal** class through the property **hasArousal** to indicate its arousal, (ii) with the **Valence** classthrough the property **hasValence** to indicate its valence, (iii) with the **VergeContainer** class through the property **isVergeContainer** to give information about the verge classifications that it has, (iv) with the **SemSegContainer** class through the property **isSemSegContainer** to give information about the segmented objects it contains, and (v) with the **Text** class through the property **hasText** to give information about the textual description that it has.

## Multimodal Fusion for Outdoor Urban Environments Use Case

The multimodal integration mechanism was constructed after the development of the ontology scheme presented in subsection 3.2, in order to map into the ontology the messages from the visual analysis and textual analysis (T5.3). The ontology after its development was uploaded into a GraphDB[[3]](#footnote-3) repository. Moreover, the integration mechanism was developed using NodeJS[[4]](#footnote-4).

As mentioned, the integration mechanism maps the messages from the visual and textual component. These messages come in the form of json files and are mapped into the ontology, in other words the ontology scheme is populated with these messages.

An example of a json file from the visual analysis message can be found in Example 1.

**Example 1:** Visual analysis message in json format

{"timestamp": "2022-03-17 10:36:22.541764+01:00CET", "segmentation":{"segm\_labels": ["tree", "road", "building", "grass", "sky", "sidewalk", "bench", "car", "bus", "truck", "streetlight", "van", "stairs", "wall"], "segm\_percentages": [0.2822998046875, 0.224456787109375, 0.20325927734375, 0.112493896484375, 0.088250732421875, 0.069927978515625, 0.01036376953125, 0.006329345703125, 0.000738525390625, 0.00068359375, 0.0005859375, 0.0005615234375, 2.44140625e-05, 2.44140625e-05], "segm\_colorfulness": [22.635735826237845, 8.505740585173912, 34.45423584723287, 30.52419048853266, 36.970240267362016, 16.54262509691435, 50.31528201748002, 21.892850725932778, 28.479133596664084, 19.33818522116958, 43.1487497376642, 18.385260802988064, 29.71799123380608, 35.04844901850117]},"classification": {"verge\_labels": ["/j/jewelry\_shop"], "verge\_probabilities": [0.38903313875198364]},"sentiment": {"valence": "positive", "valence\_probability": 0.5103972554206848, "arousal": "excited", "arousal\_probability": 0.6642298698425293}}

A high level analysis of the json file from Example 1 is that it contains information about the segmented objects along with their percentages in the image, the segmented objects colourfulness, the verge classification, and their probabilities (i.e., confidence of the classification), and the valence and arousal with their probabilities. Also, some metadata such as the timestamp are included in the json file. Next, Example 2 displays a json file from the textual analysis component.

**Example 2:** Textual analysis message in json format

{"id": "JPEG\_20220204\_153627\_4494048047500552089.jpg", "text": "Un lloc on quedava i on quedo amb amics i companys.", "language": "ca", "result":{"ner": {"directed": true, "type": "ner", "label": "ner", "nodes": [{"id": 1, "type": "concept", "label": "lloc", "metadata": {"begin": 3, "end": 7, "category": "CONCEPT", "covered\_text": "lloc"}}, {"id": 2, "type": "concept", "label": "amics", "metadata": {"begin": 34, "end": 39, "category": "CONCEPT", "covered\_text": "amics"}}, {"id": 3, "type": "concept", "label": "companys", "metadata": {"begin": 42, "end": 50, "category": "CONCEPT", "covered\_text": "companys"}}], "edges": [], "metadata": {}},"concept\_extraction": {"directed": true, "type": "concept", "label": "concept extraction", "nodes": [{"id": 4, "type": "concept", "label": "lloc", "metadata": {"begin": 3, "end": 7, "covered\_text": "lloc"}}, {"id": 5, "type": "concept", "label": "quedava", "metadata": {"begin": 11, "end": 18, "covered\_text": "quedava"}}, {"id": 6, "type": "concept", "label": "quedo", "metadata": {"begin": 24, "end": 29, "covered\_text": "quedo"}}, {"id": 7, "type": "concept", "label": "amics", "metadata": {"begin": 34, "end": 39, "covered\_text": "amics"}}, {"id": 8, "type": "concept", "label": "companys", "metadata": {"begin": 42, "end": 50, "covered\_text": "companys"}}], "edges": [], "metadata": {}},"dbpedia\_linking": {"directed": true, "type": "dbpedia", "label": "dbpedia linking", "nodes": [{"id": 11, "type": "dbpedia", "label": "lloc", "metadata": {"begin": 3, "end": 7, "uri": "http://dbpedia.org/resource/Lockdown", "confidence": 0.9328642105090836, "label": "Lloc", "categories": ["other"], "types": "", "covered\_text": "lockdown"}}], "edges": [], "metadata": {}},"emotion\_detection": {"directed": true, "type": "sentiment", "label": "polarity detection", "nodes": [{"id": 9, "type": "polarity", "label": "Un lloc on quedava i on quedo amb amics i companys.", "metadata": {"begin": 0, "end": 51, "covered\_text": "Un lloc on quedava i on quedo amb amics i companys.", "emotag": "pos", "confidence": 0.5216434959524886}}], "edges": [], "metadata": {}},"imageability\_scoring":{"directed": true, "type": "imageability", "label": "imageability scoring", "nodes": [{"id": 10, "type": "imageability", "label": "Un lloc on quedava i on quedo amb amics i companys.", "metadata": {"begin": 0, "end": 51, "imageability": 5.346666666666667, "covered\_text": "Un lloc on quedava i on quedo amb amics i companys.", "concept\_wise\_imageability": [4.33, 6.09, 5.62]}}], "edges": [],"metadata": {"jsonContent": [{"name": "segmentation", "input": {"contribution": [0.2822998046875, 0.224456787109375, 0.20325927734375, 0.112493896484375, 0.088250732421875, 0.069927978515625, 0.01036376953125, 0.006329345703125, 0.000738525390625, 0.00068359375, 0.0005859375, 0.0005615234375, 2.44140625e-05, 2.44140625e-05], "lang": "ca", "concept\_list": [["tree"], ["road"], ["building"], ["grass"], ["sky"], ["sidewalk"], ["bench"], ["car"], ["bus"], ["truck"], ["streetlight"], ["van"], ["stairs"], ["wall"]], "avgtype\_for\_concepts": "modif\_product"}, "output": {"imageability": 6.0318669854, "final\_contribution": [0.3095997683, 0.2579043365, 0.2078834454, 0.1294601508, 0.0889849573, 0.0, 0.0, 0.0041443069, 0.0008339235, 0.000748468, 0.0, 0.0004152773, 0.0, 2.53658e-05]}},{"name": "colourfulness", "input": {"contribution": [0.2822998046875, 0.224456787109375, 0.20325927734375, 0.112493896484375, 0.088250732421875, 0.069927978515625, 0.01036376953125, 0.006329345703125, 0.000738525390625, 0.00068359375, 0.0005859375, 0.0005615234375, 2.44140625e-05, 2.44140625e-05], "colourfulness": [22.635735826237845, 8.505740585173912, 34.45423584723287, 30.52419048853266, 36.970240267362016, 16.54262509691435, 50.31528201748002, 21.892850725932778, 28.479133596664084, 19.33818522116958, 43.1487497376642, 18.385260802988064, 29.71799123380608, 35.04844901850117], "lang": "ca"}, "output": {"colour\_imageability": 3.995511219164464, "overall\_colourfulness": 37.101175606527164, "overall\_segm\_imageability": 5.013689102282232}},{"name": "verge", "input": {"contribution": [0.3939639627933502, 0.33205193281173706], "lang": "ca", "concept\_list": [["street"], ["residential", "neighborhood"]], "avgtype\_for\_concepts": "modif\_product"}, "output": {"imageability": 6.3862791462, "final\_contribution": [0.5438040581, 0.4561959419]}},{"name": "comment", "input": {"lang": "ca", "concept\_list": [["lloc"], ["amics"], ["companys"]], "avgtype\_for\_concepts": "modif\_product", "avgtype\_for\_sentence": "harmonic"}, "output": {"imageability": 5.346666666666667, "concept\_wise\_imageability": [4.33, 6.09, 5.62]}}]}}},"metadata": {"date": "2022/03/10 13:38:51", "times": {"PARSING": "13:38:45", "EMOTION": "13:38:45", "NER": "13:38:46", "CONCEPT\_CANDIDATES": "13:38:46", "DBPEDIA": "13:38:51", "IMAGEABILITY": "13:38:51", "finalStep": "13:38:51"}}}

The textual analysis message contains information about the textual description of the image, and the emotion that caused it to the user. Moreover, it contains information about the named entities found in the textual description, their type (i.e., if they are a concept or not), their label, their DBpedia link, their begin and end timestamp (i.e., the moment they are spotted). The textual analysis message also contains some metadata information for the named entities and the image, which are the contributions of the named entities in the imageability score of the image, the overall imageability, colour imageability, and segmented imageability.

Finally, the integration mechanism receives one more message whose information is populated into the ontology. The last message is called a metadata message and contains mostly metadata information. An example of a metadata json message can be found in Example 3.

**Example 3** Metadata message in json format

{"endLat": 41.36191102065125, "endLng": 2.120989619781634, "startLat": 41.367969057173234, "startLng": 2.109804765061636, "routeName": "", "id": "63baab4e-895c-4b70-bc3c-16afde6381ea", "user\_id": "sstamatis14203@gmail.com", "lat": 40.5656154, "lng": 22.9953671,"pitch": "", "zoom": "", "landmark": true, "edge": false, "node": false, "district": false, "path": false, "direction": "", "meta":{"XResulution": "", "YResulution":"", "FocalLength": "", "ImageWidth":"", "ImageHeight":"", "FocalLengthIn35":""}}

The metadata message mostly contains information about the metadata of the image, such as the latitude, the longitude, the pitch, and others. Also, it gives information about the user id and if an image has a landmark or not.

Notice that the integration mechanism does not get as input three different json files from the visual and textual components, but rather a big json message which contains the three aforementioned parts (i.e., visual, textual, and metadata).

## Reasoning for Outdoor Urban Environments Use Case

This subchapter entails all relevant information and strategies regarding the corresponding reasoning mechanisms which were deployed for the PUC 1 workshop in L’ Hospitalet de Llobregat, Barcelona, Spain. The reasoning mechanisms take advantage of the ontologies which were formerly created to support the use case of outdoor urban environments as well as the actual population of the knowledge base with content and metadata deriving from both artists and users.

The main idea is to feed on demand an interactive google map with geolocated 2D points in the form of CSV files corresponding to image entries inside the knowledge base through application programming interfaces (APIs) developed in node.js and JavaScript. The graphdh.js library was used to establish the connection and transactions towards and from the GraphDB repository[[5]](#footnote-5) with authorization and authentication ensured. The file delivered in response follows a scalable and dynamic approach, meaning it is being generated on demand based on live requests, thus ensuring always up-to-date data delivery as the knowledge base supports a continuous online population and new entries may arrive anytime. The format of the file consists of as many lines as the images fulfil the SPARQL queries and 5 columns:

* *latitude* (the latitude when the corresponding image was captured),
* *longitude* (the longitude when the corresponding image was captured),
* *point size* (the size of the circle to be depicted on the map),
* *point opacity* (the opacity of the circle to be depicted on the map),
* *point colour* (the colour of the circle to be depicted on the map).

In total, 8 SPARQL queries were formulated, each satisfying a different user requirement, followed by an additional multipurpose sparse function.

**Query 1:** Retrieve images ordered by all image entries having latitude, longitude, direction, segmentation imageability, verge imageability, valence with value and probability, arousal with value and probability, all segmentation labels with percentages and all verge labels with percentages. The point size is set to 25 and the point opacity is set to 0.03. All points are given the colour red.

**Query 2:** Retrieve images ordered by all image entries having latitude, longitude, direction, segmentation imageability, verge imageability, valence with value and probability, arousal with value and probability, all segmentation labels with percentages, all verge labels with percentages and the landmark, node or edges statutes optionally if they exist. If the point is a landmark set colour to green, if it is an edge set colour to red, else if the point is a node set colour to blue. The point size is set to 33 and the point opacity to 1.

**Query 3:** Retrieve images ordered by all image entries having latitude, longitude, direction, segmentation imageability, verge imageability, valence with value and probability, arousal with value and probability, all segmentation labels with percentages and all verge labels with percentages. If valence is “positive”, set the green channel to 88 HTML colour code. If valence is “neutral”, set red, green and blue channels to 88 HTML colour code. If valence is “negative”, set the red channel to 88 HTML colour code. If arousal is “calm” and if the green channel is unset, set the green channel to 88 HTML colour code, else set the green channel to FF HTML colour code. If arousal is “neutral” and if the red channel is unset, set the red channel to 88 HTML colour code else set it to FF HTML colour code. If the arousal is “neutral” and and if the green channel is unset, set it to 88 HTML colour code, else set it to FF HTML colour code. If the arousal is “neutral” and the blue channel is unset, set it to 88 HTML colour code, else set it to FF colour code. If the arousal is “excited” and the red channel is unset, set it to 88 HTML colour code, else set it to FF HTML colour code. Finally, concatenate the red, green and blue resulting HTML colour channels in a unique string to decide the final colour of the point. Point size is set to 30 and point opacity is set to 0.01

**Query 4:** Retrieve images ordered by all image entries having latitude, longitude, direction, segmentation imageability, verge imageability, valence with value and probability, arousal with value and probability, all segmentation labels with percentages and all verge labels with percentages. Check the top 1 segmentation class and if it is a building or vegetation or open sky, give the point the colour red, green or blue respectively. The point size is set to 25 and the point opacity is set to 0.3.

**Query 5:** Retrieve images ordered by all image entries having latitude, longitude, direction, segmentation imageability, verge imageability, valence with value and probability, arousal with value and probability, all segmentation labels with percentages and all verge labels with percentages. Then group by green colour “nature and open spaces” if the top verge label is: zen garden, park, picnic area, botanical garden, orchard, broadleaf forest, playground, farm, cultivated field, vineyard, pasture, promenade, plaza, pavilion, exterior balcony or amphitheatre. Then group by blue colour “built urban spaces” if the top verge label is: residential neighbourhood, outdoor apartment building, downtown, cemetery, outdoor hotel or hospital. Finally, group by red colour “disused or avoided spaces” if the top verge label is tundra, parking lot, driveway, wild field, landfill or sandbox. For all points, point size is set to 25 and point opacity to 0.3.

**Query 6:** Retrieve images ordered by all image entries having latitude, longitude, direction, segmentation imageability, verge imageability, valence with value and probability, arousal with value and probability, all segmentation labels with percentages, all verge labels with percentages and optionally the total imageability score. Then by using a rainbow colour function give a rainbow colour to the point scaling respectively from 0 to 7 scores.

**Query 7:** Given a snap image, return a list of images as shown in the table below, where: (i) the Top 3 segmentation labels of the snap (based on the coverage percentage) exist in the images, (ii) the images must have imageability >= imageability of snap + 0.05, (iii) the Top 3 segmentation labels must exist in the images with the same coverage percentage, or a 20% difference, and (iv) the results must be limited to 8 images, if there exists as many, sorted based on their imageability.

|  |  |  |  |
| --- | --- | --- | --- |
| snap | image 1 | … | image 8 |
| label1 segmentation score in snap | label1 segmentation score in image 1 | … | label1 segmentation score in image 8 |
| label2 segmentation score in snap | label2 segmentation score in image 1 | … | label2 segmentation score in image 8 |
| label3 segmentation score in snap | label3 segmentation score in image 1 | … | label3 segmentation score in image 8 |
| label1 colourfulness score in snap | label1 colourfulness score in image 1 | … | label1 colourfulness score in image 8 |
| label2 colourfulness score in snap | label2 colourfulness score in image 1 | … | label2 colourfulness score in image 8 |
| label3 colourfulness score in snap | label3 colourfulness score in image 1 | … | label3 colourfulness score in image 8 |
| total imageability | total imageability | … | total imageability |
| URL of snap | URL of image 1 | … | URL of image 8 |

In the first row we have the label of the image, and in the next three rows we have the segmentation scores (i.e., coverage) for each one of the top 3 segmented objects that each image has, similarly, in the next three rows, we have the segmentation colourfulness scores for the same objects. Finally, in the last two rows, we have the total imageability score and the URL of the image, respectively.

**Query 8:** Given a snap image, return a list of images as shown in the table below, where: (i) the Top 3 segmentation labels of the snap (based on the colourfulness percentage) exist in the images, and their colourfulness is above 1, and (ii) for each one of the 3 segmentation classes bring the Top 5 images with the highest colourfulness for each segmentation classes.

|  |  |  |  |
| --- | --- | --- | --- |
| snap | image 1 | … | image 15 |
| label1 colourfulness score in snap | label1 colourfulness score in image 1 | … | label1 colourfulness score in image 15 |
| label2 colourfulness score in snap | label2 colourfulness score in image 1 | … | label2 colourfulness score in image 15 |
| label3 colourfulness score in snap | label3 colourfulness score in image 1 | … | label3 colourfulness score in image 15 |
| label1 segmentation score in snap | label1 segmentation score in image 1 | … | label1 segmentation score in image 15 |
| label2 segmentation score in snap | label2 segmentation score in image 1 | … | label2 segmentation score in image 15 |
| label3 segmentation score in snap | label3 segmentation score in image 1 | … | label3 segmentation score in image 15 |
| total imageability | total imageability | … | total imageability |
| URL of snap | URL of image 1 | … | URL of image 8 |

In the first row we have the label of the image, and in the next three rows we have the segmentation colourfulness scores for each one of the top 3 segmented objects that each image has, similarly, in the next three rows, we have the segmentation scores (i.e., coverage) for the same objects. Finally, in the last two rows, we have the total imageability score and the URL of the image, respectively.

**Sparse Function:** This function was constructed in native JavaScript and is applied on the results of every query and just before the generation of the CSV file. Its purpose is to diminish the resulting pool of points and provide some sort of information aggregation. More specifically, it rounds the latitude, the longitude and the distance. Then, it clusters points first by latitude, then by longitude and finally by direction. Afterwards, it merges on imageability scores, segmentation labels and averages the percentages. Moreover, it merges on verge labels and averages their percentages. Finally, the function merges on sentiment data.

## Discussion and Future Work

In this chapter we described the ontology scheme, the integration mechanism, and the reasoning mechanism for PUC1. The ontology scheme is currently at a satisfying point as it can facilitate all the information that comes from the visual and textual component. Therefore, the ontology scheme seems that it will not need any further development. As mentioned in subsection 3.1 the ontology scheme was constructed based on the information that we would receive from the other components, and the competency questions that the reasoning mechanism needed to be answered (see Table 3). For this reason, we managed to produce an ontology scheme which is concise and does not need further improvement.

For the integration mechanism, the mapping of the message from the visual analysis component is developed in order to capture any exception that the component could bring. This means that if some keys of the json message do not contain any information the integration mechanism will tackle this exception, and map only the remaining information. But for the mapping of the message from the textual analysis, further improvement can be done, as we have not yet captured each subcase of the forms that the json message could have.

The reasoning mechanism currently can answer 6 questions, and it returns an excel file for each one of them. These 6 questions were defined by domain experts (i.e., architects) as the desired information at an initial instance. But obviously, further improvement can also be done to this part of the component. We are currently working on defining some detailed use cases, in order to build the necessary SPARQL queries and retrieve the desired information. For instance, a question that could be answered by our reasoning mechanism and could help in a use case, would be “Give me all the images that contain the X,..., Y verges in them”, or “Give all the objects that were annotated with a positive emotional tag by the user, in the region with width X and length Y”.

As for future work, we have constructed, and we are integrating, a mechanism that deletes the information of an image if that is part of our populated ontology. Meaning that if a user gives information about an image that exists in our populated ontology, then all the information that already exists in the ontology for the image will be deleted, and the new one will be inserted. Moreover, as mentioned in the previous paragraph we will develop “clever” SPARQL queries, in order to support more use cases. Finally, we need to evaluate our system with regards to its rationality and quality, either with an user evaluation or with a pipeline for ontology or information retrieval evaluation.

# INSPIRING WORKSPACES (PUC 2)

In this section, we give a brief reminder of the motivation for PUC2, we then give a detailed description of the knowledge representation for this use case. Additionally, we present the integration mechanism that we constructed to populate the ontology from the messages received by the visual analysis component (sub Section 4.2). In subsection 4.3 we present the reasoning mechanism for PUC2. We conclude the section with a description of the results and some future work directions.

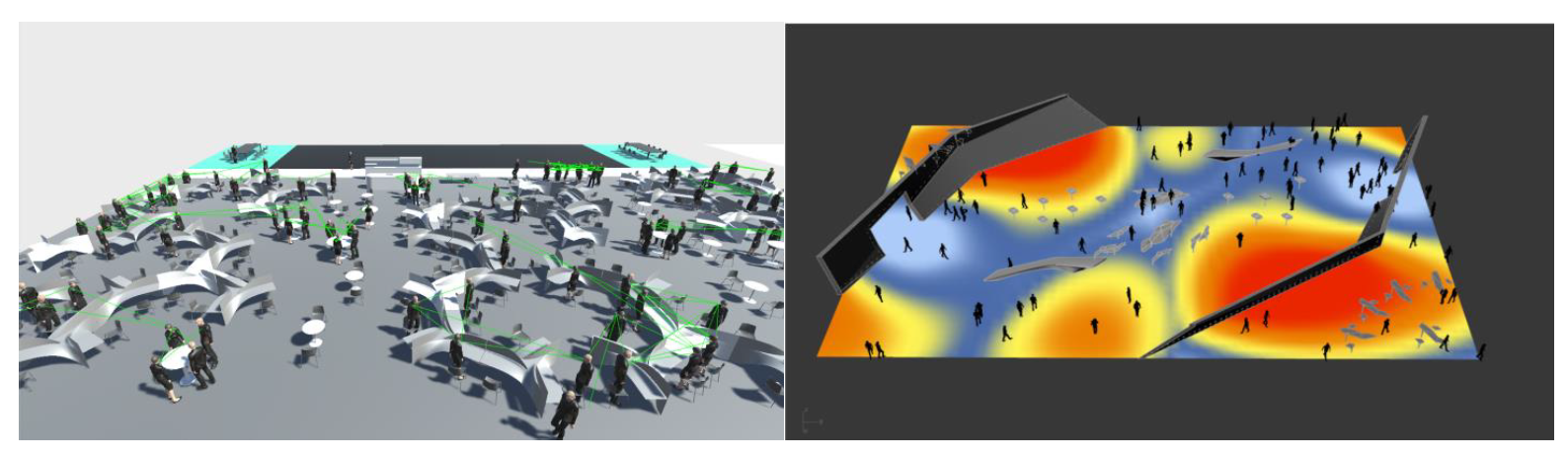
## Motivation and Description of PUC2

Workplace design used to be unimaginative and dreary, but in recent years, aesthetically and functionally creative workspaces have been established that are better prepared to enable the dynamic communication that today's networked society requires. Increasing positive social contact chances in the workplace boosts productivity and creativity across departments and teams. Emotionally appealing workspaces can be motivating, and when paired with designs that encourage more dynamic and diverse social activity, they can boost employee productivity and well-being. Modern office design can be steered in unexpected directions by architects and artists, improving its attractiveness and efficacy. Artists and architects will receive immediate user feedback while immersed in their creations in MindSpaces, and this feedback data will be directly related to the creation of better workplace solutions.

Target applications can be big companies, which occupy more than 200 employees, and need to renovate their workspace so as to maximise the engagement, productivity and interaction of their workers, architecture offices that design efficient, functional and relaxing workspaces.

A pilot testbed will be developed in Barcelona, with terrestrial laser scanners and data from a custom-built 3D sensing platform being utilised to create 3D representations of the original selected workspace. The office is excellent because it is a typical medium-sized office that accommodates a multi-disciplinary workforce in both the creative and technology industries, and most importantly, it provides a real-world environment in which data on the real-world design context can be collected and analysed. Architects can then utilise the information to offer recommendations for modifying or entirely redesigning the area. Furthermore, physiological data will be collected from home-workers changing virtual reproductions of their home offices to represent the new reality created by the pandemic, where the house becomes the office.

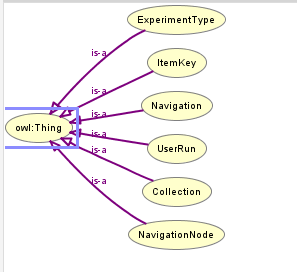
A series of experiments will be set up to explore design parameters, feature compositions, and spatial organisation possibilities. To that purpose, methods for collecting and analysing three categories of data, including explicit preferences, multi-modal internal state feedback, and human behaviour, will be developed for both individuals and groups of users. Furthermore, when users (architects and artists) investigate the data/statistics to discover inspiration for their work, a textual analysis tool will be built to enable faceted search. Furthermore, to further enhance the range of input parameter sources under examination, an experiment on collecting and managing environmental characteristics for indoor areas will be conducted. Designers will be able to draw inspiration from the toolset, evaluate choices (via simulation), and forecast how people will act in them. Making geographical occupancy maps, quantifying spatial and asset use, producing 3d spatial vision maps, and quantifying and visualising social encounters and group behaviours are just a few of the metrics available. All of this will be experienced by users in a virtual reality environment, together with a sentiment extraction approach based on multimodal fusion to forecast how well a workplace design would perform. Figure 13 illustrates a typical goal of a workplace makeover.



**Figure 14:** Workspace redesign based on people behaviour patterns and spatial needs

## Knowledge Representation for Inspiring Workspaces Use Case

The ontology scheme was constructed based on the requirements of the MindSpaces project. In more detail, we analysed the messages that we received from the visual analysis component, and we developed an ontology scheme based on those. In other words, we represent the important concepts of that message through classes and relations between them. In Figure 14 one can see the whole ontology scheme for PUC2, as can be noticed the hierarchy relations between the classes are trivial, as there was no need for a sophisticated hierarchical model. Next, Table 7 shows the classes of the ontology along with a small description for them (i.e., their purpose). Table 8 describes the object type relations of the ontology (i.e., the properties that have domain and range and class of the ontology). Finally, Table 9 describes the data type properties of the ontology (i.e., the properties that have domain a class of the ontology and range a data type).



**Figure 15:** Ontology Scheme for PUC2

|  |  |  |
| --- | --- | --- |
| **Class** | **Usage Note** | **URI** |
| Collection | A class that contains information about experiment status data change | mind2:Arousal |
| ExperimentType | A class that contains the information for the experiment type data changes | mind2:Image |
| ItemKey | A class with the item key information | mind2:Ner |
| Navigation | A class with information for the navigation configuration | mind2:SemSegContainer |
| NavigationNode | A class with information about the navigation nodes | mind2:SemSegLabel |
| UserRun | A class that contains instances for the user run | mind2:Sentence |

**Table 7:** Classes of the ontology scheme for PUC2

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Description** | **Domain** | **Range** |
| hasCollection | A property that relates the user run with the data collection status | mind2:UserRun | mind2:Collection |
| hasExperimentType | A property that relates the user run with the experiment type | mind2:UserRun | mind2:ExperimentType |
| hasItemKey | A property that relates the user run with the item keys | mind2:UserRun | mind2:ItemKey |
| hasNavigation | A property that relates the user run with the navigation data | mind2:UserRun | mind2:Text |
| hasNode | A property that indicates the nodes in a navigation instance | mind2:Navigation | mind2:NavigationNode |

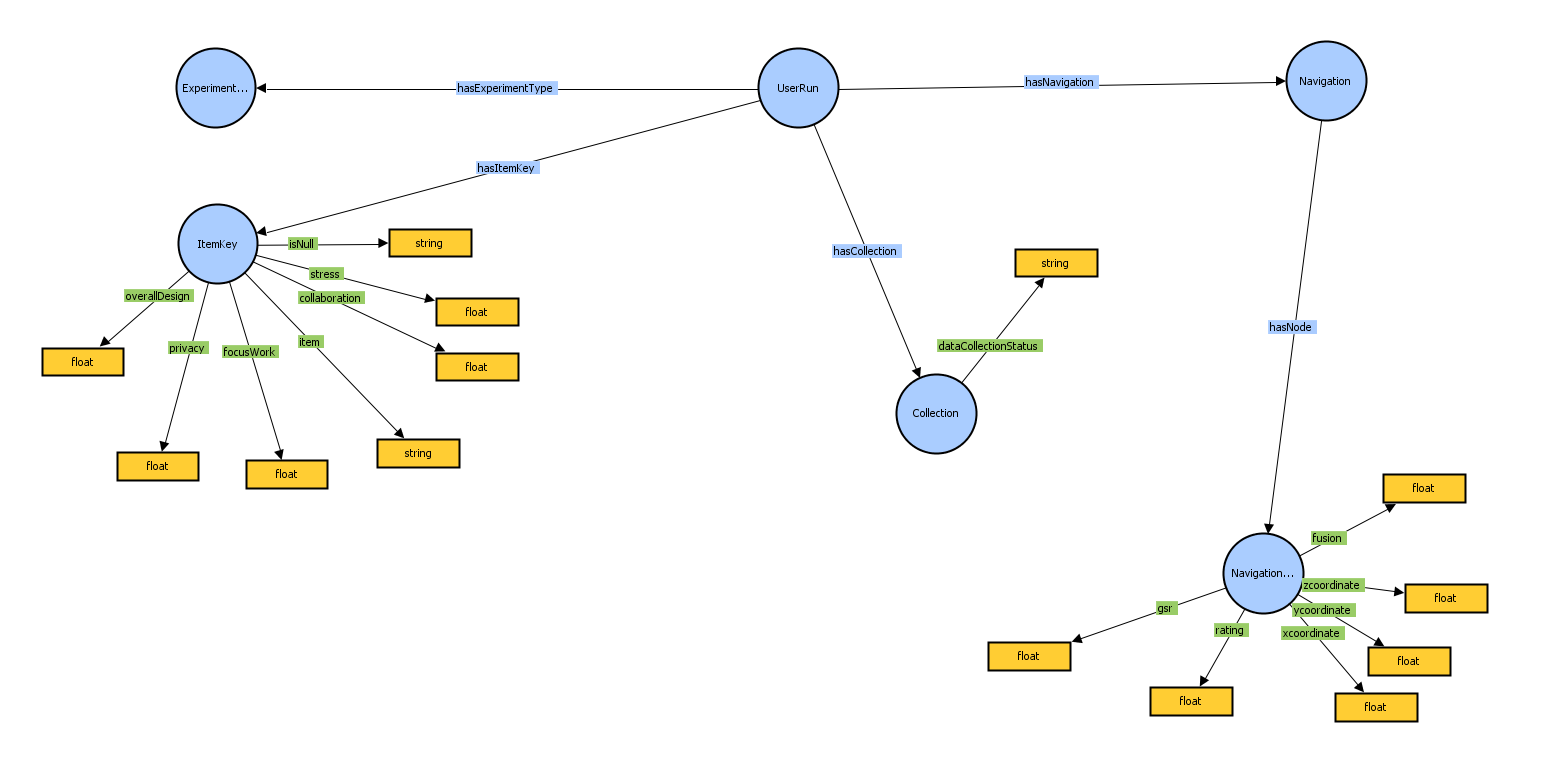
**Table 8:** Object properties of the ontology scheme for PUC2

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Description** | **Domain** | **Range** |
| collaboration | A property that shows the collaboration value of the key label | mind2:ItemKey | xsd:float |
| dataCollectionStatus | A property that shows the data collection status | mind2:Collection | xsd:string |
| focusWork | A property that shows the focus work of the key label | mind2:ItemKey | xsd:float |
| fusion | A property that shows the fusion of the navigation configurations | mind2:NavigationNode | xsd:float |
| gsr | A property that shows the gsr values of the navigation configurations | mind2:NavigationNode | xsd:float |
| isNull | A property that shows if the item key is null or no | mind2:ItemKey | xsd:string |
| item | A property that shows the item key label | mind2:ItemKey | xsd:string |
| overallDesign | A property that shows the overall design value of the key label | mind2:ItemKey | xsd:float |
| privacy | A property that shows the privacy value of the key label | mind2:ItemKey | xsd:float |
| rating | A property that shows the ratings of the navigation configurations | mind2:NavigationNode | xsd:float |
| stress | A property that shows the stress value of the key label | mind2:ItemKey | xsd:float |
| timestamp | A property that shows the timestamp | mind2:Collection OR mind2:ExperimentType OR mind2:Navigation | xsd:float |
| type | A property that shows the type/label of the navigation and the experiment type | mind2:ExperimentType OR mind2:Navigation | xsd:string |
| xcoordinate | A property that shows the x coordinate values of the navigation configurations | mind2:NavigationNode | xsd:float |
| ycoordinate | A property that shows the y coordinate values of the navigation configurations | mind2:NavigationNode | xsd:float |
| zcoordinate | A property that shows the z coordinate values of the navigation configurations | mind2:NavigationNode | xsd:float |

**Table 9:** Datatype properties of the ontology scheme for PUC2

As one can notice the namespace **mind2** was given to indicate ontology classes and properties. Next, we will give a detailed analysis of the relations between the classes and their purpose. The whole ontology scheme with the object and data type properties can be found in Figure 15. In Figure 15 the blue circles are the classes, the edges which have a domain and range classes (i.e., start and end at a blue circle) are the object type properties, and the edges which have domain a class and range a data type (i.e., start from a blue circle and end at yellow rectangle) are the data type properties.

* The **Collection** class contains information about the experiment status data change, meaning that it indicates when the user run has started, stopped, and if it goes from state ON to state OFF.
* The **ExperimentType** class contains information about the experiment type data changes, meaning that it indicates if the experiment type is a navigation task, a navigation selection, hot spot experiment, or it does not have a type.
* The **Navigation** class contains information about the navigation configuration, meaning that it contains information about the type of the configuration, and the timestamp that the node was captured. Moreover, it is related through the property **hasNode** with the class **NavigationNode** that contains information about the navigation instances (i.e., navigation nodes).
* The **NavigationNode** class contains information about the navigation nodes, such as the x, y, z coordinates of the node, the fusion score, and the gsr score.
* The **ItemKey** class contains information about the item key information, such as item key label, the collaboration, focus work, overall design, privacy, and stress scores for each item key.



**Figure 16:** Ontology scheme with object and datatype properties

* Finally, the **UserRun** class is the most important class, as it connects the information from the aforementioned classes with a user run. More specifically the UserRun class is connected: (i) with the **ExperimentType** class through the property **hasExperimentType** to indicate the experiment types that it contains, (ii) with the **ItemKey** classthrough the property **hasItemKey** to indicate the item keys that it contains, (iii) with the **Collection** class through the property **hasCollection** to give information about the collections that it contains, and (iv) with the **Navigation** class through the property **hasNavigation** to give information navigation nodes that it contains.

## Multimodal Fusion for Inspiring Workspaces Use Case

The multimodal integration mechanism was constructed after the development of the ontology scheme presented in sub section 4.2, in order to map into the ontology the messages from the visual analysis (T5.3). The ontology after its development was uploaded into a GraphDB repository. Moreover, the integration mechanism was developed using NodeJS. The messages from the visual analysis come in the form of json files and are mapped in the ontology, in other words the ontology scheme is populated with these messages. An example of a json file from the visual analysis message can be found in Example 4.

**Example 4:**

{"HotSpotExperimentData": [{"ItemData": [{"IsNull": false, "collaboration": 4.4, "focusWork": 5.0, "itemKey": "Zone\_1\_Cluster\_0", "overallDesign": 0.0, "privacy": 4.2, "stress": 1.3}, {"IsNull": false, "collaboration": 4.0, "focusWork": 4.1, "itemKey": "Zone\_1\_Cluster\_1", "overallDesign": 2.1, "privacy": 4.1, "stress": 2.2}, {"IsNull": false, "collaboration": 2.6, "focusWork": 2.8, "itemKey": "Zone\_1\_Cluster\_2", "overallDesign": 1.2, "privacy": 2.5, "stress": 2.6}, {"IsNull": false, "collaboration": 4.4, "focusWork": 4.5, "itemKey": "Zone\_1\_Cluster\_3", "overallDesign": 2.9, "privacy": 4.4, "stress": 1.9}, {"IsNull": false, "collaboration": 2.0, "focusWork": 1.8, "itemKey": "Zone\_1\_Cluster\_4", "overallDesign": 3.6, "privacy": 1.6, "stress": 0.8}], "ItemKeys": ["Zone\_1\_Cluster\_0", "Zone\_1\_Cluster\_1", "Zone\_1\_Cluster\_2", "Zone\_1\_Cluster\_3", "Zone\_1\_Cluster\_4"], "name": null, "timeStamp": null}], "configurationDesignAndNavigationRatings": [[2.2, 3.5], [2.7, 2.4], [2.6, 2.6], [3.5, 2.6], [3.6, 3.2], [3.7, 3.4], [4.0, 3.9], [4.1, 4.3], [4.3, 4.0]], "configurationRatingKeys": ["NavTest1\_con1\_concentratedAreas\_noPath", "NavTest2\_con2\_distributed\_noPath\_newFurniture","NavTest3\_con3\_distributed\_noPath", "NavTest4\_con4\_distributed\_withPath\_baseStatic", "NavTest5\_con4\_distributed\_withPath", "NavTest6\_con5\_distributed\_withPath\_withPartition", "NavTest7\_con6\_distributed\_withPath\_withWall", "NavTest8\_con7\_distributed\_withPath\_withPartition\_material", "NavTest9\_con8\_distributed\_withPath\_withPartition\_material\_plants"], "driver": 1.0, "experimentID": 1, "experimentStatusChangeData": [{"dataCollectionStatus": "Stop", "timestamp": 0.0}, {"dataCollectionStatus": "Start", "timestamp": 91.02796}, {"dataCollectionStatus": "On", "timestamp": 93.3956451}, **…** {"dataCollectionStatus": "Stop", "timestamp": 3724.323}], "experimentTypeChangeData": [{"experimentType": "None", "timestamp": 23.69223}, {"experimentType": "NavigationTasks", "timestamp": 228.092789},

**…** {"experimentType": "HotspotExperiment", "timestamp": 3533.15552}], "guardian": 0.0, "integrator": 0.0, "leastFavouriteCollabSelectionKeys": ["Cluster\_46\_meetAndCollab\_b1 (1)", "Cluster\_27\_meetAndCollab (3)", "not selected", "not selected", "Cluster\_26\_meetAndCollab", "Cluster\_26\_meetAndCollab","Cluster\_26\_meetAndCollab", "Cluster\_26\_meetAndCollab\_a1", "Cluster\_26\_meetAndCollab\_a1"], "leastFavouriteCollabSelections": [{"x": 1.990005, "y": 0.0, "z": 5.105028}, {"x": -5.476165, "y": 0.0, "z": 6.087821}, {"x": 0.0, "y": 0.0, "z": 0.0}, {"x": 0.0, "y": 0.0, "z": 0.0}, {"x": 3.81, "y": 0.0, "z": 4.24}, {"x": 3.81, "y": 0.0, "z": 4.24}, {"x": 3.81, "y": 0.0, "z": 4.24}, {"x": 3.77900028, "y": 0.0, "z": 4.27900028}, {"x": 3.77900028, "y": 0.0, "z": 4.27900028}], "leastFavouritePrivacySelectionKeys": ["Cluster\_10\_work", "Cluster\_10\_work\_2 (3)", "not selected", "not selected", "Cluster\_24\_work", "Cluster\_24\_work", "Cluster\_24\_work", "Cluster\_09\_work\_b1 (2)", "Cluster\_09\_work\_b1\_p1 (2)"],"leastFavouritePrivacySelections": [{"x": 6.08999968, "y": 0.0, "z": -0.322000027}, {"x": 3.27791572, "y": 0.0, "z": 5.18348026}, {"x": 0.0, "y": 0.0, "z": 0.0}, {"x": 0.0, "y": 0.0, "z": 0.0}, {"x": 4.04200029, "y": 0.0, "z": 6.94000053}, {"x": 4.04200029, "y": 0.0, "z": 6.94000053}, {"x": 4.04200029, "y": 0.0, "z": 6.94000053}, {"x": 4.14202642, "y": 0.0, "z": 6.358342}, {"x": 4.08192635, "y": 0.0, "z": 6.41263676}], "mostFavouriteCollabSelectionKeys": ["Cluster\_26\_meetAndCollab\_b1 (3)", "not selected", "not selected", "not selected", "Cluster\_05\_work\_nP", "Cluster\_05\_work", "Cluster\_05\_work\_a2", "Cluster\_05\_work\_a1", "Cluster\_05\_work\_a1\_p1"], "mostFavouriteCollabSelections": [{"x": -6.91, "y": 0.0, "z": 2.60999966}, {"x": 0.0, "y": 0.0, "z": 0.0}, {"x": 0.0, "y": 0.0, "z": 0.0}, {"x": 0.0, "y": 0.0, "z": 0.0}, {"x": 13.2288609, "y": 0.0, "z": 5.868676}, {"x": 13.2288609, "y": 0.0, "z": 5.868676}, {"x": 13.2288609, "y": 0.0, "z": 5.868676}, {"x": 13.299571, "y": 0.0, "z": 5.93938732}, {"x": 13.299571, "y": 0.0, "z": 5.93938732}], "mostFavouritePrivacySelectionKeys": ["unknown cluster", "unknown cluster", "not selected", "not selected", "unknown cluster", "unknown cluster", "unknown cluster", "unknown cluster", "unknown cluster"], "mostFavouritePrivacySelections": [{"x": -7.13593674, "y": 0.0, "z": -2.26849675}, {"x": -6.96, "y": 0.0, "z": -5.86057043}, {"x": 0.0, "y": 0.0, "z": 0.0}, {"x": 0.0, "y": 0.0, "z": 0.0}, {"x": -6.96, "y": 0.0, "z": -5.86057043}, {"x": -6.96, "y": 0.0, "z": -5.86057043}, {"x": -6.96, "y": 0.0, "z": -5.86057043}, {"x": -6.96, "y": 0.0, "z": -5.86057043}, {"x": 9.841255, "y": 0.0, "z": -6.98109055}], "pioneer": 1.0, "timedTasksStartingObjective": [], "totalAmountOfReachedTimeTasks": [], "navigationTasksTime": [339.691772, 337.016663, 194.98999, 228.7279, 168.514648, 196.902588, 227.228027, 207.415039, 200.0774], "navigationTasksReachTimeStamp": [[228.092789, 279.243439, 303.1812, 336.756653, 381.2947, 425.957764, 447.2705, 479.458435, 518.1714, 536.821655, 556.146851, 567.784546], **…**,[3104.57861, 3126.90381, 3171.692, 3206.01733, 3233.49268, 3241.85522, 3255.118, 3268.893, 3287.14331, 3287.156, 3296.381, 3304.656]], "userID": 349, "results": [[{"timestamp": 93.3956451, "x": 6.11500025, "y": 1.23221993, "z": -13.142, "Fusion": 0.3143229367807725, "GSR": 0.3539799895225823}, **…**, {"timestamp": 3366.781, "x": 10.4258785, "y": 1.15758216, "z": -7.717923, "Fusion": 0.428507504292409, "GSR": 0.025817681462086885}]]}

The visual analysis json message contains information about a user run in the virtual reality environment. The big picture is that the user run is restricted into 9 clusters (configurations), this is where the user can interact with the virtual environment: *NavTest1\_con1\_concentratedAreas\_noPath*, …, *NavTest9\_con8\_distributed\_withPath\_withPartition\_material\_plants*. The json file contains information for these 9 configurations, such as the navigation rating. The json also contains information about the data collection status, which indicates when the user run has started, ended, and if it goes from state ON to state OFF. The json message also contains experiment type data changes, meaning that it indicates at which timestamp the experiment type is considered a navigation task, a navigation selection, hot spot experiment, or it does not have a type. Finally, the json message contains information about the navigation nodes, such as the x, y, z coordinates of the node, the fusion score, the gsr score, and their timestamp.

## Reasoning for Inspiring Workspaces Use Case

This subchapter entails all relevant information and strategies regarding the corresponding reasoning mechanisms which were deployed for PUC 2. The reasoning mechanisms take advantage of the ontologies which were formerly created to support the use case of inspiring workspaces as well as the actual population of the knowledge base with content and metadata deriving from both artists and users.

The main idea is to return on demand to a user information about the various configurations, in which (s)he can interact in the 3D model. Information like which was the most stressful configuration, which configuration took the most time to complete, and others. The aforementioned information can be accessed through application programming interfaces (APIs) developed in node.js and JavaScript. The graphdh.js library was used to establish the connection and transactions towards and from the GraphDB repository[[6]](#footnote-6) with authorization and authentication ensured. The file delivered in response follows a scalable and dynamic approach, meaning it is being generated on demand based on live requests, thus ensuring always up-to-date data delivery as the knowledge base supports a continuous online population and new entries may arrive anytime.

In total, 8 SPARQL queries were formulated, each satisfying a different user requirement.

**Query 1:** Retrieve the average user's stress for each configuration. Meaning that for each of the 9 configurations the query will find the average stress and will return it.

**Query 2:** Retrieve the total time needed to navigate in each configuration by every user. Meaning that for each of the 9 configurations the query will find how much time all the users needed to navigate in it and return the results.

**Query 3:** Retrieve the overall design ratings for each configuration. Meaning that for each of the 9 configurations the query will find the overall design rating given by all users and will return it.

**Query 4:** Retrieve the overall design ratings for each configuration, based on the type of the user (i.e., if (s)he is a pioneer, a guardian, a driver, or an integrator).

**Query 5:** Retrieve how many users of each type exist (i.e., if (s)he is a pioneer, a guardian, a driver, or an integrator).

**Query 6:** Retrieve which clusters were classified by all users as most/least favourite based on the focus work, the privacy, and the collaboration.

**Query 7:** Retrieve which clusters were classified by each user’s type (i.e., pioneer, guardian, driver, or integrator) as most/least favourite based on the focus work, the privacy, and the collaboration.

**Query 8:** Retrieve the users stress over all configurations.

## Discussion and Future Work

In this section we presented the ontology scheme for the PUC2, and the integration mechanism that populates the ontology with messages from the visual analysis component. As mentioned in sub section 4.3 the message from the visual analysis component is in the form of json.

The ontology scheme is currently at a satisfying point as it can facilitate all the information that comes from the visual component. Therefore, the ontology scheme seems that it will not need any further development, unless the message that the integration mechanism receives adds or removes some keys in their json message. We constructed the ontology scheme based on the information that we received from the visual analysis component. For this reason, we managed to produce an ontology scheme which seems concise.

For the integration mechanism, the mapping of the message from the visual analysis component is developed in order to capture some generic exceptions (i.e., if some keys do not contain some information). But further improvement can be done, as we have not yet captured each subcase of the forms that the json message could have.

As for future work, our first goal is to construct a reasoning mechanism that will capture as many use cases as possible. On top of that, further improvement can be done on the integration mechanism by capturing more exceptions. Moreover, the ontology scheme could be enhanced with more classes and properties, according to the form of the message that we will receive and the needs of the reasoning mechanism.

# SEMANTICALLY ENHANCED INTERACTIVE 3D SPACES

This section reports the progress and final results of task T5.6 “Development of semantically enhanced interactive 3D spaces”. It presents 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 that were generated in the context of task T4.1.

MindSpaces project aimed, among other things, to develop processes that will facilitate the transformation of 3D indoor and outdoor spaces according to user responses, to artists interventions or to novel suggestions by architects and designers. In this context, the expected results are 3D models capable of adapting to Design Machine manipulation or users’ interaction. Moreover, the 3D reconstructed models should be suitable to be included in Virtual Reality experiences where the reactions of virtual visitors will be examined but also compatible with common manipulations by architects, designers and artists for redesigning space or performing “experiments”.

In the following we present custom designed Rhino’s Grasshopper scripts and tools that modify the geometry of the 3D reconstructed models of reality (from 3D scanning, image-based techniques, or the mobile mapping platform). Workflows that generate 3D CAD models from the mesh models are also presented together with details about the generation of 3D CAD models for each one of the three Pilot Use Cases. 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. In order to assist and partially automate this process, 3D semantic segmentation algorithms were implemented that assign semantic information to the reconstructed point clouds and divide them into areas of discrete semantic representations (such as objects, furniture, walls, ceilings, lights etc). For this purpose, synthetic annotation data were generated through automated processes inside Unity 3D game engine. Additionally, 2D image semantic segmentation algorithms, based on neural network architectures, were implemented, and used in order to automatically detect and remove the sky from images. This action is used as a module in a single image 3D depth prediction service that was developed in the context of T4.1 “Space sensing for 3D reconstruction”.

## Modification of reconstructed 3D geometry in Rhino Grasshopper

MindSpaces offers dedicated tools and procedures to record data for 3D reconstruction of outdoors and indoors environments (T3.3 & T3.4 respectively). The collected data are processed through the Outdoor and Indoor reconstruction services of the MindSpaces platform (T4.1). Typically, the models generated by these services are coloured 3D point clouds, or 3D photorealistic triangular mesh models. In order to directly manipulate the geometry of such models by means of applying geometric modifications, Grasshopper visual scripts were developed inside the environment of Rhino 3D CAD software.

|  |
| --- |
|  |
| **Figure 17:** Overview of Grasshopper’s canvas for modifying the geometry of 3D mesh models |

The high level workflow of the implemented grasshopper tool (Figure 17) consists of the following steps:

1. The initial 3D mesh model is linked to Grasshopper.
2. The linked 3D mesh is connected to a custom Grasshopper Python Script (*GhPython Script*) that splits the mesh into
   1. vertices
   2. 3D faces
   3. and UV coordinates.

UV coordinates 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.

1. The part of the original 3D model to be altered is selected. For the selection of a subpart of the 3D mesh, two alternative functionalities were implemented:
2. The user draws a closed curve inside the environment of Rhino 3D. Then all vertices that lie inside the curve are identified and selected.
3. 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 geometric modification is applied on the selected vertices. 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. The altered vertices are combined with the rest of the original 3D mesh model.
6. The UV coordinates of the original vertices of the 3D mesh model are assigned to the altered vertices. 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 18:** Workflow for modifying 3D mesh geometry via Grasshopper in Rhino |

## 3D reconstructed models to 3D CAD models

The aforementioned technique of modifying parts of the original 3D mesh models that are reconstructed from reality capturing techniques supported by Mindspaces, has some clear limitations. 3D mesh models from photogrammetry, 3D laser scanning and mobile mapping surveys 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 modelled 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 realise the idea of Adaptive 3D-models based on semantic reasoning, it was decided to generate 3D CAD models from the 3D triangular mesh models.

This task includes the “transcribing” of the reconstructed models into a more structured representation. In parallel, the 3D models were corrected, areas that were occluded in the original models were completed and the quality of the obtained 3D reconstructions was optimised, in areas where quality was inferior, due to limitations of the capturing technologies (highly reflective surfaces like glass, very complex objects like vegetation etc).

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 5.3). It requires a lot of effort and domain specific knowledge and skills, such as 3D and architectural design. Different approaches and workflows were followed to cover the specific needs of each Pilot Use Case. They were presented in detail in Section 3.2 of the deliverable D5.3. Here the final models of each PUC are presented and further details are given for the generation of a new CAD model of Thessaloniki Music Hall building for the needs of PUC1, that was recorded by means of a drone survey.

### PUC 1 - Outdoors urban environments (Tecla Sala & Thessaloniki Music Hall)

In T4.1, a 3D mesh model of the area around the cultural centre of Tecla Sala in the City of L’ Hospitalet, was created by combining images from a DSLR camera, 3D scans and data from the Mindspaces mobile mapping platform (Figure 18). Although the quality of the 3D mesh was in general high, several elements were not completely modelled or contained noise, since it was impossible to collect relevant data. Examples include the roof of the building, the windows of the second floor, parts of the circular bridge, as well as the vegetation.

|  |
| --- |
|  |
| **Figure 19:** Final textured 3D mesh model of PUC1 (Tecla Sala) |

Using the extracted incomplete 3D mesh model as a basis, a rich 3D CAD model was created using 3D Studio Max and Rhino 3D software tools. Unwanted objects (like people, trees etc) and stray erroneous triangles were removed and small holes were filled by using data from similar neighbouring areas. Incomplete or unmodelled areas were designed based on the existing geometry of the building and available imagery from Internet sources. Additional decorative elements (such as trees, benches, lights etc) were also added to the 3D CAD model. Figure 19 presents the final enriched 3D model that was used by Open Call artists of Mindspaces program, but also by architects and students during two workshops that were organised in the context of PUC1 and

|  |
| --- |
|  |
| **Figure 20:** Final rich 3D CAD model of PUC1 (Tecla Sala) |

A similar workflow was followed to generate a parametric and semantically rich 3D CAD model from the photogrammetric 3D mesh model of Thessaloniki Music Hall. This was a second area, situated in Greece, that was selected for PUC1 to test the generation of 3D models from drone surveys, since in the Tecla Sala area it was not permitted to perform any drone flights due to the neighbouring airport.

The workflow was performed in Rhino 3D software tool and its basic steps included:

1. Cleaning and simplifying the photogrammetric 3D mesh model (Figure 20). This process included the removal of stray triangles or erroneous noisy areas, such as trees, reflective surfaces etc.
2. Further editing was conducted to the cleaned mesh model. Certain parts that were mismodelled were designed in Rhino to achieve the optimal quality of the final model. The redesign included glass surfaces, railings, and windows.
3. Material and properties such as colour, reflectivity, opacity, and texture were assigned to the newly designed model elements.
4. Elements like plants, benches, or light fixtures that were not modelled correctly, were deleted and detailed 3D CAD models from an asset library were placed at the respective positions.
5. Finally, lighting was added to the 3D scene to achieve more realistic representation of the reconstructed 3D space in Virtual Reality environments.

|  |
| --- |
|  |
| **Figure 21:** Original photogrammetric 3D mesh model of Thessaloniki Music Hall generated in T4.1 (surface) |

|  |  |
| --- | --- |
|  | |
|  |  |
| **Figure 22:** Model improvement and completion in Rhino 3D. | |

The final 3D CAD model was inserted in Unity 3D for navigating and experiencing the area in Virtual Reality (Figure 22). The Unity model was also used to generate photorealistic synthetic images from multiple viewpoints. These, together with the original aerial views of the area from the drone mission were used in task T4.2 “Aesthetic and sentiment extraction from visual content” in an annotation task towards evaluating human sentiments on urban indoor/outdoor architectural places. More specifically, annotated synthetic and real images were used to train **image sentiment analysis algorithms** that extract the sentiment evoked by spaces depicted in real images and VR screenshots.

|  |  |
| --- | --- |
|  | |
|  |  |
| **Figure 23:** Final 3D CAD Model as seen in Virtual Reality in Unity 3D. | |

### PUC 2 - Inspiring workplaces (McNeel Office Space)

Regarding indoor spaces, in T4.1 a 3D mesh model was created for McNeel office space (Figure 23 left), 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 photo capture survey with a DSLR camera mounted on a tripod dolly. In order to generate a semantically enhanced 3D model the main building’s structural components (walls, columns, floor and ceiling) were traced and approximated by simple geometric objects. Materials and properties, such as colour, reflectivity, opacity and texture were assigned to these objects. For furniture and other movable objects like plants, lights etc, 3D models from an asset library were used. Multiple light sources were also added to the 3D model for better visualisation inside the Virtual Reality environment.

|  |  |
| --- | --- |
|  |  |
|  |  |
| **Figure 24:** Initial 3D mesh model from T.4.1 (left). Final rendered 3D CAD Model (right) | |

A set of rules for the annotation of objects inside the 3D models with the high-level concepts of the taxonomy has been also concluded with the partners of MindSpaces and added accordingly to the final enhanced 3D model. The generated model of McNeel’s office was provided to MindSpaces partners and used as a basis for the generative design tools in the context of PUC2 experiments.

### PUC 3 - Emotionally-sensitive functional interior design (Senior’s residence in Paris)

A similar workflow was used for the generation of a 3D CAD model of a senior’s residence in Paris for PUC3 (Figure 24). The final 3D model served as the basis for the design of the interior spaces in the final version of PUC3, where a Digital Village (Cap de Balloon) was constructed. Into these interior spaces virtual visitors can gather in small groups in order to discuss and interact with each other.

|  |  |
| --- | --- |
|  |  |
| **Figure 25:** Initial 3D mesh model from T.4.1 (top). Final rendered 3D CAD Model (bottom) | |

## 3D point cloud Semantic Segmentation

To assist the generation of enhanced 3D CAD models from the reconstructed 3D meshes or 3D point clouds, a valuable 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. In this context, a study of state-of-the-art algorithms for semantic segmentation on point clouds of indoor spaces was carried out and algorithms were implemented and tested on the recorded point clouds. To facilitate the training of the semantic segmentation algorithms synthetic annotation data were created from the enhanced 3D CAD models.

### State of the Art Semantic Segmentation

Semantic segmentation is the general task of assigning labels to visual or 3D data. When applied to point clouds a label is assigned to each individual 3D point and in this way a semantically enriched point cloud is generated. Although semantic segmentation initially referred to image data and assigning labels to each image pixel (see section 5.4 below), lately, research in the field of deep learning has focused on 3D space, mainly focusing, however, on structured 3D formats, i.e. voxel grids. It was not until very recently, when novel deep neural network architectures were developed to process unstructured point cloud data. **PointNet** is the most widely used network that works directly on point clouds and has been also adopted for object detection tasks. Hybrid networks that work both on points and voxels, such as **PVCNN** that has been adopted in Mind spaces, are also proposed. In the following we present the most common approaches for semantic segmentation in the literature and results on their application to a widely used dataset (**s3dis**) that is described in section 5.3.2.

**PointNet** [50] is a unified architecture whose input data are point clouds and the outputs are either a single class label for the entire input or class labels for every individual point of the input. Pointnet has a relatively simple basic network architecture. In the initial stages each point is processed identically and independently and 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 identify “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 26:** 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. |

Experiments were conducted with PointNet, on the **s3dis** dataset (see section 5.3.2). Results are shown in the following table, using the metric of **Intersection over Union (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 |

**Table 10:** Results of PointNet on s3dis dataset, using the metric of IoU

**Point-Voxel CNN (PVCNN)** [51] similar to PointNet, represents the 3D input data as point clouds to take advantage of the sparsity in order to reduce the memory footprint. However, in a subsequent step, the network employs a voxel-based convolution. For this, point coordinates are normalised before the point cloud is converted 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 is applied to aggregate the features. Similar to conventional 3D models, batch normalisation is applied together with a nonlinear activation function after each 3D convolution. Finally, devoxelization is performed. More specifically, trilinear interpolation is used to transform the voxel grids to points to ensure that the features mapped to each point are distinct.

|  |
| --- |
|  |
| **Figure 27:** 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 presented in the following table, using again the metric of Intersection over Union (IoU).

|  |  |
| --- | --- |
| **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 |

**Table 11:** Results of PVCNN on s3dis dataset.

**JSIS-3D** [52] 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 pointcloud and embedding these 3D points into high-dimensional feature vectors that allow clustering the points into object instances. More specifically, given a 3D pointcloud, 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 to 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 presented in the following table, 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 |

**Table 12:** Results of PointNet on s3dis dataset, using the metric of accuracy

Other State of the Art networks, working on point cloud semantic segmentation, include a dynamic graph CNN [53], **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 neighbours. **EdgeConv** is designed to be invariant to the ordering of neighbours, 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** [54], is 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 (colour, intensity, etc.) to learn a fine-grained labelling over the points, which can improve the initial coarse 3D-FCNN predictions. Finally, an efficient CRF implementation is used to perform the final inference.

In MindScapes **Point-Voxel CNN** was selected for point cloud semantic segmentation. Figure 27 depicts the basic pipeline of the adopted method.

|  |
| --- |
|  |
| **Figure 28:** 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 |

### State of the Art Datasets

Since most deep learning based approaches require big datasets in the training step, and the data collected in the context of MindSpaces are limited, a review of the openly available datasets was performed and the most suitable dataset was selected (namely **s3dis**). This was augmented by the data collected in MindSpaces to better fit the needs of the project.

#### S3DIS Building Parser

One of the most widely used datasets for assessing the performance of indoor semantic segmentation is the “Stanford **s3dis”** dataset. The dataset comprises 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 28 depicts certain sample scenes of the dataset. The dataset includes 13 different semantic classes.

|  |
| --- |
|  |
| **Figure 29:** Sample scenes of s3dis dataset |

#### 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 29.

|  |
| --- |
|  |
| **Figure 30:** 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 30.

|  |
| --- |
|  |
| **Figure 31:** Sample groundtruth meshes of SceneNΝ |

### 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 31 and followed a concrete workflow, which is described in Figure 32.

|  |
| --- |
|  |
| **Figure 32:** Format of S3dis dataset |
|  |
| **Figure 33:** Processing of our dataset |

Our dataset comprises 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.

|  |  |
| --- | --- |
|  |  |
|  |  |
| **Figure 34:** Semantic components of our dataset (a) floor, (b) floor and walls, (c) floor, walls and ceiling, (d) floor, walls, ceiling and sitting  To facilitate and partially automate the annotation task of the scanned pointcloud of the McNeel dataset the enhanced 3D CAD model (presented in section 5.2.2) was used. The 3D CAD model shared the same coordinate system as the original scanned pointcloud (since it was generated by it). So the two 3D models were aligned. Synthetic views of the enhanced 3D CAD model were generated in Unity 3D from selected virtual viewpoints and the semantic classes were automatically assigned to the areas of the synthetic views (Figure 34). Subsequently the scanned pointcloud was projected on the same viewpoints and the semantic class on which each point was projected was assigned to it.   |  |  | | --- | --- | |  |  | |  |  | |  |  | |  |  | | **Figure 35:** Synthetic views of the enhanced 3D CAD model of McNeel office (left). Semantic classes automatically assigned by the properties of the 3D components of the 3D CAD model. | | | |

### 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 35 and Figure 36.

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

**Table 13:** Results of modified PVCNN on McNeel room dataset, using the metric of accuracy

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **clutter** | **ceiling** | **floor** | **wall** | **beam** | **column** | **door** | **window** | **table** | **chair** | **sofa** | **Book**  **case** | **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 |

**Table 14:** Confusion matrix of predicted (rows) and ground truth (columns) labels

|  |  |
| --- | --- |
|  |  |
| **Figure 36:** Preliminary point cloud semantic segmentation on a McNeel room from modified PVCNN. Ground truth (left) vs inference results (right) | |

|  |  |
| --- | --- |
|  |  |
| **Figure 37:** Failures of the inference results. Ground truth vs inference results | |

## 2D Image Semantic Segmentation to assist Single Image Depth Prediction

In section 5.3 semantic segmentation for point clouds was presented in the context of assigning semantic information to 3D reconstructions obtained in MindSpaces. 2D image semantic segmentation was also employed on visual data, as part of a “single image” 3D reconstruction tool that was developed in MindSpaces to automatically identify pixels that correspond to sky. This tool is based on neural networks for the prediction of 3D information. It takes as input an image and outputs a depth map (depth values assigned to each pixel) and a 3D point cloud. The tool is presented in deliverable D4.2.

2D semantic segmentation refers to classifying every pixel of an input image into a fixed set of categories. The resulting image serves as a classification map that accurately describes the information presented in the input image. Semantic segmentation can significantly assist the depth-map estimation task (presented in D4.2) as a post-processing technique to reduce noise in predicted depth maps. It can also assist during the learning procedure as it offers a deeper understanding of the scenery. In the project’s scope we experimented with semantic segmentation as a post-processing step to further refine the initial results. We focused mainly on sky segmentation in order to remove the sky pixels on the predicted depth maps, leading to more crisp building and natural object edges.

For this a pre-trained semantic segmentation model was selected, **HRnet** [55]**. HRNet**, (stands for High-Resolution Net), is a convolutional neural network that can be applied in multiple tasks besides semantic segmentation, such as object detection and image classification. One main advantage is its ability to maintain high resolution representations through the whole process. It starts from a high-resolution convolution stream and high-to-low resolution convolution streams are gradually added and connected to the multi-resolution streams. In this way, repeated multi-resolution fusions are employed, by exchanging the information across parallel streams which augments the effectiveness of the approach.

A pre-trained version of **HRnet** on the widely used **ImageNet** dataset was used in the tool to identify multiple classes on each input image. The “sky” class was kept as a mask to filter out pixels of this category from the 3D depth estimation.

Results of the semantic segmentation on images of outdoors environments are presented on Table 16.

Thus, semantic sky segmentation was used as a post-processing tool to remove the inconsistencies in predicted sky regions, boosting the edges of the appearing objects. The sky segmentation module detects the sky pixels on the input image and nullifies their predicted depth values (Table 17), minimizing unnecessary noise and prediction.

|  |  |
| --- | --- |
| **Original image – complete segmentation – sky segmentation** | |
|  |  |
|  |  |
|  |  |

**Table 15:** 2D image semantic segmentation results to automatically identify the sky in images and mask it when using images for 3D reconstruction.

|  |  |  |
| --- | --- | --- |
| original image |  |  |
| without sky segmentation filtering |  |  |
| with sky segmentation filtering |  |  |

**Table 16:** Example of application of semantic segmentation for sky removal during depth prediction.

Sky segmentation masking proved to aid the most in sharpening the edges of the depicted objects. Moreover, the sky segmentation technique is a robust method to crop out the sky pixels when projecting the image into a 3D format. Thus, the resulting 3D objects appear realistic without cluttered sky information.

# 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] Hepp, M. (2008). Goodrelations: An ontology for describing products and services offers on the web. In International conference on knowledge engineering and knowledge management (pp. 329–346). Springer.

[27] Radinger, A., Rodriguez-Castro, B., Stolz, A., & Hepp, M. (2013). Baudataweb: the austrian building and construction materials market as linked data. In Proceedings of the 9th International Conference on Semantic Systems (pp. 25–32). ACM.

[28] Wagner, A., & Rüppel, U. (2019). Bpo: The building product ontology for assembled products. Proceedings of the 7th Linked Data in Architecture and Construction workshop (LDAC 2019), Lisbon, Portugal.

[29] Sanfilippo, E. M. (2018). Feature-based product modelling: an ontological approach. International Journal of Computer Integrated Manufacturing, 31 , 1097–1110.

[30] Bhattacharyya, R., Bhuyan, Z., & Hazarika, S. M. (2016). O-pro: An ontology for object affordance reasoning. In International Conference on Intelligent Human Computer Interaction (pp. 39–50). Springer.

[31] Vassiliades, A., Bassiliades, N., Gouidis, F., & Patkos, T. (2020). A knowledge retrieval framework for household objects and actions with external knowledge. In International Conference on Semantic Systems (pp. 36–52). Springer, Cham.

[32] Gouidis, F., Vassiliades, A., Patkos, T., Argyros, A., Bassiliades, N., & Plexousakis, D. (2019). A review on intelligent object perception methods combining knowledge-based reasoning and machine learning. In Proceedings of AAAI Spring Symposium on Combining Machine Learning and Knowledge Engineering in practice, March 23-25, 2020, Stanford University, Palo Alto, California, USA.

[33] Liao, Y.-H., Puig, X., Boben, M., Torralba, A., & Fidler, S. (2019). Synthesizing environment-aware activities via activity sketches. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 6291– 6299).

[34] Puig, X., Ra, K., Boben, M., Li, J., Wang, T., Fidler, S., & Torralba, A. (2018). Virtualhome: Simulating household activities via programs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 8494–8502).

[35] Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., & Hellmann, S. (2009). Dbpedia-a crystallization point for the web of data. Web Semantics: science, services, and agents on the world wide web, 7 , 154–165.

[36] Nickel, M., Murphy, K., Tresp, V. & Gabrilovich, E. (2015). A review of relational machine learning for knowledge graphs. Proceedings of the IEEE, (1)104, 11-33.

[37] Sap, M., Le Bras, R., Allaway, E., Bhagavatula, C., Lourie, N., Rashkin, H., Roof, B., Smith, N. A., & Choi, Y. (2019). Atomic: An atlas of machine commonsense for if-then reasoning. In Proceedings of the AAAI Conference on Artificial Intelligence (pp. 3027–3035). volume 33.

[38] Navigli, R., & Ponzetto, S. P. (2010). BabelNet: Building a very large multilingual semantic network. Proceedings of the 48th annual meeting of the association for computational linguistics,(pp 216-225).

[39] Fellbaum, C. (2010). Wordnet. Theory and applications of ontology: computer applications, (pp. 231–243).

[40] Liu, H., & Singh, P. (2004). Conceptnet—a practical commonsense reasoning toolkit. BT technology journal, 22 , 211–226.

[41] T. Rebele, F. Suchanek, J. Hoffart, J. Biega, E. Kuzey, G. Weikum, Yago: A multilingual knowledge base from wikipedia, wordnet, and geonames, in: International semantic web conference, Springer, 2016, pp. 177–185.

[42] N. Tandon, G. De Melo, G. Weikum, WebChild 2.0: Fine-grained commonsense knowledge distillation, in: Proceedings of ACL 2017, System Demonstrations, 2017, pp. 115–120.

[43] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J., (2008). Freebase: a collaboratively created graph database for structuring human knowledge. Proceedings of the 2008 ACM SIGMOD international conference on Management of data.

[44] Best, B. J., Gerhart, N., & Lebiere, C. (2010). Extracting the ontological structure of OpenCyc for reuse and portability of cognitive models. Proceedings of the 17th Conference on Behavioral Representation in Modeling and Simulation.

[45] 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.

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

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

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

[49] 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.

[50] 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).

[51] 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).

[52] 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).

[53] 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.

[54] 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.

[55] Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., ... & Xiao, B. (2020). Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, *43*(10), 3349-3364.

1. Lynch, Kevin (1960). The Image of the City. The MIT Press [↑](#footnote-ref-1)
2. https://www.dbpedia.org/ [↑](#footnote-ref-2)
3. https://graphdb.ontotext.com/ [↑](#footnote-ref-3)
4. https://nodejs.org/en/ [↑](#footnote-ref-4)
5. http://160.40.52.169:6161 [↑](#footnote-ref-5)
6. http://160.40.52.59:8200 [↑](#footnote-ref-6)