A Creative Data Ontology for the Moving Image Industry

A thesis submitted by

Christos Angelos Dexiades

for the degree of Doctor of Philosophy

of the

Royal Holloway, University of London

2023

Declaration

I, Christos Angelos Dexiades, hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Signed ..... (Christos Angelos Dexiades)

Date:

This thesis is dedicated to the memory of my brother, Christakis (21st February 1981 - 17th January 2019), whom I wish was here for this.

Abstract

The moving image industry produces an extremely large amount of data and associated metadata for each media creation project, often in the range of terabytes. The current methods used to organise, track, and retrieve the metadata are inadequate, with metadata often being hard to find. The aim of this thesis is to explore whether there is a practical use case for using ontologies to manage metadata in the moving image industry and to determine whether an ontology can be designed for such a purpose and can be used to manage metadata more efficiently to improve workflows. It presents a domain ontology, hereby referred to as the Creative Data Ontology, engineered around a set of metadata fields provided by Evolutions, Double Negative (DNEG), and Pinewood Studios, and four use cases. The Creative Data Ontology is then evaluated using both quantitative methods and qualitative methods (via interviews) with domain and ontology experts.

Our findings suggest that there is a practical use case for an ontology-based metadata management solution in the moving image industry. However, it would need to be presented carefully to non-technical users, such as domain experts, as they are likely to experience a steep learning curve. The Creative Data Ontology itself meets the criteria for a high-quality ontology for the sub-sectors of the moving image industry domain that it provides coverage for (i.e. scripted film and television, visual effects, and unscripted television) and it provides a good foundation for expanding into other sub-sectors of the industry, although it cannot yet be considered a "standard" ontology. Finally, the thesis presents the methodological process taken to develop the Creative Data Ontology and the lessons learned during the ontology engineering process which can be valuable guidance for designers and developers of future metadata ontologies. We believe such guidance could be transferable across many domains where an ontology of metadata is required, which are unrelated to the moving image industry. Future research may focus on assisting non-technical users to overcome the learning curve, which may also also applicable to other domains that may choose to use ontologies in the future.

Acknowledgements

I would like to express my gratitude to everyone who made my PhD project experience and this thesis possible. First and foremost I would like to take the opportunity to thank the four members of my atypically large supervisory team who have guided and supported my work on this thesis: Professor John Ellis, Dr Claude Heath, Dr Giorgios Koutsoukos, and Professor Carlos Matos. I will be forever grateful to them for taking a chance on me.

Of course, I would not have been able to pursue a PhD at all without the funding provided by Royal Holloway, University of London as part of its contribution to the StoryFutures Creative Industries Clusters Programme (AH/S002758/1). I also would not have been able to complete my PhD project without practitioners from our industry partner companies sharing their expert domain knowledge and my interview participants providing invaluable feedback that I used to evaluate and improve the ontology.

On a more personal level, I would like to thank my parents Maria and Kosta, my brother Pany, and my nouna Militsa for providing encouragement and support throughout my education. I knew I wanted to complete a PhD from the moment I knew what one was, and my father in particular consistently encouraged me to pursue this ambition.

Finally, I would like to thank the members of the Royal Holloway Writing Society between 2019 to 2023 for ensuring I had a social life during my PhD!

Contents

List of Figures

List of Tables

List of Abbreviations

DFAP	Deep Film Access Project
DOI	Digital Object Identifier
GUI	Graphical User Interface
IMDB	Internet Movie Database
KIF	Knowledge Interchange Format
KMS	Knowledge Management System
OOPS	Ontology Pitfall Scanner
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
RQL	Resource Query Language
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VFX	Visual Effects
XML	Extensible Markup Language

Project Outputs

Publications

1. C. A. Dexiades and C. P. R. Heath. Creative Data Ontology: 'Russian Doll' Metadata Versioning in film and TV Post-Production Workflows. In E. Garoufallou and M.-A. Ovalle-Perandones, editors, *Metadata and Semantic Research*, pages 204–215, Cham, 2021. Springer International Publishing.

Other Deliverables

1. Version 5.4.23 of the Creative Data Ontology, which can be accessed using its DOI or via the link to its GitHub repository:

   - `https://zenodo.org/badge/latestdoi/662246329`
   - `https://github.com/cadexiades/PhD-Deliverables`

   This is the latest version of the ontology at the time of submitting this thesis, and was last updated on 20/07/2022.

2. A video explaining the fundamentals of ontologies, which can also be accessed using its DOI or via the link to its GitHub repository:

   - `https://zenodo.org/badge/latestdoi/662246329`

- https://github.com/cadexiades/PhD-Deliverables

This video was created to introduce non-technical domain experts to ontologies before they participated in interviews to evaluate the Creative Data Ontology.

Chapter 1

Introduction

The media industry is a vital part of the worldwide economy as it accounts for a significant amount of revenue. According to the Motion Pictures Association, the global market for theatrical and home/mobile entertainment reached $101.0 billion in 2019 [1]. Meanwhile, the UK film industry had an annual turnover of £17 billion in 2018, which is an increase of 129.7% from £7.4 billion from 2013 and in 2018, the post-production sub-sector accounted for £1.8 billion (10.6%) of the annual turnover [2, 3].

The media creation workflow is formed of three stages: *pre-production*, *production*, and *post-production*. A media item is designed and planned during pre-production and then captured during the production stage. During post-production, material from various sources are combined to form a final deliverable media item. Depending on the type of project, the boundaries between the three stages can blur. For example, when producing movies that are entirely computer generated, production and post-production are merged together. Nevertheless, metadata plays a crucial role in each stage of the process [4]. This is because metadata serves both as documentation for the media creation process and as a means of retrieving information more efficiently than searching for it manually. Figure 1.1 illustrates a typical media creation workflow for films.

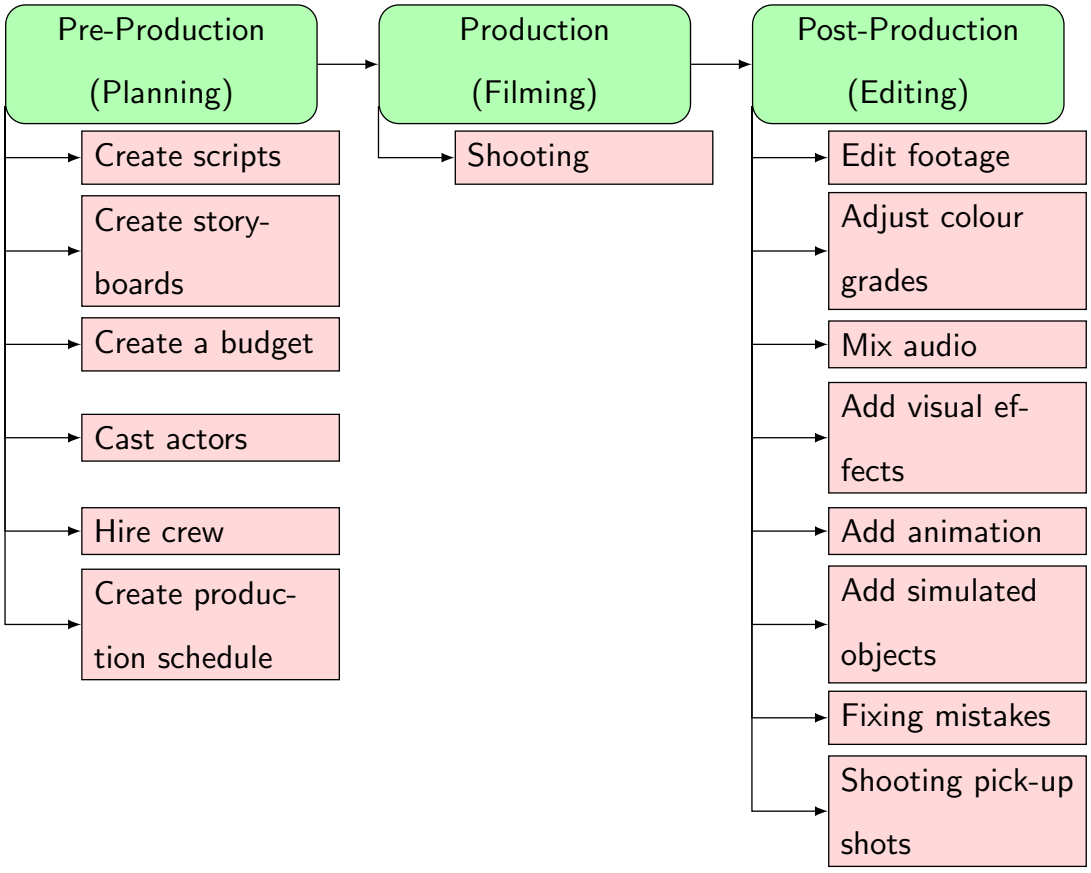


Figure 1.1: An example workflow for a producing a film [5].

In 2018, there were 2,915 post-production companies in the UK alone [3] although the majority are small and medium-sized enterprises with insufficient staff time to devote to developing solutions to deal with the challenge of managing large quantities of data. Blat et al. (2015) state that "the amount of data captured onset for film production is vastly increasing" and that currently, a high-end film generates several terabytes of data per day [6]. Atkinson et al. (2014) state that individual feature films usually have, in total, tens of terabytes of data associated with each film [7]. As stated in [7], within these datasets there are:

> "Multiple interrelated streams of data [that] are generated during the film-making process. These are currently stored separately, and their initial rich interrelations and links are lost."

Atkinson et al. [7] and Blat et al. [6] published these estimated figures in 2014 and 2015 respectively. Hence, it is very likely that the amount of data managed by media companies has increased since then.

Although no media items are created during the pre-production stage, a large quantity of important metadata is produced. Such metadata includes scripts, sketches of scenes, and photographs of shooting locations. The main output of the production stage are media items such as video material, sound recordings, and animations, but a large quantity of metadata are also created. For instance, the date and time of shoots and details of the sets. In post-production, example tasks include editing shots, colour correction, and special effects. While this stage typically consumes metadata, it also creates metadata in the form of descriptions of editing operations [4]. Metadata can come from a number of sources, and can either be automatically generated using equipment or be manually generated. Metadata capture devices includes light detection and ranging (LIDAR) scanners, spherical cameras, still cameras, video cameras, and RGB-D cameras (depth sensors) [6].

As mentioned above, the entirety of a dataset for a single feature film title usually consists of tens of terabytes. This, coupled with the recent increase in scale and complexity of the media creation process, has resulted in the emergence of a new on-set role – data wrangler – to manage the camera-generated data alone [7]. Smith and Schirling (2006) state that metadata was "invented to help computer systems and humans more efficiently and effectively organise, access, and interpret data" [8]. However, there is now a need for a more efficient and more effective means of managing the metadata (in addition to data) to ensure that it can be disseminated and used across stages in the media creation workflow, as required.

1.1 Research and Practice Gap

Following ethnographic research of industry practitioners from our partner companies – Evolutions, Double Negative (DNEG), and Pinewood Studios – performed by other members of the StoryFutures project, and a review of the existing literature conducted by the author, the following two key areas were identified in which research and practice gaps exist with respect to the use of semantic web technologies to optimise the storage and retrieval of metadata during media creation workflows.

1. The current practice for data management in the media industry is to use a relational database instead of semantic web technologies, such as an ontology. Although a relational database can be generated directly from an ontology, we must note that the issue lies with the fact that there is a lack of relevant ontologies for the media creation domain. This results in a plethora of databases being created which causes a number of other issues, such as problems with integration. For example, there is often a requirement for one-to-one integration from one database to another, which is a cumbersome task. Moreover, there is often a notable lack of common semantics in the data elements that those databases store and manage.

   While this is not an observation that is unique to the media creation domain and although the elements in ontologies can be thought of as synonymous with elements in a relational database, ontologies and databases have different goals and characteristics. For example, databases place little emphasis on a generalisation to specialisation hierarchy yet this is a key feature of ontologies. Furthermore, a data dictionary is generally kept separately from the database schema it is describing but in the case of ontologies, such annotations are usually part of the ontology itself [9]. A data dictionary is a collection of descriptions of the data items within a data model. For example, a data dictionary for a database would store information about the tables in the database (i.e. its fields' names, data types, field length, constraints applied to

the field, and a natural language description). Meanwhile, a data dictionary for an ontology would be in the form of annotations, which are explained further in Section 2.4.5. Further differences between databases and ontologies are explained in Section 2.6.2.

Given the clear differences between databases – the status quo industry metadata management solution – and ontologies, exploring the applicability of ontologies in the media industry should be considered.

2. Whereas there are a number of ontologies that are designed to capture the media domain (see Section 3.1), many of these focus only on the finished media product rather than the metadata produced during the creation process. Those that do focus on the creation process have a limited scope, often focusing on film projects despite there being a wealth of other sub-domains that could also be encapsulated, such as scripted television productions, unscripted documentaries, and visual effects (VFX).

1.2 Research Questions

Aiming to address those research and practice gaps identified in Section 1.1, two research questions arise, which we discuss below. In order to answer Research Question 2 more thoroughly, a further three sub-questions will also be addressed:

- **Research Question 1**: How can ontologies be used to manage metadata in the moving image industry? Is there a practical use case and in what sense?
- **Research Question 2**: Can an ontology be designed and be used to manage metadata more efficiently to improve workflows?
  - **Sub-Question 2.1**: What would the concrete design of such an ontology look like? What are the stages involved and the lessons learned in the process of developing such an ontology?

  - **Sub-Question 2.2**: What are the quality characteristics of such an ontology? To what extent does it meet the criteria for a high-quality ontology?

  - **Sub-Question 2.3**: Can this ontology act as a potentially "standard", widely-reusable metadata management ontology in the moving image industry?

1.3 Research Objectives

This thesis aims to investigate the applicability of using ontologies for metadata management in the moving image industry, a subset of the media industry. In this context, the main objectives of this research are to:

- **Objective 1**: Conduct a critical literature survey of ontologies for the media creation domain.

- **Objective 2**: Study the notion of ontology-based knowledge management systems to determine how ontologies can be used for knowledge management.

- **Objective 3**: Design a set of use cases common across the moving image industry and identify the metadata fields necessary to represent them.

- **Objective 4**: Select an ontology design language and an ontology engineering tool to create a domain ontology in.

- **Objective 5**: Construct a domain ontology of metadata fields related to moving image creation processes.

- **Objective 6**: Design an ontology evaluation framework centred around the use cases and the ontology quality criteria identified in the literature.

- **Objective 7**: Assess the constructed domain ontology against the evaluation framework.

- **Objective 8**: Define a reusable methodological process for developing an ontology for the moving image industry based on the lessons learned during this project.

1.4 Contributions

This thesis explored the field of semantic web technologies, specifically ontologies, and how they could be used in an ontology-based knowledge management system to optimise the storage and retrieval of metadata during the moving image creation process. This has led to the following contributions to the field:

- **Contribution 1**: The development of a domain ontology for the moving image industry, centred around four main use cases. The use cases and ontology itself were developed through an iterative and incremental cycle of development and feedback with our industry partners. (RQ 1, RQ 2.2)

- **Contribution 2**: A qualitative evaluation of the domain ontology during which both domain experts and ontology experts were consulted for feedback via semi-structured interviews. (RQ 2.2)

- **Contribution 3**: A quantitative evaluation of the domain ontology during which the ontology was evaluated for logical consistency and conciseness using automated tools and completeness through a count of metadata fields in the list provided by our partners compared to those in the ontology. (RQ 2.2)

- **Contribution 4**: A definition of a methodological process for designing and building a domain ontology for the moving image creation domain. This process is based on that undertaken to develop the Creative Data Ontology and may be reusable by other entities who may choose to follow it in order to develop an alternative ontology for the moving image industry. (RQ 2.1, RQ 2.3)

- **Contribution 5**: A literature review summarising the state-of-the-art in ontologies for

the media domain, where we identified two categories of ontology: those that focus only on the finished media product and those that focus on the metadata produced during the creation process.

- **Contribution 6**: A literature review summarising the use of ontologies in knowledge management systems. Given that the combination of the media domain and ontology-based knowledge management systems is a niche area, the scope of this review was ontology-based knowledge management systems for any domain.

1.5 Personal Motivation

The research topic that one chooses to focus on for their PhD project is typically influenced by a combination of factors unique to them. In this case, the factors that influenced the direction of this project are academic background, personal interests, and the resources available at the time of applying for a PhD position back in the first half of 2019.

At the time, I was pursuing my Master's degree in Machine Learning at Royal Holloway, University of London, having also completed my Bachelor's degree in Computer Science there a year earlier. At this stage of my life, I knew two things: firstly, I wanted to pursue a PhD in Computer Science and secondly, I had a strong personal interest in creative writing with aspirations to eventually become a published author. It was then that I realised that I wanted my PhD project to combine my academic background in computer science with my interest in the creative arts.

Once I had decided this, I began to read academic papers about the applications of computer science to creative domains. I also searched for possible supervisors and sources of funding to support my project. It was the search for funding that resulted in a specific topic being chosen: as I searched online for funded projects that combine computer science with the arts, I encountered six adverts for pre-defined interdisciplinary projects, all based within the StoryFutures project at Royal Holloway. I found that one of these projects was the

perfect match for the direction I wanted my project to take and it provided the resources to support support my project, including a research budget for three years and access to industry practitioners. For the last 3.5 years, I have explored the application of ontologies within the moving image industry and this thesis is the result of this journey.

1.6 Thesis Outline

In Chapter 1, we have outlined the motivation and aims of this thesis. The remainder of the thesis is structured as follows:

- **Chapter 2**: We start this thesis by presenting the relevant introductory information about ontologies that is needed to understand our contributions. Section 2.1 places ontologies in the context of the wider field of Semantic Web technologies. We then explain the different types of ontology in Section 2.2 before comparing the various ontology design languages in Section 2.3. Following on from this, Section 2.4 outlines the five types of component an ontology is made up of and how they work with each other. We then describe the different ontology querying languages in Section 2.5 before concluding the chapter with an exploration of the applications of ontologies for knowledge management in Section 2.6.

- **Chapter 3**: The next chapter in the thesis will review the existing state-of-the-art research into domain ontologies for media creation and archival and applications of ontology-based knowledge management systems for various domains. Section 3.1 outlines existing ontologies for media creation and archival and categorises them into two groups based on whether they handle metadata throughout the production process or only at the end. Section 3.2 presents past applications of ontology-based knowledge management systems. This section is not limited to the media or (post-) production domains because the research into such systems for media creation is extremely scarce.

- **Chapter 4**: The next chapter is focused on existing methodologies for designing and evaluating ontologies. Section 4.1 details nine methodologies defined in the literature for designing ontologies before describing and comparing existing ontology engineering tools in Section 4.2. Section 4.3 outlines eight approaches for evaluating ontologies and then Section 4.4 outlines the criteria that these approaches can be used to evaluate an ontology against. Finally, Section 4.5 describes five tools for automatically evaluating ontologies.

- **Chapter 5**: This chapter describes the methodology taken to conduct the research contained in this thesis. This chapter begins with a description of the three types of research method and the selection made for this study in Section 5.1. Section 5.2 will provide important contextual information about our three industry partner companies while Sections 5.3 and 5.4 will present the methods used for data collection and for analysing that data, respectively. Following from this, Section 5.5 describes and justifies the tools selected for designing the ontology before continuing with a discussion on ethical issues considered for this project in Section 5.6. Finally, this chapter concludes with a discussion of the impact of the COVID-19 pandemic on this project in Section 5.7.

- **Chapter 6**: This chapter describes the process of designing the Creative Data Ontology for organising metadata in the moving image industry. Section 6.1 presents three main 'pain points' identified by representatives from our partner companies as the most important problems faced by practitioners working in the moving image industry. Section 6.2 describes the five iterations of the Creative Data Ontology while Section 6.3 describes the four use cases that the ontology was built to address.

- **Chapter 7**: The next chapter explains the process and results of evaluating the quality of the Creative Data Ontology. The first section – Section 7.1 – presents the evaluation framework for the ontology including justification for including or excluding

each evaluation criteria. Section 7.2 presents the results, question-by-question, of the interviews conducted with domain and ontology experts in accordance with the evaluation framework. Section 7.3 presents the results of conducting quantitative assessments on the Creative Data Ontology before concluding the chapter with a note in Section 7.4 about how encoding bias has been minimised.

- **Chapter 8**: This penultimate chapter presents a discussion of the results shown in Chapter 7. Section 8.1 discusses the findings related to the viability of the Creative Data Ontology in the moving image industry while Section 8.2 explains the lessons learned with respect to designing an ontology of metadata for the moving image industry. Section 8.3 discusses the findings in relation to the quality of the ontology while the final set of findings are shown in Section 8.4 and are focused on the possible extensions that could be made to the ontology that were beyond the original scope. Finally, this chapter concludes with a discussion of the limitations and delimitations of the study in Section 8.5.

- **Chapter 9**: The main part of this thesis ends with with an explanation of the contributions of this thesis in relation to the research sub-questions in Section 9.1 followed by a discussion of the impact of the work contained in this thesis in Section 9.2, and lastly with suggestions for future work in Section 9.3.

- **Appendix A**: The first appendix consists of a list of classes and properties that form the Creative Data Ontology. Section A.1 contains a list of classes and their definitions while Sections A.2 and A.3 contain a list of object and data type properties respectively, and their domains and ranges. Section A.4 contains a visualisation of the Creative Data Ontology to illustrate its size and complexity and Section A.5 describes how to deploy the Creative Data Ontology in an industry setting.

- **Appendix B**: The second appendix contains a copy of the slides decks used for the ontology evaluation interviews. The first set of slides was used with domain experts

and second set was used with ontology experts.

Part I

Background

Chapter 2

An Introduction to Ontologies

Ontology is a term that was originally used in philosophy to refer to the study of existence, but it has since been used by Thomas Gruber to refer to "an explicit specification of a conceptualization" [10] In this context, a "conceptualization" refers to an abstract model of some domain while an "explicit specification" means that the model should be specified in a language that can be interpreted by both humans and computers. Gruber's definition of an ontology is the most quoted by researchers in Computer Science and has led to many adaptations [11]. Borst (1997) defined ontologies as "a formal specification of a shared conceptualisation" [12] and Studer et al. (1998) combined these to form a new definition of an ontology as "a formal, explicit specification of a shared conceptualisation" [13]. More specifically, ontologies "formally define the entities (concepts) of a domain, their attributes and the relationships among them, in a machine interpretable way" [14]. This chapter acts as a primer to ontologies, beginning with Section 2.1 which places the concept of an ontology in context amongst other semantic web technologies. This is followed by an outline of the different types of ontologies in Section 2.2 and a description of the main ontology design languages in Section 2.3. It is succeeded by an explanation of the five components that form an ontology in Section 2.4 and then a description of the families

of query languages in Section 2.5. Finally, this chapter ends with an explanation of the possibilities and limitations of using ontologies for knowledge management, along with a comparison of ontologies and databases in Section 2.6.

2.1 The Semantic Web

The World Wide Web was invented by Sir Tim Berners-Lee, while he was working at CERN, in order to provide a means of managing and sharing technical information within the organisation [15]. While the invention of the Web solved the problems faced at CERN and has since been expanded for use globally to share an enormous amount of information, it still has certain limitations in terms of accessing data. In the "standard Web", documents are indexed and retrieved via a string-based matching algorithm according to a given search term [15]. This is problematic for search terms with multiple meanings such as "Paris", which could refer to the capital city of France, a town in Texas, USA or in Ontario, Canada, any celebrity whose name is Paris, or even Paris from the Trojan War in Greek mythology.

Berners-Lee et al. (2001) states that "the Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation" [16]. In other words, semantic web technologies are used to extend the World Wide Web to provide machine-interpretable meanings to web resources which can then be used to find information that is more relevant for a specific search request. The semantic web is more concerned with the meanings of data rather than the structure, which is a focal point of the World Wide Web. Ontologies are an example of the applications of semantic web technology, which enable the meanings of concepts and the relationships between them to be understood by software-based reasoners, which enhances the quality of search results when querying the Web [15]. However, it is important to note that ontologies can also be used on a smaller scale within individual organisations' ontology-based knowledge management systems.

2.2 Classification of Ontologies

According to Nicola Guarino (1998), ontologies can be classified into four different types, based on their level of generality: top-level ontologies, domain and task ontologies, and application ontologies [17]. The relationship between these four types of ontology is pictured in Figure 2.1, where the arrows represent specialisation relationships, and are described in more detail in the following subsections.

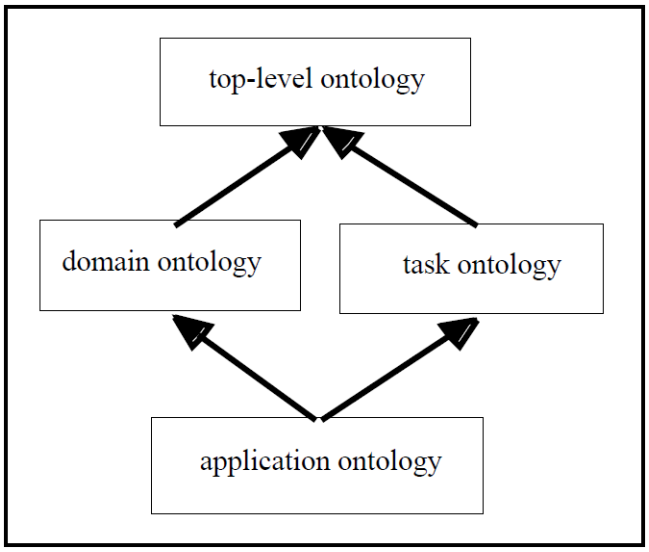


Figure 2.1: The four types of ontology [17].

2.2.1 Top-Level Ontologies

Also known as foundational ontologies, top-level ontologies describe very general concepts such as time, events, space, causality, behaviour, etc., which are independent of a specific domain or problem [18]. They aim to provide a means through which separate systems can work with a common knowledge base. Therefore, it is perfectly reasonable to have top-level ontologies that are used by large communities of users from different domains and with different requirements. Panorea Gaitanou (2009) identifies the most commonly used

upper ontologies in the literature as SUMO[1], DOLCE[2], and the Upper Cyc Ontology[3] [19].

Top-level ontologies are necessary when an ontology is being developed that needs semantic interoperability with other ontologies across different domains. For example, consider a set of enterprise systems as shown in Figure 2.2. There could be a system for the Human Resources department used to manage personnel and there could be another system for the IT department to manage access to databases [20].

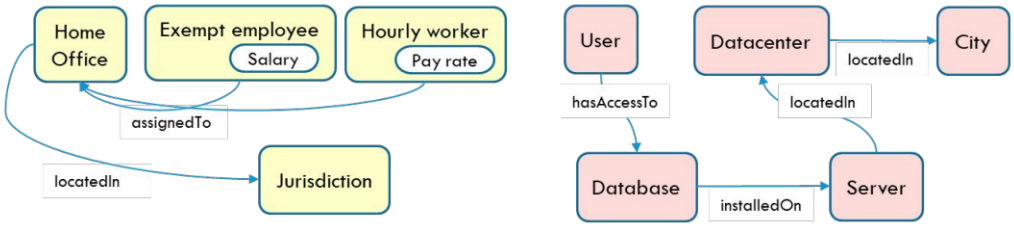


Figure 2.2: Example of two disjoint enterprise systems converted into ontologies [20].

Common terms within the two disjoint ontologies, such as `Building` and `Personnel`, can be extracted into a top-level ontology, which can then be connected to the two department-specific ontologies. This is shown in Figure 2.3.

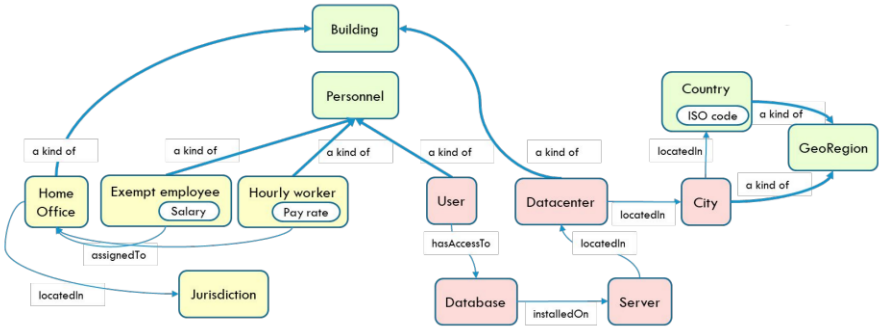


Figure 2.3: Example of two disjoint ontologies connected via a top-level ontology [20].

[1] `https://www.ontologyportal.org`
[2] `http://www.loa.istc.cnr.it/dolce/overview.html`
[3] `https://cyc.com`

It is important to note that joining two existing ontologies via a top-level ontology is not trivial, as there are a number of tasks involved with this process including agreeing terminology between the different ontologies, agreeing on or adopting methods for associating resources to an identifier, and updating the ontology schemata to link to the top-level ontology. Additionally, linking an ontology to a top-level ontology (or, in fact, any other ontology) means agreeing to accept all assertions and inferences in the top-level ontology being connected to, not just part of it. The best approach is therefore not to retrofit ontologies together but to instead begin with the top-level ontology [20].

Despite these additional tasks, using top-level ontologies is advantageous because it makes it easier to bring new information to an ontology and facilitates the development of modular, reusable ontologies. While top-level ontologies are not necessary per se, they can be very useful for certain ontology development projects [20].

2.2.2 Domain Ontologies

Domain ontologies describe the terminology related to a generic domain (such as medicine or media creation) by specialising the terms introduced in the top-level ontology. Domain ontologies offer a common language for sharing and standardising knowledge between domain experts, such as medical researchers or production and post-production practitioners. Common domain ontologies include The Gene Ontology[4] for biology and the ontologies explored in Section 3.1 for media creation and archival.

2.2.3 Task Ontologies

Task ontologies describe the terminology related to a generic task (such as diagnosing or watching) by specialising the terms introduced in the top-level ontology. Task ontologies include generic names (e.g. plan, goal, and constraint), generic verbs (e.g. assign, classify, and select), and generic adjectives (e.g. assigned) [18].

[4] http://geneontology.org

2.2.4 Application Ontologies

Application ontologies describe the vocabulary needed for an application in terms of a particular domain or task. Therefore, application ontologies are often specialisations of both a domain and a task ontology. The concepts within an application ontology often correspond to the roles played by the concepts in the domain ontology while performing a certain task.

2.3 Ontology Design Languages

As defined by both Borst (1997) and Studer et al. (1998), an ontology is formal so it must be expressed in a formal language to ensure it is machine interpretable. There are two categories of ontology design language, those based on description logic and those based on first-order logic. There are also several ontology languages, the main languages being RDF, RDFS, and OWL, which are all based on description logic [21, 22] and KIF, which is based on first-order logic. This section will explore the features of these four languages.

2.3.1 The Resource Description Framework (RDF)

RDF is a data model based on making statements about resources expressed as a *subject – predicate – object* triple, where the subject and object are both resources and the predicate, also a resource, describes a relationship between them. A resource can be thought of as anything that someone would like to talk about (e.g. books, places, people, etc.). RDF data models can be represented in XML syntax and a set of RDF triples can be visualised as a directed graph where each triple is a connection from a subject to an object with the edge between them representing the predicate [21, 22]. RDF is very limited in its capabilities as it can only use binary properties. This is problematic because predicates that accept more than two arguments are often required when modelling real-world domains. Although an ontology developer could successfully use multiple binary predicates to model a single non-binary predicate, this solution is unnecessarily complex [22]. Given that RDF forms the

foundation of the two other most commonly used ontology design languages – RDFS and OWL – this is a commonplace issue in ontology design.

2.3.2 The Resource Description Framework Schema (RDFS)

While RDF can represent relationships between two resources, RDFS extends this model by enabling the expression of class and property hierarchies and inheritance. Classes and properties become more specialised as they appear deeper in a hierarchy and they inherit the features of their parents [21, 22]. Although this is more expressive than RDF, RDFS still does not support many constructs including equivalence, inverse relations, restrictions, and cardinality constraints. RDFS also lacks the notion of contradiction. For example, if the ontology states that the range of an object property is both an item of food and an item of clothing, the reasoner will assume that all values of that property are both an item of food and an item of clothing. RDFS cannot represent class disjointness so it cannot tell that no such values exist for a property with this range [23]. Therefore, there is a need for richer ontology language without the above limitations. This is where the Web Ontology Language (OWL) is introduced but with the caveat that there is often a trade-off between efficiency and expressiveness.

2.3.3 The Web Ontology Language (OWL)

RDFS can express more ontological knowledge than RDF but still has limitations. OWL is a richer and more expressive language, which is an extension of RDFS. It provides the ability to declare range restrictions that apply only to some classes, declare classes as disjoint, implement Boolean combinations of classes using union, intersection and complement, add cardinality restrictions, and give properties special characteristics, such as transitivity [22]. Antoniou and Harmelen (2008) describe the full set of requirements for an ontology design language – "efficient reasoning support and convenience of expression for a language as powerful as a combination of RDF Schema with a full logic" – as "unobtainable". As a result, OWL has three sublanguages that each aim to fulfill different parts of these

requirements – OWL Full, OWL DL, and OWL Lite [22, 24].

OWL Full

This refers to the entirety of OWL, meaning that it provides maximum expressiveness by supporting all OWL constructs and is fully compatible with RDF. However, the downside of OWL Full having maximum expressiveness is that it is therefore too powerful to guarantee complete or efficient reasoning [22, 24].

OWL DL

OWL Description Logic is sublanguage of OWL Full that includes all OWL constructs but has restrictions on their use in order to guarantee that all conclusions are computable and that all computations will finish in finite time. Every valid OWL DL ontology is also a valid OWL Full ontology and every valid OWL DL conclusion is also a valid OWL Full conclusion [22, 24].

OWL Lite

This sublanguage imposes further restrictions on the use of OWL constructs, such as excluding enumerated classes, disjointness statements, and arbitrary cardinalities. It is meant for users that mainly need a classification hierarchy and simple constraints. Consequently, ontologies built in OWL Lite are both easier to understand and to implement, but the obvious disadvantage is that there is limited expressivity. Note that every valid OWL Lite ontology is also a valid OWL DL ontology (and therefore a valid OWL Full ontology) and that every valid OWL Lite conclusion is also a valid OWL DL conclusion (and therefore a valid OWL Full conclusion) [22, 24].

Comparison

The choice of which language to use between OWL Lite and OWL DL depends on the level of expressivity required for a given use case and the choice between OWL DL and OWL Full

depends on whether the meta-modeling facilities of RDFS are required, such as attaching properties to classes. However, due to the wide range of features provided by OWL Full, it is unlikely that any reasoner will fully support reasoning for ontologies built using OWL Full [22, 24].

2.3.4 The Knowledge Interchange Format (KIF)

KIF was originally created by Michael Genesereth and other researchers involved in the DARPA Knowledge Sharing Project [25]. It is a formal language to describe knowledge so as to facilitate the exchange of knowledge between different computer programs. Such programs can have different programmers, be programmed at different times, and in different languages [26]. KIF features full semantic expressiveness but also has a high computational complexity, often making OWL the better choice of design language. It also has declarative semantics meaning that it is possible to understand the meanings of expressions written in KIF syntax without the need for an interpreter [25]. The foundations of the KIF syntax[5] is the 'character', of which there are six types:

```
upper ::= A | B | C | D | E | F | G | H | I | J | K | L | M |
          N | O | P | Q | R | S | T | U | V | W | X | Y | Z

lower ::= a | b | c | d | e | f | g | h | i | j | k | l | m |
          n | o | p | q | r | s | t | u | v | w | x | y | z

digit ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

alpha ::= ! | $ | % | & | * | + | - | . | / | < | = | > | ? |
          @ | _ | ~ |

special ::= " | # | ' | ( | ) | , | \ | ^ | `

white ::= space | tab | return | linefeed | page
```

Figure 2.4: The six types of character in KIF syntax [27].

A 'normal' character is either an upper case, lower case, digit, or alpha character. Lexical analysis is used to convert a flow of characters from different groups into lexemes. There

[5] http://logic.stanford.edu/kif/syntax.html

are five types of lexemes [27]:

1. **Special Lexemes**: these are composed of special characters only. Each special character forms its own lexeme and cannot be combined with other characters unless the backslash (escape) character is used.

2. **Words**: these are case-insensitive and consists of normal characters plus other characters prefixed with the escape character.

3. **Character References**: these are composed of the backslash or hash characters followed by any other character. These allow us to differentiate characters from one-character symbols.

4. **Character Strings**: these are a sequence of characters enclosed in quotation marks. The escape character is used to allow the inclusion of quotation marks and the backslash character within these strings.

5. **Character Blocks**: these group a sequence of arbitrary characters together without the use of escape characters. A character block consists of the hash character followed by the decimal encoding of an integer n (where $n > 0$), the character q or Q, and then n characters.

Expressions in KIF are formed from the lexemes described above. The following example in Figure 2.5 is a part of the SUMO (mentioned in Section 2.2), expressed in KIF syntax. In Line 1, we define the `Beverage` class as a subclass of the `Food` class. Then from Lines 2 to 4, there is a natural language description of the class `Beverage` class using the `documentation` property. Finally, Lines 5 to 7 state that if there is an instance of `Beverage` (represented by the variable `?BEV`), then that instance must be characterised as a `Liquid` using the `attribute` property [25].

```
   1. (subclass Beverage Food)
2. (documentation Beverage EnglishLanguage Äny &%Food that is ingested
3. by &%Drinking. Note that this class is disjoint with the other
4. subclasses of &%Food, i.e. &%Meat and &%FruitOrVegetable.")

5. (=>
6. (instance ?BEV Beverage)
7. (attribute ?BEV Liquid))
```

Figure 2.5: Example of KIF syntax from SUMO [25].

2.4 Ontology Components

Ontologies consist of the following five types of component, each of which can be considered to be a resource within the definition of an RDF data model [28]:

- **Classes**: represents real-world concepts,
- **Properties**: represents the relationships between resources,
- **Restrictions**: represents the limitations placed on properties,
- **Instances**: represents an individual occurrence of a class,
- **Annotations**: metadata attached to classes, properties, instances, and the overall ontology.

2.4.1 Classes

A class can be thought of as synonymous to a set of elements where the individual elements within that set are instances of that class. Classes can be organised into a hierarchical structure where the bottom-level classes are a specialisation of the classes directly above it. In general, A is a subclass of B if every instance of A is also an instance of B. This makes B a superclass of A [22]. For example, given the following RDF triples, an instance of type `VideoEditor` is an instance of `Staff` and therefore an instance of `Role`:

Staff -- rdfs:subClassOf -- Role

VideoEditor -- rdfs:subClassOf -- Staff

Two classes can also be defined as equivalent using `owl:equivalentClass` or as disjoint using `owl:disjointWith`. When two classes are equivalent to one another, they have precisely the same instances, although can have a different meaning (that is, they can denote a different concept). For example, a class called `BritishPrimeMinister` and a class called `MainResidentOf10DowningStreet` denote different but related concepts and so they have the same instances.

When two classes are disjoint, it means an instance of one of those classes cannot also be an instance of the other [29].

Enumerations

Besides class names, classes can be expressed in OWL as an exhaustive enumeration of individuals and as Boolean combinations. An enumeration is a special kind of class that consists of a fixed set of instances defined using `owl:oneOf`. For example, an enumerated class of weekdays would have seven instances: `Monday`, `Tuesday`, `Wednesday`, `Thursday`, `Friday`, `Saturday`, and `Sunday` [22].

Boolean Combinations

Boolean combinations refer to the intersection, union, and complement of two or more classes which can be thought of as the logical AND, OR and NOT operators respectively. An intersection of classes can be defined using the `owl:intersectionOf` property and the resulting class contains only those instances that are members of all classes being intersected. A union of classes can be defined using the `owl:unionOf` property and the resulting class contains those instances that are members of at least one of the classes in the union. The complement of a class can be defined using the `owl:complementOf` property and the resulting class contains only those instances that are not in the complemented class

[22]. Figure 2.6 shows these Boolean combinations in the form of Venn diagrams. OWL also allows classes to be defined as restrictions on properties, which is discussed in Section 2.4.3.

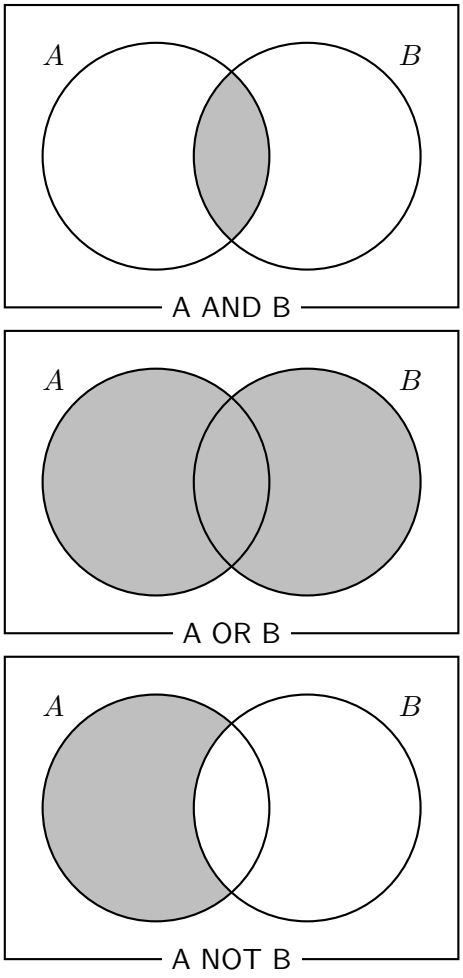


Figure 2.6: The three Boolean combinations.

2.4.2 Properties

In OWL, there are three types of property: object properties, data type properties, and annotations. Object properties create a relationship between two classes whereas data type properties create a relationship between a class and a data value. Annotations allow the components that make up the ontology to be 'tagged' with metadata, which is explained

further in Section 2.4.5.

Domain and Range

Both object and data type properties can be given a domain (using the predicate `rdfs:domain`) and a range (using the predicate `rdfs:range`), which are used by a reasoner to make inferences [22]. For example, given the following RDF triples, the reasoner can infer that `ToyStory3` is a `Project` and `Disney` is a `Client`. We will use these triples as the basis for all future examples in this chapter:

```
hasClient -- rdfs:domain -- Project
hasClient -- rdfs:range -- Client
ToyStory3 -- hasClient -- Disney
```

However, property domains and ranges are axioms, not constraints to be checked [22]. For example, given the following additional RDF triple, the reasoner can infer that `CanonXA11` is a `Client`, despite this not making sense in reference to the real world because `CanonXA11` is a camera:

```
ToyStory3 -- hasClient -- CanonXA11
```

Consider a further RDF triple below. In this case, the reasoner infers that `CanonXA11` is a `Project` and `ToyStory3` is a `Client`. Again, this does not make sense in reference to the real world.

```
CanonXA11 -- hasClient -- ToyStory3
```

However, when we combine these triples together as follows, such that `ToyStory3` and `CanonXA11` are each used as the subject and object in the triple, the reasoner infers that `ToyStory3` is both a `Project` and a `Client`. Likewise, `CanonXA11` is inferred to be both a `Project` and a `Client`.

```
hasClient -- rdfs:domain -- Project
 hasClient -- rdfs:range -- Client
ToyStory3 -- hasClient -- CanonXA11
CanonXA11 -- hasClient -- ToyStory3
```

Object properties take classes as both their domain and range, but data type properties take classes as their domain and a built-in data type, such as integers and strings, as their range.

Property Characteristics

Property characteristics allow the meaning of properties to be enriched when they are asserted. The following seven characteristics can be applied to object properties but only one (functional) can be applied to data type properties [30]:

- **Functional**: this means that for a given individual X (the subject in an RDF triple), there can be at most one individual Y (the object in an RDF triple) that is related to X via the functional property P. If multiple individuals $(Y_1, \ldots, Y_n)$ are stated to be related to the subject X, then we infer than the multiple individuals refer to the same thing. An example of a functional property, P, could be `hasBiologicalFather` which states that a person X can have at most one biological father Y.

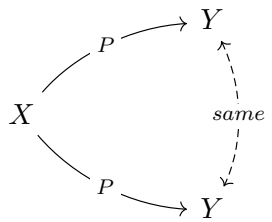


Figure 2.7: A functional property

- **Inverse Functional**: this means that for a given individual Y (the object), there can be at most one individual X (the subject) that is related to Y via the inverse

functional property P. If multiple individuals $(X_1, \ldots, X_n)$ are stated to be related to the object Y, then we infer than the multiple individuals refer to the same thing. An example of an inverse functional property, P, could be `isBiologicalFatherOf` which states that a person Y can have at most one biological father X. In other words, it is the inverse of the `hasBiologicalFather` given as an example of a functional property.

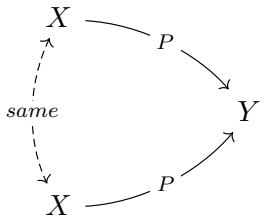


Figure 2.8: An inverse functional property

- **Transitive**: this means that an individual A is related to another individual C via the transitive property P if A is related to individual B and B is related to C via P. An example of a transitive property, P, could be `hasDescendant` which states that a person A has a descendant C if C is a descendant of B and B is a descendant of A.

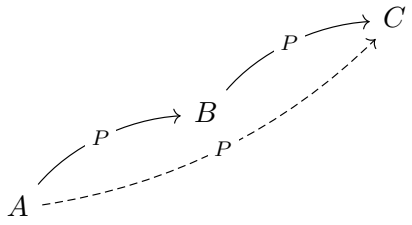


Figure 2.9: A transitive property

- **Symmetric**: this means that if an individual A is related to another individual B via a property P, then B is related to A via that same property. In other words, a symmetric property is its own inverse. An example of a symmetric property, P, could

be `hasSibling` which states that a person B has a sibling A if A is connected to B via the symmetric property `hasSibling`.

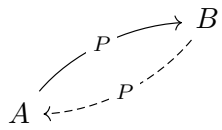


Figure 2.10: A symmetric property

- **Antisymmetric**: this means that if an individual A is related to another individual B via a property P, then B cannot be related to A via that same property. An example of an antisymmetric property, P, could be `hasChild` which states that if a person A has a child B, as asserted by the `hasChild` antisymmetric property, then B cannot also have a child A via the same antisymmetric property.

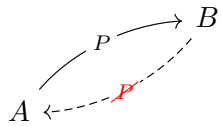


Figure 2.11: An antisymmetric property

- **Reflexive**: this means that any individual involved in a property P is also related to themselves via P, although this does not necessarily mean that two individuals related by P are equal. An example of a reflexive property, P, could be `knows` which states that if a person A knows person B, A can also know themselves via the same reflexive property `knows`. However, A and B are not guaranteed to be equal.

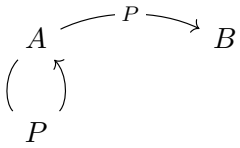


Figure 2.12: A reflexive property

- **Irreflexive**: this means that any individual involved in a property P cannot also be related to themselves via P. An example of an irreflexive property, P, could be `isBiologicalFatherOf` which states that if a person A is the biological father of B, A cannot also be connected to itself via the `isBiologicalFatherOf` irreflexive property.

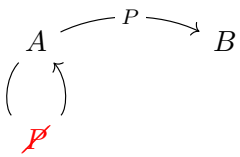


Figure 2.13: An irreflexive property

2.4.3 Restrictions

If a property has a restriction on it, that restriction must be true for an inputted assertion to be accepted. Restrictions can be applied to properties to describe the features of the values the property can take, including the value type, the allowed values, and the number of values [28]. A value constraint describes the type of values that a property can take in the context of a specific class. The more common types including literals (such as strings, numbers, or Booleans), enumerations (a choice from a list of pre-defined values), and instances of classes. There are three types of value constraint in OWL [30]:

- `owl:allValuesFrom`: this is used to define the class of possible values that the property this restriction is applied to can take. All values of the property must come from this class, which means it is a universal quantification.

- `owl:someValuesFrom`: this is used to define the class of some of the possible values that the property this restriction is applied to can take. At least one value of the property must come from the class, which means it is an existential quantification, but the property can also have values from other classes.

- `owl:hasValue`: this is used to define a specific value that a property must take.

Cardinality constraints define how many values a property can take in the context of a specific class. Some systems can only distinguish between single cardinality and multiple cardinality, but others can define a minimum and a maximum cardinality [28].

2.4.4 Instances

Instances are individual objects (resources) that belong to a class and this relationship is expressed in RDF using the `rdf:type` property [30]. For example, given the following RDF triples, `ToyStory3` is an instance of `Project` and `Disney` is an instance of `Client`.

```
hasClient -- rdfs:domain -- Project
hasClient -- rdfs:range -- Client
ToyStory3 -- hasClient -- Disney
```

Therefore:

```
ToyStory3 -- rdf:type -- Project
Disney -- rdf:type -- Client
```

There are situations where there is ambiguity over whether a real-world entity should be represented in an ontology as class or as an instance. Samwald (2006) describes the Biological Pathways Exchange (BioPAX) ontology, where the `Protein` class is one of the most specialised classes and is supposed to be directly instantiated. For example, an instance of Protein called `SerotoninReceptor` could be created. In one context, that would be enough but in another, the user may wish to represent different types of `SerotoninReceptor` – such as `SerotoninReceptor1A` and `SerotoninReceptor2A`. To achieve this in OWL, `SerotoninReceptor` would then have to be a class rather than an instance. `SerotoninReceptor1A` and `SerotoninReceptor2A` could then either be sub-classes or instances of `SerotoninReceptor`, depending on the needs of the user [31].

2.4.5 Annotations

OWL allows classes, properties, instances, and the overall ontology to be annotated with pieces of metadata information. There are five pre-defined annotation properties that can be attached to classes, properties, and instances [30]:

- `owl:versionInfo`: this property can be used to add information about changes between versions of ontology elements. Besides ontology elements, an overall ontology can be annotated with this property.
- `rdfs:label`: this property can be used to add human readable or multi-lingual names to ontology elements.
- `rdfs:comment`: this property can be used to add additional information about the ontology elements that do not fit anywhere else, such as a description.
- `rdfs:seeAlso`: this property has a range of a Uniform Resource Identifier (URI) and is used to identify related ontology elements. A URI is a string of characters that is used to uniquely identify any web resource. Web addresses, also called URLs, are a type of URI.
- `rdfs:isDefinedBy`: this property also has a range of a URI and is used to identify an ontology element that defines the subject element.

As well as `owl:versionInfo`, there are three pre-defined annotation properties that can be applied to an ontology as a whole, all of which have a range of a URI [30]:

- `owl:priorVersion`: this property identifies previous versions of the ontology.
- `owl:backwardsCompatibleWith`: this property identifies a previous version of the ontology that the current ontology is compatible with. Two ontologies are compatible when all the identifiers in one have the same intended meaning as those in the other.

- `owl:incompatibleWith`: this property identifies a previous version of the ontology that the current ontology is not compatible with.

Ontology developers also have the option to create custom annotation properties for ontologies and ontology elements.

2.5 Ontology Query Languages

To query RDF-based ontologies, we need to use an RDF query language. There are several families of RDF query languages that are distinguished by characteristics such as data model, expressivity, query type, and support for schema information [32]. This section will compare the two most widely used families of RDF querying languages, SPARQL and RQL.

2.5.1 The SPARQL Family

The SPARQL family consists of four query languages – SquishQL, RDQL, SPARQL, and TriQL [32]. The first, SquishQL, was designed with the purpose of being easy to use and similar to SQL. Queries written in SquishQL work based on triple pattern matching and conjunctions between triple patterns, where a pattern is an RDF triple with a variable in place of at least one of the resources. For example, the following are all valid triple patterns:

```
?resource -- hasClient -- Disney
ToyStory3 -- ?resource -- Disney
ToyStory3 -- hasClient -- ?resource
```

The first pattern will return all resources whose `Client` is `Disney`, the second will return a list of relations connecting `ToyStory3` to `Disney`, and the third will return the resource(s) representing the `Client` in the `ToyStory3` project.

A query written in SquishQL only supports selection and extraction and does not support RDFS concepts. Such queries are formed of five parts:

1. **The** `SELECT` **Clause**: specifies the variables that have been used in the triple patterns that are to be returned. Not all variables used in the triple pattern matching process need to be returned so specifying the required variables can reduce the amount of memory needed to process the query.

2. **The** `FROM` **Clause**: specifies the semantic model to query by URI.

3. **The** `WHERE` **Clause**: specifies a list of triple patterns that are to be matched.

4. **The** `AND` **Clause**: (optional) specifies any filters over the values of URIs and literals, including arithmetic comparisons and Boolean expressions.

5. **The** `USING-FOR` **Clause**: (optional) specifies a way to shorten the length of URIs by giving long URIs an abbreviated prefix.

The second query language in the SPARQL family is RDQL, which is an evolution of SquishQL. RDQL queries have the same form as SquishQL, as defined above, and the results for both SquishQL and RDQL are a set of bindings for the variables used in the triple patterns in the query. RDQL also supports only selection and extraction queries and does not support RDFS concepts.

The third query language in the SPARQL family is SPARQL itself, which is an extension of RDQL. It extends RDQL by providing the ability to:

- Extract RDF subgraphs.
- Specify optional triples.
- Test the absence of tuples.

SPARQL also introduces the ability to run `DESCRIBE`, `ASK`, and `CONSTRUCT` queries. In total, there are four types of SPARQL query:

- `SELECT`: This extracts a list of RDF triples that match the patterns in the `WHERE` clause and from that list, find and returns the values represented by the variables used in the patterns and requested in the `SELECT`.

- `DESCRIBE`: This extracts a list of RDF triples that involve the resource being described and then assembles an RDF graph using those triples.

- `ASK`: This returns a Boolean value indicating whether there exists an RDF triple that matches the query pattern.

- `CONSTRUCT`: Unlike the other three types of SPARQL query, `CONSTRUCT` queries can produce triples not found in the ontology that is being queried. It works by defining a graph template using a set of variables and resources in the `CONSTRUCT` clause, with the `WHERE` clause specifying the triple patterns that must be matched to obtain the variable values that will be applied to the graph template.

The final language is TriQL, which extends RDQL by introducing constructs that support the querying of named graphs. These allow a user to filter RDF statements based on their original source. Figure 2.14 summarises the relationship between the four query languages in the SPARQL family.

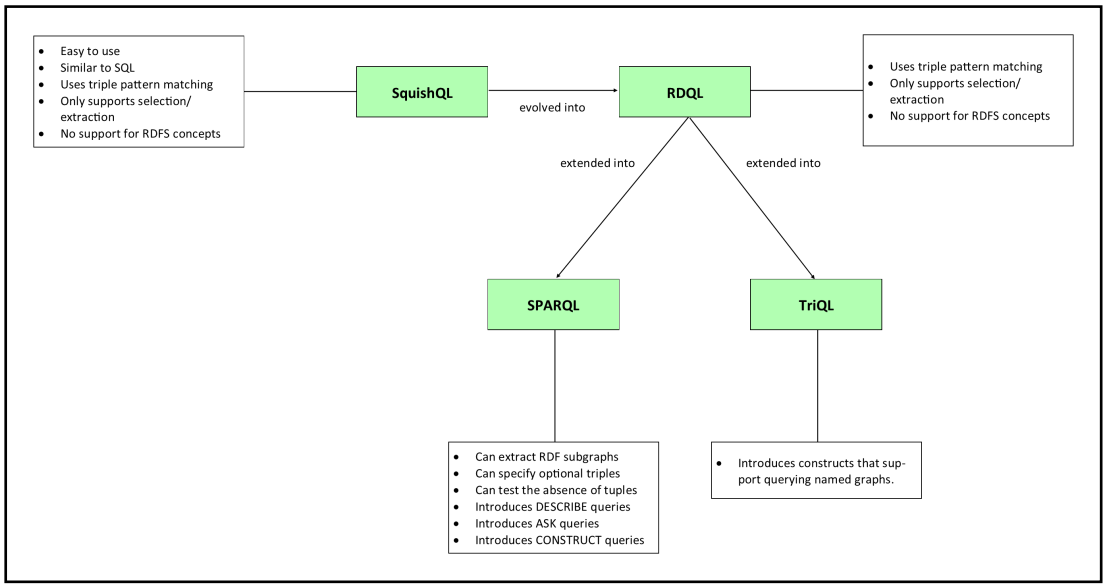


Figure 2.14: The SPARQL family of query languages.

2.5.2 The RQL Family

The RQL family consists of three query languages – RQL, SeRQL, and eRQL. RQL is the original language but was criticised for its extensive choice of syntactic constructs, resulting in the creation of the simplified versions SeRQL and eRQL [32]. Query languages in this family enable the querying of RDF triple stores with little knowledge of the underlying schema. They rely on typing and have a functional paradigm [33].

There are three types of RQL queries, specifically schema queries, data queries, and hybrid queries. Schema queries return information about the underlying structure of the data, data queries retrieve instance data, and hybrid queries combine the former two [32]. To facilitate such queries, RQL provides functions to traverse class and property hierarchies. The query `subClassOf(x)` will return the subclasses of the class x and, similarly, `subPropertyOf(y)` will return the sub-properties of the property y. The query `topclass(x)` returns the top of the class hierarchy in which x is placed and, similarly, `topproperty(y)` will return top of the property hierarchy in which y is placed [32].

Although both SeRQL and eRQL are simplifications of RQL, they have each been simplified in different ways, hence the need for both languages. For example, unlike RQL, SeRQL supports a shorthand notation for expressing several properties of a resource in a `FROM` clause but it does not support set operations or existential or universal quantification. Meanwhile, eRQL was proposed as a simplification of RQL based on a keyword interface. The developers of eRQL stated that their goal was to provide a "Google-like query language but also with the capacity to profit off the additional information given by the RDF data" [32]. eRQL has three query constructs:

- **One-Word Queries**: These refer to queries consisting of a single word. For example, the query `Disney` would return all statements in which the string "Disney" appears. However, it has been noted that so-called phrase queries (i.e. queries of more than one word) cannot be expressed in eRQL [32].

- **Neighbourhood Queries**: These are expressed by varying the number of curly braces surrounding the string being queried. The number of pairs of braces indicate the level of neighbourhood, which means not only are statements containing the string returned, but also statements related to it. For example, the query {{`Disney`}} returns all statements connected by at most two edges in the RDF graph to a node containing "Disney" [34].

- **Conjunctive and Disjunctive Queries**: Both, one-word and neighbourhood queries can be combined using the Boolean operators `AND` and `OR`. However, it is important to note that negation is not supported [32, 34].

Furche and Orsini (2004) describe the eRQL language as "more akin to an information retrieval language than a conventional query language". Despite this, eRQL is one of only a few approaches to combining information retrieval features and RDF querying [34].

Figure 2.15 summarises the relationship between the three query languages in the RQL

family.

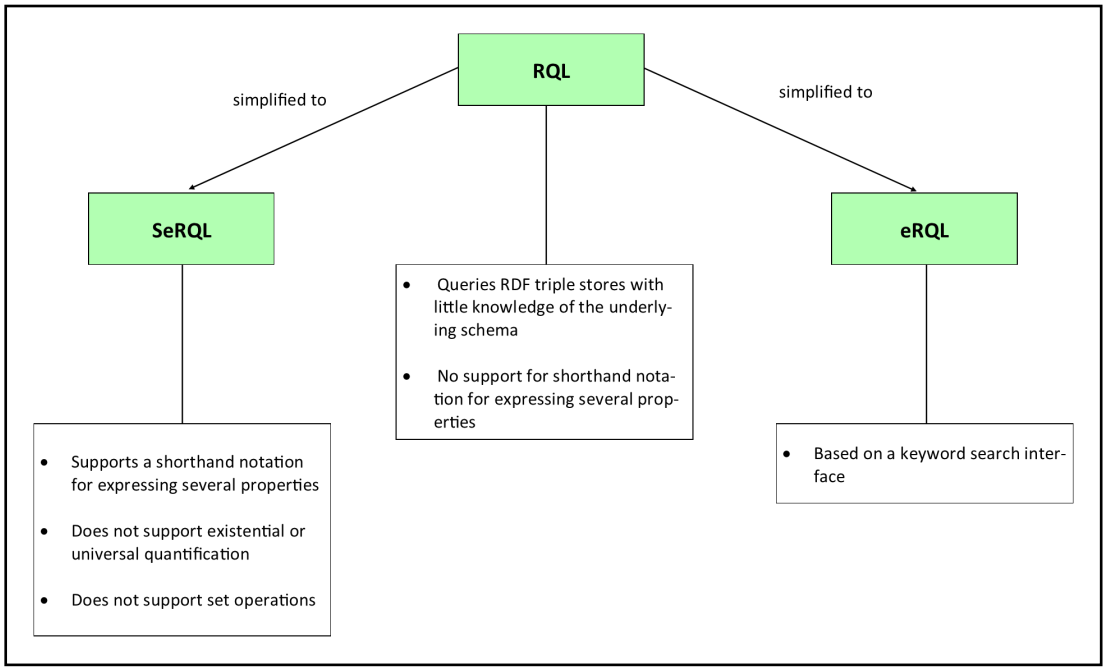


Figure 2.15: The RQL family of query languages.

2.6 Ontologies for Knowledge Management

Ontologies have a range of applications in a variety of fields, although real-world applications typically amount to some form of knowledge management. Knowledge management can be defined as ensuring the correct knowledge is available to the correct people at the correct time to help those people use information in a way that improves the performance of an organisation [35]. Knowledge management systems are therefore "enabling technologies for an effective and efficient knowledge management" [36], and these can take several forms. Ribino et al. (2009) identifies three common forms [37]:

1. **Document based**: uses technologies that facilitate the creation, management, and sharing of documents. Examples include the web and distributed databases.

2. **Ontology based**: classifies resources into a set of classes, relationships, attributes,

and instances. Examples include the knowledge management systems discussed in Section 3.2.

3. **Artificial Intelligence based**: uses artificial intelligence to represent and reason about knowledge.

Freitas et al. (2010) argues that ontology-based knowledge management systems can create a structure where information can easily be stored in its correct "granularity, position, and context" to transform it into a more useful form. The KnowMore architecture is cited as an example of an ontology-based knowledge management system with three layers: the object level encapsulating information sources such as databases and documents, the description level compiled of mostly ontologies to describe and link knowledge about the domain and information sources available, and systems to process the object-level data, and the application level where tasks based heavily on knowledge are performed using the ontologies from the description level [38]. The KnowMore architecture is shown in Figure 2.16.

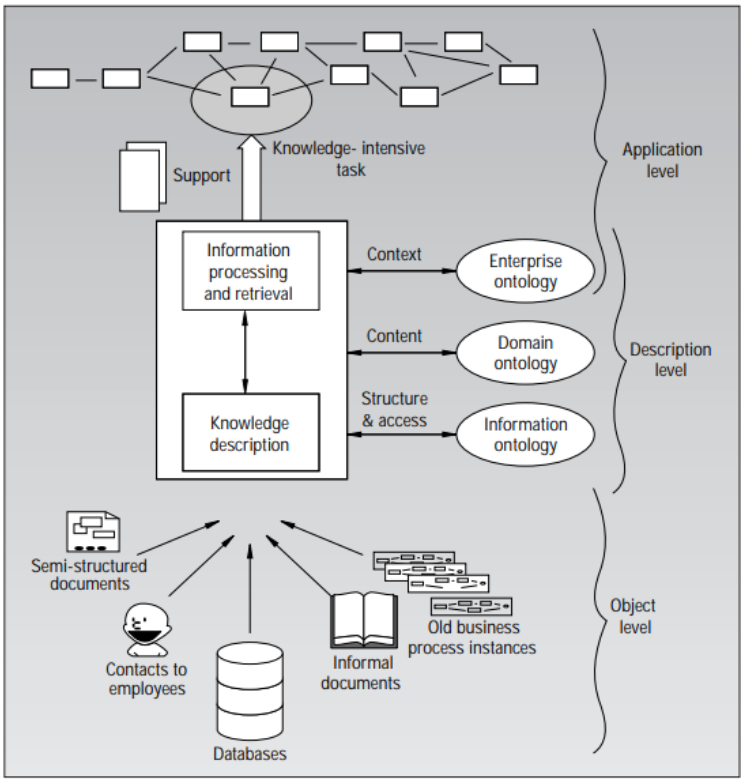


Figure 2.16: The KnowMore architecture [39].

Sureephong et al. (2007) find that a knowledge management system often uses two types of knowledge – domain and problem-solving – where the former describes the facts about the domain and the latter describes how the domain knowledge can be used to achieve various goals [40]. Therefore, ontology-based knowledge management systems typically require two main ontologies: the domain ontology that describes the studied domain, and the task ontology which describes the knowledge required to fulfil a specific use case, as described in Section 2.1.

2.6.1 Limitations of Ontologies

Although ontologies are extremely effective at encoding knowledge into a machine-interpretable form, they do have some limitations that prevents them from being a universal solution to knowledge representation problems. These limitations are that they cannot represent all

types of knowledge and that reasoner tools may not be scalable to larger and more complex ontologies.

Representing all Types of Knowledge

Ontologies are based on logic which means that they can effectively represent statements that are either true or false, such as the statement "The Toy Story 3 project has client Disney", which would return true. However, it is not possible to represent all types of knowledge about a domain with true/false statements, such as statistical or conditional knowledge. For example, OWL does not support knowledge at class level about classes that participate in relationships at one time and not at another [41]. Additionally, ontologies also do not handle well any classes that derive their meaning from a comparison to a dynamic group (e.g. "The project with the longest duration") [42].

Reasoner Scalability

As already mentioned, OWL supports using knowledge that explicitly exists (assertions) from the ontology to infer additional knowledge that is not directly stated (inferences). This is done using a software tool known as a reasoner. Despite the advancements in technology for automated reasoning tools, there are still technical limits that restrict a tool's scalability to larger and more complex ontologies [42]. For example, OWL Full provides maximum expressiveness meaning it is more computationally complex to the extent that there is no guarantee the reasoning process would end when run on a large ontology written in this OWL profile. However, other OWL profiles – OWL EL[6] and OWL QL[7] – have a worst case computational complexity of polynomial time [43]. Therefore, more complex logical constructs, such as non-binary relationships, may not be able to be represented and reasoned with, depending on the version of OWL used and the size of the ontology [42].

[6] https://www.w3.org/TR/owl2-profiles/#OWL_2_EL
[7] https://www.w3.org/TR/owl2-profiles/#OWL_2_QL

2.6.2 A Comparison with Databases

Ontologies and databases have closely related functions, in that they support the development of domain models for the purpose of knowledge management that can later be queried and updated [44]. Databases can be considered an alternative to ontologies for knowledge management, but there are some crucial differences that determine which one is suitable for a particular application.

Open World Assumption

While databases work based on a closed-world assumption, OWL ontologies operate using an open world assumption in which missing information is treated as unknown rather than false. This also means that ontologies treat OWL axioms as inferences whereas databases treat the schema as constraints. For example, given the following axioms in the ontology, it is implied that `ToyStory3` is a `Project` and `Disney` is a `Client`:

```
hasClient -- rdfs:domain -- Project
hasClient -- rdfs:range -- Client
ToyStory3 -- hasClient -- Disney
```

This means that the following axioms do not need to be explicitly declared:

```
ToyStory3 -- rdf:type -- Project
Disney -- rdf:type -- Client
```

However, in a database the schema would be interpreted as a set of constraints. Therefore, if `ToyStory3` is not already explicitly identified as a `Project` and `Disney` identified as a `Client`, adding the statement that `ToyStory3` has a client called `Disney` would create an invalid database state so the update will be rejected [44].

No Unique Name Assumption

OWL ontologies also make no unique name assumption. This means that if there is a restriction in the ontology stating that a `Project` can have only one `Client`, the following axioms allow the reasoner to infer that `Disney` and `Pixar` refer to the same `Client`:

```
ToyStory3 -- hasClient -- Disney
ToyStory3 -- hasClient -- Pixar
```

However, in a database environment, this would again be treated as a constraint violation and will be subsequently rejected. In an ontology, it is possible to assert that two different names do not refer to the same thing (i.e. add a unique name assumption for those particular objects). In this case, if `Disney` and `Pixar` are treated as two different entities, the two axioms above would create an inconsistent ontology but unlike in databases, the update would not be rejected [44].

Query Processing

When a database is queried, the schema and its constraints can be ignored because all schema constraints should already be satisfied in a valid database instance. This makes query processing far more efficient than within an ontology. For example, to determine whether `ToyStory3` is in the answer to a query for projects, it is enough to just check if this fact is explicitly present in the database. In contrast, the schema of an ontology is actively considered when processing a query, which allows us to answer conceptual queries (i.e. queries about the schema) in addition to extensional queries (i.e. queries about the data). However, in order to perform the same query with an ontology it may, in the worst case scenario, be necessary to check if `ToyStory3` would be an instance of `Project` in every possible state of the world that is consistent with the axioms in the ontology [44].

Additionally, ontologies differ in focus from databases. The primary function of ontologies is to preserve meaning to facilitate interoperability whereas a database schema is designed

to store and query large data sets [45]. Given these differences in function, an ontology queried using SPARQL would return information on relationships, graph structure and triples whereas a database queried using SQL will return a flat table of information [46].

Purpose and Reusability

When a database is designed and implemented, it is done so from scratch with a specific purpose in mind, meaning it is not reusable or shareable [45, 47]. This is because databases are software system implementation-oriented. On the other hand, although ontologies may also be created from scratch for a specific purpose, it is still possible to reuse an ontology for other, unforeseen applications [45, 47]. This is because ontologies are dependent on semantics rather than implementation (i.e. they are software system implementation-agnostic) and are instead domain-oriented. Figure 2.17 illustrates the relationship between the two.

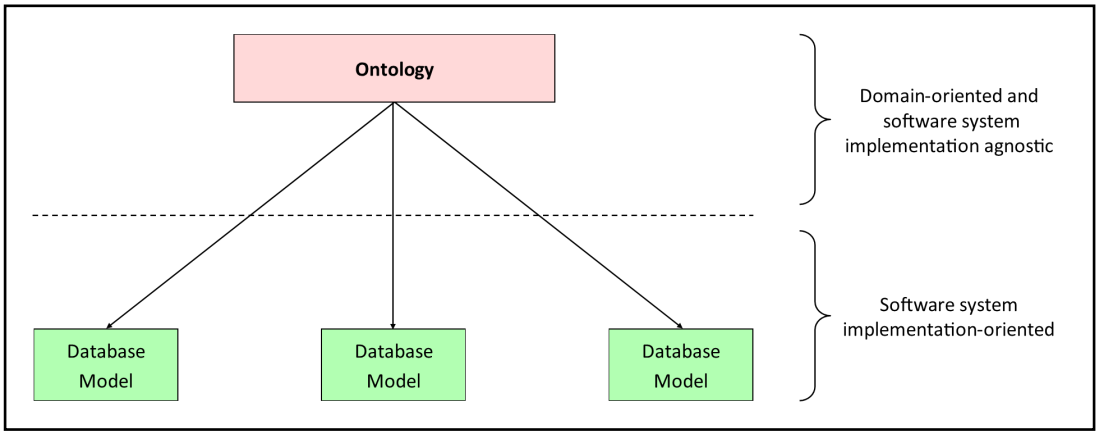


Figure 2.17: Orientation of ontologies versus databases.

Motivation for Using Ontologies

We have already established that both databases and ontologies can be considered suitable solutions to knowledge management problems. We have also already stated in Chapter 1 that databases are far more commonplace in the moving image industry than ontologies are, despite ontologies being a possible data and metadata management solution. The

choice to use an ontology in this study is therefore justified by the lack of ontologies for metadata management for use in the moving image industry. More specifically, it would be interesting to explore the possibility of creating a hierarchy of media terminology, as databases typically place little emphasis on hierarchical structures. We have also chosen to use ontologies over databases because, once an ontology is created, it would create opportunities for future research to explore the impact of an open world versus a closed world on the workflows of media companies that specialise in moving image.

2.7 Summary

This chapter started by outlining the different types of ontologies, followed by a description of RDF, RDFS, and OWL, which are the three main ontology design languages. It was followed by an explanation of the five components that form an ontology: classes, properties, restrictions, instances, and annotations. The chapter then moved on to describe the two main families of ontology querying languages – SPARQL and RQL – before ending with an explanation of possible applications of ontologies and their limitations, along with a comparison of ontologies and databases. This chapter has acted as a primer for ontologies which has in turn highlighted the features of ontologies compared to databases. Although databases are the de facto approach to handling data and metadata across a range of industries, the comparison has brought attention to the possibilities of using ontologies for modelling a domain for the purposes of data and metadata management. Thus, it was decided to utilise ontologies in this project to model the pre- to post-production workflows.

Chapter 3

The State of Play

Although this thesis was formed due to a gap in the literature with respect to existing ontology-based solutions to metadata management in the media creation domain, there have been many examples of research into ontologies within the media creation and archival domains and in knowledge management systems for other domains. In order to fully contextualise the impact of the research contained within this thesis, it is important to conduct a survey into the state of the art. This chapter begins with a review of existing ontologies that conceptualise the media creation and archival domains in Section 3.1 before examining ontology-based knowledge management systems in Section 3.2. The scope of this chapter is limited to ontologies designed for the media creation and archival domains only but extends to ontology-based knowledge management systems for any domain because while there is a wealth of ontologies conceptualising the media creation and archival domains, there are few ontology-based knowledge management systems that cover this so there is a need to explore such systems for other domains.

3.1 Ontologies in Media Production

Note: *this section uses material previously published in and adapted from [48].*

There have been several avenues of research into the use of ontologies for organising data and metadata in the media industry. These include Loculus [49, 50], the Creative Works Ontology [51], COMM [52], OntoFilm [53], and The Deep Film Access Project [7]. Each of these can be placed into one of two categories: 1) ontologies for managing metadata associated with a final media product or 2) ontologies for managing metadata during media creation. This section will describe the purpose of each ontology and the scope of the research surrounding it.

3.1.1 Loculus

Loculus is a metadata wrapper for managing, distributing, and reusing digital motion pictures by enabling information about an artefact from different stages of the production process to be encapsulated together. It has a modular structure, shown in Figure 3.1, consisting of a core module (labelled as 'Core Module'), with the opportunity for functionality expansion through plug-ins for digital artefact repository management, rights management, and other facilities. It is important to note that the Core Module is useless on its own. Its purpose is to act as a control node that issues instructions to the plugins to perform a task. As a result, the user will interact only with the core module, not with the individual plugins [49].

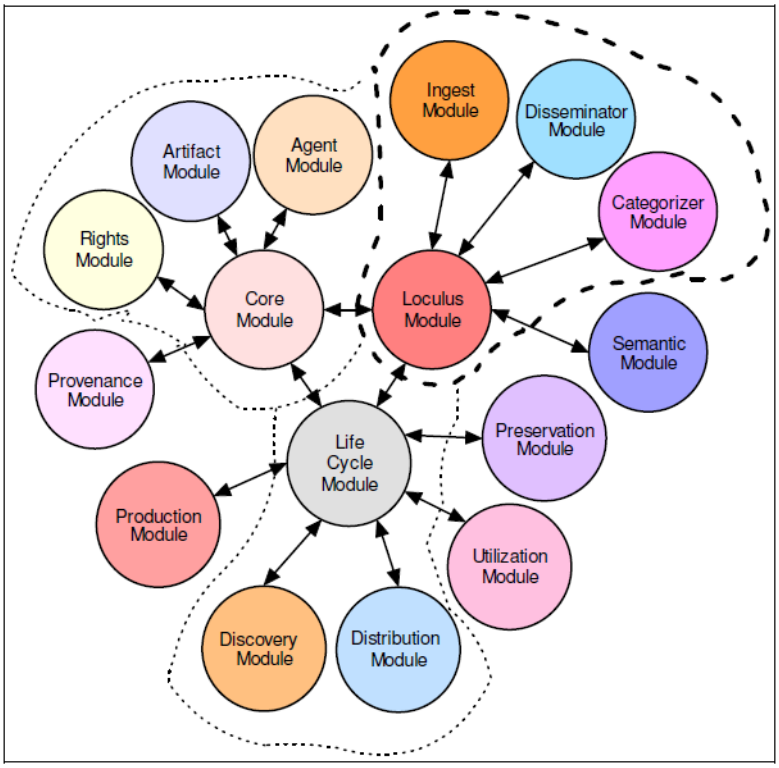


Figure 3.1: The modular structure of Loculus [49].

The Loculus Ontology consists of three sub-ontologies: the Motion Picture Industry (MPI) concepts ontology, the agent concepts ontology, and the common concepts ontology. The MPI concepts ontology contains the concepts that are unique to the motion picture industry. For example, *cross-cutting* is an editing technique unique to the industry while *treatment*, although a term used in MPI to refer to a document prepared as part of a pitch for a new project, is not unique to the industry. Therefore, cross-cutting would be modelled in the MPI concepts ontology but *treatment* would not. The agent concepts ontology models those that are involved in the processes of the motion picture industry. For example, if the work is of historical significance, an archivist would be involved and would be modelled in the agent concepts ontology. Finally, the common concepts ontology models concepts that frequently occur in the motion picture industry but are not unique to the industry, such as *fee* [50].

The Loculus Ontology itself is not shown in Figure 3.1. Instead, the Semantic Module is shown, which contains an Ontology Reader class that can parse the Loculus Ontology and use it with semantic relatedness metric calculations to enable information exploitation [50]. Figure 3.2 shows the components that form the Semantic Module.

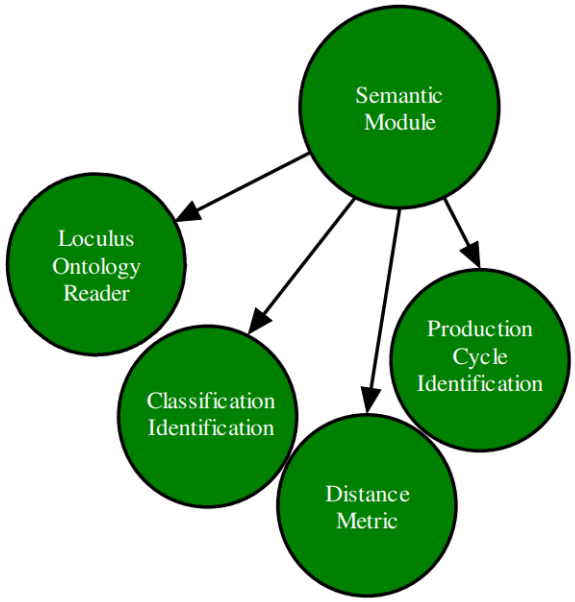


Figure 3.2: The structure of the Semantic Module in Loculus [50].

We should note that Loculus' high-level system architecture can be applied to any ontology, as Loculus is not coupled to a specific one. Due to the modular structure of its architecture, Loculus could be applied to other ontologies by adapting the ontology reader and semantic relatedness metric calculations to be able to work with a different ontology. The rest of the system architecture would remain the same [50].

Although Loculus attaches metadata from various stages of media creation to a final digital media product, it is not designed to act as a metadata management system. Therefore, it is not suitable for media practitioners to track, view and modify metadata as they progress from pre- to post-production [49].

3.1.2 The Creative Works Ontology

The Creative Works Ontology is similar to Loculus in that it is used to capture relationships between creative works and other relevant entities, initially focusing on film and television works but soon transitioning to solely film. The ontology has five top-level classes, defined as those that can exist in isolation without being attached to other objects [51]:

- `Creative Work`: Although the ontology now focuses solely on film, this class encapsulates all types of creative media product and has therefore been given a generic name to provide flexibility to accommodate future extensions. Each `Creative Work` has metadata assigned to it such as awards nominated for or won, a rating for censorship and audience suitability, a ranking based on user reviews provided from a source like IMDB[1], and production information such as cost, companies involved, and contributors. `Creative Works` also have release information associated with it which includes: the start and end date of the releases, the countries in which the release took place, the distributor, the channel, the distribution model, the format, and aggregated consumption data.

- `Person`: This class represents a person separately from their role(s) in a creative work, although an instance of `Person` can be associated with a `Creative Work` via their role. Each `Person` can be connected to multiple `Creative Works` with different roles in each. Each `Person` has the following metadata attached to it: names (including billed name, contractual name, credited name, real name, translated name, and preferred name), country of citizenship and birth place, whether or not they are seen as a star, gender, and date of birth and death.

- `Location`: This class represents a real or fictional location and is used in places in the ontology, such as a `Creative Work`'s production country and a `Person`'s birth place. Each `Location` has the following metadata attached to it: name, country,

[1] `https://www.imdb.com`

country code, latitude-longitude coordinates, any landmarks, a Boolean property indicating whether or not the location is fictional, and any further information about the location.

- `Group`: This class represents a collection of `Creative Works`, some of which can be inferred using information already in the ontology (e.g. genre collections) and others which can be manually created by adding instances of `Creative Works` to an instance of `Group`. Each `Group` has the following metadata attached to it: the type of group, a Boolean property indicating whether it is an official group, the source (i.e. who created the grouping), a description, additional notes, and members (i.e. instances of `Creative Works`).

- `Award`: This class represents the awards that a `Creative Work` has either been nominated for or won. Each `Award` has the following metadata associated with it: year of the award, sequence number (e.g. "The 50th Annual Award for..."), further details about the award, and type of award.

While the Creative Works Ontology is capable of handling many forms of metadata, it has not been designed with the media creation process in mind. Rather, it has been designed for a finished media product, so it cannot store metadata generated during production or post-production.

3.1.3 The Core Ontology for Multimedia

The Core Ontology for MultiMedia (COMM) is an ontology built in OWL DL for annotating multimedia using a re-engineered version of MPEG-7 (see [54] for more information), which is the current de facto standard for this task. It was built to explain the composition of multimedia objects and to label what its parts represent [52]. Arndt et al. (2008) gives the following scenario as a means of presenting an example of how COMM can be used:

A history student would like to create a multimedia presentation on the Yalta Conference, starting with the "Big Three" picture (see Figure 3.4[2]) showing the heads of the United States, the United Kingdom, and the Soviet Union governments during World War II. The student uses facial recognition software to detect the faces of these government heads and would like to import the extraction results into the presentation authoring tool in order to automatically create links from the detected face regions to detailed textual information.

– adapted from [52]

Figure 3.3: The Big Three at the Yalta Conference

Although MPEG-7 could be used for representing the results of the facial recognition software, Arndt et al. (2008) states that it is not possible to guarantee that the MPEG-7

[2] https://en.wikipedia.org/wiki/Yalta_Conference

metadata generated by different software will be understood due to the lack of formal semantics. For example, one piece of software may use a different MPEG-7 tag to connect a facial region to a name compared to another piece of software, resulting in the authoring tool not correctly linking textual information to the relevant images. Consider the following example MPEG-7 annotation applied to Figure 3.4:

```
<Mpeg7>
 <Description xsi:type="ContentEntityType">
  <MultimediaContent xsi:type="ImageType">
   <Image id="IMG1">
    <SpatialDecomposition>

     <StillRegion id="SR1">
      <Semantic>
       <Label><Name> Roosevelt </Name></Label>
      </Semantic>
     </StillRegion>

     <StillRegion id="SR2">
      <TextAnnotation>     <!-- TextAnnotationType -->
       <KeywordAnnotation><Keyword> Churchill </Keyword></KeywordAnnotation>
      </TextAnnotation>
     </StillRegion>

     <StillRegion id="SR3">
      <Semantic>
       <Definition>   <!-- Also TextAnnotationType -->
        <StructuredAnnotation><Who><Name> Stalin </Name></Who></StructuredAnnotation>
       </Definition>
      </Semantic>
     </StillRegion>
     ...
```

Figure 3.4: The Big Three photo annotated using MPEG-7 [52].

In this example, the student used three different facial recognition services to identify the faces of the three leaders and label those regions in the image with IDs `SR1`, `SR2`, and `SR3`. While the XML code is syntactically correct, each region uses a different MPEG-7 tag to assign a name to that region. Service 1 uses the `Semantic` tag to assign a `Label` of value *Roosevelt* to `SR1` while Service 2 uses `KeywordAnnotation` to assign a `Keyword` of value *Churchill* to `SR2`. The third service uses `StructuredAnnotation` to assign a `Name` of value *Stalin* to `SR3`. Therefore, while the image is annotated correctly, the use of different tags results in the retrieval of the facial recognition results being near to impossible because the query designed to retrieve this information will need to account for all possibilities [52]. This problem can be alleviated by using the COMM ontology, which is accompanied by a Java API to translate MPEG-7 class objects into instances of the COMM concepts and

vice versa [52].

Like Loculus and the Creative Works Ontology, COMM focuses on storing metadata about a final media product. Its purpose is to label segments of a media product to track what its parts represent, which means that the metadata managed by COMM would not be comprehensive enough for tracking a media project as it progresses through the stages of the media creation process.

3.1.4 OntoFilm

OntoFilm is an ontology that conceptualises the domain of film creation and its accompanying workflows. It was developed in consultation with industry professionals after collaborative meetings confirming the need for a common ontological framework to bridge the semantic gap between pre-production concepts and post-production metadata. OntoFilm has been modelled with the three main stages of film creation in mind – pre-production, on-set production, and post-production [53]:

- The **Pre-Production Model** covers the metadata generated during the planning stage of a film production. Writers and directors use paper-based scripts to plan shots and they often add handwritten annotations to the script to record their ideas. These notes can be useful during post-production but paper-based scripts cannot be easily converted into machine interpretable metadata. As a result, the Movie Script Markup Language was developed as part of the wider ontology schema within OntoFilm to represent the main elements of a script, such as scenes, shots, characters, and dialogue, and the relationships between them. Software tools can then annotate and parse scripts according to this ontology schema.

- The **On-Set Production Model** covers the metadata generated on-set while a film is actually being recorded. OntoFilm uses the Director Notation ontology to capture scene descriptions during shooting and has also conceptualised the filming conventions

that directors can choose to follow when shooting a scene, such as the thirty degree rule:

> *"If a shot is going from one character to another without an intervening shot of something else, the camera angle should change by at least 30 degrees."* [53]

- The **Post-Production Model** covers the significant amount of metadata generated during the post-production stage, from the point that footage is ingested to the point when a final product is ready for release. Each step in post-production has been modelled in OntoFilm, such as Commercial Preview in which a candidate edit is screened to an audience to assess their reaction, and have been linked to both technical details of the media files and the scene descriptions produced on-set.

While OntoFilm is designed to handle metadata from pre-production to post-production, it only focuses on the creation of films. Researchers working on OntoFilm expressed intentions to incorporate animation for games design in future work but there are many other types of production that could benefit from an ontology for metadata management, such as VFX and unscripted television production.

3.1.5 The Deep Film Access Project

Like OntoFilm, the Deep Film Access Project (DFAP) explored the role of semantic technology in film creation. The DFAP focussed on how it could contribute to advancing the collection and management of the data and metadata through the integration of metadata automatically generated (by equipment, for example) with metadata manually inputted, such as documentation of creative decisions. The DFAP also aimed to improve the efficiency of knowledge exchange within the media creation workflow [7]. Adventure Pictures Ltd. supported this research by providing the researchers involved in the DFAP access to the full set of digital assets for one of their recent productions (*Ginger & Rosa*), their interac-

tive film production website[3], and discussions with practitioners involved in the production process [55]. These resources were analysed and used to develop an initial understanding of the knowledge that forms the foundation of the film creation process, which is shown in Figures 3.5 and 3.6.

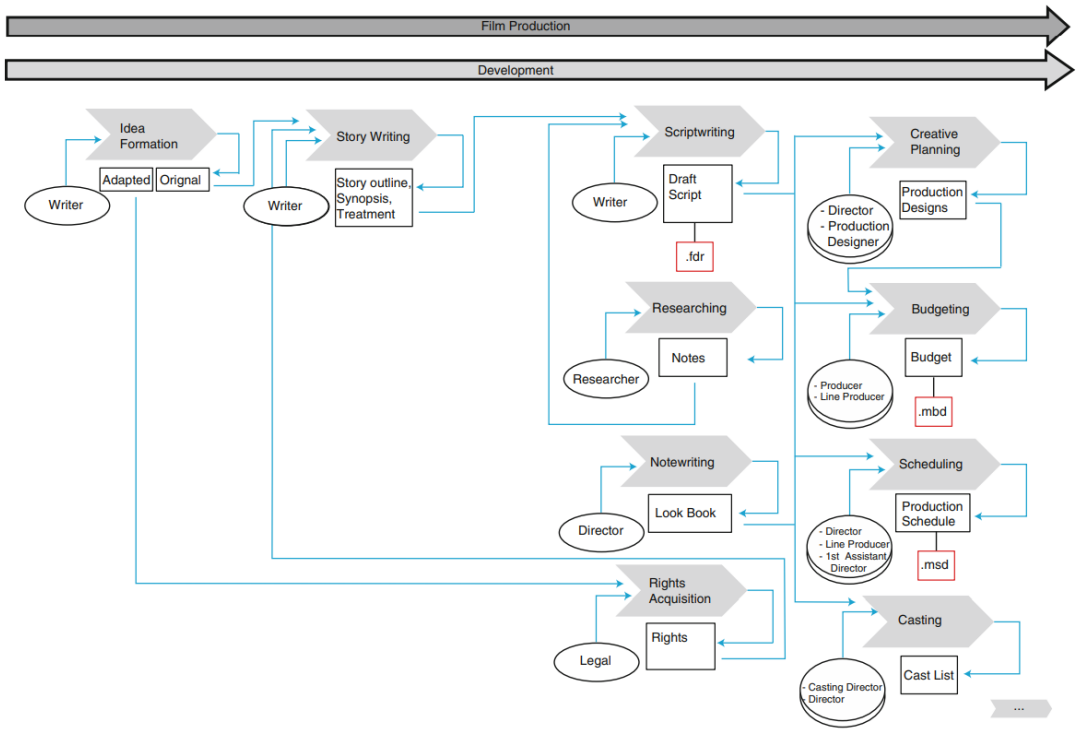


Figure 3.5: The development stage of a film creation project [55].

[3] http://www.sp-ark.org

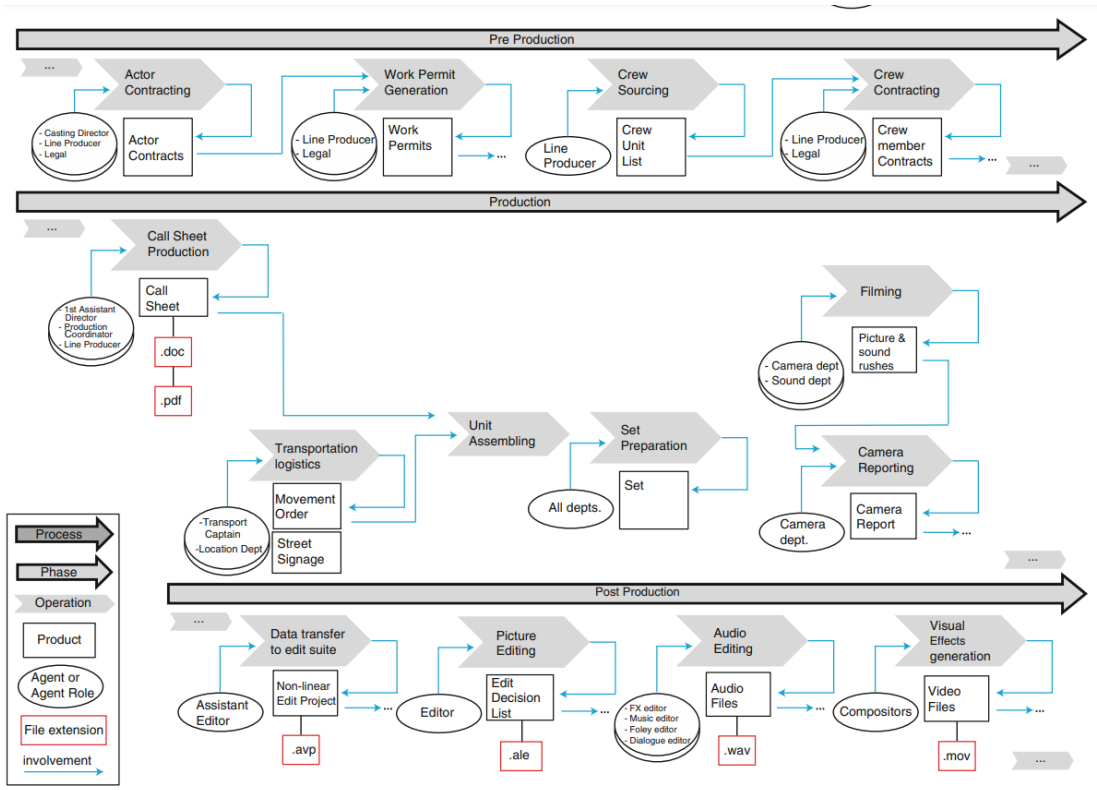


Figure 3.6: The pre- to post-production stages of a film creation project [55].

As Atkinson (2018) notes, it is important that we fully acknowledge the "sheer volume of the data that is produced and replicated" during these workflows and the "highly repetitious nature" of such data. For example, a scene can have multiple takes that are very similar to one another or a script can have multiple versions as a new copy is created every time an amendment is made [56]. The metadata associated with different versions of such data is likely to be similar too. The "highly repetitious nature" of the data associated metadata created during the media creation process demonstrates the need for a versioning feature in an ontology of metadata.

The workflows depicted in Figures 3.5 and 3.6 were formalised as an ontology using OWL. Original versions of the DFAP ontology imported the COMM and Loculus ontologies for multimedia, foundation, and film production terminology. However, as the DFAP ontology

expanded, these ontologies became "entangled" causing problems with efficiency when reasoning. As a result, the Loculus import was removed and instead the Loculus ontology was remodelled directly into the DFAP ontology [55].

3.1.6 Comparison of Existing Ontologies

In summary, there are two main types of ontology in the literature that focus on media: ontologies for archiving and ontologies for managing metadata during the creation process. Loculus, the Creative Works Ontology and COMM fall into the former category as they work with metadata describing the final product rather than the processes involved in developing the product. Meanwhile, both OntoFilm and The Deep Film Access Project enable metadata to be available at the appropriate stages of the media creation workflow but are limited in their scope, as they focus solely on film production. As a result, there is a clear gap in the current literature which indicates the need for an ontology-based metadata management solution for other sub-sectors in the media industry. This is discussed further in Chapter 6.

3.2 Ontology-based Knowledge Management Systems

Although there does not seem to be any ontology-based knowledge management systems for managing metadata during the moving image production life-cycle described in the literature, there are examples of such systems designed for other domains. These domains include industrial safety, the metal industry, flow and water quality, and industry clusters, which will be discussed in this section.

3.2.1 A Knowledge Management System for Industrial Safety

The KoMIS project [57] was a collaborative study between the University of Technology of Compiègne (UTC) and INERIS, a French institute that studies industrial environment and risk. It aimed to produce an ontology-based knowledge management system for in-

dexing and retrieving information about industrial safety from a corpus of domain-specific resources. More precisely, there are two main categories of resource: 1) text-based resources (for example, reports and articles), and 2) software tools used for specific applications within INERIS.

The system developed during this project was structured as two layers, as shown in Figure 3.7. The upper layer consisted of two ontologies – the domain ontology and the application ontology – while the bottom layer contains the instance data of these ontologies. The domain ontology describes the field of industrial safety and the application ontology describes the types of indexed resources. The resources index is a many-to-many mapping from each resource instance described by the application ontology to its corresponding instances described by the domain ontology.

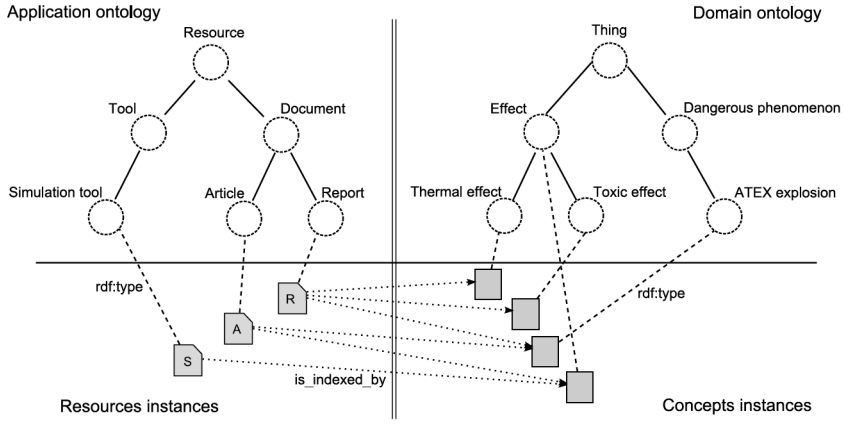


Figure 3.7: The structure of the KoMIS ontology-based KMS [57].

The indexing process consists of two principal phases, which is shown in Figure 3.8. During the first phase – instantiation – we create an instance for each resource in the corpus. The instance of a resource is an instantiation of the class in the application ontology that represents the resource's type (for example, the `Report` class). We then fill in the values of the properties of this instance and, if relevant, create relations with other resource instances. The second phase – semantic indexing – is when each resource instance is

indexed by mapping those instances to instances of the domain ontology. This is achieved using the object property `is_indexed_by`, whose domain is any resource class and whose range is any domain class.

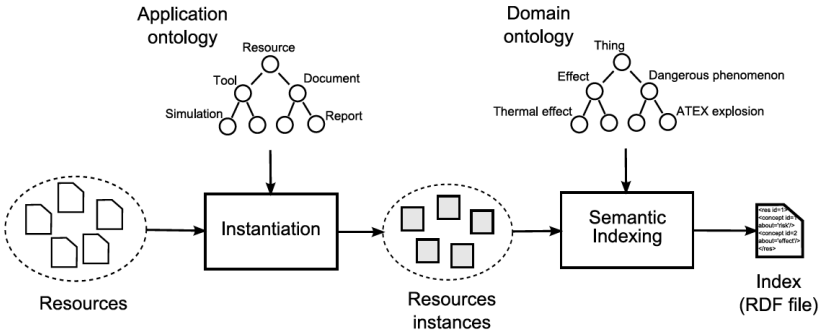


Figure 3.8: The indexing process of the KoMIS ontology-based KMS [57].

Information can then be retrieved from KoMIS using the web application, also developed as part of the project. It allows users to write their own queries using classes in the domain ontology and the application will return resources related to these classes.

3.2.2 A Knowledge Management System for the Metal Industry

Another example of an ontology-based knowledge management system is one that was designed to support the knowledge management tasks relating to the metal industry in Taiwan, undertaken at the Metal Industries Research and Development Centre (MIRDC) [58]. Figure 3.9 illustrates the structure of the knowledge management system proposed by Li et al. (2003) which, like KoMIS, consists of two ontologies.

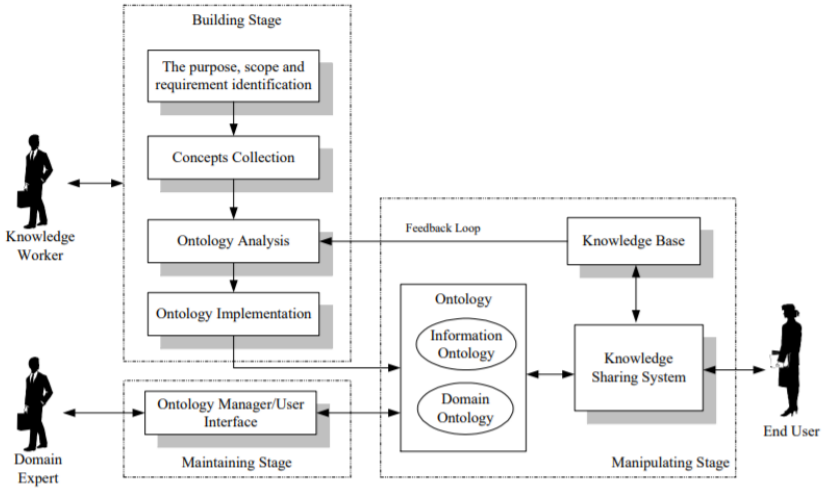


Figure 3.9: The structure of the KMS for Taiwan's metal industry [58].

The first ontology is the information ontology, which describes knowledge objects such as an electronic file, a database record, a book, etc. In this case, the Dublin Core[4] ontology is used and it consists of 15 fields that can be used to describe objects that contain knowledge. These fields are shown in Figure 3.10.

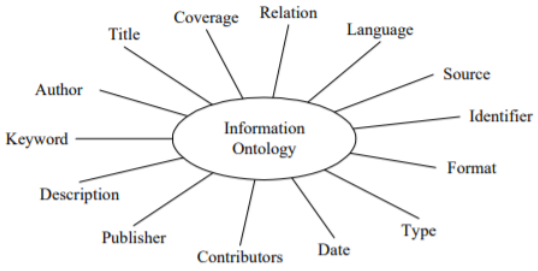


Figure 3.10: The information ontology in the KMS for Taiwan's metal industry [58].

The second ontology used is the domain ontology, which was constructed by the researchers involved in developing this knowledge management system, and its purpose is to conceptualise the metal industry. In doing this, users can search for knowledge objects relevant

[4] https://www.dublincore.org

to a particular topic related to the metal industry. A snapshot of the domain ontology is shown in Figure 3.11.

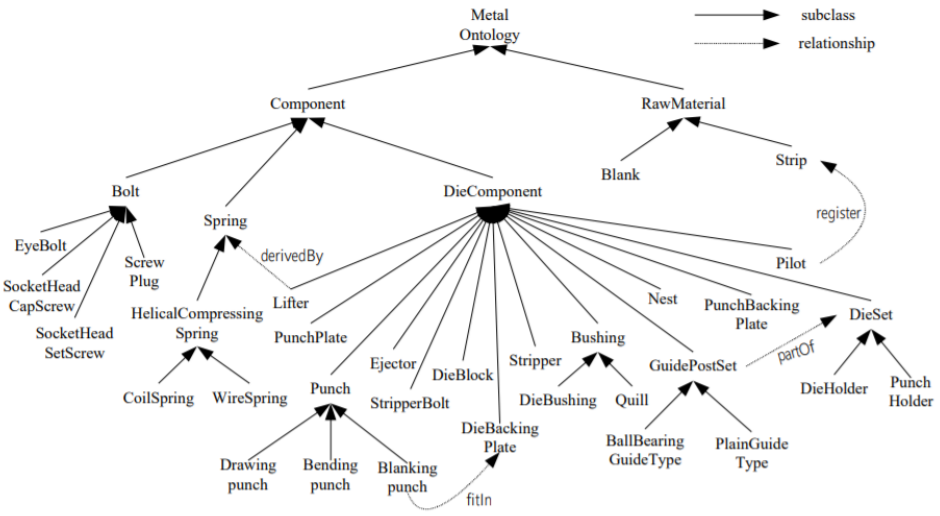


Figure 3.11: The domain ontology in the KMS for Taiwan's metal industry [58].

Users can access the knowledge management system via a web interface where they can log in and then perform knowledge management tasks, such as creating, editing, browsing, and searching for knowledge objects. However, this system is a prototype and so the researchers involved in this project have identified shortfalls including that in future work, inferences within the domain ontology should also be incorporated into knowledge searching to allow for more precise results [58].

3.2.3 A Knowledge Management System for Flow and Water Quality Modelling

The techniques for modelling flow and water quality are highly specialised tasks, where the accuracy of their predictions are dependent on the accuracy of the open boundary conditions, model parameters, and the numerical scheme used. It is therefore recognised that expert knowledge is the most important asset and these can take the form of books, technical manuals, and documents written by experts with many years of experience. In

[59], Chau (2007) presents an ontology-based knowledge management system to assist engineers in sharing, searching for, and managing knowledge relevant to the flow and water quality modelling domain.

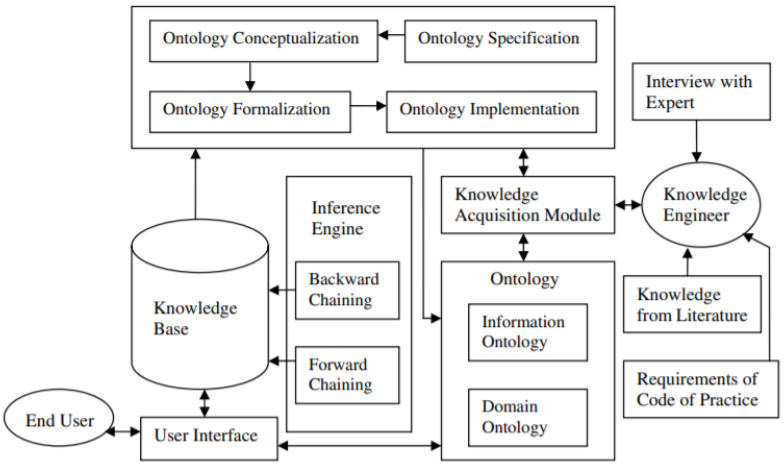


Figure 3.12: The structure of the KMS for flow and water quality [59].

As shown in Figure 3.12, the proposed knowledge management system has a three-level architecture – the application level, the description level, and the object level – whereby the ontologies are situated in description level. The ontologies then allow the user to intelligently access sources of knowledge, such as numerical data, text streams, models, video clips, etc., found in the object level. Like the system for the metal industry domain described in Section 3.2.2, this too consists of an application ontology and a domain ontology. Both systems also use the Dublin Core ontology as their information ontology to describe knowledge objects while domain-specific concepts, relations, attributes, and instances are found in the domain ontology. Figure 3.13 shows a segment of the domain ontology.

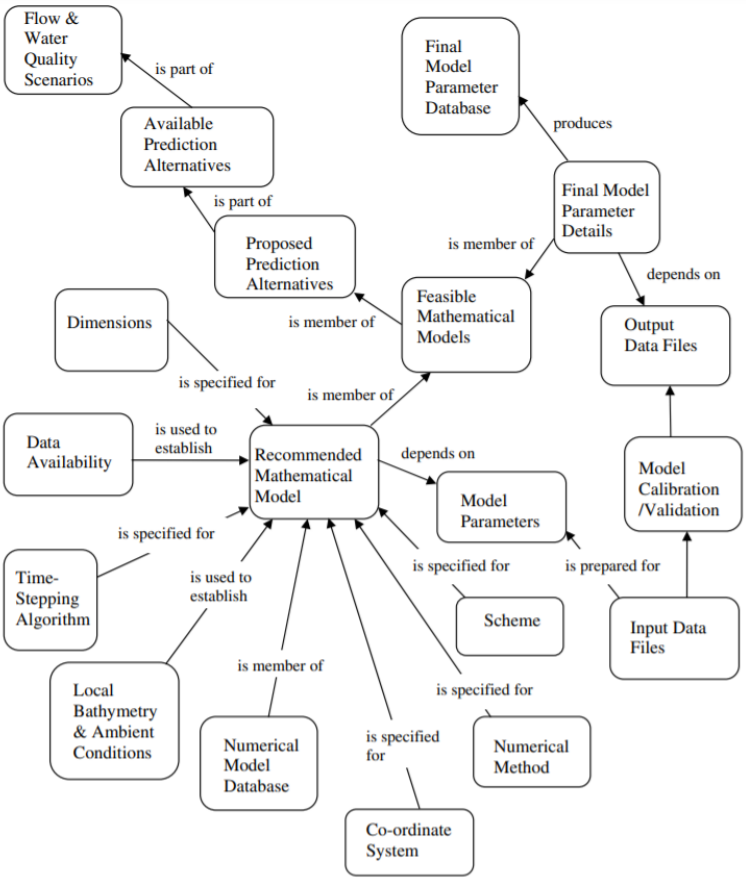


Figure 3.13: The domain ontology in the KMS for flow and water quality [59].

When a user would like to select a model for flow or water quality, they must first complete a questionnaire via a graphical user interface. Once this is done, the user must click on the INFER button to allow the knowledge management system to automatically select an appropriate model based on the user's questionnaire answers and a set of built-in rules. The system will then display the features of the suggested model. However, this system also has some shortfalls due to the recent advancement of technology. For example, Chau (2007) acknowledges that geographic information systems (GIS) and web-based applications could be integrated into future iterations of this prototype system in order to provide decision support and better accessibility [59].

3.2.4 A Knowledge Management System for Industry Clusters

In previous years, three production factors – land, labour, and capital – were considered to be the reason for the economic success of a firm. Nowadays, however, these factors are no longer sufficient and knowledge is known to play a more prominent role in giving a firm a competitive advantage. This means that most industries now use information to gain an advantage on their competitors. Sureephong et al. (2007) propose an ontology-based knowledge management system that can be used by groups of businesses that have formed a "strategic alliance" to collaborate and to share knowledge. The purpose of the system is to group rival firms into industry clusters to gain competitiveness in their market [40].

Unlike the previous three ontology-based knowledge management systems, this one is compiled of three ontologies: the generic ontology, the domain ontology, and the task ontology. The generic ontology is reusable across all domains (where a domain refers to a specific industry cluster) and contains concepts such as organisation or contact details. The domain ontology conceptualises a specific industry such as the handicraft industry while the task ontology defines the terminology associated with tasks and the problem-solving structure of those tasks. For example, one of the tasks in the handicraft industry is to select products for exportation. However, not all products are suitable for export due to their specifications, functions, etc. and there are further criteria for selecting a product to export to specific countries. Therefore, the task ontology for the handicraft industry would be designed for product selection [40].

The obvious downside of such a system is that a separate domain ontology and an associated task ontology will need to be created for each industry that this system will be used in. This is a complex and time-consuming task so there would need be an assessment of how much a business (or businesses in a collaboration) could benefit from investing resources in this, before developing new ontologies.

3.3 Summary

In summary, we have already established that the literature contains two main types of ontology for media. Those are ontologies for archiving and ontologies for managing metadata during media creation. On the other hand, we have now seen that there is a lack of literature on ontology-based knowledge management systems for the media industry, which forms a clear research gap. However, there are systems described in the literature that are designed for other domains. These systems show that the ontology-based knowledge management system being designed for this project should separate the domain knowledge from the knowledge used to solve the metadata problem. In this case, the ontology that models the structure of the metadata should be a separate layer in the overall system from the layer that stores, processes, and retrieves metadata values when required. It is important to note that we do not claim that the list of ontologies and ontology-based knowledge management systems described in this section is comprehensive.

Chapter 4

Designing and Evaluating an Ontology

An ontology design methodology outlines the process of building an ontology. Most methodologies have the same high level steps defined in Uschold and King's (1995) skeletal methodology: 1) identifying the purpose, 2) building the ontology, 3) evaluation, and 4) documentation [60]. In this chapter, we first present a selection of ontology design methodologies in Section 4.1 before describing a selection of software tools for building ontologies in Section 4.2. Then, we present popular ontology evaluation approaches in Section 4.3, the criteria that can be used to assess the quality of an ontology in Section 4.4, and lastly a set of ontology evaluation tools in Section 4.5.

4.1 Ontology Design Methodologies

There is no single correct method for modelling a domain, but it is almost always dependent on the intended application of the ontology. Developing ontologies is an iterative process which is likely to continue throughout the lifecycle of the ontology [28]. There are many different ontology design methodologies including Knowledge Engineering, DOGMA,

TOVE, Methontology, SENSUS, DILIGENT, On-To-Knowledge, UPON, and NeOn. This section will outline the stages involved in these methodologies.

4.1.1 The Knowledge Engineering Methodology

The Knowledge Engineering Methodology [61] has six stages:

1. **Definition of the domain, purpose, and scope of the ontology**: this stage clarifies the aim of the ontology and its scope to ensure that the ontology constructed can fulfil its purpose. The output of this stage is an ontology specification document defined in natural language.

2. **Acquisition and conceptualisation of domain knowledge**: this stage involves acquiring knowledge about a given domain to ensure that it can be modelled accurately in the ontology. Methods for collecting and analysing information about a domain include interviewing domain experts and textual analysis of relevant documentation. This knowledge is then conceptualised into an informal ontology containing a concepts glossary, a definition of relationships between them, and any restrictions on their use. Concepts can be placed into a hierarchy using one of three approaches: top-down, bottom-up, or a combination method. The top-down method begins with the most general classes and then creates more specialised classes, the bottom-up method begins with the most specialised classes and then generalises them, and the combination method defines the most important concepts first and then generalises and specialises them as appropriate.

3. **Reuse of existing ontologies**: this stage focuses on building a new ontology by either merging or aligning with existing ontologies to obtain some consistency across ontologies. Given two or more source ontologies, merging results in a single merged ontology and aligning results in the source ontologies remaining in their original form with links created between them.

4. **Formal specification of the ontology**: the previous stages have outputted an informal ontology written in natural language which now needs to be formalised so that it can be machine interpretable.

5. **Population with instances**: the final stage is to create instances of classes in the ontology, which can be used to analyse the consistency of the ontology. Defining an instance requires identifying the appropriate class, creation of an instance of this class, and filling in their properties' values.

6. **Evaluation and documentation**: the ontology is evaluated with respect to its consistency and completeness and then each stage of the development process must be documented.

4.1.2 The DOGMA Methodology

An ontology designed using the DOGMA methodology [61, 62] consists of two layers: the ontology base and the commitment layer. The ontology base formally defines the concepts and the relationships using lexons to enable contextual identification of these concepts and relationships. Lexons are denoted as $<\mu, t1, r, cr, t2>$ where μ is the context identifier, and $t1$, r, cr, $t2$ are term one, role, co-role, and term two respectively. The context μ is used to clarify the intended meaning of a term. For example, in the context of media production, the term 'Shot' refers to "The output of a single camera pointed at a specific angle". However, in the context of the military, the term 'Shot' refers to "The firing of a gun". A term within a context forms a concept. Meanwhile, a role expresses how one concept relates to another concept. Given that a lexon is a binary relationship, it always involves two roles. The role and co-role are inverse relationships such that we can form the following RDF triples:

```
t1 -- r -- t2
t2 -- cr -- t1
```

The commitment layer consists of a set of ontological commitments which contain a set of constraints and domain rules applied to a subset of the ontology base and a set of mappings between the ontology and application elements. The relationship between the ontology base, the commitment layer, and the applications is shown in Figure 4.1.

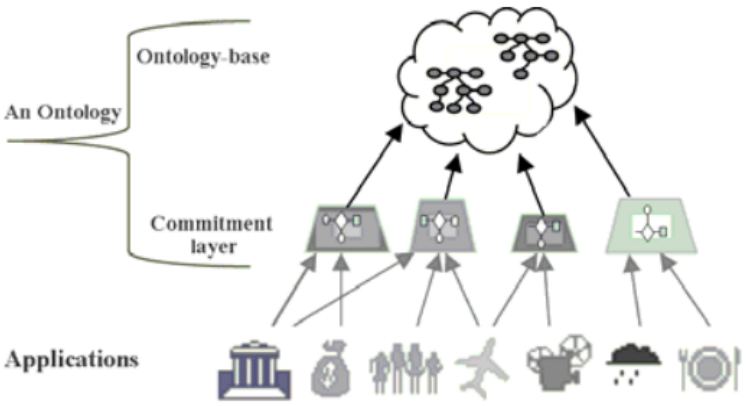


Figure 4.1: The DOGMA methodology [63].

Although the commitment layer contains the constraints and domain rules applied to a subset of the ontology base, it is actually the responsibility of the applications using the ontology to maintain logical consistency. While this may seem counter-intuitive, we should consider an example adapted from [63].

We consider an ontology engineered using the DOGMA methodology to model the domain of academic conferences and two academic conferences – Conference A and Conference B – that use this ontology. Each conference has its own set of rules that do not necessarily agree with the rules of the other conference. For example, Conference A may uniquely identify a paper using a paper number whereas Conference B may instead identify a paper using a combination of paper title and the surname of the first author. For such an ontology, a subset of the ontology base is presented in Table 4.1.

Context	Term 1	Role	Term 2
Conference	Author	Writes	Paper
Conference	Paper	WrittenBy	Author
Conference	Paper	Has	PaperTitle
Conference	PaperTitle	IsOf	Paper
Conference	Paper	Has	PaperNumber
Conference	PaperNumber	IsOf	Paper

Table 4.1: Example DOGMA ontology base.

In the commitment layer, there would be two set of rules. The first would formally define the rule that each Paper is uniquely identified by a PaperNumber and the second would formally define the rule that each (Author, PaperTitle) uniquely identifies a Paper.

Both conferences are treated as a separate applications. Therefore, it is the responsibility of each application to commit to a subset of rules defined in the commitment layer and to ensure that the rules committed to are logically consistent.

4.1.3 The TOVE Methodology

The TOVE methodology [61] was developed based on the experiences obtained during the development of an ontology for the business processes and activities domain. Like the Knowledge Engineering methodology, the TOVE methodology also has six stages:

1. **Identify motivating scenarios**: motivating scenarios are problems or questions that cannot be answered using the current knowledge base and its associated management system, such as a database. These scenarios, coupled with the possible solutions to these problems, create an informal definition of the ontology's concepts and relationships.

2. **Formulate informal competency questions**: once the ontology has been expressed

in a formal language, its expressiveness will be evaluated against its ability to answer competency questions, which are written in a natural language. The ontology must be developed to be able to both represent these questions and to answer them.

3. **Specify terminology within a formal language**: this stage consists of two parts: first the key terminology is extracted from the informal competency questions for use as the foundation for a specification of the terms in a formal language. The terminology is then formalised to allow ontology definitions and constraints to be expressed in a machine-interpretable way.

4. **Formulate formal competency questions**: the informal competency questions defined in Step 2 are then defined formally using the specifications developed in Step 3.

5. **Specify axioms and definitions within a formal language**: an ontology consists of a set of terms, whose meaning and associated constraints are defined by a set of axioms. If the axioms are insufficient to represent and answer the competency questions, further axioms need to be added, making ontology development an iterative process.

6. **Specify conditions characterising ontology completeness**: finally, the conditions used as a starting point for developing ontology completeness theorems are defined. These are the conditions where the solutions to the competency questions are complete.

4.1.4 The Methontology Framework

The methontology framework [61] consists of two main stages: the ontology development process and the ontology lifecycle, which is based on evolving ontologies. The ontology development process refers to the three categories of activities involved when building an ontology:

- **Ontology Management**: management activities include identifying the tasks that need to be performed and the time and resources needed to do this, monitoring the scheduled tasks to ensuring they are carried out as planned, and ensuring that the outputted ontology or documentation are of satisfactory quality.

- **Ontology Development**: these are grouped into pre-development, development, and post-development activities. Pre-development activities include the analysis of platforms and applications where the ontology will be used and the analysis of possibility of ontology design. Development activities include analysing the purposes, intended uses and end-users of the ontology, creating a conceptual model to represent domain knowledge and then formalising it, and building a model in a machine-interpretable language. Post-development activities include maintaining and updating the ontology.

- **Ontology Support**: support activities include using domain experts to acquire knowledge of the domain, evaluating the ontology and its associated software environments and documentation, integrating existing ontologies where applicable, writing documentation, and recording all versions of the ontology, software, and documentation by configuration management.

The ontology lifecycle is based on developing and then evolving ontology prototypes to allow incremental development. The lifecycle schedules the above ontology development activities.

4.1.5 The SENSUS Methodology

The SENSUS methodology [61] assumes that the knowledge between two ontologies can be shared easily if they are based on a common ontology. SENSUS is an ontology based on natural language and contains over 70,000 nodes obtained from various electronic sources. When building a domain ontology, SENSUS acts as the common ontology:

1. **Identify seed terms and link to SENSUS**: seed terms represent domain-specific concepts. Seed terms are identified and then manually linked to the SENSUS ontology via one of its nodes.

2. **Add paths to the root**: all the concepts on the path between the linking node and the SENSUS's root node are added to the domain ontology.

3. **Add new domain terms**: other relevant terms that have not yet been included in the domain ontology are added.

4. **Add complete subtrees**: for nodes that have many paths going through them, the entire subtree under it is sometimes added. This assumes that if many of the nodes in a subtree are relevant, then the other nodes in the subtree are probably relevant too. This step is performed manually because the decision requires some knowledge of the domain and an understanding that high-level nodes in the ontology will always have many paths so adding their entire subtrees would probably be inappropriate.

The focus of this methodology is to connect domain ontologies with the overarching SENSUS ontology and, unlike the skeletal methodology proposed by Uschold and King (1995), it does not include any steps for evaluating or documenting the ontology.

4.1.6 The DILIGENT Methodology

The DILIGENT methodology [61] was developed to support domain experts to build and evolve ontologies in a distributed setting, which is necessary as different stakeholders have varied needs and locations. This methodology begins with the build stage, where domain experts, users, and ontology developers design and build an initial ontology. The initial ontology is then made available for use, which users can adapt for their own needs in their own local environment, but the original ontology shared between all users must not change. The control board analyses and either approves or rejects any requests for changes to the shared ontology and then implements the approved changes in the next version. The new

version is then released, and users can each locally update their own ontologies.

4.1.7 The On-To-Knowledge Methodology

The On-To-Knowledge (OTK) project [64, 65] aims to "apply ontologies to electronically available information to improve the quality of knowledge management in large and distributed organisations". The OTK methodology is a set of guidelines for introducing knowledge management into such organisations. The methodology is divided into four phases: kick-off, refinement, evaluation, and maintenance, which are illustrated in Figure 4.2:

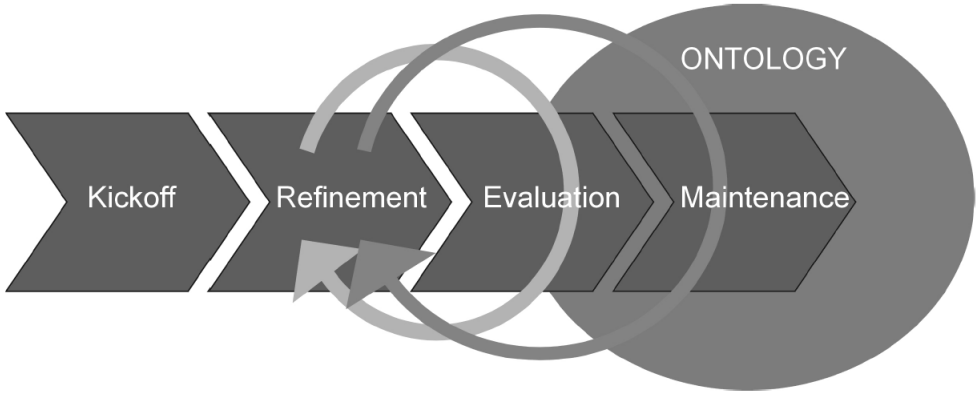


Figure 4.2: The On-To-Knowledge methodology [64].

1. **The Kick-Off Phase**: this phase aims to produce an ontology requirements specification document. This document should define the goal of the ontology, its domain and scope, applications supported by the ontology, sources of knowledge (e.g. domain experts, internal documents, and public documents), potential users and use cases, and competency questions.

2. **The Refinement Phase**: this phase aims to evolve a first version of a taxonomy (the seed taxonomy) into a seed ontology and expands to a target ontology. Refinement is split into four subphases:

   (a) **Gather a seed taxonomy**: the competency questions defined in the kick-

off phase are used to form an initial list of concepts and is-a relations. The list of concepts may be hierarchical and include obvious generalisations and specifications.

(b) **Develop the seed ontology**: the seed taxonomy is expanded into an ontology by adding relations to the taxonomy. Specifically, this means adding non-hierarchical relations between concepts (object properties) and attributes to concepts (data type properties).

(c) **Conceptualisation and formalisation**: the seed ontology is formalised into a machine-interpretable language using an ontology editing tool.

(d) **Tool supported refinement**: the ontology is further refined to get a target ontology for the application. If gaps or misconceptions are found when the ontology is analysed in the evaluation phase, these results are used as an input for the refinement phase in a later iteration.

3. **The Evaluation Phase**: this phase aims to judge the ontology and its documentation with respect to the requirements specification document. The target ontology is first checked that it can answer the competency questions analysed in the kick-off phase. It is then tested in the target application environment and feedback from users may be useful for refinement at a later stage.

4. **The Maintenance Phase**: this phase refers to updating the ontology. Specifically, ensuring that it has been thoroughly tested for all possible effects on the application before the current version is replaced with the new one.

The main feature of the On-To-Knowledge methodology is that it cycles back-and-forth between adjacent steps until the ontology reaches a satisfactory state, before moving on to the next step. For example, an ontology is refined and evaluated repeatedly before moving on to refining and maintaining the ontology. Also, unlike the skeletal methodology

proposed by Uschold and King (1995) which identifies producing documentation as the final stage of the process, the documentation stage takes place early on in the On-To-Knowledge methodology. This documentation is then referenced throughout the ontology engineering life cycle.

4.1.8 The UPON Methodology

De Nicola et al. (2005) designed the Unified Process for ONtology building (UPON), based on the widely-accepted Unified Software Development Process, after acknowledging the need for a standard method for building large-scale ontologies. UPON is an iterative, incremental, and use-case driven approach to ontology engineering consisting of a series of cycles, each with four phases resulting in a new release of the ontology at the end of a cycle. Each phase is subdivided into iterations during which five workflows take place [66]. Figure 4.3 shows the structure of one cycle.

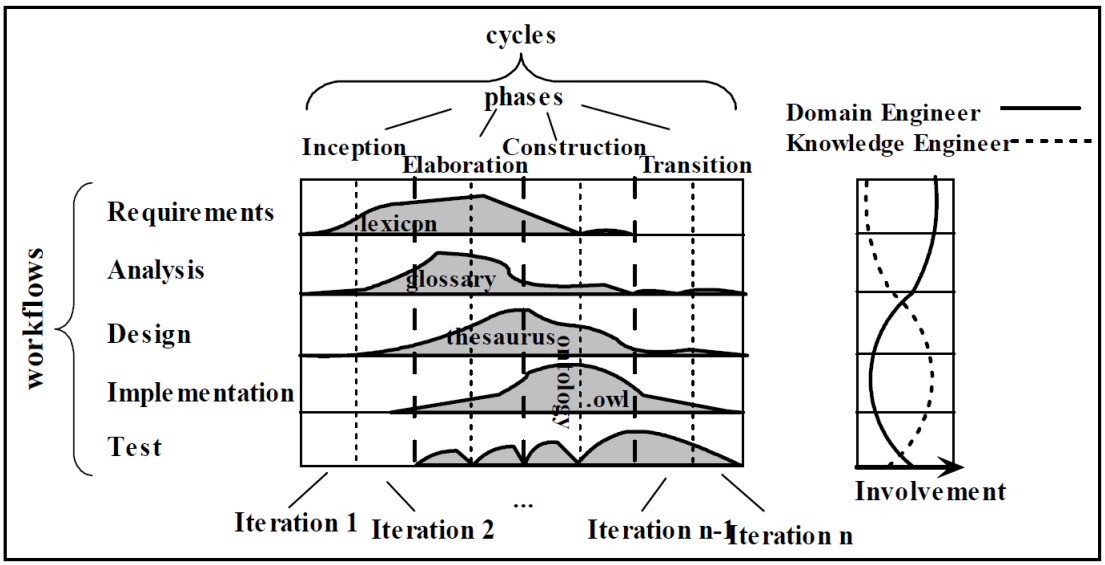


Figure 4.3: The UPON framework [66].

However not all workflows take place in every phase:

1. The **inception phase** focuses on capturing the requirements with some conceptual

analysis. The implementation and test workflows do not take place.

2. The **elaboration phase** focuses on analysis in order to identify the key concepts and structure them in a loose form.

3. The **construction phase** focuses on the design and implementation of the ontology, with some minor additional analysis taking place to identify concepts to be added to the ontology.

4. The final phase, **transition**, involves heavily testing the ontology and then releasing it for use.

The Workflows

As mentioned above, there are five workflows involved in the UPON Framework. This subsection will explain each one in detail:

- **Requirements**: This workflow aims to identify the "semantic needs and knowledge to be encoding in the ontology" [66] in order to reach an agreement between the ontology developers, the domain experts, and the end users. This involves six steps:

    1. **Determining the domain and scope**: Determining the domain refers to the identification of the elements of the real-world that are to be modelled by the ontology and defining the scope is singling out the most important concepts to be represented, and their characteristics.

    2. **Defining the purpose**: This refers to establishing the need for a new ontology to be created, its intended uses, and intended users.

    3. **Drawing a storyboard**: The domain expert draws a series of rough sketches outlining the sequence of activities in a particular scenario. In addition to giving ontology developers an greater insight into the scenarios, storyboards can also

be used to extract domain specific terminology.

4. **Creating the application lexicon**: An application lexicon is obtained from the domain experts by extracting knowledge from a document base.

5. **Identifying the competency questions**: Competency questions are identified through interviews with domain experts, brainstorming, analysing the document base, etc. and are the questions the new ontology must be able to answer. These questions are used to test that ontological commitments have been met.

6. **Use-case identification and prioritisation**: A use case refers to the scenarios that an ontology will be used in to answer competency questions. During this step, the use cases are identified and prioritised to state which ones should be focused on during early iterations.

- **Analysis**: This workflow consists of the refining and structuring of the ontology requirements. This can be done through reusing existing resources, such as interviews, documents, standards, glossaries, etc., by modelling the scenario using UML diagrams and then building the glossary of domain concepts.

- **Design**: This workflow refers to the refinement of entities and processes identified during analysis and identifying the relationships between them.

- **Implementation**: This workflow formalises the ontology in a machine interpretable language and to implement it in terms of components.

- **Test**: The test workflow aims to verify that the ontology correctly meets the requirements set during analysis. The UPON Framework considers two types of test: 1) the ontology's ability to cover the application domain, and 2) the ontology's ability to answer the competency questions.

4.1.9 The NeOn Methodology

The NeOn methodology for ontology engineering [67] is one of the outputs of the NeOn European Project[1]. It aims to provide concrete guidelines for reusing and re-engineering ontologies by suggesting a "variety of pathways" for building ontologies in the form of nine flexible scenarios. These scenarios have been identified as the ones that occur most often during ontology development, with Scenario 1 being most common, although this is not meant to be an exhaustive list. The relationships between the scenarios are illustrated in Figure 4.5:

- **Scenario 1**: *From specification to implementation*. The ontology is developed from scratch, which means without reusing existing domain knowledge resources or ontologies. In this scenario, ontology developers should:

    1. Specify the requirements that the ontology will need to fulfill and create a requirements specification document containing the purpose, scope, and the requirements of the ontology.

    2. Look for knowledge resources that could potentially be used during ontology development. They should use terms found in the requirements specification to seek out these resources.

    3. Establish the ontology life cycle and the human resources needed for the ontology development project.

    4. Perform the ontology conceptualisation activity where knowledge is organised into a meaningful structure.

    5. Perform the ontology formalisation activity where the organised knowledge is transformed into a semi-computable model.

[1] http://neon-project.org

6. Perform the ontology implementation activity where the ontology is created in a machine-interpretable ontology language, such as OWL.

- **Scenario 2**: *Reusing and re-engineering non-ontological resources*. Ontology developers analyse non-ontological resources and decide which ones can be reused to build the ontology, based on the requirements the ontology should fulfill. In this scenario, ontology developers should:

  1. Search for non-ontological resources in highly reputable domain-related websites and in repositories of internal resources within organisations.

  2. Assess the set of candidate non-ontological resources obtained in Step 1 by using the following criteria: coverage, precision, and agreement by consensus on the knowledge and terminology that the resource uses.

  3. Select the most appropriate non-ontological resources based on the results of the assessment performed in Step 2.

  4. Reverse engineer the non-ontological resources to identify their underlying components.

  5. Transform each non-ontological resource into a conceptual model and use this model to output an implementation of an ontology.

  If the ontology developers find that multiple non-ontological resources are useful for the ontology development process, then Steps 4 and 5 should produce multiple ontologies which should then be used as an input for the activities in Scenario 1.

- **Scenario 3**: *Reusing ontological resources*. Ontology developers are now more frequently reusing ontological resources, such as whole ontologies or ontology modules, as a way of avoiding "reinventing the wheel". To do this, ontology developers should:

1. Search in online repositories for existing ontologies that could potentially be reused.

2. Assess the content and granularity of the ontological resources found in Step 1 to identify whether they satisfy the requirements identified in the requirements specification document.

3. Compare the ontological resources assessed in Step 2, taking into account criteria such as content quality and code clarity.

4. Based on the comparisons made in Step 3, select the ontological resources that are the most appropriate for their own ontology's requirements.

5. Decide on the most appropriate way to reuse the selected ontological resources from the following three options: 1) reuse the resources as is, 2) perform Scenario 4 on the selected ontological resources, or 3) merge the ontological resources to obtain a new ontological resource.

6. Integrate the ontological resources selected in Step 4 into the ontology being built in Scenario 1.

- **Scenario 4**: *Reusing and re-engineering ontological resources*. Ontology developers have identified ontological resources that could be useful for their own ontology (Scenario 3) but require modification (re-engineering) to before they can be used. The ontological resource re-engineering process defined in the NeOn methodology was inspired by the software re-engineering process and it is composed of the following activities:

  – Ontological resource reverse engineering,

  – Ontological resource restructuring

– Ontological resource forward engineering

These activities can be performed at four different levels of abstraction, which are as follows (from highest to lowest level of abstraction):

– Specification

– Conceptualisation

– Formalisation

– Implementation

Figure 4.4 shows the ontological resource re-engineering model in the form of a series of paths that take into account the four levels of abstraction and the three activities listed above.

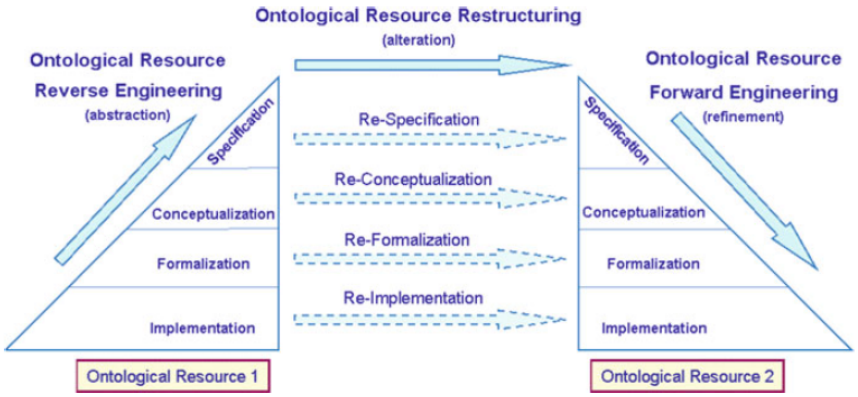


Figure 4.4: Ontological resource re-engineering model for the NeOn methodology [67].

The choice of path depends on the characteristics of the ontological resource that needs to be changed but the following types of changes can be determined:

– Restructuring at the specification level refers to the process of modifying the requirements specification. This can include changing the purpose or scope

of the ontological resource or making additions, changes, or deletions to the requirements.

- Restructuring at the conceptualisation level could include changes such as the modifying of the structure or granularity of the ontology or the addition of axioms or new concepts.

- Restructuring at the formalisation level refers to the modification of the formalisation characteristics, such as by changing ontology paradigm from description logic to a frame-based model.

- Restructuring at the implementation level is often related to the language the ontology has been implemented in. For example, re-implementation could involve translating the ontology from RDFS to OWL. It could also involve ensuring the implementation conforms to coding standards or renaming code items.

Once the ontological resources selected for reuse in Scenario 3 have been re-engineered, developers should then use these resources as an input to some of the steps in Scenario 1.

- **Scenario 5**: *Reusing and merging ontological resources*. This occurs when several, potentially overlapping, ontological resources in the same domain are selected for reuse and are merged to create a new ontological resource. Unlike Scenario 4, the ontological resources are valid separately (so no re-engineering is needed) but incomplete, so merging is required instead. Before carrying out the activities in this scenario, ontology developers should first perform the ontological resources selection activities presented in Scenario 3 (reusing ontological resources). Afterwards, the developers can:

  1. Obtain a set of alignments (overlaps) among the ontological resources.

2. (Optional) Use the alignments identified in Step 1 to create a new ontological resource by merging the existing, selected ontological resources.

Once the ontological resources selected for reuse in Scenario 5 have been processed, developers should then use this resource as an input to some of the steps in Scenario 1.

- **Scenario 6**: *Reusing, merging, and re-engineering ontological resources*. This scenario occurs when several ontological resources in the same domain are selected for reuse and are merged to create a new ontological resource, in the same way as in Scenario 5. However, this merged resource is not useful as is, so it is then re-engineered so make it fit for purpose. The ontology developer should:

  1. Use the processes described in Scenario 3 to select ontological resources to reuse.

  2. Use the processes described in Scenario 5 to find alignments among the ontological resources and then merge them.

  3. Use the processes described in Scenario 4 to re-engineer the ontological resource produced as a result of merging the ontological resources selected in Step 1.

  Once the ontological resources have been selected, merged, and re-engineered, developers should then use this outputted ontological resource as an input to some of the steps in Scenario 1.

- **Scenario 7**: *Reusing ontology design patterns*. This scenario can occur during ontology conceptualisation, formalisation, or implementation when ontology developers encounter problems in deciding how to model certain knowledge. Ontology developers can use online libraries of design patterns, such as the Ontology Design Pattern Wiki[2], to find a solution to their modelling problem, which they can integrate into their own ontology.

[2] `http://ontologydesignpatterns.org/wiki/Main_Page`

- **Scenario 8**: *Restructuring ontological resources*. In this scenario, ontology developers restructure the ontology being built as it does not meet the requirements in its current form. Ontology restructuring can involve any of the four following activities in any order or combination:

  - Ontology modularisation: when developers create different modules in the ontology which facilitates the reuse of the knowledge.

  - Ontology pruning: when developers prune branches of the ontology's hierarchy that are considered unnecessary for covering the requirements of the ontology.

  - Ontology extension: when developers extend the ontology by adding new concepts and relations.

  - Ontology specialisation: when developers specialise branches of the ontology that require more granularity.

  This scenario can take place either independently or as part of Scenario 4.

- **Scenario 9**: *Localising ontological resources*. While there are many high-quality ontologies available to anyone across the world, they are often only available in English. This scenario concerns itself with translating an ontology into natural languages different to that used in conceptualisation. To do this, ontology developers must:

  1. Select linguistic assets to improve the quality of the translation.

  2. Select the ontology labels to be translated.

  3. Obtain the translation of the ontology label in the target language.

  4. Evaluate the label translations.

  5. Update the ontology with the label translations.

These scenarios provide flexibility in how they can be combined, although any combination should include Scenario 1 as it is composed of the core activities required for any ontology development project. Additionally, the ontology support activities identified in other methodologies – namely knowledge acquisition, documentation, configuration management, evaluation, and assessment – should be performed in any of these scenarios selected for development.

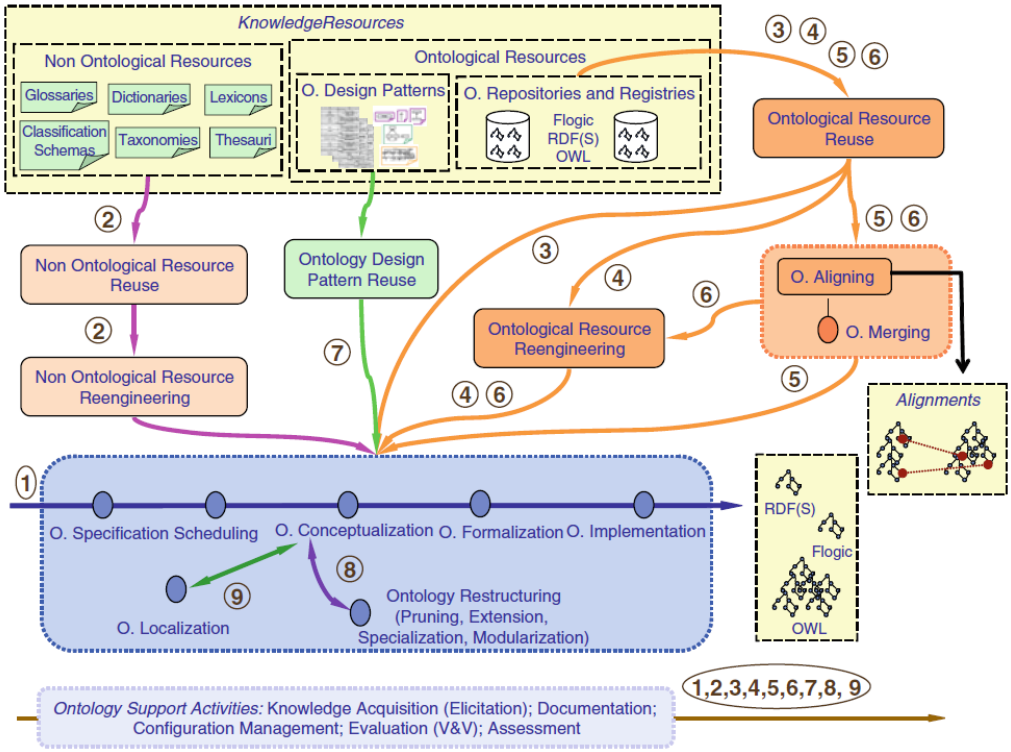


Figure 4.5: The NeOn methodology [67].

4.2 Ontology Design Tools

There are a number of ontology editors referenced in the literature, the most popular ones being Protégé, Swoop, TopBraid Composer, and the NeOn Toolkit. In this section, we shall describe these four ontology engineering tools in detail before mentioning other ontology editors that were no longer available to download due to expired links.

4.2.1 Protégé

Protégé[3] is a free, open-source ontology editor developed by the Stanford Center for Biomedical Informatics Research at the Stanford University School of Medicine. The Protégé project began in the 1980s and is the industry-standard software for building and maintaining ontologies [68]. Protégé exists as both a desktop and as a web-based application. The desktop version supports many advanced features while the web version is simpler to use and can be accessed using any web browser. As of 2015, the web application version exceeded the desktop client in its degree of usage [69]. Figures 4.6 and 4.7 show the Pizza[4] ontology, an example ontology often used in tutorials for OWL, in the Protégé desktop client and Figure 4.8 shows this ontology in Web Protégé.

[3] https://protege.stanford.edu
[4] https://protege.stanford.edu/ontologies/pizza/pizza.owl

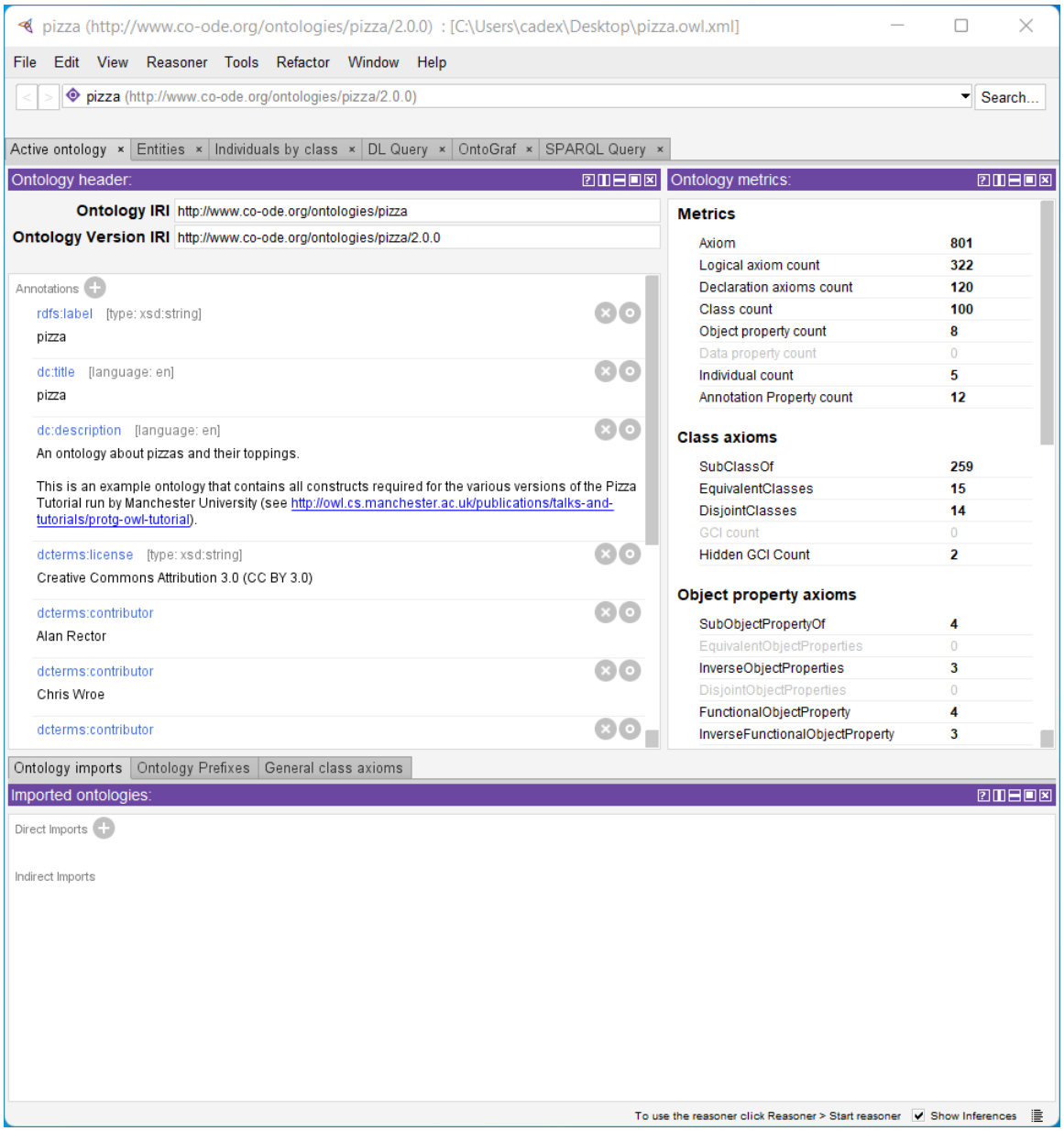


Figure 4.6: The ontology summary view in Protégé Desktop.

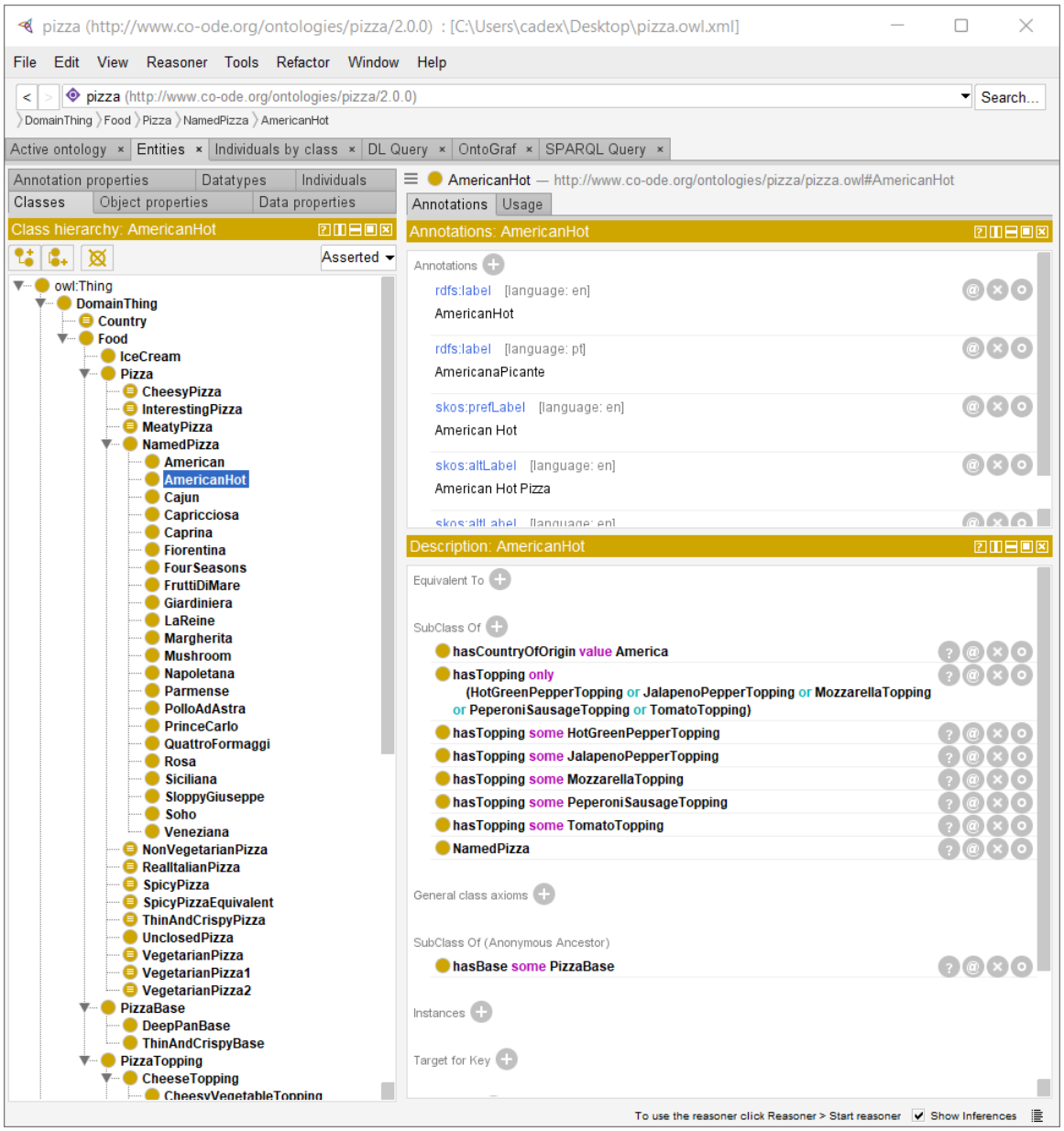


Figure 4.7: The details of a class in Protégé Desktop.

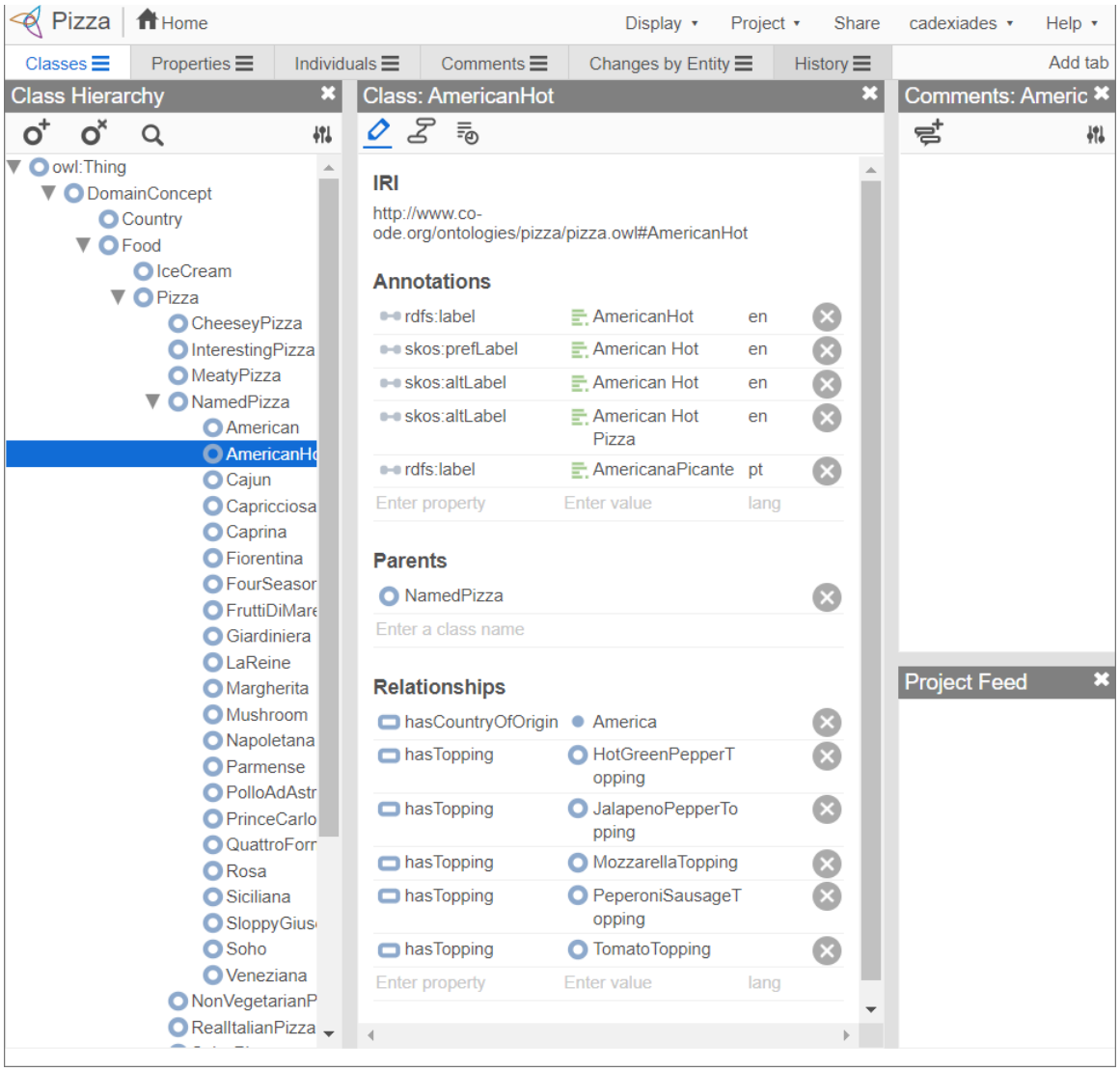


Figure 4.8: The details of a class in Web Protégé.

4.2.2 Swoop

Like Protégé, Swoop[5] is also a free, open-source ontology editor. It was developed by the University of Maryland and is tailored for OWL, offering a number of syntax views such as abstract syntax, RDF/XML, and Turtle. Swoop also offers reasoning capabilities and allows ontology developers to navigate an ontology through hyperlinks displayed within the

[5] https://github.com/ronwalf/swoop

user interface [70]. Figures 4.9 and 4.10 show the Pizza ontology in the Swoop ontology editor.

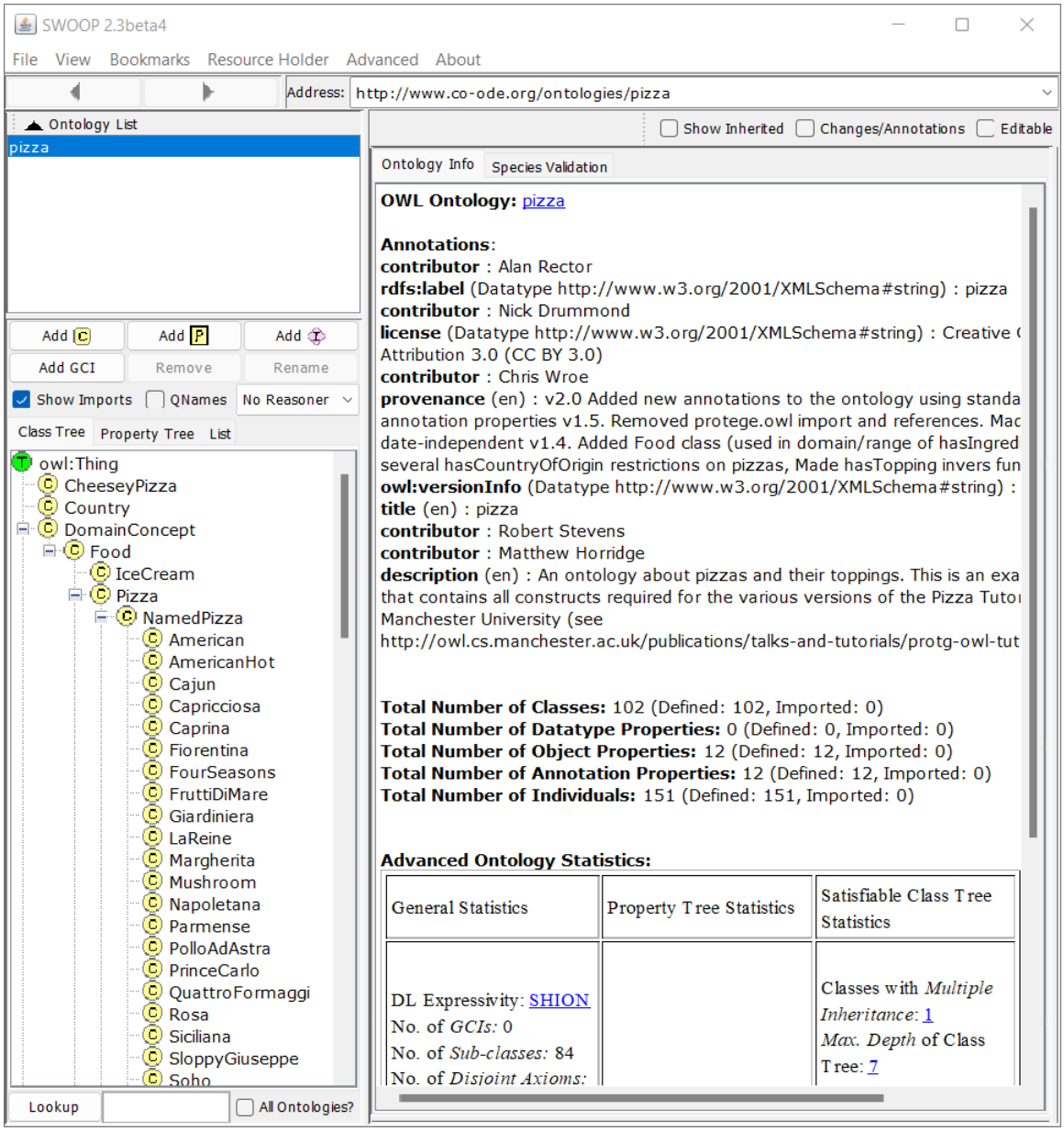


Figure 4.9: The ontology summary view in Swoop.

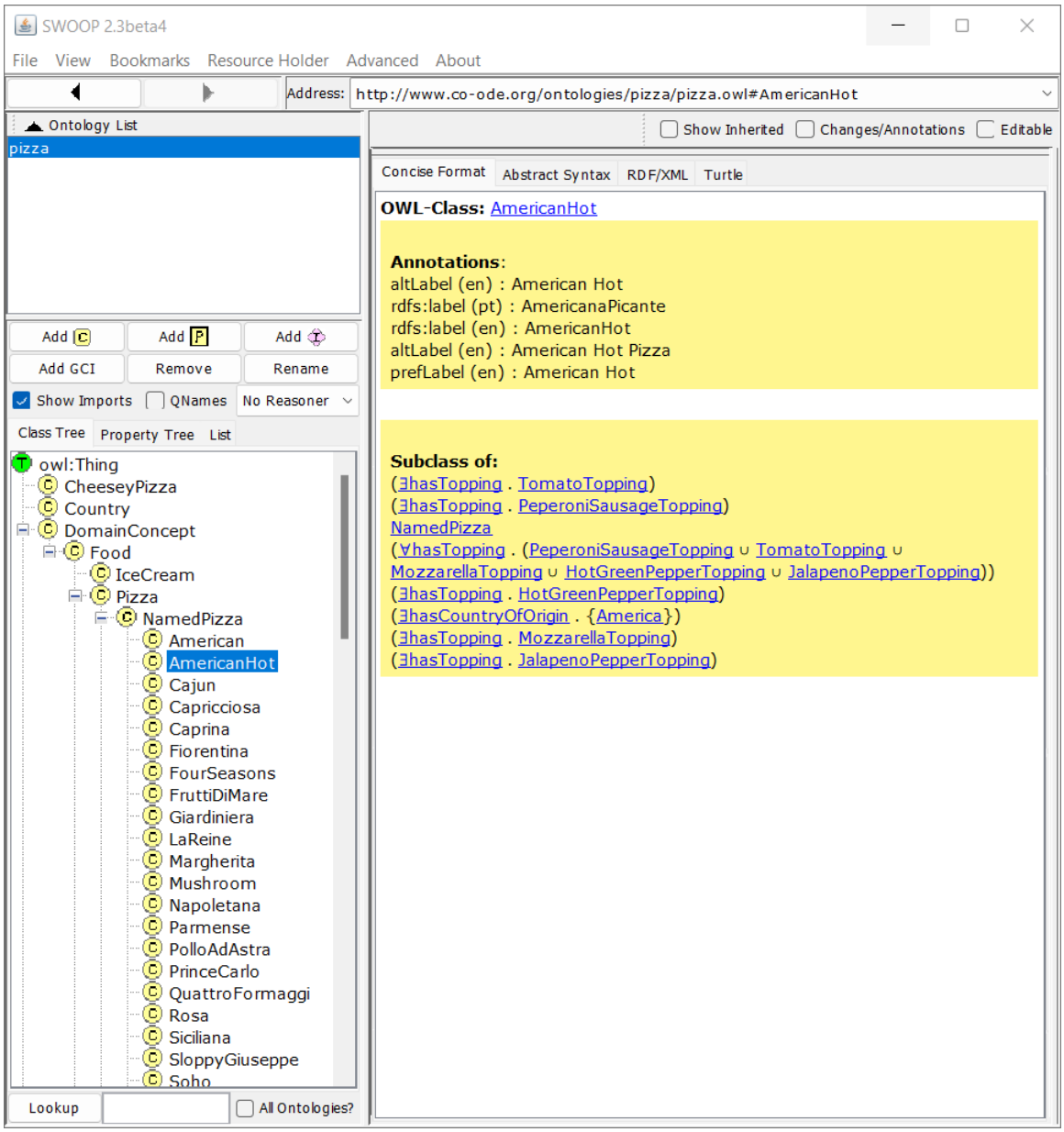


Figure 4.10: The details of a class in Swoop.

4.2.3 TopBraid Composer

Unlike Protégé and Swoop, TopBraid Composer[6] is a commercial ontology editor developed by Top Quadrant with similar core features to other free ontology editors. It comes in three

[6] https://www.topquadrant.com/products/topbraid-composer

editions – Free, Standard, and Maestro Edition – although this comparison will focus on only the Free edition. The Free Edition only provides a core set of features and functionality which includes [70]:

- Loading and saving any OWL2 file in formats such as RDF/XML or Turtle.
- Supporting development of RDF(S) and OWL-based ontologies.
- Supporting reasoning and consistency checking mechanisms.
- Querying ontologies using SPARQL.

Although the Standard and Maestro Editions of TopBraid Composer have some additional features to streamline the ontology engineering process such as import facilities, advanced refactoring, and support for SPARQLMotion[7] (a scripting language that uses diagrams to describe data processing pipelines), these were not needed for this research.

4.2.4 The NeOn Toolkit

The NeOn Toolkit[8] is another open source ontology engineering tool which was developed as another part of the NeOn European Project. It has been designed with a modular architecture that is inherited from its underlying Eclipse platform. The core version of the NeOn Toolkit provides the following core features [71]:

- Loading and saving any OWL2 file in formats such as RDF/XML or Turtle.
- Supporting development of RDF(S) and OWL-based ontologies.
- Searching for and finding references within an ontology.
- Viewing the ontology in a textual form (e.g. as an XML file) via the interface.

[7] `https://sparqlmotion.org`
[8] `http://neon-toolkit.org/wiki/Main_Page.html`

However there is often a need for further functionality, especially from ontology developers, and so there is a range of plugins which can be integrated into the toolkit due to its modular structure. Currently, there are 12 plugins for the latest version of the NeOn Toolkit (Version 2.5) linked on its official website and these provide features such as:

- Querying ontologies using SPARQL.
- Exporting from the ontology a series of HTML pages that act as documentation.

4.2.5 Other Tools

The literature also makes references to other ontology editors such as Apollo[9] and OntoStudio[10] [70]. However, during this work, the download links provided for both of these tools were broken and so were unable to be considered for use in this project.

4.3 Ontology Evaluation Approaches

Gómez-Pérez (2001) describes the process of building an ontology as "anarchistic" because development teams usually follow their own steps in the development process [72]. There are many ontology design methodologies, but they all typically have the same core stages, including an evaluation stage. Generally speaking, evaluations are "conducted for one or two main reasons: to find areas for improvement and/or to generate an assessment of overall quality or value (usually for reporting or decision-making purposes)" [73]. More specifically, Gómez-Pérez (2001) describes *ontology* evaluation as "ensuring that its definitions (a definition is written in natural language and in a formal language) correctly implement ontology requirements and competency questions or perform correctly in the real world" [72]. Meanwhile, Cantador et al. (2006) defined ontology evaluation as "the process of assessing the quality and the adequacy of an ontology for being used in a specific context, for a specific goal" [74]. Developing methods for evaluating ontologies is an emerging field

[9] http://apollo.open.ac.uk
[10] https://www.w3.org/2001/sw/wiki/OntoStudio

with academics proposing various approaches and metrics for achieving this. In this section, we describe the common approaches for ontology evaluation, while Table 4.2 illustrates the approaches suggested by various academics in the literature.

Evaluation Approach	**Yu et al. [75]**	**Raad and Cruz [76]**	**Tartir et al. [77]**	**Brank et al. [78]**
Gold Standard	x	x		x
Data or Corpus-based		x		x
Criteria-based	x	x		
Task-based	x	x		x
Evolution-based			x	
Logical or Rule-based			x	
Metric or Feature-based			x	
Human Inspection				x

Table 4.2: Ontology evaluation approaches by academic.

4.3.1 Gold Standard Evaluation

This approach, also known as ontology alignment or ontology mapping, is usually applied to ontologies that have been generated automatically to assess the generation process. It works by comparing a candidate ontology with another ontology deemed to be the benchmark, or the 'gold standard'. This approach is efficient in evaluating ontological accuracy where high accuracy is defined as having correct definitions and descriptions of classes, properties and individuals [76].

4.3.2 Data-driven or Corpus-based Evaluation

A corpus-based approach, also known as a data-driven approach, evaluates how far an ontology covers a given domain. Rather than comparing one ontology with another, the candidate ontology is compared with the content of a text corpus that significantly covers

the domain in question [76]. Brewster et al. (2004) presents a number of options for using a corpus to evaluate a candidate ontology [79]:

1. Perform an automated term extraction on the corpus and count how many of the extracted terms can be found in the ontology.

2. Use a vector space model to obtain a representation of the terms in both the corpus and the candidate ontology to produce an overall measure of 'fit'. A vector space model is a representation of data and their relationships with one another in vector form. One of the most well-known methods for producing a vector space model is term frequency–inverse document frequency (tf-idf), which is where the frequency of words occurring in a set of documents is used to determine the similarity of those documents to one another. In the case of ontologies, we can use tf-idf to match terminology in a candidate ontology with terminology in a corpus or corpora. Gulić et al. (2013) propose a matcher that combines tf-idf with synonym recognition to improve the results of this process [80].

3. Use conditional probability to find the 'best fit' of an ontology. Several ontologies can be compared and the one that maximises the conditional probability given the corpus is the best fit ontology.

Gold standard and corpus-based evaluation have several similarities so they cover the same evaluation criteria – accuracy, completeness, and conciseness. However, corpus-based approaches are often more applicable because it is easier to find a corpus that covers the same domain of a candidate ontology than it is to find another ontology to compare it with. This is because corpora are often provided by domain experts in the early stages of the ontology engineering process whereas an ontology that covers the same domain may not exist or, if it does, it may not be publicly available due to licensing restrictions [76].

4.3.3 Criteria-based Evaluation

This approach evaluates an ontology using a pre-determined set of criteria. These criteria are used to assess an ontology separately from the application, meaning that while the ontology criteria may be met, it may not satisfy the needs of the application. Therefore, criteria-based evaluation is most suitable for evaluating the clarity of an ontology. For example, one could automatically evaluate the ontology with respect to its structure by measuring the number of nodes or whether there are any cycles in the directed graph [76].

4.3.4 Task-based Evaluation

This approach evaluates an ontology based on its ability to complete a set of tasks. Unlike criteria-based evaluation, this approach can be used to assess whether an ontology is suitable for the intended application. However, the evaluation process for one application may not apply to another so a different evaluation needs to be carried out for each task. As such, task-based evaluation approaches are most efficient in assessing how adaptable an ontology is [76].

4.3.5 Evolution-based Evaluation

Because knowledge is always being added to domains, ontologies naturally evolve over time to include that new knowledge. Of course, this knowledge needs to be added properly in order to model the domain accurately. Evolution-based evaluation tracks how an ontology changes over time across different versions to detect any invalid changes made [77].

4.3.6 Logical or Rule-based Evaluation

This approach to ontology evaluation uses rules either provided by users or built into the ontology design languages to detect inconsistencies in the ontology. For example, if two classes in an ontology are said to be disjoint, the ontology cannot then have a single instance that is a member of both classes. Developers can also use restrictions in the ontology to

create additional rules [77].

4.3.7 Metric or Feature-based Evaluation

Unlike the two other approaches suggested by Tartir et al. (2010), metric-based evaluation techniques "offer a quantitative perspective of ontology quality". Such techniques iterate through the ontology to compile statistics (or metrics) about both the schema itself and the knowledge base modelled by the schema. Examples of metrics include relationship and attribute richness, class connectivity, and class importance. Metric-based evaluation approaches can also be thought of as synonymous to Raad and Cruz's criteria-based evaluation because the criteria can be quantified using metrics [77].

4.3.8 Human Inspection

This final approach refers to when evaluation is performed by humans who manually assess how well an ontology meets a set of predetermined criteria and requirements. While human inspection has the drawback of being susceptible to human error when used to measure quantifiable criteria, it is a good approach for generating qualitative data about the quality of the ontology, such as when assessing completeness [78].

4.4 Ontology Evaluation Criteria

In order to apply evaluation approaches to an ontology, a set of criteria needs to be established which will be used to test the ontology against. Some evaluation criteria, such as consistency, can be measured automatically with reasoners or evaluation tools. However, it is not possible for all criteria to be measured using automatic tools. For example, accuracy is inquantifiable so it requires a domain expert to manually verify that the definitions are correct in terms of the real-world, which is not always feasible with large ontologies. Meanwhile, completeness can be demonstrated by an ontology but cannot always be proven [75]. Various criteria has been proposed in the literature for evaluating ontologies and upon anal-

ysis, significant overlap was found between the criteria suggested by different researchers. These criteria have been collated and explained in the following subsections.

4.4.1 Consistency

An individual definition in an ontology is considered consistent if and only if no other definition exists that produces contradictory results. Therefore, an ontology is consistent if no such contradictory definitions exist in the whole ontology. Zhu et al. (2017) states that there are two types of consistency – internal and external – which can both be quantified by counting the total number of instances of inconsistencies, where a lower value is better [81].

Internal Consistency

Internal consistency is when there is no self-contradiction within the ontology. Internal inconsistencies include circularity errors, which is when a class is defined as a specialisation or generalisation of itself such that there is a cycle in the class hierarchy, and partition errors such as defining an individual as an instance of two disjoint classes. Further inconsistencies can arise in an ontology due to custom restrictions that contradict other assertions. Given that these errors centre around issues in the ontology's logic, these can be measured automatically using reasoner tools [72, 76].

External Consistency

Meanwhile, external consistency or 'accuracy' is when the ontology is consistent with the domain knowledge. For example, errors caused by the developer classifying a concept as a subclass of a concept that it does not belong to causes an external inconsistency [72, 76]. Unlike internal inconsistencies, measuring external inconsistencies require manually comparing the knowledge in ontology with the domain knowledge, which is very time consuming. Lopez et al. (1999) suggests that one way of measuring correctness is to use interviews with domain experts and any literature provided during the knowledge acquisition

stage of the ontology design process to verify that real-world concepts have been modelled correctly [82]. In contrast, Zhu et al. (2017) suggests a more scientific approach, using the following metrics based on the accuracy of a candidate ontology, O, with respect to a benchmark or gold-standard ontology Ω [81]:

1. The ratio of correct concepts of an ontology with respect to a benchmark ontology can be calculated as a percentage of the overall number of classes in the candidate ontology. This is shown in Equation 4.1 where C denotes the set of classes in the candidate ontology and $\Omega \vdash c$ denotes that a class c can be found in or can be derived from the benchmark ontology. Therefore, $\|\{c \in C | \Omega \vdash c\}\|$ is the number of classes in the candidate ontology that are found in or can be derived from the benchmark ontology and $\|C\|$ is the number of classes in the candidate ontology overall.

$$\text{RCC}^{\Omega}(O) = \frac{\|\{c \in C | \Omega \vdash c\}\|}{\|C\|} \tag{4.1}$$

2. The ratio of correct instances (RCI) for a given class is defined as the number of instances for that class c (denoted as I^c) in the candidate ontology that are found in or can be derived from the benchmark ontology, divided by the total number of instances of that class. If a class has no instances, the RCI for that class is 1. This is shown in Equation 4.2.

$$\text{RCI}^{\Omega}(c) = \begin{cases} 1 & \text{if } I^c = \emptyset \\ \dfrac{\|\{\alpha \in I^c | \Omega \vdash \alpha \in I^c\}\|}{\|I^c\|} & \text{if } I^c \neq \emptyset \end{cases} \tag{4.2}$$

The average ratio of correct instances in an ontology with respect to a benchmark ontology can be calculated as a percentage of the overall number of classes in the candidate ontology. The RCI for every class in the candidate ontology is summed and the total divided by the total number of classes. This is shown in Equation 4.3 where

$\text{RCI}^{\Omega}(c)$ is the ratio of correct instances (as defined in Equation 4.2) for a given class c and $\|C\|$ is the number of classes in the candidate ontology.

$$\text{ARCI}^{\Omega}(O) = \frac{\sum_{c \in C} \text{RCI}^{\Omega}(c)}{\|C\|} \tag{4.3}$$

3. The ratio of correct attributes (RCA) for a given class is defined as the number of attributes for that class c (denoted as A^c) in the candidate ontology that are found in or can be derived from the benchmark ontology, divided by the total number of attributes for that class. If a class has no attributes, the RCA for that class is 1.

$$\text{RCA}^{\Omega}(c) = \begin{cases} 1 & \text{if } A^c = \emptyset \\ \dfrac{\|\{\varphi \in A^c | \Omega \vdash \varphi \in A^c\}\|}{\|A^c\|} & \text{if } A^c \neq \emptyset \end{cases} \tag{4.4}$$

The average ratio of correct attributes in an ontology with respect to a benchmark ontology can be calculated as a percentage of the overall number of classes in the candidate ontology. The RCA for every class in the candidate ontology is summed and the total divided by the total number of classes. This is shown in Equation 4.5 where $\text{RCA}^{\Omega}(c)$ is the ratio of correct attributes (as defined in Equation 4.4) for a given class c and $\|C\|$ is the number of classes in the candidate ontology.

$$\text{ARCA}^{\Omega}(O) = \frac{\sum_{c \in C} \text{RCA}^{\Omega}(c)}{\|C\|} \tag{4.5}$$

4. The average ratio of correct relations in an ontology with respect to a benchmark ontology can be calculated as a percentage of the overall number of relations in the candidate ontology. This is shown in Equation 4.6 where $\|\{(x, y) \in r | \Omega \vdash (x, y) \in r\}\|$ is the number of object property instances in the candidate ontology that are

found in or can be derived from the benchmark ontology and $\|R\|$ is the number of object properties defined in the candidate ontology.

$$\text{ARCR}^{\Omega}(O) = \frac{\sum_{r \in R} \|\{(x,y) \in r | \Omega \vdash (x,y) \in r\}\|}{\|R\|} \tag{4.6}$$

It is important to note that comparing a candidate ontology to a gold-standard ontology is not always possible because a gold-standard ontology may not be available. Based on a non-exhaustive search of the literature, there is no documented estimate of how often a gold standard ontology is available.

4.4.2 Completeness

An ontology can only be considered complete if and only if all knowledge that should be in the ontology is either explicitly asserted or can be inferred via existing definitions. Examples of errors resulting in incomplete ontologies include omitting the partition definitions between a set of classes and classifying concepts without accounting for all of them, such as only considering string and brass instruments in an ontology about the musical instruments domain while overlooking other categories of instrument [72, 76, 83]. Zhu et al. (2017) state that there are two types of incompleteness that a candidate ontology, O, should be measured for. These are syntactic incompleteness and semantic incompleteness, and both have four metrics used to quantify them, which compares the candidate ontology to a benchmark ontology denoted by Ω [81].

Syntactic Incompleteness

Syntactic completeness refers to the vocabulary coverage of a candidate ontology, O, with respect to a benchmark or gold-standard ontology Ω on O's classes (Cov_C^{Ω}), instances (Cov_I^{Ω}), attributes (Cov_A^{Ω}), and relations (Cov_R^{Ω}). In other words, syntactic completeness refers to how much of the vocabulary in the candidate ontology matches exactly that of

the benchmark ontology.

1. The vocabulary coverage of an ontology's classes with respect to those in a benchmark ontology can be calculated as a percentage of the number of classes in the benchmark ontology. This is shown in Equation 4.7 where $\|C \cap C'\|$ is the number of classes that appear in both the candidate ontology and the benchmark ontology and $\|C'\|$ is the number of classes in the benchmark ontology.

$$\text{Cov}_C^{\Omega}(O) = \frac{\|C \cap C'\|}{\|C'\|} \tag{4.7}$$

2. The vocabulary coverage of an ontology's instances with respect to those in a benchmark ontology can be calculated as a percentage of the number of instances in the benchmark ontology. The number of instances appearing in both the candidate and the benchmark ontology is calculated by summing the size of the intersection for each class. This is shown in Equation 4.8 where $\|I^c \cap I'^c\|$ denotes the number of instances for a class c that appear in both the candidate ontology and the benchmark ontology and $\|I'^{c'}\|$ is the number of instances in the benchmark ontology for a given class, c.

$$\text{Cov}_I^{\Omega}(O) = \frac{\sum_{c \in C} \|I^c \cap I'^c\|}{\sum_{c' \in C'} \|I'^{c'}\|} \tag{4.8}$$

3. The vocabulary coverage of an ontology's attributes with respect to those in a benchmark ontology can be calculated as a percentage of the number of attributes in the benchmark ontology. The number of attributes appearing in both the candidate and the benchmark ontology is calculated by summing the size of the intersection for each class. This is shown in Equation 4.9 where $\|A^c \cap A'^c\|$ denotes the number of attributes for a class c that appear in both the candidate ontology and the bench-

mark ontology and $\|A'^{c'}\|$ is the number of attributes in the benchmark ontology for a given class, c.

$$\text{Cov}_A^{\Omega}(O) = \frac{\sum_{c \in C} \|A^c \cap A'^c\|}{\sum_{c' \in C'} \|A'^{c'}\|} \tag{4.9}$$

4. The vocabulary coverage of an ontology's relations with respect to those in a benchmark ontology can be calculated as a percentage of the number of relations in the benchmark ontology. This is shown in Equation 4.10 where $\|r \cap r'\|$ is the number of instances of object properties that appear in both the candidate ontology and the benchmark ontology for an object property (or 'relation') R and $\|r'\|$ is the number of object property instances in the benchmark ontology for a given object property.

$$\text{Cov}_R^{\Omega}(O) = \frac{\sum_{r \in R} \|r \cap r'\|}{\sum_{r' \in R'} \|r'\|} \tag{4.10}$$

The overall vocabulary coverage of a candidate ontology compared to a benchmark ontology can be calculated by combining the previous four syntactic coverage calculations, as shown in Equation 4.11. Essentially, this equation divides total number of components (i.e. classes, instances, attributes, and relations) that appear in both the candidate and the benchmark ontology by the total number of components in the benchmark ontology.

$$S_C = \|C \cap C'\|$$
$$S_I = \sum_{c \in C} \|I^c \cap I'^c\|$$
$$S_A = \sum_{c \in C} \|A^c \cap A'^c\|$$
$$S_R = \sum_{r \in R} \|r \cap r'\|$$

$$\text{Cov}^{\Omega}(O) = \frac{S_C + S_I + S_A + S_R}{\|C'\| + \sum_{c' \in C'} \|I'^{c'}\| + \sum_{c' \in C'} \|A'^{c'}\| + \sum_{r' \in R'} \|r'\|} \tag{4.11}$$

Semantic Incompleteness

Semantic completeness refers how much of the vocabulary in the benchmark ontology Ω can be derived from the candidate ontology O. Like syntactic incompleteness, there are four metrics for calculating semantic incompleteness based on classes (SCov^{Ω}_{C}), instances (SCov^{Ω}_{I}), attributes (SCov^{Ω}_{A}), and relations (SCov^{Ω}_{R}).

1. The semantic coverage of an ontology's classes with respect to those in a benchmark ontology can be calculated as a percentage of the number of classes in the benchmark ontology. This is shown in Equation 4.12 where $\|\{c' \in C' | O \vdash c'\}\|$ is the number of classes in the benchmark ontology that are found in or can be derived from the candidate ontology and $\|C'\|$ is the number of classes in the benchmark ontology.

$$\text{SCov}^{\Omega}_{C}(O) = \frac{\|\{c' \in C' | O \vdash c'\}\|}{\|C'\|} \tag{4.12}$$

2. The semantic coverage of an ontology's instances with respect to those in a benchmark ontology can be calculated as a percentage of the number of instances in the

benchmark ontology. This is shown in Equation 4.13 where $\|\{\alpha \in I'^{c'} | O \vdash c', O \vdash \alpha\}\|$ is the number of instances for a class c in the benchmark ontology that are found in or can be derived from the candidate ontology and $\|I'^{c'}\|$ is the number of instances in the benchmark ontology for a given class, c.

$$\text{SCov}_I^{\Omega}(O) = \frac{\sum_{c' \in C'} \|\{\alpha \in I'^{c'} | O \vdash c', O \vdash \alpha\}\|}{\sum_{c' \in C'} \|I'^{c'}\|} \tag{4.13}$$

3. The semantic coverage of an ontology's attributes with respect to those in a benchmark ontology can be calculated as a percentage of the number of attributes in the benchmark ontology. This is shown in Equation 4.14 where $\|\{\varphi \in A'^{c'} | O \vdash c', O \vdash \varphi\}\|$ is the number of attributes for a class c in the benchmark ontology that are found in or can be derived from the candidate ontology and $\|A'^{c'}\|$ is the number of attributes in the benchmark ontology for a given class, c.

$$\text{SCov}_A^{\Omega}(O) = \frac{\sum_{c' \in C'} \|\{\varphi \in A'^{c'} | O \vdash c', O \vdash \varphi\}\|}{\sum_{c' \in C'} \|A'^{c'}\|} \tag{4.14}$$

4. The semantic coverage of an ontology's relations with respect to those in a benchmark ontology can be calculated as a percentage of the number of relations in the benchmark ontology. This is shown in Equation 4.15 where $\|\{(a,b) \in r' | O \vdash (a,b) \in r'\}\|$ is the number of object property instances in the benchmark ontology that are found in or can be derived from the candidate ontology and $\|r'\|$ is the number of object property instances in the benchmark ontology for a given object property.

$$\mathsf{SCov}_R^{\Omega}(O) = \frac{\sum_{r' \in R'} \|\{(a,b) \in r' | O \vdash (a,b) \in r'\}\|}{\sum_{r' \in R'} \|r'\|} \tag{4.15}$$

The overall semantic coverage of a candidate ontology compared to a benchmark ontology can be calculated by combining the previous four semantic coverage calculations, as shown in Equation 4.16. Essentially, this equation divides total number of components (i.e. classes, instances, attributes, and relations) that are found in or can be derived from the candidate ontology by the total number of components in the benchmark ontology. In order to simplify the equation for the overall semantic coverage of an ontology, the sub-equations to compute the total number of each component found in or can be derived from the candidate ontology are defined separately, before being summed in the numerator of Equation 4.16.

$$\begin{aligned}
D_C &= \|\{c' \in C' | O \vdash c'\}\| \\
D_I &= \sum_{c' \in C'} \|\{\alpha \in I'^{c'} | O \vdash c', O \vdash \alpha\}\| \\
D_A &= \sum_{c' \in C'} \|\{\varphi \in A'^{c'} | O \vdash c', O \vdash \varphi\}\| \\
D_R &= \sum_{r' \in R'} \|\{(a,b) \in r' | O \vdash (a,b) \in r'\}\|
\end{aligned}$$

$$\mathsf{SCov}^{\Omega}(O) = \frac{D_C + D_I + D_A + D_R}{\|C'\| + \sum_{c' \in C'} \|I'^{c'}\| + \sum_{c' \in C'} \|A'^{c'}\| + \sum_{r' \in R'} \|r'\|} \tag{4.16}$$

Limitations of Completeness Metrics

Measuring the completeness of a candidate ontology with respect to a benchmark ontology ensures that weaknesses with respect to components missing from the candidate ontology

can be identified. However, these metrics are, by definition, limited to comparing the candidate ontology to the benchmark ontology. This means that a candidate ontology cannot be assessed for completeness with respect to any additional domain knowledge not in the benchmark ontology using these metrics. Therefore, these metrics are more applicable when we are evaluating ontology *learning* methodologies [79], which is when ontologies are automatically or semi-automatically generated from a text corpus.

4.4.3 Conciseness

An ontology can be considered concise if it does not contain redundant definitions. Examples of such definitions include grammatical redundancies such as explicitly defining class A as a subclass of both B and C where B is also a subclass of C, and defining two or more classes or instances with same formal definitions such that the only difference between them are their names. A concise ontology should also have minimal ontological commitment which means that it should make only the minimum number of claims required about the world being modelled to sufficiently support the intended knowledge. This is to grant any interested parties the freedom to specialise and instantiate the ontology, meaning that concepts should not be over-defined in such a way that it prevents some potential users from using the ontology [72, 76, 84]. The conciseness of an ontology can therefore be measured by counting the number of redundancies and unnecessary definitions in the ontology. There are four metrics defined in by Zhu et al. (2017) for quantifying the redundancies in an ontology: concept redundancy (CR), instance redundancy (IR), attribute redundancy (AR), and relation redundancy (RR) [81].

Concept Redundancy

The concept redundancy of an ontology (O) is represented as a percentage of the overall number of classes in the ontology. This is shown in Equation 4.17, where $\|C_R\|$ denotes the number of redundant classes and $\|C\|$ denotes the number of classes in the ontology. $\|C_R\|$ is calculated by counting the number of unnecessary classes in the ontology, where

an unnecessary class is one that shares its definition with another class.

$$\mathsf{CR}(O) = \frac{\|C_R\|}{\|C\|} \tag{4.17}$$

Instance Redundancy

The instance redundancy of an ontology is represented as a percentage of the overall number of instances in the ontology. The number of redundant instances in the ontology is calculated by summing the number of redundant instances for each class. This calculation is shown in Equation 4.18, where $\|I_R^c\|$ denotes the number of redundant instances for a class c and $\|I^c\|$ denotes the total number of instances the class c has.

$$\mathsf{IR}(O) = \frac{\sum_{c \in C} \|I_R^c\|}{\sum_{c \in C} \|I^c\|} \tag{4.18}$$

Attribute Redundancy

The attribute redundancy of an ontology is represented as a percentage of the overall number of attributes in the ontology. The number of redundant attributes in the ontology is calculated by summing the number of redundant attributes for each class. This calculation is shown in Equation 4.19, where $\|A_R^c\|$ denotes the number of redundant attributes for a class c and $\|A^c\|$ denotes the total number of attributes the class c has.

$$\mathsf{AR}(O) = \frac{\sum_{c \in C} \|A_R^c\|}{\sum_{c \in C} \|A^c\|} \tag{4.19}$$

Relation Redundancy

The relation redundancy of an ontology is represented as a percentage of the overall number of relations in the ontology. This is shown in Equation 4.20, where $\|r'\|$ denotes the number of redundant object property instances per object property and $\|r\|$ denotes the number of object property instances per object property in the ontology.

$$\mathrm{RR}(O) = \frac{\sum_{r' \in R'} \|r'\|}{\sum_{r \in R} \|r\|} \tag{4.20}$$

Using Automated Tools

Calculating the redundancy metrics above would require input from experts and so such a process would be a hybrid of the metric-based and human inspection evaluation approach. An alternative to this is a fully automated metric-based approach where we can use automated tools to analyse and quantify the structure of the ontology from which we can then draw conclusions with respect to the conciseness of the ontology. Lantow (2016), the developer of the OntoMetrics[11] online tool for validating and quantifying ontologies (see Section 4.5.2), recommends using average population and class richness to measure conciseness quantitatively [85]. Meanwhile, Vrandecic (2010) also suggests using the maximum depth of the taxonomy and class to relation ratio, which are also supported by OntoMetrics [86]. Wei et al. (2020) also uses attribute richness, inheritance richness and relationship richness to evaluate their own domain ontology for conciseness [87].

4.4.4 Expandability and Sensitivity

Gómez-Pérez (2001) defines expandability as the ability of an ontology to be extended easily by adding new definitions without altering the set of already established properties [72]. This is also referred to as extendability by Gruber (1995) [84]. Gómez-Pérez (2001)

[11] `https://ontometrics.informatik.uni-rostock.de/ontologymetrics`

also identifies another criterion to use for evaluating an ontology – sensitivity – whereby an ontology is considered too sensitive if a small change in a given definition alters the set of established properties. For the purposes of this survey, expandability and sensitivity have been combined because an ontology that is not expandable is likely to have high sensitivity. Currently, the literature does not define a metric or process for measuring the expandability or sensitivity of an ontology.

4.4.5 Clarity

Ontological clarity is when the definitions in the ontology are defined unambiguously, which means being free from *class overload*, *class redundancy* (not the same as grammatical redundancies), and *class excess*. An overload occurs when a class is used to represent more than one real-world entity, redundancy occurs when more than one class represents single real-world entity, and excess is when a class does not map to an entity in the real world [76, 83]. These issues are illustrated in Figure 4.11 and we can measure clarity by counting instances of these issues within the ontology.

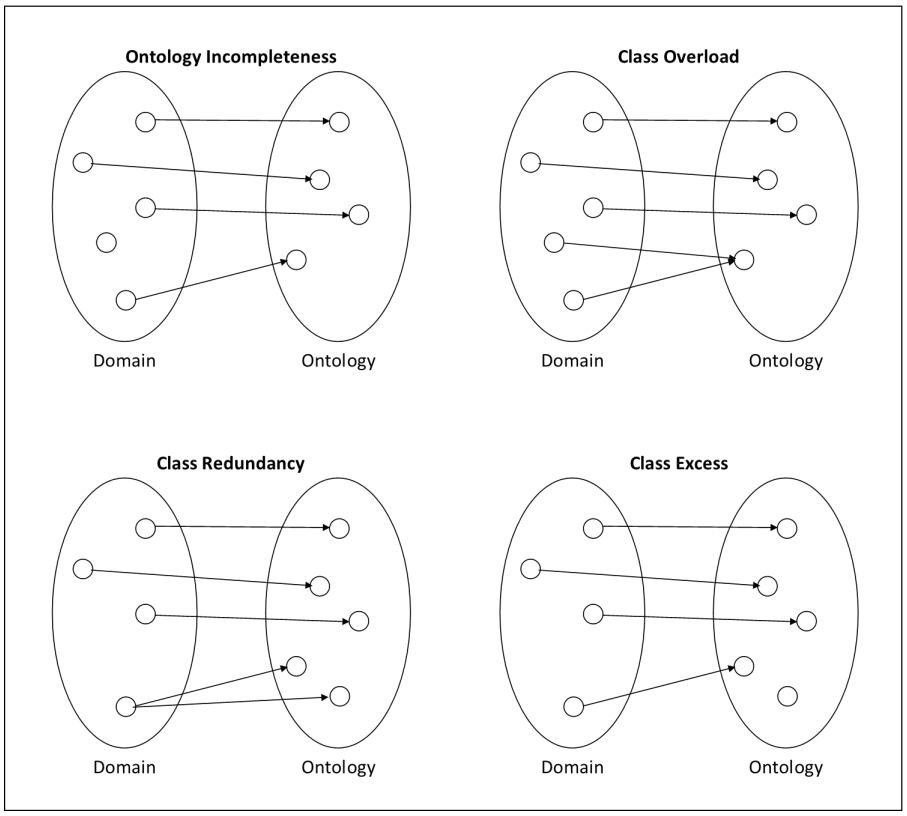


Figure 4.11: Issues arising from mapping a real-world domain to an ontology.

In addition to the formal definitions stated in logical axioms, all concepts should be documented with natural language, typically using the `rdfs:comment` annotation [84].

4.4.6 Adaptability

Ontological adaptability "measures how far the ontology anticipates its uses". Raad and Cruz (2015) suggest using the ontology in new situations and evaluate its performance depending on the task [76]. Although expandability and adaptability appear to be similar, the latter refers to how an existing ontology works in new applications while the former refers to how the existing ontology can be modified to create a new version of it.

4.4.7 Computational Efficiency

Computational efficiency "measures the ability of the used tools to work with the ontology, in particular the speed that reasoners need to fulfil the required tasks". Some types of axioms can cause efficiency problems for certain reasoners, in addition to the obvious issue of larger ontologies taking longer to process. Therefore, the efficiency of an ontology can be evaluated by measuring its performance in different tasks, specifically its speed. In addition, approaches that evaluate the ontology size can also be a means to evaluate the computational efficiency of an ontology [76].

4.4.8 Minimal Encoding Bias

Encoding bias occurs when "representation choices are made purely for the convenience of notation or implementation". This should be minimised so as to prevent an ontology being specific to a particular ontology implementation only, as "knowledge-sharing agents may be implemented in different representation systems and styles of representation". For example, consider an ontology for sharing bibliographic data: we could insist that all publication dates be represented in a specific format, such as day/month/year. Specifying a precise format for the dates reflects an implementation detail rather than the real-world knowledge, as different systems using the ontology could use different date formats. Therefore, this is an example of encoding bias [84].

4.4.9 Strategies for Measuring the Criteria

We have seen that there are a number of approaches towards ontology evaluation and several criteria to apply these approaches to. However, not all criteria are suitable for every evaluation approach. For example, Raad and Cruz (2015) presented a grid comparing the gold standard, corpus-based, task-based, and criteria-based evaluation methods based on their suitability for evaluating each criteria defined in their paper [76]. This grid states that the gold standard and corpus-based methods are the most suitable for evaluating ontology accuracy, completeness, and conciseness while task-based and criteria-based methods are

more appropriate for assessing an ontology's computational efficiency and consistency. In order to cover the maximum amount of criteria, we can evaluate an ontology by combining multiple approaches.

4.5 Ontology Evaluation Tools

In order to support ontology evaluation, a number of ontology evaluation tools have been created and subsequently referenced in the literature. Each of these tools have their own focus in terms of which criteria to evaluate an ontology against and the methods and metrics used to do this. This section will describe five ontology evaluation tools: the OntOlogy Pitfalls Scanner (OOPS!), OntoMetrics, OQuaRE, OntoCheck, and ODEval.

4.5.1 OOPS!

OOPS! was developed to "provide an automated tool to help ontology practitioners detecting common pitfalls during the ontology development" [88]. The tool was developed in 2009 with 29 pitfalls supported [88] and since then support for detecting for a further 11 types of pitfall has been added [89]. These 40 pitfalls are listed in Table 4.3 and they are quoted directly from the catalogue presented in [89]. It is important to note that OOPS! can only detect some of the pitfalls in a fully-automated way. Others can only be done in a semi-automated way and require manual confirmation. OOPS! can be downloaded and installed as a plugin[12] for Protégé or accessed via a web interface[13], which uses the Jena API[14] to scan the ontology [88].

Evaluation Criteria Measured: consistency (P05, P06, P07, P19, and P24), completeness (P04, P10, P11, P12, and P13), and conciseness (P02, P03, and P21).

[12] `https://github.com/lukasged/oops-plugin`
[13] `http://oops.linkeddata.es`
[14] `https://jena.apache.org`

Identifier	Summary
P01	Creating polysemous elements
P02	Creating synonyms as classes
P03	Creating the relationship `is` instead of using OWL primitives
P04	Creating unconnected ontology elements
P05	Defining wrong inverse relationships
P06	Including cycles in the hierarchy
P07	Merging different concepts in the same class
P08	Missing annotations
P09	Missing basic information
P10	Missing disjointness
P11	Missing domain or range in properties
P12	Missing equivalent properties
P13	Missing inverse relationships
P14	Misusing `owl:allValuesFrom`
P15	Misusing `not some` and `some not`
P16	Misusing primitive and defined classes
P17	Specialising a hierarchy exceedingly
P18	Specifying the domain or the range exceedingly
P19	Swapping intersection and union
P20	Misusing ontology annotations
P21	Using a miscellaneous class
P22	Using different naming criteria in the ontology
P23	Using ontology elements incorrectly
P24	Using recursive definition
P25	Defining a relationship inverse to itself
P26	Defining inverse relationships for a symmetric one
P27	Defining wrong equivalent relationships
P28	Defining wrong symmetric relationships
P29	Defining wrong transitive relationships
P30	Missing equivalent classes
P31	Defining wrong equivalent classes
P32	Several classes with the same label
P33	Creating a property chain with just one property
P34	Using a resource as a class without declaring it as one
P35	Using a resource as a property without declaring it as one
P36	URI contains file extension
P37	Ontology not available on the web
P38	No `owl:Ontology` declaration
P39	Ambiguous namespace
P40	Namespace hijacking

Table 4.3: Common ontology design pitfalls detected by OOPS!

4.5.2 OntoMetrics

OntoMetrics[15] is a web-based tool for evaluating ontologies. It was created by Michael Poppe and Martin Lichtwark and is based on the OWL API[16] [90]. The OntoMetrics tool currently consists of an interface through which users can upload OWL ontologies and compute a set of quality metrics for them, the option to download a report of the results in XML format, and a wiki that explains the what each metric is and how they are calculated. OntoMetrics supports the calculation of five types of metric [91]:

1. **Base Metrics**: These metrics are a count of the various types of ontology elements, such as classes, object properties, and data type properties.

2. **Schema Metrics**: These metrics quantify the design of the ontology by indicating the richness, width, depth, and inheritance of the ontology schema.

3. **Knowledgebase Metrics**: These metrics assess both the structure of an ontology and the instances inside it. Such metrics include those that describe the knowledge-base as a whole and those that describe the way each class is being used.

4. **Class Metrics**: These metrics focus on evaluating a single class at a time and as a result, require more computational resources.

5. **Graph Metrics**: These metrics calculate the structure of ontologies based on the class hierarchy only, rather than also using object properties.

While OntoMetrics is a lightweight tool for ontology evaluation, it does not provide specific feedback on the cause of ontology quality issues. We are instead given aggregated statistics so ontology developers have limited information about how to improve the ontology.

Evaluation Criteria Measured: accuracy (using the attribute richness, inheritance richness, and relationship richness metrics), conciseness (using average population and class

[15] `https://ontometrics.informatik.uni-rostock.de/ontologymetrics`
[16] `https://owlapi.sourceforge.net`

richness metrics), and computational efficiency (using the tangledness metric) [85].

4.5.3 OQuaRE

The OQuaRE framework for evaluating the quality of ontologies was developed based on the SQuaRE standard (ISO/IEC 25000:2005) for evaluating the quality of software. OQuaRE requires the definition of both a quality model and quality metrics where the model defines a set of quality characteristics and sub-characteristics and the quality metrics refer to measurements used to quantify the (sub-)characteristics [92]. This framework and an example instantiation is shown in Figure 4.12.

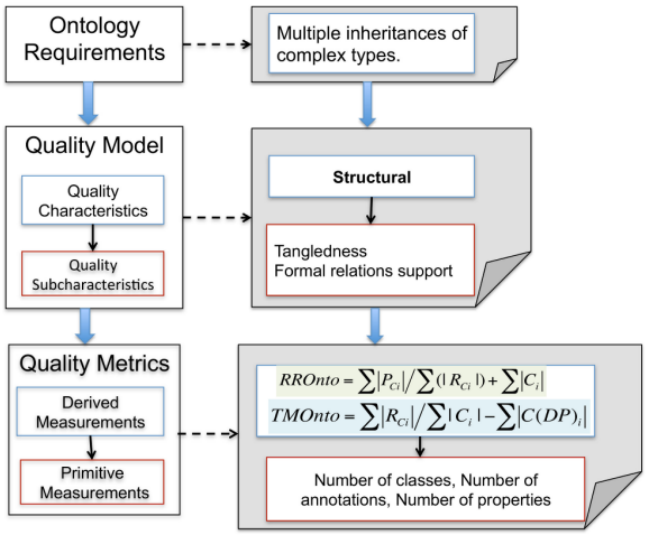


Figure 4.12: The OQuaRE ontology evaluation framework [92].

An OQuaRE quality model consists of the following seven characteristics [92, 93]:

1. **Structural**: this characteristic refers to the structure of the ontology. Examples of sub-characteristics include:
   - Cohesion: the extent that the classes are strongly related cycles.
   - Cycles: cycles are an indicator of bad design and the cause of inconsistencies.

- Consistency: ensuring that there are no contradictions in the ontology.
- Tangledness: the ratio of classes with multiple direct ancestor classes to the total number of classes in an ontology. A high ratio is an indicator of bad design.

2. **Functional Adequacy**: this characteristic refers to the extent in which the ontology fulfills a set of functional requirements. Examples of sub-characteristics include:
   - Inference: the extent that a reasoner can infer implicit knowledge from the explicit knowledge contained in the ontology.
   - Precision: the extent that the ontology provides results with the required degree of accuracy.
   - Knowledge Reuse: the extent that the knowledge in the ontology can be used to build other ontologies.
3. **Operability**: this characteristic refers to the effort required by a stated set of users to use the ontology. Examples of sub-characteristics include:
   - Learnability: the extent that the ontology supports users in learning its application, which often refers to the effectiveness of the user documentation.
   - Ease of Use: the extent that the ontology makes it easy for users to operate and control it.
   - Helpfulness: the extent that the ontology provides help to the user when they need assistance.
4. **Maintainability**: this characteristic refers to the capability of ontologies to be modified in response to changes in environments, requirements, or functional specifications. Examples of sub-characteristics include:

- Modularity: the extent that the ontology is composed of separate, connected components whereby a change to one has minimal impact on the others.
- Reusability: the extent that part of the ontology can be used in more than one ontology.
- Testability: the extent that the ontology can be validated after it has been modified.

5. **Compatibility**: this characteristic refers to the capability of two or more ontologies to perform their required functions while sharing the same hardware and/or software environment. Examples of sub-characteristics include:
   - Replaceability: the extent that the ontology can be used in the place of another ontology designed for the same purpose and within the same hardware and/or software environment.
   - Interoperability: the extent that the ontology can operate together with one or more other ontologies by combining its knowledge.
6. **Transferability**: this characteristic refers to the extent that the ontology can be transferred from one environment to another. Examples of sub-characteristics include:
   - Portability: the extent that the ontology or a part of the ontology can be transferred from one environment to another.
   - Adaptability: the extent that the ontology can be adapted for different environments, such as languages or levels of expressivity, without applying actions other than those provided for this purpose.
7. **Quality in Use**: this characteristic refers to the quality of the ontology in the context of a particular use case. Examples of sub-characteristics include:

- Usability in Use: the extent that users can achieve specific goals with accuracy and completeness within a specific context (effectiveness in use), the extent that users consume appropriate amount of resources to achieve this (efficiency in use), and the extent that users are satisfied with the ontology in a specified context of use (satisfaction in use).

- Flexibility in Use: the extent that Usability in Use meets the requirements in all the intended contexts of use and in contexts beyond those originally intended for the ontology.

Some of these sub-characteristics can then evaluated automatically based on a set of quality metrics. In these cases, the value for each metric is mapped to a normalised value between 1 to 5, where 5 represents the best possible value. In order to obtain the raw value for a sub-characteristic, the mean of all the metrics associated with that sub-characteristic is calculated. The score for each characteristic is then found by calculating the mean of the scores of its sub-characteristics and, to find a global score for the quality of the ontology, the weighted average of the overall scores of all the characteristics is computed [94].

Evaluation Criteria Measured: internal consistency (using the cycles and consistency sub-characteristics) and computational efficiency (using the tangledness and usability in use sub-characteristics) [94].

The OQuaRE Tool

The OQuaRE tool[17] is a web application that implements the OQuaRE framework. The user must provide the URL of the ontology and the tool extracts the measurements required to calculate the quality metrics. However, like OntoMetrics, it does not provide specific feedback on the cause of ontology quality issues so it is hard for developers to identify specific areas to improve the ontology.

[17] `http://miuras.inf.um.es:9080/oqmodelsliteclient`

4.5.4 OntoCheck

OntoCheck[18] is a plugin for the Protégé ontology editor developed by Schober et al. (2012) that assists with cleaning up an ontology by enforcing naming conventions and metadata completeness [95]. The plugin provides three tabs – Check, Compare, and Statistics – the first of which allows the user to choose which annotation properties to check. For example, one could choose to check that the names of all classes match (or not match) a certain regular expression. These checks can also be saved and reloaded for future testing. The Compare tab allows the user to compare the values of specific annotation properties. For example, one could compare the names of all classes with their labels (facilitated by `rdfs:label`) and the plugin will return a list of potentially problematic classes. Finally, the Statistics tab provides quantifies ontology measurements which can be used for complexity analysis and ontology evaluation. For example, one could compute the percentage of classes having exactly n subclasses and retrieve a list of potentially problematic classes [96].

Evaluation Criteria Measured: completeness and conciseness [96].

4.5.5 ODEval

ODEval[19] was developed by Corcho et al. (2004) to automatically detect circularity issues and partition errors (internal inconsistencies) and problems with redundant subclass of and instance of relations. Although a URL to this tool is provided, the link is broken so the tool can no longer be used [97].

Evaluation Criteria Measured: internal consistency and conciseness [97].

4.6 Summary

In this chapter, we presented a detailed description of a selection of ontology design methodologies and tools that can be used to construct the ontology. We also examined eight

[18] `http://www2.imbi.uni-freiburg.de/ontology/OntoCheck`
[19] `http://minsky.dia.fi.upm.es/odeval`

approaches to ontology evaluation and a range of criteria to measure the quality of an ontology, alongside four tools that can assist with automated ontology evaluation. A summary list is presented in Table 4.4. Each approach has a different use case so not all of them will be suitable for assessing the quality of the Creative Data Ontology that will be described in Chapter 6.

Design Methodologies	• Knowledge Engineering [61] • DOGMA [61, 62] • TOVE [61] • Methontology [61] • SENSUS [61] • DILIGENT [61] • On-To-Knowledge [64, 65] • UPON [66] • NeOn [67]
Design Tools	• Protégé [69, 70] • Swoop [70] • TopBraid Composer [70] • The NeOn Toolkit [71] • Apollo [70] • OntoStudio [70]
Evaluation Approaches	• Gold Standard [75, 76, 78] • Data-driven or Corpus-based [76, 78] • Criteria-based [75, 76] • Task-based [75, 76, 78] • Evolution-based [77] • Logical or Rule-based [77] • Metric or Feature-based [77] • Human Inspection [78]
Evaluation Criteria	• Accuracy [72, 76, 81] • Consistency [72, 76, 81] • Completeness [72, 76, 81, 83] • Conciseness [72, 76, 84, 85, 86, 87] • Expandability and Sensitivity [72, 84] • Clarity [76, 83, 84] • Adaptability [76] • Computational Efficiency [76] • Minimal Encoding Bias [84]
Evaluation Tools	• OOPS! [88, 89] • OntoMetrics [85, 90] • OQuaRE [92] • OntoCheck [95] • ODEval [97]

Table 4.4: A summary of ontology design and evaluation.

Part II

Contributions

Chapter 5

Methodology

This project was funded by a studentship from Royal Holloway, University of London as part of a contribution to the StoryFutures[1] research project. The StoryFutures project involved researchers from multiple disciplines including geography, psychology, media arts, and computer science collaborating to support research and development in creative industries through a number of sub-projects. These sub-projects can be categorised into four overarching themes:

- **Theme 1**: Next generation story lab
- **Theme 2**: Emergent value networks and business models
- **Theme 3**: Creative data in the collaborative workflow
- **Theme 4**: Audience engagement

This research contained in this thesis is situated in Theme 3, and given the multidisciplinary approach of the StoryFutures project, it is unsurprising that both computer science and media arts disciplines have informed my research methodology.

[1] https://www.storyfutures.com

Crotty (1998) defines research methodology as "the strategy, plan of action, process or design lying behind the choice and use of particular methods and linking the choice and use of methods to the desired outcomes" [98]. This chapter will explore this methodology in more detail, beginning with a description and justification of the research methods in Section 5.1, followed by a section containing contextual information about our industry partner companies in Section 5.2. We then explain the data collection and data analysis methods used for this study in Section 5.3 and Section 5.4, respectively. This is followed by a justification of the ontology design technologies used in this project in Section 5.5. We then continue with a discussion of ethical considerations made when conducting this study in Section 5.6 before concluding with a discussion of the impact of the COVID-19 pandemic on this project in Section 5.7.

5.1 Research Methods

Research methods is defined by Crotty (1998) as "the techniques or procedures used to gather and analyse data related to some research question or hypothesis" [98]. There are three categories of research method: quantitative, qualitative, and mixed methods, which is a hybrid of the former two. Providing a detailed discussion of these methods is not in the scope of this thesis given that they are well known in scientific literature. Instead, a brief overview is provided.

5.1.1 Quantitative Research

Creswell (2009) defines quantitative research as "a means for testing objective theories by examining the relationship among variables", where these variables typically represent numerical data obtained through polls, questionnaires, surveys, or measurements from specialist instruments. These data can be processed and analysed using statistical methods to draw conclusions that either support or reject the theory. The conclusions can then be presented using tables, graphs, and statistics [99].

5.1.2 Qualitative Research

Qualitative research is sometimes described as an example of interpretive research, which is research that has a strong reliance on observers interpreting the meaning of what they see and hear [100]. Qualitative research often requires researchers to observe and interpret human behaviours using techniques such as interviews, case studies, focus groups, and observations. These techniques are particularly beneficial in situations where the researcher needs to interact with the participants, such as to ask follow-up questions or to elicit more detailed answers. Qualitative research can also include the analysis of artifacts and documents which can be used to obtain information on or by analysing the experiences of individuals or groups [101]. The nature of qualitative research is subjective, meaning that the analysis of data performed by the researcher involves some level of interpretation [102].

5.1.3 Mixed Methods Research

A mixed methods research approach combines elements of both qualitative and quantitative research in the same study [103]. This means that researchers will work with both structured numerical data (quantitative) and unstructured narrative data (qualitative) either concurrently or sequentially. For example, they may use a survey with closed questions to obtain numerical data and then conduct a follow-up interview to elicit more in-depth answers with open-ended questions, thus forming narrative data [103]. Table 5.1 is adapted from [104] (originally published in [105]) and summarises the differences between qualitative and quantitative research methods.

Characteristics	Quantitative	Qualitative
Data Types	Numbers	Words
Research Questions	How many? How much?	How? Why?
Data Collection Methods	Surveys	Interviews, observations
Sampling Type	Statistical	Snowball or quota
Sample Size	Large is preferred	Small is preferred[2]
Goal	Prove, verify	discover, explore
Generalisability	Is a goal	Is not a goal

Table 5.1: Comparison of qualitative versus quantitative research

5.1.4 Choice of Research Method

According to Kothari (2004), the term 'research methodology' considers not only the research methods (e.g. interviews or analysis of documents) but also "the logic behind the methods we use in the context of our research study and explain why we are using a particular method or technique and why we are not using others so that research results are capable of being evaluated either by the researcher himself or by others" [102]. In this section we justify our selection of the mixed methods approach to conduct our study. While the selection of research methodology is made "perhaps unconsciously or through default" [106], the choice is often influenced by factors such as similar research studies conducted previously, resources, and research objectives.

The aim of this thesis has been to understand how semantic web technologies can be used in the moving image industry to optimise their workflows by ensuring the availability of metadata throughout all the stages in the life-cycle of a moving image project. Given that the aim involves the development and use of technologies in industry settings, there is a necessity to take a mixed methods approach to data collection. In doing this, qualitative data will be collected through ontology definition and evaluation interviews and quantitative

[2]This is due to resource constraints.

data will be collected via other methods, such as through a count of metadata fields provided by the partners compared to those in the ontology.

5.2 Industry Partner Companies

Before defining our data collection and analysis methods, we must first provide contextual information about our industry partner companies that supported this research by providing vital domain expertise and extensive feedback about the iterations of the Creative Data Ontology. Our three partner companies were Evolutions, Double Negative (DNEG), and Pinewood Studios.

5.2.1 Evolutions

Evolutions[3] is a post-production house founded in 1994, based in London and Bristol, United Kingdom. Evolutions provides end-to-end post-production services, specialising in unscripted television productions, but also have some experience in scripted animation. They have worked on popular titles such as 24 Hours in Police Custody, Tipping Point, Taskmaster, and The Only Way Is Essex, and the productions they have worked on have been broadcast by the BBC, Channel 4, and Channel 5 [107].

Due to the impact of COVID-19 lockdown restrictions on commissioning and production of televisions shows in 2020, several projects were delayed which impacted on Evolutions' revenue in the first half of 2021. However, after restrictions were lifted, demand for post-production services increased and to cater for this, Evolutions opened a new building in November 2021 on the ground floor of their Sheraton Street premises [108].

[3] `https://evolutions.tv`

5.2.2 Double Negative (DNEG)

Double Negative[4] (DNEG) is a post-production house founded in 1998, based in London, United Kingdom. DNEG specialises in visual effects, animation, and virtual production and have worked on popular films such as Fantastic Beasts and Where To Find Them (2016), Aladdin (2019), Harry Potter and the Philosopher's Stone 3D (2020), and Ghostbusters: Afterlife (2021) [109]. DNEG have also won seven Academy Awards for 'Best VFX' in recent years [110].

Like Evolutions, DNEG was also impacted by the COVID-19 pandemic. As productions were disrupted and release dates delayed, the volume of work available to DNEG was reduced, which led to a decline in revenue in the year ending 31st March 2021. This led to staff being furloughed and staff headcount reductions [111]. However, DNEG are recovering from the effects of the pandemic, with the number of staff increasing from 650 in March 2021 to 750 in March 2022 [111, 112].

5.2.3 Pinewood Studios

Pinewood Studios[5] was founded in 1945 and is owned by parent company Pinewood Group Limited. Pinewood Studios provides production facilities such as stages and accommodation such as wardrobe storage, dressing rooms, and make-up and hair rooms [113]. They also provide post-production services including audio post, localisation, and preview screenings [114]. Pinewood Studios have worked on popular films such as Les Misérables (2012), Bridget Jones's Baby (2016), and Johnny English Strikes Again (2018) and television shows such as Tenable, The IT Crowd, and Dragons' Den [115].

Pinewood Studios was also negatively affected by the COVID-19 pandemic. Due to the production hiatus in Spring 2021, there was reduced activity in the TV studios, which was the main contributor to the £11 million decrease in turnover for the year ended 31st

[4] https://www.dneg.com
[5] https://pinewoodgroup.com/studios/pinewood-studios

March 2021 [116]. However, as the disruption to the TV studios caused by the pandemic decreased, revenue increased [117].

5.3 Data Collection Methods

Hox and Boeije (2005) define primary data as "original data collected for a specific research goal", such as the responses to surveys and interview transcripts. Secondary data refers to "data originally collected for a different purpose and reused for another research problem", such as literature reviews and data archives [118].

5.3.1 Literature Review

Conducting a review of existing literature is a crucial part of a research project and is usually one of the first stages, as it supports the identification of a research topic and, more specifically, a gap in the research for that topic. For the purposes of this study, the following topics were selected for review:

- The general concept of ontologies, including design and query languages, and ontologies for knowledge management.
- Existing ontologies for the media domain and existing ontology-based knowledge management systems for any domain.
- Methodologies for ontology engineering and evaluation, including approaches to and criteria for ontology evaluation.

5.3.2 Phase 1: Database Dumps

Following a comprehensive literature review of the existing ontologies for the media domain, we then required details of the metadata that our own ontology would need to manage. As part of the StoryFutures project, a partnership had been established between post-production companies based in London, United Kingdom. These companies agreed to

provide details of the metadata that they store by supplying us with exports of their database schemata. These schemata were given to us in spreadsheet format, although the structure for each company's export varied slightly. Therefore, once the spreadsheets of database schemata had been acquired, we needed to restructure them to ensure consistency. This restructured spreadsheet served as a corpus of domain knowledge from which we could extract terms that would form classes and properties within the Creative Data Ontology. In particular, most metadata fields would form data type properties.

5.3.3 Phase 2: Verbal Feedback on Ontology Iterations

Following the acquisition of database schemata from our industry partners, an initial iteration of the Creative Data Ontology was designed. Phase 2 concerns itself with obtaining feedback from our partners on this and future iterations of the ontology. Through regular meetings with representatives from our partner companies, we obtained verbal feedback on areas of the ontology that did not accurately reflect the domain in the real world. These changes were made and feedback was sought in an incremental and iterative way.

5.3.4 Phase 3: Feedback from Interviews with Experts

After representatives from our industry partners approved the proposed ontology, further feedback was sought from other domain (and ontology) experts during Phase 3. Two slide decks were designed for the purposes of facilitating interviews with both domain experts, such as practitioners working in the media industry, and ontology experts, such as researchers and academics. Both slide decks can be found in Appendix B. Twenty-one domain and ontology experts were invited to participate in these semi-structured interviews and seven accepted: four domain experts and three ontology experts. There were two reasons for interviews being chosen over surveys to obtain feedback from experts. Firstly, given the nature of the questions, a survey would be inappropriate because it would require the participants to write several long answers, which would likely be off-putting due to the time and effort required and would therefore result in fewer experts being willing to partic-

ipate. Secondly, it was likely that domain experts would need clarification on the technical aspects of an ontology while ontology experts would need further information about the use cases. Conducting an interview would allow experts to obtain these clarifications and, as a result, give feedback that is more relevant and more accurate. Semi-structured interviews were chosen over structured interviews because the former allows the interviewer to ask follow-up questions based on the interviewee's initial response to a question, either to obtain clarification or to elicit further details.

Information Collected

The semi-structured interviews for domain experts focused on the accuracy, completeness, conciseness, and clarity of the ontology while the interviews for ontology experts only focused on conciseness and clarity. This is because accuracy and completeness would require the interviewee to have domain expertise in order to be able to assess an ontology against these criteria, which an ontology expert is unlikely to have.

Interview Protocol

The length of the interviews ranged from 25 to 75 minutes and were all conducted over Microsoft Teams or Zoom between November 2021 and July 2022. The decision to perform the interview remotely was firstly because a number of participants were based in different geographical regions. The second reason was to ensure the safety of participants during the COVID-19 pandemic. The interviews were recorded after obtaining consent via a consent form, which was completed and returned by the interviewee prior to the interview starting. The interviewer also took notes during the interviews.

Prior to the interview, some domain experts were also asked to watch a brief video explaining the basics of ontologies. This task was introduced after the first two domain experts interviewed experienced difficulties with understanding the mechanics behind an ontology. The video was prepared by the interviewer and consisted of a slideshow presentation with

a voice-over. This video can be accessed via the following links:

- `https://zenodo.org/badge/latestdoi/662246329`
- `https://github.com/cadexiades/PhD-Deliverables`

5.4 Data Analysis Methods

The data analysis process began with restructuring the exports of our partners' database schemata into a shared structure. This allowed key terms to be identified which later formed the classes in an initial ontology design (Phase 1). The ontology was then formalised using OWL and shown to representatives from our partner companies to obtain feedback to use to improve the ontology in a feedback loop (Phase 2). Once the partners agreed on a final version of the ontology, interviews with domain experts and ontology experts were conducted in order to evaluate the ontology (Phase 3). This section presents the data analysis process in detail.

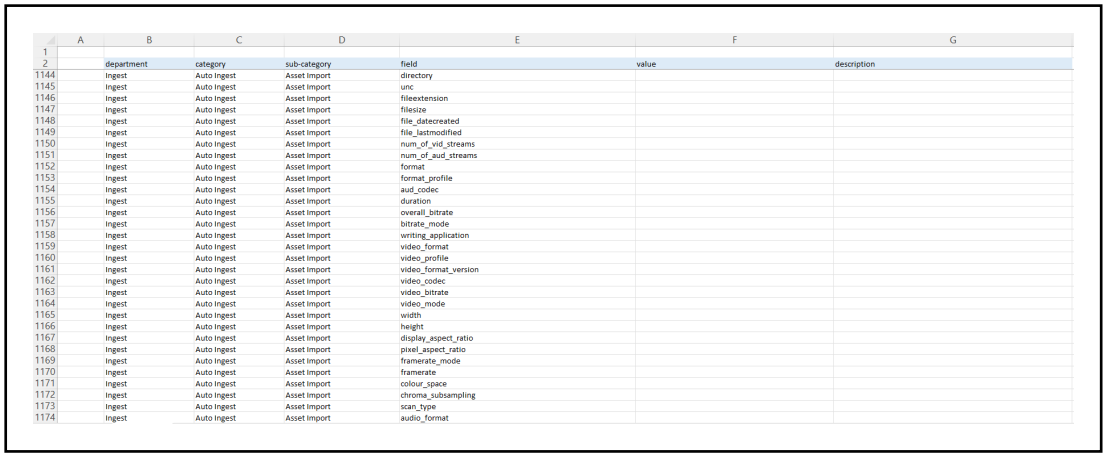
5.4.1 Phase 1: Database Dumps

Once we received the subset of fields which were exported directly from the partners' database systems, they needed to be analysed in order to build an initial class hierarchy. We first reorganised the list of fields from each partner based on the three-level categorisation system described below, before merging them into a larger spreadsheet. The categories were created based on the structure of the metadata fields dump received from the partners:

- **Evolutions** provided a table in a spreadsheet which included columns for the *Department*, *Category*, *Subcategory*, and *Field Name*.
- **Double Negative (DNEG)** provided a spreadsheet with a number of tabs. Each tab contained a table of metadata fields with a column for the entity type and a further column for the field name.

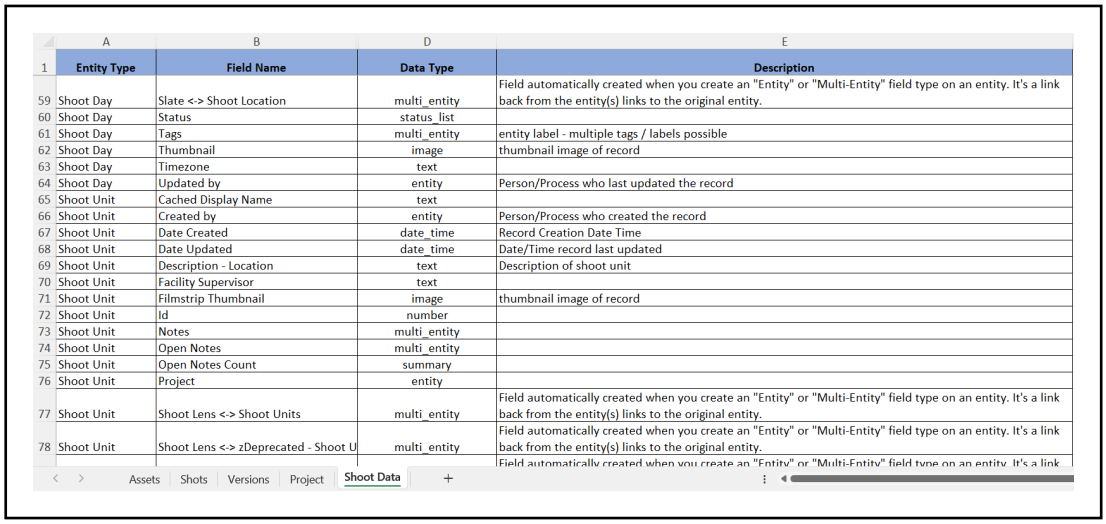
- **Pinewood Studios** provided a spreadsheet with a table for each category of metadata and a column for the field name.

Figures 5.1, 5.2, and 5.3 show a sample of the database dumps provided by Evolutions, DNEG, and Pinewood, respectively.



	A	B	C	D	E	F	G
1							
2		department	category	sub-category	field	value	description
1144		Ingest	Auto Ingest	Asset Import	directory		
1145		Ingest	Auto Ingest	Asset Import	unc		
1146		Ingest	Auto Ingest	Asset Import	fileextension		
1147		Ingest	Auto Ingest	Asset Import	filesize		
1148		Ingest	Auto Ingest	Asset Import	file_datecreated		
1149		Ingest	Auto Ingest	Asset Import	file_lastmodified		
1150		Ingest	Auto Ingest	Asset Import	num_of_vid_streams		
1151		Ingest	Auto Ingest	Asset Import	num_of_aud_streams		
1152		Ingest	Auto Ingest	Asset Import	format		
1153		Ingest	Auto Ingest	Asset Import	format_profile		
1154		Ingest	Auto Ingest	Asset Import	aud_codec		
1155		Ingest	Auto Ingest	Asset Import	duration		
1156		Ingest	Auto Ingest	Asset Import	overall_bitrate		
1157		Ingest	Auto Ingest	Asset Import	bitrate_mode		
1158		Ingest	Auto Ingest	Asset Import	writing_application		
1159		Ingest	Auto Ingest	Asset Import	video_format		
1160		Ingest	Auto Ingest	Asset Import	video_profile		
1161		Ingest	Auto Ingest	Asset Import	video_format_version		
1162		Ingest	Auto Ingest	Asset Import	video_codec		
1163		Ingest	Auto Ingest	Asset Import	video_bitrate		
1164		Ingest	Auto Ingest	Asset Import	video_mode		
1165		Ingest	Auto Ingest	Asset Import	width		
1166		Ingest	Auto Ingest	Asset Import	height		
1167		Ingest	Auto Ingest	Asset Import	display_aspect_ratio		
1168		Ingest	Auto Ingest	Asset Import	pixel_aspect_ratio		
1169		Ingest	Auto Ingest	Asset Import	framerate_mode		
1170		Ingest	Auto Ingest	Asset Import	framerate		
1171		Ingest	Auto Ingest	Asset Import	colour_space		
1172		Ingest	Auto Ingest	Asset Import	chroma_subsampling		
1173		Ingest	Auto Ingest	Asset Import	scan_type		
1174		Ingest	Auto Ingest	Asset Import	audio_format		

Figure 5.1: Sample of the database dump provided by Evolutions.



	A	B	D	E
1	**Entity Type**	**Field Name**	**Data Type**	**Description**
59	Shoot Day	Slate <-> Shoot Location	multi_entity	Field automatically created when you create an "Entity" or "Multi-Entity" field type on an entity. It's a link back from the entity(s) links to the original entity.
60	Shoot Day	Status	status_list	
61	Shoot Day	Tags	multi_entity	entity label - multiple tags / labels possible
62	Shoot Day	Thumbnail	image	thumbnail image of record
63	Shoot Day	Timezone	text	
64	Shoot Day	Updated by	entity	Person/Process who last updated the record
65	Shoot Unit	Cached Display Name	text	
66	Shoot Unit	Created by	entity	Person/Process who created the record
67	Shoot Unit	Date Created	date_time	Record Creation Date Time
68	Shoot Unit	Date Updated	date_time	Date/Time record last updated
69	Shoot Unit	Description - Location	text	Description of shoot unit
70	Shoot Unit	Facility Supervisor	text	
71	Shoot Unit	Filmstrip Thumbnail	image	thumbnail image of record
72	Shoot Unit	Id	number	
73	Shoot Unit	Notes	multi_entity	
74	Shoot Unit	Open Notes	multi_entity	
75	Shoot Unit	Open Notes Count	summary	
76	Shoot Unit	Project	entity	
77	Shoot Unit	Shoot Lens <-> Shoot Units	multi_entity	Field automatically created when you create an "Entity" or "Multi-Entity" field type on an entity. It's a link back from the entity(s) links to the original entity.
78	Shoot Unit	Shoot Lens <-> zDeprecated - Shoot U	multi_entity	Field automatically created when you create an "Entity" or "Multi-Entity" field type on an entity. It's a link back from the entity(s) links to the original entity.

Assets | Shots | Versions | Project | Shoot Data | +

Figure 5.2: Sample of the database dump provided by Double Negative.

Data Field Type	Example/Descritption
METADATA BY TITLE	
Client Company	Disney
Production	Endgame Inc.
Title	Avengers Engame
Genre	Live Action
Client Name	Jane Disneyperson
Number of Reels	9
Framerate	24
Length	02:46:22
Audio Layout	7.1
Subtitle Languages	English, French, Italian
Picture file format	Pro Res 4444
Aspect Ratio	1.85:1
Picture resolution	4K
Release Date	01/05/2019
METADATA BY REEL	
Reel Number	2
LFOA	02:18:12:06 (Last Frame Of Action)
Audio Version Identifier	v3.2 or 02-03-2019a
Picture Version Identifier	v6.3 or 17-03-2019b
Sound Type	M&E (DIALOGUE, M&E, MUSIC, EFFECTS etc.)
Audio Language	Italian
Dubbing Studio	Studio Italia
Dubbing Mixer	Mixer Dave

Figure 5.3: Sample of the database dump provided by Pinewood Studios.

Given the structure of each partners' spreadsheet, we restructured them into the following three levels:

- **Level 1**: these categories would form the top-level classes in the early versions of the Creative Data Ontology. Every metadata field belonged to a Level 1 category, and these categories were obtained from the *Department* column of Evolutions' spreadsheet and the tab names of DNEG's spreadsheet. In the case of Pinewood Studios' fields, we were informed by them that these fields all relate to the same department so that department formed the Level 1 category for all of their fields.

- **Level 2**: these subcategories would form the second-level classes in the early stages of designing the ontology. These categories were obtained by concatenating the *Category* and *Subcategory* columns of Evolutions' spreadsheet, the *Entity Type* column of DNEG's spreadsheet and the table categories of Pinewood Studios' spreadsheet. In the case of Evolutions, the concatenation is necessary because their spreadsheet was structured to naturally have four levels whereas DNEG's and Pinewood Studios'

spreadsheets had three. Therefore, we chose to merge the two columns in Evolutions' spreadsheet to simplify the creation of an initial ontology, with a consideration for the fact that they may need to be separated again in a future iteration if the partners requested this.

- **Level 3**: this level consists of the field names taken from the relevant column within each spreadsheet. Unlike Levels 1 and 2 which would form the classes in the ontology, the field names would form the data type properties belonging to those classes and, in some instances, object properties connecting two classes.

This merged spreadsheet could then be used to create a base ontology, which we could elicit feedback on from the partners and improve.

5.4.2 Phase 2: Verbal Feedback on Ontology Iterations

The ontology was built incrementally from December 2019 to March 2021. During this time, five iterations were produced and presented to representatives from our partner companies. These representatives would then provide verbal feedback on matters such as whether there were any concepts, relationships, or metadata fields missing from the ontology, whether any real-world concepts or relationships have been represented inaccurately, and whether any parts of the ontology were unclear. The iterations are described further in Section 6.2.

5.4.3 Phase 3: Feedback from Interviews with Experts

After each evaluation interview was conducted, the recording of the interview was automatically transcribed using Microsoft Word on Office 365. This auto-generated transcript was then manually edited to ensure that the transcription accurately reflected the contents of the recording. Following this, quotes from the interview containing feedback were copied from the transcript into a document with the following categorisation system:

- **Use Cases 1 to 4**: Each use case was assigned its own category to group feedback

into and within those, the feedback was subdivided into groups for each of the four ontology evaluation criteria (accuracy, completeness, conciseness, and clarity), as well as general feedback.

- **Definitions of Components**: This category focuses on feedback about the natural language definitions of components within the ontology. Due to the size of the ontology, it would have been very time-consuming for interview participants to check the definition of every class and property so feedback was sought for only components used to represent the four use cases. This feedback was then subdivided into groups for feedback about:

  1. The clarity and accuracy of the definitions
  2. The existence of overlapping definitions
  3. Whether a domain expert would easily understand the ontology
  4. How the clarity of the ontology could be improved

- **Classes in the Ontology**: This category focuses on feedback about the classes in the ontology. This feedback was subdivided into groups for feedback about:

  1. The correctness of the class hierarchy
  2. Whether any of the classes were irrelevant or unnecessary
  3. Whether there were any important classes missing
  4. Whether the chaining of a set of classes[6] was correct

  The chaining of classes refers to a set of classes where each class in the set is connected to every other class in the set, although not necessarily directly via a single object property. Consider the set of classes `A`, `B`, `C`, and `D` which are chained

[6] `Shoot` ↔ `ShootDay` ↔ `Scene` ↔ `Shot` ↔ `Take` ↔ {`Clip` | `Slate`}

as A ↔ B ↔ C ↔ D. As an example, A is connected to C indirectly: A is connected to B which is then connected to C.

- **General Feedback on the Ontology**: This category focuses on the feedback given to questions about the ontology as a whole, including those that do not fit under the previous categories. General feedback was subdivided into groups based on the question they were in response to:

  1. What is your overall opinion of the ontology?
  2. What difference could this ontology make to your business?
  3. How could the ontology be improved?
  4. How could the ontology be extended?

By following the above system, feedback could be grouped according to the theme, making it easier to identify areas were feedback was frequently given and therefore are more likely to require the most improvement.

5.5 Tools for Ontology Design

After receiving and analysing the exports of our partners' database schemata, the ontology had to be defined in a machine readable form before moving to the next phase. The ontology was first defined using the Protégé ontology editing tool, which could then be visualised using the Cameo Concept Modeler[7] plugin for MagicDraw[8]. This methodology allowed us to use diagrammatic outputs to gather feedback from the industry partners at different stages of the ethnographic research and during collaborative meetings where the requirements of a metadata management tool were specified. In this way, the veracity of the modelling was authenticated by domain experts and the capturing of domain-specific

[7] https://www.nomagic.com/product-addons/magicdraw-addons/cameo-concept-modeler-plugin
[8] https://www.nomagic.com/products/magicdraw

information regarding several specialised areas of post-production was assured.

5.5.1 Choice of Ontology Technologies

As explained in Section 2.3, OWL is the industry standard family of ontology authoring languages, consisting of three sublanguages – OWL Full, OWL DL, and OWL Lite – each with different levels of efficiency and expressiveness. Additionally, Section 4.2 reviewed a selection of ontology editors, which were later considered for use in building the Creative Data Ontology. Given that Apollo and OntoStudio were unavailable for download and Top-Braid Composer is a commercial product, the three most promising options were Protégé, Swoop, the NeOn Toolkit.

For the purposes of building the Creative Data Ontology, we opted to use the OWL Full language in order to benefit from all possible OWL functionality. This decision was made because at the start of the building phase, we could not know for certain which OWL constructs we would need to accurately represent the domain. OWL was chosen over the KIF language because OWL is supported by Protégé, and both OWL and Protégé are industry standard technologies for building and editing ontologies [68]. This meant that both have a vast amount of more readily available documentation and a wider online support community to assist ontology developers, which is lacking for the KIF language and for other ontology editors.

5.5.2 Choice of MagicDraw for Visualisations

In order to present the iterations of the Creative Data Ontology to our domain experts in a way that would elicit useful feedback, we needed to visualise the ontology in the form of diagrams. Although there are a number of ontology visualisation tools described in the literature (see [119]), many of these were no longer available online or had limitations that made them unsuitable for our use. We needed a tool that would allow us to import an ontology file (in OWL/XML format) and generate diagrams automatically. These diagrams

needed to be editable and the tool should also allow us to generate diagrams of segments of the ontology.

We initially considered using OntoGraf[9], which is a plugin that comes pre-installed in Protégé, and WebVOWL[10], which is a web-based visualisation tool. We considered these tools both because of their availability and because of the number of features they have. According to Dudáš et al. (2018), OntoGraf and WebVOWL have implemented the most interaction techniques out of all the publicly available Protégé plugins and web-based tools respectively [119]. However, we found that OntoGraf was not appropriate because the diagrams would not show the data type properties, which are crucial for this research as the majority of metadata fields are represented using data type properties. We also found WebVOWL was not suitable because the ontology (in OWL format) would need to be converted into a JSON format before uploading to WebVOWL, which was not practical. Although WebVOWL has a Protégé plugin version, ProtégéVOWL, which would allow us to generate visualisations of the ontology without the need to manually convert it first, the resulting visualisations could not be edited to show only segments of the ontology, so ProtégéVOWL did not fulfil our needs.

Instead, we later found that the Cameo Concept Modeller plugin for MagicDraw was the most suitable because it was able to read an ontology file directly from OWL/XML format and generate editable diagrams automatically. These diagrams could also be generated with different layouts and could be exported as both vector and bitmap images for use in presentation slides, publications, and other mediums.

5.6 Research Ethics

Carlo Ghezzi (2020) defines ethics as "a branch of philosophy that studies human behavior and provides rules and guidelines for individuals and for groups distinguishing between

[9] https://protegewiki.stanford.edu/wiki/OntoGraf
[10] http://vowl.visualdataweb.org/webvowl.html

wrong and right conduct, according to an ideal behavioral model" [120]. Researchers are not an exception – they are also expected to behave ethically when conducting activities relating to their work. Ethics in research encompasses a number of different issues, which will be explained in this section including how these considerations have been applied to the research contained in this thesis.

5.6.1 Academic Integrity

The most obvious ethical issue involved in research is academic dishonesty, of which there are three main types: fabrication, falsification, and plagiarism. Fabrication refers to the creation of fake results and recording them as if they were real while falsification refers to the manipulation of research materials, equipment, processes, data, or results without justification, such that the research is misrepresented. Meanwhile, plagiarism is the use of other people's work without giving proper credit. All three are significant issues in the academic community and are examples of research misconduct, so researchers must ensure their work is free from fabrication, falsification, and plagiarism during the proposal, performance, and review of the research and when reporting results [120]. In order to ensure academic integrity, all results presented in this thesis are genuine and free from manipulation while all non-original material used in this thesis will be referenced and credit given to the author.

5.6.2 Data Protection and Confidentiality

Researchers are often given access to confidential information, including information provided by organisations that are protected under non-disclosure agreements and information about individual research subjects such as audio recordings of interviews. The collection of such information comes with not only an ethical responsibility to keep that information confidential, but also a legal one under the General Data Protection Regulation 2018 (GDPR). The GDPR regulates data protection and privacy for all individuals within the European Union and researchers operating within Europe are legally required to comply with GDPR

when performing research. This includes not intentionally disclosing it to unauthorised parties and protecting the information from access by unauthorised parties [120].

In order to maintain the confidentiality of our interview participants, the consent form provided an option to be quoted either anonymously within publications or with their position and their organisation's specialism within the media industry attached to quotations. However, the names of subjects and the names of their organisations were not published. Additionally, to comply with GDPR, the subject's interview files, full transcripts and recordings were only disclosed to the researchers involved in this study.

5.6.3 Informed Consent

Informed consent is an important principle in research ethics which states that "human participants can enter research freely (voluntarily) with full information about what it means for them to take part, and that they give consent before they enter the research" [121]. As part of this research, an ontology was developed which later needed to be evaluated by media industry professionals to ensure that it was of high quality. Informed consent for participation in interviews was obtained through an email dialogue in which the subjects were furnished with the following information:

- a background to the research and the type of assistance being requested from the subject.
- an explanation of the reasons the subject has been approached to take part in this study.
- an explanation of what the interview will involve, the duration, and that it will be recorded for audio.
- an understanding that others, including those within the media industry, may read the subject's views in any publications resulting from the research and in this thesis.

- a commitment to protect the confidentiality of the subject's interview files, transcripts and recordings, and a copy of the university's GDPR notice is provided.

Moreover, before any potential participants were contacted, approval was obtained from the Ethics Review Board at Royal Holloway, University of London, to ensure that the documentation provided to potential participants and the proposed data collection methods were ethically sound.

5.7 Impact of the COVID-19 Pandemic

The COVID-19 virus began circulating in the United Kingdom in early 2020, resulting in a series of country-wide and localised lockdowns and other restrictions to prevent its spread. Such restriction remained in place in some form until February 2022, at which time any remaining restrictions were lifted. As a result, this project was impacted in two ways:

1. **Industry Partner Engagement**: Before the first lockdown was introduced on 23 March 2020, there were regular lines of communication with our industry partners via email, video calls, and in-person meetings with the entire StoryFutures Theme 3 team. However, when the lockdown began in the United Kingdom, we could no longer meet in person. The restrictions also caused a large proportion of staff at our partner companies to be furloughed and those who remained had significantly higher workloads, meaning they had to prioritise their time more rigorously. This resulted in contact with partner companies becoming more infrequent. To mitigate the impact of the loss of in-person meetings, we instead opted to schedule a weekly 1-hour call every Friday morning from June to September 2020. These weekly calls ensured that work on this project continued to progress.

2. **Availability of Interview Participants**: The ontology evaluation stage of this project began in November 2021, which was just after the government had relaxed restrictions put in place following the third wave of the pandemic driven by the Delta variant of the

virus. Although restrictions were relaxing, companies within the the media industry, including our own partner companies, were still recovering from the financial impact of the pandemic. This meant that the availability of domain experts to participate in ontology evaluation interviews was extremely limited. As a result, the pool of evaluation participants was expanded to include ontology experts. While ontology experts could not comment on whether the ontology represents the real-world moving image domain correctly, they can provide feedback on the quality of the ontology from a technical perspective.

Overall, the impact of the COVID-19 pandemic on this project has been centred around the availability of experts in the media industry. This project has been reliant on such experts having the availability to share their knowledge and experiences with us so it was vital that the impact of the pandemic was mitigated as far as possible.

5.8 Summary

In this chapter we presented the quantitative, qualitative, and mixed methods research approaches and explained the reasons that mixed methods is the most appropriate research approach to use in this thesis. We then provided contextual information about our three industry partner companies. Next we described the three phases of data collection before explaining how the data collected in each of these phases will be analysed. We also discussed choices of tools for ontology design before proceeding to consider the ethical issues that are pertinent for this research project. Penultimately, we explained the various limitations and delimitations of this research and how we sought to minimise their impact before concluding with a discussion of the impact of the COVID-19 pandemic on this thesis. Over the next three chapters we will present the design of the Creative Data Ontology, the evaluation process applied to this ontology, and a discussion of the results.

Chapter 6

Designing the Creative Data Ontology

Note: *this chapter uses material previously published in and adapted from [48].*

The previous chapters highlight issues of inefficiency and availability of metadata during the many stages of media creation. While research towards developing a semantic model has already been carried out, it has been dominated by a focus on film creation, with little attention given to semantic models for visual effects and scripted and unscripted television creation. A suggested solution is the Creative Data Ontology, which has been constructed based on an extensive set of metadata fields provided by three post-production companies that they deem to be common across the industry. This chapter will first introduce the Creative Data Ontology in Section 6.1 before presenting the five iterations of the ontology in Section 6.2. We will then describe the four use cases that the ontology should support in Section 6.3.

6.1 An Introduction to the Creative Data Ontology

The Creative Data Ontology was formed following extensive discussions with three post-production companies over a period of time to identify its concrete scope. During this, members of the Creative Data team interviewed practitioners from our partners in the post-production industry about their workflows and, more specifically, the issues they face with regards to metadata. After being presented with a long-list of issues, the partners agreed on three main 'pain points' which result in metadata either being unavailable or not used efficiently:

1. *A lack of notes from set and a lack of accuracy within notes*. It is often the case that notes from set get lost or are simply not passed on to those working in post-production and, when they are passed on, they can be inaccurate. One practitioner working in VFX said in reference to the availability of colour information that "it's the same with colour, sometimes it's in there, but it's not necessarily been put in in the right way, so it's hard".

2. *Good practice gets broken up by the pressures of production*. The production process is a busy and intense one. It can be impacted by delays with a knock-on effect on other stages in the process, new staff could be assigned to work on the project, and existing staff could leave. As a result, the quality and available of metadata is often impacted. For example, one practitioner said that "sometimes good practices get broken up by the pressures of production – e.g. when someone moves to a location – or when power is not available on a site; things get left undone".

3. *Misunderstandings and misreadings*. This issue can present itself in a number of forms including misunderstanding other practitioners through verbal communication, misreading emails, or by misunderstanding the metadata that a tool requires. One practitioner clearly exemplified this pain point as follows: "emails can be misread; misunderstandings about a piece of required camera information; if via time-differences

this can make for some considerable delays".

Other pain points that appeared in the long-list of issues include:

- Massive amount of heterogeneous data
- Rigid productions
- Productions lacking in consistency
- Idiosyncratic renaming of files
- Quality issues with the data (e.g. camera wobbling)
- Removal of human-readable reference points in data
- The need to loop back and correct things
- Pre-filtering of data before it reaches post-production

The Creative Data Ontology aims to rationalise the various concepts and vocabulary currently employed in the field, many of which have overlapping meanings and can cause misunderstandings across disciplinary boundaries. The ontology supports a common vocabulary by providing a set of moving image creation concepts using a sample set of metadata fields identified by our partners as essential for supporting the use cases given in Section 6.3. The original set of fields numbered several thousand, reflecting the complexity of the many workflows, but for the purposes of creating an initial ontology the partners identified a smaller subset of approximately 350 fields (although some of these were later marked as deprecated).

6.2 Ontology Iterations

The Creative Data Ontology underwent five iterations, each one incorporating the feedback received from our creative industry partners and the software development company that developed the metadata management tool that uses the ontology.

6.2.1 Iteration 1

The first iteration of the Creative Data Ontology was constructed using approximately 350 metadata fields identified between the three industry partners as a 'core' part of the workflows within their sub-sectors of post-production. This iteration began by incorporating these fields into a basic ontology in order to group them into concepts (or classes). In parallel with this step, a practitioner from one of our industry partners and we from the Creative Data team attended a training workshop led by an ontology specialist. During this workshop, we collaboratively drew a diagram of an ontology with workflows, rather than the metadata fields, in mind. A photograph of this diagram is shown in Figure 6.1.

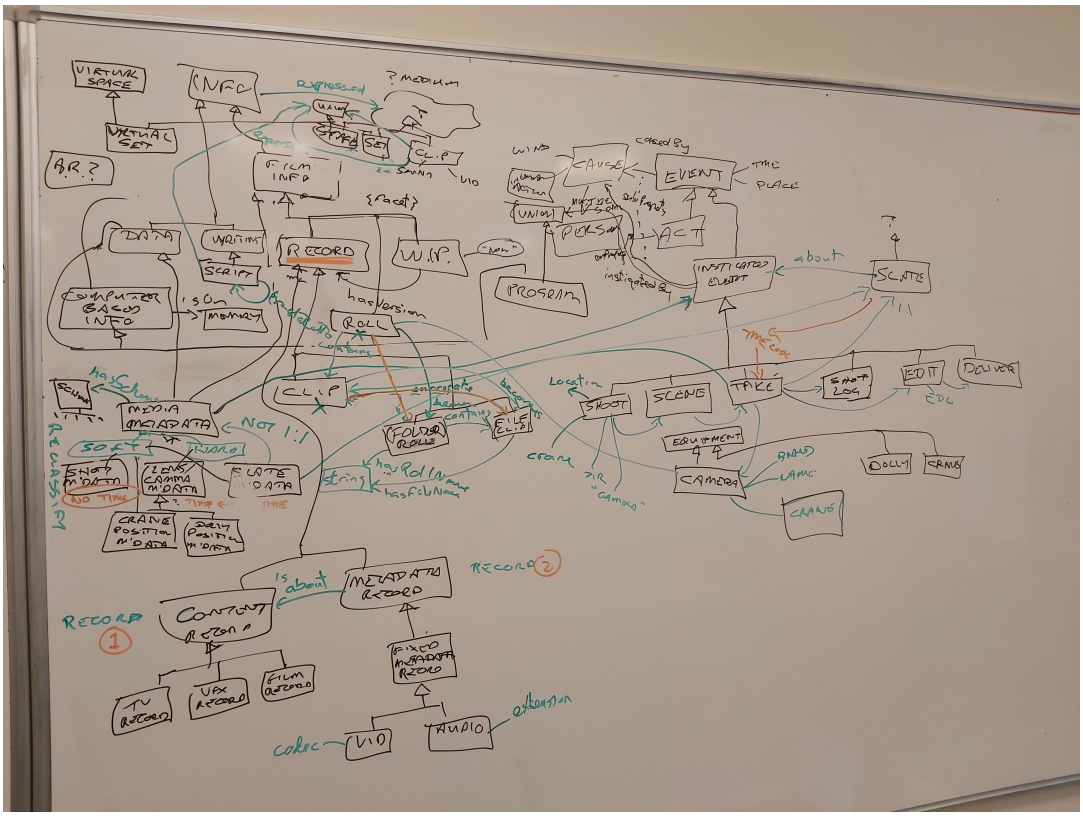


Figure 6.1: Ontology design sketch produced at the training workshop.

After this workshop, the design decisions illustrated in the diagram were incorporated into the existing ontology, which was then reviewed by the ontology expert. The ontology expert pointed out that the attributes currently belonging to the `Person` class should be split into those attributes about a person and those that are specific to their role.

> "On 'Person' you have a bunch of things that are really a person performing some role. I would be put these under a 'Relative Thing' partition at least in the conceptual ontology [...] The test is, is there data specific to the person in the role (data about a person as a Director) that is distinct from data intrinsic to the person?"

In response to this feedback, a new class – `Role` – was created, with an object property con-

necting it to the `Person` class. The subclasses beneath the `Person`, i.e. those representing roles within a production (such as `Client` and `Staff`), were then moved to beneath `Role` class. Figures 6.2 and 6.3 show these classes before and after these changes were made, respectively.

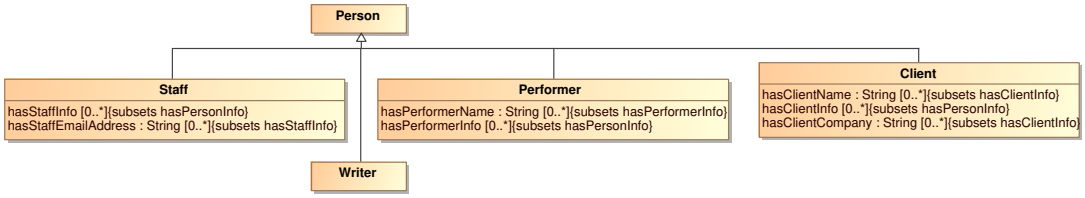


Figure 6.2: Original representation of `Person` roles in Iteration 1.

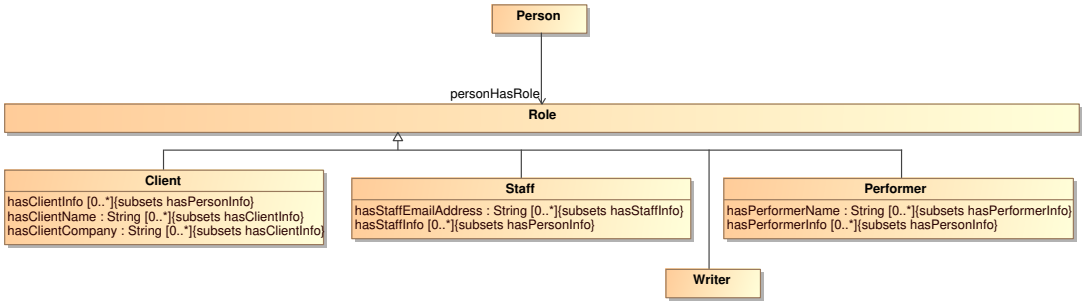


Figure 6.3: New representation of `Person` roles in Iteration 1.

The ontology at this stage of development formed the first iteration ready for feedback and its structure is summarised in Table 6.1. For the purposes of clarity, we have distinguished between usable object and data type properties versus all properties (i.e. the union of usable and non-usable properties). Usable properties refer to those with both a specified domain and range whereas non-usable properties do not and the distinction is necessary because properties in higher levels of the property hierarchy are used as a means of organising its sub-properties rather than as a means of connecting two ontology resources.

Structural Statistics	
Classes	57
Usable Object Properties	79
Total Object Properties	121
Usable Data Type Properties	189
Total Data Type Properties	222

Table 6.1: Structural statistics about the Creative Data Ontology after Iteration 1.

6.2.2 Iteration 2

The second iteration of the ontology was formed following initial discussion with the software development company that was commissioned to build a metadata management tool that will eventually be integrated with the ontology. During a briefing call, it was pointed out that there was ambiguity surrounding the difference between the meanings of the `Take` and `Clip` classes. As a result their meanings were refined so that a `Take` refers to the event of shooting one scene while a `Clip` is the output of a `Take`. This was expressed in the ontology by adding the object properties `clipIsFromTake` and `takeProducesClip` to reflect this relationship. Similarly, a further pair of relationships were added between the `Clip` and `Frame` classes to express that one clip is made up of many frames. Figure 6.4 shows the revised relationship.

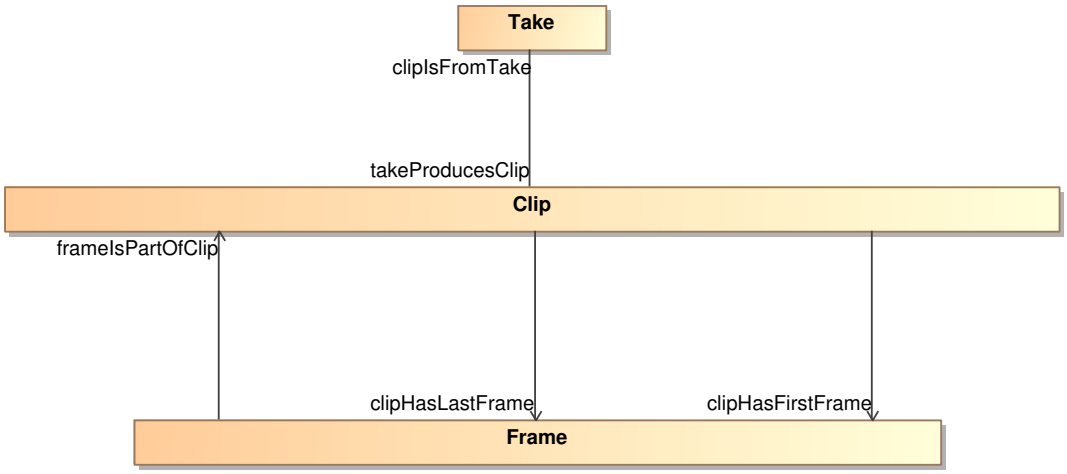


Figure 6.4: New representation of the relationship between `Take` and `Clip` in Iteration 2.

The ontology also aims to provide mechanisms for metadata versioning, which was incorporated into the ontology as part of the design sketch from Iteration 1. Although the classes required were present, they did not fully implement versioning so these were removed and a streamlined implementation was put in place instead. The original design was that the metadata of a class should be attached to a class with the same name but with the word "Metadata" appended to the end, such as the `CameraMetadata` class. The classes are subclasses of the `Record` class. The current version of a class instance's metadata record was an instance of the `WorkInProgressRecord` class with an object property connecting it to instances of the `Record` class, which represent previous versions. The versioning takes place in the metadata tool by converting the instance of `WorkInProgressRecord` to an instance of `Record` once the metadata is updated. However, the classes whose names are suffixed with the word "Metadata" are redundant because the attributes describing a class are directly connected to that class rather than its associated metadata class. For example, attributes are attached to the `Camera` class rather than the `CameraMetadata` class. Therefore, metadata versioning from this iteration of the ontology is implemented as described in Section 6.3.4.

Structural Statistics	
Classes	59
Usable Object Properties	102
Total Object Properties	152
Usable Data Type Properties	231
Total Data Type Properties	270

Table 6.2: Structural statistics about the Creative Data Ontology after Iteration 2.

6.2.3 Iteration 3

Following feedback from our industry partners, the third iteration of the ontology was formed by restructuring the object properties that connect the following classes: `Shoot`, `Scene`, `Take`, `Slate`, `Shot`, and `Clip`. The ontology from Iteration 2 contained these classes but they were not connected in a clear and concise way. For example, the `Scene` class is connected to the `Slate` class, which is connected to the `Shoot` class which is, in turn, connected to `Scene` which creates a cycle. Additionally, the `Slate` class is connected to both `Shot` and `Take` classes but `Shot` is also directly connected to the `Take` class. As a result, there are a number of redundant object properties, which could be simplified by chaining these classes as follows:

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Shot} \leftrightarrow \texttt{Clip}, \text{ and}$$

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Slate}$$

An additional class – `ShootDay` – was also added because a partner pointed out that a shoot can last multiple days and still be considered one shoot. The ontology from Iteration 2 only allowed the `Shoot` class to represent a single day of a shoot so this change allows each instance of the `ShootDay` class to represent a single shoot day and be assigned to an overarching multi-day instance of `Shoot`.

Structural Statistics	
Classes	62
Usable Object Properties	105
Total Object Properties	155
Usable Data Type Properties	231
Total Data Type Properties	272

Table 6.3: Structural statistics about the Creative Data Ontology after Iteration 3.

6.2.4 Iteration 4

Following the restructuring during Iteration 3, the partners gave feedback indicating that the representation of the relationship between a `Take` and a `Shot` is incorrect:

> "An example of a scene description would be 'an interview with James using two cameras, one focused on James and the other on the interview'. A scene is made up of multiple shots from multiple cameras. An example of a single shot would be the output of the camera pointed at James. A take is a version of a shot, and a clip is the digital video file representing a single take."

The ontology produced in Iteration 3 connects the two classes by stating that a `Take` produces multiple `Shots`. However, in the real-world, a `Shot` has multiple `Takes`, which meant that the chain of classes defined in Iteration 3 needed to be revised to reflect this new order. This meant that some existing object properties that connected `Scene` to `Take`, `Take` to `Shot`, and `Shot` to `Clip` had to be edited to reflect the new order:

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Shot} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Clip}, \text{ and}$$

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Shot} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Slate}$$

Figure 6.5 shows the original and the new chainings of the above classes.

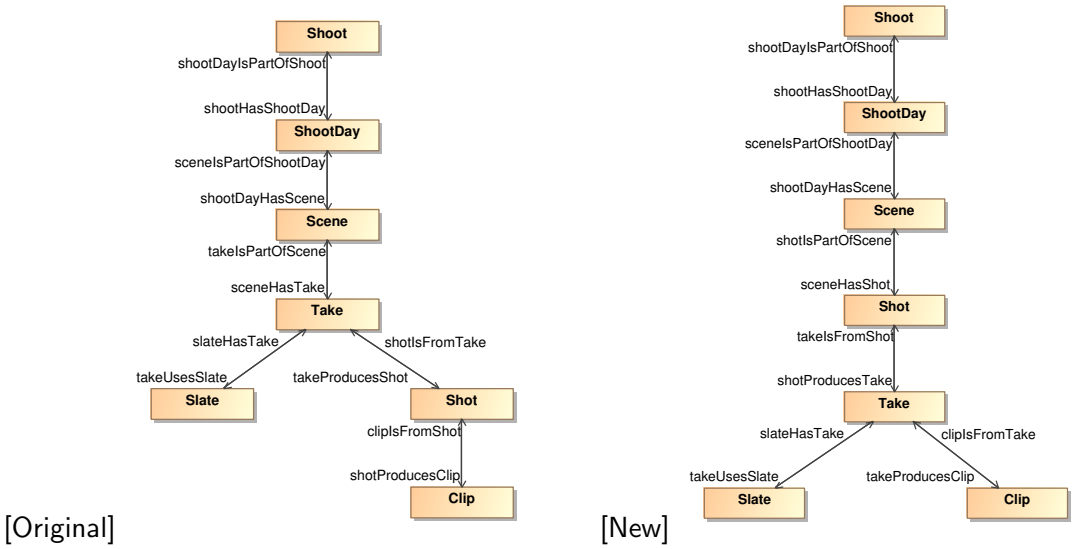


Figure 6.5: Original and new representations of the class chain.

Furthermore, during the development of the fourth iteration of the ontology, one of our industry partners requested that some additional real-world concepts be added to the ontology. Consequently, a `Rental` class was created and connected to the `Equipment` class to allow staff to rent equipment for use in a project, where each instance of the ontology represents one project. In addition, the following subclasses of `Asset` were also added: `Storyboard`, `ScriptBreakdown`, and `ShootingSchedule`.

Finally, as all subclasses (or 'types') of `Asset` can be represented as a digital file on the user's file system, the current version of the ontology contained a specialised object property for every subclass of `Asset` connecting them to the `File` class. This is shown in Figure 6.6.

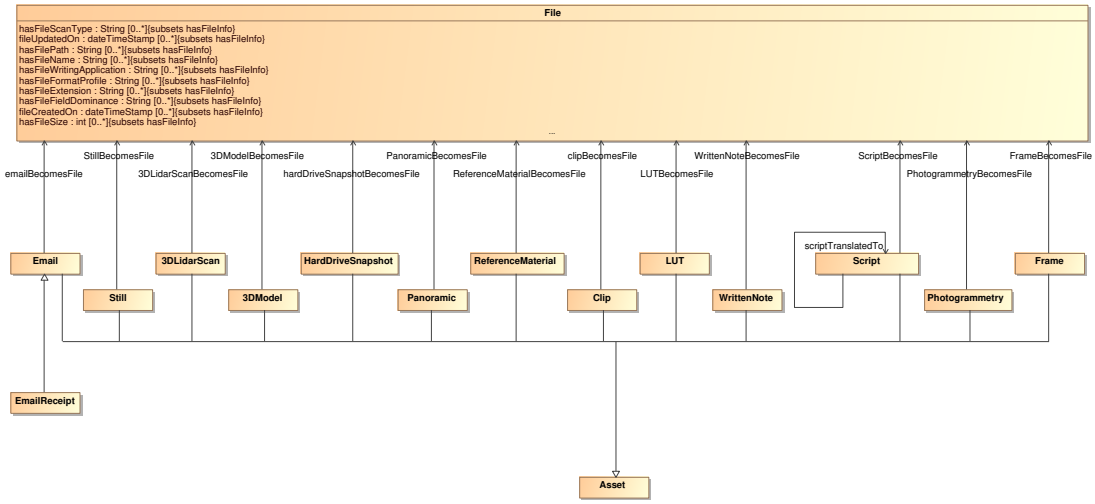


Figure 6.6: Original representation of the `Asset` and `File` classes from Iteration 3.

The `File` class encapsulates information about where an `Asset` that is conceptualised within the ontology is stored on a user's file system. These properties were all sub-properties of `becomesFile` with a range of `File` but each with a different domain, depending on the `Asset`. It was observed that this would soon become unmanageable because every time a new `Asset` type is added to the ontology, the user would have to remember to manually create a relationship with the `File` class. It would also make the ontology overcrowded and therefore impacting its clarity and conciseness.

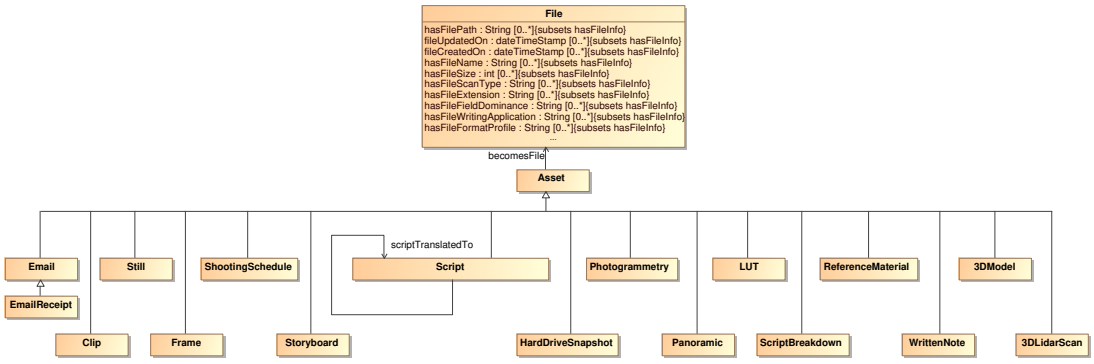


Figure 6.7: New representation of the `Asset` and `File` classes in Iteration 4.

In order to alleviate these concerns, the sub-properties of `becomesFile` were removed and instead the `becomesFile` object property, with a domain of `Asset` and range of `File`, will be used on its own. Figure 6.7 above shows this new representation.

Structural Statistics	
Classes	67
Usable Object Properties	101
Total Object Properties	159
Usable Data Type Properties	235
Total Data Type Properties	278

Table 6.4: Structural statistics about the Creative Data Ontology after Iteration 4.

6.2.5 Iteration 5

Unlike previous iterations, the development of Iteration 5 focused on making technical rather than semantic adjustments to the ontology while liaising with our commissioned software development company. The purpose of this was to ensure that the ontology file would be compatible with the technology being used to develop the metadata management tool. Specifically, this involved converting existing descriptions of classes and properties to use the `rdfs:comment` annotation rather than the custom `description` annotation, which was created during the early stages of the project. Using `rdfs:comment` meant that that third-party ontology conversion tools would recognise these annotations as descriptions and place them where appropriate. This iteration of the ontology formed the final version before the evaluation stage.

Structural Statistics	
Classes	65
Usable Object Properties	103
Total Object Properties	162
Usable Data Type Properties	234
Total Data Type Properties	278

Table 6.5: Structural statistics about the Creative Data Ontology after Iteration 5.

6.3 Ontology Use Cases

The Creative Data Ontology has two top-level classes – `DataItem` and `Record` – which are used to handle metadata versioning for Use Case 4, as described in Section 6.3.4. `Record` is a leaf class, meaning it has no subclasses, but instead encapsulates versioning metadata, including temporal details about a data item instance. `DataItem` is a superclass for all concepts that can have metadata stored about them. In order to define the scope of a prototype ontology, three use cases were defined in collaboration with the industry partners after those partners identified them as representative of key metadata generation and sharing practices in the industry. In addition, the fourth use case was defined by the Creative Data team as a means of facilitating metadata versioning within the ontology. It is not possible to describe the whole ontology within this thesis but the key classes and relationships used to represent these use cases are presented.

6.3.1 Use Case 1: Media Support in Television Post-Production

Media support operatives prepare the data received from production clients, ready for use in the later stages of post-production where media files are aligned into 'sync maps', which can then be edited by a team of creatives (sound, colour, and picture editors). Workflows in media support begin with accepting media from the in-house librarian and sending a receipt to clients containing a hard drive snapshot of the contents of the drives delivered.

The operative then makes a copy of this data, conforms it according to the desired format, and finally ingests it into the workflow. Pain points in the media support role centre on the absence of key metadata required for data ingestion and for the workflows. For various reasons, the required input from production teams is omitted or made in error and fixing this requires a great deal of "back and forth" communications between post-production and production, taking up the time of media support. One practitioner says:

> "A lot of people [on the production side] don't understand why we need roll numbers for example. There's a lot of feedback looping, a lot of people involved in getting media set up properly."

Support for this Use Case begins with the `Asset` class, which has three subclasses – `Email`, `EmailReceipt` and `HardDriveSnapshot` – to allow communications and email receipts sent to the client to be represented in the ontology. Like instances of other subclasses of `Asset`, all instances of these subclasses are also assigned to an instance of `File`, which encapsulates the metadata that describes how an asset is stored on the file system. They can also be linked to an instance of the `Client` class to indicate who it has been sent to. This is shown in Figure 6.8. When ingesting data, a new instance of a subclass of `Asset` – typically `Clip` – is created by the metadata management tool and the values of its data type properties are entered either automatically (by importing the existing metadata) or manually. Importantly, this supports practitioners by enabling them to keep track of communications with clients in relation to a production and thereby creating a contact log.

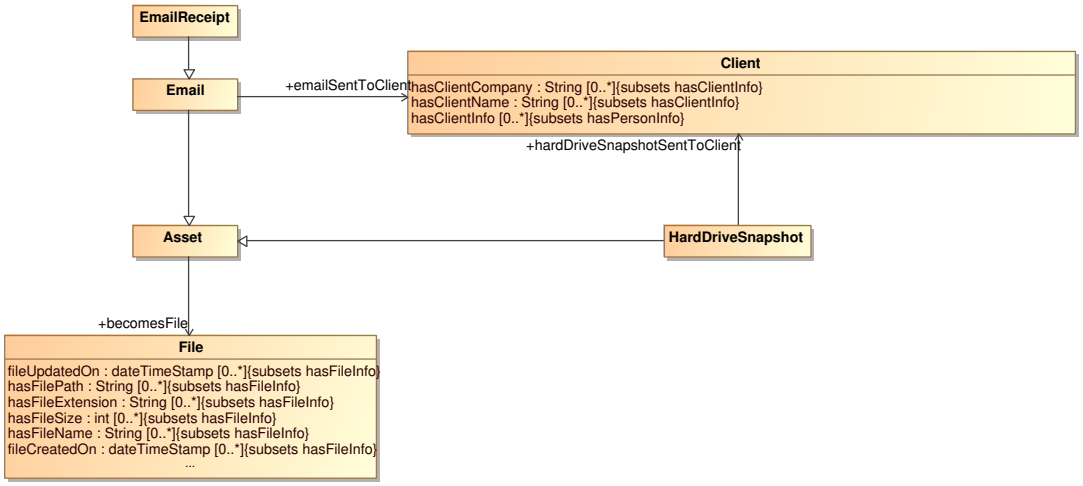


Figure 6.8: Support for Use Case 1 in the ontology

6.3.2 Use Case 2: Descriptive Logging in Television Post-Production

Unscripted and documentary content requires the addition of descriptive tags to the unedited footage in order to be able to find and edit content in vast collections of data captured on camera rigs set up on location. This process is referred to as 'live logging' because it involves logs written by people trained by the post-house. These loggers write annotations into a bespoke timestamped tool and the aim is to record the relevant facts of a given capture, as to what the footage depicts. The post-production staff and editors are then able to search for, recover and create edits with specific content. Pain points for this use case are connected with the lack of persistence of metadata across post-production re-edits, for example, when descriptive logs are lost after a near-final edit has been made. Hence, if and when a creative tweak to the final edit is needed, this requires the link to the descriptive logs to be painstakingly re-established.

> "What you'd hope to see is my guy sitting there, typing in descriptive metadata, when the lights go up on the lighting rig, and its suddenly got brighter; they're describing the scene and what's just happened; this gets described in

> the metadata at the same time ideally; the question is how to preserve this information so that it gets to the people who need it most – and send it to the people who are making an edit on this."

As the `Clip` class was added to the Creative Data Ontology to represent a single video file containing footage from production, this was the most appropriate class to attach logging tags to. Each instance of `Clip` can have any number of instances of `LoggingTag` assigned to it to describe its contents. Each tag is given a name and type which would enable the user to search for and generate a list of clips that match these attributes. Each entry in this list would have details of the clip and the timecode associated with the tag that matched the search criteria. Multiple instances of the `LoggingTag` can have both the same name and be allocated to the same `Clip` so they are distinguished by their combination of `Clip` and timecode. This is shown in Figure 6.9. This supports practitioners by enabling them to quickly generate lists of clips and timecodes by tag, which allows them to efficiently find the content that they require later in the post-production stage and thereby preventing such content from being lost.

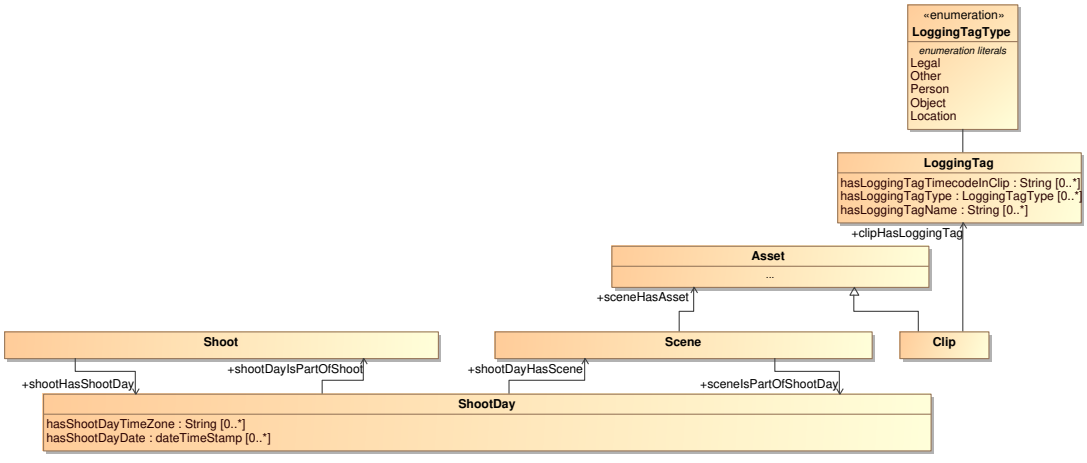


Figure 6.9: Support for Use Case 2 in the ontology

6.3.3 Use Case 3: On-Set Shoot Data in VFX Post-Production

During the pre-production phase of a feature film, the production company may secure one or more vendors to supply visual effects. This may involve the VFX company being on set to acquire the additional data and associated metadata required for VFX in a process referred to as 'Shoot'. This metadata is then collated into an on-set VFX database, which must be organised in such a way that it can be passed on to the larger central data management database for users throughout the companies' internal VFX pipelines. Pain points connected to this use case are similar to those in Use Case 1 regarding the absence of key metadata. An ingest practitioner stated:

> "there's been so many times when I've wanted to call up on set, and say 'Guys, can you provide the paperwork of what stuff is spherical and what's anamorphic?' Just so we don't have to view every one, and flip every one, and use eye."

Data types such as clips, 3D models, and reference materials are represented as subclasses of the `Asset` class. All instances of `Asset` are also assigned to an instance of `File`, which encapsulates the metadata that describes how an asset is stored on the file system. These instances of `Asset`, and in turn the instances of `File`, are then associated with a `Shoot` by assigning them to a `Scene` which in turn belongs to a `Shoot`. This is shown in Figure 6.10. This supports practitioners by connecting any given `Asset` to its associated metadata and also to the `Scene` and to the `Shoot` that it has originally come from. In the design of a metadata management tool, this feature of the sub-ontology improves the process whereby metadata of interest is located – often at later stages of post-production where finding or re-acquiring this information requires a great deal of back-and-forth communications.

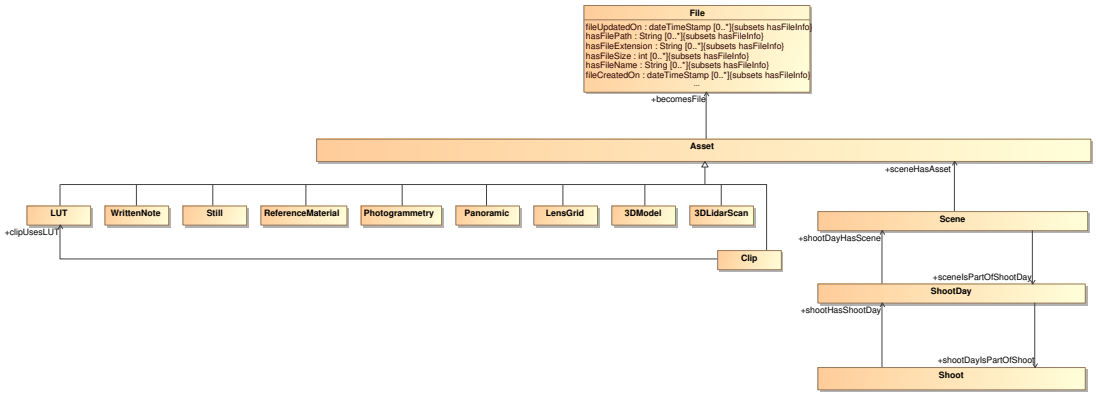


Figure 6.10: Support for Use Case 3 in the ontology

6.3.4 Use Case 4: Metadata Versioning

During the post-production phase of a media project, practitioners completing post-production need to have convenient access to the metadata generated throughout the life cycle of the project. This includes a full history of any metadata item in order to allow changes to be tracked. The pain point associated with this use case centres around metadata being lost due to changes overwriting the original metadata values without a log being kept to track these changes. One practitioner stated:

> "People just don't know how the changes they're doing affects other people. [Input/Output] folder is quite a good example of that. Your asset supervisor might go in and go 'I'll take that and I'll use that' [...] Or, he'll be like, that's not that person's name, their character's actually called this. The worst thing that can happen is that when we process those I/O days, and then they get changed afterwards. Because then it's gone so far down our part of the pipeline, somebody's changing it here and that breaks all the links that we've made."

Metadata versioning has been implemented using principles that are analogous to Russian dolls. That is, a record should be able to 'wrap' or contain another record relating to the versioning of that metadata. The ontology has two top-level classes aimed at link-

ing metadata and connecting records that describe the transitions that metadata typically undergoes: `DataItem` and `Record`. The former is an umbrella class for all entities in the ontology that can have metadata associated with it while the `Record` class encapsulates versioning metadata, including temporal details about a data item. To enable metadata versioning, every instance of `DataItem` and its subclasses should have an instance of `Record` assigned to it upon creation, using the object properties `dataItemHasRecord` and `recordIsAboutDataItem`. Instances of `Record` have a creation date, which is implemented through the `recordCreatedOn` data property. This schema is shown in Figure 6.11.

When the metadata of a `DataItem` instance is changed, a copy of that instance of `DataItem` is made with the metadata adjustments applied and a new `Record` instance is also created for the new `DataItem` instance. These are again connected using the `dataItemHasRecord` and `recordIsAboutDataItem` properties. The two instances of `Record` – the original and the updated – are connected using the object property `hasRecordVersion`.

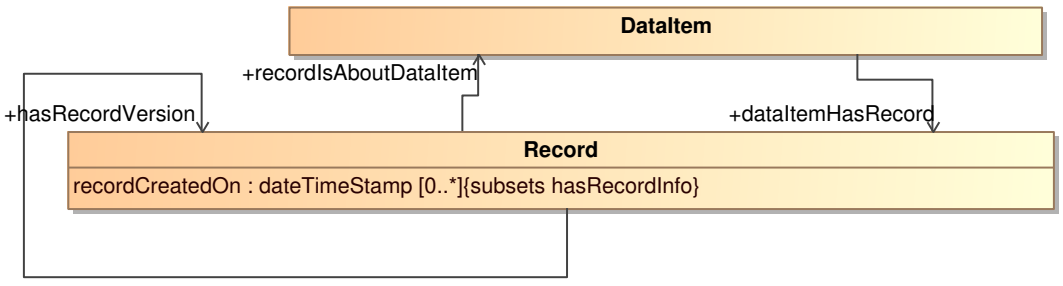


Figure 6.11: Implementation of metadata versioning in the ontology

6.4 Summary

In this chapter, we presented the process taken to design and build the Creative Data Ontology, including knowledge acquisition and the selection of tools that were used. We then explained in detail the iterations of the ontology, including justifying changes and

design decisions. Finally we described the four use cases that the partners stated that the ontology should support accompanied by an explanation of how each one was implemented in the ontology.

Chapter 7

Evaluating the Creative Data Ontology

In the previous chapter, we presented and discussed the implementation of the Creative Data Ontology in OWL using the 'Entities' tab in Protégé (specifically the 'Classes', 'Object Properties', and 'Data Properties' views). Once the building stage of the ontology engineering process has been completed, the evaluation of that ontology is important to validate its usability and ensure that it meets the criteria for a high-quality ontology. This evaluation involves two parts: 1) assessing the usefulness of the ontology with respect to the use cases it was developed for and the moving image industry as a whole, as defined in Section 6.3, and 2) evaluating the quality of the ontology in terms of the ontology evaluation criteria established in Section 4.4. This chapter will first define an evaluation strategy designed specifically for the Creative Data Ontology in Section 7.1 before presenting the results obtained by implementing this strategy in Sections 7.2 (interview results), 7.3 (quantitative results), and 7.4 (a note on minimal encoding bias).

7.1 The Evaluation Framework

Before evaluating the ontology, we must first define a strategy or 'framework' for doing so in order to ensure that we obtain the data required to thoroughly validate the ontology. To do this, we must consider which criteria the ontology should be assessed against and using which evaluation approaches. This section will justify the inclusion or exclusion of each of the nine ontology evaluation criteria, described in Section 4.4 while describing the evaluation process, including the approach, for each one that is included.

7.1.1 Internal Consistency

Internal consistency is defined by a set of rules embedded in the ontology design language. Specifically, Section 4.4.1 explains that internal inconsistencies of an ontology take the form of either cycles in the class hierarchy or when disjoint classes or custom restrictions cause contradictions. Evaluating an ontology for logical consistency is important because this will assist with identifying contradictions that could adversely impact on the intended meanings of the concepts involved. Therefore, the Creative Data Ontology should be evaluated for consistency, as should all ontologies.

Logical Evaluation

The OntOlogy Pitfall Scanner (OOPS!) plugin for Protégé described in Section 4.5.1 is able to detect cycles but not contradictions caused by class disjointness or custom restrictions. In order to evaluate the internal consistency of the Creative Data Ontology, we can use the OOPS! plugin by checking for Pitfall 6: *"Including cycles in a class hierarchy"*. This uses the logical evaluation approach and is more efficient and reliable than human inspection because it eliminates the risk of human error introduced by manually searching for and counting cycles. As mentioned previously, one of the shortfalls of the OOPS! plugin is that it cannot detect inconsistencies caused by contradicting disjointness assertions or custom restrictions. However, given that the Creative Data Ontology does not contain any disjointness assertions

or custom restrictions, there isn't a need to account for them when evaluating the ontology's internal consistency.

7.1.2 External Consistency

External consistency refers to the accuracy of the ontology when representing a domain. In order to ensure that the Creative Data Ontology can be used successfully, it needs to model the moving image creation domain correctly. If the domain is represented inaccurately, then the metadata management tool will not be able to work with the metadata correctly and is likely to return incorrect information. This will result in wasted time while the correct information is found, meaning that it is important that the Creative Data Ontology is evaluated for external consistency.

Human Inspection Approach

The Creative Data Ontology can be shown to domain experts in the form of diagrams during semi-structured interviews to elicit feedback on whether all necessary concepts have been correctly defined. Initially, the domain expert can be shown three segment diagrams of the ontology – one for each use case, excluding Use Case 4 – and asked to check that the classes, relationships, and attributes are correct in terms of the relationships between concepts accurately representing the real world and whether the attributes (or metadata fields) have been attached to the correct concepts. Additionally, as highlighted in Section 6.2, one major source of confusion with respect to defining concepts in the ontology was the order of the following chain of classes:

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Shot} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Clip}, \text{ and}$$

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Shot} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Slate}$$

All domain experts participating in the evaluation will also be shown a segment diagram of this chain and asked to confirm whether this order is now correct.

Gold Standard Approach

The gold standard approach would have been an effective and efficient way of quantifying the accuracy of the Creative Data Ontology. However, given that there was no known ontology available that could be used as a benchmark, we could not use this approach.

7.1.3 Completeness

In order to ensure that the moving image creation domain is adequately represented with no pertinent concepts or fields missing, it is important that the Creative Data Ontology is evaluated for completeness. Section 4.4.2 defines a set of metrics for evaluating a candidate ontology for completeness against a gold standard ontology. Given that such a benchmark ontology does not exist, these metrics cannot be used to measure the completeness of the Creative Data Ontology. Instead we can opt for both a corpus-based approach and a human inspection by domain experts.

Corpus-Based Approach

Our industry partners provided a set of metadata fields exported from their database systems, which forms a corpus that can be used to check the ontology's coverage against. In order to ensure that there is a balance between completeness and conciseness, each metadata field in the corpus will be classified as either explicitly represented within the ontology, not explicitly represented but instead derivable from other metadata fields that are explicitly represented, or not represented at all. There will also be a category for fields that were initially included in the set of core metadata fields but were later deliberately omitted for various reasons (described in Section 7.3.2).

Human Inspection Approach

The ontology can be shown to domain experts using diagrams to obtain feedback on whether all required concepts have been included. This would take place during a semi-structured interview, during which the interviewee can comment on any missing fields. The interviewer

can then either clarify that the missing field is derived from others fields or can make a note that the field is missing.

Gold Standard Approach

As with evaluating accuracy, the gold standard approach would have been a good way of measuring the completeness of the Creative Data Ontology. However, there was no benchmark ontology available.

7.1.4 Conciseness

A high quality ontology should be concise so as to enable ontology developers to make changes to and understand the ontology without being overwhelmed by redundant or irrelevant definitions. The Creative Data Ontology should therefore be checked that it does not contain such definitions and that it has minimal ontological commitment because it will eventually be integrated into a tool that will be used by non-technical domain experts.

Metric-Based Approach

Conciseness can be measured quantitatively using the four metrics defined in Section 4.4.3, which produce a percentage of redundant and irrelevant classes, instances, attributes, and relations. However, in order to identify which components are not needed from both a domain and technical perspective, domain and ontology experts would need to be consulted. It then becomes important to acknowledge that not all components will be relevant to all domain experts because the moving image industry (and the media industry in general) has a variety of sub-domains so obtaining a pseudo-accurate numerical measurement of conciseness would not be feasible. Instead we choose to use automated tools to evaluate conciseness by first quantifying the structure of the ontology and then drawing conclusions from the results. However, the metrics suggested by Lantow (2016) – average population and class richness – require the presence of instances within the ontology being validated, which are not found in the Creative Data Ontology [85]. Therefore, we use those suggested

by Vrandečić (2010) and Wei et al. (2020): maximum depth of the taxonomy, class to relation ratio, attribute richness, inheritance richness, and relationship richness [86, 87].

Human Inspection Approach

An alternative approach would be to measure conciseness qualitatively by eliciting feedback from both domain and ontology experts. Each domain expert participating in an evaluation interview would be shown segment diagrams of the representation of the use cases in the ontology and of the class hierarchy. They would then be asked to indicate if they consider any of the classes or relations within the use cases or the class hierarchy to be irrelevant or unnecessary. Meanwhile, each ontology expert would be shown only segment diagrams of the use cases[1] in the ontology and asked whether the the ontological representation of each use case is concise. Although the issue regarding different components being relevant to different people still applies, there is a greater capacity for discussion when taking this approach, meaning the results will be more informative.

7.1.5 Expandability

The domain of post-production is constantly changing so it is important that those changes can be incorporated into the Creative Data Ontology easily and without impacting on the set of established properties. An evolution-based approach to evaluating expandability would be appropriate, ideally when new knowledge needs to be added to the ontology. However, new knowledge would not be available at the time of evaluating the ontology so this criteria would not be able to be assessed. As a result, this may form a limitation with the evaluation process.

7.1.6 Clarity

A good ontology should be clear and easy to understand in order to ensure that non-technical domain experts can understand and use it correctly, and to assist ontology devel-

[1] Ontology experts are also asked to evaluate the representation of the metadata versioning use case.

opers to make amendments or additions to it in the future. As the Creative Data Ontology was designed to be integrated with a tool aimed at non-technical users, its clarity would need to be assessed.

Human Inspection Approach

Given that clarity relates to the representation of the domain and because clarity cannot be measured automatically, this criterion should be appraised using a manual inspection approach. This involves domain experts being shown use case diagrams of the ontology accompanied by a definitions list for the concepts, relationships, and attributes involved so they can then provide feedback centred around the following questions:

1. For each component in the ontology used to represent the use cases, is the definition for that component clear and accurate?

2. For each component in the ontology used to represent the use cases, does the definition for that component appear to overlap with the definition of another component?

3. Do you think most domain experts would understand this ontology quickly? If not, why? How could it be improved?

Ontology experts could also be consulted for feedback on the clarity of the ontology from a technical perspective. They would also be shown the use case diagrams[2] but are instead asked whether the the ontological representation of this use case is clear and intuitive.

7.1.7 Adaptability

Evaluating the adaptability of an ontology refers to assessing its ability to be adapted for new applications. However, the Creative Data Ontology is a domain ontology, not an application ontology, so the information that would otherwise be handled by an application ontology would instead be handled by the metadata management tool that the Creative

[2]Ontology experts are also asked to evaluate the representation of the metadata versioning use case.

Data Ontology will eventually be integrated into. Therefore, it would be not be appropriate to assess the adaptability of the ontology.

7.1.8 Computational Efficiency

One of the aims of the Creative Data Ontology is to streamline the workflows of creative practitioners, which includes making the processing and retrieval of metadata more efficient. Although a computationally efficient ontology is important, it is far more beneficial to ensure that common tasks can be completed in an appropriate time frame. This would be measured by observing industry practitioners while they use the ontology integrated with the metadata management tool to retrieve information and compare this with retrieving the same information from their current, comparable systems, such as a spreadsheet or a relational database. There is no specific benchmark times for performing such tasks defined in the literature but it is possible to define a set of times deemed reasonable by industry practitioners and then compare the time it takes for test users to complete these or similar tasks using the Creative Data Ontology. However, in this case it would be inappropriate to evaluate the ontology for efficiency because the metadata management tool bears most of the responsibility for ensuring users can complete common tasks efficiently, with the ontology holding a supporting role only. The tool would also be in a prototype phase at the time of conduct the ontology evaluation so it would not be possible to obtain a full set of results.

7.1.9 Minimal Encoding Bias

Given that the Creative Data Ontology will be integrated into a metadata management tool, it is necessary to assess the ontology for encoding bias and minimise it if possible. The ontology is a schema designed to organise the metadata only while the metadata management tool provides an interface between the user, the database storing metadata, and the ontology used to organise and retrieve it. Therefore, the metadata management tool should be the only component responsible for encoding the metadata. We should

therefore aim to minimise encoding bias within the ontology because such biases could conflict with the encoding implemented by the metadata management tool.

The ontology can be checked for any encoding bias through manual inspection. However, it would be inappropriate for a non-technical domain expert to perform this inspection so this must instead be carried out by a technical expert. In this case, the ontology could be assessed by the software development company commissioned to build and test the metadata management tool. The assessment would take place when they integrate the ontology with the tool, as any possible issues would be easily identifiable at that stage.

7.1.10 General Feedback

Although it is good practice to evaluate an ontology against a standard set of criteria such as those discussed within the literature, each ontology presents a unique set of requirements and challenges. Therefore, general feedback from domain experts should be sought as these criteria may not cover all comments that a domain expert may have. In addition to the criteria-based evaluation questions, domain experts evaluating the Creative Data Ontology would also be asked the following four open-ended questions while ontology experts were asked only Questions 1 and 3.

1. What did you think of the ontology overall?
2. What difference could the ontology make to your business?
3. How could the ontology be improved?
4. How could the ontology be extended?

Domain experts are also asked to give the ontology a score out of ten for each of the four evaluation criteria assessed during interview (accuracy, completeness, conciseness, and clarity) to allow us to quantify the quality of the ontology in terms of an average score. Ontology experts were asked to rate only conciseness and clarity because advanced domain

knowledge would be required to rate accuracy and completeness.

7.2 Interview Results

The sixth objective of this research was to design an ontology evaluation framework centred around the use cases and the ontology quality criteria identified in the literature. The evaluation framework used to obtain data on the quality of the Creative Data Ontology has been presented in Section 7.1 and so we shall now present the semi-structured interview data that was obtained as a result of executing this framework. Four domain experts and three ontology experts out of the twenty-one approached agreed to be interviewed and were asked to comment on the quality of the ontology. Every interview was recorded (with consent from the interviewee) and transcribed to facilitate the extraction of interviewees' comments. For the purpose of clarity, the evaluation process was performed on Iteration 5 of the ontology. Figures 7.1 to 7.4 show the diagrams of the use cases as shown to domain and ontology experts during the interviews:

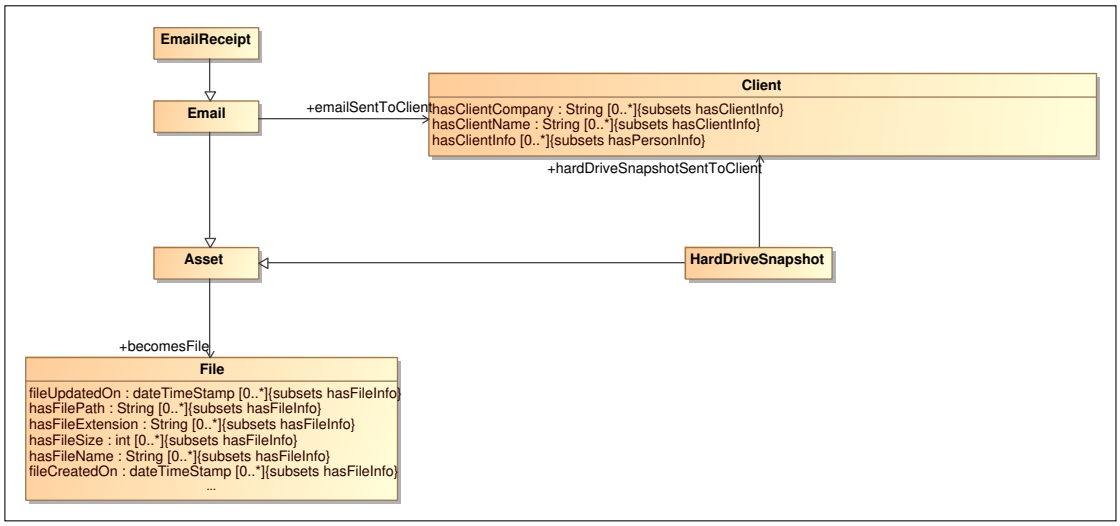


Figure 7.1: Diagram representing Use Case 1 as shown to experts.

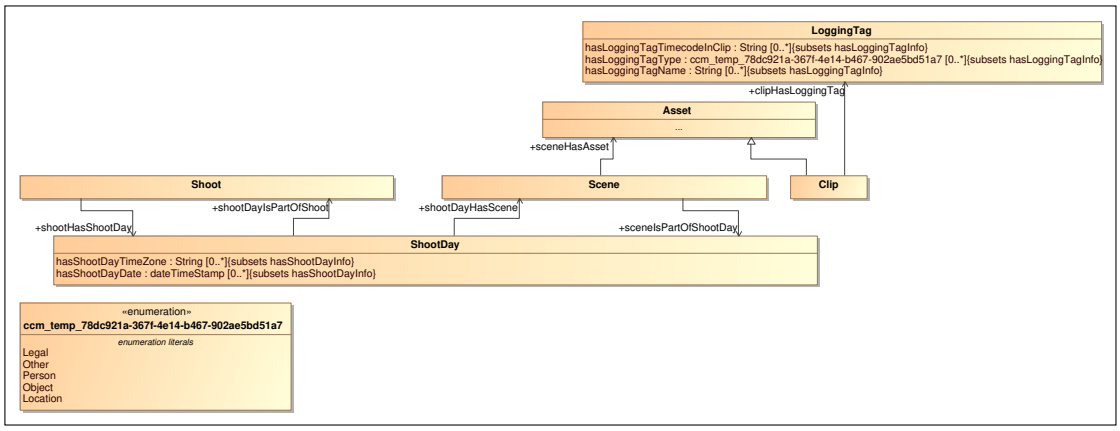


Figure 7.2: Diagram representing Use Case 2 as shown to experts.

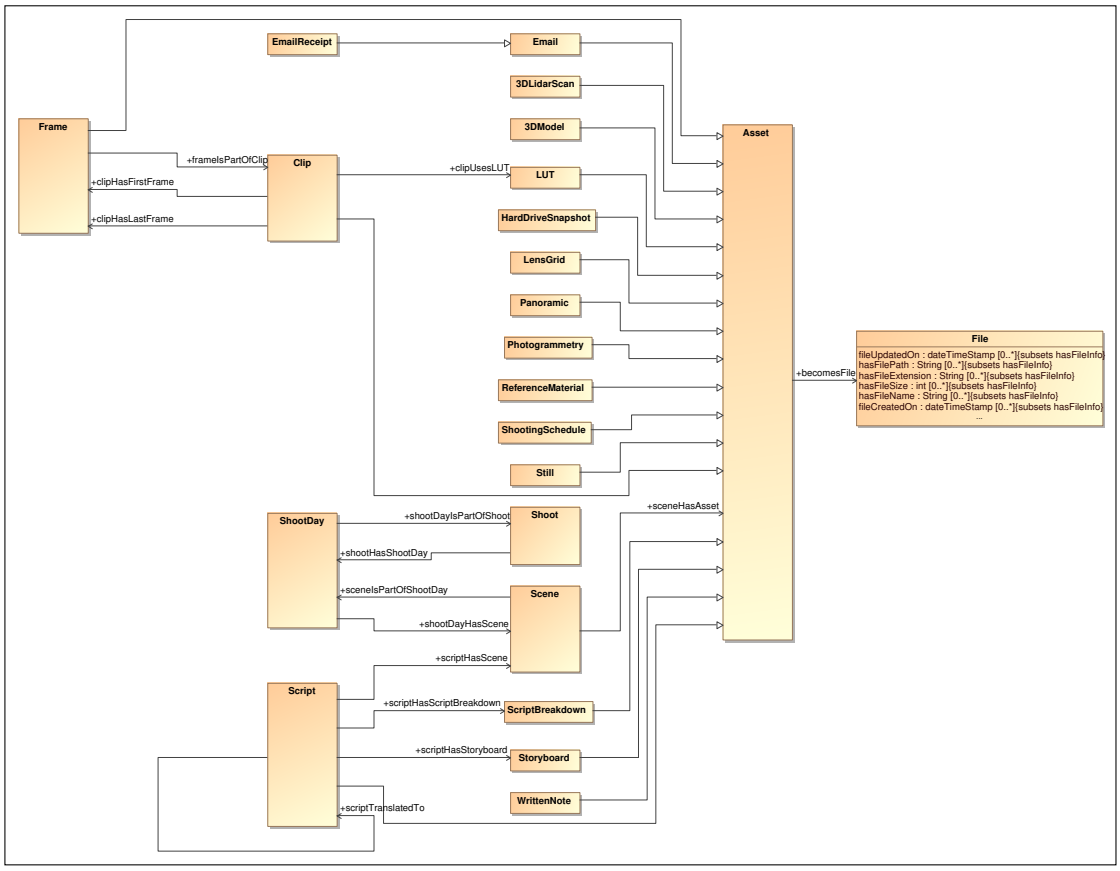


Figure 7.3: Diagram representing Use Case 3 as shown to experts.

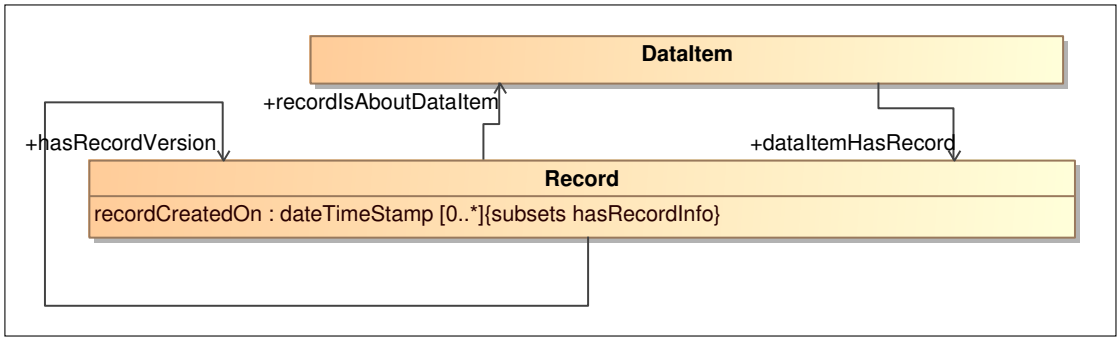


Figure 7.4: Diagram representing Use Case 4 as shown to ontology experts.

In order to contextualise the qualitative results, Table 7.1 maps each expert to their job title, the type of organisation they work for, and whether they are a domain expert or an ontology expert.

ID	Expert Type	Job Title	Organisation Type
D1	Domain	Director of Technology and Visual Services	Post-production (film, games, documentaries)
D2	Domain	Technical Project Supervisor	Post-production (unscripted, documentary)
D3	Domain	Head of Digital Humanities Lab	University in the EU
D4	Domain	Research Fellow in Creative Professions	University in the UK
O1	Ontology	Professor in Information Science and Media Studies	University in the EU
O2	Ontology	Director	Ontology Training Provider
O3	Ontology	Professor and Dean of Science and Technology	University in the UK

Table 7.1: Contextual information about the interview participants.

As can be seen in Appendix B, the slide deck used to facilitate the interviews was dependent on whether the interviewee was a domain or an ontology expert. Both domain and ontology experts were asked to comment on clarity and conciseness and domain experts were also asked to give feedback on accuracy and completeness of the Creative Data Ontology. Table 7.2 maps each criterion to the questions used to evaluate it.

Quality Criterion	Domain Expert Questions	Ontology Expert Questions	Shared Questions
Accuracy	2, 4, 6, 7, 10	-	3
Clarity	11, 12, 13, 14	1, 4, 7, 10	3
Completeness	1, 3, 5, 8	-	3
Conciseness	9	2, 5, 8, 11	3

Table 7.2: A mapping of ontology evaluation criteria to evaluation interview questions.

It is important to note that some questions asked during the interviews were not designed to address the evaluation criteria and were instead included to obtain general feedback, such as suggestions for expanding the scope of the ontology in future work. These questions are not included in this table.

7.2.1 Domain Expert Questions

Each domain expert interviewed was asked 19 questions, three of which were also asked to ontology experts. This section will present the answers received in response to the 16 questions asked uniquely to domain experts.

Domain Expert Question 1

> *Does the ontology have all the classes and properties required to represent Use Case 1? If not, which are missing?*

D2 and D3 agreed that the ontology contained all the required classes and relationships needed to represent Use Case 1. D4 agreed but also noted that there are other workflows for media support incorporated in the industry: "that does seem to make sense to me as one kind of workflow but I can't imagine it's the only kind of workflow that would happen though". D1 suggested adding further attributes to the `File` and `Client` classes: "The only other attributes I would normally associate with files are permissions and checksum.

[...] And then on the company: client company, client name, client info. What about client email address? Or is it implied that you must know that to send the email? [...] You might want the client address, you might want the client account number." Despite not being asked this question, O1 noticed that the ontology was missing a vital object property between `Email` and `HardDriveSnapshot` in this use case: "The email has the hard drive snapshot attached to it but that's not really clear in this diagram. It says that the email is a subclass of asset and hard drive snapshot is also a subclass of asset. Then the email is sent to the client and the hard drive snapshot is sent to client. But it's not very clear that the snapshot is part of the email."

Domain Expert Question 2

> *Have the classes been connected correctly in Use Case 1? Have the fields been attached to the correct classes?*

The four domain experts who were asked whether the classes in Use Case 1 have been connected correctly and the fields attached to the correct classes unanimously agreed that the ontology was accurate with respect to this.

Domain Expert Question 3

> *Does the ontology have all the classes and properties required to represent Use Case 2? If not, which are missing?*

D3 agreed that the ontology contained all the required classes and relationships needed to represent Use Case 2. The remaining three domain experts (D1, D2, and D4) stated that the number of logging tag types is too low. D2 said "you've got legal, other, person, object, location. You probably could put something like emotions in there and then action" while D4 said "those logging categories are very, very vague and wouldn't really allow you to pull out enough information to build a story from all your assets". D4 then went on to explain that when they were doing logging "we were able to set our own titles and we had

many more because we would have individual people names".

D1 also made suggestions for additional attributes that would be useful to have in this use case. These attributes were the logger creating the link between an instance of `Clip` and `LoggingTag` and the date and time of creating the link:

> "What about the loggER? The person who logged it? Is that an important concept in this use case? And I don't know if it's overkill, but the time and date that they created that log."

whether the item being logged is audio or video:

> "And it might be worth as well identifying whether the thing being logged is picture or sound. I don't know if that's entirely relevant to the use case, but in theory you're not just looking at pictures, are you, you're listening to the audio that's associated with it."

and the number, n, used to represent the number of the shoot day (i.e. the nth day of the shoot):

> "I appreciate this is unscripted in this particular use case, but often although the day will be important, it might be the 6th day of the shoot or the 10th day and those things sort of become important because Day 10 is often used as descriptive by production rather than the actual date."

Domain Expert Question 4

Have the classes been connected correctly in Use Case 2? Have the fields been attached to the correct classes?

D1, D2, and D3 agreed that the classes have been connected correctly and that all fields have been attached to the correct classes. However, D4 noticed that the ontology is missing

an object property between `Scene` and `Clip` in this use case, which was supported by two ontology experts (O1 and O2). O1 said "I don't see anything there that directly relates a scene to a clip or a clip to a scene". However, due to the availability of space in the slide decks shown to the interviewees, it was not possible to show that the `Scene` and `Clip` classes are connected via a chain of classes, as follows:

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Shot} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Clip}, \text{ and}$$

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Shot} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Slate}$$

As a result, this issue was raised three times across all interviews.

Domain Expert Question 5

> *Does the ontology have all the classes and properties required to represent Use Case 3? If not, which are missing?*

D2 and D3 agreed that the ontology contained all the required classes and relationships needed to represent Use Case 3. D4 pointed out that as this use case involves the VFX company being on-set to acquire additional metadata which is then collated into an on-set VFX database, the process of backing up this database should be reflected in the ontology: "Backing up is something that always happens somewhere on set as well, because of the inherent dangers of losing footage and data if you don't back it up. So, there will be someone with that job and that should be reflected somewhere in the workflow."

D1 suggested the addition of witness camera materials as a subclass of the `Asset` class because often in VFX production, witness cameras are used to record what is happening out of shot: "Might there be any witness cam material? You've got your main cameras shooting the shots, so to speak. But what they often use, particularly with VFX is additional cameras, recording almost as a witness to what's going on to show certain things that are out of shot, where everything was positioned."

Domain Expert Question 6

> *Have the classes been connected correctly in Use Case 3? Have the fields been attached to the correct classes?*

The four domain experts who were asked whether the classes in Use Case 3 have been connected correctly and the fields attached to the correct classes unanimously agreed that the ontology was accurate with respect to this.

Domain Expert Question 7

> *Does the class hierarchy look correct and if not, what changes should be made?*

The four domain experts who were asked unanimously agreed that the class hierarchy does look correct, although at this point, D4 suggested changing the name of the `Deliverable` class to `Deliverables` because "you rarely only deliver one file. It's almost always a suite of things that you deliver".

Domain Expert Question 8

> *Are there any classes missing from the ontology? If yes, which ones?*

All four domain experts had suggestions for classes that should be added to the ontology, all centred around specialising higher level classes by giving them a set of subclasses.

D1, D2, and D4 highlighted that the `Equipment` class should have more than five subclasses, with D4 saying that "those are five out of maybe 200 bits of equipment that you might have on a set". D1 said that the list of equipment should be "much bigger and longer" and suggested that there should be subclasses representing "recording devices like hard drives, for example, or mags". D2 specified a few items of equipment and their associated attachments that were missing: "Maybe equipment you could have something like a computer, because those are important, and then equipment attachments you might be

able to have something like carts because you shoot on carts, which are important to keep track of."

D3 and D4 specified that the `Staff` class could also have additional subclasses in addition to the four already there in order to fully capture all roles in production and post-production. D4 highlighted the sheer number of roles: "there are literally hundreds of production jobs on a film set. So if you were going to be totally comprehensive and you wanted to cover all of those people's roles, then that would be another place that would be expanded.". D3 suggested specific roles to be added: "When I was at a shoot you'd have camera person, runner, you know, all those subclasses. [...] I don't know if it's a role, but you used to have the person that (this was in film) but people who managed the storyboarding but that's probably not a role people have anymore."

D3 and D4 suggested that the `Deliverable` should be broken down into subclasses to account for different versions of a final production. D4 said that "You might deliver the film as it's supposed to be for broadcast, you might deliver a film that has different music on it for a different audience, you might deliver a film that has presenter bits taken out because they're going to be put in by someone from, you know, it's going to be sold in a foreign territory so they're going to put in their own presenter bits but anyway. You're quite often delivering lots of different versions of the thing that you have made. And again, you might want to break down deliverable into that." D3 also made the same observation when asked how the ontology could be extended: "Sometimes you shoot two different scenes for different markets. So for example, things to do with sensitive subjects like pornography or what's considered pornography. You sometimes would shoot a drama and then you would have some scenes which are acceptable for one audience and not for another."

Finally, D4 also commented that the ontology is missing some subclasses of `Asset` that represent the paperwork involved in a production, in addition to the `WrittenNote` class: "Written notes... So then there's other production paperwork such as budgets and schedules

which also feed into the workflow, and can affect the workflow a lot. So if you've got fixed budget and your shoot overruns because something goes wrong, then you need to adjust something else so that you don't go over budget. So that kind of production paperwork alongside script breakdown, storyboards, and scripts are an important part of the process."

Domain Expert Question 9

> *Do you consider any of the classes within the ontology irrelevant or unnecessary? If so, which ones and why?*

D2 and D3 agreed that there were no unnecessary classes in the ontology, based on a visual inspection of the class hierarchy. D1 asked why a lens type is represented as a subclass of `Lens` rather than as a data type property: "Why aren't they effectively just one thing like lens type where you put that data inside. Why do they exist as subclasses?" The final domain expert, D4, felt that the `Email` class was out of place as a subclass of `Asset`: "The one that keeps jumping out is email because email is such a ubiquitous tool. Yes, it can be an important way of getting a message from A to B, but it doesn't feel to me... It kind of sticks out as being unlike everything else there, weirdly, but I suppose it's how you join up the assets, isn't it? One gets emailed to another, confirmed by email so maybe it is right..."

Domain Expert Question 10

> *Are these classes chained in the correct order and if not, what is the correct order?*

This question was in reference to the chaining of the following classes, whose order was the subject of multiple discussions during the ontology development stage:

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Shot} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Clip}, \text{ and}$$

$$\texttt{Shoot} \leftrightarrow \texttt{ShootDay} \leftrightarrow \texttt{Scene} \leftrightarrow \texttt{Shot} \leftrightarrow \texttt{Take} \leftrightarrow \texttt{Slate}$$

D1, D2, and D3 agreed that the class order in the chain was correct. The final domain

expert, D4, also agreed but stated that the correct class chain may not work with unscripted content: "When you've got unscripted stuff or it is stuff that's using remote cameras, say in a hospital or whatever, you haven't got someone going 'action' and 'cut' because you know, it's like basically the camera operator will be tasked with covering this certain area of the hospital. And when something interesting is happening or he hears in head, in his earpiece or her earpiece: quick, pick up that, pick up that piece over there it's... those aren't arranged in takes in the same way like in a drama."

Domain Expert Question 11

> *Are these definitions clear and accurate? If not, which definitions are unclear or inaccurate?*

D3, who was interviewed early in the evaluation process, declined to answer this question because the list of definitions was too long as it covered the whole ontology. As a result, the list of definitions was reduced to only those of the components used to represent the three use cases for future interviews. However, all three domain experts who did answer this question (D1, D2, and D4) agreed that the definitions of the use case components were clear and accurate.

Domain Expert Question 12

> *Do any of these definitions appear to overlap? If so, which ones?*

D3 declined to answer this question, due to the size of the definitions list, and D4 did not identify any overlapping definitions. D1 did point out that there was an overlap between the definitions of `Roll`, `CameraRoll`, and `SoundRoll` and D2 was asked in a later interview whether they shared this opinion to which they responded: "I guess [Roll] could overlap. It's more vague than camera roll or sound roll. So you only have a camera and a sound roll. So if you have both of those as definitions, then roll might be superfluous because you've got more specific definitions."

Domain Expert Question 13 and 14

> *Do you think most domain experts would understand this ontology quickly? If not, why?*

> *How could the clarity of this ontology be improved?*

D3 declined to answer these questions too, as they felt unable to comment, given that they declined to answer the previous questions assessing the clarity of the ontology. Of the three domain experts that did respond, they all agreed that most domain experts would understand the ontology quickly. However, D1 and D2 stated that the ontology itself was clear and easy to understand but with the caveat that domain experts would need to understand the concept of ontologies in general. For example, D1 said: "The ontology itself I don't think is [unclear]. But particularly when you begin to describe the relationships and the way they affect one another, I think it is a complicated thing to explain."

Domain Expert Question 15

> *What difference could this ontology make to your business?*

Every domain expert was asked what difference the ontology could make in their organisation and the responses were varied but encouraging. D1 suggested that it would "hopefully bring some order to the chaos" while D3 said "it would bring consistency" in terms of labelling the outputs of a production when the time comes to archive them. D2 believed the ontology could act as "a means of keeping information because the biggest problem that we have is loss of information". D4 pointed out that the ontology could be used to capture knowledge that is otherwise gained from experience. D4 gave the example of the virtual production sub-domain, which is relatively new and knowledge is often not documented as it is learnt and they suggested the ontology allows users to "understand how [concepts] fit together in a structure".

Domain Expert Question 16

How could the ontology be extended?

This was a very open-ended question which, as a result, returned a range of suggestions on areas to expand on in future work. Note that some of the answers provided to this question related to the completeness of the existing ontology so were presented in previous sections instead.

D3 suggested expanding the ontology to cover details that are specific to genre domains: "We've only really touched on scripted and non-scripted and those are quite big but even in those there will be smaller sub-domains of comedy, drama, natural history, reality and each one of those will want to extend their own ontology." Meanwhile, D2 suggested extending the ontology to cover other types of production besides scripted film, scripted and unscripted television, and VFX: "There's also probably scope then for [virtual reality], and that's going to become, well, in my opinion it will become bigger." D2 then continued to say that the ontology is a "very broadcast-centric one" and so they suggested extending it to cover streaming: "There's more sort of facets to this industry that didn't get involved with this chat so that's probably where you would extend it [...] extend it into streaming initially." D3 expanded on this idea by suggesting that the ontology could handle the metadata involved in producing interactive content: "One thing which came along was interactive content. So for example, at Wimbledon, you would have it where people could choose the red button and watch different games. So maybe that's not essential in a film set where the event isn't live, but in live events you may need a different kind of logging label."

D1 believed that a significant portion of the ontology would be relevant to other sub-domains such as virtual production or immersive media: "I suspect probably 90% of it will already work for [virtual production or immersive subdomains], but potentially we'll need to add some more classes or subclasses and potentially some relationships." However, it was

suggested by D3 that we have sub-ontologies derived from the main or 'master' ontology for each of the various types of production: "You could have different forms of your ontology, so you would have ontology for live events or ontology for drama. Those kinds of things so it could be specific. [...] It could be that some types of ontologies are useful for different genres or different types of production. Because shooting a commercial in a fixed studio is so different to doing a sports event, for example."

A final recommendation made by D3 was to provide coverage for drone footage and other aerial shots: "Increasingly now you would have to have a label for drone footage or for many of the productions now use remote control things to get scene shots they couldn't get before. You know, aerial shots or they're using drones."

7.2.2 Ontology Expert Questions

Each ontology expert interviewed was asked 15 questions, three of which were also shared with domain experts. This section will present the answers received in response to the 12 questions asked only to ontology experts.

Ontology Expert Question 1

> *Is the ontological representation of Use Case 1 clear and intuitive? If not, how could clarity be improved?*

All three ontology experts agreed that the ontological representation of Use Case 1 was clear and intuitive but O2 also stated that the diagrams could be presented as top-down (i.e. general to specific) as it is more intuitive to non-technical people. While this was a suggestion for improving the diagrams rather than the ontology schema, it is important to note this for any future discussions with non-technical experts.

Ontology Expert Question 2

> *Is the ontological representation of Use Case 1 concise? If not, how could conciseness be improved?*

All three ontology experts agreed that the ontological representation of Use Case 1 was concise and had no further feedback for this question.

Ontology Expert Question 3

> *Do you have any further comments on the ontological representation of Use Case 1?*

Only O2 had further feedback on this use case, which was that the last sentence of the use case description, "The operative then makes a copy of this data, conforms it according to the desired format, and finally ingests it into the workflow" is not included in the diagram.

Ontology Expert Question 4

> *Is the ontological representation of Use Case 2 clear and intuitive? If not, how could clarity be improved?*

O1 explicitly agreed that the ontological representation of this use case was clear and intuitive but advised that there should be a relationship between `Scene` and `Clip`, which was an issue also highlighted under Domain Expert Question 4. O2 also shared this view but O3 did not identify this as an issue.

Ontology Expert Question 5

> *Is the ontological representation of Use Case 2 concise? If not, how could conciseness be improved?*

O2 agreed that the ontological representation of this use case was concise, while O1 and

O3 commented that the diagram contained unnecessary details that caused the diagram to become cluttered. O1 said: "I think the shoot and shoot day stuff. It sort of adds detail that maybe it's getting in the way where you could have other details about the post production staff and all that stuff instead."

Ontology Expert Question 6

> *Do you have any further comments on the ontological representation of Use Case 2?*

Only one ontology expert, O1, had further feedback on this use case, which was that it could include the camera rigs and their locations, as mentioned in the use case description, and the participants involved in the logging: "You mentioned data captured on camera rigs setup on location so you could include camera rig and location" and "Maybe just have asset and scene and clip and take and leave out all the shoot day stuff. Then you've got room to put production staff and editors in there - roles in relation to the logging tag and the clip."

Ontology Expert Question 7

> *Is the ontological representation of Use Case 3 clear and intuitive? If not, how could clarity be improved?*

O1 suggested that the subclasses of the `Asset` class should be further grouped in order to improve the clarity of the ontology: "The asset class could be subclassed a little bit, because I can see there's emails and there's written notes and there's storyboards, so there's kind of written materials and then there's film materials, and then they are subclassed again. So it might make it a little bit more intuitive if you found a couple of other subclasses within asset." O2 was asked if they agreed with this feedback to which they agreed, stating that "it would be worth grouping them into taxonomic hierarchy of generalisation relationships" if there is any commonality between the types of asset. A domain expert (D4) who was

interviewed after this suggestion was made was also asked and they agreed that the clarity of the ontology would be improved by grouping the subclasses of `Asset`. O3 agreed that the representation of this use case was clear and intuitive and when asked whether further grouping the subclasses of `Asset` would improve the clarity of the ontology, they stated that they could not give a firm answer as this would be a question for domain experts.

Ontology Expert Question 8

> *Is the ontological representation of Use Case 3 concise? If not, how could conciseness be improved?*

O1 and O2 agreed that the ontological representation of Use Case 3 was concise and had no further feedback for this question. O3 suggested removing from the diagram any subclasses of `Asset` that were not VFX-related, such as `Email`, as they were unnecessary to successfully represent the use case.

Ontology Expert Question 9

> *Do you have any further comments on the ontological representation of Use Case 3?*

Only O2 had further feedback on this use case which, like Use Case 1, was that the last sentence of the use case description "described stuff that's off the page". However, they did agree that the last sentence is "more like narrative about the use case rather than part of the use case itself" so it wasn't necessary to include it in the ontology.

Ontology Expert Question 10

> *Is the ontological representation of Use Case 4 clear and intuitive? If not, how could clarity be improved?*

Two ontology experts commented on the clarity of this use case. O1 stated that "this

one is the most clear [use case] I think" while O2 agreed that this use case was mostly clear with one exception: "The thing that might be less intuitive to some readers would be how the record version is itself a record. It's not clear if that means that there's another kind of record that is the version of a record or that each record has a prior version, or a next version or something so that it's bit unclear what that `hasRecordVersion` property means." Upon receiving clarification, this expert went on to say: "That should really be something like hasPreviousVersion or something like that. Or prior version? So what you're saying is that the hasRecordVersion property, an instance of that would relate two records about the same stuff created at a different date and time." O3 agreed that the ontology was clear and intuitive with respect to this use case.

Ontology Expert Question 11

> *Is the ontological representation of Use Case 4 concise? If not, how could conciseness be improved?*

All three ontology experts agreed that the ontological representation of Use Case 4 was concise and had no further feedback for this question.

Ontology Expert Question 12

> *Do you have any further comments on the ontological representation of Use Case 4?*

Only O2 had further feedback on this use case which was that we "could put a sub-property for the DataItem to say what's its data about? But I don't think you necessarily need that." As `DataItem` is a top-level class, it was included as a means of organising the class hierarchy and so it should not have any instances. Instead, any use of the `dataItemHasRecord` or `recordIsAboutDataItem` object properties (which are defined as having `DataItem` as their domain and range respectively) would use instances of the subclasses of `DataItem`. However, the `DataItem` class does have a description defined using the `rdfs:comment`

annotation to clarify its purpose.

7.2.3 Shared Questions

Only three questions were applicable to both domain and ontology experts and these were focused on the ontology as a whole. More specifically, experts were asked to share their opinion of the ontology overall and suggestions for how it could be improved. They were also asked to rate the ontology out of 10 for the evaluation criteria. This section will present the answers received in response to these questions.

Shared Question 1

> *What did you think of the ontology overall?*

Seven domain and ontology experts gave a wide range of concluding opinions on the ontology, which were generally positive. D2 described the ontology as "really thorough" while D1 said it was "a good concept" but also pointed out that as time progresses, changes and additions will need to be made so the ontology needs to be capable of facilitating that: "I think it's a really good starting point and I think as long as it can be built upon and, you know, as technology evolves and methods evolve and change it seems likely that additions will have to be made to it." O1 also agreed it was "a good start" but stated that previous issues related to subclass hierarchies and missing relationships need to be resolved: "but I think it needs those refinements in terms of the subclass hierarchies and some of the relationships which are not clear at the moment." D3 commented that they "like it, it's clear from the sense of the structure. I would say it's complex, but it probably has to be complex." The final three experts (D4, O2, and O3) also gave generic positive responses while also reiterating the key suggestions for improvements they had made in response to previous questions.

Shared Question 2

How could the ontology be improved?

All three ontology experts interviewed commented on the diagrammatic representations of the use cases rather than focusing solely on the ontological representation. O2 agreed that Use Case 1 was clear but suggested that "if you're presenting stuff to a business audience, I wouldn't have generalizations going upwards and sideways. They're just a little bit harder for somebody who is not a modeller to understand what that means. So I'd go top down." This standpoint was reinforced by D4 who said "in my head visually, having a more hierarchical structure so you have the biggest thing at the top [...] and it kind of flows down the page would make more visual sense to me." Meanwhile, another ontology expert, O1, said that "this isn't the kind of diagram I would look at as an ontology because it's kind of an unusual diagram with your own particular semantics for those arrowheads. So it wasn't intuitive for me straight away. If you showed me the original Protégé ontologies, it would have possibly been a lot more obvious to me." O3 also commented on the diagrams, often asking why certain classes were included when they were not directly relevant to the use case.

Another suggestion, which is related to the issue regarding the diagrammatic representations, is that the idea of ontologies are complex in general, especially when presented to non-technical domain experts. D2 highlighted this and suggested that the ontology could be improved by adding a "layer" on top to shield them from the technical details of the ontology: "It's not really the ontology that's the problem. It is the presentation of it to people that don't understand the ontologies, and getting them to understand the value of that information, because it's not immediately clear. [...] You could almost put like a layer on top that makes it more appealing to people that don't want to think about all the technical stuff." A second domain expert, D3, also indicated that the complexity of ontologies could be an issue.

The final suggestion for improvement, made by D3, was to better handle alternative terminology that refer to the same concept and to clarify any ambiguous terms by "putting something in brackets [...] for the untrained person". The justification behind this suggestion was to improve the search functionality by having more than one way of finding something and D3 presented the example of the logger in Use Case 2: "what you would call a logging person or a logger, somebody else might have a different term for it. And the reason is, in America they have different terms for production roles than in England, so it's nice having a kind of catchall."

Shared Question 3

> *Out of 10, how would you rate the ontology for accuracy, completeness, conciseness, and clarity?*

All interviewees were asked to give a rating for conciseness and clarity while domain experts were also asked to give a rating for accuracy and completeness. The range for the ratings was 0 to 10, where 0 means that the expert believes the criterion has not been met at all and 10 means that the expert has no or has very few, minor suggestions to improve the ontology with respect to that criterion. The results for each criterion are presented in Table 7.3, accompanied by the average (mean) score:

Criterion	Results	Mean
Accuracy	4, 8, 9, 10	7.75
Completeness	4, 7, 9, 10	7.50
Conciseness	7, 7, 8, 8, 9, 9, 10	8.29
Clarity	3, 5, 7, 7, 8, 9, 10	7.00

Table 7.3: Creative Data Ontology evaluation criterion ratings.

7.3 Quantitative Evaluation Results

We have so far described and discussed the results of evaluating the ontology through a qualitative approach, by interviewing experts and analysing their feedback based on a manual inspection of segments of the ontology. However, it is also important to quantitatively evaluate the ontology as it enables us to gain objective insights into areas for improvement. This section will present the results obtained through a quantitative analysis of the ontology.

7.3.1 Internal Consistency

The OOPS! plugin for Protégé described in Section 4.5.1 did not detect any cycles in the structure of the ontology and, as mentioned previously, there were no disjointness assertions or custom restrictions within the ontology.

7.3.2 Completeness

In Figure 7.5, we present the results of evaluating the Creative Data Ontology for completeness using the corpus-based evaluation approach. Given the provided list of metadata fields identified by the partners as 'core', each field was then checked for representation within the ontology and placed into one of the following four categories:

1. **Explicit**: metadata fields that have explicit representation within the ontology have either an object property (relation) or data type property (attribute) that directly represents the same domain knowledge as the metadata field in question. For example, a metadata field representing a note that is attached to the record of a shoot is explicitly represented in the ontology using the object property `hasShootNote`, where its domain is the `Shoot` class and its range is the `Note` class.

2. **Derived**: metadata fields that have derived representation within the ontology do not have a property that directly represents it. Instead it can be derived from one or

more other properties within the ontology that are used to explicitly represent another metadata field. For example, a metadata field representing a list of *open* notes that are attached to the record of a shoot can be derived from the `hasShootNote` object property by filtering all instances of `Note` that are attached to the instance of `Shoot` in question to display only those with a status of 'open' (where the status of a `Note` is represented by the `hasNoteStatus` data type property).

3. **No Representation**: metadata fields that been placed into this category are those that have been identified as not having any form of representation within the ontology. As such, this is a limitation of the Creative Data Ontology in its current form and these fields should be added in future iterations to ensure an optimal level of completeness.

4. **Excluded**: metadata fields in this category have been deliberately excluded from the ontology. The reasons for choosing to omit these fields are described later in this section.

Through the evaluation process, we found that 88.50% of fields were explicitly included within the Creative Data Ontology while a further 2.41% of fields could be derived. In contrast, none of the fields had no representation in the ontology when they should have while 9.09% of fields were deliberately excluded. If those excluded fields are removed from consideration, then all fields (97.35% explicit, 2.65% derived) are included in the ontology.

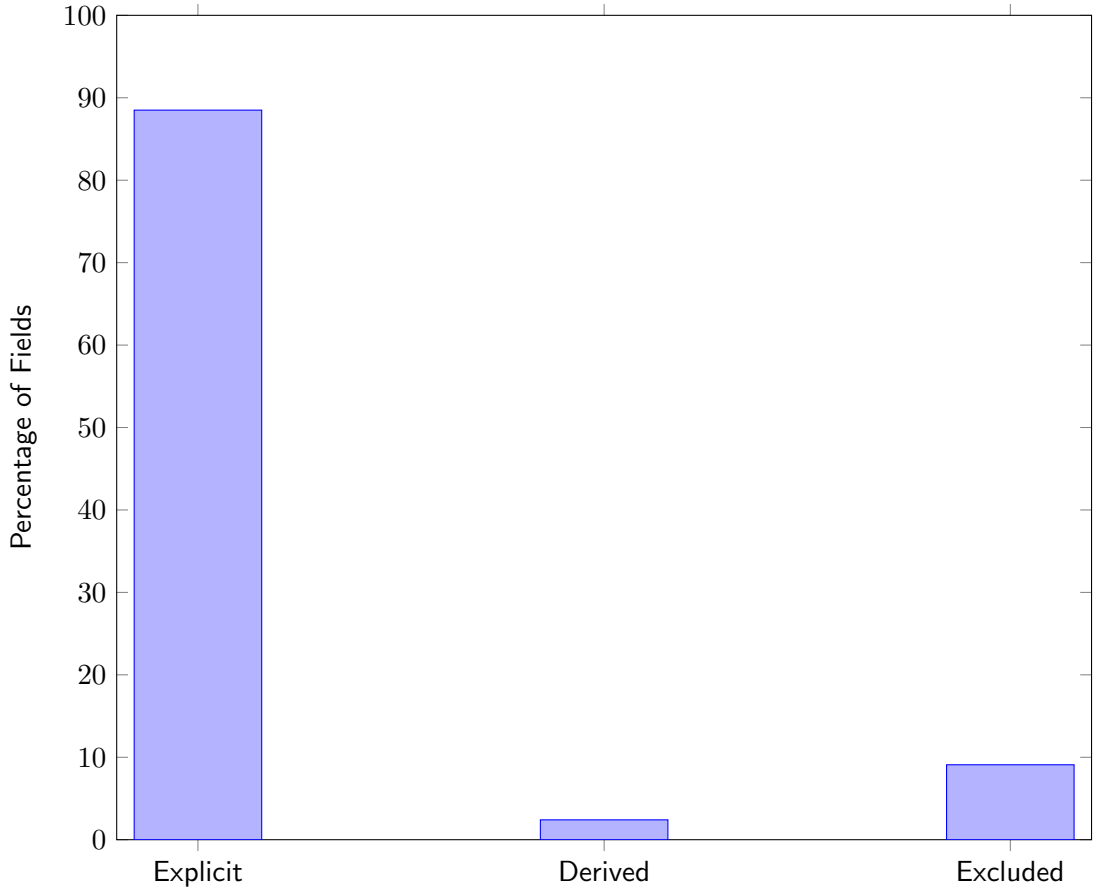


Figure 7.5: Representation of metadata fields in the Creative Data Ontology.

Figure 7.6 shows, in proportion, the reasons for a metadata field in the corpus being excluded from the ontology. These reasons are as follows:

1. **Deprecated**: there were several fields that were identified early in the ontology engineering process as no longer required by the partners.

2. **Generic**: a number of fields in the corpus had generic names, such as *Generic Text 01*. Upon discussion with our partners, we were told that these fields were created on an ad-hoc basis when existing fields within their systems were insufficient. Partners were given the opportunity to identify the specific uses of these fields so that they could be represented in the ontology with a clear, descriptive name. They were also

given opportunities throughout the ontology engineering process to request additional fields to be added in case the information stored in these generic fields had not been included in earlier ontology iterations.

3. **ID Field**: a number of fields in the corpus referred to the ID used to identify records in the partners' existing database systems. For example, each record of a scene would have a unique value in the ID field. These fields were excluded from the ontology, firstly because ontologies use URIs to uniquely identify instances of classes so such fields would be redundant but also because the metadata management tool is expected to handle unique identifiers for objects.

4. **Other**: finally, there were also a small number of fields that had their own unique reason for exclusion. Before these fields were excluded, a discussion was held with the partners to ensure that this would not hinder the completeness of the ontology.

The excluded fields category is made up as follows: 26.47% of fields were omitted as they were identified as deprecated while 29.41% were because they were too generic for inclusion in the ontology. Meanwhile, 26.47% of excluded fields represented ID fields and 17.65% of excluded fields were omitted for other reasons.

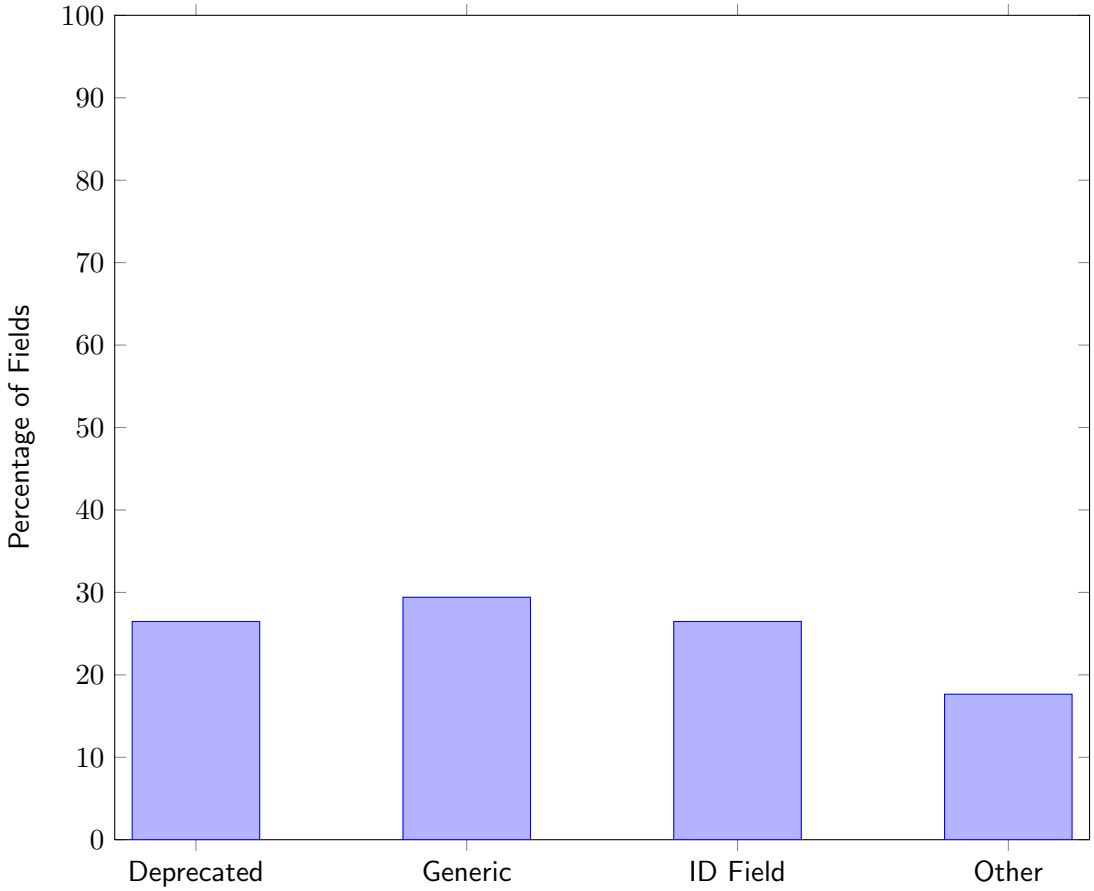


Figure 7.6: Reasons for excluding metadata fields from the Creative Data Ontology.

7.3.3 Conciseness

In addition to seeking feedback from domain and ontology experts to evaluate the Creative Data Ontology for conciseness, it was also submitted to the OntoMetrics tool to obtain statistics for the following five evaluation metrics: attribute richness, inheritance richness, relationship richness, maximum depth of the taxonomy, and class to relation ratio. Table 7.4 shows the statistics generated by OntoMetrics.

Metric	Value
Attribute richness	4.28
Inheritance richness	0.97
Relationship richness	0.72
Maximum Depth	4
Class/Relation Ratio	0.29

Table 7.4: Creative Data Ontology conciseness metrics.

7.4 A Note on Minimal Encoding Bias

During the development of Iteration 5 of the ontology, any instances of encoding bias that would interfere with the ontology's integration with the metadata management tool would have been identified by the software development company commissioned to build and test the tool. As explained in Section 6.2, the ontology compatibility issue that arose between the ontology and the metadata management tool were with the annotation properties used to attach natural language definitions to ontology components. Given that this evaluation was performed on the ontology outputted at the end of Iteration 5, we can reasonably conclude that any relevant encoding bias has been removed.

7.5 Summary

In this chapter, we presented the process taken to evaluate and validate the Creative Data Ontology. We began by selecting the criteria against which the ontology would be evaluated before choosing the most appropriate evaluation methods for each one. Next, we presented the results from the ontology evaluation interviews conducted with domain and ontology experts to assess the accuracy, completeness, conciseness, and clarity of the ontology. This was followed by the results of using quantitative techniques to measure the internal consistency, completeness, and conciseness of the ontology. The evaluation results

are collated in Table 7.5. This chapter was then concluded with a final comment about how we ensured the ontology had minimal encoding bias.

Evaluation Criterion	**Evaluation Results**
Accuracy	Use Cases 1 and 3 are accurately represented in the ontology, according to all four domain experts interviewed. The chain of classes connecting `Shoot` to `Clip` and `Slate` is also accurate but one domain expert, D4, stated that it may not work with unscripted content. Use Case 2 is accurately represented in the ontology according to three domain experts interviewed, but D4 said that an object property was missing between `Scene` and `Clip`. This was omitted from slide deck diagrams but present in the ontology. The class hierarchy is accurate according to all four domain experts but D4 suggested that the name of the `Deliverable` class be pluralised to `Deliverables`. Finally, domain experts were asked give the ontology a score out of 10 for accuracy. The average score given was 7.75.
Clarity	Use Cases 1 and 2 are clearly and intuitively represented in the ontology, according to the three ontology experts interviewed. Two ontology experts, O1 and O2, agreed that the subclasses of the `Asset` class shown in Use Case 3 should be further grouped in order to improve the clarity of the ontology. O3 stated that the representation of this use case was clear and intuitive but could not give a firm answer about further grouping the subclasses of `Asset`, as this would be a question for domain experts.

Clarity (cont.)	O1 and O3 agreed that the representation of Use Case 4 is clear and intuitive. The final ontology expert, O2, agreed that this use case was mostly clear, except for the meaning of the `hasRecordVersion` object property. After receiving clarification, this expert suggested that the property be renamed. Four domain experts were asked to comment on the definitions of components in the ontology. One domain expert, D3, declined to answer while the remaining three agreed that the definitions of components involved in the use cases are clear and accurate. D4 also stated that there are no overlapping definitions while D1 and D2 agreed that the definition of the `Roll` class overlaps with the `CameraRoll` and `SoundRoll` classes. When asked about the clarity of the ontology overall, three domain experts (D1, D2, and D4) agreed that most domain experts would understand the ontology quickly but of those three, D1 and D2 stated that they would need to understand the concept of ontologies in general before attempting to use the Creative Data Ontology. Although this is not an issue with the ontology itself, it was also suggested that diagrams of the ontology should be presented with a top-down layout as it is most intuitive to non-technical people. Finally, domain and ontology experts were asked give the ontology a score out of 10 for clarity. The average score given was 7.00.

Completeness	Use Case 1 is fully represented according to two domain experts interviewed (D2 and D3). D4 agreed it was complete but stated that there are other workflows for media support in the industry. The final domain expert, D1, suggested adding further attributes to the `File` and `Client` classes. Furthermore, although they were not asked about completeness, O1 noticed that the ontology was missing an object property to connect an `Email` to the `HardDriveSnapshot` attached to it. D3 stated that Use Case 2 is fully represented but D1, D2, and D4 mentioned that the number of logging tags is too low. D1 also suggested including additional attributes, such as details of the person making the logging entry, time and date of the logging entry, whether the item being logged is audio or video, and the number used to represented the number of the shoot day. Use Case 3 is fully represented according to D2 and D3. One domain expert, D4, suggested the process of backing up the on-set VFX database should be reflected in the ontology and D1 suggested the addition of witness camera materials as a subclass of `Asset`. When asked about missing classes, all four domain experts interviewed had suggestions for additional classes, all centred around specialising higher level classes. D1, D2, and D4 suggested that there should be more types of `Equipment` and their attachments represented in the ontology. D3 and D4 said that the `Staff` class could have additional subclasses to fully capture all roles in production and post-production. D3 and D4 also suggested that `Deliverable` should have subclasses to account for different versions of a final production. One domain expert, D4, commented that the ontology should have subclasses of `Asset` that represent the paperwork involved in a production.

Completeness (cont.)	All domain experts were asked give the ontology a score out of 10 for completeness. The average score given was 7.50. Through a corpus-based quantitative evaluation approach, we found that 88.50% of fields were explicitly included within the Creative Data Ontology while a further 2.41% of fields could be derived. In contrast, none of the fields had no representation in the ontology when they should have while 9.09% of fields were deliberately excluded. If those excluded fields are removed from consideration, then all of the fields (97.35% explicit, 2.65% derived) are included in the ontology.
Conciseness	Two domain experts, D2 and D3, agreed that there were no unnecessary classes in the ontology, based on a visual inspection of the class hierarchy. D1 questioned why a lens type is represented as a subclass of `Lens` rather than as a data type property. The final domain expert, D4, felt that the `Email` class was out of place as a subclass of `Asset`. All three ontology experts interviewed agreed that the ontological representation of Use Cases 1 and 4 are concise. One ontology expert, O2, agreed that the representation of Use Case 2 was also concise while O1 and O3 commented that the diagram contained unnecessary details that caused the diagram to become cluttered. Similarly, O3 stated that subclasses of `Asset` that are unrelated to VFX should be removed from the diagram for Use Case 3. All domain and ontology experts were also asked give the ontology a score out of 10 for conciseness. The average score given was 8.29.

<table>
<tr><td>Conciseness (cont.)</td><td>Finally, we used a metric-based quantitative evaluation approach to evaluate the ontology for conciseness. The following table presents the results for the five metrics used, which were suggested in [86] and [87]:

<table><tr><th>Metric</th><th>Value</th></tr><tr><td>Attribute richness</td><td>4.28</td></tr><tr><td>Inheritance richness</td><td>0.97</td></tr><tr><td>Relationship richness</td><td>0.72</td></tr><tr><td>Maximum Depth</td><td>4</td></tr><tr><td>Class/Relation Ratio</td><td>0.29</td></tr></table></td></tr>
<tr><td>Consistency</td><td>The OOPS! plugin did not detect any cycles within the class hierarchy and we did not include any disjointness assertions or custom restrictions when designing and building the ontology.</td></tr>
<tr><td>General Feedback</td><td>When asked about the impact of the ontology on their business, D1 said it would “hopefully bring some order to the chaos” while D3 said it would bring consistency in terms of labelling the outputs of a production when the time comes to archive them. One domain expert, D2, believed the ontology could mitigate the problem of information loss and the final domain expert, D4, believed that the ontology could be used to capture knowledge that is otherwise gained from experience.

When asked about how the ontology could be improved, all three ontology experts commented on the diagrammatic representations of the use cases rather than focusing solely on the ontological representation. Domain experts often stated that ontologies are complex in general and could be an issue when presented to non-technical domain experts. The final comment was that the ontology could better handle alternative terminology that refer to the same concept and that ambiguous terms could be clarified in brackets.</td></tr>
</table>

General Feedback (cont.)	When asked about how the ontology could be extended, the following suggestions were made: provide coverage for genre domains (such as comedy and drama), provide coverage for other types of production (such as virtual reality, streaming, and interactive content), and provide coverage for drone footage and other aerial shots. D1 believed that a significant portion of the ontology would be relevant to other sub-domains and D3 suggested that we could have sub-ontologies derived from the main or 'master' ontology for each of the various types of production.

Table 7.5: A summary of evaluation results.

Chapter 8

Discussion

The purpose of this study was to explore whether there is a practical use case for using ontologies to manage metadata in the moving image industry and to determine whether an ontology designed for such a purpose can be used to manage metadata more efficiently to improve workflows. This chapter presents the principal findings from the primary research conducted to determine whether there is a use case for using ontologies in the moving image industry and to evaluate the quality of the proposed Creative Data Ontology. A survey of the literature revealed that similar research into the applications of ontologies for metadata management in the media industry had been previously conducted but was limited in scope. These gaps, alongside the specialisms of our industry partner companies, guided the scope of the ontology. A further literature review into ontology evaluation techniques then provided direction for the evaluation strategy used to assess the Creative Data Ontology. This chapter will first explore the four groups of findings from this research in Sections 8.1, 8.2, 8.3, and 8.4 before discussing the limitations and delimitations of the research process in Section 8.5.

8.1 Theme 1: Viability of an Ontology in the Moving Image Industry

The first group of findings is centred around the feasibility of deploying and using an ontology of metadata in the moving image industry. We have made two findings in relation to this theme:

1. An ontology would have a positive impact on media creation processes.

2. Ontologies are not well understood by domain experts.

8.1.1 Finding 1

An ontology would have a positive impact on media creation processes.

Overall, the feedback on the Creative Data Ontology provided by domain experts was positive. Every domain expert was asked what difference the ontology could make in their organisation and the responses were varied but encouraging. One domain expert summarised it well by saying that it would "hopefully bring some order to the chaos", which was supported by other experts who suggested that it would bring "consistency" and that it would act as a means of tackling the issue of information loss. In essence, there are multiple use cases for which an ontology of (post-)production metadata would be useful, in addition to those use cases formed in conjunction with our industry partners as part of the ontology development process. Additionally, the research carried out by Atkinson et al. (2014, 2015) as part of The Deep Film Access Project supports these use cases as the aim was to "advance the methodology for the collection and management of data and metadata" in production and post-production [7, 55].

8.1.2 Finding 2

Ontologies are not well understood by domain experts.

It is important to acknowledge that while there are several viable use cases for a domain ontology in the industry, ontologies are not well understood by domain experts. During the interviews with domain experts they all indicated that they experienced some difficulty in understanding the concept of an ontology, which ranged from asking for clarification on some aspects of how ontologies work to spending a significant portion of the interview with the interviewer explaining how class specialisation works and the difference between subclass relationships versus object properties.

On one end of this spectrum, there was a domain expert who stated that the ontology itself was clear once the mechanics of ontologies in general were explained. This interviewee did not face much difficulty in understanding ontologies. On the other end was a domain expert who agreed the ontology itself was clear, but only after struggling significantly with understanding how it worked due to having no prior knowledge of ontologies. It was because of this confusion that a video was created that explained the basics of ontologies, which future domain experts were asked to watch before their interview.

This finding is supported by Dimitrova et al. (2008), whose paper acknowledges that most domain experts "lack knowledge modelling skills and find it hard to follow logical notations in OWL". Because the ontology engineering process requires active participation from domain experts by sharing the knowledge that needs to be represented, the ROO tool was developed. ROO allows domain experts to author an ontology in OWL using a controlled natural language called Rabbit [122].

Therefore, while the data suggests that there is a use for ontologies for metadata management in the moving image industry, we must ensure that non-technical domain experts are either shielded from, or thoroughly trained in, the intricacies of ontologies.

8.2 Theme 2: Designing an Ontology of Metadata

The second group of findings is centred around the process of designing and building an ontology of metadata and the lessons learned during this process. We have made three findings in relation to this theme:

1. An ontology of metadata can be created in six main stages:

    1) Identify the use cases to focus on.

    2) Obtain a list of the core metadata fields from the stakeholders.

    3) Design an initial ontology, validate it with stakeholders, and act on feedback.

    4) Obtain sample data from the stakeholders and use to instantiate the ontology.

    5) (optional) Validate the ontology with other domain and ontology experts.

    6) (optional) Extend the ontology by facilitating other use cases.

2. Clear and concise diagrams should be used to present the ontology to domain experts.

3. An ontology of metadata will rely on data type properties to represent metadata fields.

8.2.1 Finding 3

An ontology of moving image creation metadata can be created in six main stages.

It is important to document the process that was undertaken to build the Creative Data Ontology, an ontology of metadata for the moving image industry, to allow the steps to be followed for similar ontology development projects in the future. The process of building this ontology consisted of six main steps:

1. Identify the use cases to focus on. In the case of the Creative Data Ontology, there

was a focus on four use cases, which were decided in collaboration with our industry partners. These are described in Section 6.3.

2. Obtain a list of the core metadata fields from companies that are stakeholders of the ontology being created. For example, when developing the Creative Data Ontology, representatives from each partner company furnished us with a list of the 100-150 most important metadata fields as determined by them. These were then analysed as described in Section 5.4.1.

3. Design an initial ontology using the core metadata fields and then validate the ontology with the stakeholders who are domain and ontology experts and act on any feedback given. This step is repeated as many times as is necessary to ensure the ontology meets the requirements of the stakeholders. For example, this stage of creating the Creative Data Ontology involved several months of back-and-forth between the Creative Data Team and representatives from each partner company, resulting in five iterations of the ontology as described in Section 6.2. During this stage, we found it useful to begin by identifying a concept that represents the right level of abstraction. We then capture the relations and attributes by identifying the metadata fields associated with this concept. For example, `Asset` was chosen as the concept that represents the correct level of abstraction in the Creative Data Ontology. From there, a number of subclasses (such as `Clip` and `Script`) and relations (such as `sceneHasAsset`) were created.

4. Obtain sample data from the ontology stakeholders and use to instantiate the ontology.

5. (optional) Validate the ontology with other domain and ontology experts. This step is optional because although it would be extremely beneficial to obtain further feedback from an outside perspective, the following considerations need to be made:

(a) Experts outside of the original group of stakeholders may not be available to validate the ontology.

(b) The ontology has already been validated by experts so the most critical issues are likely to have been identified already and so the constraints of the budget may mean that further validation is considered to be unnecessary.

As the Creative Data Ontology was designed with the intention of it being open-source, we were able to seek further feedback from ontology experts and other domain experts who were not involved in the development of the ontology. This was done in order to ensure that the ontology could be reused outside of our partner companies and the outcome of this stage can be found in Section 7.2.

6. (optional) In order to widen the scope of the ontology, the final step would be to extend the ontology by facilitating other use cases. This is an optional step because the industry is continuously changing and expanding so it would be impossible to provide full coverage for all possible use cases. Once this step is complete, Steps 4 and 5 should be repeated to ensure that the additions have not compromised the integrity of the original ontology. The Creative Data Ontology was not extended beyond the scope of the four original use cases due to issues surrounding availability of funding. However, this would be an area for future work.

Most ontology engineering methodologies share the same high level steps as Uschold and King's (1995) skeletal methodology, which are: 1) identifying the purpose, 2) building the ontology, 3) evaluation, and 4) documentation [60]. The process of building the Creative Data Ontology mostly followed these same steps, with the exception that the documentation phase was ongoing simultaneously with the other steps. This was to ensure that documentation was accurate and up-to-date at all times and to ensure that vital information was not lost or forgotten.

Comparison with Existing Methodologies

Despite our methodological process for ontology engineering following the same high level steps as most of the ontology engineering methodologies described in Section 4.1, it is necessary to critique those processes for two reasons. Firstly, each methodology has their own nuances that either made them unsuitable for engineering the Creative Data Ontology, or informed the process that was followed. The second reason considers the possibility of future ontology engineering work for another area of the arts. Specifically, we must consider whether existing methodologies are suitable for developing ontologies for a domain where there is not much prior work:

- **The Knowledge Engineering Methodology**: This methodology has similarities with the ontology engineering process we followed but with two key differences:

    1. We did not reuse any existing ontologies so Step 3 of the Knowledge Engineering Methodology did not apply. The purpose of the Creative Data Ontology was to cover subdomains of the moving image industry that had not yet been covered, so this step was not possible.

    2. The evaluation and documentation stage in the Knowledge Engineering Methodology (Step 6) takes place at the end of the process (i.e. after the ontology has been formalised and populated with instances). However, when developing the Creative Data Ontology, the evaluation and documentation stages took place in parallel with other stages. For example, the ontology was created iteratively and incrementally. Each iteration was built on the one before it and the feedback given by our industry partners was used to improve the ontology and form the next iteration. Once the industry partners were satisfied, the ontology was then evaluated by domain and ontology experts who were not involved in the project to enable experts with varied expertise to also provide feedback. We also did not have a specific stage for documenting the ontology because this took place

throughout the ontology engineering process. We chose this approach to ensure that vital information about the concepts in the ontology and about design decisions were not lost over time.

We found that the simplicity of the Knowledge Engineering Methodology compared to others, such as the UPON and NeOn methodologies discussed below, was helpful when working with stakeholders because it enabled us to show a clear direction for the project.

- **The DOGMA Methodology**: Unlike the other ontology engineering methodologies described in the literature, the definition of this methodology was focused on the structure of the ontology being engineered, rather than the process being being followed to engineer it. Ontologies produced using the DOGMA methodology have two layers, the ontology base and the commitment layer, which allow the same ontology to be used in a variety of applications. As the Creative Data Ontology is intended to support multiple subdomains of the moving image industry, the separation of the ontology base from the constraints and domain rules defined in the commitment layer would be useful. For example, one domain expert stated that the class chain first introduced in Section 6.2.3 and later refined in Section 6.2.4 may not work with unscripted content. Therefore, an easy way to define different constraints and domain rules for different applications (in this case, subdomains of the moving image industry) would have been beneficial. However, given the difficulty faced by non-technical domain experts in understanding ontologies in general, introducing an additional layer of complexity where different rules apply in different scenarios would have been counterproductive at that stage.

- **The TOVE Methodology**: This methodology focuses on engineering ontologies based on competency questions that the ontology should be able to answer. The use of competency questions to engineer the Creative Data Ontology was considered

very early on in the project but we ultimately decided that it would be better to design the ontology based on use cases rather than competency questions. This is because each use case could have a number of competency questions. For example, Use Case 2 for live logging in unscripted television projects (see Section 6.3.2) could have competency questions such as:

- Which clips have a logging tag of type x?
- Which clips with a logging tag of type x belong to a scene y?
- Which clips with a logging tag of type x were filmed on a shoot day z?

Due to the volume of metadata fields that we wanted to represent in the ontology, we felt it was too time-consuming to generate a list of competency questions to cover every metadata field in the limited time available to us. Therefore, we instead designed a set of use cases in collaboration with our industry partners.

- **The Methontology Framework**: This methodology consists of two main stages, the ontology development process and the ontology lifecycle, where the ontology development process consists of three categories of activities: ontology management, ontology development, and ontology support. The Methontology Framework has many similarities with the methodology followed in developing the Creative Data Ontology in terms of the activities involved. However, the Methontology Framework is structured in terms of stages and categories of activities, where activities in the same category can take place at different times in the engineering process. For example, ontology support activities include acquiring domain knowledge and evaluating the ontology. The former takes place mostly in the early stages of the process whereas the latter usually takes place near the end. The methodology presented in this thesis differs to the Methontology Framework because it breaks down the activities involved into a set of instructions that can be followed step-by-step.

- **The SENSUS Methodology**: This methodology focuses on using the overarching SENSUS ontology, made up of over 70,000 nodes, to connect two or more domain ontologies. This approach was deemed unsuitable for the Creative Data Ontology because our objective was to create a new ontology of metadata associated with the media creation process, rather than to connect existing domain ontologies together. Another important consideration was that the literature does not explicitly state that there is an evaluation or documentation stage as part of the SENSUS methodology. Evaluation and documentation are both vital stages for any ontology engineering project but it is especially important for projects that are intended for use by non-technical domain experts. The feedback obtained by our industry partners after each iteration of the ontology, and during the evaluation interviews with domain and ontology experts, was invaluable for identifying misrepresentations of the the domain and technical issues with the ontology. Without the ongoing evaluation stage, the Creative Data Ontology would lack important domain knowledge and contain some severe inaccuracies. It is also important to produce comprehensive documentation to ensure the ontology can be understood and used easily by non-technical domain experts and extended in the future by ontology developers.

- **The DILIGENT Methodology**: This methodology was developed to support domain experts to build and evolve ontologies in a distributed setting. An initial ontology is built and released for use and then users can adapt it for their own use cases. Any changes made between each new version is first analysed and approved by a control board before being implemented in the new version and released to users. As discussed in Section 1.1, the current practice in the media industry is to use relational databases, resulting in many databases being created. Therefore, while the DILIGENT methodology did not inform the process followed to develop the Creative Data Ontology, we must acknowledge that its distributed workflow would be invaluable if this ontology were to become commonplace within industry. A distributed

ontology engineering methodology would allow changes to be made to an industry standard master ontology while allowing each company to adjust their local version to their particular needs.

- **The On-To-Knowledge Methodology**: The main feature of this methodology is that it cycles back-and-forth between adjacent steps until the ontology reaches a satisfactory state, before moving on. This is a feature that is emulated in the ontology engineering process presented in this thesis. Specifically, when extensions are being made to the ontology post-evaluation, Steps 4 and 5 (instantiation and validation) must be repeated to ensure the changes have not compromised the integrity of the original version of the ontology. This creates a cycle in the process. Also, unlike the skeletal methodology proposed by Uschold and King (1995), the documentation stage takes place early on in the On-To-Knowledge Methodology. Similarly, the methodology used to produce the Creative Data Ontology had a documentation step that takes place in parallel with the other steps to ensure that vital information was not lost. However, our methodology is slightly different to the On-To-Knowledge Methodology as we did not develop a seed ontology before adding non-hierarchical relations and attributes. Instead, the approach to developing the initial ontology is left to the developer(s)'s discretion.

- **The UPON Methodology**: This methodology presents the ontology engineering process as a set of workflows which take place across a set of phases within an ontology engineering cycle. Each phase is subdivided into iterations and a new release of the ontology is made at the end of a cycle. The aim of this structure was to facilitate an iterative and incremental approach to ontology engineering. The development of the Creative Data Ontology also required an iterative and incremental approach, as we had to liaise with our industry partners to obtain domain-specific information and to get feedback on the ontology at different stages. Despite this, we found the UPON methodology unsuitable for our needs. This is because we needed a clear and relatively

simple process to follow when developing the ontology in order to keep our industry partners engaged. However, presenting the process in terms of workflows, phases, iterations, and cycles would have been unnecessarily complicated for our needs.

- **The NeOn Methodology**: This methodology aims to provide concrete guidelines for reusing and re-engineering ontologies by suggesting pathways for building ontologies in the form of nine scenarios. Seven of these nine scenarios involve reusing existing ontological or non-ontological resources. As these resources were not available when we were developing the Creative Data Ontology, these seven scenarios were not suitable for our needs. The penultimate scenario – Scenario 9 – focuses on translating an ontology into another language, which is not an objective of this ontology engineering project. However, the remaining scenario – Scenario 1 – focuses on developing a new ontology from scratch, which is the aim of this project. The development of the Creative Data Ontology followed all but one of the steps defined as Scenario 1: performing an ontology formalisation activity where the organised knowledge is transformed into a semi-computable model. Instead, the output from the ontology conceptualisation activity was directly implemented in OWL due to time constraints. However, we did not find any evidence that this had a negative impact on the engineering process or the resulting ontology.

In summary, the ontology engineering methodology defined in this thesis is fundamentally the same as the skeletal methodology proposed by Uschold and King (1995) from a high-level perspective except that the evaluation and documentation steps take place in parallel with the other steps. The methodology outlined in this thesis also emulates some features from the Knowledge Engineering Methodology, the Methontology Framework, and the On-To-Knowledge Methodology but does have differences, as explained above.

The Evaluation Stage

In Section 7.1 we defined a framework for evaluating the Creative Data Ontology. This framework was organised in terms of the evaluation criteria described in Section 4.4. However, it is important to also discuss the suitability of the evaluation approaches described in Section 4.3 because the evaluation process is influenced largely by the approaches that are used to assess the ontology against the evaluation criteria. Furthermore, not every evaluation approach will be relevant or even possible for a given ontology engineering project. It is therefore important that we share our experience of evaluating the Creative Data Ontology in order to provide guidance for choosing evaluation approaches for future ontology engineering projects:

- **Gold Standard**: The gold-standard approach requires the availability of a benchmark ontology to compare a candidate ontology against. This can be an effective approach to evaluating new ontologies in domains where there are already existing ontologies, as it allows ontology developers to identify areas in which a candidate ontology is weak. However, it is not possible to evaluate ontologies for domains where there are no preexisting benchmark ontology because a benchmark ontology is needed to compare the candidate ontology against. The Creative Data Ontology models the moving image industry with a focus on film, scripted and unscripted television, and visual effects. As there was no benchmark ontology for the latter two subdomains of the moving image industry domain, we could not use the gold standard evaluation approach.

- **Data-driven or Corpus-based**: This approach requires the availability of a corpus of information about the domain that the ontology attempts to cover. It covers the same evaluation criteria as the gold standard approach – external consistency, completeness, and conciseness – but it is easier to find a corpus that covers the same domain of a candidate ontology than it is to find a benchmark ontology. When evaluating the Creative Data Ontology using a corpus-based approach, we focused on using this

approach to evaluate the ontology's completeness. As part of our collaboration with our industry partners, we were furnished with a set of core metadata fields extracted from their existing metadata management systems, which we used as a corpus to evaluate the Creative Data Ontology against.

- **Criteria-based and Metric-based**: Although metric-based evaluation was differentiated from criteria-based evaluation in Section 4.3, it is important to acknowledge the link between them. Criteria related to the structure of an ontology typically rely on metrics to to quantify them in order for us to evaluate the ontology. Although Section 4.3.3 explains that criteria-based evaluation is best suited to evaluating the clarity of an ontology, in the case of the Creative Data Ontology, this would instead be performed using human inspection by domain and ontology experts. This is because we felt a human inspection approach would elicit feedback on *how* to improve the ontology which is more useful that the aggregate statistics yielded by a criteria/metric-based approach, which only tell us *what* to improve.

- **Task-based**: This approach focuses on using the ontology for completing tasks within an application. Usually, such an approach for evaluating a domain ontology would not be suitable because the evaluation would only be relevant for that particular application only. As such, a new task-based evaluation would need to take place whenever the ontology is used in another application. However, given that the Creative Data Ontology is designed to be integrated into a metadata management tool with four main use cases in mind, task-based evaluation is an appropriate approach to evaluating it. However, this approach is reliant on the availability of users within the domain to perform user testing and so it may not always be possible to use this approach.

- **Evolution-based**: This approach focuses on the evolution of an ontology as new knowledge is added. Although the Creative Data Ontology went through several it-

erations before being available for use in the metadata management tool, evolution-based evaluation is not suitable because only one iteration of the ontology has been released for use within the tool. Evolution-based evaluation would be more suitable once the metadata tool is complete and new knowledge needs to be added to the ontology. Although it was not possible to use an evolution-based approach to evaluating the Creative Data Ontology, we would have taken such an approach if new knowledge had needed to be added. Therefore, it is advisable that similar ontology engineering projects in the future use an evolution-based approach as part of its evaluation framework.

- **Logical or Rule-based**: The logical evaluation approach is very important because it focuses on identifying areas in a candidate ontology where there are logical inconsistencies. Logical inconsistencies can have serious adverse effects on the intended meanings of the concepts involved in the inconsistencies and of concepts in the wider ontology. As the logical approach focuses on using automated reasoner tools to identify inconsistencies, it can be done so quickly and with a high level of accuracy while requiring minimal human input.

- **Human Inspection**: Some criteria for assessing the quality of an ontology, such as completeness, cannot be measured in a fully automated way. When assessing whether an ontology is complete in terms of covering the entire domain, we can combine corpus-based evaluation with the human inspection approach. Given that this project had access to creative practitioners from the media industry, the Creative Data Ontology benefited immensely from being manually checked by domain experts.

8.2.2 Finding 4

Clear and concise diagrams should be used to present the ontology to domain experts.

During both the discussions with representatives from our partners companies and the

evaluation interviews with other domain experts, we found that clear and concise diagrams should be used to present the ontology to non-technical audiences. In order to maximise clarity and conciseness, diagrams should therefore:

- Only focus on specific slices of the ontology, such as a single use case, rather than attempt to show the entire ontology in one diagram. As the ontology becomes larger and more complex, so will any diagram trying to show the ontology in its entirety. In the case of the Creative Data Ontology, a diagram of the whole ontology after Iteration 1 was several metres long, which was unreadable. Therefore, the scope of a diagram should be narrowed down.

- Eliminate components that are unnecessary for the scope of the diagram. As we have established, each diagram has limited space available before it becomes overcrowded and difficult to understand. Any ontology components included in the diagram must serve a purpose and any that are included for context should be done so sparingly. This was raised by one expert who pointed out during an evaluation interview that the diagram for Use Case 2 (shown in Figure 8.1) included the `Shoot` and `ShootDay` classes which "adds detail that maybe it's getting in the way where you could have other details about the post production staff". Alqadah et al. (2016) states that diagrams with a lot of information in them, coupled with choices in layout, can cause visual clutter. This is described as a "barrier to cognition" [123]. However, it is important to note that contextual information may be necessary depending on the audience, so thought should be given to what information the audience already has and what information they now need. Taking these opposing views into consideration, we must acknowledge that there is a large amount of subjectivity involved in determining whether a diagram appears cluttered [124].

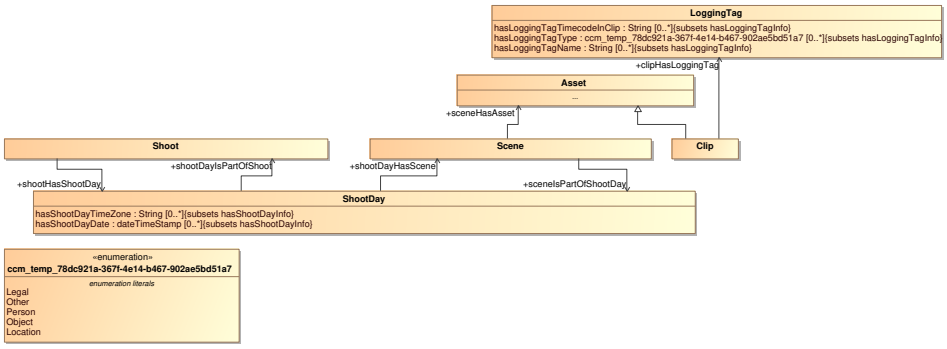


Figure 8.1: Diagram representing Use Case 2 as shown to experts.

- Present the hierarchy of the ontology in a top-down layout. During the evaluation interviews, it was pointed out by participants that it was difficult to interpret some of the more complex use case diagrams, such as the one for Use Case 3 (shown in Figure 8.2). This is particularly important in situations where the use case spans across multiple levels of the ontology's hierarchy. The advice to follow a top-down layout is also given by Burch et al. (2020), who state that top-down hierarchical diagrams are "perceptually effective". However, it is also the case that top-down hierarchical diagrams typically waste display space at the lower levels of the hierarchy due to the increasing number of classes situated at those levels [125].

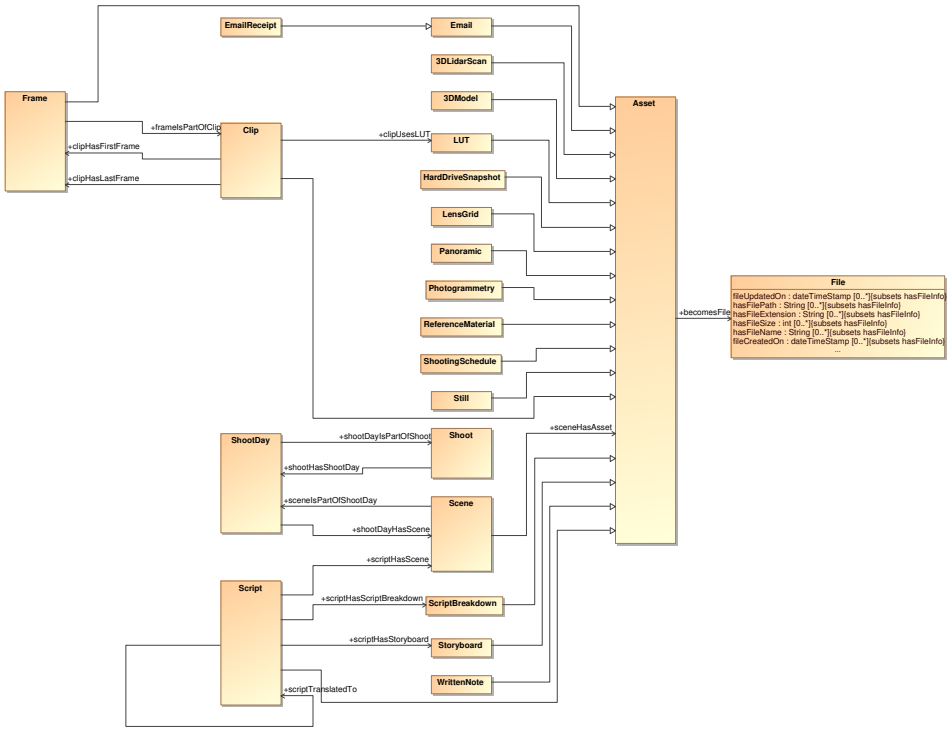


Figure 8.2: Diagram representing Use Case 3 as shown to experts.

8.2.3 Finding 5

An ontology of metadata will rely on data type properties to represent metadata fields.

During the construction of the ontology and in the quantitative evaluation stage, we have found that an ontology of metadata will rely heavily on the use of data type properties to represent metadata fields. The construction phase saw us consider whether to use classes or data type properties to represent metadata fields with either instances or literals used to represent individual values of metadata fields. The decision to use data type properties over classes was made in order to follow an object-oriented design, wherein each instance belongs to a set of objects and each object has a set of metadata attached to it. For example, the ontology has a class, `Clip`, which has over 60 metadata fields associated with it in the form of data type properties. In addition, having individual classes for each

metadata field will result in a disproportionate number of classes in the ontology due to the high number of metadata fields we needed to represent. The results of the quantitative evaluation phase presented in Section 7.3 reinforces this because the ontology was found to have an attribute richness of 4.28 (i.e. each class has an average of 4.28 data type properties each).

8.3 Theme 3: Quality of the Ontology

The third group of findings is focuses on the quality of the Creative Data Ontology based on the set of evaluation criteria identified in Section 7.1. We have made five findings in relation to the quality of the ontology, which are that the ontology is:

1. accurate, based on a non-exhaustive inspection of the use cases.
2. mostly complete but needs minor improvements.
3. clear but needs minor improvements.
4. sufficiently concise.
5. logically consistent.

These are discussed in order in Sections 8.3.1 to 8.3.5.

8.3.1 Finding 6

Based on a non-exhaustive inspection of the use cases, the Creative Data Ontology is accurate.

There was only one minor suggestion made about the accuracy of the ontology, which was to pluralise the name of the `Deliverable` class because "you rarely only deliver one file". However, due to the naming conventions adopted in the ontology, it was decided that the `Deliverable` class should remain singular in order to remain consistent with the

names of other classes. Using the singular form for class names is also recommended by Montiel-Ponsoda et al. (2011) in [126], which outlines a set of style guidelines for naming and labelling ontologies.

Besides this suggestion, the consensus was that the ontology was accurate and there were no suggestions for improvement given in terms of the accuracy of the three use cases presented to the domain experts. This is reinforced by the accuracy rating given by domain experts, where the average rating was 7.75/10 with only one score lower than 8. However, due to the size of the ontology, it was impossible to evaluate it in its entirety and so the process of evaluating it for accuracy was focused on three use cases, the class hierarchy, and the chaining of the classes described in Section 6.2. Therefore, it is possible that some accuracy issues may exist in the ontology that have not yet been detected but the impact of this has been minimised by ensuring the evaluation concentrated on the most important parts.

8.3.2 Finding 7

The Creative Data Ontology is mostly complete but needs minor improvements.

The questions requesting feedback about the completeness of the ontology yielded the largest amount of qualitative feedback out of all of the evaluation criteria. The feedback received ranged from requiring minor additions to requiring major additions. For example, the missing object property connecting the `Email` and `HardDriveSnapshot` classes in Use Case 1 is an important issue to resolve but requires only minimal effort to do so. Likewise, the suggestion to add more attributes to the `File`, `Client`, and `LoggingTag` classes would only result in minor changes to the ontology and thus is trivial to implement. On the other hand, there were some suggestions that would take longer to implement, as they would require further research into the media creation domain. Such changes include the addition of new subclasses to the `Asset`, `Staff`, `Equipment`, and `EquipmentAttachment` classes to ensure that all asset types, roles, and equipment used within the industry are captured.

However, we must note that while this would definitely make the ontology more complete, it could negatively impact on the conciseness of the ontology so we must ensure there is a balance between completeness and conciseness based on the intended scope of the ontology.

There were some suggestions made that were beyond the original scope of the ontology, but would be a viable focus for future extensions. For example, it was indicated that while the workflow presented in Use Case 1 is complete, there would be other workflows in industry that have the same purpose of providing media support, which should be represented in the ontology. These workflows were not in place at our partner companies so implementing them in the ontology would require further research, most likely in collaboration with other post-production companies. Additionally, it was pointed out that the process of backing up the on-set VFX database described in Use Case 3 should be reflected in the ontology. However, this would again require further research into the specific details of the processes taken by production and post-production companies. Therefore, these changes were beyond the intended scope of the ontology but would enhance the ontology in future iterations.

There were also some components of the ontology that were highlighted as missing by the interviewees that were actually in the ontology but not shown in the segment diagrams due to lack of space. These components were the connection of `Scene` and `Clip` in Use Case 2 and, to a small extent, logging tag types. It was pointed out several times that in order for a `Scene` to be linked to a `Clip` and its associated `LoggingTag`, there needs to be an object property or a chain of object properties between `Scene` and `Clip`. However, due to the chaining of classes described in Section 6.2, a `Scene` is connected to `Clip` via the `Shot` and `Take` classes, which was not shown in the use case diagram. There were also several suggestions about the logging tag types enumeration, all revolving around the view that there needed to be more. Two interviewees stated that there needed to be more tag types with one suggesting "emotions" and "action" while the third stated that when they worked in logging, they "were able to set our own titles and we had many more because

we would have individual people names". The issue about needing more tag types is a fair and accurate one and so "emotions" and "action" will be added to the enumeration in future iterations. However, each instance of `LoggingTag` (which is unique to the `Clip` being tagged) has an attribute for the tag name, which would allow users to set their own titles.

When considering whether or not the ontology is complete, we must first acknowledge that there are two different types of suggestion. The first is *improvements*, which are changes that will enhance the ontology to ensure that the use cases are better represented while keeping within the original scope. The second type of suggestion is *extensions*, which are additions made to expand the ontology beyond its original scope. The quantitative data would suggest that the ontology is complete in terms of the original scope because 100% of fields provided by the partners were included either explicitly or implicitly. However, the qualitative data suggests that there are a few minor improvements that are still required before the ontology can be considered 'complete', such as adding an object property in Use Case 1 to connect an `Email` to a `HardDriveSnapshot`. Although the subjectivity of qualitative data could arguably be the cause of these oversights, they are instead likely to be due to minor changes in scope as the ontology development progressed. Therefore, the ontology can be considered complete in terms of the original scope but the suggested improvements must still be made. There were many possible extensions suggested but those would not need to be implemented for the ontology to be considered complete with respect to the original scope.

The following list is of all the improvements that should be made to the ontology before considering it complete:

- Add data type properties to the `File` class to represent a file's checksum and access permissions.
- Add data type properties to the `Client` class to represent a client's email address,

postal address, and client account number.

- Add the missing object property between `Email` and `HardDriveSnapshot` to represent the snapshot being sent as an email attachment.
- Add 'emotion' and 'action' to the enumeration used as the range of the `hasLoggingTagType` data type property.
- Add an object property connecting a `LoggingTag` to a `Logger` and the date and time that a link was created between an instance of `Clip` and `LoggingTag` and an instance of `Logger` and `LoggingTag`.
- Add a data type property stating whether an item being logged by a `LoggingTag` is audio or video.
- Add a data type property to `ShootDay` to represent the number, n, used to indicate the number of the shoot day.
- Add subclasses of `Asset` to represent the paperwork involved in a production, such as budgets.
- Add a data type property to the `Deliverable` class to represent its description. This attribute will be used to describe the differences in deliverable versions (e.g. 'Spanish version' or 'scenes of a sexual nature removed').

Meanwhile, this list is of all the extensions that could be made to the ontology in order to expand it beyond its original scope:

- Add, for the first use case, representation for other media support workflows used in the industry.
- Add, for the third use case, representation for the process of backing up the on-set VFX database.

- Add further subclasses to the `Staff` class to capture all roles in production and post-production.
- Add further subclasses to the `Equipment` and `EquipmentAttachment` classes to capture all equipment used in a production.
- Add support for witness camera materials as a subclass of `Asset`.

Completeness obtained an average rating of 7.50/10 by domain experts with only one score falling below 7. The expert who gave this score made several of the above suggestions for extensions to the ontology, so it is possible that their score was partially influenced by an overestimation of the original scope. Regardless, the improvements suggested above must be implemented before the ontology can be considered complete based on the original scope, and the suggestion for extensions would enhance the ontology significantly so they should be considered for future work.

8.3.3 Finding 8

The Creative Data Ontology is clear but needs minor improvements.

As discussed in Section 8.1.2, all domain experts indicated that they experienced some difficulty in understanding the concept of an ontology during their interview. This section will discuss the clarity of the Creative Data Ontology, but this is an important consideration because the feedback given by domain experts with respect to the clarity of the ontology may have been affected by this unfamiliarity with ontologies in general. For example, three domain experts agreed that most domain experts would understand the ontology quickly, but two of those three stated this was only if the domain expert already understood how ontologies work. One domain expert declined to comment on whether domain experts would understand the ontology quickly.

Domain experts were also asked two questions about the clarity of some of the natural

language definitions in the ontology. The same domain expert declined to answer these questions too. All three domain experts who did answer agreed that the definitions of the use case components were clear and accurate. However, one did point out that there was an overlap between the definitions of `Roll`, `CameraRoll`, and `SoundRoll` and another agreed in a later interview. It is important to note that `Roll` is a superclass of `CameraRoll` and `SoundRoll` so an overlap between `Roll` and each of its two subclasses are to be expected, due to the nature of specialisation relationships.

From the point of view of an ontology expert, the ontology itself is mainly clear and intuitive. All three ontology experts agreed that Use Cases 1 and 2 were clearly represented in the ontology. One ontology expert did recommend presenting the diagrams with a top-down structure which, although it does not highlight any issues with the ontological representation of the ontology, is important to ensure that non-technical domain experts can interpret the diagrams with ease. This is necessary because non-technical people are often opposed to technological change in the workplace. Haymes (2008) suggests a so-called 'Three-E' strategy for overcoming resistance to technological change [127]:

1. **Evident**: There are three parts to ensuring that the new technology is evident. The first is that users must be made aware of new technology and then second, their expectations must be set appropriately in terms of the costs versus benefits of the change. Finally, the technology must be marketed in a way that would "appeal to harried and overstretched" staff.

2. **Easy to Use**: The new technology must be "easy and intuitive to use" for a wide range of audiences, in particular non-technical people, otherwise they will not use it.

3. **Essential**: The new technology must become essential to staff in completing their workflows.

We have established in Section 8.1.2 that non-technical domain experts had significant

difficulties in understanding how ontologies worked, which is in contradiction with the second E (easy to use). As a result, it places emphasis on the importance of ensuring that diagrams presented to potential users are clear and easy to interpret. Otherwise, there is an increased chance of resistance towards using an ontology of metadata in real-world settings.

It was agreed by all three ontology experts that Use Case 3 was clear and intuitive but two also agreed that the subclasses of `Asset` should be grouped further. `Asset` currently has 17 subclasses plus one sub-subclass (subclass of a subclass). These 18 classes are heterogeneous and so the ontology could be better organised by separating them into two groups: `MediaAssets` and `TextAssets`. This is especially true because one domain expert said that there should be additional subclasses for other production paperwork, such as budgets. The inclusion of these classes would result in further increasing the flatness of the class hierarchy. Use Case 4 was described as the most clear use case by one ontology expert and the other agreed but also stated that the `hasRecordVersion` property should be renamed to better describe its purpose and to avoid confusion over the way metadata versioning has been implemented.

Overall, the two most important findings with respect to clarity is that for a non-technical domain expert to understand the Creative Data Ontology, they would first need to be familiar with ontologies as a concept and then the diagrammatic representations of the ontology would need to make the hierarchy clearer. The evidence suggests that the ontology itself is mostly clear but the following suggestions would need to be implemented before any real-world use:

- Group the subclasses of `Asset` into `MediaAssets` and `TextAssets`.
- Rename the `hasRecordVersion` object property to `hasPreviousRecordVersion`.

Clarity obtained an average rating of 7.00/10 by domain and ontology experts with two scores falling below 7. The lowest score was 3, which meant that clarity received the lowest

average score out of all the criteria, so it is an area for further thought. There are three likely causes for a low score being given:

1. The ontology itself is unclear or unintuitive in places. This can be overcome by implementing the feedback provided by domain and ontology experts during the interview process. However, very little feedback was focused on the ontology so further intervention would be needed before deploying this ontology into industry.

2. The diagrams of the ontology are unclear or unintuitive. This can also be resolved by following the feedback provided by domain and ontology experts. This includes restructuring the diagrams to follow a top-down layout and ensuring only necessary ontology components are included to avoid clutter.

3. Ontology-based technologies are complex or unfamiliar to non-technical users. This is the most likely cause of the low score as all domain experts raised this as a problem during their interviews. Therefore, while the ontology itself is mostly clear and intuitive, the end users are likely to have little to no experience of ontologies so they would need to undertake significant training before deploying this or any ontology in industry.

8.3.4 Finding 9

The Creative Data Ontology is sufficiently concise.

All ontology experts agreed, based on the ontological representation of the use cases, that the ontology is concise. The only suggestion given was to ensure the diagrams of the use cases do not contain unnecessary details that cause it to become cluttered. As stated above, it is important to ensure that non-technical domain experts can interpret the diagrams of the ontology easily, so this is something to consider but is not representative of the ontology itself.

On the other hand, domain experts were asked for comments on the conciseness of the ontology in terms of whether there were any unnecessary classes in the class hierarchy. One domain expert asked why specific lens types are represented as subclasses of `Lens` rather than as a data type property. In Section 2.4.4, we discussed that ontology developers are often required to make decisions in respect of how to represent concepts. Specifically, we discussed the example of the BioPAX ontology and whether a concept should be represented as a class or as an instance [31]. The feedback given about lens types refers to representation as a class or as a data type property (rather than an instance) which, although different to the example given for the BioPAX ontology, the point around having to make design decisions still applies. Therefore, to ensure maximum flexibility, we chose to give each lens type its own separate class. However, we do acknowledge that as the list of lens types grow, the ontology will no longer be as concise and will result in a flatter hierarchy.

The final suggestion – that the `Email` class was out of place as a subclass of `Asset` was highlighted by only one domain expert. The domain expert could not say with certainty that `Email` definitely should not be a subclass of `Asset` as they concluded their suggestion by saying "maybe it is right". Given that we have already established that we do need a class to represent emails in the ontology, such as for Use Case 1, we only need to consider whether it is better placed elsewhere. However, no other domain experts raised this issue and the `Asset` class was defined during the engineering stage as:

> *"Any data (not metadata) that can be used in the pre-, post-, and production processes. These are usually stored as a `File` on the file system."*

As an email is form of data used for reference during the pre- to post-production processes and is stored as a file on a file system rather than in a database of metadata, it fits the definition of an `Asset` and is therefore in the correct place in the ontology.

According to the qualitative data obtained through interview, the Creative Data Ontology is sufficiently concise. This is reinforced by the conciseness rating given by both domain

and ontology experts where the average rating was 8.29/10, with no score lower than 7. However, given the subjective nature of requesting a rating, additional (quantitative) evaluation methods were used to confirm that the ontology was adequately concise. Most of the scores returned by the OntoMetrics tool were low, indicating a good level of conciseness while ensuring the ontology contains the core metadata fields provided by our industry partners. Meanwhile, the score returned for attribute richness is high, which is due to most metadata fields being represented in the ontology as a data type property. Given that one of the objectives of the Creative Data Ontology is to organise metadata that is produced during the various stages of moving image creation, this is to be expected. Therefore, no alterations to the ontology are required in order to improve its conciseness.

8.3.5 Finding 10

The Creative Data Ontology is logically consistent.

Logical (or internal) consistency of an ontology can be automatically assessed using reasoner tools. The Creative Data Ontology was evaluated for consistency before each iteration was presented to representatives from our industry partner companies in order to ensure that the feedback we received would be focused on issues that could only be detected by manual inspection. We used the OOPS! plugin for Protégé to check the ontology schema, which did not detect any cycles, and there were no disjointness assertions or custom restrictions within the ontology so there was no possibility for these to cause inconsistencies. Therefore, we can conclude that the ontology is logically consistent.

8.4 Theme 4: Possible Extensions to the Ontology

The fourth and final theme is centred around the possible extensions to the Creative Data Ontology that were suggested by experts during the ontology evaluation interview process. There was one overarching suggestion for ontology extensions:

1. The ontology could be expanded to cover other areas of the media industry.

8.4.1 Finding 11

The ontology could be expanded to cover other areas of the media industry.

When asked for areas in which to expand the ontology, domain experts suggested genre domains (e.g. comedy, drama, natural history, and reality), other types of production (e.g. virtual reality and streaming), interactive content, and types of footage (e.g. aerial shots and drone footage). During the construction of the ontology, we found that there was significant overlap between classes and properties required to conceptualise the sub-domains that the ontology was originally focused on. One domain expert also stated this, saying that "probably 90% of it will already work for [virtual production or immersive subdomains]". However, although the ontology in its current form could hypothetically provide coverage for each of these suggestions, each one has its own nuances that would need to be researched before we included them within the scope of the ontology.

8.5 Limitations and Delimitations

Theofanidis and Fountouki (2019) define limitations in the research process as "potential weaknesses that are usually out of the researcher's control, and are closely associated with the chosen research design, statistical model constraints, funding constraints, or other factors". Meanwhile, they define delimitations as "the limitations consciously set by the authors themselves. They are concerned with the definitions that the researchers decide to set as the boundaries or limits of their work so that the study's aims and objectives do not become impossible to achieve." [128] During the course of this study, three limitations and one delimitation were identified.

Limitation: Evaluation of Large Ontologies

Due to the size of the ontology, it was impossible to evaluate it in its entirety. A diagram containing all classes, object properties, and data type properties in the ontology was created but it was too large and too complex to be interpretable by the interviewees. It is also possible that interviewees would become overwhelmed with information if they were asked to examine a diagram of the whole ontology. Instead, interviewees were shown either three or four smaller use case diagrams (depending on whether they were a domain or ontology expert), which enabled them to give detailed feedback on the most important parts of the ontology. While this means that errors or omissions in the ontology that exist outside of these use cases may be overlooked, the cycle of ontology development and feedback used when collaborating with our industry partners should mitigate this issue, as any major problems are likely to have been discovered then.

Limitation: Interviewee Expertise

The ontology evaluation phase required access to a number of domain experts to be interviewed. However, due to the heavy workload of many industry professionals which is likely to have been exasperated by the COVID-19 pandemic, it was difficult to find industry professionals who could commit time to being interviewed. Additionally, some domain experts who were able and willing to be interviewed did not have a professional background in all three moving image sub-domains that the ontology focused on (scripted film and television, unscripted television, and VFX). This would sometimes result in a domain expert giving feedback on sub-domains outside of their expertise. For example, one interview was conducted with a film and television production expert who also gave feedback on the VFX use case. Although this was not ideal, the alternative option would have been to only interview domain experts on the use cases that they specialise in. However, even workflows and terminology within the same sub-domain are sometimes different between companies and given the shortage of available participants, the decision was made that it is better to get as much feedback as possible.

Limitation: Interviewee Sample Size

As mentioned above, there were very few domain experts willing to be interviewed about the ontology, which would result in a low sample size. This means that issues with the representation of the use cases within the ontology may not have been found. However, while a larger sample size would be more thorough, Guest et al. (2006) conducted a study which states that 6-12 qualitative interviews are enough to reach 70% to 92% saturation, where saturation is defined as the point when new data produces little or no new information [129]. Therefore, we aimed to conduct at least six interviews overall and opened up the interviews to ontology experts to expand the potential participant base. Although we knew ontology experts could not comment on the accuracy and completeness of the ontology, they could still give feedback on the conciseness and clarity of the ontology from a technical perspective.

Delimitation: Ontology Scope

The media creation domain consists of a large number of sub-domains, including scripted film, scripted television, unscripted television, VFX, animation, immersive, and virtual production. In order to keep the scope of this ontology manageable and realistic and due to the specialisms of our industry partners, the decision was made to concentrate the ontology on moving image creation with a specific focus on scripted film and television, unscripted television, and VFX only. More specifically, the ontology mainly focused on the production and post-production stages of the media creation workflow with pre-production being a secondary focus. Figure 8.3 illustrates the scope of the Creative Data Ontology.

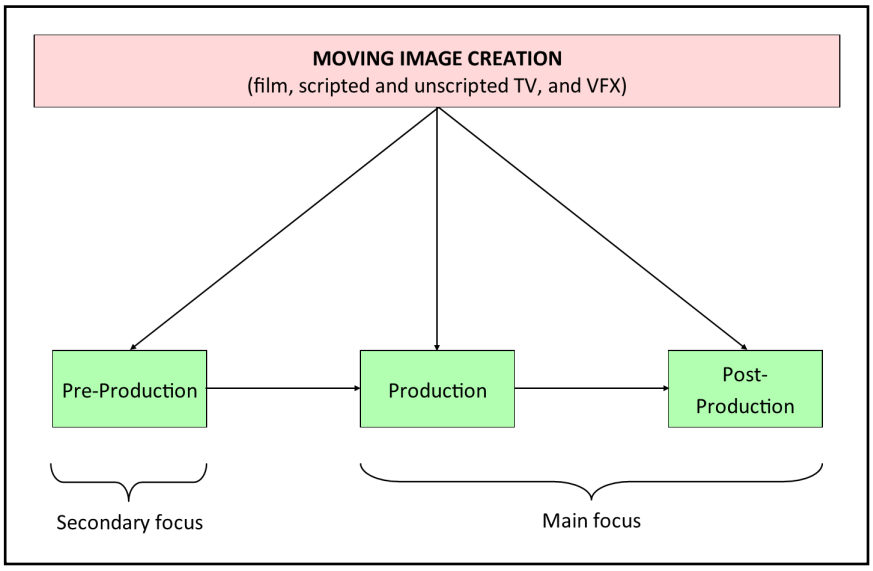


Figure 8.3: The scope of the Creative Data Ontology.

Although the decision to focus on these sub-domains meant excluding others, it did mean that we could dedicate resources to ensure that the included sub-domains were fully understood. Alternatively, we could have sought out additional partner companies with other specialisms, such as animation and virtual production, however this would then mean collaborating with a larger team of domain experts, which would be difficult in terms of scheduling meetings and due to budgetary constraints. During Phase 1 of data analysis, we noticed that there was significant overlap in some of the metadata fields in the scripted film and television, unscripted television, and VFX sub-domains so if there is need to extend the ontology to cover other sub-domains in future research, the impact on the ontology in its current form would be minimal.

8.6 Summary

In this chapter, we have answered our research questions by first exploring the feasibility of implementing an ontology-based solution to metadata management in the moving image industry. We have found that there was a use case for the ontology but also that there

would be a steep learning curve for most domain experts and other non-technical users. Next we examined and documented the process of developing an ontology for the moving image industry and any lessons learned from doing so. We then addressed the quality of the ontology for use in an ontology-based metadata management tool by using data obtained through both quantitative and qualitative methods. We also discussed improvements needed before the ontology could be integrated into such a tool, which have since been implemented following the evaluation process. Table 8.1 presents a mapping of each research question to the relevant findings.

Research Question	**Findings**
1	1, 2
2.1	3, 4, 5
2.2	6, 7, 8, 9, 10
2.3	11

Table 8.1: A mapping of research questions to findings contained in this thesis.

Finally, we discussed the limitations and delimitations of this study, which involved difficulties in evaluating large ontologies in full, the unavailability of interviewees with the required expertise influencing the sample size and feedback given, and the scope of the ontology allowing us to focus our research direction.

Chapter 9

Conclusion and Future Work

The work in this thesis was motivated by clear issues surrounding the management of metadata within the moving image industry. In this thesis, we described the process of obtaining a dataset of metadata fields currently used in the databases of our industry partner companies. Using these fields and a collaborative, iterative, and incremental development process, we designed and formalised an ontology of metadata fields – dubbed The Creative Data Ontology – using Protégé and OWL. This ontology was then evaluated through a set of interviews with domain and ontology experts and through a series of quantitative tests. This chapter will provide some concluding remarks regarding the contributions made during this thesis in Section 9.1, discuss possible impacts of this thesis in Section 9.2, and will outline some possible directions for future work in Section 9.3, which will build on the research conducted as part of this thesis.

9.1 Contributions

Research Question 1: How can ontologies be used to manage metadata in the moving image industry? Is there a practical use case and in what sense?

This thesis has argued that there is a viable use case for a domain ontology for metadata management in the moving image industry. In Chapter 3, we presented a set of five ontologies designed for managing media-related metadata and through this we identified two categories: ontologies that focus only on the finished media product and those that focus on the the metadata produced during the creation process. The research conducted into the two ontologies in the latter category suggest that other researchers have already identified a use case for such an ontology. In addition, during the ontology design phase, we collaborated with three post-production companies based in London, whose practitioners support the notion that their current methods and tools for metadata management are no longer working. This was supported during the ontology evaluation interview process, where all four domain experts interviewed supported the idea of using an ontology to manage their metadata or, as one domain expert put it, "bring some order to the chaos".

However, the interview process also revealed that our target users are unlikely to have any prior experience with ontologies and will therefore struggle to understand how they work. All four domain experts expressed that they experienced some level of difficulty understanding the Creative Data Ontology during the interview due to a lack of technical expertise in this area. These difficulties ranged from requiring minor explanations to the interviewer spending the majority of the interview explaining how ontologies worked. Therefore, while this thesis has established that there is a viable use case for a domain ontology in the moving image industry, domain experts will either require significant training in how an ontology works or will need to be shielded from the technical intricacies.

Research Question 2.1: What would the concrete design of such an ontology look like? What are the stages involved and the lessons learned in the process of developing such an ontology?

In Chapter 4, we presented nine ontology engineering methodologies, of which most contain the same high level steps defined in Uschold and King's (1995) skeletal methodology. These

steps are 1) identify the purpose, 2) build the ontology, 3) evaluation, and 4) documentation [60]. The process of engineering the Creative Data Ontology follows these four steps, except that the documentation phase was ongoing simultaneously with the other steps to ensure that documentation was accurate and up-to-date at all times and that vital information was not lost or forgotten. The process of creating this ontology is outlined below and described further in Section 8.2.1:

1. Identify the use cases to focus on.
2. Obtain a list of the core metadata fields from the stakeholders.
3. Design an initial ontology, validate it with stakeholders, and act on feedback.
4. Obtain sample data from the stakeholders and use to instantiate the ontology.
5. (optional) Validate the ontology with other domain and ontology experts.
6. (optional) Extend the ontology by facilitating other use cases.

We also found that an ontology of metadata would benefit from an object-oriented design, where each class represents an object with a series of data type properties associated with it to represent attributes (i.e. metadata fields). Such an ontology will therefore rely heavily on the use of data type properties.

The final lesson learned by following this process was that non-technical domain experts rely heavily on the use of diagrams in order to understand the structure and inner workings of the ontology. As a result, diagrams should be clear and concise, following the guidance described in Section 8.2.2 and outlined below:

- Only focus on specific slices of the ontology rather than attempt to show the entire ontology in one diagram.
- Eliminate components that are unnecessary for the scope of the diagram, although

context is sometimes needed.

- Present the hierarchy of the ontology in a top-down layout.

Research Question 2.2: What are the quality characteristics of such an ontology? To what extent does it meet the criteria for a high-quality ontology?

In Chapter 6, we presented the Creative Data Ontology, designed in collaboration with three post-production companies based in London. This ontology aimed to encapsulate the workflows employed in these companies and provide a means of improving the management of the metadata associated with these workflows. In Chapter 7, we developed a framework for evaluating the quality of the ontology, which mainly focused on obtaining qualitative feedback by conducting interviews with domain and ontology experts, but also used quantitative methods to assess some quality criteria. Various literature surrounding ontology evaluation defines a total of nine quality criteria against which to evaluate an ontology: internal (or logical) consistency, external consistency (or accuracy), completeness, clarity, conciseness, expandability, adaptability, computational efficiency, and minimal encoding bias. During the evaluation process, the Creative Data Ontology was evaluated against five of these criteria:

1. **Accuracy**: Based on an inspection of the use cases carried out by domain experts, we concluded that the ontology is accurate. However, it is also important to acknowledge that the ontology was too large to evaluate in its entirety so it is possible that errors or omissions that did not affect the accuracy of the use cases may still be present. This was mitigated during ontology development as representatives from our industry partners were consulted for feedback between iterations of the ontology, so any significant problems would have been noticed then.

2. **Completeness**: During the evaluation process, we found that the ontology is mostly complete but needs minor improvements. These improvements have since been made

in a final iteration of the ontology – Iteration 6 – the details of which can be found in Appendix A. Again, as the ontology was too large to evaluate in full, the interview process was focused on the use cases and could form a limitation with the evaluation process. However, during the knowledge acquisition phase, our industry partners provided a set of metadata fields to use a corpus of knowledge to build the ontology against. During quantitative evaluation, each field was checked for representation within the ontology and we found that they were all included in the ontology.

3. **Clarity**: The evaluation process found that the ontology is mostly clear, excluding the issues arising as a result of domain experts being unfamiliar with ontologies in general. It was suggested that in order to improve understanding of the use case diagrams, they should be presented in a top-down fashion. However, in terms of the ontology itself, the only suggestion was to group the subclasses of `Asset` into `MediaAssets` and `TextAssets`. This has now been completed so we can conclude that the ontology is clear and intuitive, based on an inspection of the use cases.

4. **Conciseness**: The interview process resulted in some suggestions for improving the conciseness of the ontology. This feedback has been considered but found that changes to the ontology was unnecessary, as explained in Section 8.3.1. Furthermore, a quantitative evaluation of conciseness using the OntoMetrics tool would further suggest that the ontology is concise, as most of the scores returned were low. The only high score was attribute richness, which is to be expected given the purpose of the ontology is to organise metadata. We can therefore conclude that the ontology is sufficiently concise.

5. **Logical Consistency**: Unlike the other four criteria, logical consistency could be measured using solely quantitative methods. We already knew that disjointness assertions and custom restrictions were not used in the Creative Data Ontology so these would not cause inconsistencies. The other possible cause of inconsistencies in

an ontology were loops in its class hierarchy. The ontology was processed using an automated reasoner tool which indicated that there were no loops, so the ontology is logically consistent.

Based on the above summaries of the application of evaluation criteria to the Creative Data Ontology, the ontology now meets the criteria for a high-quality ontology. There are limitations to the ontology evaluation process, as outlined in Section 8.5, but the impact of these have been minimised as far as possible.

Research Question 2.3: Can this ontology act as a potentially "standard", widely-reusable metadata management ontology in the moving image industry?

Although the Creative Data Ontology does provide coverage for the four use cases defined in Section 6.3, it is not ready to act as a standard metadata management ontology in the moving image industry. The expertise of our industry partners resulted in the ontology focusing on scripted film and television production, unscripted television, and visual effects. As a result, a number of participants in the ontology evaluation interviews suggested areas in which the ontology could be extended, such as to encapsulate genre domains (e.g. comedy and drama) and other types of production (e.g. virtual reality and streaming). These suggestions are presented in more detail in Section 8.4.1

Additionally, each partner company provided only their core 100-150 metadata fields upon which to base the ontology so a large portion of fields are missing. It is important to note that one domain expert interviewed believes that 90% of the core fields would cover other sub-sectors of the industry well, such as virtual reality and immersive. Therefore, we can consider this ontology reusable but only for the industry sub-sectors listed above. We should instead label this ontology as a prototype or proof-of-concept to demonstrate the possibilities of ontology-based metadata management solutions in the moving image industry.

Summary of Contributions

In summary, this thesis has identified a viable use case for a domain ontology of metadata for use in the moving image industry. It then presented and validated an ontology that aims to fill this gap. This ontology is targeted towards four specific subdomains or sectors within this industry, which are: film production, unscripted television, scripted television, and visual effects.

Through developing and validating an ontology of metadata for a small slice of the moving image industry, we were able to define a generalised ontology engineering process with a set of recommendations which can be reused to develop similar ontologies for a wider segment of the moving image industry. Due to the universal applicability of the engineering process followed in this thesis, it should be possible to use it to develop a metadata ontology for other industries too.

Finally, the conclusions reached using the feedback from domain and ontology experts, as well as the results from the quantitative evaluation of the Creative Data Ontology, reinforces the belief that the engineering process outlined in this thesis was correct and provides confidence that it, and the Creative Data Ontology, has great potential to be used by the media industry.

9.2 Impact of Research

The work contained in this thesis has a number of benefits both to the moving image industry that it was centred around, and for other industries that wish explore the potential uses of ontologies for metadata management.

1. **Information Retrieval**: First and foremost, we must consider the impact of the Creative Data Ontology on the moving image industry. In Section 6.1, we identified a list of pain points that result in metadata either being unavailable or not used

efficiently during the media creation process. One of those pain points is the idiosyncratic renaming of files, which the Creative Data Ontology can mitigate against. The Creative Data Ontology has a class, `File`, that can represent any type of file stored on a computer or server. The user can attach metadata about a file to the instance of the `File` class that represents it in the ontology. More importantly, contextual information about the file is also available. For example, a video clip file can be represented in the ontology and be associated with information, such as which shoot it is from. This contextual information means that even if file names do not follow company- or project-level naming conventions, it is still possible identify a file, as long as the current name of the file is stored in the ontology. We should also note that Use Case 4 for metadata versioning (described in Section 6.3.4) enables a history of filenames to be kept to further support this.

2. **Documenting Industry Knowledge**: One domain expert stated during the ontology evaluation interviews that often, domain knowledge is not documented formally but is instead passed on to other industry practitioners verbally or is gained through experience. They gave the example of virtual reality being a relatively new subdomain of the moving industry and a lot of domain knowledge is being learned through experience. Although the subdomains within the scope the Creative Data Ontology are not new, as new people join the industry early in their careers, the ontology could serve as a form of documentation for these subdomains.

3. **Insights into Ontology Engineering**: Engineering an ontology for domains where similar ontologies do not already exist is a challenging task. One of the reasons is the lack of a benchmark ontology to use for comparisons and the lack of guidance on the engineering process that is specific to that domain. In producing the Creative Data Ontology, we have also documented the process taken to develop it and any lessons learned. The intention behind this is to provide guidance for designing, building, and evaluating ontologies for other domains, such as those adjacent to the moving image

industry, in order to streamline the engineering process for future developers.

9.3 Future Work

There are many possible paths that our future work could take. In this section, we give an outline of some of the possible directions that our further research could take:

1. **Implement Extensions Suggested by Experts**: During the ontology evaluation process, our interviewees made a number of suggestions for improving and extending the Creative Data Ontology. The improvements listed in Section 8.3.2 have now been implemented but the extensions have not, as they were beyond the original scope. Additionally, the ontology could be extended to provide coverage for other areas of the media creation domain, such as specific genres (e.g. comedy and drama), virtual reality, streaming, interactive content, and other types of footage (e.g. aerial and drone footage). Therefore, future work could involve conducting ethnographic research into the workflows of companies that specialise in these sub-domains and expanding the ontology accordingly.

   In order to support this proposal for future work, the Creative Data Ontology has been made open-source with an MIT license, available via the following links:

   - `https://zenodo.org/badge/latestdoi/662246329`
   - `https://github.com/cadexiades/PhD-Deliverables`

   When deciding to make the ontology open-source, we wanted a license that allowed other entities to use the ontology freely and without limitation, as long as attribution is included, and as long as there was no expectation of warranty or license holder liability. The MIT license was chosen due to its permissive nature: the only condition of use is that the copyright and license notices are preserved, which ensures attribution. The MIT license permits commercial use, distribution, and modification and such

allowances will mean future developers with not be restricted in their use of the Creative Data Ontology. This will allow the ontology to be improved, extended, and/or adapted to allow improvements to be made to metadata management practices in the industry. MIT also does not provide a warranty or an acceptance of liability on behalf of the license holder [130].

2. **Integration into a Software Application**: Another direction for future research would be to integrate the ontology into a metadata management tool. Work to develop a prototype has already been completed by the software development company seconded by the StoryFutures Theme 3 Research Team but, due to budgetary constraints at the time of development, the functionality of the tool is limited to asset importation. Therefore, future work could focus on extending the prototype to provide additional functionality, such as:

    - Support for the remaining use cases defined in Section 6.3.
    - A search feature to locate items of metadata.
    - A notification feature to notify relevant users of updates to metadata.
    - Link generation for individual items of metadata to facilitate quick access.
    - Data maturity modelling.

3. **Context-Aware Sub-Ontologies**: One domain expert suggested investigating the feasibility of automatically generating sub-ontologies derived from a master ontology to only include fields that are specific to a production type (e.g. a VFX project or a scripted film). Unbeknown to this domain expert, this had already been briefly investigated during the course of this project, in collaboration with with software development company commissioned to build metadata management tool that uses the ontology.

We considered that this could be implemented by tagging ontology components with the project types that they are applicable to. The metadata management tool would allow the user to select a project type and then a mapping (or 'translation') layer within the tool would generate a custom ontology by filtering the master ontology to only include components that have been tagged with the selected project type. In order to implement context-awareness using component tagging, we first had to investigate whether there was a convenient method for tagging the components in the ontology. We found that the only suitable method for tagging would be to use custom annotation properties. The outcome of the research into project customisation was that there was no convenient way of creating a context-aware ontology using component tagging, as it would involve a lot of manual tagging for the following reasons:

(a) We considered creating a custom annotation property for every project type, which will accept a Boolean value indicating whether it is applicable to that project type. However, to ensure that ontology scaled well if expanded for other project types in the future, we decided that this would be impractical.

(b) We also considered a single custom annotation property – `projectType` – instead. This annotation would take a text value consisting of a list of comma-separated project type names (for example, "VFX, unscripted"). However, this method would require every class and every property within the ontology to be tagged manually so, in order to avoid this, components that are applicable to all project types would not have a `projectType` annotation assigned to it. These components will be considered 'generic' components and included in every sub-ontology generated, regardless of the project type selected.

(c) We then realised that when the ontology is expanded to include a new project type, the ontology developer would need to manually check every existing com-

ponent to ensure that it is applicable to that project type and, if it is not, tag them accordingly.

Due time and budgetary constraints, there wasn't enough time to investigate other options for creating sub-ontologies for each project type. Therefore, investigating an efficient method of generating sub-ontologies would be another direction for future research to take.

Bibliography

[1] "Theme report 2019," tech. rep., Motion Pictures Association, 2019.

[2] "Film industry companies 2017," tech. rep., British Film Institute, 08 2017.

[3] "Film industry companies 2019," tech. rep., British Film Institute, 08 2019.

[4] W. Bailer and P. Schallauer, *Metadata in the Audiovisual Media Production Process*, pp. 65–84. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[5] "The 3 stages of film production." `https://fxhome.com/blog/the-3-stages-of-film-production`.

[6] J. Blat, A. Evans, H. Kim, E. Imre, L. Polok, V. Ila, P. Zemcík, A. Tefas, P. Smrz, A. Hilton, and I. Pitas, "Big data analysis for media production," *Proceedings of the IEEE*, vol. 104, pp. 1–30, 12 2015.

[7] S. Atkinson, J. Lehmann, and R. Evans, "The deep film access project: Ontology and metadata design for digital film production assets," in *The Second IEEE Big Data 2014 Workshop*, pp. 1–4, Oct. 2014. The Second IEEE Big Data 2014 Workshop ; Conference date: 27-10-2014.

[8] J. R. Smith and P. Schirling, "Metadata standards roundup," *IEEE MultiMedia*, vol. 13, no. 2, pp. 84–88, 2006.

[9] M. Uschold, "Ontology and database schema: What's the difference?," *Applied Ontology*, vol. 10, pp. 243–258, 12 2015.

[10] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquis.*, vol. 5, p. 199–220, June 1993.

[11] A. Gomez-Perez, M. Fernandez-Lopez, and O. Corcho, *Ontological Engineering: With Examples from the Areas of Knowledge Management, E-Commerce and the Semantic Web*. 01 2004.

[12] W. N. Borst, "Construction of engineering ontologies for knowledge sharing and reuse," 1997.

[13] R. Studer, V. Benjamins, and D. Fensel, "Knowledge engineering: Principles and methods," *Data Knowledge Engineering*, vol. 25, no. 1, pp. 161–197, 1998.

[14] A. Goldstein, L. Fink, and G. Ravid, "A framework for evaluating agricultural ontologies," *Sustainability*, vol. 13, no. 11, 2021.

[15] J. Domingue, D. Fensel, and J. A. Hendler, "Introduction to the semantic web technologies," in *Handbook of Semantic Web Technologies*, 2011.

[16] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," 2001.

[17] N. Guarino, "Formal ontologies and information systems," 06 1998.

[18] A. Gomez-Perez, "Ontological engineering: A state of the art," *Expert Update*, vol. 2, 01 1999.

[19] P. Gaitanou, "Ontologies and ontology-based applications," in *Metadata and Semantics* (M.-A. Sicilia and M. D. Lytras, eds.), (Boston, MA), pp. 289–298, Springer US, 2009.

[20] P. Winstanley, "Is an upper-level ontology useful?." Endorse: The European Data Conference on Reference Data and Semantics, 2021.

[21] D. Allemang and J. Hendler, *Semantic Web for the Working Ontologist*. Morgan Kaufmann Publishers, 2nd ed., 2008.

[22] G. Antoniou and F. v. Harmelen, *A Semantic Web Primer, 2nd Edition (Cooperative Information Systems)*. The MIT Press, 2nd ed., 2008.

[23] B. McBride, *The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS*, pp. 51–65. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[24] "Owl web ontology language overview." `https://www.w3.org/TR/2004/REC-owl-features-20040210`.

[25] A. Marchetti, F. Ronzano, M. Tesconi, and S. Minutoli, "Formalizing knowledge by ontologies: Owl and kif," 05 2008.

[26] M. R. Genesereth, R. E. Fikes, *et al.*, "Knowledge interchange format-version 3.0: reference manual," 1992.

[27] "Knowledge interchange format specification." `http://logic.stanford.edu/kif/syntax.html`.

[28] N. Noy and D. Mcguinness, "Ontology development 101: A guide to creating your first ontology," *Knowledge Systems Laboratory*, vol. 32, 01 2001.

[29] "Owl web ontology language reference." `https://www.w3.org/TR/owl-ref`.

[30] M. Horridge, S. Jupp, G. Moulton, A. Rector, R. Stevens, and C. Wroe, "A practical guide to building owl ontologies using protege 4 and co-ode tools," 2007.

[31] M. Samwald, "Classes versus individuals : Fundamental design issues for ontologies on the biomedical semantic web," 2006.

[32] J. Bailey, F. Bry, T. Furche, and S. Schaffert, *Web and Semantic Web Query Languages: A Survey*, pp. 35–133. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005.

[33] G. Karvounarakis, A. Magkanaraki, S. Alexaki, V. Christophides, D. Plexousakis, M. Scholl, and K. Tolle, "Querying the semantic web with rql," *Computer Networks*, vol. 42, pp. 617–640, 08 2003.

[34] T. Furche, F. Bry, S. Schaffert, R. Orsini, I. Horrocks, M. Krauss, and O. Bolzer, "Survey over existing query and transformation languages revision 2.0," 01 2004.

[35] C. O'Dell and C. J. Grayson, "If only we knew what we know: Identification and transfer of internal best practices," *California Management Review*, vol. 40, no. 3, pp. 154–174, 1998.

[36] R. Maier and T. Hädrich, "Knowledge management systems," *Encyclopedia of Knowledge Management*, vol. 1, 01 2010.

[37] P. Ribino, A. Oliveri, G. L. Re, and S. Gaglio, "A knowledge management system based on ontologies," in *2009 International Conference on New Trends in Information and Service Science*, pp. 1025–1033, 2009.

[38] F. Freitas, H. Stuckenschmidt, and N. Noy, "Ontology issues and applications guest editors' introduction," *Journal of the Brazilian Computer Society*, vol. 11, pp. 5–16, 2010.

[39] A. Abecker, A. Bernardi, K. Hinkelmann, O. Kuhn, and M. Sintek, "Toward a technology for organizational memories," *Intelligent Systems and their Applications, IEEE*, vol. 13, pp. 40 – 48, 06 1998.

[40] P. Sureephong, N. Chakpitak, Y. Ouzrout, and A. Bouras, "An ontology-based knowledge management system for industry clusters," *Global Design to Gain a Competitive Edge: An Holistic and Collaborative Design Approach Based on Computational Tools*, 09 2007.

[41] S. Schulz, H. Stenzhorn, M. Boeker, and B. Smith, "Strengths and limitations of formal ontologies in the biomedical domain," *Revista electronica de comunicacao, informacao inovacao em saude : RECIIS*, vol. 3, pp. 31–45, 03 2009.

[42] J. Hastings, *Primer on Ontologies*, pp. 3–13. New York, NY: Springer New York, 2017.

[43] K. Sengupta and P. Hitzler, *Web Ontology Language (OWL)*, pp. 2374–2378. New York, NY: Springer New York, 2014.

[44] I. Horrocks, *What Are Ontologies Good For?*, pp. 175–188. 01 2013.

[45] B. Ramis Ferrer, W. M. Mohammed, M. Ahmad, S. Iarovyi, J. Zhang, R. Harrison, and J. L. Martinez Lastra, "Comparing ontologies and databases: A critical review of lifecycle engineering models in manufacturing," *Knowl. Inf. Syst.*, vol. 63, p. 1271–1304, jun 2021.

[46] C. Martinez-Cruz, I. J. Blanco, and M. A. Vila, "Ontologies versus relational databases: are they so different? a comparison," *Artificial Intelligence Review*, vol. 38, pp. 271–290, Dec 2012.

[47] N. Konstantinou, D.-E. Spanos, and N. Mitrou, "Ontology and database mapping: A survey of current implementations and future directions.," *Journal of Web Engineering (JWE)*, vol. 7, pp. 1–24, 03 2008.

[48] C. A. Dexiades and C. P. R. Heath, "Creative data ontology: 'russian doll' metadata versioning in film and tv post-production workflows," in *Metadata and Semantic*

Research (E. Garoufallou and M.-A. Ovalle-Perandones, eds.), (Cham), pp. 204–215, Springer International Publishing, 2021.

[49] S. Choudhury, B. Pham, R. Smith, and P. Higgs, "Loculus: A metadata wrapper for digital motion pictures," in *Proceedings of the 11th IASTED International Conference on Internet and Multimedia Systems and Applications* (A. Hac, ed.), pp. 151–156, ACTA Press, 2007.

[50] S. T. Choudhury, *Loculus: An ontology-based information management framework for the Motion Picture Industry*. PhD thesis, Queensland University of Technology, 2010.

[51] "A creative works ontology for the film and television industry," tech. rep., Motion Picture Laboratories Inc., 2018.

[52] R. Arndt, R. Troncy, S. Staab, and L. Hardman, *COMM: A Core Ontology for Multimedia Annotation*, pp. 403–421. 12 2008.

[53] A. Chakravarthy, R. Beales, N. Matskanis, and X. Yang, "Ontofilm: A core ontology for film production," in *Semantic Multimedia* (T.-S. Chua, Y. Kompatsiaris, B. Mérialdo, W. Haas, G. Thallinger, and W. Bailer, eds.), (Berlin, Heidelberg), pp. 177–181, Springer Berlin Heidelberg, 2009.

[54] F. Nack and A. Lindsay, "Everything you wanted to know about mpeg-7. 1," *Multimedia, IEEE*, vol. 6, pp. 65 – 77, 08 1999.

[55] J. Lehmann, S. Atkinson, and R. Evans, "Applying semantic technology to film production," in *The Semantic Web: ESWC 2015 Satellite Events* (F. Gandon, C. Guéret, S. Villata, J. Breslin, C. Faron-Zucker, and A. Zimmermann, eds.), (Cham), pp. 445–453, Springer International Publishing, 2015.

[56] S. Atkinson, *From Film Practice to Data Process: Production Aesthetics and Representational Practices of a Film Industry in Transition*. Edinburgh University Press, 04 2018.

[57] A. Abou Assali, D. Lenne, and B. Debray, "Komis: An ontology-based knowledge management system for industrial safety," pp. 475–479, 09 2007.

[58] S.-T. Li, H.-C. Hsieh, and I.-W. Sun, "An ontology-based knowledge management system for the metal industry.," 01 2003.

[59] K. W. Chau, "An ontology-based knowledge management system for flow and water quality modeling," *Adv. Eng. Softw.*, vol. 38, p. 172–181, Mar. 2007.

[60] M. Uschold and M. King, "Towards a methodology for building ontologies," in *In Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with IJCAI-95*, 1995.

[61] M. Hadzic, P. Wongthongtham, T. Dillon, and E. Chang, *Ontology Design Approaches*, pp. 75–91. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.

[62] M. Jarrar and R. Meersman, *Ontology engineering - The DOGMA approach*, pp. 7–34. 12 2008.

[63] M. Jarrar and R. Meersman, "Formal ontology engineering in the dogma approach," pp. 1238–1254, 10 2002.

[64] H. Schnurr, Y. Sure, H. Akkermans, and R. Studer, "On-to-knowledge methodology baseline version," 2000.

[65] Y. Sure, S. Staab, and R. Studer, *On-To-Knowledge Methodology (OTKM)*, pp. 117–132. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004.

[66] A. De Nicola, M. Missikoff, and R. Navigli, "A proposal for a unified process for ontology building: Upon," pp. 655–664, 08 2005.

[67] M. C. Suárez-Figueroa, A. Gomez-Perez, and M. Fernández-López, *The NeOn Methodology for Ontology Engineering*, pp. 9–34. 01 2012.

[68] K. Patel, I. Dube, L. Tao, and N. Jiang, "Extending owl to support custom relations," in *2015 IEEE 2nd International Conference on Cyber Security and Cloud Computing*, pp. 494–499, 2015.

[69] M. A. Musen, "The protégé project: A look back and a look forward," *AI Matters*, vol. 1, p. 4–12, June 2015.

[70] E. Alatrish, "Comparison of ontology editors," *ERAF Journal on Computing*, vol. 4, pp. 23–38, 01 2012.

[71] M. Erdmann and W. Waterfeld, *Overview of the NeOn Toolkit*, pp. 281–301. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[72] A. Gomez-Perez, "Evaluation of ontologies," *International Journal of Intelligent Systems*, vol. 16, pp. 391 – 409, 03 2001.

[73] E. J. Davidson, *What is Evaluation?* London, UK: Sage Publications, 2004.

[74] I. Cantador, M. Fernández Sánchez, and P. Castells, "A collaborative recommendation framework for ontology evaluation and reuse," 2006.

[75] J. Yu, J. Thom, and A. Tam, "Requirements-oriented methodology for evaluating ontologies," *Inf. Syst.*, vol. 34, pp. 766–791, 12 2009.

[76] J. Raad and C. Cruz, "A survey on ontology evaluation methods," 11 2015.

[77] S. Tartir, I. Arpinar, and A. Sheth, *Ontological Evaluation and Validation*, pp. 115–130. 09 2010.

[78] J. Brank, M. Grobelnik, and D. Mladenić, "A survey of ontology evaluation techniques," 01 2009.

[79] C. Brewster, H. Alani, S. Dasmahapatra, and Y. Wilks, "Data driven ontology evaluation," in *International Conference on Language Resources and Evaluation (30/05/04)*, 2004. Event Dates: 24-30 May.

[80] M. Gulić, I. Magdalenić, and B. Vrdoljak, "Ontology matching using tf/idf measure with synonym recognition," in *Information and Software Technologies* (T. Skersys, R. Butleris, and R. Butkiene, eds.), (Berlin, Heidelberg), pp. 22–33, Springer Berlin Heidelberg, 2013.

[81] H. Zhu, D. Liu, I. Bayley, A. Aldea, Y. Yang, and Y. Chen, "Quality model and metrics of ontology for semantic descriptions of web services," *Tsinghua Science and Technology*, vol. 22, no. 3, pp. 254–272, 2017.

[82] M. Lopez, A. Gomez-Perez, J. Sierra, and A. Pazos, "Building a chemical ontology using methontology and the ontology design environment. ieee intelligent systems," *Intelligent Systems and their Applications, IEEE*, vol. 14, pp. 37 – 46, 02 1999.

[83] Y. Wand and W. RY, "On the ontological expressiveness of information systems analysis and design grammars," *Information Systems Journal*, vol. 3, pp. 217 – 237, 06 2008.

[84] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing?," *International Journal of Human-Computer Studies*, vol. 43, no. 5, pp. 907–928, 1995.

[85] B. Lantow, "Ontometrics: Application of on-line ontology metric calculation," in *BIR Workshops*, 2016.

[86] D. Vrandečić, *Ontology evaluation*. PhD thesis, Karlsruhe Institute of Technology, 2010.

[87] T. Wei, C. Roche, M. Papadopoulou, and Y. Jia, "An ontology of chinese ceramic vases," pp. 53–63, 01 2020.

[88] M. Poveda-Villalón and M. C. Suárez-Figueroa, "Oops!–ontology pitfalls scanner!," tech. rep., Tech. rep., Ontology Engineering Group. Departamento de Inteligencia . . . , 2009.

[89] M. Poveda-Villalón, A. Gómez-Pérez, and M. C. Suárez-Figueroa, "OOPS! (OntOlogy Pitfall Scanner!): An On-line Tool for Ontology Evaluation," *International Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 10, no. 2, pp. 7–34, 2014.

[90] B. Lantow, "Ontometrics: Putting metrics into use for ontology evaluation.," in *KEOD*, pp. 186–191, 2016.

[91] "Ontometrics wiki." `https://ontometrics.informatik.uni-rostock.de/wiki/index.php/OntoMetrics`.

[92] A. Duque-Ramos, M. Boeker, L. Jansen, S. Schulz, M. Iniesta, and J. Fernandez-Breis, "Evaluating the good ontology design guideline (goodod) with the ontology quality requirements and evaluation method and metrics (oquare)," *PloS one*, vol. 9, p. e104463, 08 2014.

[93] "Oquare wiki." `http://miuras.inf.um.es/oquarewiki/index.php5/Main_Page`.

[94] A. Duque-Ramos, J. Fernandez-Breis, R. Stevens, and N. Aussenac-Gilles, "Oquare: A square-based approach for evaluating the quality of ontologies," *Journal of Research and Practice in Information Technology*, vol. 43, pp. 159–176, 05 2011.

[95] D. Schober, V. Svátek, and M. Boeker, "Checking class labels against naming conventions: First experiences with the ontocheck protégé plugin," in *ICBO*, 2012.

[96] "The ontocheck plugin." `http://www2.imbi.uni-freiburg.de/ontology/OntoCheck`.

[97] O. Corcho, A. Gomez-Perez, R. González-Cabero, and M. Suárez-Figueroa, "Odeval: A tool for evaluating rdf(s), daml+oil and owl concept taxonomies.," pp. 369–382, 01 2004.

[98] M. Crotty, *The Foundations of Social Research: Meaning and Perspective in the Research Process*. SAGE Publications, 1998.

[99] J. W. Creswell, *Research Design. Qualitative, Quantitative, and Mixed Methods Approaches*. SAGE Publications, Ltd, 3rd ed., 2009.

[100] R. E. Stake, *Qualitative Research. Studying How Things Work*. The Guilford Press, 2010.

[101] N. Denzin and Y. Lincoln, "The discipline and practice of qualitative research," 2005.

[102] C. R. Kothari, *Research Methodology. Methods and Techniques*. New Age International Publishers, 2nd ed., 2004.

[103] C. Williams, "Research methods," *Journal of Business Economics Research (JBER)*, vol. 5, Feb 2011.

[104] M. Yilmaz, *A Software Process Engineering Approach to Understanding Software Productivity and Team Personality Characteristics: An Empirical Investigation*. PhD

thesis, 01 2013.

[105] S. VanderStoep and D. Johnson, *Research methods for everyday life: Blending qualitative and quantitative approaches*, vol. 24. Jossey-Bass, 2008.

[106] W. F. Chua, "Theoretical constructions of and by the real," *Accounting, Organizations and Society*, vol. 11, no. 6, pp. 583–598, 1986.

[107] "Evolutions." `https://evolutions.tv`. Accessed: 2023-07-17.

[108] "Evolutions post production holdings limited annual report." `https://find-and-update.company-information.service.gov.uk/company/12904418/filing-history`, 2022. Uploaded to Companies House UK: 2022-09-07.

[109] "Dneg shows." `https://www.dneg.com/shows`. Accessed: 2023-07-17.

[110] "Dneg about." `https://www.dneg.com/about`. Accessed: 2023-07-17.

[111] "Double negative limited annual report and financial statements." `https://find-and-update.company-information.service.gov.uk/company/03325701/filing-history`, 2022. Uploaded to Companies House UK: 2022-04-05.

[112] "Double negative limited annual report and financial statements." `https://find-and-update.company-information.service.gov.uk/company/03325701/filing-history`, 2023. Uploaded to Companies House UK: 2023-04-11.

[113] "Pinewood studios production accommodation." `https://pinewoodgroup.com/studios/pinewood-studios/stages-facilities/production-accommodation`. Accessed: 2023-07-17.

[114] "Pinewood studios post-production." https://pinewoodgroup.com/studios/pinewood-studios/post-production. Accessed: 2023-07-17.

[115] "Pinewood studios credits." https://pinewoodgroup.com/pinewood-today/credits?studioid=1598. Accessed: 2023-07-17.

[116] "Pinewood studios limited report and financial statements." https://find-and-update.company-information.service.gov.uk/company/00392619/filing-history, 2021. Uploaded to Companies House UK: 2021-12-29.

[117] "Pinewood studios limited report and financial statements." https://find-and-update.company-information.service.gov.uk/company/00392619/filing-history, 2023. Uploaded to Companies House UK: 2023-01-04.

[118] J. J. Hox and H. R. Boeije, "Data collection, primary vs. secondary," *Encyclopedia of social measurement*, vol. 1, no. 1, pp. 593–599, 2005.

[119] M. Dudáš, S. Lohmann, V. Svátek, and D. Pavlov, "Ontology visualization methods and tools: a survey of the state of the art," *The Knowledge Engineering Review*, vol. 33, 07 2018.

[120] C. Ghezzi, *Research Ethics*, pp. 111–127. 06 2020.

[121] "Informed consent." https://researchsupport.admin.ox.ac.uk/governance/ethics/resources/consent.

[122] V. Dimitrova, R. Denaux, G. Hart, C. Dolbear, I. Holt, and A. G. Cohn, "Involving domain experts in authoring owl ontologies," in *The Semantic Web-ISWC 2008: 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008. Proceedings 7*, pp. 1–16, Springer, 2008.

[123] M. Alqadah, G. Stapleton, J. Howse, and P. Chapman, "Evaluating the impact of clutter in linear diagrams.," in *SetVR@ Diagrams*, pp. 4–18, 2016.

[124] C. John, "Measuring and reducing clutter in euler diagrams," *Electronic Notes in Theoretical Computer Science*, vol. 134, pp. 103–126, 2005. Proceedings of the First International Workshop on Euler Diagrams (Euler 2004).

[125] M. Burch, H. van de Wetering, and N. Klaassen, "Multiple linked perspectives on hierarchical data," in *Proceedings of the 13th International Symposium on Visual Information Communication and Interaction*, VINCI '20, (New York, NY, USA), Association for Computing Machinery, 2020.

[126] E. Montiel-Ponsoda, D. Vila-Suero, B. Villazon Terrazas, G. Dunsire, E. Rodriguez, and A. Gomez-Perez, "Style guidelines for naming and labeling ontologies in the multilingual web," vol. 0, 09 2011.

[127] T. Haymes, "The three-e strategy for overcoming resistance to technological change," *Educause Quarterly*, vol. 31, no. 4, pp. 67–69, 2008.

[128] D. Theofanidis and A. Fountouki, "Limitations and Delimitations in the Research Process," *Perioperative Nursing (GORNA), E-ISSN:2241-3634*, vol. 7, pp. 155–162, Jan. 2019.

[129] G. Guest, A. Bunce, and L. Johnson, "How many interviews are enough?: An experiment with data saturation and variability," *Field Methods*, vol. 18, no. 1, pp. 59–82, 2006.

[130] "Choose a license: Appendix." `https://choosealicense.com/appendix`.

[131] "Owlready2's documentation." `https://owlready2.readthedocs.io/en/latest`. Accessed: 2023-07-17.

[132] "Owlready2 worlds." https://owlready2.readthedocs.io/en/latest/world.html. Accessed: 2023-07-17.

Appendix A

The Creative Data Ontology

A.1 Ontology Classes

The latest iteration of the Creative Data Ontology consists of 68 classes. A list of these classes and their descriptions can be found Table A.1, which have been exported directly from Protégé.

Class Name	Description
`3DLidarScan`	The output of a LIDAR (remote sensing technology) scan, which is then used to create 3D models and maps of objects and scenes.
`3DModel`	A computer-generated representation of an object or character in 3D.
`Asset`	Any data (not metadata) that can be used in the pre-, post-, and production processes. These are usually stored as a `File` on the file system.

Budget	The spreadsheet containing financial information related to the cost of a production.
Camera	A camera, which creates clips during a shoot.
CameraRoll	This is the video output of a camera after a take.
Cast	A person who performed in a shoot.
Client	A subclass of Role and refers to anyone who works for the project client.
Clip	A single video file.
Crane	A camera crane, which allows shots to be taken from a height.
DataItem	Represents any entity that can have metadata associated with it.
DataTeamMember	A member of the data team who works for the company using the ontology.
Deliverable	An item delivered as a result of a project.
Dolly	A camera dolly, which is a wheeled cart for moving the camera horizontally.
EditProducer	An editorial producer who works for the company using the ontology.
Email	Any email communication, both internal and external, sent in relation to this project.
EmailReceipt	Any email communication, both internal and external, sent in relation to this project that acts as receipt (for example, to confirm the receipt of a drive).

`Equipment`	Any physical equipment (e.g. a camera, etc.) used in a project.
`EquipmentAttachment`	Anything that can be attached to piece of equipment (e.g. a lens, etc.) used in a project.
`Event`	Any action that requires metadata to be stored about.
`File`	A single file stored on the file system which can represent, for instance, a clip.
`FisheyeLens`	A fisheye lens used on cameras.
`Folder`	A single directory stored in the file system which can represent, for instance, a roll.
`Frame`	A single frame that, when combined with others, form a clip.
`HardDriveSnapshot`	A file containing the structure and details of files and folders stored on a hard drive.
`Issue`	Stores details of an issue which can be associated with instances of other classes.
`Lens`	A lens used on cameras.
`LensGrid`	An image of the checkered grid placed in front of a camera lens to record the physical imperfections that are natural to all camera lenses.
`Location`	Encapsulates metadata referring to any locations, including shoot locations.
`Logger`	A member of the live logging team who works for the company using the ontology.

`LoggingTag`	A descriptive tag that can be applied to a `Clip` by the live logging team to enable quick searching of clips.
`LUT`	A lookup table which provide a pre-determined colour grading settings for a `Clip`.
`MediaAssets`	Any media-based assets.
`Microphone`	A microphone, which adds sound to a clip.
`Note`	A comment that can be linked to other entities.
`Panoramic`	A panoramic photograph, i.e. a wide-format photo.
`Person`	Any human entity that requires representation in the ontology.
`Photogrammetry`	Photographs used to make a map, drawing or 3D model of a real-world object or scene.
`Project`	Represents a single project (e.g. a production).
`Prop`	An item used by a cast member when being filmed to make the scene more realistic.
`Record`	Encapsulates versioning metadata, including temporal details, about an entity.
`RectilinearLens`	A rectilinear lens used on cameras.
`ReferenceMaterial`	Reference material typically about texture, surface and colour (e.g. Macbeth colour charts).

`Rental`	Encapsulates the information regarding the rental of equipment for the purposes of a shoot.
`Rig`	A large setup of multiple cameras.
`Role`	Represents a real-world role that a `Person` entity can have in a project. This class encapsulates the specific roles in the form of its subclasses.
`Roll`	This is the output of a camera after a take. The terminology refers to a pre-digital time period.
`Scene`	The action in a single location and continuous time.
`Script`	Represents the physical script which is performed by actors in a production.
`ScriptBreakdown`	Represents a breakdown of the physical script which is performed by actors in a production.
`Shoot`	Refers to the entire event of producing raw, unedited content.
`ShootDay`	Refers to a single day of a shoot.
`ShootingSchedule`	Represents a timetable of shoots.
`Shot`	The event of shooting one scene. A scene is made up of multiple shots from multiple cameras.
`Slate`	The clapperboard used to assist in synchronizing of picture and sound.

SoundRoll	This is the audio output of a camera after a take.
Staff	Anyone who works for the company using the ontology.
StereoLeftEyeLens	A stereo left eye lens used on cameras.
StereoRightEyeLens	A stereo right eye lens used on cameras.
Still	A still image.
Storyboard	Represents a visual representation of a script.
Take	A single version of a shot.
Task	Stores details of a task which can be associated with instances of other classes.
TextAssets	Any text-based assets.
Unit	A shoot unit, which is the crew working at a particular shoot location.
VideoEditor	A video editor who works for the company using the ontology.
Writer	A person responsible for writing something (e.g. a script).
WrittenNote	A handwritten set of notes that has been scanned and saved on to the file system.

Table A.1: Classes in the Creative Data Ontology and their descriptions.

A.2 Ontology Object Properties

The latest iteration of the ontology consists of 166 object properties, 106 of which are 'usable', as defined in 6.2. A list of usable object properties and their domains and ranges can be found Table A.2, which have been exported directly from Protégé.

Object Property Name	Domain	Range
becomesFile	Asset	File
cameraIsPartOfRig	Camera	Rig
cameraProducesCameraRoll	Camera	CameraRoll
cameraProducesSoundRoll	Camera	SoundRoll
cameraUsesCrane	Camera	Crane
cameraUsesLens	Camera	Lens
cameraRollBecomesFolder	CameraRoll	Folder
cameraRollContainsClip	CameraRoll	Clip
cameraRollIsFromCamera	CameraRoll	Camera
castIsPartOfShoot	Cast	Shoot
castUsedInProject	Cast	Project
clipHasFirstFrame	Clip	Frame
clipHasIssue	Clip	Issue
clipHasLastFrame	Clip	Frame
clipHasLoggingTag	Clip	LoggingTag
clipHasSoundRoll	Clip	SoundRoll
clipIsFromCameraRoll	Clip	CameraRoll
clipIsFromTake	Clip	Take
clipUsesLens	Clip	Lens
clipUsesLUT	Clip	LUT
hasClipNote	Clip	Note

dataItemHasRecord	DataItem	Record
emailSentToClient	Email	Client
emailUsesHardDriveSnapshot	Email	HardDriveSnapshot
equipmentHasAttachment	Equipment	EquipmentAttachment
hasEquipmentNote	Equipment	Note
folderContainsFile	Folder	File
frameIsPartOfClip	Frame	Clip
hardDriveSnapshotIsPartOfEmail	HardDriveSnapshot	Email
hardDriveSnapshotSentToClient	HardDriveSnapshot	Client
issueCheckedBy	Issue	Staff
issueCreatedBy	Issue	Staff
issueFixedBy	Issue	Staff
hasLensNote	Lens	Note
lensCreatedBy	Lens	Staff
lensHasLensGrid	Lens	LensGrid
lensUpdatedBy	Lens	Staff
lensUsedInClip	Lens	Clip
lensUsedInUnit	Lens	Unit
shootLocationUsedInShootDay	Location	ShootDay
loggingTagCreatedBy	LoggingTag	Logger
personHasRole	Person	Role
projectHasClient	Project	Client
projectHasDeliverable	Project	Deliverable
projectHasRental	Project	Rental
projectHasShoot	Project	Shoot
projectHasShootingSchedule	Project	ShootingSchedule

hasPreviousRecordVersion	Record	Record
recordIsAboutDataItem	Record	DataItem
rentalContainsEquipment	Rental	Equipment
rigContainsCamera	Rig	Camera
hasSceneContactList	Scene	Staff
hasSceneNote	Scene	Note
sceneCreatedBy	Scene	Staff
sceneHasAsset	Scene	Asset
sceneHasShot	Scene	Shot
sceneHasTask	Scene	Task
sceneIsPartOfShootDay	Scene	ShootDay
sceneUpdatedBy	Scene	Staff
scriptHasScene	Script	Scene
scriptHasScriptBreakdown	Script	ScriptBreakdown
scriptHasStoryboard	Script	Storyboard
scriptTranslatedTo	Script	Script
scriptWrittenBy	Script	Writer
hasShootNote	Shoot	Note
shootCreatedBy	Shoot	Staff
shootHasShootDay	Shoot	ShootDay
shootIsPartOfProject	Shoot	Project
shootUpdatedBy	Shoot	Staff
shootUsesCast	Shoot	Cast
shootUsesUnit	Shoot	Unit
shootDayHasScene	ShootDay	Scene
shootDayhasShootLocation	ShootDay	Location

shootDayIsPartOfShoot	ShootDay	Shoot
shotIsPartOfScene	Shot	Scene
shotProducesTake	Shot	Take
hasSlateDataOpNotes	Slate	Note
hasSlateNote	Slate	Note
slateCreatedBy	Slate	Staff
slateHasEquipment	Slate	Equipment
slateHasFile	Slate	File
slateHasShootLocation	Slate	Location
slateHasTake	Slate	Take
slateHasUnit	Slate	Unit
slateUpdatedBy	Slate	Staff
slateUsesCamera	Slate	Camera
slateUsesDriftCamera	Slate	Camera
slateUsesLens	Slate	Lens
slateUsesWitnessCamera	Slate	Camera
takeIsFromShot	Take	Shot
takeProducesClip	Take	Clip
takeUsesCamera	Take	Camera
takeUsesFisheyeLens	Take	FisheyeLens
takeUsesLens	Take	Lens
takeUsesRectilinearLens	Take	RectilinearLens
takeUsesSlate	Take	Slate
takeUsesStereoLeftEyeLens	Take	StereoLeftEyeLens
takeUsesStereoRightEyeLens	Take	StereoRightEyeLens
hasUnitNote	Unit	Note

unitCreatedBy	Unit	Staff
unitHasFacilitySupervisor	Unit	Staff
unitHasSlate	Unit	Slate
unitHasVFXSupervisor	Unit	Staff
unitIsPartOfShoot	Unit	Shoot
unitUpdatedBy	Unit	Staff
unitUsesLens	Unit	Lens

Table A.2: Usable object properties in the Creative Data Ontology.

A.3 Ontology Data Type Properties

The latest iteration of the ontology consists of 287 data type properties, 243 of which are 'usable'. A list of usable data type properties and their domains and ranges can be found Table A.3, which have been exported directly from Protégé.

Data Type Property Name	Domain	Range
hasCameraBodyID	Camera	xsd:string
hasCameraBrandName	Camera	xsd:string
hasCameraModelName	Camera	xsd:string
hasCameraType	Camera	xsd:string
hasCameraRollNumber	CameraRoll	xsd:string
hasCastName	Cast	xsd:string
hasClientAccountNumber	Client	xsd:string
hasClientCompany	Client	xsd:string
hasClientEmailAddress	Client	xsd:string
hasClientName	Client	xsd:string
hasClientPostalAddress	Client	xsd:string
clipUpdatedOn	Clip	xsd:dateTimeStamp
hasClipActiveHeight	Clip	xsd:int
hasClipActiveWidth	Clip	xsd:int
hasClipASCSAT	Clip	xsd:float
hasClipASCSOP	Clip	xsd:string
hasClipAudioBitDepth	Clip	xsd:float
hasClipAudioBitRate	Clip	xsd:float
hasClipAudioCodec	Clip	xsd:string
hasClipAudioEndianness	Clip	xsd:string
hasClipAudioFormat	Clip	xsd:string

hasClipAudioMode	Clip	xsd:string
hasClipAudioSamplingRate	Clip	xsd:float
hasClipBitRateMode	Clip	xsd:string
hasClipCameraFormat	Clip	xsd:string
hasClipCameraRole	Clip	xsd:string
hasClipCameraTimecode	Clip	xsd:string
hasClipChromaSubsampling	Clip	xsd:string
hasClipColourSpace	Clip	xsd:string
hasClipComment	Clip	xsd:string
hasClipDisplayAspectRatio	Clip	xsd:string
hasClipDuration	Clip	xsd:string
hasClipEditName	Clip	xsd:string
hasClipEndTimecode	Clip	xsd:string
hasClipEvent	Clip	xsd:int
hasClipFix	Clip	xsd:boolean
hasClipFormat	Clip	xsd:string
hasClipFrameCountEnd	Clip	xsd:int
hasClipFrameCountStart	Clip	xsd:int
hasClipFrameRate	Clip	xsd:float
hasClipFrameRateMode	Clip	xsd:string
hasClipFramingType	Clip	xsd:string
hasClipGPSTimeStamp	Clip	xsd:dateTimeStamp
hasClipHeight	Clip	xsd:int
hasClipInkDuration	Clip	xsd:string
hasClipInkNumber	Clip	xsd:int
hasClipInputFormat	Clip	xsd:string

hasClipISO	Clip	xsd:int
hasClipKNCopy	Clip	xsd:string
hasClipKNStart	Clip	xsd:string
hasClipLabRoll	Clip	xsd:string
hasClipLengthInFrames	Clip	xsd:int
hasClipLTO	Clip	xsd:string
hasClipMarkIn	Clip	xsd:string
hasClipMasterName	Clip	xsd:string
hasClipNumberOfAudioStreams	Clip	xsd:int
hasClipNumberOfVideoStreams	Clip	xsd:int
hasClipOutputFormat	Clip	xsd:string
hasClipOverallBitRate	Clip	xsd:float
hasClipPixelAspectRatio	Clip	xsd:string
hasClipQualityCheck	Clip	xsd:boolean
hasClipStartTimecode	Clip	xsd:string
hasClipTapeName	Clip	xsd:string
hasClipTrack	Clip	xsd:string
hasClipTXSlate	Clip	xsd:string
hasClipUnitRole	Clip	xsd:string
hasClipVideoBitRate	Clip	xsd:float
hasClipVideoCodec	Clip	xsd:string
hasClipVideoFormat	Clip	xsd:string
hasClipVideoFormatVersion	Clip	xsd:string
hasClipVideoMode	Clip	xsd:string
hasClipVideoProfile	Clip	xsd:string
hasClipVideoResolution	Clip	xsd:string

hasClipWhiteBalance	Clip	xsd:int
hasClipWidth	Clip	xsd:int
isClipSelected	Clip	xsd:boolean
hasDeliverableAspectRatio	Deliverable	xsd:string
hasDeliverableAudioLanguage	Deliverable	xsd:language
hasDeliverableAudioLayout	Deliverable	xsd:string
hasDeliverableAudioType	Deliverable	xsd:string
hasDeliverableAudioVersion	Deliverable	xsd:string
hasDeliverableDescription	Deliverable	xsd:string
hasDeliverableDubbingMixer	Deliverable	xsd:string
hasDeliverableDubbingStudio	Deliverable	xsd:string
hasDeliverableFrameRate	Deliverable	xsd:float
hasDeliverableLength	Deliverable	xsd:string
hasDeliverableLFOA	Deliverable	xsd:string
hasDeliverablePictureFileFormat	Deliverable	xsd:string
hasDeliverablePictureResolution	Deliverable	xsd:string
hasDeliverablePictureVersion	Deliverable	xsd:string
hasDeliverableReelCount	Deliverable	xsd:int
hasDeliverableReelNumber	Deliverable	xsd:int
hasDeliverableReleaseDate	Deliverable	xsd:dateTimeStamp
hasDeliverableSubtitleLanguage	Deliverable	xsd:language
fileCreatedOn	File	xsd:dateTimeStamp
fileUpdatedOn	File	xsd:dateTimeStamp
hasFileAccessPermissions	File	xsd:string
hasFileChecksum	File	xsd:string
hasFileExtension	File	xsd:string

hasFileFieldDominance	File	xsd:string
hasFileFormatProfile	File	xsd:string
hasFileName	File	xsd:string
hasFilePath	File	xsd:string
hasFileScanType	File	xsd:string
hasFileSize	File	xsd:int
hasFileWritingApplication	File	xsd:string
hasIssueCheckDate	Issue	xsd:dateTimeStamp
hasIssueDate	Issue	xsd:dateTimeStamp
hasIssueDescription	Issue	xsd:string
hasIssueFixDate	Issue	xsd:dateTimeStamp
hasIssueMissingLines	Issue	xsd:boolean
hasIssueMixerResponse	Issue	enumeration
hasIssueTimecodeIn	Issue	xsd:string
hasIssueTimecodeOut	Issue	xsd:string
hasIssueType	Issue	xsd:string
hasLensComment	Lens	xsd:string
hasLensDescription	Lens	xsd:string
hasLensGridAngle	Lens	xsd:string
hasLensGridCameraRoll	Lens	xsd:string
hasLensGridDistance	Lens	xsd:string
hasLensGridTake	Lens	xsd:string
hasLensGridTStop	Lens	xsd:string
hasLensGroup	Lens	xsd:string
hasLensHeroScanCount	Lens	xsd:int
hasLensName	Lens	xsd:string

hasLensNameAlternative	Lens	xsd:string
hasLensReel	Lens	xsd:string
hasLensSerialNumber	Lens	xsd:string
hasLensStatus	Lens	xsd:string
hasLensTag	Lens	xsd:string
hasLensType	Lens	xsd:string
lensCreatedOn	Lens	xsd:dateTimeStamp
lensUpdatedOn	Lens	xsd:dateTimeStamp
hasLocationAddress	Location	xsd:string
hasLocationAltitude	Location	xsd:string
hasLocationLatitude	Location	xsd:string
hasLocationLatitudeReference	Location	enumeration
hasLocationLongitude	Location	xsd:string
hasLocationLongitudeReference	Location	xsd:string
hasLocationName	Location	xsd:string
hasLoggingTagName	LoggingTag	xsd:string
hasLoggingTagTimecodeInClip	LoggingTag	xsd:string
hasLoggingTagTimestamp	LoggingTag	xsd:dateTimeStamp
hasLoggingTagType	LoggingTag	enumeration
isLoggingTagAudioOrVideo	LoggingTag	enumeration
hasNoteStatus	Note	enumeration
hasProjectEpisodeTitle	Project	xsd:string
hasProjectGenre	Project	xsd:string
hasProjectName	Project	xsd:string
hasProjectReelCount	Project	xsd:int
hasProjectSeriesNumber	Project	xsd:int

hasProjectTitle	Project	xsd:string
hasPropID	Prop	xsd:string
hasPropName	Prop	xsd:string
recordCreatedOn	Record	xsd:dateTimeStamp
hasRentalEndDate	Rental	xsd:dateTimeStamp
hasRentalStartDate	Rental	xsd:dateTimeStamp
hasSceneCode	Scene	xsd:string
hasSceneCuts	Scene	xsd:string
hasSceneDeliveries	Scene	xsd:string
hasSceneDescription	Scene	xsd:string
hasSceneLaunches	Scene	xsd:string
hasSceneName	Scene	xsd:string
hasSceneNameAlternative	Scene	xsd:string
hasSceneNumber	Scene	xsd:int
hasSceneScriptLocation	Scene	xsd:string
hasSceneSequenceLink	Scene	xsd:string
hasSceneSequencePrefix	Scene	xsd:string
hasSceneStatus	Scene	xsd:string
hasSceneTag	Scene	xsd:string
hasSceneType	Scene	xsd:string
hasSceneVersionLink	Scene	xsd:string
sceneCreatedOn	Scene	xsd:dateTimeStamp
sceneUpdatedOn	Scene	xsd:dateTimeStamp
hasShootCode	Shoot	xsd:string
hasShootDeliveries	Shoot	xsd:string
hasShootDescription	Shoot	xsd:string

hasShootEnvironment	Shoot	xsd:string
hasShootLocationDescription	Shoot	xsd:string
hasShootNameAlternative	Shoot	xsd:string
hasShootScriptLocation	Shoot	xsd:string
hasShootSequence	Shoot	xsd:string
hasShootStatus	Shoot	xsd:string
hasShootTag	Shoot	xsd:string
shootCreatedOn	Shoot	xsd:dateTimeStamp
shootUpdatedOn	Shoot	xsd:dateTimeStamp
hasShootDayDate	ShootDay	xsd:dateTimeStamp
hasShootDayNumber	ShootDay	xsd:int
hasShootDayTimeZone	ShootDay	xsd:string
hasShotNumber	Shot	xsd:int
hasCameraHead	Slate	xsd:string
hasCameraHeight	Slate	xsd:string
hasCameraMount	Slate	xsd:string
hasCameraMove	Slate	xsd:string
hasCameraNav	Slate	xsd:string
hasCameraReportNumber	Slate	xsd:int
hasCameraTrackT/O	Slate	xsd:string
hasSlateAspectRatio	Slate	xsd:string
hasSlateDataOp	Slate	xsd:string
hasSlateDataOpChecklist	Slate	xsd:string
hasSlateDeliveries	Slate	xsd:string
hasSlateDescription	Slate	xsd:string
hasSlateDuration	Slate	xsd:string

hasSlateElement	Slate	xsd:string
hasSlateFocusUnit	Slate	xsd:string
hasSlateFullCameraList	Slate	xsd:string
hasSlateHDRI	Slate	xsd:string
hasSlateHeightUnit	Slate	xsd:string
hasSlateISO	Slate	xsd:string
hasSlateKeyMeasure	Slate	xsd:string
hasSlateName	Slate	xsd:string
hasSlateNameAlternative	Slate	xsd:string
hasSlatePlanning	Slate	xsd:string
hasSlatePrevis	Slate	xsd:string
hasSlateStatus	Slate	xsd:string
hasSlateStereoConvergence	Slate	xsd:string
hasSlateStereoIO	Slate	xsd:string
hasSlateStereoType	Slate	xsd:string
hasSlateStereoUnderOverSlung	Slate	xsd:string
hasSlateStock	Slate	xsd:string
hasSlateStoryboard	Slate	xsd:string
hasSlateSunPosition	Slate	xsd:string
hasSlateTag	Slate	xsd:string
hasSlateTakeCopies	Slate	xsd:string
hasSlateTime	Slate	xsd:dateTimeStamp
hasSlateTimecodeIn	Slate	xsd:string
hasSlateTimecodeOut	Slate	xsd:string
hasSlateVersionLink	Slate	xsd:string
hasSlateVFXDescription	Slate	xsd:string

hasSlateWeather	Slate	xsd:string
slateCreatedOn	Slate	xsd:dateTimeStamp
slateHasVFX	Slate	xsd:boolean
slateUpdatedOn	Slate	xsd:dateTimeStamp
hasSoundRollNumber	SoundRoll	xsd:int
hasStaffEmailAddress	Staff	xsd:string
hasTakeNumber	Take	xsd:int
hasUnitDescription	Unit	xsd:string
hasUnitName	Unit	xsd:string
hasUnitNameAlternative	Unit	xsd:string
hasUnitStatus	Unit	xsd:string
hasUnitTag	Unit	xsd:string
unitCreatedOn	Unit	xsd:dateTimeStamp
unitUpdatedOn	Unit	xsd:dateTimeStamp

Table A.3: Usable data type properties in the Creative Data Ontology.

A.4 Ontology Visualisation

A visualisation of the Creative Data Ontology is shown below. This image was created using the WebVOWL tool and although it is not meant to be legible, it has been included to illustrate the size and complexity of the ontology.

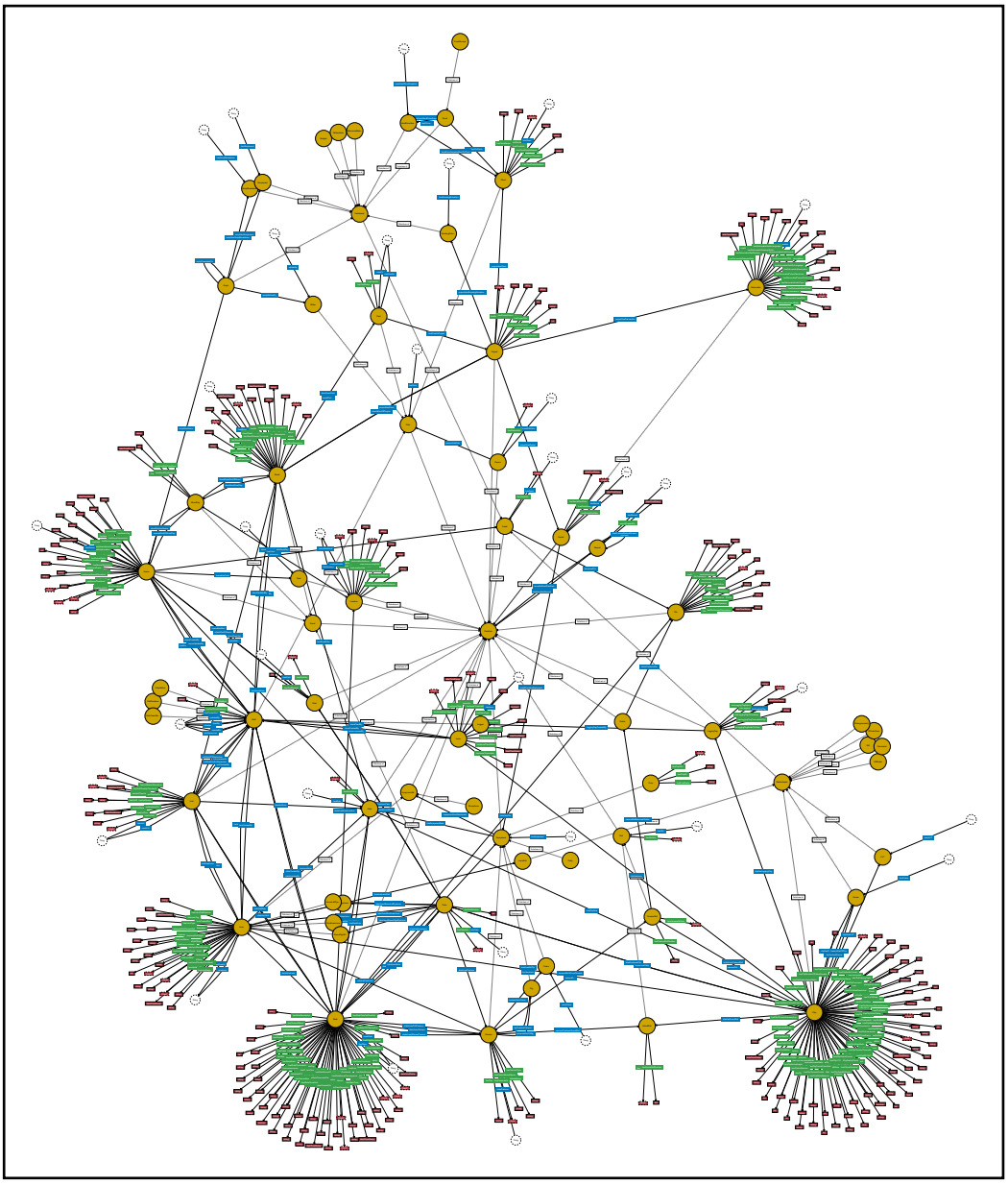


Figure A.1: A visualisation of the Creative Data Ontology.

A.5 Ontology Deployment

The Creative Data Ontology cannot be used for metadata management as is. It must be integrated into a software application. There are two high-level approaches to doing this:

1. connect the ontology to an existing software application, or

2. create a new software application designed around the idea of using the ontology.

The first approach is likely to be preferred by industry practitioners because it does not involve changing their current systems or providing additional resources associated with this, such as costs of development and time allocated to training staff. However, it is important to acknowledge that their existing software applications may have a proprietary license. Therefore, it may not be possible to integrate the ontology parser directly into the code base so we shall instead focus on the second option: creating a new software application from scratch.

There is no universal way of creating a new software application that is centred around using the ontology. Because of this, we shall provide a high-level overview of the architecture of such an application, with additional detail provided to explain how to connect the ontology to the rest of the software components. Figure A.2 shows the high-level components that should make up the software application:

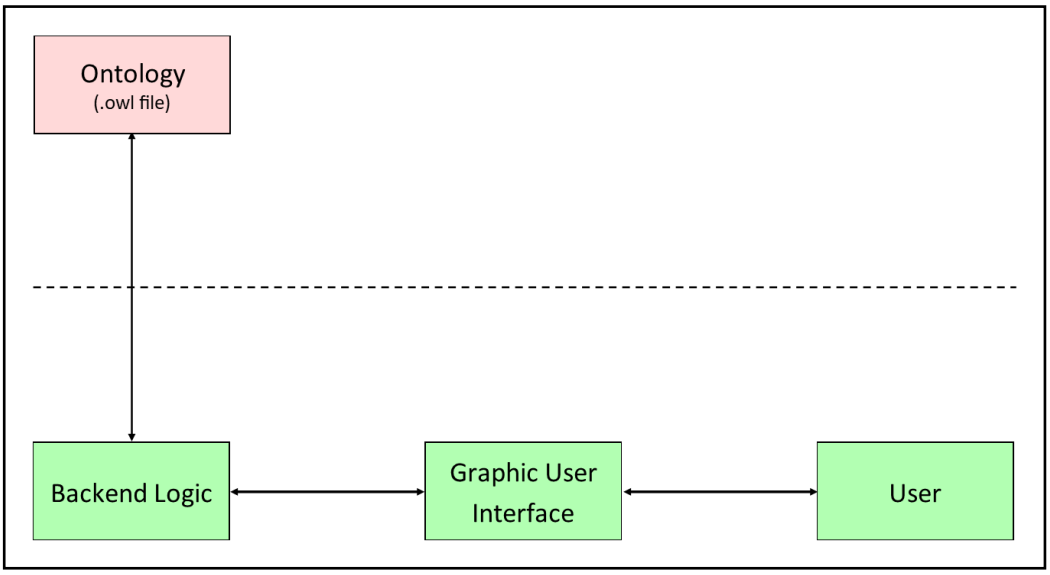


Figure A.2: Architecture of an application integrated with the Creative Data Ontology.

It is extremely likely that the users of this software application with interact with it via a Graphical User Interface (GUI) rather than a command line interface. The user can interact with the GUI components such as by clicking on buttons and entering information into textboxes in order to submit commands to the backend. The backend then processes these commands by querying the ontology and its associated instance data to either insert, modify, or retrieve instance data. The result is then displayed to the user via the GUI.

Connecting the Ontology to the Backend

We can use code packages to connect the backend of the software application to the Creative Data Ontology. For example, if the software application is to be built in Python, we can use the OWLReady2 package. OWLReady2 can load OWL ontologies as Python objects, modify them, save them, and perform reasoning via the HermiT reasoner, which is included in the library [131]. The links to download OWLReady2 and to access its documentation are:

Download: `https://pypi.org/project/Owlready2`

Documentation: `https://owlready2.readthedocs.io/en/latest`

An alternative Python package for loading OWL ontologies is RDFLib. This package can also modify, save, and query ontologies, although the OWLReady2 documentation states that it has a more optimised RDF store than RDFLib [132]. The links to download RDFLib and to access its documentation are:

Download: `https://rdflib.dev`

Documentation: `https://rdflib.readthedocs.io/en/stable`

Using these libraries, we can query the Creative Data Ontology and insert, modify, or retrieve instance data from the OWL file via a Python backend. This data can then be displayed to the user via the GUI. The user experiences the benefits of using the ontology while being shielded from its technical details.

Appendix B

Ontology Evaluation Slide Decks

This appendix contains the two slide decks used to support the ontology evaluation interviews, the results of which are presented in Chapter 7. The first slide deck was used in interviews with practitioners from the media industry (domain experts) and the second slide deck was used with ontology experts.

Creative Data Ontology Evaluation Pack

Ontology Version 5.4.21

https://www.cs.rhul.ac.uk/home/zcva113/ontology-eval/

StoryFutures

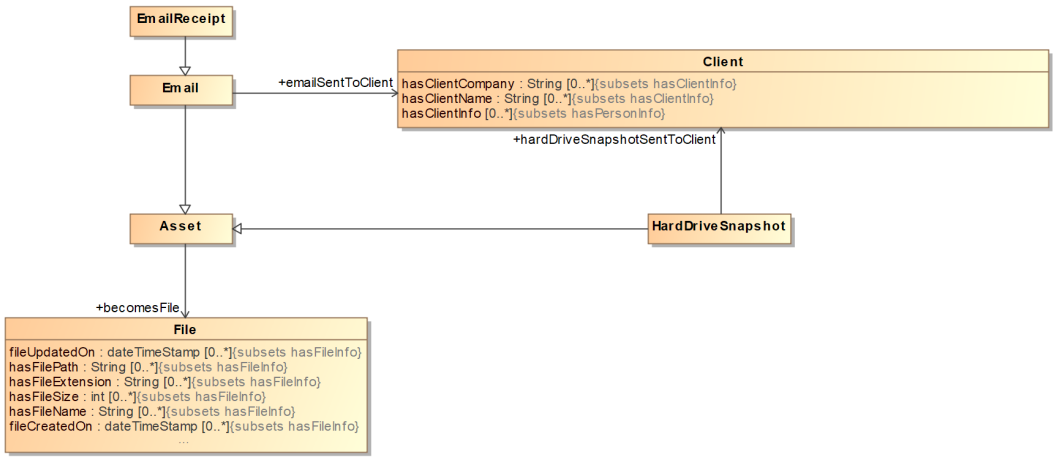
Representation of Use Cases

Use Case 1:

Unscripted: Media Support Workflow

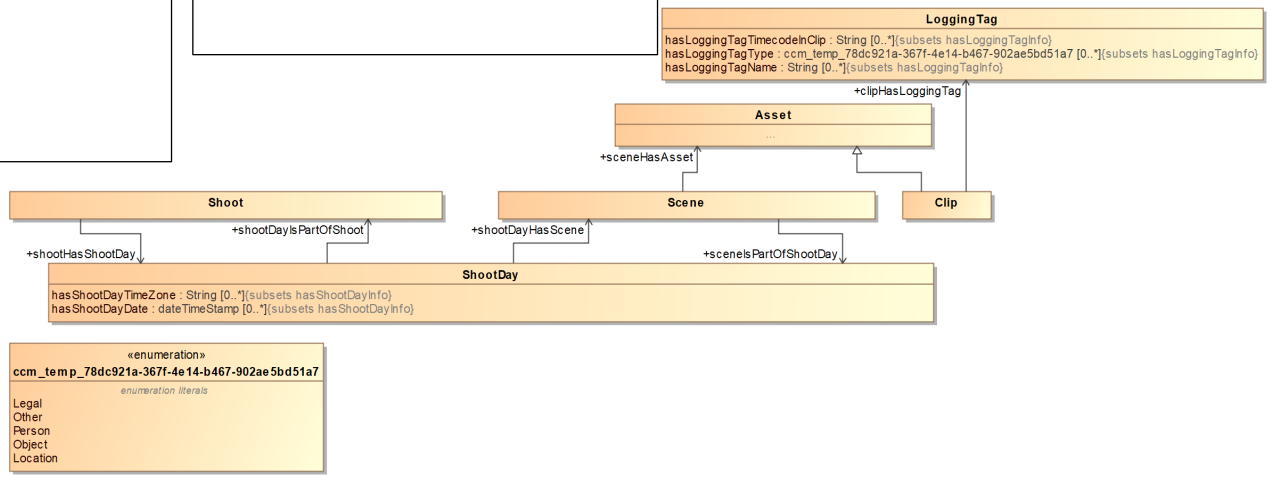
Completeness
Does the ontology have all the classes and properties required to represent the use case? If not, which ones are missing?

Accuracy
Have the classes been connected correctly? Have the fields been attached to the correct classes?



Use Case 2:
Unscripted: Live Logging

Completeness	*Accuracy*
Does the ontology have all the classes and properties required to represent the use case? If not, which ones are missing?	Have the classes been connected correctly? Have the fields been attached to the correct classes?



Use Case 3:

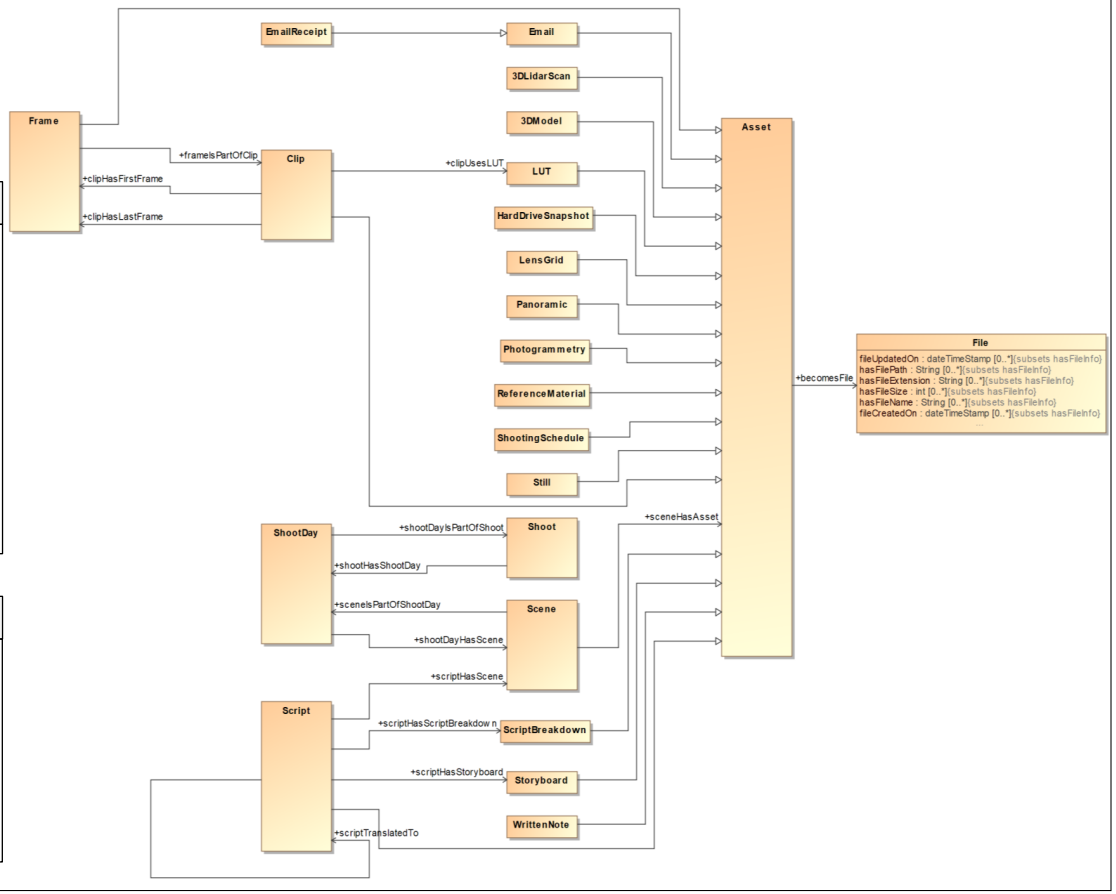
VFX: On-Set Shoot Data

Completeness

Does the ontology have all the classes and properties required to represent the use case? If not, which ones are missing?

Accuracy

Have the classes been connected correctly? Have the fields been attached to the correct classes?



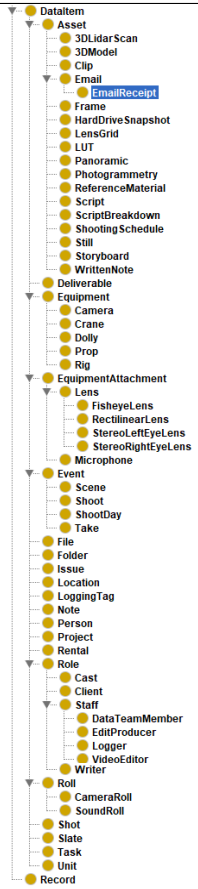
The Ontology as a Whole

The Classes

Accuracy

In an ontology, concepts are arranged in a class hierarchy where a lower level class is a specialisation of its upper level class.

Does the class hierarchy look correct and if not, what changes should be made?



Completeness

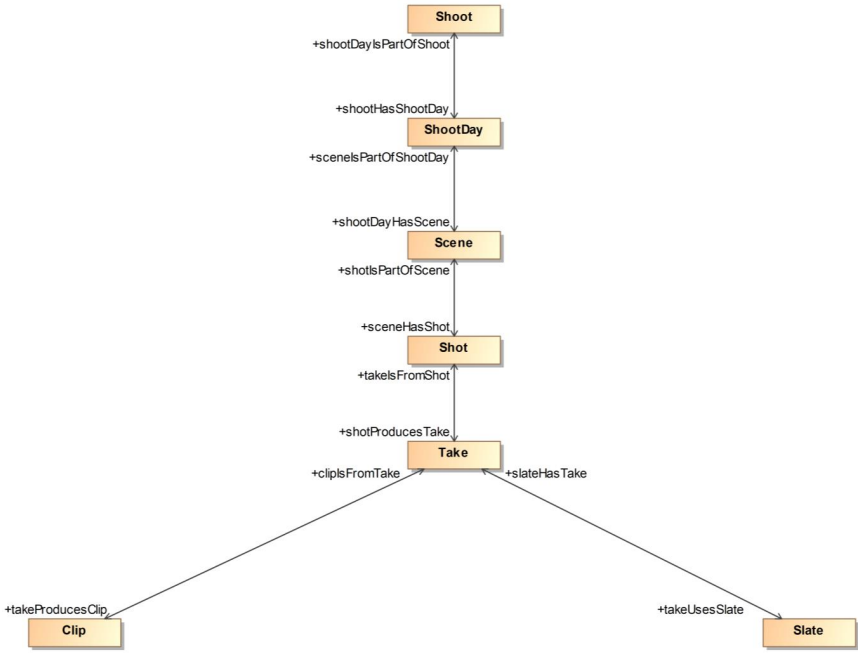
Are there any classes missing from the ontology? If yes, which ones?

Conciseness

Do you consider any of the classes within the ontology irrelevant or unnecessary? If so, which ones and why?

Chaining of Classes

Accuracy
During ontology development, the relationships between the classes in the diagram were unclear, specifically the order of the classes in the chain. Are these classes chained in the correct order and if not, what is the correct order?



Clarity of Definitions

Clarity
The following link contains the definitions for all classes, relations, and attributes in the ontology involved in the three use cases: *https://www.cs.rhul.ac.uk/home/zcva113/ontology-eval/list-files/all-definitions-list.pdf* Are these definitions clear and accurate? If not, which definitions are unclear or inaccurate? Do any of these definitions appear to overlap? Is so, which ones?

Clarity
Do you think most domain experts would understand this ontology quickly? If not, why? How could the clarity of this ontology be improved?

Summary

General Feedback

What did you think of the ontology overall?

What difference could this ontology make to your business?

General Feedback

How could the ontology be improved?

How could the ontology be extended? Could it be expanded to include, for example, virtual production or immersive subdomains?

Please give the ontology a score out of 10 for each of the following criteria:

Accuracy:	/10
Completeness:	/10
Conciseness:	/10
Clarity:	/10

Creative Data Ontology Evaluation Pack

Ontology Version 5.4.21

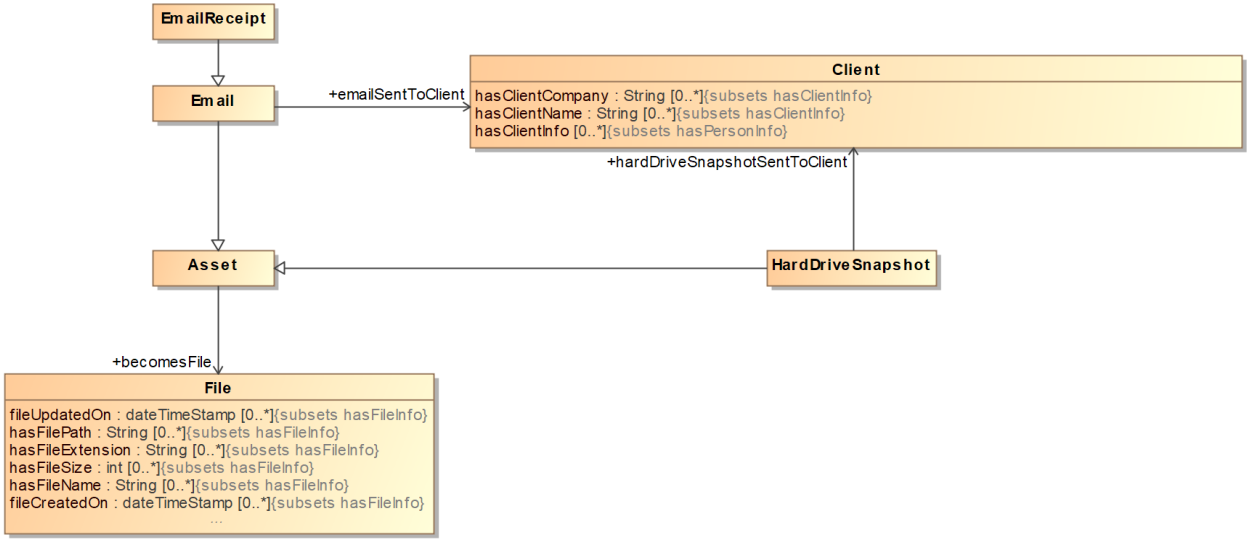
https://www.cs.rhul.ac.uk/home/zcva113/ontology-eval/

StoryFutures

Use Case 1:

Unscripted: Media Support Workflow

Media support operatives prepare the data received from production clients, ready for use in the later stages of post-production. Workflows in media support begin with accepting media from the in-house librarian and sending a receipt to clients containing a hard drive snapshot of the contents of the drives delivered. The operative then makes a copy of this data, conforms it according to the desired format, and finally ingests it into the workflow.



Use Case 1:
Unscripted: Media Support Workflow

Clarity
Is the ontological representation of this use case clear and intuitive? If not, how could clarity be improved?

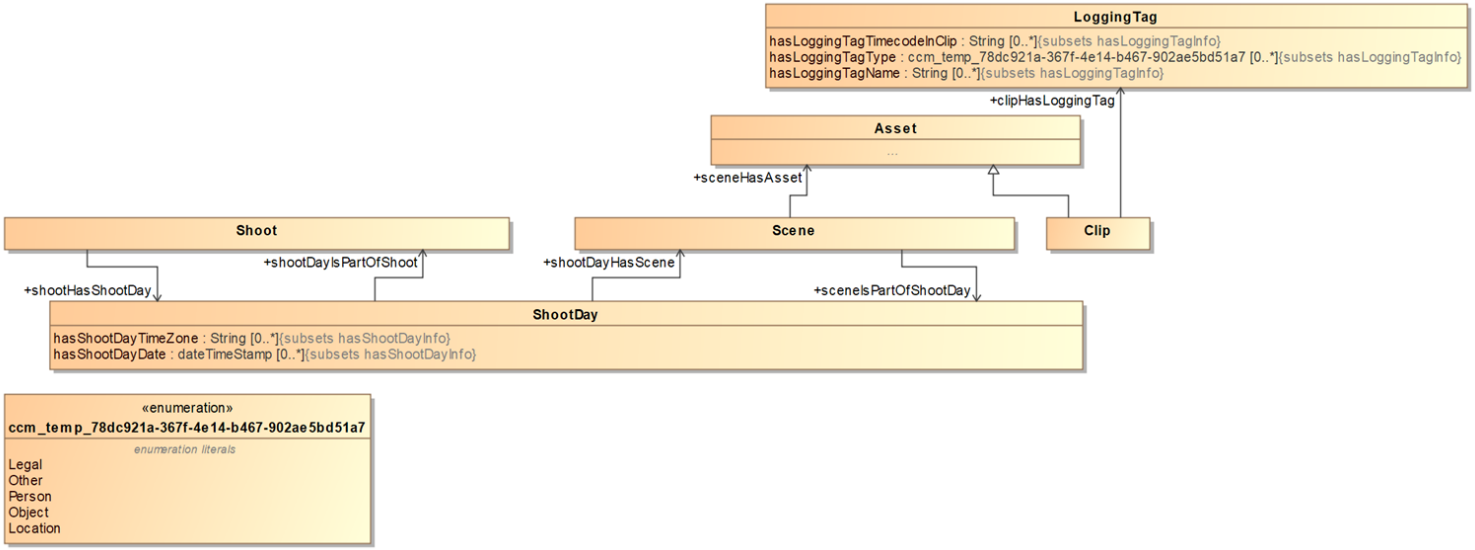
Conciseness
Is the ontological representation of this use case concise? If not, how could conciseness be improved?

General
Do you have any further comments on the ontological representation of this use case?

Use Case 2:

Unscripted: Live Logging

Unscripted and documentary content requires the addition of descriptive tags to the unedited footage in order to be able to find and edit content in vast collections of data captured on camera rigs set up on location. This process is referred to as 'live logging', where logging staff write annotations into a bespoke timestamped tool and the aim is to record the what the footage depicts. The post-production staff and editors are then able to search for, recover and create edits with specific content.



Use Case 2:
Unscripted: Live Logging

Clarity
Is the ontological representation of this use case clear and intuitive? If not, how could clarity be improved?

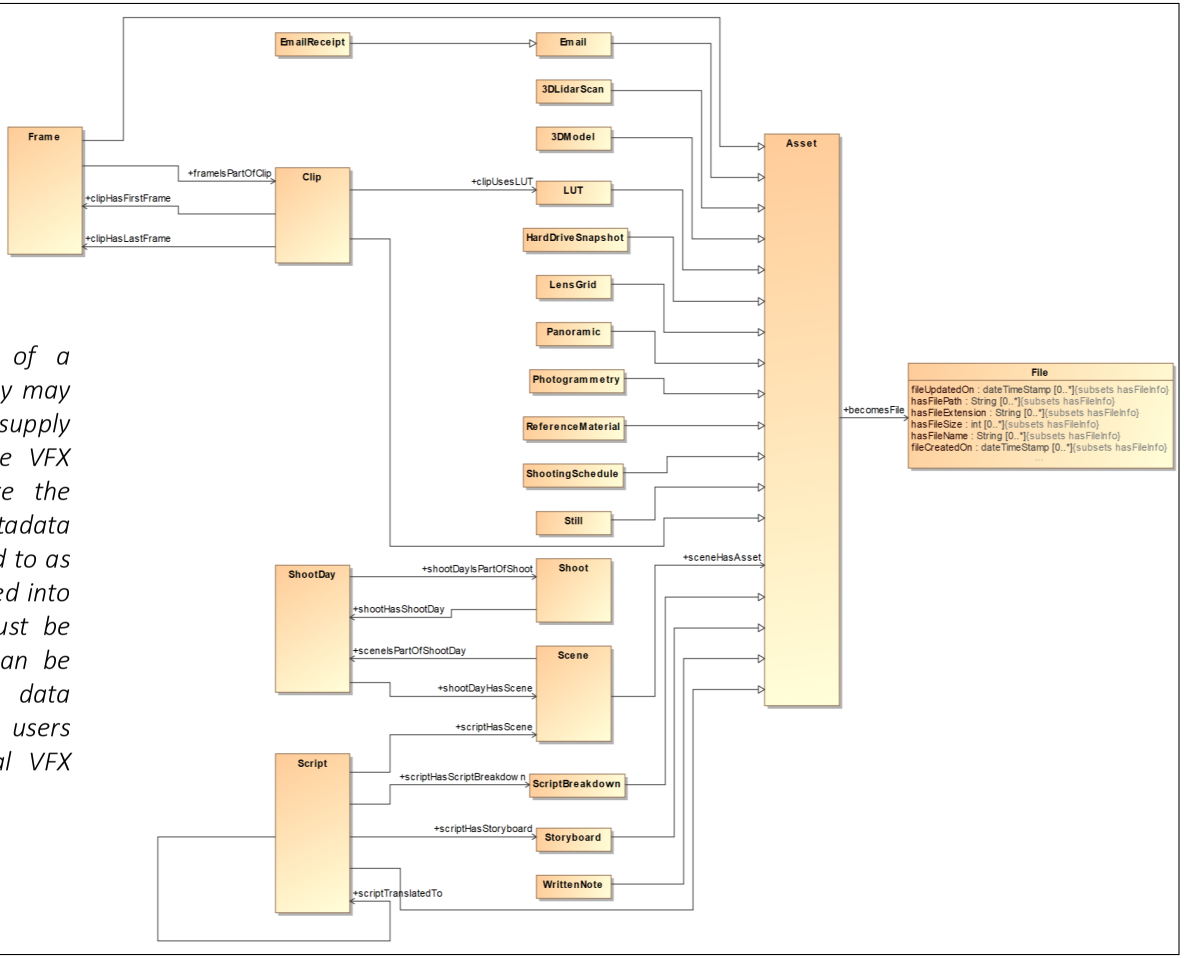
Conciseness
Is the ontological representation of this use case concise? If not, how could conciseness be improved?

General
Do you have any further comments on the ontological representation of this use case?

Use Case 3:

VFX: On-Set Shoot Data

During the pre-production phase of a feature film, the production company may secure one or more vendors to supply visual effects. This may involve the VFX company being on set to acquire the additional data and associated metadata required for VFX in a process referred to as `Shoot'. This metadata is then collated into an on-set VFX database, which must be organised in such a way that it can be passed on to the larger central data management database for users throughout the companies' internal VFX pipelines.



Use Case 3:
VFX: On-Set Shoot Data

Clarity
Is the ontological representation of this use case clear and intuitive? If not, how could clarity be improved?

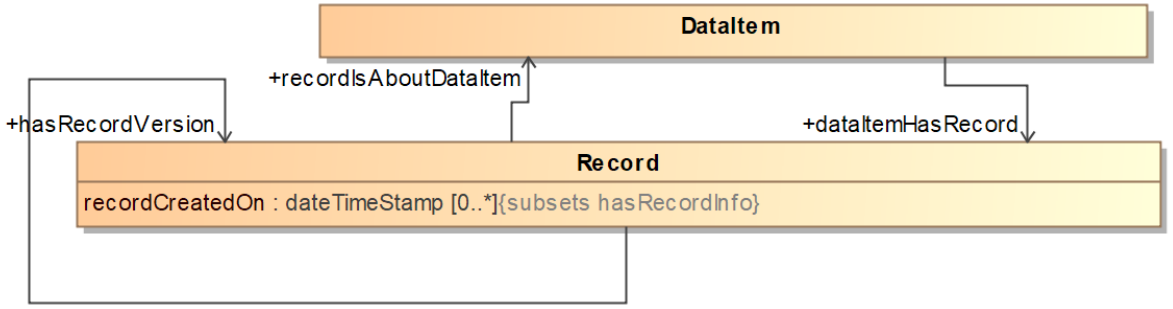
Conciseness
Is the ontological representation of this use case concise? If not, how could conciseness be improved?

General
Do you have any further comments on the ontological representation of this use case?

Use Case 4:
Metadata Versioning

Metadata versioning has been implemented using principles that are analogous to Russian dolls. The ontology has two top-level classes aimed at linking metadata and connecting records that describe the transitions that metadata typically undergoes: DataItem and Record. The former is an umbrella class for all entities in the ontology that can have metadata associated with it whilst the Record class encapsulates versioning metadata, including temporal details about a data item.

To enable metadata versioning, every instance of DataItem and its subclasses should have an instance of Record assigned to it upon creation and when the metadata of a DataItem instance is changed, a copy of that instance of DataItem is made with the metadata adjustments applied and a new Record instance is also created for the new DataItem instance. The two instances of Record – the original and the updated – are connected using the object property hasRecordVersion.



Use Case 4:
Metadata Versioning

Clarity
Is the ontological representation of this use case clear and intuitive? If not, how could clarity be improved?

Conciseness
Is the ontological representation of this use case concise? If not, how could conciseness be improved?

General
Do you have any further comments on the ontological representation of this use case?

General Feedback

What did you think of the ontology overall?

How could the ontology be improved?

Please give the ontology a score out of 10 for each of the following criteria:

Conciseness:	/10
Clarity:	/10