



# Collaborative Annotation, Search and Categorisation

## T H E S I S

Submitted for the degree of

**Doctor of Philosophy**

by

Yi Hong

**DEPARTMENT OF COMPUTER SCIENCE**

June, 2016



## Declaration

I declare that this dissertation is the product of my own work, that it has not been submitted before for any degree or examination in any other university, and that all the sources I have used or quoted have been indicated and acknowledged as complete references.

Research work presented in some chapters has been previously published. In particular, the basic idea about collaborative annotation in Chapter 3 has been published in SSW2011 [1], the ECA-based transformation approach from relational database to ontological database in the domain of archaeology discussed in Chapter 3.2 was published in CAA2010 [2]. The idea of graph-based search and reasoning in Chapter 4 has been published in ICGT08 [3], CONTEXT07 [4] and CAA2011 [5]. Evaluation of collaborative categorisation and related applications were published in [105]. There are some further publications which are referred in each chapter.

Signature \_\_\_\_\_



## Acknowledgments

First of all, I would like to thank my supervisor Stephan Reiff-Marganiec for his continual support and the advices he has given me in the writing of this report. I would also like to thank José Fiadeiro and all project members from Leverhumle-fund Tracing Networks Programme, for their generous help in data provision and their constructive suggestions.

A special thanks to my wife Xiaoyan Ren, for her love and for always being here for me.





## Abstract

The purpose of this research is to develop a collaborative framework for annotation, search and categorisation. The basis of this research is to define an ontology-based data model that allows users to create semantic tags, establish relationships among tags and provide contextual information by hierarchical concepts and properties structure derived from a lexical knowledge base. A computational model is introduced to record uncertainty, establish user credibility and compute the truthfulness or reliability of the statements, which can then be used for ranking search results. A method to transform a relational database to the ontology-based repository is developed to populate the proposed data model. The second stage of the research is to develop an expressive yet intuitive querying technique for searching semantically annotated data. There are many questions that arise when querying complex datasets. For example, how to help average non-tech users to write queries without excessive reliance on external technical support? How to utilise a rich knowledge base and how to enable members of a collaborative team to construct queries collectively, considering their opinions on the importance of searching criteria? Traditional keyword-based or form-based approaches fail to address these issues due to lack of expressive power or flexibility. A visual querying technique is presented for the collaborative team, based on graph pattern matching. This method allows members of a collaborative team to collectively construct complex queries in a more convenient manner. Then the possibility of applying various categorisation techniques to help sort annotated objects is investigated. A new workflow model is proposed that help a collaborative team build a universally-accepted categorisation system and develop a systematic way for team members to create a training data set, taking into account various criteria and degrees of uncertainty in human decision-making. Eventually, a modified Naive Bayes classifier was built for storing a large number of objects. In the end, in collaboration with members of an archaeological research team, a series of experiments was conducted to evaluate our methodologies.

---



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Problems and Challenges . . . . .	3
1.1.1	Annotation . . . . .	4
1.1.1.1	Semantic Expressivity in Annotation . . . . .	6
1.1.1.2	Uncertainty of Annotations . . . . .	7
1.1.1.3	Reliability of Data Sources . . . . .	8
1.1.1.4	Model Extensibility and Interoperability with Legacy System . . . . .	8
1.1.2	Searching and Ranking . . . . .	9
1.1.2.1	Needs for Collaborative Search . . . . .	9
1.1.2.2	Usability of Query Interface . . . . .	11
1.1.2.3	Insufficient Automated Reasoning Support . . . . .	13
1.1.2.4	Combining Weighted Criteria in WHERE Clauses . . . .	14
1.1.3	Collaborative Categorisation . . . . .	15
1.1.3.1	Universal Categorisation Generation . . . . .	16
1.1.3.2	Ambiguous Class Definition . . . . .	16
1.1.3.3	Complexity of Multi-criteria Group Decision-making Process . . . . .	17
1.2	Thesis Statement . . . . .	17
1.3	Research Scope and Contributions . . . . .	17
1.4	Thesis Overview and Summary . . . . .	19
<b>2</b>	<b>Background and Literature Review</b>	<b>21</b>
2.1	Collaborative Team Formation . . . . .	21
2.1.1	Focus Groups: Tracing Networks Programme . . . . .	22
2.2	Methods for Collecting Qualitative Data . . . . .	23
2.2.1	Unstructured Free Text . . . . .	23
2.2.2	Structured and Semi-structured Form . . . . .	24

2.2.3	Keyword-based Tagging . . . . .	25
2.2.4	Related work . . . . .	28
2.3	Storage and Archiving Solutions for Collected data . . . . .	30
2.3.1	RDBMS (Relational Database Management Systems) . . . . .	30
2.3.2	Problems with RDBMS . . . . .	30
2.3.3	XML documents . . . . .	34
2.3.4	RDF Triplestores and Ontology-based repositories . . . . .	35
2.3.5	Tracing Networks Ontology . . . . .	37
2.4	Mapping RDB Schema to RDFS/OWL . . . . .	39
2.5	Usability of Search Interfaces . . . . .	42
2.5.1	Single Textbox with Autocomplete Functionality . . . . .	42
2.5.2	Filter-based Search Interface . . . . .	43
2.5.3	Native Query Statement . . . . .	43
2.5.4	Related Work . . . . .	44
2.6	Uncertainty Models . . . . .	45
2.6.1	W3C XG on Uncertainty Reasoning . . . . .	46
2.6.2	Fuzzy Description Logic and RuleML . . . . .	46
2.6.3	PR-OWL: A Bayesian Ontology Language . . . . .	47
2.7	Reputation Models . . . . .	47
2.7.1	WOM-based eBay/Amazon models . . . . .	48
2.7.2	Sporas and Histos . . . . .	49
2.8	Categorisation Techniques . . . . .	50
2.8.1	Description Logic A-Box Classification . . . . .	50
2.8.2	Clustering . . . . .	51
2.8.3	Support Vector Machines . . . . .	51
2.8.4	Naive Bayes Classifier . . . . .	51
2.8.5	Neural Networks . . . . .	52
2.8.6	Social Annotation-classification tool: Zooniverse . . . . .	52
<b>3</b>	<b>Collaborative Annotation</b>	<b>55</b>
3.1	Ontology-based Data Model . . . . .	55

3.1.1	Collaborative Annotation Ontology . . . . .	55
3.1.2	Uncertainty of the Assumptions . . . . .	57
3.1.3	Domain-Specific User Credibility Measurement . . . . .	59
3.1.3.1	User Credibility Measurement Based on User Activities	60
3.1.3.2	Extracting Contributor Reputation From Peer Reviews	62
3.1.3.3	Combining Two Reputation Modules . . . . .	64
3.2	Integrating Existing Dataset . . . . .	65
3.2.1	Problems with Mapping . . . . .	65
3.2.2	Transformation Framework . . . . .	68
3.2.3	ORM Reversed Engineering . . . . .	69
3.2.4	ECA Rule-based Transformation . . . . .	69
3.2.5	Instances Generation . . . . .	71
<b>4</b>	<b>Search and Reasoning</b>	<b>75</b>
4.1	Graph-based Query Pattern and Colour Scheme . . . . .	75
4.1.1	Illustrative Examples: House of Menander (Pompeii) and Loom- weight (Metaponto) datasets . . . . .	80
4.1.1.1	Basic Triple patterns . . . . .	80
4.1.1.2	Aggregate Function . . . . .	81
4.1.1.3	Conditions and Filters . . . . .	82
4.1.1.4	Pattern alternatives . . . . .	84
4.1.1.5	Predicates as variables . . . . .	85
4.1.2	Implementation - Graph to Query transformation . . . . .	86
4.2	Deductive Rule-based Reasoning by Graph Transformation . . . . .	89
4.2.1	Generic Reasoning Rules . . . . .	93
4.2.2	Custom Reasoning Rules . . . . .	98
4.3	Personalised and Collaborative Search with Multiple Objectives . . . . .	99
4.3.1	Collective Weights for Triple Patterns . . . . .	100
4.3.2	Topology-based Weightings . . . . .	102
4.4	Ranking . . . . .	104

<b>5</b>	<b>Collaborative Categorisation</b>	<b>105</b>
5.1	Categorisation Through Social Tagging and Clustering . . . . .	105
5.1.1	Calculating Semantic Relatedness Using Lexical database . . . . .	106
5.1.2	Grouping with Unsupervised Clustering . . . . .	106
5.1.3	Faceted Browsing . . . . .	109
5.2	Categorisation Through Supervised Learning . . . . .	110
5.2.1	Fuzzy Multiple-criteria Collaborative Categorisation (FMCCC) . . . . .	111
5.2.1.1	MCCC Conceptual Models . . . . .	112
5.2.1.2	MCC Mathematical Model . . . . .	113
5.2.1.3	MCCC Process . . . . .	113
5.2.1.4	Proposed Approach . . . . .	113
5.2.2	Collaborative Classification Ontology . . . . .	115
5.2.2.1	Raw Data Entry . . . . .	117
5.2.2.2	Selection of Training Dataset . . . . .	117
5.2.2.3	Categories Generation Process . . . . .	118
5.2.2.4	Manually Classify Sample Data Sets . . . . .	123
5.2.3	Criteria Weighting for Single Decision-maker . . . . .	124
5.2.3.1	Combining Uncertainties from Multiple Decision-makers . . . . .	126
5.2.3.2	User Credibility Extraction . . . . .	128
5.2.4	Training Modified Naive-bayes Classifier . . . . .	128
5.2.5	Applying the Classifier . . . . .	132
<b>6</b>	<b>Application and Evaluation</b>	<b>135</b>
6.1	Evaluation and Comparison: KR and Uncertainty Models . . . . .	135
6.2	Evaluation and Comparison: Querying and Ranking Methods . . . . .	136
6.3	Collaborative Image Annotation Framework for Human Representation . . . . .	139
6.4	Evaluation: Graph-based Query Language . . . . .	140
6.4.1	Usability and Learnability . . . . .	141
6.4.2	User Experience Assessment . . . . .	147
6.4.3	Adaptability and Expressiveness . . . . .	148
6.5	Evaluation of Proposed Categorisation Method . . . . .	152

---

6.5.1	Semi-automatic Sense Detection for Ancient Art . . . . .	152
<b>7</b>	<b>Conclusions and Future Work</b>	<b>157</b>
7.1	Summary of Research . . . . .	158
7.2	Future work . . . . .	159
7.2.1	Application: Mobile Social Annotation for Museums . . . . .	162
7.2.2	Application: Graphical Modelling Tool for DOTL . . . . .	164
<b>A</b>	<b>Appendix</b>	<b>171</b>
A.1	Terminology for Archaeologists . . . . .	171
A.2	Tracing Networks Ontology (OWL-DL) . . . . .	173
A.3	Worked Example of Credibility Measurement . . . . .	174
A.4	Artifact Gallery . . . . .	175
	<b>Bibliography</b>	<b>177</b>



---

## Abbreviations and Glossary

**Ontology** “A formal specification of conceptualization”

**OWL** Web Ontology Language.

**RDF** Resource Description Framework

**SPARQL** SPARQL Protocol and RDF Query Language

**CWE** Collaborative Working Environment

**DOTL** Database to Ontology Transformation Language

**MCCC** Multi-criteria collaborative classification

**CF** Certainty Factor

**CCF** Certainty-Credibility Factor

**CCO** Collaborative Classification Ontology

**CIDOC-CRM** Conceptual Reference Model (CRM)





# Introduction

---

In recent years, the amount of available data has increased significantly leading to a problem of “information overload” in the sense that finding what one is looking for becomes hard. Much research has been done regarding knowledge representation and storage. How to store and organise information in a machine-understandable manner is a crucial step when dealing with data from various sources. Tagging or annotating has become a popular way of adding searchable information to the resource, in particular, for images resource. A tag is a small piece of plain text or some keywords attached to an item. It helps users in organising and searching content collections. In a shared environment such as social networking websites, tagging has become a popular activity. However, it raises a challenging question about how to structure the metadata to build an expressive data model capable of semantically annotating items in a more flexible way and with more expressive power than conventional simple keyword-based tagging. Some relevant background techniques are discussed in Chapter 2, the work was also presented in the Digital Humanities Colloquium (Milton Keynes, 2011)[7]

Chapter 3 introduces a novel collaborative framework for annotation using the semantic web technologies [8] and ontologies [9]. The framework allows users to create tags that are based on a concept repository, which provides a hierarchical context for the tags and allows us to define relationships among said concepts. It also provides a new and systematic way to establish user credibility as well as to compute the truthfulness or reliability of statements in a collaborative working environment. This part of the research has been published in the Proceedings of SSW2011 [1] and DE2011 [10].

Since the approach takes advantage of semantic web techniques and an ontology-based data model, this raises another challenging question in finding a solution for mapping legacy dataset to the ontology. This process can be cumbersome, in particular

when the relation between table and class is far more complex than a simple one-to-one correspondence. Chapter 3.2 investigates the feasibility of transforming relational database to an ontology-based repository by means of Event-Condition-Action rules in contrast to a simple one-to-one class-to-table mapping. The research in this field has been published in the proceedings of CAA2010 [2].

Once semantically-enriched data has been collected through the annotation process, the next step would be to provide a means for ordinary non-tech users to search and retrieve information stored in the ontology-based repository. An important aspect of research is to provide a better search experience for end-users. Simple keyword search is easy and effective, but with rather poor expressive power while approaches using conventional semantic structured query language (like SPARQL) is expressive, but lacks flexibility and ease of use. Chapter 4 discusses the feasibility of using graphical patterns for end-users for searching annotated data sets. This part focuses primarily on how to improve the usability of the search interface for non-technical team members and to make user experience and interaction with the system as simple and efficient as possible, while providing the maximum expressiveness. Further the idea to improve the result of search by applying intelligent ranking and clustering techniques is presented, such that search results can be displayed in a more meaningful fashion. The work in this part has been published in ICGT08[3], CONTEXT07 [4] and CAA2011 [5].

Classifying annotated data is another common task in everyday life. Most classification methods group similar objects together based on common characteristics rather than a formal definition of the category. Decision-making for classification is a rather complex process. Many questions have been raised, particularly around how to classify multi-dimensional data in an efficient and accurate manner. Generally, a team consisting of a variety of members is often more capable of making critical decisions than individuals. Hence, the problem of object classification nowadays usually requires a group of people to work collectively, group decision-making becomes increasingly common in many organisations and collaborative team. However, in a collaborative environment, people might see the same classification problem from their own perspective and subsequently sort objects according to their own understanding and categorisation system. The problem becomes even more complicated when some degree of uncertainty is in-

volved. Chapter 5 discusses a semi-automatic framework to assist collaborative teams to categorise objects. The framework allows a distributed team to categorise a small number of sample objects collectively based on team members' own perception. Decisions made by individuals along with associated degree of uncertainty and fuzziness in decision-making process are captured in our ontology-based data model and later used as training set to build an extended Naive-Bayes classifier, which can be used for automatic classification of a larger dataset. Research evaluation regarding this part has been conducted at a British Academy outreach event (April 23rd, London) [11]. The research outcome was subsequently published in [105].

Several prototype applications have been implemented to evaluate our methodology. This research is partially supported by an archaeological research project – “Tracing Networks: Craft Traditions in the Ancient Mediterranean and Beyond” [12]. Most of them have been evaluated by archaeologists in relation to this project. Our experimental results show the ontology-based data model developed is capable of handling complex archaeological data.

In terms of ease of use, the annotation and graphical query system developed received positive feedback from the research teams from the Tracing Networks Programme working in the “Loomweight: weaving relationships” project led by Professor Lin Foxhall and Dr Alessandro Quercia. Although work still needs to be done to improve the interface. A prototype categorisation system has also been developed based on the proposed methodology to help archaeologists detect human scenes on ancient artefacts. The evaluation results show that the method can achieve reasonable performance.

While the development was driven by the demands posed in the archaeological domain, the same fundamental problems exist in other domains. Hence, the results and methods developed will readily transfer to such other domains.

## 1.1 Problems and Challenges

This section will highlight some challenges we are facing today regarding search, ranking and categorisation of annotated datasets in a collaborative environment. The problem can be explained and exemplified in real-world processes and situations in Tracing

Network Project. That is how to help archaeologists document and categorise artefacts and conduct post-excavation analysis as a collaborative team. This problem would then break down into three sub-problems:

- Part 1: Annotate artefacts and images as a team
- Part 2: Search annotated content
- Part 3: Computer-assisted categorisation

### 1.1.1 Annotation

Currently, there are a number of annotation approaches available. Generally, these are based on keywords, but there have also been some efforts centred around ontologies. This section will now highlight some problems with the keyword-based tagging approaches as well as some known issues existing in semantic-oriented annotation frameworks. To overcome the limitation of traditional tagging approaches, many semantic tagging applications have been developed e.g. [13] [14] [15]. However, these still have their limitations.

Figure 1.1 shows a ceramic vessel fragment found at Sopron-Varhely, Hungary depicting a wagon-ride scene. Archaeologists with different expertises (e.g. pottery experts, field archaeologists and possibly amateur archaeology enthusiasts with no academic background) wanted to add tags to the image to express their opinions about this unknown four-legged creature:

Person A (Field archaeologist): *“It is definitely a horse.”*

Person B (Rock art expert): *“Look like a horse, but I am not quite sure.”*

Person C (Pottery expert): *“I think it’s probably an ox, I am 70% certain it’s an ox.”*

Person D (Oracle DBA): *“It must be a fox.”*

One may choose to annotate the selected area with keywords (e.g. “horse”, “fox” and “ox” ). However, the traditional keyword-based tagging method is unable to record uncertainty associated with the keyword. For example, it cannot differentiate between

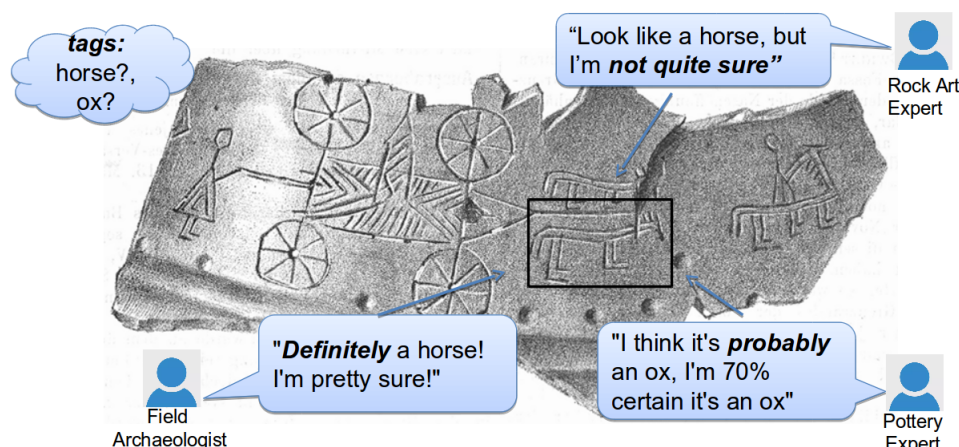


Figure 1.1: Keyword-based tagging: ceramic vessel fragment found at Sopron-Varhely depicting a wagon-ride scene)

“definitely a horse” and “probably a horse”. Archaeology research is usually based on incomplete data (for example, small fragments of pottery), hence, uncertainty is an important factor that needs to be considered

Also, it is not possible to aggregate opinions from different sources while taking into account annotators’ reputation in a specific domain. For example, Person C is considered to be more credible in this domain and therefore more important than Person D.

Figure 1.2 shows some other challenges in the traditional tagging technique. For instance, Person A and Person B wanted to make two statements.

Person A: “A *hunter* is riding *horse1* and leading *horse2*”.

Person B: “*horse2* is pulling the *waggon*”.

First of all, annotators might use different terminologies. Different terms might be used to describe what should be the same objects. For example, people might use tags such as “waggon”, “carriage”, “coach” or “cart” to describe the four-wheeled vehicle pulled by two unknown creatures. Keyword-based tagging method is an ideal choice for annotating objects because keywords are concise, but on the other hand, this makes

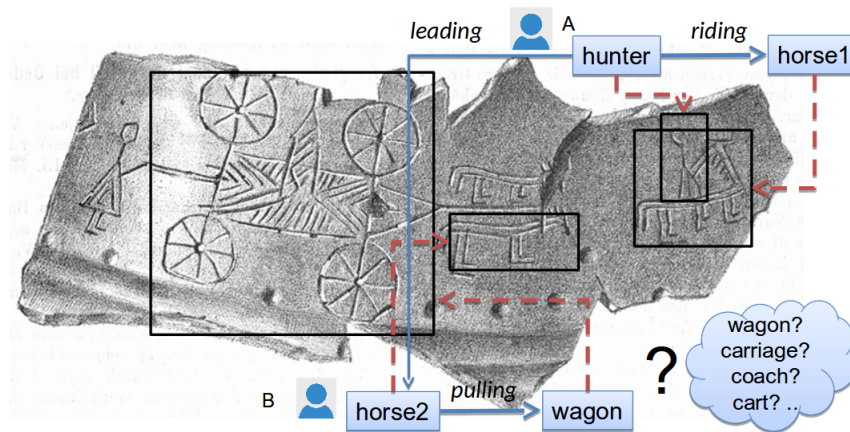


Figure 1.2: Making statements and specifying relations (ceramic vessel fragment depicting a waggonride sense)

it difficult to describe relations. Given a statement “The *hunter* is riding *horse1* and leading *horse2*”. In the case of traditional tagging, attaching a keyword “ride” to the image only describes the predicate (verb) of the statement. It does not indicate clearly the subject (“who” or “what”), nor the object (any noun that is part of the action of the subject) of such statement. Some challenges arose with keyword-based tagging:

- Represent uncertainty
- Aggregate opinions from multiple annotators
- Consider domain-specific reputations
- Semantic ambiguity
- Annotate objects as well as relationships
- Express complex statements

#### 1.1.1.1 Semantic Expressivity in Annotation

Raw data is usually annotated with simple tags in most conventional annotation frameworks. A tag is a freely-chosen, non-hierarchical keyword or term. Tags appearing in different places could be syntactically identical, but might have ambiguous meanings.

For example, the plain text “date” tagged on an image might have a different interpretation. It may refer to a day on the calendar, fruit or the image is showing someone out on a date. Because the word has several meanings and the context of its use is not clear. It is unclear what it meant. Similarly, a system is unable to tell whether the picture with tag “coach” is referring to a type of vehicle for long-distance travelling or a person involved in the training of a sports team, something entirely different.

Also, current tagging systems are focusing more on labelling objects rather than describing relationships among them, but in the real world, specifying relationships between entities are as important as identifying the entities themselves. For example, a user can label two different areas on the image and annotates them with “hunter” and “ox” respectively, but most current annotation techniques are not suitable for describing relationships between entities in a formal way. In this case, tagging is not expressive enough to describe statements such as “the hunter is chasing the fox” (but not “the fox is chasing the hunter”, or even “the Fox [Leicester F.C, an English Premier League Football Club] is in contact with Martin Hunter [Football coach at Southampton F.C]”). Since the traditional keyword approach is incapable of annotating links between objects due to its limited expressiveness in semantics, a novel semantic annotation approach is needed to address these issues.

#### 1.1.1.2 Uncertainty of Annotations

First of all, we need to distinguish uncertainty and probability. Certainty is about how certain you believe the assumption you made is correct – the uncertainty over a given statement depends entirely on your personal judgment, while probability is the estimation of the likelihood of occurrence of an event [16]. Current semantic tagging applications do not have the functionality to say how truthful or reliable a user’s statement is. Few available frameworks can be used for describing uncertainty. For example, most annotation techniques do not support statements such as “I have *strong* evidence to believe that this is a horse” or “It is *probably* a horse in the picture, but I am not sure about it”, nor a way to express these in any other form.

In distributed team contexts, the situation could become problematic when more than one person express their degree of uncertainty. Another challenging question we

face is how to combine multiple uncertainties over the same assumption from different sources.

#### 1.1.1.3 Reliability of Data Sources

Most collaborative tagging applications do not take into account the trustworthiness of a statement, nor the reputation of annotators based on their expertises and reviews. The reliability of a statement depends on the domain-specific reputation of the person who made this statement. For example, the statement “Richard III was killed at the Battle of Bosworth Field” or “The fragment of the pottery describes a waggon ride scene” sounds more reliable if claimed by (A) *an archaeologist* than (B) *an Oracle Database Administrator*. Because the domain of this statement is history and apparently the former has a good knowledge of history and archaeology – this makes A’s statement more truthful than B in this particular case. However, most current systems only use a global ratings or review model, which does not differentiate between ratings made by those who have a speciality in a particular domain and ratings made by those who are unfamiliar with the topic.

As we discussed earlier, the trustworthiness of an opinion or statement should be determined by a combination of user uncertainty factor and user credibility. If a credible person claims a statement with a relatively high degree of certainty, such statement is generally more reliable.

#### 1.1.1.4 Model Extensibility and Interoperability with Legacy System

In a collaborative working environment, new concepts and properties are frequently added to the existing vocabulary, and more information is collected as time goes on. Currently, relational databases require that schema be defined before users can add data. A fixed schema is not suitable for representing arbitrary information. In most cases, schema changes are required at some points. However, people cannot anticipate changes in advance and it is not practical to develop a universal schema to accommodate all requirements. Ideally, the schema should be updated and extended gradually to incorporate corrections or changes. However, such changes to the schema after the application becomes operational might lead to major unintended impacts for the whole



system. Relational database designers are usually reluctant to update the schema to maintain data integrity and system stability. As a result, a new and extensible data model is required.

As discussed earlier, to improve the expressiveness of the annotation, a semantic approach should be used, which requires content to be a semantically compatible. It raises the question of how to extract useful information from existing legacy systems and convert it into semantic-ready contents to be used for annotation.

### 1.1.2 Searching and Ranking

Once we collect the annotations, the next step is to provide an interface to search semantically annotated content and return a ranked list with the most desirable results on the top based on ranking criteria and weights. This section will highlight some of the issues and problems of the traditional search and ranking techniques.

#### 1.1.2.1 Needs for Collaborative Search

The demand for collaborative search is primarily driven by the cross-craft interaction research. One of the sub-projects in Tracing Network program is to use the method of “chaîne opératoire”<sup>1</sup> for cross-craft interaction research. In [12], cross-craft interaction is defined as follows: cross-craft interaction can best be regarded as “a phenomenon emanating from human interaction with one or more technologies while being social beings” and “the contact between one or more crafts with adoptive/and adaptive behaviour as a consequence”. Technologies are thus essentially social phenomena. Cross-craft interaction is also a unifier because we know from the archaeological record that several crafts have aspects of their procurement, production, distribution or consumption patterns in common.”

As we can see, cross-craft interaction is not looking only for artefacts, qualitative data or archaeological context such as the location of one discovery, but also its production processes, technological make-ups, distribution patterns and the social and economic impact of such cross-overs [12] [17]. Therefore, it is crucial for researchers

---

<sup>1</sup> “chaîne opératoire” is French, which means “operational chain”

within different areas of expertise to study socio-economic impacts of various factors. For example, the mineral composition of a ceramic pot and the location where it was discovered, two seemingly unrelated factors help archaeologists to reconstruct the steps and the techniques used during the production process. It also helps to find possible migration routes (e.g. non-native artefacts made out of certain raw materials passed from some regions to there through trade or networks or military alliances). The importance of each factor need to be highlighted by the experts in their respective areas – in this case, it would require collaboration between ceramics specialists and field archaeologists, and collaborative search offers a possible solution to this.

Even for the same archaeological records or excavation data, the outcome of the research may vary considerably depending on various factors where archaeologists and historians within different areas of expertise have contributed different outcome criteria for their analysis. Collaborative search provides a mechanism to help archaeologists and historians collectively to adjust criteria and the ranking measurement model to reflect the impact of these factors in cross-craft interaction studies.

Furthermore, information about the past is often incomplete. The researchers rely mostly on fragments or archaeological remains. There are always some degrees of uncertainty over the data due to the incomplete nature of the archaeological records. It poses another significant challenge for collaborative search in the domain of archaeology. Therefore, uncertainty is another essential element that must be incorporated into the model.

**Example: Collaborative searching** The example in Figure 1.3 shows the challenges faced in querying the dataset. Suppose there are two queries makers,  $u1$  and  $u2$ , and they want to write this query:

*Query: find images with a hunter and a two-wheeled vehicle pulled by some animals. What was the hunter riding?*

Here are a few questions concerning this search:

- (1) What would be the most expressive yet intuitive interface for this task? Whether

to use a form-based interface with various types of filters or a simple Google-style single text box? The former has more expressive power, but the latter is more user-friendly.

(2) How to make use of the knowledge-base to enable automatic reasoning? How does the query engine know the fact that a waggon is a vehicle and oxen (or horses) are animals? How does the system query over relations between entities? How do we know “ride” is an action performed by the hunter?

(3) How to deal with the relative importance of different criteria suggested by various searchers? Suppose there are two users  $u1$  and  $u2$ , each of them looks at the same question but from different perspectives. As an expert in ancient human representation, user  $u1$  believes that “two-wheeled” is of less significance while user  $u2$ , who was more concerned about the type of vehicle, thinks that the person being a hunter is not essential (e.g. could also be a warrior or farmer). It is necessary to allow impact factors in various aspects to be adjusted to suit particular research focus (e.g. whether this is more about the transition from hunting to farming society or the gender roles in the community).

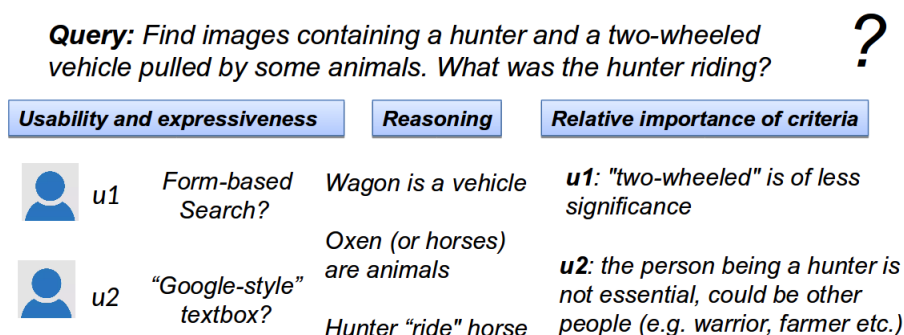


Figure 1.3: Collaborative search conducted by two users that searches for images with a hunter and a two-wheeled vehicle pulled by some animals.

### 1.1.2.2 Usability of Query Interface

Many existing applications provide a full-text search function with a single text field, others use form-based filters, which generate a native query through a series of drop down boxes and text fields, while at the same time hiding the complexity from users.

However, this type of computer-assisted search is still very limited in its expressivity. Suppose we want the following two scenarios:

- *Find all artefacts that were found in Adelfia and Monte Sannace with similar decoration motif in the same position.*
- *Which is the most common type of the objects across all sites?*

To answer the queries (1) and (2), the system will have to provide at least two entirely different parameterised querying interfaces. For every parameterised query, a dedicated interface has to be built accordingly. For this reason, traditional parameterised query interfaces constructed with drop-down menus and text fields virtually restrict the types of question a user can ask.

Most RDBMS provide native standardised SQL-like languages designed for retrieving information from the repository. However, due to the complexity of the language syntax, average non-technical users often find it hard to understand and write a correct executable query statement.

Another question is how to query over relations between entities. Currently, most conventional query interfaces apply property restriction using a filter, but the property itself is not considered as a variable in the query. Traditional form-based search interfaces could construct a native query to answer (a), but for (b), because the attribute on which the restriction is applied is not specified, it is cumbersome to implement such query in most structured querying languages.

**Query (a) :** *Find objects that are not less than 120mm in height*

**Query (b) :** *Find objects with any single dimension exceeding 120mm (inc. height, length, base diameter, etc)*

Similarly, considering another example:

**Query (c) :** *Find images with a man and an ox, show any possible relations between them.*

At the moment, semantically it is impractical to implement such queries in a straightforward manner with traditional tagging and conventional web forms. To address these issues, an expressive and intuitive interface is needed to bridge the gap between native structured query language and traditional form-based querying interface.

### 1.1.2.3 Insufficient Automated Reasoning Support

In a traditional keyword-based annotation framework, at the time of this writing, there is few standard way to incorporate automated reasoning into the search engine for searching tags. Most existing text-based query engines are not able to use semantics and deduce new knowledge. For example, given the query below:

**Example 1** : *Find images with two animals in it.*

The traditional keyword search will not be able to find any image annotated with "horse" or "ox" because the keywords entered do not exactly match any tags that already exist in the system. The system neither understands the semantics of the tags nor has sufficient background knowledge to reason about the facts, such as e.g. horses and oxen being, in fact, animals. Due to the previous point, there can also not be any reasoning applied to search for pictures where for example "horses are pulling the waggon".

Apart from basic subsumption reasoning, current annotation and query systems do not take advantage of relationship characteristics to deduce implicit knowledge.

**Example 2:** *List all objects excavated from the same site.*

$s_1$  *Object a **was located in** Trench 103*

$s_2$  *Trench 103 **was located in** Adelfia*

$s_3$  *Object b **was located in** Trench 105*

$s_4$  *Trench 105 **was located in** Adelfia*

Considering the list of statements above, as one might expect, new statements  $s_5$  and  $s_6$  can be deduced through the transitive closure of the relation "*is located in*" respectively:  $(s_1, s_2) \rightarrow s_5$ ,  $(s_3, s_4) \rightarrow s_6$  .

$s_5$  *Object a was located in Adelfia*

$s_6$  *Object b was located in Adelfia*

Finally,  $(s_5, s_6) \rightarrow s_7$

$s_7$  *Object a and Object b are located on the site.*

Unfortunately, this part of the knowledge is implicit, a direct query over explicit statements  $s_1 \sim s_4$  will return nothing because statements  $s_5 \sim s_6$  are derived from the knowledge base. For this reason, an effective search engine should take into account implicit information discovered from the reasoning process.

#### 1.1.2.4 Combining Weighted Criteria in WHERE Clauses

Queries to the annotated dataset will be ultimately converted into native queries, typically a WHERE clause (or an equivalent of WHERE clause ) is used to filter query results with a list of conditional expressions combined using logical operators. Currently, WHERE clause in most structured query languages extract only those records that fulfil a specified criterion, to be more precise, a WHERE clause returns either “true” or “false”. Suppose we have a list of properties, considering the following query.

**Example: searching image database** Suppose we would like to search for object that meets the following conditions:

1. *pyramidal pot with base diameter < 3cm*
2. *made from terracotta*
3. *have decorative motifs similar to the ones found in Monte Sannace*
4. *production technique is graffito-carved*

These criteria might not always have the same degree of importance. For example, (1) and (4) might be critical for some researchers while criterion (3) might be a desirable requirement for others. A certain degree of uncertainty is usually associating with (3). However, WHERE clause always gives the same weight to each criterion. It does

not distinguish essential requirements from desirable ones, nor does it rank the results according to the relative importance of each impact factor. There are a few questions that need addressing:

**Personalised search experience** – How to personalise search results considering (1) user preferences for each criterion, and (2) uncertainty in the data. The system should be able to combine them in some ways to calculate the overall degree of satisfaction for every possible match on the query, and then display the results with the most desirable one at the top.

**Construct query collectively as a team** – In a team environment, how to summarise and aggregate different people’s opinions to determine the weight for each criterion.

### 1.1.3 Collaborative Categorisation

Various classification techniques have been developed such as Neural Networks, Decision trees, k-Nearest neighbour, support vector machine and Naive-Bayes [18]. However, many of them are not primarily designed to be used in a collaborative environment. Nowadays, virtual groups and inter-organisational and interdisciplinary collaboration are becoming increasingly common, which brings additional challenges to the table. For example, in an archaeological excavation, objects are often discovered in the form of fragments. Different techniques (chemical analysis, x-ray, microscopic analysis of the rock and mineral inclusions, etc.) have been used to help archaeologists classify objects by answering questions like “what was it used for?”; “How was it made?” and “where did it come from?” - However, there is always considerable uncertainty associated with the conclusions as people might have a different interpretation of the evidence, or look at the question from their personal perspective - a systematic way of integrating evidence from multiple sources is needed before any definitive conclusions can be made.

Nowadays, the problem of object classification usually requires a group of people to work collectively. Group decision-making becomes increasingly common. A team

consisting of a variety of members is often more capable of making critical decisions than an individual. However, people might see the same classification problem from their personal perspective and subsequently sort objects according to their understandings and categorisation systems. Again, uncertainty increases the complexity of the problem.

#### 1.1.3.1 Universal Categorisation Generation

Sometimes a community cannot agree on a universal terminology or categorisation system, and objects might be categorised in different ways or associated with more than one class. If a universally-agreed taxonomy is not available, one of the most common approaches for categorisation is unsupervised learning (clustering). In this case, clustering builds a set of categories depending on the characteristics of the objects. However, clustering results are not very predictable, especially when some values are missing from multi-dimensional datasets. The initial attempts of this research to create a similarity matrix for clustering by measuring the pairwise distances between tags using lexical semantic relatedness [19] failed to obtain desired hierarchical clusters archaeologists wanted but its potential application is discussed in section 7.2.

#### 1.1.3.2 Ambiguous Class Definition

Classification can be achieved through a variety of approaches such as ontological A-Box reasoning [20]. However, these methods are primarily based on Description Logic (DL) [21] which follows the traditional route of grouping objects under a well-established definition of concepts. They attempt to examine the value of an object's property to suggest whether it belongs to a particular class. In ideal circumstances, where necessary and sufficient conditions on classes are well defined, we can classify objects in this way. However, the process of determination of these criteria and claims is cumbersome in practice. In most real-world projects, people tend to categorise objects in "natural" clusters based on experience rather than formally defining every single class upfront.



### 1.1.3.3 Complexity of Multi-criteria Group Decision-making Process

In the typical distributed environment, people work together as a team, so classification problems will need to go through a group decision making process – where each of a group always tries to categorise objects into groups based on their interpretation of evidence. In general, the decisions made collectively by a team or teams of experts are more reliable than these from individuals. However, reaching these decisions is not always easy. A challenging question is how can we classify objects in the absence of agreed classes but with specific expectations.

## 1.2 Thesis Statement

This thesis analyses the nature of the collaborative working environment and investigates the models and techniques for collecting and aggregating annotated data from a distributed team and proposes a novel semantic framework, which takes into account criteria preferences, uncertainty and reliability of data sources, to improve the usability and user experience in collaborative annotation, search and classification.

## 1.3 Research Scope and Contributions

The main research contributions are as follows:

1. Proposed and developed a semantic data model for collaborative annotation, which enables members in a distributed team to annotate datasets collectively using more expressive triple statements compared to traditional keyword tagging. The data model also captures the uncertainty of annotated data and the reliability of the data source, which provides a logical infrastructure to support intelligent search and classification afterwards.
2. Developed an intuitive framework for searching annotated data (See contribution 1). Regarding usability, our graph-based query framework bridges the gap between native structured query languages and traditional form-based interfaces. This intuitive framework provides a better user experience for the non-technical

end-user, while maintaining the expressiveness of more complex query languages. The structure allows a collaborative team to create queries collectively and provides users with personalised ranked results according to their preferences of criteria while taking into account the uncertainty of statements and the reputation of annotators. A graphical representation of deductive reasoning rules was also proposed to help users discover implicit information from a knowledge base through inferencing.

3. Presented a novel computer-assisted collaborative classification method to help distributed team members categorise a large number of objects. The research proposes a formal collaborative category generation method for a group which is supported by a custom variant of a Naive-Bayer classifier built through learning. The process of developing a classification model also considers (1) the uncertainty in respect to classification decision-making process and (2) what facts or attributes led to the final classification decision.

The research is primarily focused on data modelling to accommodate different types of annotations from collaborative teams, as well as methods and techniques for improving the performance of search, reasoning and categorisation. The following issues are NOT included in the scope of this thesis:

- The research is about the uncertainty, not the probability as differentiated earlier.
- We treat the criteria as independent because the underlining assumption is that all attributes are independent of each other. Therefore, the complexity of dependency among criteria and attributes is out of scope.
- The management and organisational structure of a team are not covered in this thesis. Because in this case, someone in the group could have authority to make any decisions on behalf of the whole team.
- In our framework, reputation measurement is entirely based on users' activities conducted online, although factors such as educational background and work experiences recorded in their profiles would have some impacts on user credibility

evaluation, we are currently unable to verify the genuineness of such information and hence do not evaluate them in the reputation model.

## 1.4 Thesis Overview and Summary

This thesis presents a novel semantic data model for annotating objects in a collaborative working environment and investigates how the uncertainty of statements, the reputation of annotators and the user's preferences of criteria would affect the result of search and categorization. Chapter 1 gives an overview of existing collaborative tagging, search and categorisation models and techniques and highlights some of the problems and challenges. Background knowledge and relevant technologies, along with a survey of existing approaches, methods and tools are introduced in Chapter 2. In Chapter 3, a novel semantic collaborative annotation framework is presented as well as a fundamental ontology-based data model, which has not only the capability to record more expressive statements compared to traditional keyword tagging, but also captures the uncertainty of statements and the reliability of the data source.

Chapter 4 discusses the feasibility of using graphical patterns to search annotated data. In this section, an intuitive yet expressive graph-based querying technique is proposed, where queries can be tailored specifically to an individual's interests. The approach takes into account user preferences for different criteria, the degree of uncertainty and the reputations of annotators as discussed in Chapter 3. It also explains how custom deductive reasoning rules can be used to tailor search results to improve the user experience in personalised search. Chapter 5 presents a collaborative computer-assisted classification framework to help collaborative teams categorise objects. Firstly, this research discusses the process of generating a category collectively as a team and then describes the steps of building an automatic classifier by learning from a relatively small number of objects that have been manually classified by team members according to object characteristics based on individual's perspective and interpretation. Chapter 6 presents a few prototype applications developed for a focus group to illustrate the proposed methodology. In addition to that, experiments have been carried out with several datasets used by an archaeology research project to evaluate the proposed

approach, and they are discussed here. In the end, Chapter 7 gives an overview of the research, summarise the achievements and discusses possible directions for future studies.

# Background and Literature Review

---

This chapter will discuss relevant background information linked to this research. Firstly it presents different team formations: non-cooperative group and collaborative team and introduces a group of archaeologists and historians working together on the Tracing Networks programme across Europe, as a typical example of a collaborative team. As a focus group, the team provides a valuable source to gather feedback and opinions, in particular for usability evaluation. Then a survey of existing techniques and data storage solutions for collaborative annotation is presented based on the published work in [11], including a brief overview of the collaborative team formation and existing techniques for annotation, search and classification.

## 2.1 Collaborative Team Formation

A collaborative team is a group of people working together toward a common goal across time, geographical space and organisational boundaries [34] [35] [36]. First of all, it is necessary to distinguish between a **non-cooperative group** and **collaborative team**. Teams are forms of groups, but not all groups are teams. In contrast to collaborative teams, members of non-cooperative groups might not share the same goal, intergroup conflict and competition are commonly seen in a non-cooperative group.

Secondly, communication between team members plays a major role in the formation of a team. A collaborative team can be a virtual team, where distance separates members of the team, and their communication is not limited to face-to-face communication. Nowadays, use of mobile devices has become a common communication strategy for a virtual group. Members can communicate via the Internet regardless of their geographical location.

The structure of a collaborative team also has a significant impact on how a team functions. Some collaborative teams have no hierarchy and no defined leadership while others might clearly specify the roles and responsibilities of each team member. However, this research primarily focuses the process of collecting and integrating inputs and feedback from individuals in a team to reach a solution, in particular how to record and aggregate different users' statements, represent uncertainty, perform collective/personalised search and categorisation. Factors such as the hierarchical structure of the team or virtual communication are not the primary focus of this research. In this thesis, it is assumed that no defined group leader has the authority to make the decision on behalf of the whole team.

### **2.1.1 Focus Groups: Tracing Networks Programme**

The archaeological research team working on the Tracing Networks programme [12] is a typical example of a collaborative team. Tracing Networks programme, funded by the Leverhulme Trust, is a joint research project that combines expertise in archaeology, museum studies and computer science in seven closely linked sub-projects. The Tracing Networks programme investigates the network of contacts across and beyond the Mediterranean region, between the late Bronze Age and the late classical period (1500-200 BC). It focuses on networks of crafts-people and craft traditions, asking how and why traditions, techniques and technologies change and cross cultural boundaries, and exploring the impact of this phenomenon. This long time-span is characterised by widespread changes involving many societies and unified by the emergence of state-societies involving new ways of organising production and consumption. The programme sets technological networks in their larger social, economic and political contexts to expand our understanding of wider cultural developments [12]. The team consists more than 20 researchers based in Leicester, Glasgow, Exeter, Athens and Southern Italy working in 7 sub-projects.

- Cross-craft Interaction in the cross-cultural context of the late Bronze Age Eastern Mediterranean
- Weaving relationships: loom-weights and cross-cultural networks in the ancient

Mediterranean

- Plain cooking: ceramics, networks of technological transfer and social change, from the late bronze age to the iron age in the Greek world
- Colonial traditions: ceramic production in Iron Age and Punic Sardinia
- Mint condition: coinage and the development of technological, economic and social networks in the Mediterranean and beyond
- Salt of the earth: the exotic and the everyday in bronze age Europe
- Translating art and craft: Human representations, identities and social relations in the Late Bronze and Iron Age of Central Europe

In particular, data sets used in subprojects "weaving relationships" and "Translating art and craft" are most relevant to my research. Data collected in these sub-projects have been used to evaluation our methodologies. We will discuss more detail in Chapter 6.

## 2.2 Methods for Collecting Qualitative Data

This section compares three different approaches for adding searchable content to an object. A real-life example is the picture of waggon-ride scene mentioned earlier in Figure 1.1. People want to express their opinions on it. This section discusses some existing solutions as well as the advantages and disadvantages of each option.

### 2.2.1 Unstructured Free Text

Collecting data from a group of people as the unstructured free text is the simplest option as a multi-line text field will do the job without any problem as shown in Figure 2.1. People often use unstructured, free-form text, in a variety of natural languages. These are human-readable and highly expressive. Frequently there is no universally agreed terminology, and vocabulary is shared with a very limited group of experts. Texts reflect individual interpretations and explanations of evidence. A focus group

formed as part of the Tracing Networks programme gave us an excellent opportunity to study how members of various backgrounds describe their knowledge and how we share information with others. Most archaeologists prefer to use unstructured text rather than a formatted data model to describe their data.

### Description

This ceramic depicts a human figure, possibly a male hunter, riding a four-legged horse-like animal, followed by a four-wheeled wagon pulled by two ox-like creatures.

SUBMIT

Figure 2.1: Data entry page for free text

People usually adopt their category and classification systems, if terms used in the description have not been given a strict definition, and there is no standard terminology accepted by the whole community.

Much research has been done in Natural Language Processing (NLP), but the technique is still not capable of performing unrestricted full-text understanding [37]. We are still far from carrying out any intelligent query. Lack of a pre-defined data model and standard terminology make the unstructured text not suitable for query and automated reasoning. Although users can enter any text as they wish, it is hard to combine inputs in a meaningful way.

### 2.2.2 Structured and Semi-structured Form

Various data collection methods have been developed . One of the most favourite ways is to collect data through online surveys form. A form usually consists of a combination of text fields, text boxes, radio buttons and check boxes Figure 2.2 is an example of the data entry page for semi-structured data describing the same content as shown in 2.1.



The form consists of several labeled sections with input fields and dropdown menus:

- Object**: A text input field containing "Wagon led by an animal".
- Figure type**: A dropdown menu with "Human-Male" selected.
- Description**: A text input field containing "possible a hunter".
- Animal**: A dropdown menu with "Four-legged" selected.
- Description**: A dropdown menu with "a horse-like animal" selected.
- SUBMIT**: A blue button.

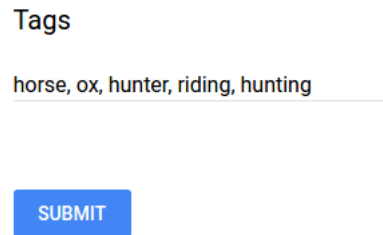
Figure 2.2: Data entry page for semi-structured data

In this case, semi-structured data requires a schema to be defined in advance. Information stored as structured or semi-structure data is not as expressive as a natural language, but it is machine readable and processable as they validate against given schema (e.g. XSD, RDB Schema), which makes it possible to run sophisticated queries using structured query language and perform statistical analysis. However, a fixed schema is not flexible enough to represent arbitrary data provided by the user. For example, adding a piece of information such as “state of preservation” requires changes to the schema and such change would have a significant impact on the existing system – it is not possible to add such data without updating the schema and inserting additional text fields. In other words, users can only add information that is already defined in the schema.

### 2.2.3 Keyword-based Tagging

Tagging has become a popular way of adding searchable information to the resource, which is used in many social networking websites. A tag is a small piece of freely-chosen plain text helping users to organise the content. Figure 2.3 shows how the keyword tagging method would look like in our case. Compared to documents, keywords are simple, concise and explicit. A keyword captures the essence of the object and people are free to use any vocabulary terms. As a tag is only a simple keyword, some aggregate

methods such as tag clouds could be used to provide tag-usage statistics.



Tags

horse, ox, hunter, riding, hunting

SUBMIT

Figure 2.3: Data entry page for keyword tagging

However, keywords are ambiguous, and the links between them are usually unclear. In some cases, it is not clear what they are referring to. Besides, people sometimes get confused with keywords as the relationship between them are not clarified. For example, in Figure 2.3, it is still not clear to us what is the connection between horse, hunter and ox. Was the hunter riding the ox (bull)? Or, was he hunting the horse? Keyword tagging method is ideal for annotating entities but describing the relations among these entities is problematic. The advantages and disadvantages of different methods for collecting qualitative data are discussed in Table 2.2.3.

Table 2.1: Comparison of data collection methods

Technique	Advantages	Disadvantages
Unstructured free text	<ul style="list-style-type: none"> <li>• Highly expressive, capable of describing any statement with uncertainty</li> <li>• Human readable natural language</li> </ul>	<ul style="list-style-type: none"> <li>• No schema definition or universally-agreed terminology</li> <li>• Unable to perform sophisticated search</li> <li>• Data cannot be use for computer-supported reasoning</li> <li>• Difficult to summaries texts from multiple persons</li> </ul>
Structured and semi- structured form	<ul style="list-style-type: none"> <li>• Machine readable</li> <li>• Pre-defined fixed schema, controlled vocabulary</li> <li>• Support relatively complex schema-based search</li> <li>• Reasoning capability (limited)</li> </ul>	<ul style="list-style-type: none"> <li>• Limited expressiveness, not suitable for representing arbitrary data</li> <li>• Schema changes could be cumbersome</li> <li>• Problem with merging input from different sources</li> </ul>
Keyword- based tagging	<ul style="list-style-type: none"> <li>• Restricted expressiveness, not suitable for representing arbitrary data</li> <li>• Schema changes could be cumbersome</li> <li>• Relationships between keywords are not clear</li> <li>• Problem with merging input from different sources</li> </ul>	<ul style="list-style-type: none"> <li>• Simple and concise</li> <li>• Free to add any tags without restriction</li> <li>• Possibility to perform tag grouping and statistics</li> </ul>

### 2.2.4 Related work

Many studies have been conducted to address the weaknesses of the traditional methods for collecting qualitative data. Ontology-based tools or applications have already been used in many areas, ranging from describing services to annotating images. For example, SWEET [25] is a tool that allows users to annotate RESTful web services semantically to support automatic web service discovery and composition.

Cohere [23] is a Collective Intelligence (CCI) framework for annotation and argumentation. Cohere was created for connecting people, ideas and web documents. *Cohere Conceptual Model*, the underlying data structure used by the framework consists of several core elements including *Group*, *User*, *Idea*, *Annotation*, *Document* and *Connection*. Core elements can be extended to create new concepts. For example, *Problem*, *Opinion*, *Data*, *Theory*, *Prediction* are subclasses of class *Idea*, connection-types can also be positive, neutral or negative. Cohere also provides a Mozilla extension that allows users to annotate web pages. The relations among people, ideas and web pages can be visualised as an interactive mind-map. CCI has gradually developed or evolved into Evidence Hub [24], a new platform which helps community members to contribute their collective intelligence. In some way, the idea of Cohere conceptual model is similar to the ontology-based model developed in this research. Instead of the 3-tier (person-idea-document) model adopted by CCI, the primary focus of Collaborative Image Annotation/Classification Ontology is to describe entities and any assumptions associated with these objects, as well as to record uncertainty in a systematic way.

Some research has been carried out in the past to develop a domain-specific language for describing different interpretations of artefacts in the domain of archaeology. One of the earliest attempts is KIVA [26], KIVA is a text-based prototype interpreter developed in late 1980 which aimed to provide a BASIC-like pseudo programming language for archaeologists to describe assumptions based on different interpretations of the evidence or theory. Assumptions were described in propositional logic, which is written in a form resembling IF-THEN-ELSE statements in BASIC.

The studies of these methods inspired the researchers in this area to develop better

data models and languages.

## 2.3 Storage and Archiving Solutions for Collected data

Once data is collected, the next concern is data storage. In this sections, we give an overview and comparisons of the existing approaches for storing qualitative data. The advantages and disadvantages of each technique below will be discussed and summarised.

### 2.3.1 RDBMS (Relational Database Management Systems)

Digital recording and the publishing of data have increased significantly in recent years, which have gradually replaced paper-based records in many sectors. Traditionally, digital data is stored in Relational Databases (RDB). Many commercial relational database management systems (RDBMS) exist and are widely used, including Microsoft Access, SQL Server, MySQL, Oracle and Filemaker, etc. It is worth noting that a significant amount of data is stored in flat file databases such as Microsoft Excel spreadsheets. A relational database has a collection of tables of data items. It consists of three basic concepts: Tables, Columns (fields) and Records (rows). Figure 2.4 illustrates the relationships between table, column and records. A database schema defines the logical and physical definition of a relational database structure. A table organises the information about a particular item or topic into rows and columns. Each table uses one or several columns as primary key, which uniquely identifies each record in the table. Tables are linked through primary and foreign keys. Relational database modelling concerns the normalisations of data before it can be stored in an RDB. The database schema defines attributes for entities, relationships between the records.

### 2.3.2 Problems with RDBMS

**Schema Flexibility** – Relational databases require that schemas be defined before users can add data. However, fixed schemas are not suitable for representing every form of information. For example, archaeologists cannot anticipate the data structure in advance, and it is not practical to develop a universal schema to accommodate all requirements of any artefacts discovered in the excavation. Ideally, relational database schemas should be updated and extended gradually to incorporate correc-

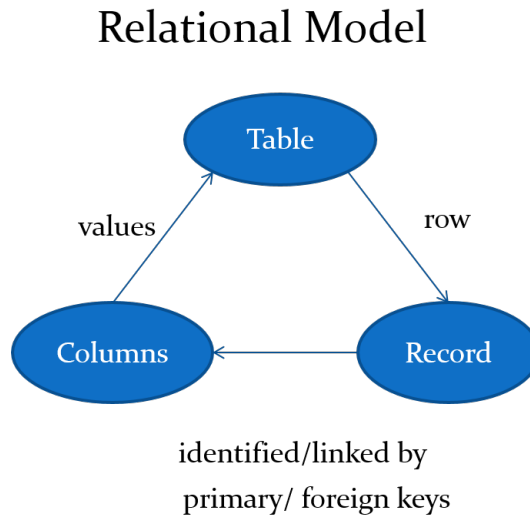


Figure 2.4: Core elements of the relational database

tions or changes, however, such changes to the schema after the application becomes operational might lead to major unintended impacts for the whole system. As a result, relational database developers are usually reluctant to update the schema to ensure data integrity and system stability.

**Indirect relationship** – The relational storage mechanism is robust and there are various commercial relational databases management systems available on the market, but one of the limitations of defining a schema in such a way is that the links between tables are implicitly defined. Therefore it does not give a clear indication of the relationships between concepts. This poses a challenging task for non-technical domain experts to understand and design their database schema without necessary skills and knowledge.

**Interoperability** – Another issue with relational databases is the reusability of the schema. For instances, assuming a table for “loom-weight”<sup>1</sup> already exists – if we want to add a more accurate type of loom-weight (e.g. disc-loomweight) to the system, it

---

<sup>1</sup>A small weight used to hold down the warp threads on a weaving loom

would be great if we can reuse some generic attributes and definitions (e.g. material, decoration, degree of preservation, etc.) from the existing loom-weight table, on the other hand, we should be able to add new attributes (e.g. diameter for disc loom-weight, base major for Pyramid loom-weight). RDBs do not natively support inheritance. Table inheritance is usually achieved by duplicating existing tables and fields. Some research has been done in the archaeology community regarding data structures for describing various artefacts, but without a universally accepted standard, people find it difficult to share or reuse fragments of schema or vocabulary with others because most databases were created and maintained internally within the organisation. Interorganisational knowledge exchange remains a big issue.

**Advanced search** – Most RDBMs use SQL (Structured Query Language) for retrieving information from RDBs. However, non-IT professional usually find it hard to understand the syntax to write a correct statement. Although many applications provide advanced full-text search function or form-based query builders to help generate the underlying SQL by hiding complexity from users. Regarding expressivity, what this interface can express is still quite limited.

**Inferencing** – Traditional text-based search engines lack the ability to support automated reasoning. For example, if loom-weight *A* has similar decorative motif as *B*, and *B* has similar decorative motif as *C*, assuming the user wants to run the following query: Find all loom-weights that have the same decorative motif. Relational database management systems cannot perform simple logical reasoning to deduce that *A* and *C* have similar motif as we do not describe such information explicitly. Similarly, a text-based search for “Ritual” site will not return a site tagged as “Place\_of\_cult” as the text in the database does not match the search keyword.

We already discussed the issues of distributed teams and difficulties to agree on a standard vocabulary. Even if such vocabulary exists, people would still use different terms for the same concept. For example, archaeologists use different chronological systems when dating an object, and one might describe an object dating from the 2nd half of 3rd BC, another might label it as early Hellenistic age – querying the database



while taking into account the background knowledge becomes problematic. Consider the seemingly straightforward scenario we previously mentioned in Figure 2.1 again. The following is a possible database schema for describing such statement:

### Sample solution in RDB:

Listing 2.1: RDB solution

```

DROP TABLE IF EXISTS 'Object';
CREATE TABLE 'Object' (
  'InventoryID' varchar(255) NOT NULL,
  'Object_name' varchar(255) DEFAULT NULL,
  'Description' varchar(255) DEFAULT NULL,
  'Excavation_date' datetime DEFAULT NULL,
  'Excavation_location' varchar(255) DEFAULT NULL,
  PRIMARY KEY ('InventoryID')
);

DROP TABLE IF EXISTS 'Opinion';
CREATE TABLE 'Opinion' (
  'PersonID' varchar(255) DEFAULT NULL,
  'ObjectID' varchar(255) DEFAULT NULL,
  'Comment' varchar(255) DEFAULT NULL,
  'ID' int(11) NOT NULL AUTO_INCREMENT,
  PRIMARY KEY ('ID'), KEY 'PID_PK' ('PersonID'), KEY 'OID_FK' ('ObjectID'),
  CONSTRAINT 'OID_FK' FOREIGN KEY ('ObjectID')
    REFERENCES 'Object' ('InventoryID'),
  CONSTRAINT 'PID_PK' FOREIGN KEY ('PersonID')
    REFERENCES 'Person' ('PersonID')
);

DROP TABLE IF EXISTS 'Person';
CREATE TABLE 'Person' (
  'PersonID' varchar(255) NOT NULL,
  'PersonName' varchar(255) DEFAULT NULL,
  'reputation' int(255) DEFAULT NULL,
  PRIMARY KEY ('PersonID')
);

```

As we can see, the database would contain three tables – Table `Object` stores the information about the object and table `Person` contains a list of annotators. A third table is needed to describe each annotator’s comment with respect to the given object (due to the many-to-many relation between `Person` and `Object`). The problem with this solution is obvious: (1) fixed schema lacks scalability and flexibility, and (2) relationships between tables are specified through foreign key constraints, links are not explicitly represented, and (3) comments are still in the form of free text hence there is no way to aggregate them, and (4) there is still no formal method to represent uncertainty in this database, and (5) though each person has one global reputation

value stored in a column, the current model is still not able to describe how reputation changes over time and varies across subject areas.

### 2.3.3 XML documents

XML(Extensible Markup Language) is a semi-structured markup language. XML uses a tree structure to organise contents. It starts with one root element that might contain nested child elements. An XML Schema (XSD) describes the structure of an XML document. Let us look at the same statement we wanted to record in Figure 2.1 again. The following code shows a possible XML-based solution to the question.

#### Sample solution in XML:

```
<annotations>
  <objects>
    <object id="INV008">
      <description>Fragment</description>
    </object>
    .....
  </objects>

  <comments>
    <comment id="COMM1" obj_id="INV008">
      <text>This ceramic vessel depicts a human figure, possibly a
        male hunter, riding a four-legged horse-like animal,
        followed by a our-wheeled wagon pulled by two ox-liked
        creatures.</text>
      <person>Katharina Rebay-Salisbury </person>
    </comment>
    .....
  </comments>

</annotations>
```

There are still problems with this XML solution: (1) The hierarchical model for representation is limited compared to the relational database model, describing not hierarchical content in XML usually requires extra efforts. (2) Although people tend to

pick meaningful words as XML tags name, the meaning of XML tags is still ambiguous because the XML schema only defines the structure of the documents, but not the semantics. A validated XML document is syntactically correct, but its interpretation might be confusing semantically. In a word, such document is machine-readable and human-friendly but not machine-understandable. (3) Again, the relationships among entities are not clear by only looking at the tree structure of the document. In the example above, we placed a `<person>` element inside `<comment>` to indicate that the person made the comment. However, the XML schema could have defined the structure the other way around by placing a `<comment>` element inside `<person>`. Semantically the computer cannot tell the difference between labels or tag names because only the syntax is defined and there is no strict definition of what these tags mean; (4) Another problem is that working with XML documents is considerably slower than using relational databases. There is a performance penalty, especially when parsing and updating a large XML document.

### 2.3.4 RDF Triplestores and Ontology-based repositories

In recent years, Semantic Web and ontologies have become a hot topic for research. Semantic Web is an extension of the current Web; most resources on the World Wide Web are machine-readable but not machine understandable. The current WWW is document-centric, the content of the document is for human readers, the syntax of the document is defined by schemas as discussed in section 2.3.3. Typical languages of the web include HTML (HyperText Markup Language) and XML (Extensible Markup Language). In contrast, the Semantic Web is for knowledge representation. Semantic web content is not only machine-readable but also machine-understandable. The most popular Semantic Web languages and standards include RDF (Resource Description Framework) [38] and OWL (Web Ontology Language) [39] .

**Ontology** The core of the Semantic Web is the Ontology. The term “Ontology” is a compound word originated in Greek, it is composed of onto (being or the nature of things.) and logia (theory). It was used in the study of the nature of existence in classical philosophy. In computer science, Ontology is a formal specification of a

conceptualisation - Thomas Gruber, who is the pioneer of ontology engineering, gave his definition. In an ontological world, knowledge is modelled by concepts, which are specified by the ontology – In other words, an ontology describes the domain of the reality. In contrast to the RDB, the central concepts in the ontology-based database are Class, Property and Individual.

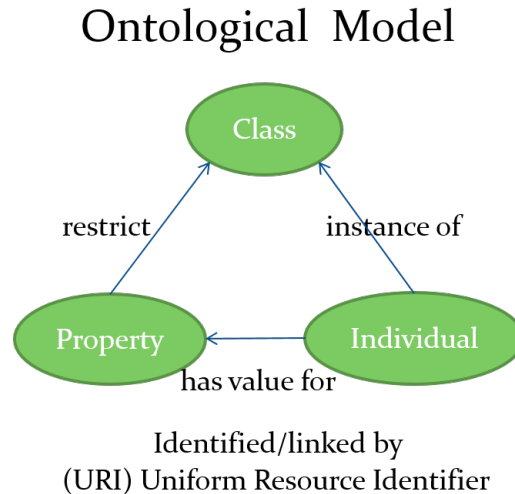


Figure 2.5: Core Elements of the Ontology-based database

A class is a template for the definition of a particular type of objects that share a common structure and behaviour (e.g. Artefact, Motif and Site are classes). The definition of a class can be described using Description Logic (DL), which is a formal knowledge representation language. Individuals are instances of the class.

Properties are attributes of a class or relations between classes. There are two different types of property: (1) Object property and (2) Data type property. Data type properties link individuals to data values while Object properties link individuals to individuals. For each property, domains and ranges can be specified. The domain of a property limits the individuals to which the property can be applied. The range of a property limits the individuals that the property may have as its value. Records are the basic units in the relational database; each row in a table represents a set of related data, which are identified by primary keys. In contrast, RDF triples are used to define statements about resources in an ontology-based database; a URI (Uniform

Resource Identifier) is a string of characters used to identify a name or a web resource. An RDF triple consists of three parts: a subject, a predicate, and an object.

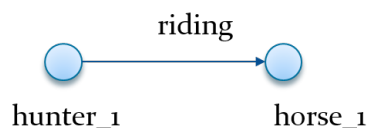


Figure 2.6: Example of an RDF subject-predicate-object triple statement

Consider an RDF triple statement as shown in Figure 2.6

*“hunter\_1 was riding horse\_1”*

where

- *hunter\_1* is the **subject**
- *riding* is the **predicate** (or property)
- *horse\_1* is the **object** of the triple.

A set of RDF triple statements becomes an RDF graph. As a matter of fact, an ontology-based database is a triple graph. A triple-store is used to store and query RDF triple graphs. The RDF graph in Figure 2.7 shows how information, as described in Figure 1.1 is stored internally within a triple-store.

### 2.3.5 Tracing Networks Ontology

One of the primary objectives of the Tracing Networks Project is to provide a logical infrastructure to support classification and analysis of data [12]. More precisely, to develop an ontology for use in repositories of data/elements together with meta-models for tools used by sub-projects. The ontology offers a uniform representation for data and findings from the other sub-projects together with versatile tools through which unforeseen relationships among heterogeneous datasets may emerge semi-automatically. As well as to support technically the different teams in using the tools and interacting

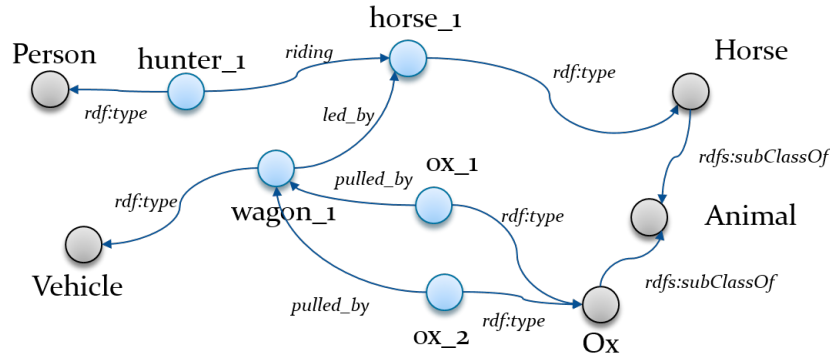


Figure 2.7: RDF graph for Figure 1.1

through them, to ensure the future collaboration of teams and enable future research by others. The benefits of the Ontology-based database include:

**Flexibility** – a flexible and schema-less data model.

**Expressivity** – ability to express arbitrary statements with RDF triples.

**Easy navigation** – explicitly describe relationships among entities and various logical constraints.

**Inferencing** – derive new context data via various rule-based and description logic reasoner. Ontology languages support many characteristics of properties (transitive, symmetric, inversed)

**Interoperability** – Ontology-based database can incorporate and reuse vocabularies from existing ontologies. There are many standardised upper domain ontologies available for use. For instance, CIDOC-CRM [40] is an ISO standard [CIDOC conceptual reference model (ISO 21127:2006)] used in cultural and heritage sector.

## 2.4 Mapping RDB Schema to RDFS/OWL

In the past few years of years, digital recording and publishing of archaeological data have rapidly replaced the traditional print medium of publication. Indeed, the Internet has contributed largely to the success of this paradigm shift. The benefits of disseminating archaeological data via the web have been highlighted as early as 2001 [41]. Through the World Wide Web, up-to-date information is being made available on demand, for very large and complex datasets of archaeological information. Along with text, visual information such as maps and photographs are also being digitised and widely disseminated. In most cases, the digital representation of artefacts is accompanied by metadata that describes the contextual information associated with it. Archaeological datasets by their very nature are large, complex and often fragmented bodies of information. They record knowledge about artefacts that represent circumstances underlying diverse historical, social, political and economic contexts. Unsurprisingly in most cases, the information spans across several eras. It is also not uncommon to find data related to an artefact dispersed across several institutions and made available as disparate and diversely published datasets. The relationships between the individual elements comprising these islands of information/knowledge span across spatial and temporal boundaries amongst others. To fully exploit the potential of the web as a dissemination medium, the datasets and their inter relationships need to be rigorously structured. A large number of linkages (cross references) between individual entities, both within and across the datasets, have to be captured using recommended standards for the Web. This is required to ensure their interoperability, while retaining the flexibility needed to develop applications over the data sources. Representing and publishing information in this fashion allows easy navigation, systematic analysis and efficient information retrieval across the vast number of datasets. The potential impact of Semantic Web [8] on archaeology has been well recognised [42]. Traditionally, archaeological experts are used to recording data in relational databases. This is usually an SQL repository or the so-called "Deep web"[43]. These repositories are only accessible via web-based end-points provided by the holders of the data sources. The majority of the rich and contextually relevant data, which would be of benefit to a larger com-

munity, lies buried underneath layers of application interfaces. Indeed, relational data silos provide scalable storage and efficient query execution mechanism, but they are inaccessible to the most popular and powerful web-based search engines.

To increase the uptake and usage of archaeological data, these resources need to be openly available and accessible both to humans and machines. Semantic Web provides the infrastructure that is necessary for data to be smartly marked up and made available as ontologies. Archaeological data exposed as ontologies immediately opens up the domain to the extensive suite of semantic web aware applications such as data browsers and search engines. Vast amounts of fragmented archaeological data could thus be systematically structured and made available to a wider community. Yet another Semantic Web initiative which is gaining increasing prominence and is emerging as an important paradigm in connecting the web via networks of data rather than hyperlinks of documents is the notion of “Linked data” [44]. From an archaeological perspective, a very useful benefit of linked data is that as the published archaeological data space on the web grows and new data is linked to existing information sources, via the linked data cloud searches over these datasets deliver explicitly asserted and implicitly inferred results based on the latest curated datasets.

Traditionally, archaeological experts are used to recording data in simple relational databases (RDB) such as Microsoft Access or FileMaker. Although these data storage mechanisms are robust, they are inaccessible to search engines. To increase the uptake and usage of data, these resources need to be openly available and accessible both to humans and machines. To use the ontology-based model, the first step will be to transform existing datasets from RDB to RDF.

**Existing RDB to RDF Mapping Frameworks** : Some studies have also been conducted to motivate archaeologists to adopt semantic web technologies for managing and analysing data. We obtained interesting results in the domain of cultural heritage and museums. However, most of these efforts focused more on archiving data rather than performing interesting reasoning and statistical analysis. Archaeology by its very nature focuses on establishing linkages between past events, places, people and things. The Semantic Web infrastructure, therefore serves as a potential solution because of its



emphasis on capturing relationships and should be exploited to provide archaeological data management solutions.

Much research has been done on mapping between ontology models. A survey of approaches for mapping RDB to RDF is presented in [45]. D2RQ [46] and Virtuoso [47] provide a declarative language to describe mappings between relational database schemata and OWL/RDFS. The language is limited to specifying basic triple pattern match. An approach closely related to this research is R2O [48]. The authors specify an XML-based language for the transformation. In related archaeology projects [49], the creation of initial mappings between database columns and RDF entities was a manual exercise undertaken with the benefit of domain knowledge from English Heritage. In [50] bespoke tools are provided that guide the data curators through the mapping process, using basic natural language processing.

Also, W3C specification for mapping RDB data to RDF is explained in [32]. This method allows direct mapping between RDB schema and RDF to be defined. The approach supports simple schema-to-ontology mapping, but the structure and the vocabulary of the generated ontology only reflect the source RDB schema. There is no way for users to customise the target ontology via such mapping technique. R2RML [33] was developed to address this issue. R2RML is a custom-tailored ontology mapping language that allows developers to define the customised mapping from RDB schema to RDF datasets. R2RML mappings are written in Turtle syntax with embedded SQL queries to support more complex mappings. R2RML is relatively flexible in comparison to [32]. However, R2RML uses a mixture of mapping rules in Turtle syntax and complicated native SQL queries. These limitations pose a significant challenge for domain experts and ontology developers.

Some work has been done on ontology alignment that maps RDB to more than one ontologies. Context OWL (C-OWL) [31] is a language that extends OWL specification. C-OWL can be used to defining contextual ontologies whose contents are locally defined, as well as specify the mapping between contextual ontologies and other ontologies.

**Proposed ECA Transformation Approach** : A transformation approach for converting of archaeological RDB datasets to ontology-based [51] data Section 3.2 will

discuss the data model. Relational data sources are first transformed into data objects and these objects are then mapped to ontology instances via DOTL (Database to Ontology Transformation Language), an Event-Condition-Action (ECA) based [52] scripting language developed in this research. A novel feature of the transformation is the generation of linked data sources. We exemplify the framework using a case study from the Tracing Networks project [12]. The ontology used to showcase the capability of the transformation language is an extension of CIDOC-CRM standard vocabulary.

RDB schemas can be quite complicated concerning integrity constraints. The proposed framework, which builds on top of Hibernate framework [53], allows users to specify complex patterns for ontology transformation. The proposed approach generated instances that are linked with popular datasets such as DBpedia [54] and Geonames [55] using their search APIs.

## 2.5 Usability of Search Interfaces

Once data is collected and stored in a model, the next step would be to provide a query interface for users to interact with the system. This section discusses the characteristics of some of the most commonly used search interfaces and compares the usability of these querying techniques.

### 2.5.1 Single Textbox with Autocomplete Functionality

Nowadays, autocomplete has become a common feature of many mainstream search engines [56]. When a user types one or more keywords into the text field, the search engine will suggest a list of possible search terms predicted by a certain algorithm based on several factors (such as the popularity of search terms, user's previous search history). Figure 2.8 shows how Google suggested a list of search term when a user entered "*Leicester*" into the Textbox - in such case Google autocomplete suggested 6 possible search terms, some of them were based on the user's previous searching history (e.g. opening hours of the Leicester University Library ), the rest were calculated on the basis of the popularity of certain search phrases (e.g. Leicester F.C.).

In summary, autocomplete provides a convenient way for users to enter search

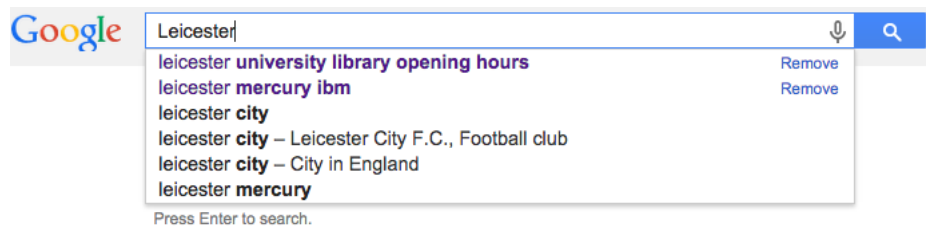


Figure 2.8: Google searches with autocomplete

terms into the text field. However, the support offered by autocomplete is still very limited because it is text-based. Despite research and development in Natural Language Processing (NLP) in recent years, current search engines (even with NLP parser) are still unable to give any reasonable search results to most queries written in unstructured free text.

### 2.5.2 Filter-based Search Interface

Another method is to use a combination of drop-down list and text fields where the user can input search terms and criteria to filter on results. Figure 2.9 is a filter-based advanced search interface provided by the University of Leicester library website where users can filter the search results on a set of criteria (i.e. author, title, series, language and year of publication). Users typically need to use a combination of text boxes and drop-down lists to specify various missing parameters. However, a static form with a fixed number of options limits the expressiveness of the interface. In theory, a dynamically-generated form could improve the flexibility to some extent, but too many inputs make interfaces tedious and complicated to use.

### 2.5.3 Native Query Statement

Another method is to provide users with a facility to write queries expressed in SQL (Structured Query Language) or other SQL-like native query languages such as the object-oriented query language JPQL (Java Persistence Query Language), HQL (Hibernate Query Language) and LINQ (Microsoft Language-Integrated Query for the .NET framework). An interface equipped with native query languages offers the most

**Advanced Search**

words or phrase  And

author  And

title  And

subject  And

series  And

journal title

**Select library:** ALL

**language:** ANY

**collections:** ANY

**type of item:** ANY

**match on:** Keywords

**pubyear:**

**sort by:** None

Figure 2.9: Filter-based search University of Leicester Advanced Search

flexible and expressive power. Given the database scheme in 2.1, the code below in Listing 2.2 shows how to execute an HQL query in Java, which returns all *Opinion* objects annotated by users.

Listing 2.2: Querying options with Hibernate Query Language (HQL)

```
query = session.createQuery("from_Opinion");
opinionList = query.list();
for(Opinion op : opinionList){
    System.out.println("Opinion:"+op.getComment());
}
```

However, due to the complexity of their syntax and semantics, ordinary non-technical users usually find it difficult to translate from natural language to a syntactically correct query statement. To provide average users with a more intuitive way to describe queries in a more user-friendly way becomes a challenging task.

#### 2.5.4 Related Work

In recent years, various querying techniques have been proposed and developed. Some work has been done in the past to improve the usability and expressivity of the form-based search interface. Lois [28] is a flexible form-based interface for knowledge retrieval developed by KMI, Open University. Lois has a UI resembling the "Individual" tab in

Protégé 5.0 which let users construct a variety of OCML queries [29] to search RDF repository.

Apart from form-based querying techniques, there are also some querying approaches based on Natural Language Processing (NLP) technologies. Aqualog [30] is a question-answering system which takes queries written in natural language and a domain-specific ontology as the input; then the system will try to understand the nature of the queries and search the knowledge bases (KBs) for answers. NLP-powered searching is an innovative approach which requires much work in statistical and linguistic analysis. It has great potential, but there is still a long way to go for the researchers to fully understand human languages.

Recently, graph-based search methods have gained popularity, partially because more and more NoSQL such as RDF triplestores, Oracle Spatial and Graph databases are increasingly used for processing big data. Graph-based searching provides a way to retrieve information based on pattern matching. The proposed visual querying technique will be discussed in Chapter 4. SPARQL is the underlying query language used in this research, which is based on graph pattern matching.

## 2.6 Uncertainty Models

Several research groups have proposed uncertainty models for describing uncertain data. This section will give a brief overview of existing models as well as the advantages and shortcomings of these approaches.

- Fuzzy Logic
  - Fuzzy Propositional Logics
  - Ontology language extension with fuzzy concepts and predicates
- Probability Theory
  - Bayesian Networks
  - Description Logics with Probabilistic Extensions
  - First-Order Probabilistic Approaches

### 2.6.1 W3C XG on Uncertainty Reasoning

The closest effort on uncertainty is the W3C Uncertainty Reasoning for the World Wide Web Incubator Group (URW3-XG), which proposed an Uncertainty Ontology in their report in 2008 [57]. The report presented an ontology for describing uncertain information, in which existing uncertainty models could be adopted (e.g. Probability, FuzzySet). However, their focus is more defining the data structure to hold uncertainty information than on its calculation and use. The model proposed is more suitable for representing uncertainty in a single-user system than recording and combining multiple uncertainties in a collaborative environment. The report discussed some common approaches for representing uncertainty in the semantic web.

### 2.6.2 Fuzzy Description Logic and RuleML

Another method is to use Fuzzy Description Logic (DL) which combines fuzzy logic with description logic. For example, suppose the type of DL we extend is  $\mathcal{ALC}$  [58]. Suppose  $C$  and  $D$  are concepts.  $R$  denotes the relation between two concepts.  $\mathcal{ALC}$  can be described using notations below.

$\top$  (top concept, super class of all concepts owl:Thing)

$\perp$  (empty concept)

$C$  (concept)

$C \sqcap D$  (conjunction)

$C \sqcup D$  (disjunction)

$\neg C$  (negation)

$\exists R.C$  (existential quantification)

$\forall R.C$  (universal quantification)

"An interpretation  $\mathcal{I}$  is a pair  $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$  consisting of a non empty set  $\mathcal{I}$  (called the domain) and of an interpretation function  $\cdot^{\mathcal{I}}$  mapping different individuals into different elements of  $\Delta^{\mathcal{I}}$ " [59]. An assertion can be described in one of the two forms: (1)  $c$  is an instance of  $C$  (denoted by  $C(c)$ ) (2)  $c$  is related to  $b$  via relation  $R$  (denoted by  $R(a, b)$ ). For example,  $(Female \sqcap Teenager)(Alice)$  asserts that Alice is

a teen girl;  $hasFather(Alice, John)$  asserts that John is Alice's father. The basic idea of fuzzy  $\mathcal{ALC}$  is to introduce  $d \in \Delta^{\mathcal{I}}$  [59] where  $d$  is an object in domain  $\Delta^{\mathcal{I}}$  so that the degree of object  $d$  being an instance of concept  $C$  is denoted by  $C^{\mathcal{I}}(d)$  under the interpretation  $\mathcal{I}$ . Apart from this, the following rules should apply.

$$\top^{\mathcal{I}}(d) = 1$$

$$\perp^{\mathcal{I}}(d) = 0$$

$$(C \sqcap D)^{\mathcal{I}}(d) = \min\{C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)\}$$

$$(C \sqcup D)^{\mathcal{I}}(d) = \max\{C^{\mathcal{I}}(d), D^{\mathcal{I}}(d)\}$$

$$(C \neg C)^{\mathcal{I}}(d) = 1 - C^{\mathcal{I}}(d)$$

$$(\forall R.C)^{\mathcal{I}}(d) = \min_{d' \in \Delta_x} \{\max\{1 - R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')\}\}$$

$$(\exists R.C)^{\mathcal{I}}(d) = \max_{d' \in \Delta_x} \{\min\{1 - R^{\mathcal{I}}(d, d'), C^{\mathcal{I}}(d')\}\} \text{ (universal quantification)}$$

Fuzzy DL provides a formal way to describe imprecise concepts and allows DL reasoning to be performed. However, Fuzzy DL is more appropriate for defining imprecise concepts rather than describing arbitrary statements.

### 2.6.3 PR-OWL: A Bayesian Ontology Language

PR-OWL is an extension to OWL for representing probabilistic ontologies based on Bayesian first-order logic. [60]. The aim of this language is to allow representation of complex Bayesian probabilistic models by extending the syntax and semantics of the OWL language. The proposed model primarily focus on how to represent uncertainty. How to combine and aggregate uncertainty from different sources is not the primary goal of this extension.

## 2.7 Reputation Models

We can classify reputation models from various perspectives. Conceptually reputation models can be divided into two categories [61]: Cognitive and Game-theoretical models. In the first type, to determine if someone is trustworthy is based on the mental status of the one person. In such case, the assignment of reputation value to another person is based on one's cognitive sense. In contrast to this, the Game-theoretical reputation model is not based on the mental status of the decision-maker but more on the past

activities conducted by another person.

We can also divide reputation models according to the information sources taken into account in the calculation of the reputation values [61]. The most common types of information used by the reputation models are **direct experiences** and **indirect information**.

**Direct experiences** are considered to be the most reliable information source for the reputation model. The direct experience could be obtained by either interacting directly with the partner or monitoring the interaction between other members of the community. In both cases, it is inevitable our judgment is affected by emotional bias.

**Indirect information** is also called witness information or Word-Of-Mouth (WOM). In contrast to direct experiences, indirect information is the information acquired from direct experiences or data gathered from other members of the community indirectly.

There are other reputation models based on **sociological information** or prejudice. The **sociological data** model takes into account the social relationships between individuals and the society utilising social network analysis. **Prejudice** models consider the preconceived opinions or hostile attitude toward other community members. However, the use of these two models is not common due to the complexity of the human societies. Several computational reputation models were proposed for online communities (e.g. Amazon/eBay model, Histos or Sporas model)

### 2.7.1 WOM-based eBay/Amazon models

Online shopping sites such as eBay and Amazon use similar reputation models derived from a simple Word-Of-Mouth (WOM) model. The primary objective of this model is to help customers select high-reputation sellers when making purchasing decision. In other words, establish trust between buyers and sellers who have never interacted before.

For example, on Amazon's website, the reputation value of a seller is based on the ratings given by other users. A rating given by another user who made direct contact with the seller will be recorded after a transaction is completed. For a specific seller, confirmed buyers are able to give three possible values for ratings: (1) positive (2)



neutral or (3) negative. The reputation value is calculated as the mean of all ratings with respect to the seller over a certain period (e.g. 3 months). In all these models, the reputation value for a specific seller is global and unique, which means it is a context-independent and domain-nonspecific value. These reputation models are simple but effective, although a drawback of this model is that (1) biases and false information intended to mislead the customers might affect the result. (2) For a specific customer, the model provides a single reputation value, so it is not possible to differentiate a seller's reputations in different domains (e.g. a manufacturer specialised in mobile devices may not have the same reputation when it comes to mobile accessories).

### 2.7.2 Sporas and Histos

Sporas is a reputation model derived from simple WOM-based model as presented in section 2.7.1. In contrast to the reputation models adopted by Amazon and eBay, which are the mean of all ratings over a certain period, the Sporas model only takes into account the most recent rating. The rating will be updated when there is any changes in circumstances. An important features of this model is that, users with higher reputation values are less likely to be affected by the update, which means a user with higher reputation will experience smaller ratings changes when the data is updated, similarly a user with lower reputation is more likely to be affected by the updates.

Histos is another reputation model where pairwise ratings are depicted as directed graphs which nodes representing different users and the edges between nodes describing the latest rating values given by one user to another. The first step to calculate the reputation of a specific user  $u$  is to label the user as the "root node" of the graph and the reputation of user  $u$  can be computed recursively as a weighted mean of all its immediate neighbors, where the weights are the reputation of other users' rates  $u$ . In this model, it is also possible to restrict the number of connected paths or limit the depth of recursions.

## 2.8 Categorisation Techniques

Various algorithms and models have been proposed to solve the classification problem. This section will give a brief overview of some existing approaches and discusses their advantages and shortcomings.

- Description Logic A-Box classification
- k-Nearest Neighbours
- Support Vector Machines
- Naive Bayes
- Neural Networks

### 2.8.1 Description Logic A-Box Classification

One of the most common classification approaches used in ontology-based classification is the ABox reasoning which is related to assertions on individuals. TBox (terminological box) and ABox (assertional box) are two different types of statements in the ontology [58]. Tableau algorithms [62] can be used to check satisfiability (or consistency) of the ontology (since the DL reasoning is not the main research focus, we will not discuss the details of tableau algorithm here). Classification can be seen as a procedure to determinate whether a specific individual  $a$  is an instance of  $C$ . All standard DL reasoning problems can be resolved through a decision process to solve consistency. Therefore the task to check if an instance  $a$  belongs to a category (class)  $C$  is equivalent to solving consistency question " $O \models a : C$  iff  $O \cup \{a : \neg C\}$  is not consistent" using tableau algorithm. Tableau algorithm is robust and works well under ideal situations, but it also has its limitation. For example, It is often the case that some degree of uncertainty is involved, this results in undecidable satisfiability problems that tableau algorithm cannot solve.

### 2.8.2 Clustering

The use of clustering algorithms for objects categorisation is very common. Various clustering techniques such as k-Nearest Neighbours, Density-based spatial clustering of applications with noise (DBSCAN) are developed, many of them can be used to group objects into clusters according to their characteristics. But there are still problems with clustering techniques. For example, dealing with high-dimensional dataset could become complicated. Another problem is that categories (natural clustering) are generated automatically as a result of unsupervised learning, which is not always consistent with the desired categories we want. Besides, for most distance-based clustering algorithms, their performance is depending on the definition of "distance", which is not always obvious in many cases. More detail regarding the experiments conducted using clustering algorithm will be discussed in Chapter 5.1.2.

### 2.8.3 Support Vector Machines

In contrast to clustering, Support Vector Machines (SVM) [63] are based on supervised learning. Object classification can be performed using Support Vector Machines. In [64], SVMs are defined as follows: "SVMs are based on the Structural Risk Minimization principle from computational learning theory. The idea of structural risk minimization is to find a hypothesis  $h$  for which we can guarantee the lowest true error. The true error of  $h$  is the probability that  $h$  will make an error on an unseen and randomly selected test example". SVMs has been proven to be very efficient in categorising text content. However, one of its drawbacks is the problem with class membership probabilities, which is inevitable when uncertainty is involved.

### 2.8.4 Naive Bayes Classifier

Naive Bayes classifier is another popular classification method based on Bayes' Theorem. It is a simple yet efficient method for text categorisation comparing to some more advanced methods such as SVMs or Neural Networks. Naive Bayes classifier is based on a conditional probability model, assuming that all features of the object are conditionally independent of each other. Naive Bayes classifier shows surprisingly good

performance despite the fact that this assumption is not always true. In this thesis, a customised Naive Bayes classifier will be introduced to classify annotated items. More details about this will be discussed in Chapter 5.2.4.

### 2.8.5 Neural Networks

Artificial Neural Network (ANN) [65] is an approach widely used in pattern recognition and object classification inspired by human biological nervous systems. Similar to a human brain, which has a large number of nerve cells (neurons), artificial neural networks consist of a series of interconnected neurons. An artificial neuron is a unit that takes several inputs and produces one output. These "units" are activated under certain conditions, and they begin communicating with each other until an output unit is activated. Theoretically, Neural Network has a great potential to achieve the learning abilities of human brains by imitating the way how they process information, but we still cannot resolve some issues efficiently. For example, ANNs are more like black boxes that do not reveal the underlying procedure how the results can be achieved, and that means it is difficult to express learned knowledge using ANNs. ANNs perform well in pattern recognition but are not so good at incorporating learned domain knowledge or existing rules into a knowledge base.

### 2.8.6 Social Annotation-classification tool: Zooniverse

Several applications have been developed that combines collaborative annotation with classification. Zooniverse [27] is a social tagging platform where volunteers can help researchers classify images by answering a few simple questions relating to images of certain types (e.g. galaxies, whales, etc.). Anyone can contribute to the project so volunteers do not need any expertise or training.

The motivation of Zooniverse is similar to the proposed collaborative classification framework developed in this research, but there are many differences. For example, volunteers on Zooniverse have no experiences or professional training. Therefore, manual classification depends entirely on pre-defined multiple choice questions given by the researchers. In contrast, with the proposed collaborative classification framework,

participants are domain experts (e.g. professional archaeologists) and the classification decisions are based on their interpretations and analysis of the evidence. In this case, participants need to indicate which evidence had led to the conclusion. Also, Zooniverse model does not support uncertainty and the majority of the questions are basic true/false questions whose expected answer is either "yes" or "no".



# Collaborative Annotation

---

This chapter describes a framework for image annotation and search in collaborative working environments to address the problems listed earlier. The framework is built on the basis of an *Image Annotation Ontology*, which functions as an abstract data model for organising and storing tagging data. It provides the infrastructure for annotators to link a tag to a predefined concept within a conceptual-semantic lexical database. This forms our knowledge base. Therefore a reasoner can be used to exploit implicit knowledge because we stored the tagging information in a machine readable and understandable way. The tagging framework also allows users to define certainty factor, which is the degree of confidence towards a statement. The certainty factors will subsequently be used in conjunction with user credibility to compute the truthfulness of statements. This additional user context will significantly improve the efficiency and accuracy of a query. The main components and steps will be illustrated with a few examples in the following sections.

## 3.1 Ontology-based Data Model

This section will present an ontology-based data model for collaborative annotation and discuss how the uncertainty of the assumptions and users' domain-specific reputation can be captured and represented in this model.

### 3.1.1 Collaborative Annotation Ontology

To allow for reasoning and structuring the tagging information an underlying structure is needed. The basic structure of the Image Annotation Ontology (as shown in Figure 3.1), allows storing information about a tagging area and the relevant annotators as

in conventional image tagging system. In addition to that, assumptions concerning a subject-predicate-object triple or a link between tagged area and resources can be stored. We associate an original (raw) certainty factor (given by the annotator) with every assumption. Each user is assigned a set of credibility values for the different domains in which they annotate according to their expertise and reputation in that field.

Resources are a group of synonyms defined in the WordNet lexical database, where they are organised into hierarchies by hypernym (or hyponyms). A hypernym is referred as an “is a” relationship. For example, the phrase *waggon is a vehicle* can be used to describe the hyponymic relationship between waggon and vehicle. The WordNet hierarchy for the word “waggon” ( noun.artefact) [66] is shown in Table 3.1.

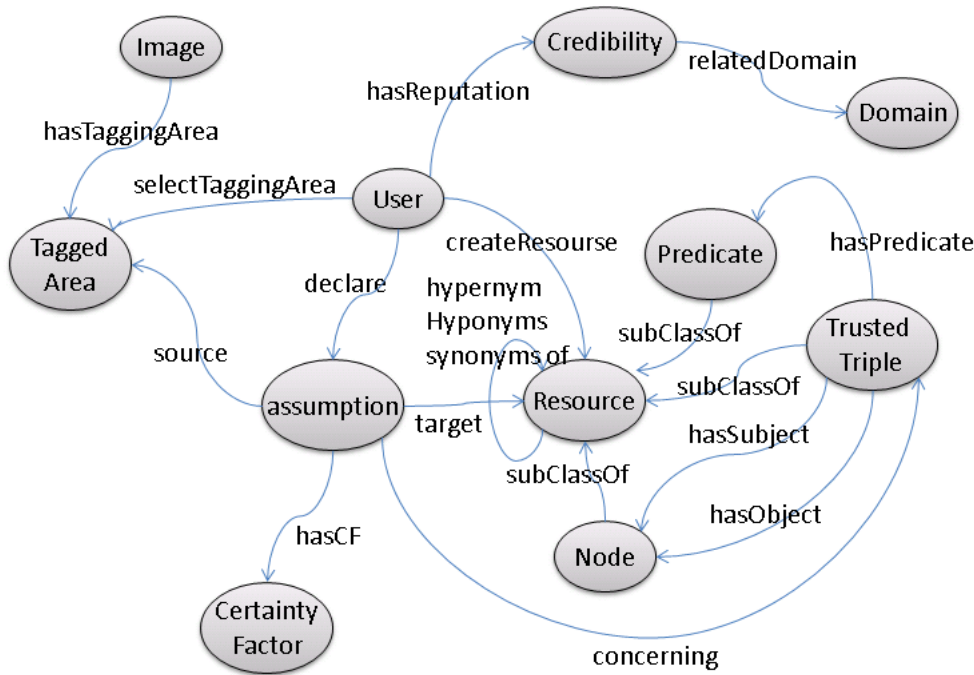


Figure 3.1: Image Annotation Ontology Overview

While linking a tagging area to a term defined within WordNet, the definition of the phrase as well as the whole branch of the hypernym tree describing the semantic relations between these nodes, will be added to the triple store to populate the knowledge base. A user is free to create new terms by adding a new branch to the tree. A new



Table 3.1: hypernym, hypernyms and synonyms of “wagon” in WordNet

```

oxcart
  cart
    wagon, waggon
      wheeled vehicle
        vehicle
          conveyance, transport
            instrumentality, instrumentation
              artefact, artefact
                .....

```

term is defined if only of all leaf nodes in its branch are linked to already predefined words.

Using WordNet provides us with an ontology that is not domain specific, but rather generic and maintained in a collaborative fashion. Also, it is not unique to our application, and hence maintenance becomes a property somewhat distant from our immediate concerns.

### 3.1.2 Uncertainty of the Assumptions

The subject-predicate-object relation mentioned in the previous section is common to ontologies and the typical structure used for storage is a Triple Graph. We are proposing an extension to Triple graphs also to include a certainty factor. A Certainty factor (cF) [67] is a number from -1.0 to 1.0, indicating how certain or how accurate a user feels about an assumption. Positive certainty means the user believes the assumption is a true statement, but possibly not 100% certain. A negative certainty means a disagreement with a given assumption but cannot rule out the possibility. For example, a statement such as “The person carrying a shield with his left hand is a warrior” might be given a certainty factor 0.95 by an expert in ancient human representation who would be reasonably confident. However, as an amateur archaeology enthusiast (like me) then

you will probably give a lower certainty number, say 0.5.

Assume  $cF(u, s)$  is the original certainty factor given by a user  $u$  for assumption  $s$ . To combine the certainty factor and user reputation, we define the concept of a composite user certainty-credibility factor  $CF(u, s, r_d)$ . This applies to an assumption  $s$  made by a user  $u$  with reputation  $r$  in a domain  $d$ .  $CF(u, s, r_d)$  is defined as follows:

$$CF(u, s, r_d) = r_d \cdot cF(u, s) \quad (3.1)$$

where

- a statement  $s$  is a subject-predicate-object triple or an assumption about the relationship between tagging area and a predefined concept,
- $cF(u, s)$  is the certainty factor given by user  $u$  on  $s$ , and
- $r_d$  is the user's reputation  $r$  in domain  $d$ .

We will come back in section 3.1.3 to explain how  $r_d$  is calculated. To compute the overall certainty-credibility factor  $CF_s$  for a statement based on all available judgements of all users (e.g.  $u1, u2$ ) over the same statement  $s$ , we combine  $CF(u_1, s, r_{d1})$  and  $CF(u_2, s, r_{d2})$  using the parallel function [67] in Equation 3.3:

$$CF_1 = CF(u_1, s, r_{d1}), CF_2 = CF(u_2, s, r_{d2}) \quad (3.2)$$

$$CF = \begin{cases} CF_1 + CF_2(1 - CF_1), & \text{if } CF_1, CF_2 \geq 0, \\ CF_1 + CF_2(1 + CF_1), & \text{if } CF_1, CF_2 < 0 \\ \frac{CF_1 + CF_2}{1 - \min(|CF_1|, |CF_2|)}, & \text{otherwise} \end{cases} \quad (3.3)$$

For example, consider the statement “a hunter is riding a horse” shown in Figure 3.2. For  $CF_1 = 0.8$  and  $CF_2 = 0.7$ , we have

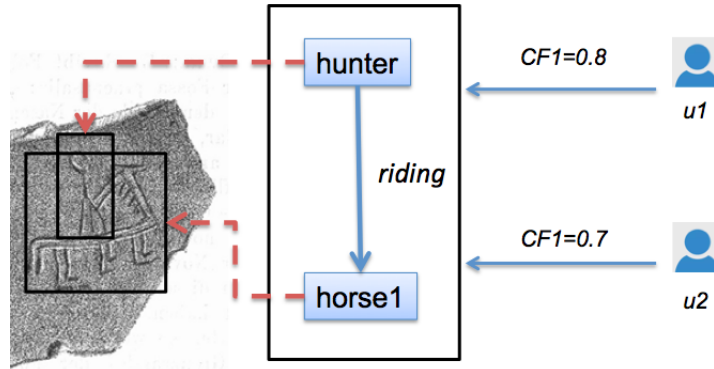


Figure 3.2: An example of user's opinion on a statement: a hunter is ride a horse

$$CF = 0.8 + 0.7 \cdot (1 - 0.8) = 0.94$$

In this case, the overall composite user certainty-credibility factor of this statement is 0.94, significantly above the average of two individual users' claims, which might come as a surprise. However, there is a certain sense in this: if many users tend to claim the same thing with reasonable confidence then one can overall be more confident in it being true and the effect of people being somewhat 'shy' in claiming certainty becomes reduced. However, it also stops us from arguing that a statement is 'twice as true' as another just because it has double the score.

### 3.1.3 Domain-Specific User Credibility Measurement

The credibility of a given user in a particular domain can be measured using two reputation models from different perspective:

- Comparing past activity with community average
- Feedback ratings from reviewers

The sections below will explain two models respectively in section 3.1.3.1 and 3.1.3.2. How these two models can be combined will be described in section 3.1.3.3.

### 3.1.3.1 User Credibility Measurement Based on User Activities

We need to obtain the reputation [68]  $r$  of a user to calculate the user certainty-credibility factor. The expertise factor [69] defines the degree of a user's competency to provide an accurate prediction in a particular field. In our case, the reputation  $r_a$  of a user  $u$  in a domain  $d$  based on their past activity is defined by (3.4)

$$r_a(u, d) = \beta(u, d) \left( 1 - \frac{\sum_{s \in D(s)} \sum_{x \in U(s)} |cF(u, s) - cF(x, s)|}{|D(s)|} \right) \quad (3.4)$$

where

- $cF(u, s)$  is the original certainty factor given by user  $u$  on  $s$
- $U(s)$  is a set of users who commented on statement  $s$  (excluding user  $u$ )
- $D(s)$  is the statement set in the domain of target statements  $s$
- $\beta(u, d)$  is the activity weighting and is defined as  $1 - (1/n)$  ( $n$  is the number of comments within this domain  $d$ ), meaning that a user will be assigned a higher value of expertise if making more comments for more statements within a particular category. (Note: we are not claiming that just because someone makes lots of comments, he or she is more qualified to do so).

As can be seen in Equation (3.4), the reputation of an annotator is determined by several aspects. In general, the more statements in the same category a user commented, the more likely are they to have expertise in the field. This result will increase their reputation in this category, but only if the opinions the user provided reflect the truthfulness of the actual statement as measured by the similarity between their judgment and decisions made by other users. Moreover, also we should consider, the higher positive certainty value one got from all statements he or she created, what this user said is more likely to be true.

For example, a pottery expert  $A$  commented on many pictures of Roman pots and in most cases, these observations were very accurate, so generally  $A$ 's opinion in this domain does not differ much from the general public including other pottery experts. The majority rule is applied in the group decision-marking process. Our assumption

is that the majority of the community members make reasonable choices. [70]. If the community generally accepted the annotations added by  $A$  then  $A$  should have a higher reputation in the domain relating to pottery. While (say) a computer scientist  $B$  also made lots of comments on pottery, but those opinions received negative feedback; therefore  $B$  is less credible than  $A$  in this particular domain.

Users made statements with certain degrees of uncertainty so they could be subjective. The general approach adopted in this model is based on the rule of the majority, which is the foundations of democracy. Instead of deriving reputation from absolute correctness or accuracy of disputed claims (which often have no absolute proof of correctness), Equation (3.4) measures the user domain-specific reputation based on the level of confidence toward statements in a given domain. The underlying assumption is that, if the majority of the users with a similar degree of confidence endorsed a statement then we treat such statement along with the collective attitude towards it as the closest truths (or the most sensible choice). So the reputation score is calculated based on the similarity of the confidence level towards a statement between the user and the community average (the closest truths).

#### Worked Example of Credibility Measurement (User Activity Based)

Considering a set of statements  $S$  in domain  $d$  (size=100). Suppose person  $p_1$  made twenty comments in domain  $d$  and these comments have relatively high confidence level (e.g. average  $cF_{p_1} = 0.9$ ). Assuming all other eighty comments (excluding those claimed by  $p_1$ ) all have the same  $cF = 0.6$ . According to Equation 3.4, the reputation of  $p_1$  in domain  $d$  is denoted by  $r_a(p_1, d)$ . In this case,  $r_a(p_1, d) = 0.722$ ; If  $p_2$  made only three comments with the same degree of confidence then the overall  $r_a(p_2, d) = 0.343$ . Similarly, if  $p_2$  made fifty comments in domain  $d$  with overall confidence slightly lower than average (e.g.  $cF_{p_2} = 0.5$ ) then  $r_a(p_2, d) = 0.798$ . If  $p_2$  only made three comments then  $r_a(p_2, d) = 0.537$ . A summary of this worked example can be found in Appendix A.3.

Table 3.2: Conversion of linguistic terms to review scores (RS)

Linguistic terms	review score (RS)
strongly agree	0.9
mostly agree	0.6
slightly agree	0.3
no opinion	0
slightly disagree	-0.3
mostly disagree	-0.6
Strongly disagree	-0.9

### 3.1.3.2 Extracting Contributor Reputation From Peer Reviews

This section explains how to consider feedback on statements given by contributors to determinate the quality of their contributions. In a real world, some of the contributors produce high-quality annotations while there might be data from spammers or users who deliberately abuse the system. It has become increasingly important to identify annotations from users with higher credibility. That means the more reliable the source, the higher weight should be given to its annotation accordingly.

#### Peer Reviews Model

A reputation model called *annotation-reviews* model is adopted. The model introduces a "was this annotation helpful to you?" mechanism that has been used by some commercial sites to help customers find the most helpful reviews [71]. In this way, a user can be both a statement contributor and a reviewer. Any user can annotate items with RDF triple statements along with uncertainty, he or she can also review statements given by others.

*Review Score* can be used to describe how far a user agrees with a particular statement annotated by another user. *Review Score (RS)* is a number from -1.0 to 1.0, indicating how strong a user support a given statement. A positive *RS* means the user generally agrees with the annotator's opinion while a negative *RS* means a disagreement with a given assumption. Table 3.2 shows how to convert linguistic terms to review scores.

For example, given a specific RDF statement  $s(subject, predicate, object)$  annotated by user  $u_1$  whose topic is in domain  $d$  as shown in Figure 3.3. Two other users,  $u_2$  and  $u_3$ , as reviewers, each provided his or her feedback concerning  $t$ . User  $u_2$  strongly agrees that  $t$  suggested by  $u_1$  is an accurate reflection of what the annotation target is. As a result,  $u_2$  is given a positive RS score  $0.9$  with respect to  $t$ . In contrast, user  $u_3$  mostly disagrees with user  $u_1$ . In this case,  $u_3$  is assigned a negative RS value of  $-0.6$ .

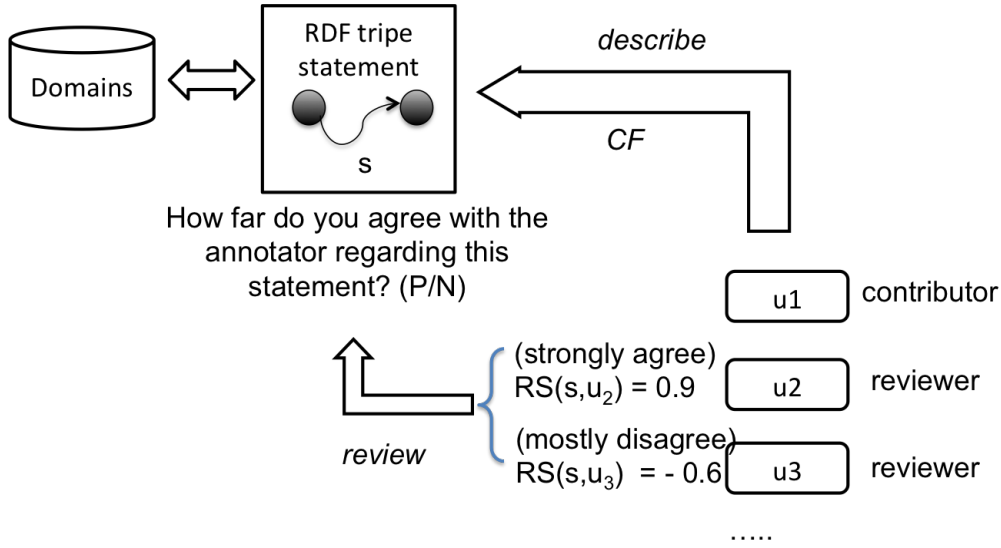


Figure 3.3: Reviewing annotations

### Summarising user feedback

Once feedback has been collected, the next step is to combine and aggregate peer reviews to calculate contributors' feedback-based reputation. The positive feedback rates for user  $u$  in domain  $d$  can be calculated using Equation 3.5.

$$r_{pr}(u, d) = \left( \frac{\sum_{s \in S(d, u)} RS(s)}{|S(d, u)|} \right) \quad (3.5)$$

$$RS(s) = \left( 1 - \frac{1}{n_s + 1} \right) \left( \frac{\sum_{u' \in U(s)} Pos(u', s)}{|\sum_{u' \in U(s)} Pos(u', s)| + |\sum_{u' \in U(s)} Neg(u', s)|} \right) \quad (3.6)$$

Where

- $r_{pr}(u, d)$  is the reputation of user  $u$  in domain  $d$  derived from peer reviews.

- $RS(s)$  is the overall review score for an RDF statement  $s$ .
- $S(d, u)$  is a set of statements in domain  $d$  claimed by user  $u$ .  $|S(d, u)|$  is the number of statements in domain  $d$  made by user  $u$ .
- $1 - 1/(n_s + 1)$  is the feedback weighting,  $n_s$  is the number of reviews given to  $s$ .
- $Pos(u, s)/Neg(u, s)$  are the positive/negative review score given by user  $u$  with respect to statement  $s$ .
- $U(s)$  is a set of users who commented on  $s$  (excluding  $u$  who made this statement).

### Worked Example of Credibility Measurement (Peer-review Based)

Considering two users  $p_1$  and  $p_2$ . Suppose  $p_1$  has made ten statement:  $s_1...s_{10}$  in domain  $d$ . For simplicity purposes, we assume that each statement received ten reviews, and nine of them have positive review scores (0.7) while one of them is negative (-0.5). According to Equation 3.5, we can conclude that  $s_1$  claimed by  $p_1$  have an overall positive review score  $RS(s_1) = 0.84$ , hence under our assumptions, the overall peer-review based reputation score of  $p_1$  in domain  $d$  will be  $r_{pr}(p_1, d) = 0.706$ .

Similarly, suppose  $p_2$  also made ten statements:  $n_1...n_{10}$  in domain  $d$ . Under the same condition, suppose each statement received ten reviews and only one of them being positive (0.7) and the rest of them are negative (-0.5). In this case, according to Equation 3.5,  $RS(n_1) = 0.12$ , the overall peer-review based reputation score of  $p_2$  in domain  $d$  will be  $r_{pr}(p_2, d) = 0.06$ . This worked example is illustrated in Appendix A.3.

#### 3.1.3.3 Combining Two Reputation Modules

In section 3.1.3.1, we discussed the method of getting the domain-specific reputation of a given user based on the similarity between this user's annotation and the annotations contributed by the other users in the community (Equation 3.4). Section 3.1.3.2 explained how feedback-based reputation can be computed (Equation 3.6).

A given user's overall credibility is determined by both (1) The accuracy of user's annotations derived from average community ratings and (2) Positive feedback rate



derived from peer review scores. Therefore, peer review reputation scores  $r_{pr}(u, d)$  (as discussed in Equation 3.6 and 3.5) should be combined with activity-based reputation scores  $r_a(u, d)$  (as discussed in Equation 3.4) to become the overall credibility:

$$r = r_a + r_{pr}(1 - r_a) \quad (3.7)$$

The equation above (3.7) is used to combine two reputation models. (Note that we refer to  $r_a(u, d)$  as  $r_a$  and  $r_{pr}(u, d)$  as  $r_{pr}$  if it is clear which user/domain is concerned.)

## 3.2 Integrating Existing Dataset

This section highlights the problems and challenges that arise in RDB-to-Ontology transformation.

### 3.2.1 Problems with Mapping

Relational database modelling concerns the normalisations of data before it can be stored in an RDB. RDB schema defines the structures of rows and the relationships between rows. One of the limitations of defining a schema in such a way is that the relationships are implicitly defined. Tables in an RDB are related via primary and foreign key constraints. However, it does not give a clear indication of the relationships. The most interesting and useful aspect of an ontology is its ability to capture and explicitly define relationships, but this makes the transformation process not trivial.

Secondly, when converting data from an RDB to ontologies, the traditional scripting languages used provide simple mapping rules. Existing frameworks [72] [48] map tables to RDF and OWL classes and columns to properties in OWL or RDFS. These methods provide limited support in terms of what can be mapped. The mapping rules are generally one-to-one, i.e., one column mapped to one concept, as illustrated in Figure 3.4.

In most scenarios, the connection between a column and property is far more complex than a simple one-to-one correspondence. This is very common when the target

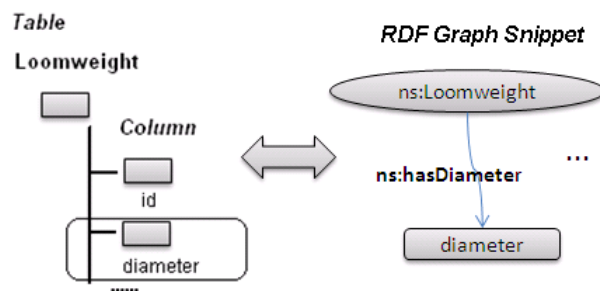


Figure 3.4: One-to-one mapping example

schemas for mapping has been extended from some standard vocabularies or those used elsewhere. Instead of generating arbitrary RDF/OWL instances from relational data sources, we would like our ontological instances to conform to a standard domain ontology such as CIDOC-CRM [40]: the CIDOC Conceptual Reference Model (CRM) provides definitions and a formal structure for describing the implicit and explicit concepts and relationships used in cultural heritage documentation". It has become an ISO standard since 2006. The latest version of CIDOC-CRM is now available (ISO 21127:2014). It could also be the case that several ontologies are used and the some might have to be mapped to more than one property.

As an example, we consider data from a sub-project in the Tracing Networks programme. This sub-project investigates loomweights across the Mediterranean and beyond, Late Bronze Age 3rd century BCE. A much neglected artefact-type, they provide valuable information about textile production on the warp-weighted loom, mainly a women's activity in many Mediterranean societies. They come in a wide range of sizes, weights and clay fabrics, partly related to the particular cloth manufactured. Loomweights can reveal social links and personal/group (often feminine) identities. Styles change broadly in harmony over a wide area of the Mediterranean, with many local variations. Types jump both across regions and across cultures (e.g. Southern Italy, where indigenous cultures adopt local versions of Greek-style loomweights, but indigenous types also appear in Greek contexts). They may be individualised with stamps, fingerprints, etc., suggesting that they were valued as personal possessions. Stamps and loomweights can be tracked geographically and chronologically (e.g. ex-

amples of 4th century loomweights with 6th century marks on them in Metaponto). Some are professionally made; others appear home-made. A systematic study of their manufacture and use over a range of contexts (kilns, houses, graves, sanctuaries) will illuminate textile production across the Mediterranean world, adding new insights on identities and social relationships, especially networks of women.

A loomweight is characterised by attributes such as the number of holes, type of stamps (impressions), thickness and diameter. Figure A.4 in Appendix A.4 shows a two-hole loomweight with imprinted meander and wave design.

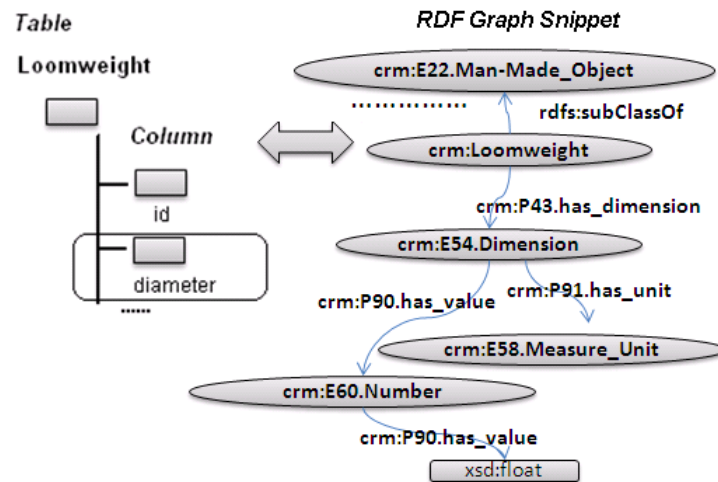


Figure 3.5: RDB table and how data can be represented in CIDOC-CRM ontology

The ontology used in this section for demonstrating the transformation procedure is extended from CIDOC-CRM as shown in Figure 3.5. We can see that the column 'diameter' in the table cannot be mapped to the target ontology in a straightforward manner, i.e., as a datatype property. To specify links between diameter and the concept "Loom-weight", we have to create several intermediate instances. These instances have to be contextually related to each other to ensure that each Loom-weight is assigned the correct diameter value.

### 3.2.2 Transformation Framework

To address this problem, a framework for transforming archaeological RDBs to ontological datasets is developed. The transformation workflow, as depicted in Figure 3.6, consists of three main steps:

- ORM Reverse Engineering
- ECA Rule-based Transformation
- Ontology Instance Generation

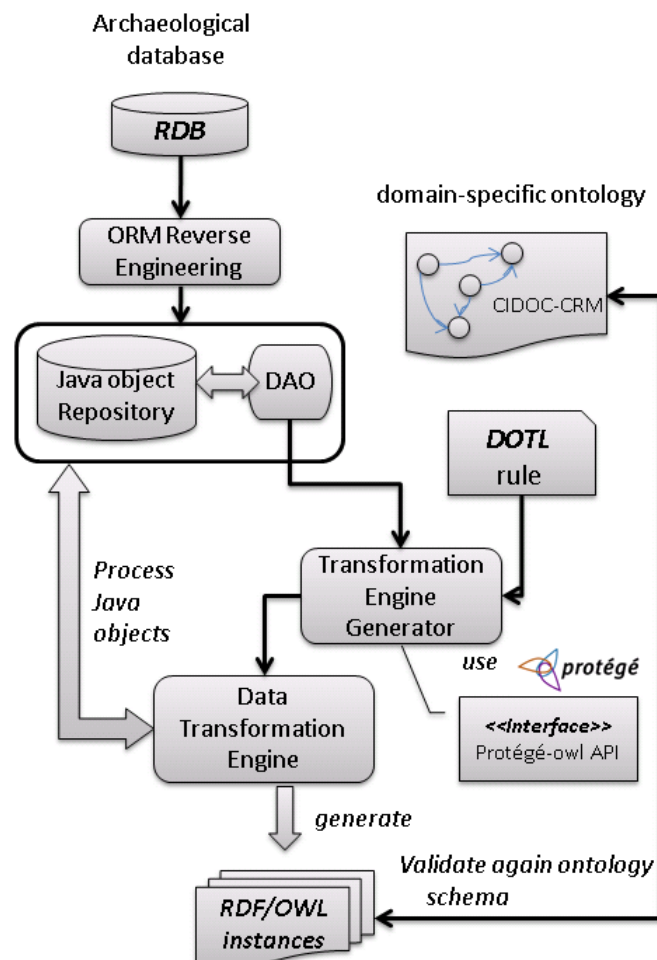


Figure 3.6: Transformation Framework.

It is worth noting that although the framework is developed for archaeological databases, the approach is generic and can be used for transforming data from most RDBs. We will explain the transformation steps in the following sections.

### 3.2.3 ORM Reversed Engineering

First of all, ORM (Object-Relational Mapping) Reverse Engineering Technique is used [73] to extract concepts and relationships from the RDB. ORM is usually used for storing persistent object-oriented data within heterogeneous RDBs. ORM Reverse Engineering technique, on the other hand, can also be used to export existing tables, columns, relationships (inc. primary key, foreign key, join, etc) and indexes from an RDB to object-oriented data structures. It can represent RDB tables as classes, and rows in a table can be instantiated as objects, which can be manipulated and processed using Object-Oriented Programming (OOP). Since we have a huge amount of data distributed across several different legacy RDBs, e.g., MySQL and MS Access, amongst others, the transformation framework should be able to connect to heterogeneous databases to retrieve required data. One of the common data persistence frameworks is the Open Source Hibernate ORM [53] and Apache JPA framework. In this research, Hibernate ORM Reverse Engineering tool has been used for converting database records into Java objects and to generate DAO (Data Access Object) which acts as an interface to access the Java objects. Technical details and database compatibility issues in this regards are further discussed next.

### 3.2.4 ECA Rule-based Transformation

After converting datasets records into objects, the next and the most important step is to define transformation rules for these objects. The conventional mapping mechanism defines mapping rules in their languages such as D2RQ [72] and Virtuoso [74]. These approaches map tables to OWL/RDFS classes and columns to data type properties or object properties. Such method works well with straightforward one-to-one mapping case as shown in Figure 3.4. However it is not suitable for describing complex mappings as shown in Figure 3.5. The figure illustrates the kind of mapping that is

very common when transforming databases to sophisticated target ontologies such as CIDOC-CRM. To facilitate the transformation, an ECA-based [52] (Event-Condition-Action) text-based transformation language is developed — **DOTL** Database Ontology Transformation Language. Many existing mapping transformation frameworks only allow one-to-one mapping rules at the schema level and work only with simple models, their primary objective is to extract data from the database as ontologies without any emphasis on the resulting structure. ECA-based approach allows us to define triggers and transformation rules based on logical conditions. Also, it is possible to define a set of actions that can generate ontological instances and complex connections between them. The proposed approach is generic and can be used for large and more expressive ontology.

The fundamental construct of a DOTL transformation rule is of form:

### On Event if Condition Do Action

A basic DOTL rule consists of three parts:

- The event part specifies the triggers of a transformation rule, usually the occurrence of an object of a specific class;
- The condition part is a logical expression, which checks the pre-condition of the action to be carried out, the default conditions being “if condition is undefined”;
- The action part usually consists of a series of steps that create ontological instances and establish links among them.

The DOTL code snippet in Figure 3.7 outlines the transformation procedures as described in Figure 3.5. For each Loomweight element in the object collection, create a corresponding RDF Loomweight individual; if the value of attribute “diameter” is not null, then perform a sequence of operations as listed in Figure ???. This includes the creation of an instance of intermediate class Dimension, specifying URI pattern, establishing a link between Dimension and Loomweight instance, assigning property values.

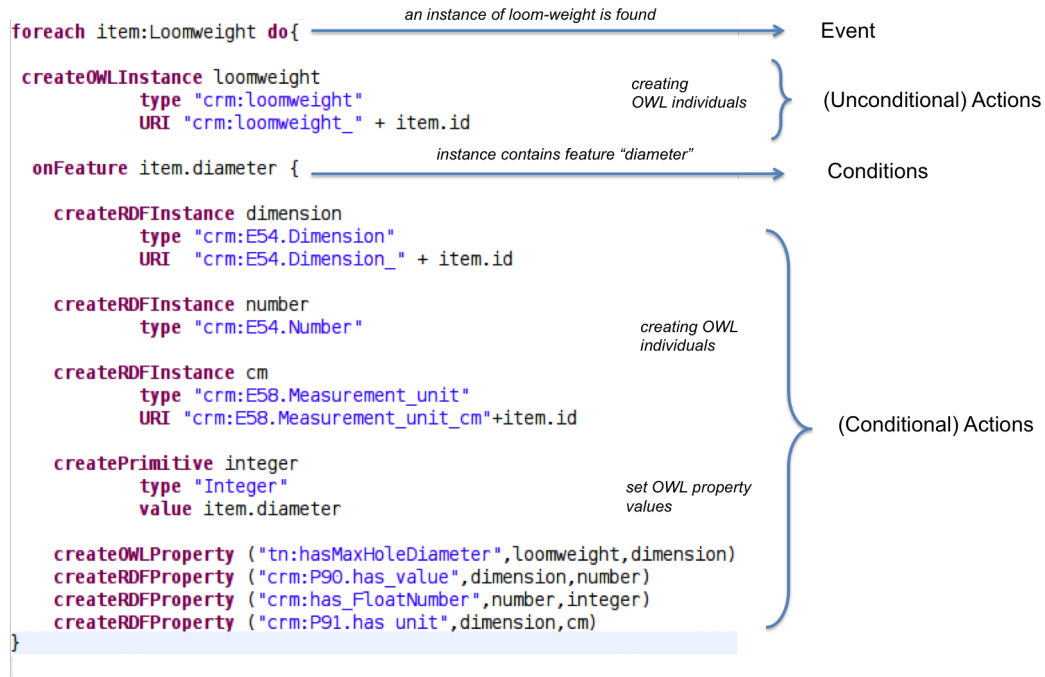


Figure 3.7: DOTL transformation rules

### 3.2.5 Instances Generation

Once the transformation rules are defined, the last process is to generate ontological instances as linked data. It is a two step process. The framework uses Transformation Engine Generator (TEG) to generate Java code that will be used to convert the database. To initiate the transformation, TEG takes DOTL rules and the DAO as inputs then generates the Data Transformation Engine (DTE). DTE is responsible for converting the data accessed through the DAO to ontological instances as per the schema defined for the ontologies. DTE is able to generate loomweight instances as linked data. Currently, instances are linked to the DBpedia [54] and Geonames [55].

Instead of creating a virtual in-memory model on the fly, the framework saves the converted datasets to a triplestore or exports them as RDF/XML documents. A prototype of the framework has been implemented in Java. The prototype uses open source Hibernate reverse engineering framework for object/relational mapping, which provides connectivity support for Oracle, DB2, SQL Server, MySQL and most mainstream relational databases. Therefore, it is possible to retrieve data from various

archaeological datasets as well as integrate data sources in a distributed environment.

A DOTL Editor plugin for Eclipse has also been implemented (Figure 3.9), which supports syntax colouring, code completion and syntax checking. Figure 3.9 shows how transformation rules as described in Figure 3.5 can be specified in DOTL editor plugin. The EBNF (Extended Backus–Naur Form) grammar of DOTL is defined in Xtext [75] syntax while the meta-model of the language is described using the EMF [75] (Eclipse Modeling Framework). The DOTL Editor also contains an integrated code generator implemented in Xpand [75] that generates Java code written in Protégé OWL API. The generated code can be used to create OWL instances.

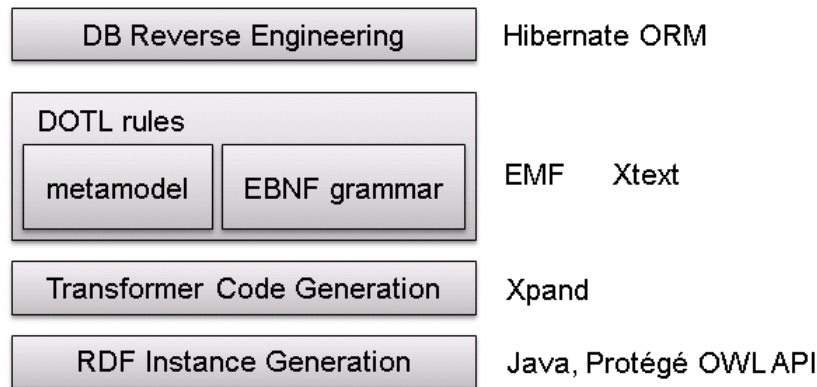


Figure 3.8: DOTL-layer

Figure 3.8 illustrates the layers of the transformation framework and relevant techniques used for the prototype.

This section describes a transformation framework for migrating archaeological data stored in RDBs to ontology-based model. An ECA-based scripting language DOTL is proposed, which allows users to specify rules that transform data objects to OWL instances. We showed a motivating example based on the CIDOC-CRM ontology as a case study. A prototype is implemented to demonstrate the methodology. One future direction in this area is to refine the grammar to enhance the expressivity of DOTL. Another ongoing development is to implement a more user-friendly graphical modeling environment for the language in GMF (Graphical Modeling Framework) to allow easy creation and editing of transformation rules. More detail will be discussed in section



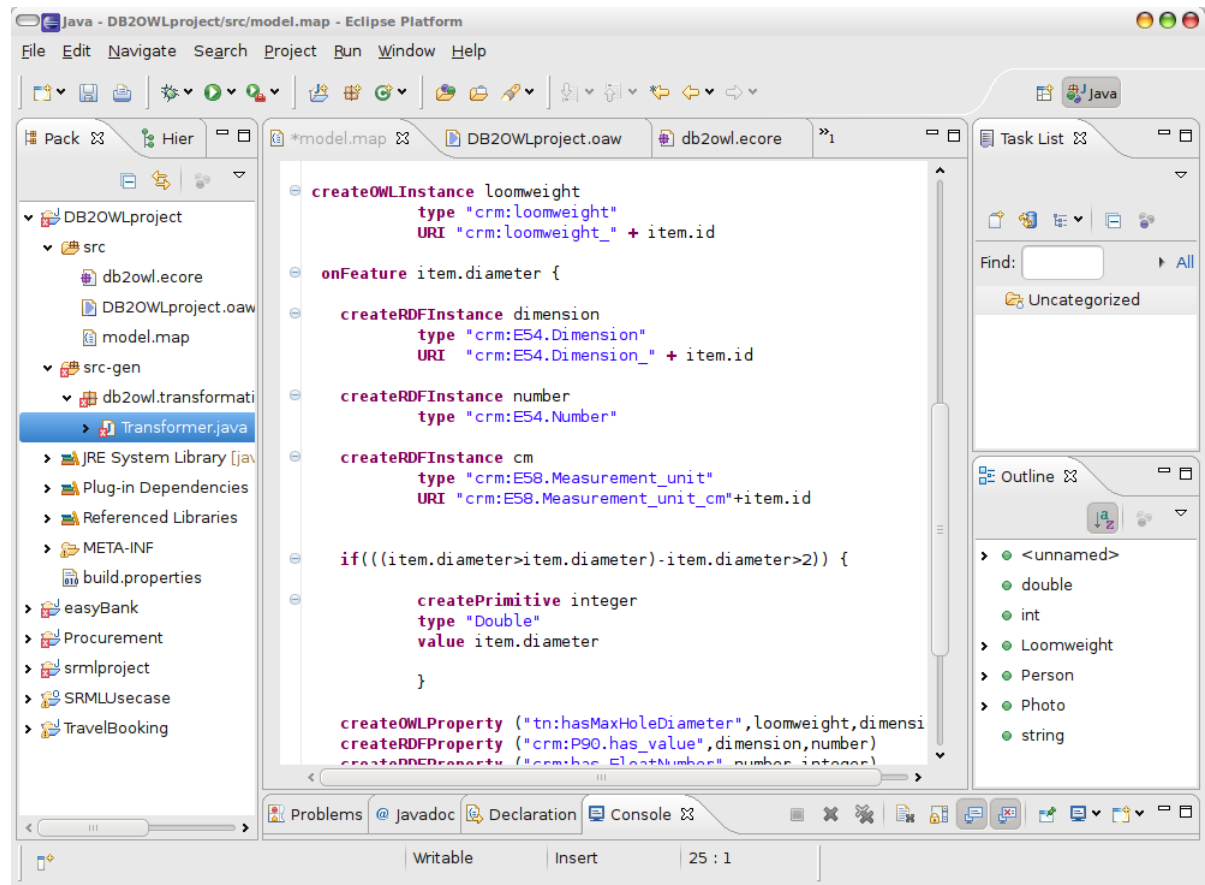


Figure 3.9: DOTL Editor plugin for eclipse

7.2.2.



# Search and Reasoning

---

As discussed in section 1.1.2.2, one of the main challenges is to consider the trade-off between ease of use and expressivity of the search interface. It is challenging to design an interface that is expressive easy to use at the same time because two requirements usually contradict each other, we have to make sacrifices and compromise between them. A native query language offers high flexibility and expressivity but it is not suitable for non-technical end-users. Form-based advanced search uses a template which consists of text fields, combo-boxes and radio buttons to help generate queries. But this approach is only capable of producing limited types of query depending on its filters. Regarding user experience, it brings unnecessary complexity and yet restrict the kinds of question user can ask.

To provide an intuitive search interface and maximising its expressivity at the same time, we propose a novel graph-based search interface to help non-technical users build sophisticated queries without the use of over-complicated forms and filters. The idea is to introduce graph-pattern as a visual tool for ordinary end users who have limited knowledge of database and query languages.

## 4.1 Graph-based Query Pattern and Colour Scheme

Figure 4.1 illustrates the architecture of our graph-based query and reasoning framework. The main idea is to allow users to draw queries as graph patterns and pass the pattern to a graph-to-query transformation engine. The transformation engine transfers the graph patterns into SPARQL queries, queries together with universal/-custom deductive reasoning rules are passed to SPARQL Query engine, queries are subsequently executed to search the RDF repository. The results of the execution can

be exported as CSV format.

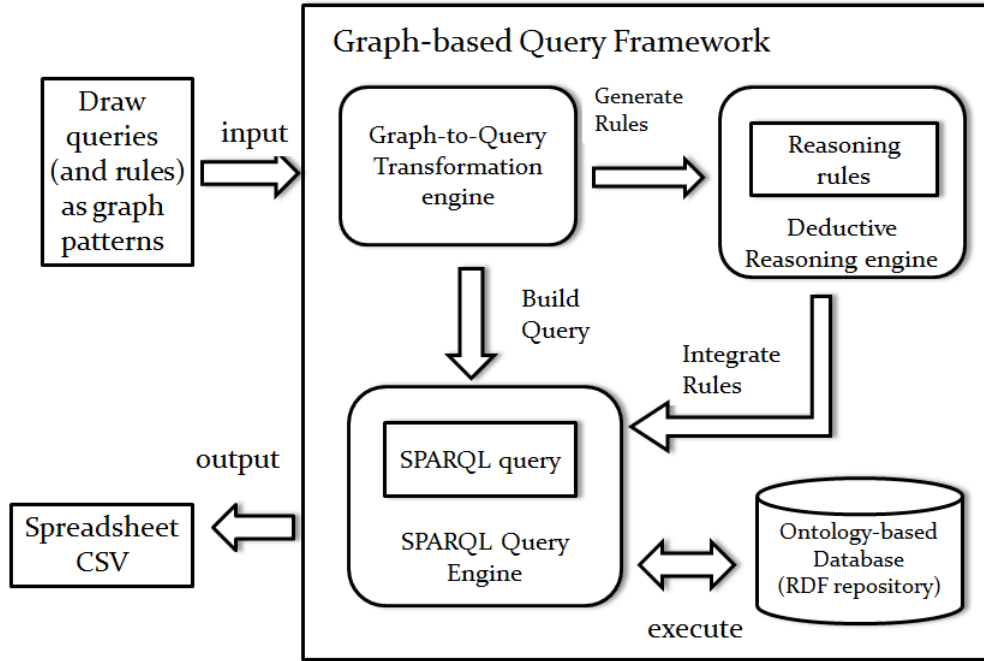


Figure 4.1: Graph-based Query and Reasoning Framework

An SPARQL query to the ontology is indeed a graph pattern. This thesis proposes propose the following graphical notation for representing SPARQL query, which consists of a set of graphical symbols and associations. In general, there are two types of main element in the proposed graphical notation: **nodes** and **edges**.

A **node** can be a variable or constant. There are several types of nodes in a query pattern, on the basis of the what they are representing, nodes are classified as follows:

- *variables*: SPARQL query variables in the graph pattern, to be included in the result set.
- *intermediates variables*: SPARQL query variables in the graph pattern but not presented in the result set.
- *constants*: named class, non-varying string, numeric or other literals.

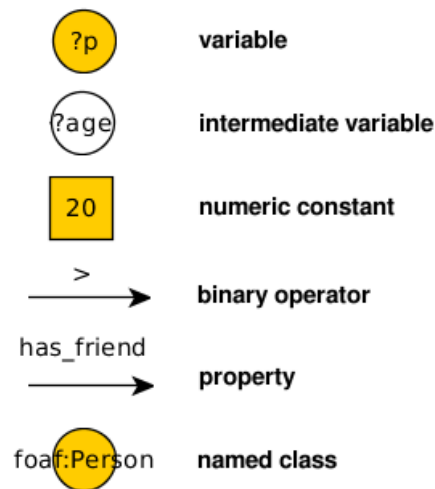


Figure 4.2: Graphical notations

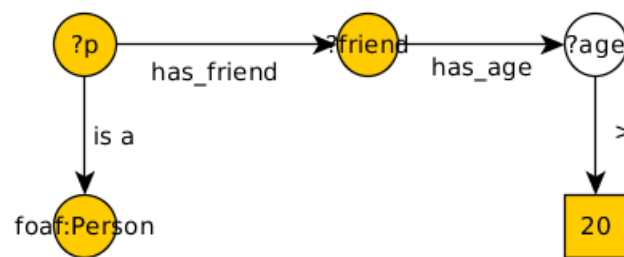


Figure 4.3: Example of graph representation of a SPARQL query

An **edge** can be a data type property OR object property OR built-in binary operator.

- *RDF or OWL property*: also known as predicate, depicted as arrows between triple nodes e.g. *has\_dimension*, *is\_a*, *has\_value* etc.
- built in binary logical operator: e.g. *=*, *<*, *>*, *starts-with*, *contains*, *ends-with* etc.

A colour-coding scheme is also introduced to support aggregate functions such as COUNT, AVG, SUM, MIN and MAX along with GROUP BY clause in SPARQL as seen in the list as followed:

- **Yellow:** nodes matched in the graph pattern and included in the result set.
- **White:** nodes matched in the graph pattern but excluded from the result set.
- **Green:** “Group by” clause will be applied to the given node.
- **Pink:** aggregate functions COUNT will be applied to the given node.

Figure 4.3 shows a query that searches for any person who has at least one friend who is more than 20 years old.

### Graphic notations and colour scheme







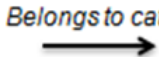
	<b>Variables</b> ( <i>any nodes, starting with ? symbol</i> ) Should be included in the result set if node matches the graph pattern.
	<b>Intermediate variable</b> ( <i>unknown nodes, starts with ? symbol</i> ) Same as variable but will be excluded from the result set.
	<b>Named class or instance</b> A node that is already known to the user
	<b>Group results</b> Equivalent to “GROUP BY” clause
	<b>Count results</b> Equivalent to aggregate function COUNT()
	<b>Numeric and string constant</b> (e.g. 10, 20.5, “abc” etc)
	<b>Property (predicate)</b> (e.g. <i>has_archaeological_context</i> , <i>excavation_location</i> , etc)

Figure 4.4: Graphical representation colour-coding scheme

Figure 4.5 illustrates the graph representation of SPARQL query for *Loomweight* database constructed in yEd graph editor following proposed colour-coding schema. The query is intended to return the number of different loomweight instances for each motif, whose dimension has an upper bound (height > 0.1). A prototype of the diagram-to-query tool has been implemented to demonstrate our methodology.

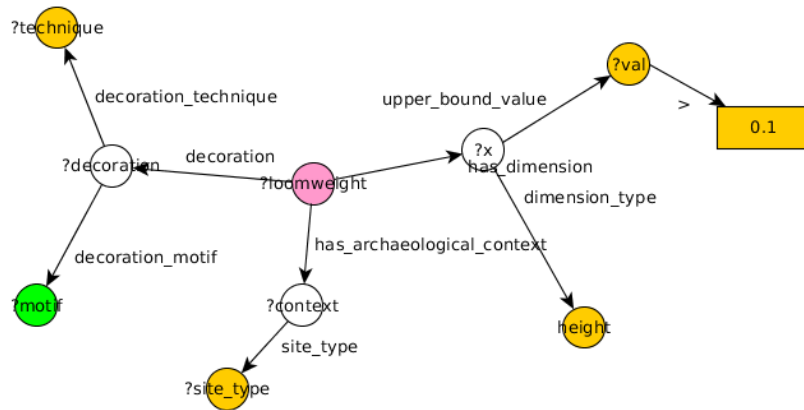


Figure 4.5: Graphical representation of an SPARQL query on Loomweight database

The graph pattern in Figure 4.5 can be translated to its corresponding executable SPARQL in Listing 4.1.

```
PREFIX tn:<http://www.tracingnetworks.ac.uk/loomweight.owl#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?motif (count(DISTINCT ?loomweight) as ?loomweight_count)
WHERE{
  ?loomweight rdf:type tn:Loomweight.
  ?loomweight tn:has_archaeological_context ?context.
  ?loomweight tn:has_dimension ?x.
  ?x tn:upper_bound_value ?val.
  ?x tn:dimension_type tn:height.
  ?loomweight tn:decoration ?decoration.
  ?decoration tn:decoration_technique ?technique.
  ?context tn:site_type ?site_type.
  ?decoration tn:decoration_motif ?motif.
}
FILTER (?val>0.1).
GROUP BY ?motif
```

Listing 4.1: SPARQL query corresponds to Query 1

### 4.1.1 Illustrative Examples: House of Menander (Pompeii) and Loom-weight (Metaponto) datasets

For clarification purposes, this section uses Tracing Networks ontology (see Figure A.1 in Appendix A.2) populated by instance data from House of Menander (Pompeii) and Loom-weight (Metaponto) datasets to explain how queries can be created using the graph notation and colour scheme discussed in section 4.1. Some of the queries presented here can be written in conventional SQL-like query language while others are not possible.

#### 4.1.1.1 Basic Triple patterns

The screenshot below (Figure 4.6) is taken from the visual query editor developed in this research. Such query shows a graph pattern that searches for bone fitting objects that were found in a room inside House I-10 in Menander dataset.

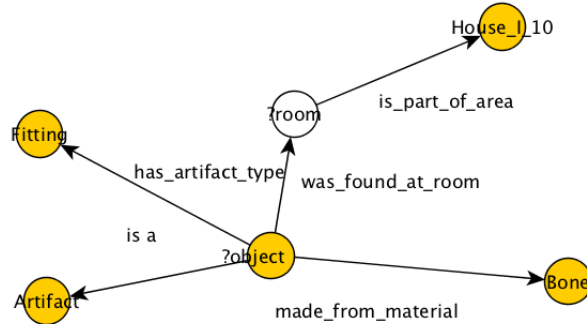


Figure 4.6: List all bone fitting objects that were found in a room inside House I 10

In this query, *artefact*, *Bone*, *Fitting* and *House\_I\_10* are named individuals with known URIs; *?object* (yellow node starting with "?") is a non-intermediate variable. Non-intermediate variables will appear in the SELECT statement of the generated query. *?room* is an intermediate variable (transparent node), hence all *artefact* instances matching this node will be excluded from the result set. As a result, this query will only return objects that matched the pattern. The SPARQL query listed in List 4.2 corresponds to the graph pattern shown in Figure 4.6.



```

PREFIX :<http://www.tracingnetworks.ac.uk/ontologies/tn-lite.owl#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX tn:<http://www.tracingnetworks.ac.uk/ontologies/tn-lite.owl#>
SELECT ?object
WHERE{
    ?object rdf:type tn:artefact.
    ?object tn:has_artefact_type tn:Fitting.
    ?object tn:made_from_material tn:Bone.
    ?object tn:was_found_at_room ?room.
    ?room tn:is_part_of_area tn:House_I_10.
}

```

Listing 4.2: SPARQL for query in Figure 4.6

#### 4.1.1.2 Aggregate Function

The graphical pattern in Figure 4.7 shows the distribution of functional category for all objects found in room 3

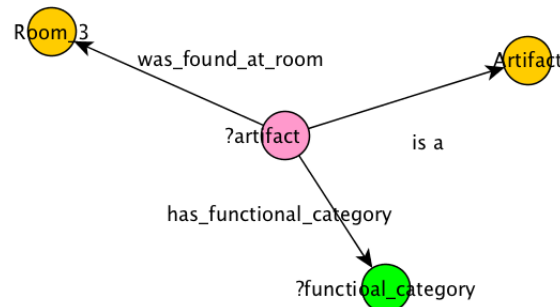


Figure 4.7: List the distribution of functional category for all objects found in room 3

```

PREFIX :<http://www.tracingnetworks.ac.uk/ontologies/tn-lite.owl#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX tn:tn-lite.owl#>
SELECT
    ?functional_category
    (count(DISTINCT ?artefact) as ?artefact_count)
WHERE{
    ?artefact rdf:type tn:artefact.

```

```

    ?artefact tnlite:has_functional_category ?functional_category.
    ?artefact tnlite:was_found_at_room tnlite:Room_3.
}
GROUP BY ?functional_category

```

Listing 4.3: SPARQL query for the graph pattern in Figure 4.7

In this case, there are two variable nodes and one object property in the graph pattern: variable `module` is a GROUP node (green) resembling GROUP BY clause in SQL; `artefact` is a node of type COUNT (pink). It means this aggregate function COUNT will be applied to a specified variable node to count all distinct objects that binds to this variable.

#### 4.1.1.3 Conditions and Filters

The graphical pattern in Figure 4.8 searches for metal objects that date back to a period which is overlapping a given time-span (400 BC to 300 BC), and keyword "food" should be mentioned somewhere in the description. This query should consider any object that is made from metal (or a subcategory of metal e.g. bronze, iron). SPARQL as an RDF-centric query language does not directly support OWL reasoning<sup>1</sup>. As a result, such query will need to be executed in conjunction with custom subsumption inference rules. Details regarding rule-based reasoning will be discussed in Chapter 4.2.

We can express binary operators (including logical and arithmetic) as a directed link in the graph pattern. For example, the query filter condition *value* > -400 can be described as shown in Figure 4.8 as two nodes and a directed edge. Variables are depicted as transparent yellow circle while literal constants and numbers are represented as rectangle boxes in the diagram. Various string comparison operators and methods such as *contains()*, *startsWith()* will be expressed in regular expressions and used in SPARQL filters.

SPARQL in Figure 4.8 is equivalent to pattern depicted in 4.4.

---

<sup>1</sup>Proposed SPARQL extension such as SPARQL-DL is aligned with OWL 2 specification but it does not fully support rule-based reasoning.

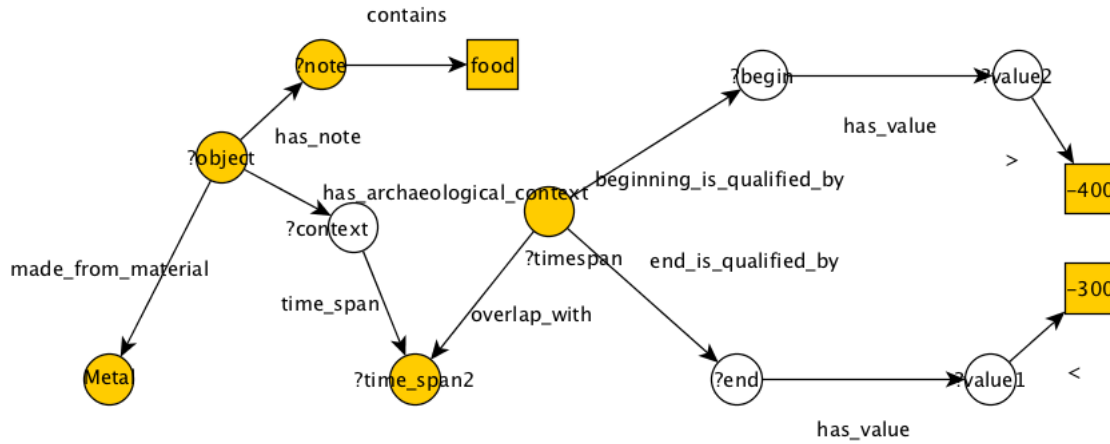


Figure 4.8: List all metal objects that date back to a period which is overlapping a given time-span (400 BC to 300 BC), and keyword "food" should be mentioned somewhere in the description.

```

PREFIX :<http://www.tracingnetworks.ac.uk/ontologies/tn-lite.owl#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX tn:<http://www.tracingnetworks.ac.uk/ontologies/tn-lite.owl#>
SELECT ?object
WHERE{
    ?object tn:has_archaeological_context ?context.
    ?object tn:made_from_material tn:Metal.
    ?object tn:has_note ?note.
    ?context tn:timespan ?timespan2.
    ?timespan tn:overlap_with ?timespan.
    ?timespan tn:beginning_is_qualified_by ?begin.
    ?timespan tn:end_is_qualified_by ?end.
    ?begin tn:has_value ?value2.
    ?end tn:has_value ?value1.

    FILTER (value2>-400 && value<-300).
    FILTER (regex(?note,"food","i")).
}

```

Listing 4.4: SPARQL query for the graph pattern in Figure 4.8

#### 4.1.1.4 Pattern alternatives

The graphical pattern in Figure 4.9 searches for loom-weights whose motif is similar to *cross* or *circle*. Blue rectangle boxes that link to the same node represent the alternative patterns in the query (UNION in SPARQL). The pattern matches if any of the alternative patterns match.

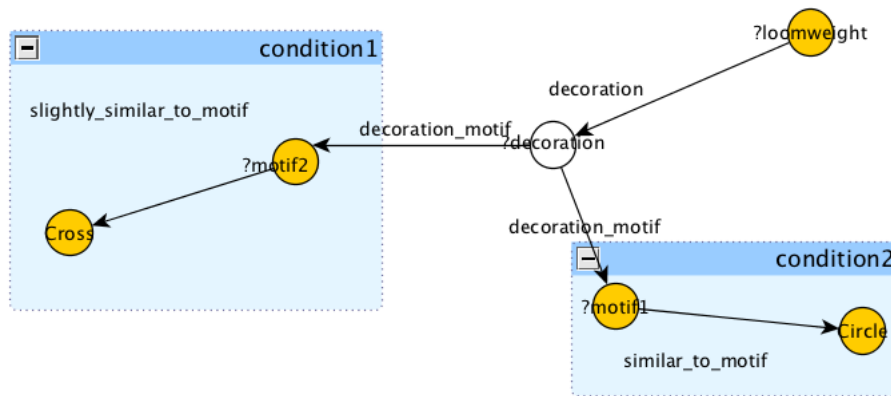


Figure 4.9: List loom-weights whose motif is similar to *cross* or *circle*

```
PREFIX :<http://www.tracingnetworks.ac.uk/ontologies/tn-lite.owl#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX tn-lite:<http://www.tracingnetworks.ac.uk/ontologies/tn-lite.owl#>
SELECT
    ?loomweight ?motif1 ?moif2.
WHERE
{
    {
        ?decoration tn:decorative_motif ?motif1.
        ?motif1 tn:similar_to_motif tn:Circle.
    }
    UNION
    {
        ?decoration tn:decorative_motif ?motif2
        ?motif2 tn:slightly_similar_to_motif tn:Circle.
    }
}
```

```
}

```

Listing 4.5: SPARQL query for the graph pattern in Figure 4.9

#### 4.1.1.5 Predicates as variables

The graphical pattern in Figure 4.10 finds any object that was excavated f House\_I\_10 and also show any possible relations between such object and *Menander\_00012*.

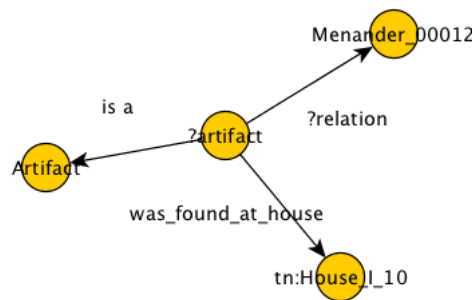


Figure 4.10: Finds any object from *House\_I\_10* and list any possible relation between such object and *Menander\_00012*

```

PREFIX :<http://www.tracingnetworks.ac.uk/ontologies/tn-lite.owl#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX tn:<http://www.tracingnetworks.ac.uk/ontologies/tn-lite.owl#>
SELECT ?artefact
WHERE {
    ?artefact rdf:type artefact.
    ?artefact tn:has_location tn:Oberstimm.
    ?artefact ?relation tn:Menander_00012.
}

```

Listing 4.6: SPARQL query for the graph pattern in Figure 4.10

In Figure 4.10, *?artefact* can be a direct instance of the *artefact* or an instance of any class that inherits *artefact*. Inferencing rule that enables transitive closure of the *rdfs:subClassOf* property is required to run this query. This query is different from others listed above as it searches for entities as well as the relationships. In other examples, relations between entities (predicates) in the pattern were explicitly given,

but this query asks for the possible links two objects. In this example, *?artefact* is a variable node and the predicate *?relation* is also a variable. As discussed earlier in section 1.1.2, querying over links between arbitrary records across several tables is problematic in RDB. But the proposed graph-based query can search for links between two given objects which can not be easily achieved using SQL-like query languages.

It is worth noting that writing some of the queries with traditional RDB query languages such as SQL is very cumbersome, if not impossible. In the above examples, the proposed graph-based searching method shows some advantages over its counterparts in RDB in terms of expressivity and usability. However, it also has its drawbacks. For example, it does not support negation-as-failure (NAF) because NAF is contradictory to the Open World Assumption (OWA), the principle of the Semantic Web.

#### 4.1.2 Implementation - Graph to Query transformation

A proof-of-concept graph to query transformation tool has been implemented which converts query patterns drawn in yEd Graph editor [76] to executable SPARQL queries. Groups of non-technical personnel have tested the tool, mainly archaeologists working in the Tracing Networks Programme [12], and many researchers in the archaeology community. A survey was conducted on the usability of the graphical query builder in the Faculty of Archaeology, the University of Leiden on 10 May 2014. More information regarding the survey and the result of the experiment will be discussed in section 6.4.

The detail of the transformation from graph pattern to executable SPARQL is explained in Algorithm 1a, 1b and 1c. Function `graph2query` takes a graph pattern as the input ( $g : Graph$ ) and converts it to an executable SPARQL query [77]. Algorithm 1a firstly traverses the graph pattern to obtains all non-intermediate variable nodes (non-blank variable node as described in Figure 4.4) as columns to be selected for the result set. The function will then construct corresponding COUNT/DISTINCT clause if the graph contains node representing aggregation function (For example, a pink and green node represents aggregation function `count()` and GROUP BY variables. The prototype only supports a limited number of aggregation functions, but the system could be easily extended to include additional functions).

**Algorithm 1a** Graph to SPARQL query transformation - *Part 1*


---

```

1:  $G$  : Graph
2:  $TS$  is a set of triples in  $G$ 
3:  $NS$  is a set of nodes in  $G$ 
4:  $variables$  is a set of non-intermediate variable nodes
5:  $q$ : Query
6: add namespace prefix and initiate SELECT statement  $q = \text{"SELECT"}$ 
7: function GRAPH2QUERY( $G$ :Graph): Query
8:   for each  $n \in NS$  do
9:     if  $n$  is a non-intermediate variable node 1 then
10:        $variables.add(n)$ 
11:     end if
12:   end for
13:    $q.setResultColumn(variables)$ 
14:   if  $NS$  contains node  $p$  where aggregation function  $AF()$  is applied 2 then
15:      $aggregate\_result = AF(p) \text{ groupBy } gb$ 
16:      $q.append(aggregate\_result);$ 
17:   end if
18:   initiate WHERE clause
19:   CREATE_QUERY_CLAUSE( $G, q$ )
20:   CREATE_FILTER( $G, q$ )
21:   if  $NS$  contains exactly one GROUP BY node  $gb$ 3 then
22:      $q.append(\text{"GROUP BY "})$ 
23:      $q.append(gb)$ 
24:   end if
25:   end WHERE clause
26: end function

```

---

The next step is to invoke Function `create_query_clause` to construct

---

<sup>1</sup>Non-intermediated variables are represented as WHITE nodes as described in Figure 4.4

<sup>2</sup>Aggregation function COUNT() is represented as PINK node as described in Figure 4.4

<sup>3</sup>GROUP BY variable is represented as GREEN node as described in Figure 4.4

WHERE clauses from subgraphs recursively. If the graph pattern contains subgraphs then the function will join multiple clauses using UNION statement.

---

**Algorithm 1b** Graph to SPARQL query transformation - *Part 2*


---

```

1: function CREATE_QUERY_CLAUSE( $G : \text{Graph}, q : \text{Query}$ )
2:    $q.append(\{ \})$ 
3:   for each immediate subgraph  $SG \in G$  do
4:     if  $SG$  contains multiple nested subgraph then
5:        $q.append(\text{"UNION{ "})$ 
6:     end if
7:     if  $SG$  does not contain nested subgraph then
8:       for each  $RDFtriplet \in SG$  do
9:         if  $t.predicate$  is NOT a binary operator 4 then
10:           $q.addTripleClause(t)$ 
11:        end if
12:      end for
13:    else
14:      call CREATE_QUERY_CLAUSE( $SG, q$ )
15:    end if
16:  end for
17:  if  $SG$  contains multiple nested subgraphs then
18:     $q.append(\text{"})$ 
19:  end if
20: end function

```

---

The last step is to find all edges of type binary logical connective (other than predicate in a triple statement) to generate FILTER clauses using regular expressions. (Note that implementation detail such as namespace, primitive data type mapping, regular expression generation are omitted in the pseudo-code for simplicity)

---

<sup>4</sup>The binary operators are depicted as directed edges in the graph as described in Figure 4.4



**Algorithm 1c** Graph to SPARQL query transformation - *Part 3*


---

```

1: function CREATE_FILTER( $G : \text{Graph}, q : \text{Query}$ )
2:    $q.append(\text{"FILTER  ("})$ 
3:   for each  $t \in G$  do
4:     if  $t.subject$  or  $t.object$  is a Literal constant5 then
5:        $q.append(\text{"regex (" + REGX}(t) + \text{") "})$ 
6:     end if
7:   end for
8:    $q.append(\text{" "})$ 
9: end function

```

---

## 4.2 Deductive Rule-based Reasoning by Graph Transformation

In this research, the data is organised as RDF/XML documents storing in triplestores. Typically querying is performed through the SPARQL, both querying and reasoning are essential steps.

SPARQL is intended for RDF. Therefore it does not natively support the OWL2 specification. SPARQL-DL [78] is an extension to SPARQL, which aims to support the OWL2 standard. SPARQL-DL can incorporate inferencing in some ways but this extension is not adequate for some complicated queries. First of all, SPARQL-DL enables all generic rules at the same time, but inferencing is selective in many cases. For example, given a query "get all instances of class Animal", one user might want to activate the transitive closure of the subsumption hierarchy to include all instances of its subclasses; others might only want to include the direct instance of this class – SPARQL-DL cannot help in this case because it would enable the OWL reasoner by default.

DataTypeProperty and ObjectProperty always have a direction from the domain to the range. We often define relations in both directions using inverse property. For

---

<sup>5</sup>Literal constant is represented as rectangle box as described in Figure 4.4

example, given an RDF triple statement "*waggon* was pulled by *oxen*". Even the statement was recorded in the other way around, the system should know that Object-Property "*was\_pulled\_by*" is the inverse property of "*ispulling*";

SPARQL-DL support OWL 2 characteristics such as transitivity and symmetricity. However, SPARQL-DL only takes into account the OWL2 specification, and it does not provide supports for custom-tailored reasoning rules. The proposed graph-based query method generates SPARQL along with deductive reasoning rules so that users can use customise reasoning rules to suit their needs.

Suppose there are two triple statements in the knowledge base: (1) Person *p1* was riding a horse, and (2) *p1* appears to be a male figure who was holding a spear. We can add one custom reasoning rule: if *h1 wasRiding h1* and *h1 rdf:type Horse* and *p1 hasGender male* and *p1 wasHolding w1* and *w1 rdf:type Spear* then *p1* is an instance of *Hunter*. When querying the type of *p1*, the system uses the reasoning rules to deduce a new statement: *p1* is an instance of class *Hunter*. Reasoning rules allow users to specify preconditions and postconditions of the inferencing procedures. An RDF graph can be seen as nodes and edges in a directed and labelled graph. These characteristics are the foundation of graph transformation theory. RDF schema can be viewed as type graph (TG) while instances of the model can be seen as instance graph (IG). Furthermore, pre- and post-conditions of reasoning rules are in fact RDF graph patterns, which means graph transformation would be an ideal way to illustrate inference mechanism. The general approach that implements the graph transformation includes four major steps (See Figure 4.11): (1) Using graph transformation to specify reasoning rules both for query and inference. (2) Rewriting reasoning rules in Jena's syntax [79]. (3) Executing SPARQL queries along with reasoning rules. (4) Getting the query results as an XML document [80].

An OWL document can be seen as an RDF graph that contains a set of triples. Both RDF subject and object are vertices  $V$  in the graph while predicates (properties) are labelled edges  $E$  that represents a relationship between them. RDF graphs include two levels of abstraction: static structure of the model can be represented in OWL schema, which corresponds to type graph TG, while a snapshot at a specified time corresponds to an instance graph. OWL provides a mechanism to specify additional

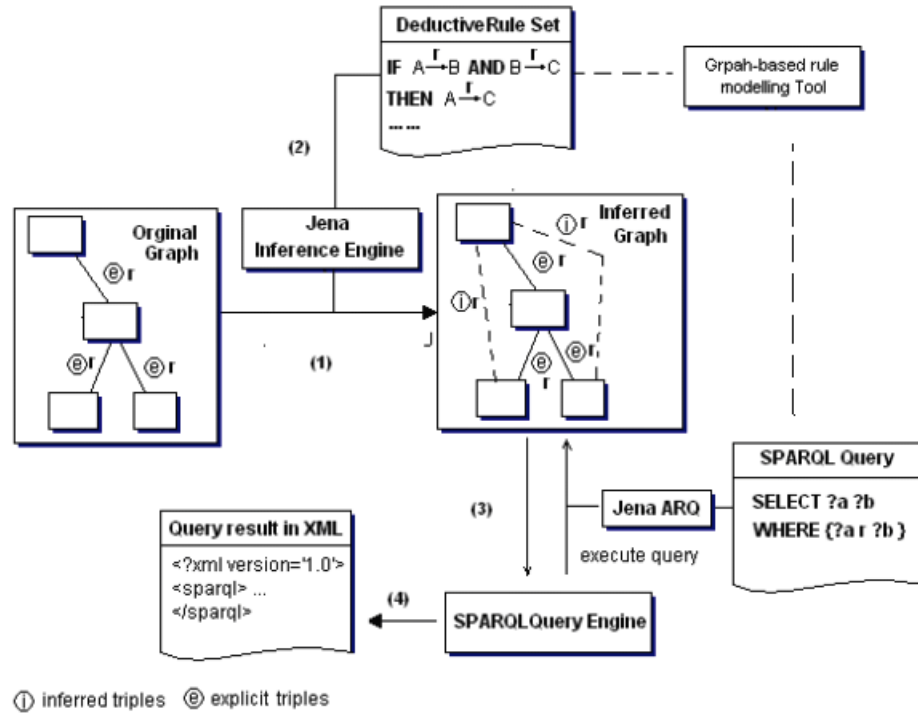


Figure 4.11: Deductive reasoning - General approach

property restrictions such as symmetric property, transitive property and functional property etc. For example, *hasSimilarDecorativeMotifAs* is a symmetric property. It means that if object *a* *hasSimilarDecorativeMotif* *b* then *b* *hasSimilarDecorativeMotif* *a*. Similarly, *isLocatedIn* relationship can be declared as a transitive property. Therefore, the approach of reasoning is to derive implicit triples from existing triples by means of graph transformation. The creation of implicit triples is modelled as graph transformation rules. Figure 4.12 shows how OWL transitive property is defined by the graph transformation rules.

A graph-based reasoning rule consists of two parts: **antecedent** and **consequent**.

- *Antecedent*: left-hand side of the graph represents the precondition in the form of graph pattern. The precondition is satisfied if there exist a sub-graph in the knowledge base matching the graph pattern.

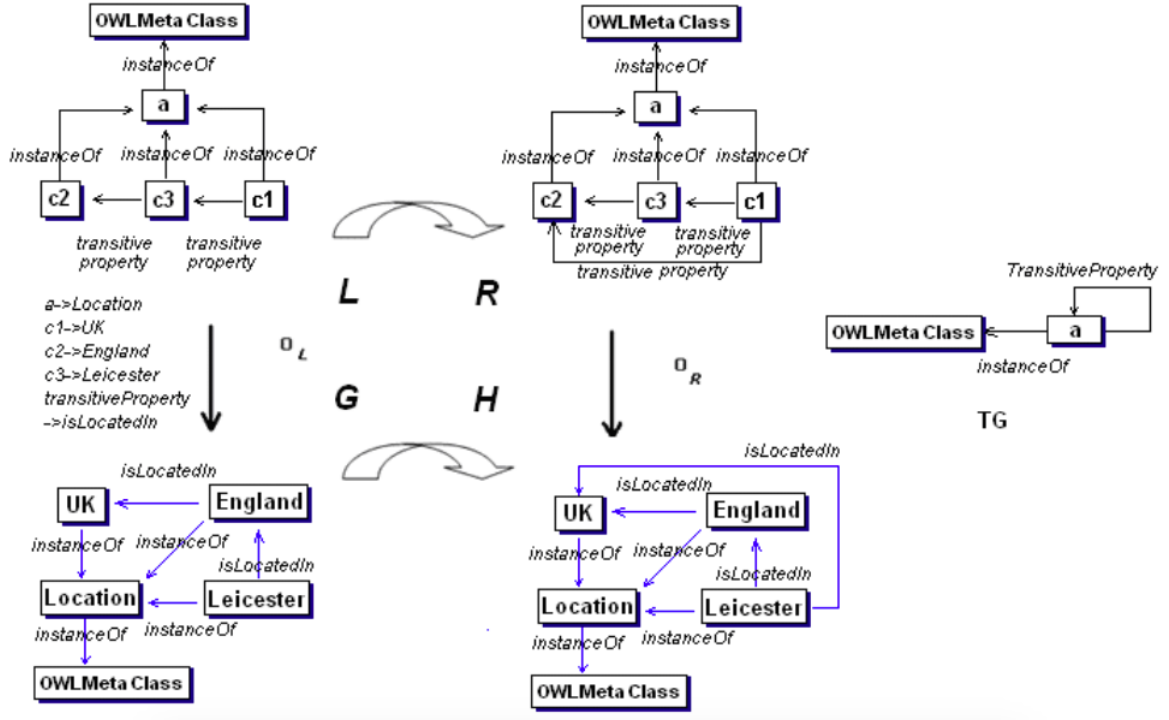


Figure 4.12: Representing transitive property with Graph Transformation

- *Consequent*: right-hand side of the graph represents an action to be taken. If any subgraph matches the pattern in the antecedent then implicit relations will be deduced and added to the knowledge base.

Similar graph transformation rules can be used to define other metamodel level properties such as symmetric and inverse property. Moreover, graph transformation rules can also specify domain-specific inferencing rules, which address problems in a particular domain. To make the graph transformation rules executable, we map abstract rules to SWRL-like [81] rules written in Jena inference syntax, which can be defined as follow:

```
[rule_name: (pre-condition 1), (pre-condition 2), ... ->
(post-condition 1), (post-condition 2), ...]
```

The left-hand side of “->” consists of a set of triple patterns which correspond to graph  $L$ , the right-hand side stands for graph  $R$ .

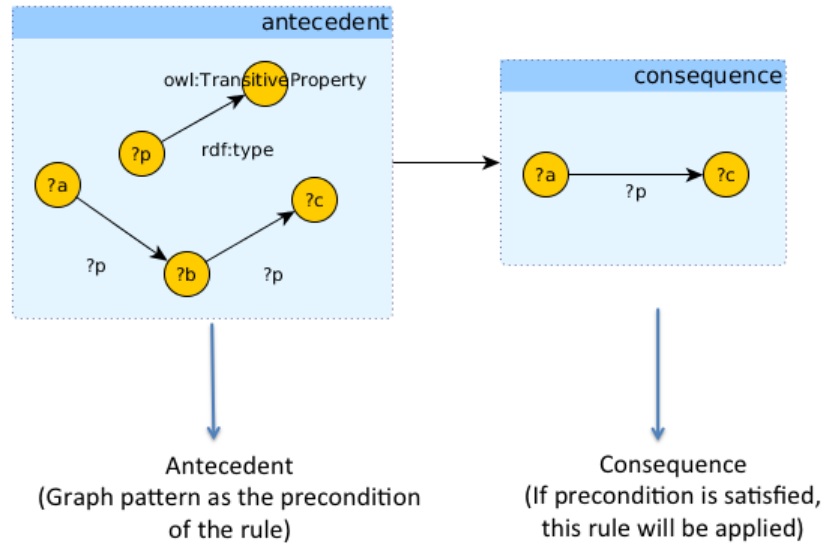


Figure 4.13: Screenshot of the graph representation of deductive reasoning rules

```
(?p rdf:type owl:TransitiveProperty) (?a ?p ?b) (?b ?p ?c) -> (?a ?p ?c)
```

Listing 4.7: Textual representation of reasoning rules corresponds to Figure 4.13

### 4.2.1 Generic Reasoning Rules

This section will discuss how some of the most commonly used generic deductive rules can be expressed in the form of the proposed graph pattern. The visual reasoning rules editor implemented supports some basic OWL characteristics. For example:

- Subsumption
- Transitivity
- Symmetricity
- Subproperty
- inverse

**Subsumption** – given a class hierarchy in Listing 4.8. When asking which classes are subclasses of *Vehicle* with a query as shown in Figure 4.14 without any reasoning, the search would only include direct subclasses of *Vehicle*. However, the `rdfs:subClassOf` property is transitive. Therefore, the result should contain not only the direct subclasses (*waggon*) but also any indirect subclasses (*Oxcart* and *Bullock\_waggon*) derived from them.

A graph-based query on its own only returns *waggon* because it does not consider the transitivity of property. However, if we combine our graph-based query patterns together with reasoning, then the information that is not explicitly represented in the knowledge base can be deduced. Figure 4.15 is a typical example of generic reasoning rule. It defines the transitive closure over the class hierarchy. If the *waggon* is a subclass of *Vehicle*; *Oxcart* and *Bullock\_waggon* are subclasses of *waggon* then *Oxcart* *Bullock\_waggon* are also subclasses of *Vehicle* too. Executing query pattern (Figure 4.14) in conjunction with reasoning rule (Figure 4.15) would get the transitive closure of the `rdfs:subClassOf` property. In such case, the subclasses of *Vehicle* should contain all its direct and indirect subclasses: *waggon*, *Oxcart* and *Bullock\_waggon*.

```
Vehicle-+
  |-waggon-+
    |-Oxcart
    |-Bullock_waggon
```

Listing 4.8: Class hierarchy: *waggon* is a subclass of *Vehicle*; *Oxcart* and *Bullock\_waggon* are subclasses of *waggon*

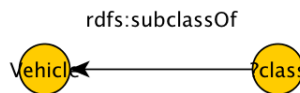


Figure 4.14: Query pattern for getting direct and indirect subclasses of *Vehicle*

```
(?c2 rdfs:subClassOf ?c1) (?c3 rdfs:subClassOf ?c2) -> (?c3 rdfs:subClassOf ?c1)
```

Listing 4.9: Deductive reasoning rules for Figure 4.15

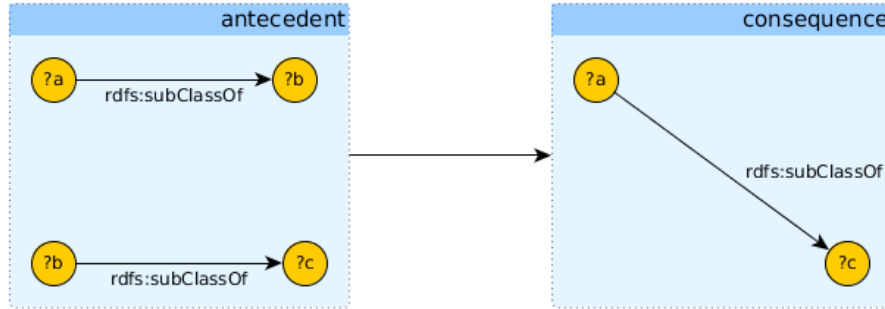


Figure 4.15: Reasoning rules enabling the transitive closure of class hierarchy

Figure 4.16 describes another common generic reasoning rule. This graph-pattern shows that all instances of the subclass are also instances of the superclass – if  $?x$  is a subclass of  $?y$  and  $?a$  is an instance of  $?x$  then  $?a$  is also an instance of  $?y$ . (e.g. *h1* is a *Horse*, *Horses* are subclass of *Animals*. Therefore we can say *h1* is an instance of *Animal*).

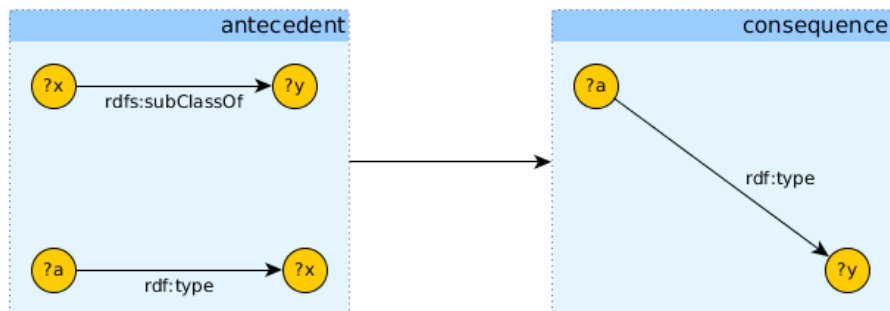


Figure 4.16: Rules for subsumption (subclass) reasoning

```
(?x rdfs:subClassOf ?y) (?a rdf:type ?x) -> (?a rdf:type ?y)
```

Listing 4.10: Deductive reasoning rules corresponding to Figure 4.16

**Transitive property** – Transitivity of OWL property can be applied using reasoning rule as described in Figure 4.16 – assume property  $?p$  is a transitive property, if these two triples exist: (1)  $?a - ?p - ?b$  (2)  $?b - ?p - ?c$  then an implicit triple  $?a - ?p - ?c$  could be deduced from this rule.

For instance, property *isLocatedIn* is transitive, therefore if (1) *taggingArea1* is located in *taggingArea2* and (2) *taggingArea2* is located in the *taggingArea3* then (3) *taggingArea1* is located in the *taggingArea3*.

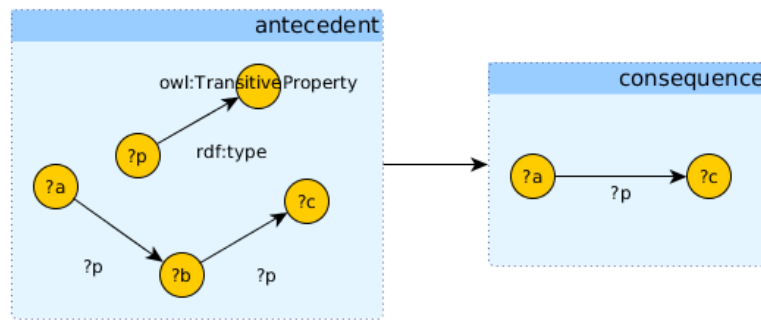


Figure 4.17: Generic reasoning rules for OWL transitive properties

```
(?p rdf:type owl:TransitiveProperty) (?a ?p ?b) (?b ?p ?c) -> (?a ?p ?c)
```

Listing 4.11: Jena reasoning rules that correspond to Figure 4.17

**Symmetric property** - reasoning rule in Figure 4.18 describes the characteristics of symmetric properties. A symmetric property holds in both directions. Suppose  $?p$  is symmetric, if object  $?a$  is linked to  $?b$  via property  $?p$  then  $?b$  is also linked to  $?a$  via  $?p$ . For example, *overlapped\_with* is also a symmetric property: Iron Age overlaps with Roman Period so Roman Period overlaps with Iron age, vice versa. As can be seen here, the domain and range of transitive and symmetric properties are always the same class.

```
(?p rdf:type owl:SymmetricProperty) (?a ?p ?b) -> (?b ?p ?a)
```

Listing 4.12: Jena reasoning rules that corresponds to Figure 4.18



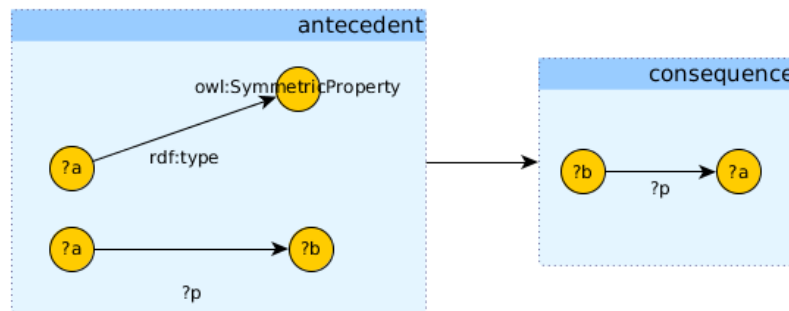


Figure 4.18:

**Subproperty subsumption** – similar to a class, property or predicate can also have a hierarchy. When applying the reasoning rule in Figure 4.20, it should include the transitive closure of sub property relations. The inferencing rule shown in Figure 4.19 takes into account the inferred property hierarchy. For example, *spearing* is a sub property of *hunting* - Hunter *?h1* was *spearing* a wild boar *?b* implies *?h1* was *hunting* *?b* because *hunting* is a more general relationship than *spearing*.

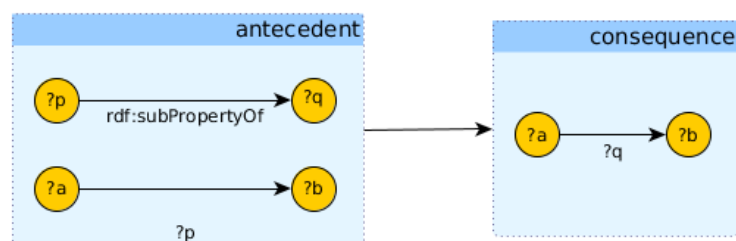


Figure 4.19: Generic reasoning rules for OWL sub-property

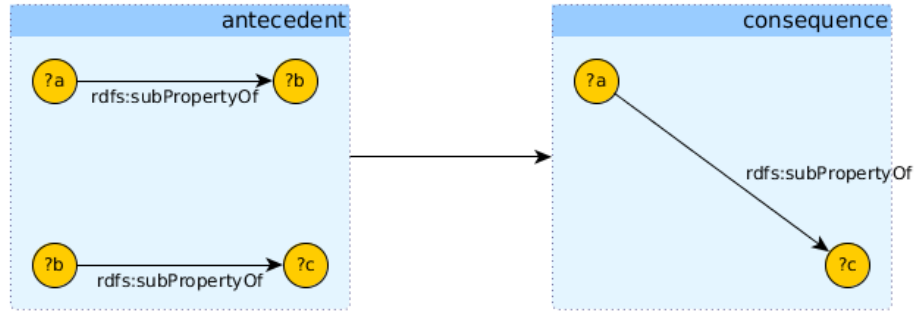


Figure 4.20: Sub-property transitive closure

**Inverse property** – datatype and object property always has a direction. In many cases, property in one direction could have an inverse property in the other direction. For example, property *wasFoundAt* is the inverse of *excavationSiteOf*, if artefact *a* was discovered at location *Loc*, this implies *Loc* was the excavation site where artefact *a* was unearthed.

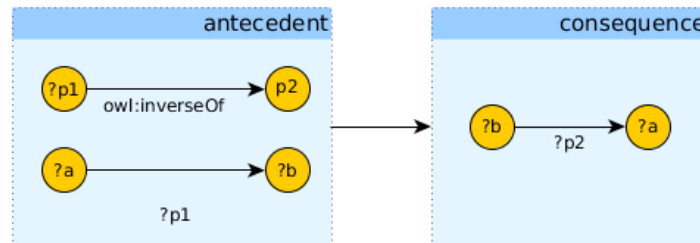


Figure 4.21: Generic inverse property rule

### 4.2.2 Custom Reasoning Rules

In some situations, it is necessary to introduce custom reasoning rules. For example, user-defined deductive reasoning rules can deduce implicit relationship *overlap\_with* based on existing relations (e.g. Object Property *overlap\_with* can be defined using *beginning\_is\_qualified\_by* and *end\_is\_qualified\_by*). For example, in CIDOC-CRM both *IronAge* and *RomanPeriod* are instances of *TimeSpan*, each of them is defined by two properties (1) *beginning is qualified by* (2) *end is qualified by*. Because *IronAge*

starts from 18th century BC and finishes by 1st century AD, the Roman period is between 27BC and 476AD. So Iron age overlaps with Roman period.

**Example:**

```
# reasoning rules to determine if two TimeSpan ranges overlap

(?t1 rdf:type TimeSpan) (?t2 rdf:type TimeSpan)
    (?t1 beginning_is_qualified_by ?t1b)
    (?t1 end_is_qualified_by ?t1e)
    (?t2 beginning_is_qualified_by ?t2b)
    (?t2 end_is_qualified_by ?t2e)
    (?t2b occurs_before ?t1b)
    (?t2e occurs_after ?t1b) -> (?t1 overlap_with ?t2)

(?t1 rdf:type TimeSpan) (?t2 rdf:type TimeSpan)
    (?t1 beginning_is_qualified_by ?t1b)
    (?t1 end_is_qualified_by ?t1e)
    (?t2 beginning_is_qualified_by ?t2b)
    (?t2 end_is_qualified_by ?t2e)
    (?t2b occurs_before ?t1e)
    (?t2e occurs_after ?t1e) -> (?t1 overlap_with ?t2)

(?t1 overlap_with ?t2) -> (?t2 overlap_with ?t1)
```

Listing 4.13: overlap\_with rules

A graph-based reasoning tool has been implemented. More detail about this will be discussed in Chapter 6.

## 4.3 Personalised and Collaborative Search with Multiple Objectives

The level of importance of criteria in the pattern graph are determined by two factors: (1) the relative importance of criteria based on query makers' collective preferences over different triple patterns and (2) the importance of variable nodes in a query derived

from the graph topology.

### 4.3.1 Collective Weights for Triple Patterns

In a traditional pattern graph, the weights of every criterion (triple) in the pattern graph are the same. However, in a collaborative environment, a group of query builders might want to express their preference for the degree of importance for each triple in the pattern. For example, considering the following triple patterns for searching object:

```
line 1: ?object rdf:type Terracotta
line 2: ?object hasSimilarMotif ?object2
line 3: ?object isMadeFrom Clay
```

Listing 4.14: Query pattern with three triple clauses

Members of the groups who construct the query collaboratively might have very different views over which of these triple pattern are more important than others. The basic idea is to build a pairwise comparison matrix for each member to express its personal preference between every two distinct triples in the pattern graph. The relative importance of the each triple in the query can be described using a variant of nine-point intensity scale pairwise comparison matrix. The degree of importance of triples in query patterns given by a member can be modelled as followed in Equation 4.1.

$$\begin{cases} t_1...t_n \in G \\ W(G, u) = \{W_{(t_1, u)}, W_{(t_2, u)}..W_{(t_n, u)}\} \end{cases} \quad (4.1)$$

where  $t_1...t_n$  are RDF triple query patterns consisting a query  $G$ . Every user can express their preference on a triple pattern using the nine-point pairwise comparison matrix in Table 4.1.

Considering the query in Listing 4.14, there are 3 triple patterns.

- $t_1$ : *?object rdf:type Terracotta*

Table 4.1: nine-point intensity scale for pairwise comparison of triple pattern [82]

Preference on triple patterns in query $G$	Score
$t_x$ is equally important as $t_y$	1
$t_x$ is moderately important than $t_y$	3
$t_x$ is strongly more important than $t_y$	5
$t_x$ is very strongly more important than $t_y$	7
$t_x$ is extremely more important than $t_y$	9

•  $t_2$ : *?object hasSimilarMotif ?object2*

•  $t_3$ : *?object isMadeFrom Clay*

User  $u_1$ 's preference is listed as followed:

- $t_1$  is very strongly more important than  $t_2$
- $t_1$  is strongly more important than  $t_3$
- $t_3$  is moderately important than  $t_2$

A pairwise comparison matrix can be constructed as shown in Equation 4.2.

$$\begin{bmatrix} 1 & t_1/t_2 & t_1/t_3 \\ t_2/t_1 & 1 & t_2/t_3 \\ t_3/t_1 & t_3/t_2 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 7 & 5 \\ 1/7 & 1 & 1/3 \\ 1/5 & 3 & 1 \end{bmatrix} \quad (4.2)$$

From equation 4.2, relative weights given by user  $u_1$  can be calculated for each triple pattern  $(t_1, t_2, t_3)$  as shown in equation 4.3

$$\begin{bmatrix} w(t_1, u_1) \\ w(t_2, u_1) \\ w(t_3, u_1) \end{bmatrix} = \begin{bmatrix} (1 \times 7 \times 5)^{1/3} = 3.271 \\ (1/7 \times 1 \times 1/3)^{1/3} = 0.189 \\ (1/5 \times 3 \times 1)^{1/3} = 0.843 \end{bmatrix} = \begin{bmatrix} 3.271 \\ 0.189 \\ 0.843 \end{bmatrix} \quad (4.3)$$

The overall weight for a given triple  $t \in G$  derived from collective preferences given by a set of users  $u \in U$  denoted as  $W_{cp}(t)$ , can be calculated using Equation 4.4:

$$W_{cp}(t) = \sum_{u \in U} w(t, u) \quad (4.4)$$

### 4.3.2 Topology-based Weightings

One can also measure the importance of variable nodes in a query based on triple graph topology. This step is different from calculating and aggregating the weights for each triple pattern as discussed in section 4.3.1. In this step, the degree of importance of variable nodes is computed entirely based on the structure of a query.

To identify which variable nodes are considered to be more important, the underlying assumption is that more important variable nodes are likely to appear in an RDF subject-predicate-object triple as a subject. According to Oxford dictionary [83], a subject is "a noun or noun phrase functioning as one of the main components of a clause, being the element about which the rest of the clause is predicated" and an object is "a noun or noun phrase governed by an active transitive verb or by a preposition". In our context, predicate denotes the relationship between subject and object in a triple statement, while the object acts as the receiver of an action, the subject is considered more prominent than the object [84]. The reason for that is obvious – important nodes are more likely to have been mentioned in other statements. When a node is referred in a statement, it could be used as a subject or an object. Subject is considered to be more important than object so it is likely to have more connection with other nodes. To measure the importance of the variable nodes, we adopted a variation of PageRank algorithm [85] called *Triple Graph PageRanker* to measure the importance of variable nodes in query patterns. The following paragraphs will explain the algorithm with a few examples.

The weight assigned to a given variable node  $a$  is referred as *Triple Graph Rank* for  $a$ , denoted by  $W_r(a)$ . The node rank for each node (regardless of a variable node, named class or named constant) will be given an initial value, the sum of  $W_r$  over all nodes should be 1. For example, given the query in Figure 4.22, the initial value for each node is 0.333. The triple node rank transferred from object to subject is divided equally among all outbound subject nodes. For example, variable  $?b$  appears in 2 triples

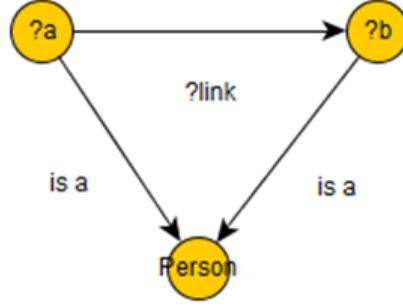


Figure 4.22: Get a list of possible relationship between any two person

$(?a, ?link, ?b)$  and  $(?b, rdf:type, Person)$ , in the latter one it appears as the object therefore  $?b$  would transfer all its existing  $W_r$  value 0.333 to  $?a$ . Similarly,  $Person$  would transfer half of its existing  $W_r$  value 0.1667 to  $a$ .  $W_r(?a)$  can be calculated using Equation 4.6.

$$W_r(?a) = \frac{W_r(?b)}{1} + \frac{W_r(Person)}{2} \quad (4.5)$$

In general, triple node weight for any node  $n$  in a pattern graph can be expressed as:

$$W_r(n) = \sum_{x \in T_O(n)} \frac{W_r(x)}{|T_O(x)|} \quad (4.6)$$

where

- $W_r(n)$  is the triple graph rank for node  $n$ .
- $T_O(x)$  is a set containing all subject-object-predicate triples in which  $x$  appears as the *object*.
- $|T_O(x)|$  is the number of subject-predicate-object triples where  $x$  appears as the *object*.

For example, given the query pattern in Figure 4.22, the triple node weight for variable node  $?a$  is the sum of  $W_r$  over all nodes involved in an RDF triple where  $?a$

is the subject and  $m$  is the *object*, divided by the number of triples where  $m$  appears as the *object*.

## 4.4 Ranking

As discussed in Chapter 3.1.2 and Chapter 3.1.3, the uncertainty of the assumptions and annotators' domain-specific reputation can be captured and represented in our model. How annotators can express their preferences over the relative importance of each criterion in collaborative search and how weights can be measured from graph pattern were explained in Chapter 4.3.1 and Chapter 4.3.2 respectively. These factors need to be combined and aggregated for ranking.

Once the search results are returned, they are ranked by a combination of the degree of truthfulness and user preferences. Weight is calculated for every sub graph in the result set that matched the pattern. The function defines the overall weight  $W_G$  of the subgraph  $G$ :

$$W_G = \prod_{t \in T} CF_t \times W_{cp}(t) \times (W_r(s_t) + W_r(o_t)) \quad (4.7)$$

where

- $G$  is a sub graph that matched the search pattern.
- $T$  is a set of triples in  $G$ .

and

- $CF_s$  is the overall composite certainty-credibility factor of triple  $t$ .
- $W_{cp}(t)$  is the collective weights for triple  $t$  in  $G$ .
- $s_t$  and  $o_t$  are respective *subject* and *object* of RDF triple  $t$ .
- $W_r(s_t)$  and  $W_r(o_t)$  are topology-based weights for  $s_t$  and  $o_t$

$W_G$  will be used later in the ORDER BY clause of the query implementation providing a result set with the most trusted results at the top.



# Collaborative Categorisation

---

This chapter will discuss different categorisation methods for distributed team members to classify items. Firstly, it will investigate the feasibility of adopting a variant of the clustering algorithm for sorting annotated objects. As distance measure plays an important role in clustering, we examine the possibility of using the semantic relatedness of the tags derived from a lexical database for distance measurement. Then a formal way is introduced to help collaborative teams build a taxonomy or categorisation system collectively. Manually classification results are used to train and build a modified Naive-Bayes classifier. In the end, the classifier will be used for sorting the remaining uncategorised objects.

## 5.1 Categorisation Through Social Tagging and Clustering

If a universally accepted taxonomy is not available and a team cannot agree on a hierarchically categorised system then automated tag clustering will be used to group tags. Assuming there are a set of photos, each of them is annotated with a few keywords in various topics. The next step we want to do is to generate category and filters to help users browse objects and narrow down the search results using faceted search.

In a faceted classification system, the category is not a single, predefined taxonomy, but a multiple-layer, characteristic-based classification approach. For example, a group of artefact's can be classified using a chronology facet, a material facet, a provenance facet, etc. The challenge is how to determinate which facet a keyword might belong to.

### 5.1.1 Calculating Semantic Relatedness Using Lexical database

The first step of grouping items is to calculate a pairwise similarity matrix (overall degree of similarity of any two tags). In our case, we tried to apply the similarity measurement matrix in [86] (Leacock and Chodorow's normalised path length) to calculate the pairwise-similarity between keywords in the annotation with helps from WordNet dictionary [66]. Wordnet dictionary groups English words into sets of synonyms called synsets [87]. According to [86] "the relatedness of two words is equal to that of the most-related pair of concepts that they denote". Assume we have two keywords  $w_i$  and  $w_j$ , we devoted  $s(w)$  a collection of senses of word  $w$ , the similarity between words  $w_1$  and  $w_2$  can be defined in equation 5.1.

$$similarity(w_i, w_j) = -\log \frac{len(c_i, c_j)}{2 \times MaxDepth(C)} \quad (5.1)$$

where  $c_i \in s(w_i)$ ,  $c_j \in s(w_j)$ ,  $MaxDepth(C)$  is the maximum depth of the WordNet hierarchy;  $len(c_i, c_j)$  is the length of the shortest path between  $c_i$  and  $c_j$  in the WordNet hierarchy. Here we can then transform similarities measures matrix  $S$  into a dissimilarities matrix, to build a distance matrix  $D$  by using standard transformation method:

$$d_{i,j} = \sqrt{s_{ii} + s_{jj} - 2 \times s_{ij}} \quad (5.2)$$

### 5.1.2 Grouping with Unsupervised Clustering

To group the keywords belonging to the same facet, we applied a variation of DBSCAN clustering algorithm [88] (Density-based Spatial Clustering of Applications with Noise). The original DBSCAN algorithm is based on the principle of density reachability. We can say point  $A$  is *directly density-reachable* from  $B$ , if (1)  $B$  is further than a semantic-distance  $r$  from  $A$ , and (2)  $A$  is surrounded by sufficient points  $MinPts$  within a specified radius  $r$ . *Density-reachability* is the transitive closure of the *direct density reachability*. Density-reachability is transitive but not symmetric. If  $A$  and  $B$  are

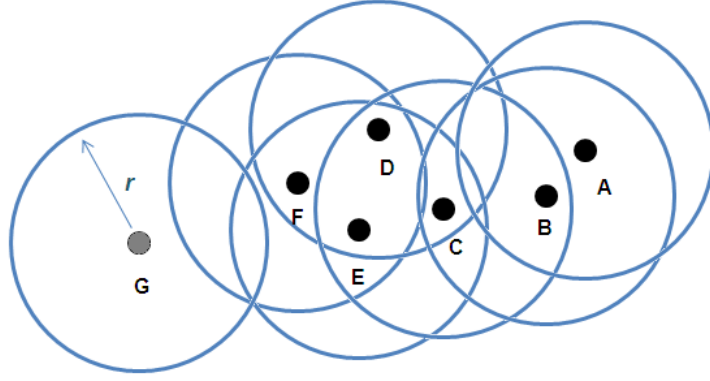


Figure 5.1: (Point  $ABCDEF$  are density-connected and belong to the same cluster while point  $G$  is a noise point. (MinPts=3))

both *density-reachable* from the same point  $O$  then  $A$  and  $B$  are *density-connected* and they should belong to the same cluster. For example, in Figure 5.1, Point  $A$  and  $F$  are density-connected because they are both density-reachable from the same point  $C$ , thus  $A$  and  $F$  belong to the same cluster. Point  $G$  is considered as an outsider and will be marked as noise because it is not density-reachable from any nodes in the cluster. (MinPts=3)

The modified DBSCAN clustering algorithm is based on the same principle. The DBSCAN algorithm is originally designed for the task of detecting clusters in a spatial database. Although we cannot reproduce a 2-dimensional distribution of points merely based on the similarity matrix, the input data necessary to apply DBSCAN algorithm are the distances between any given 2 points in the spatial distribution. In our case, we have given pairwise distances (pairwise dissimilarity) instead of the actual data points. Therefore, the algorithm can be adapted to solve the problem of clustering. The basic idea is to replace DBSCAN's distance measure function between any two points  $dist(a, b)$  within  $regionQuery(P, eps)$  to our dissimilarity function  $s(e_i, e_i)$  by the dissimilarity measure matrix obtained in Equation 5.2.

In contrast to original *Eps-neighbourhood* explained in [89], our modified DBSCAN algorithm uses  $min-dissimilarity(P, s)$ , to obtain a set of items that are associated with  $P$  within a minimum degree of similarity  $s$ , denoted as  $(N_s(p) = \{q \in$

$E[\text{disimilarity}(p, q) > s\}.$ ) We present the pseudo-code of our modified DBSCAN algorithm for identifying clusters of a set of unstructured keywords, omitting the details that are not essential for understanding the procedure:

```

C:Cluster
N:ItemSet

PROCEDURE Modified_DBSCAN (E, s_threshold, min_objects){
  C = NULL
  FOR EACH P in set E {
    IF (P.visit=FALSE) THEN
      P.visit=TRUE;
      N = RegionQuery(P, s_threshold)
      IF (N.size < min_objects) THEN P.state= UNRECOGNISED_GROUP
    ELSE
      C = next Cluster
      EXPAND_CLUSTER(P,N,C, s_threshold, min_objects)
    END IF
  END IF
END FOR

PROCEDURE ExpandCluster(P,N,C,s_threshold,min_objects)
  C.add(P)
  FOR EACH PX in N THEN
    IF (PX.visit=FALSE ) THEN
      PX.visit=TRUE
      NX=  RegionQuery(PX, s_threshold)
      IF (NX.size>= min_objects) THEN
        N = N joined with NX
      END IF
    END IF
    IF (PX.state= NULL) THEN C.add(PX)
  END FOR

```

Listing 5.1: Modified DBSCAN algorithm

A function call to `regionQuery(P, s_threshold)` will return a set of sur-

rounding items that are similar to  $P$  with a certain degree ( $s\_threshold$ ). Keywords labelled as noise do not belong to any clusters at the end of the process will be marked as “unclassified” words. An interesting question arising is how to determine the optimal values for two parameters:  $s\_threshold$  (similarity threshold) and  $min\_objects$  (the minimum number of objects required to form a cluster).

### 5.1.3 Faceted Browsing

**An example of facet browsing with WordNet keyword clustering:** Faceted browsing allows objects to be categorised using multiple facets. In Figure 5.2, suppose various objects of different shapes, colours and borders are annotated by a group of people. The first object has been tagged "dotted line", "triangle" and “yellow” – Instead of classifying objects according to a single taxonomy, these tags describe different attributes (or facets). The basic idea is to use modified DBSCAN algorithm to put keywords into clusters according to lexical semantic relatedness derived from WordNet dictionary and use the clustering for the facet classification:

- Step 1: calculates pairwise distances between different tags based on lexical semantic relatedness in WordNet dictionary according to Equation 5.1 and Equation 5.2. For example, according to the WordNet hypernym hierarchy of nouns, the lexical semantic distance (or relatedness) between tags "yellow" and "red" is shorter than the distance between tag "yellow" and "circle".
- Step 2: apply our modified DBSCAN algorithm to put keywords into clusters. Based on lexical semantic distances relatedness, tags in Figure 5.2 are classified into three groups as shown in Figure 5.3, For example, "yellow" and "red" belong to the same cluster.
- Step 3: use facets to filter the results as shown in Figure 5.4.

However, our initial attempts to create a similarity matrix for clustering by measuring the pairwise distances between tags using lexical semantic relatedness [19] failed to get the desired hierarchical clusters archaeologists wanted. Despite the fact that the automatic classification technique based on unsupervised learning does not generate

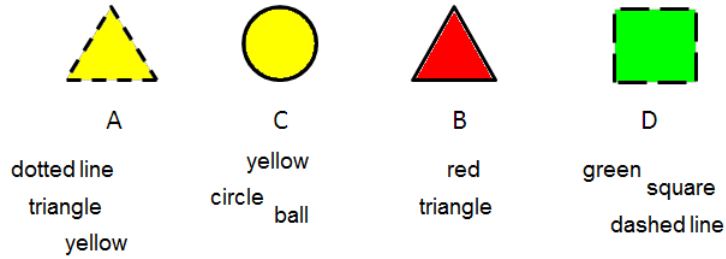


Figure 5.2: Example of keywords attached to different symbols

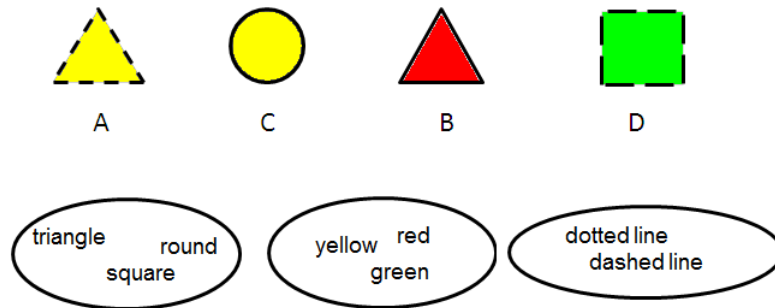


Figure 5.3: Clustering keywords into different facets – shape, colour and border

results we expected in the archaeological domain, it is worth considering this technique and its potential applications in other domains. Further research and possible future development will be discussed in section 7.2.

## 5.2 Categorisation Through Supervised Learning

Chapter 5.1 discussed how semantic relatedness can be used to categorise objects through clustering. This method has successfully categorised the sample data sets, however, in some experiments clustering seems to not working very well with real data set in certain domains as the algorithm often led to natural clusters that are undesirable for our categorisation task.

In this section, an alternative way to classify objects through supervised learning will be discussed. The section will start with some basic notation, terminology and

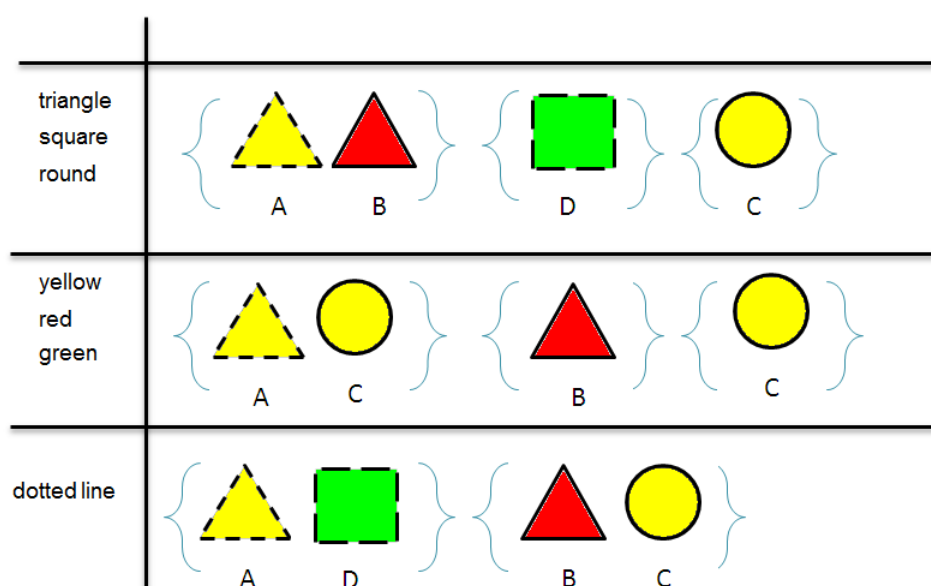


Figure 5.4: Faceted browsing

models used in multiple-criteria collaborative categorisation process and further discuss the procedure to create categories collectively as a team. In the end, how an individual's categorisation result and decision-making process can be recorded and addressed (e.g. what criteria were considered and how certain was the decision) and how to use manually annotate data to train a customised classifier.

### 5.2.1 Fuzzy Multiple-criteria Collaborative Categorisation (FM-CCC)

Collaborative Categorisation is defined as the process where a group of people classify objects and attempt to reach a collective decision according to their own categorisation system, attitude and interpretation of evidence. In most real world contexts, the situation is very complex, which brings many challenging questions such as selection and aggregation of multi-criteria weights and the relative importance or fuzziness in the categorisation criteria. In many cases, classification requires multiple perspectives as one single person might not have sufficient knowledge to make that decision alone. Hence, categorisation in a collaborative working environment is even more complex.

### 5.2.1.1 MCCC Conceptual Models

This section will discuss different MCCC (Multiple-Criteria Collaborative Classification) Conceptual Models that can be used to address the problem discussed above. Similar to group decision making-process discussed in [82], there are two different types of model for collaborative classification.

**Rational Model** - the Rational Model is grounded on evidence and consequences. This model assumes that most information regarding the classification task is available. In this case, members of a team have to reach agreement on objectives, a set of destination classes is known. Every decision-maker needs to evaluate the probability of an object being classified into every possible candidate category with goals and objectives in mind. Then the optimal choice with the highest probability will be selected. This model utilises a logical approach that makes the classification decisions by evaluating possible destination classes based on available information.

**Political Model** - In contrast to the Rational Model, information in the political model is usually incomplete. Members might not share common objectives. Each member of the team is motivated by their personal perception and interpretation. This process involves negotiation and discussion among group members. In this case, conflict of interest usually leads to debates, every decision-maker in the group tries to persuade others to adopt their viewpoints, in an attempt to influence the rest of the members.

Both the **Rational Model** and **Political Model** have their advantages and disadvantages. It is not sufficient to use the rational model alone because information regarding categorisation are usually incomplete, uncertainty and conflicts of interests remain. To reach an optimal classification result the process needs to go through negotiation and discussion. However, like in a General Election, the political model cannot guarantee to find an optimal solution. Therefore, in practice, a combination of both models could help achieve better performance.



### 5.2.1.2 MCC Mathematical Model

Mathematically, a typical Multi-Criteria Categorisation (MCC) problem can be modelled as followed in Equation 5.3:

$$MCC = \begin{cases} select : C_1, C_2 \dots C_n \\ s.t. : CR_1, CR_2 \dots CR_m \end{cases} \quad (5.3)$$

where  $C = (C_1, C_2 \dots C_n)$  denotes  $n$  classes,  $CR = (CR_1, CR_2 \dots CR_m)$  represents  $m$  criteria. The *select* operation here is normally based on maximising a criteria aggregation function. To achieve this, we have to compute the contribution of each attribute to the overall goal - an optimal classification decision. In a collaborative environment, we will need to take into account other factors, for example, each member's opinion regarding the importance degree of criteria, degree of uncertainty and the reputation of the each decision-marker.

### 5.2.1.3 MCCC Process

In general, Multi-Criteria Collaborative Categorisation (MCCC) consists of these five steps

- Examining items to be categorised
- Generate category tree
- Determine criteria
- Evaluate destination categories
- Rank aggregation

### 5.2.1.4 Proposed Approach

The proposed computer-assisted framework is aiming at helping a collaborative team in classifying a large volume of objects. First of all, the team will build a universally

accepted taxonomy and categorisation system collectively. Then we obtain a small number of manually categorised objects from team members through annotation and then the framework utilises supervised learning technique to train a classifier. Figure 5.6 and 5.5 illustrates the process of computer-assisted collaborative classification as supported by our framework.

The process consists of the following steps:

**Raw Data entry** – a group of people observe and describe the objects using a semantic annotation model. We populate the database with observed data, including the degrees of fuzziness in the description.

**Select training and evaluation dataset** – we divide un-categorised objects into two parts - a few will become training dataset while the rest of them can be used as evaluation dataset.

**Generate hierarchical categories** – a category structure as one or several trees will be constructed collectively by team members.

**Determine criteria** – Each decision-maker must indicate what attributes were relevant when sorting objects in the training data sets and may have led to the conclusion for each one.

**Manually classifies training samples** – Ask a group of people to categorise training data sets based on their perspectives and interpretation of supporting evidence.

**Building classifier** – Use the result of manually categorised objects to build a classifier.

**Applying classifier** – Apply the classifier to classify remaining un-categorised objects.

We will discuss the first four steps in this section and dedicate the subsequent sections to step 5 and 6 respectively.

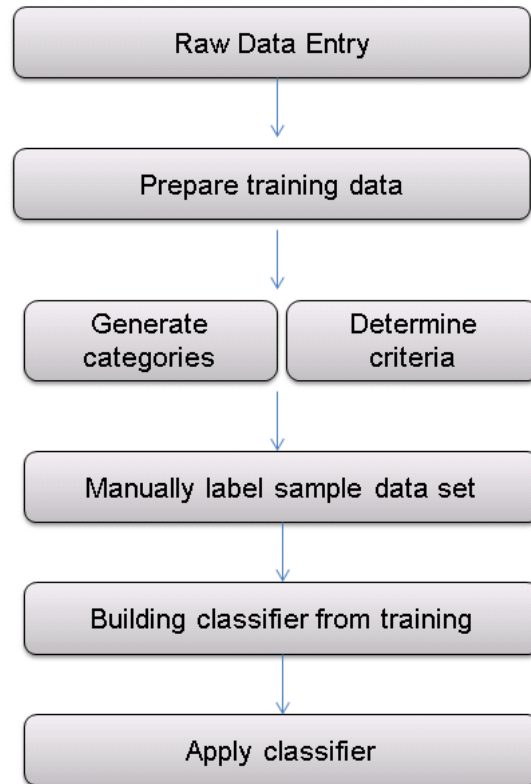


Figure 5.5: Process of Computer-assisted classification

### 5.2.2 Collaborative Classification Ontology

A good data model is required to obtain a good annotation. For our uses, we introduce a collaborative classification ontology. Consider the following statement regarding Figure 5.7.

“The human figure in this drawing could be a female — this conclusion was based on the figure’s posture and surrounding objects (e.g. the pottery vessel she is holding)”

– by *an archaeologist specialised in ancient human representation*

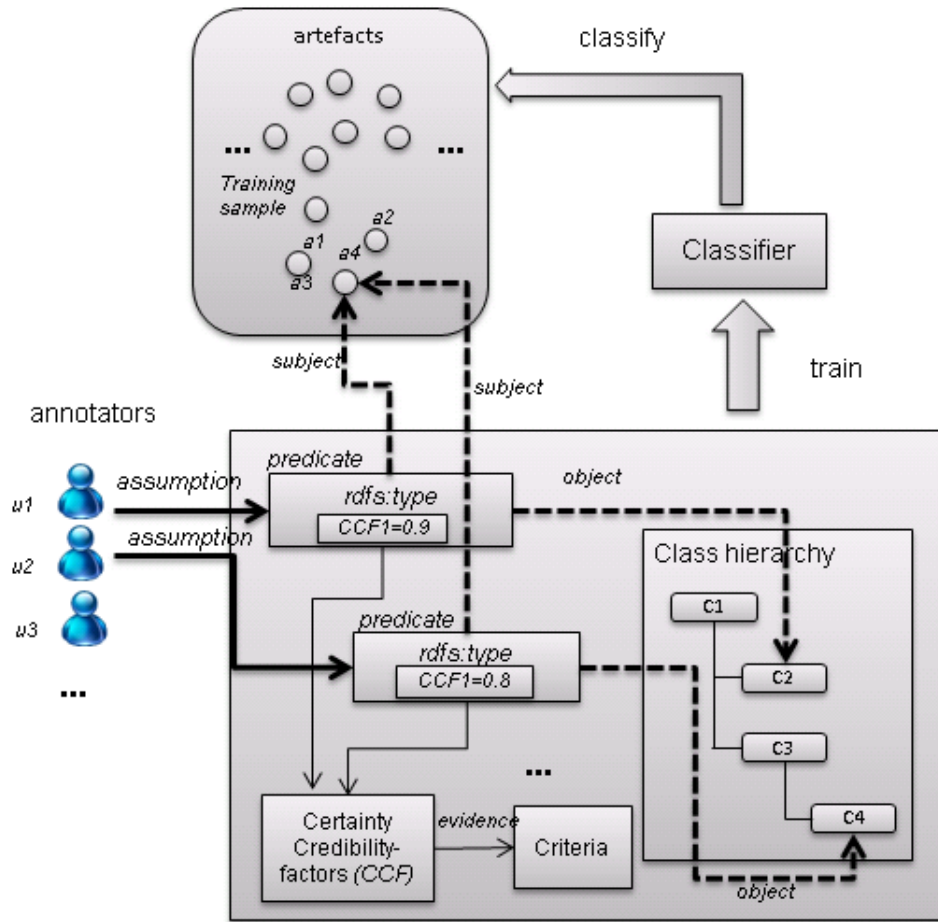


Figure 5.6: Workflow of computer-assisted collaborative categorization

This statement forms one of the typical information that we need to record in a structured way. We developed the *Collaborative Classification Ontology (CCOnt)* in Figure 5.8 as an abstract data model for our classification framework. CCOnt is built on the basis of a data model used for image tagging as described in Chapter 3, it serves as the meta-model for the system and provides an infrastructure for storing annotations from users.

The class *Assertion* is the core element of *CCOnt*. *Assertion* refers to a statement claimed by someone with some degrees of certainty. Theoretically, this model is capable of describing any “instance of” assumptions and other arbitrary statements through RDF triples [51] — a triple is usually written in the order *subject*, *predicate*, *object*.



Figure 5.7: Ancient human representation

For example, the statement “Object  $a$  is an instance of class  $A$ ” can be represented as an RDF triple:  $a \text{ --- } \textit{rdf:type} \text{ --- } A$ , where  $a$  is the *subject*, RDF property *rdf:type* (“instance-of”) is the predicate and  $A$  is the *object* of the triple. Every *Assertion* is linked to a certainty-credibility factor [1] model (*CCFModel*) which records (1) The person who made this “instance-of” assumption. (2) The person’s degree of uncertainty reflecting one’s own judgement on how truthful and reliable the statement is. (3) A timestamp recording when the statement was made; and (4) A domain-specific credit-rating of the contributor at the moment when this statement was last updated, bearing in mind that as time passes a user’s reputation might change but this does not modify the truthfulness and reliability of statements previously made.

#### 5.2.2.1 Raw Data Entry

The first step is to get the raw data (attribute values regarding unlabelled objects) into the ontological database. We can either transfer data from existing relational data set via the transformation method discussed in section 3.2 or enter manually by a group of persons or manually entered by a group of persons by hand. When entering raw data into the system, the fuzziness of data concerning an object is also represented.

#### 5.2.2.2 Selection of Training Dataset

The next step is to select a subset of uncategorised objects. In other words, annotators will categorise a small amount of randomly selected items based on their own understanding and interpretation. The result of classification will be used for building an

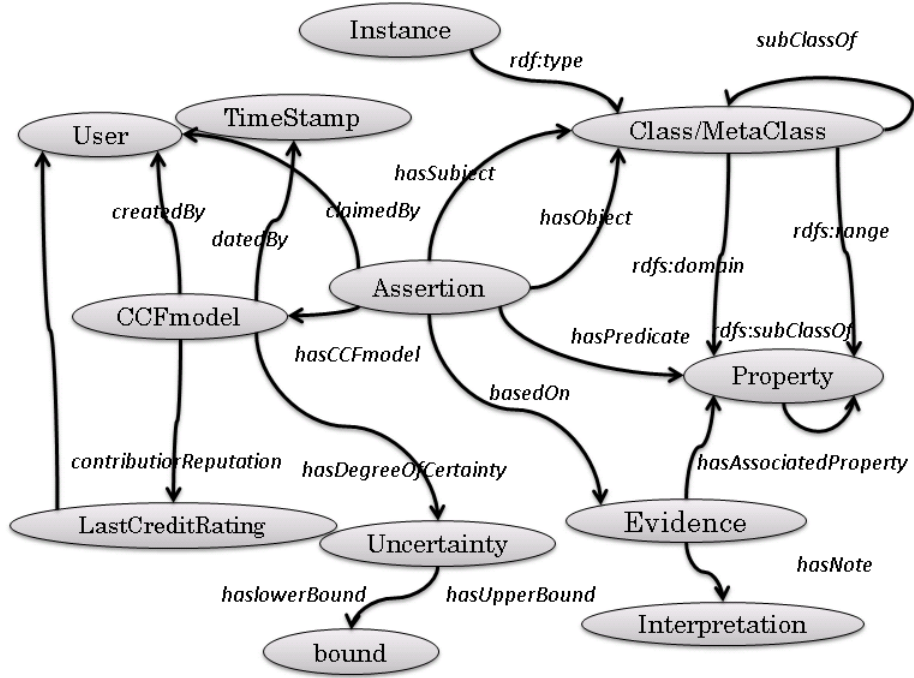


Figure 5.8: Collaborative Classification Ontology (CCOnt)

automatic classifier.

Several aspects would affect the quality of a representative training set: training samples must be sufficiently random and large enough to achieve good generalization capacity. The right size depends on the complexity of the deliberate action. It is critical to have sufficient annotated instances before proceeding to the next step. Random sampling method is used to select representative instances to avoid starting with a particular batch of instances.

### 5.2.2.3 Categories Generation Process

One of the most important steps in collaborative classification is to build a taxonomy that is accepted by the group and how to coordinate participants.

The process requires group members to use a collaborative ontology editor such as *Collaborative Protégé* [90] to co-edit a tree-style hierarchical categorization system. As discussed in section 5.2.2, our framework utilises the class hierarchy of the OWL

ontology as categories.

We can manipulate the class hierarchy by performing these actions:

- Create new sub-category and indicate its parent node(s)
- Remove existing category
- Update parent node(s) of given sub-category

A clear and effective group decision-making process is needed to help a group to design and construct a category collectively. There are several kinds of methods for a group to establish a class hierarchy, here are some of the most popular methods:

### **Authority**

Under the authority rule, a nominated team leader has the authority to make the ultimate decision for the group about whether or not to create/remove a sub-category. Members generate ideas for new categories through discussion, but only one person is responsible for making the final decision. This method can usually produce a final decision very quickly. However, the effectiveness of this methods entirely depends on the ability of the leader. This method does not exploit the advantages of collective intelligence, and nether does it exploit the strengths of every individual in the team.

### **Majority Voting System (MVS)**

The majority's opinion will be chosen as the final solution. Classification decisions are made based on a vote. Such method could reach a group decision soon following a clear rule, and everyone is given a fair chance to participate. Each step for constructing the category is chosen from a list of proposed actions as defined in 5.2.2.3 by a simple majority system in which each person casts one vote. The action with the most votes will be performed. However, the disadvantage of this method is that there are a large number of wasted votes, team members' votes for any options other than the one being selected will have no impact on the final result.

For example, suppose we have a proposed actions “Adding  $x$  as a sub-category to category  $y$ ” suggested by a member. A simple majority system would have two options “approve” or “denied”. Whichever option has the most votes will be the final decision.

### Nominal group technique (NGT)

Nominal group technique (NGT) is a ranking-based method for group decision-making. In contrast to simple majority system, where only the largest group is considered, there are no wasted votes when using NGT. Everyone’s opinions are considered. NGT lets every member of the group give a score to each option for ranking. The votes are calculated, and the one with the highest total score will be selected.

For example, suppose we have three decision-makers  $p_1, p_2, p_3$  and 4 mutually exclusive proposed actions  $A_1, A_2, A_3, A_4$ . Each decision-maker ranks these options by giving a number denoting the rank of this option, the most desirable option is assigned the highest number. When rating the operations, each decision-maker needs to distribute a set of points across the options (in this case, 100). More detail about how to allocate points (weights) to each option using AHP (Analytic Hierarchy Process) [91] will be discussed in section 5.2.3. Table 5.1 shows an example of a solution in NGT – in this case, according to the total score, the preference will be  $A_2 > A_3 > A_1 > A_4$ .

Table 5.1: Group decision-making for category generation in NGT

	$p_1$	$p_2$	$p_3$	Total
$A_1$	10		25	35
$A_2$	50	100	25	175
$A_3$	40		25	65
$A_4$			25	25

### The Proposed Approach for Category Generation

To achieve better performance, our approach adopts a combination of MVS and NGT strategy. The category generation process consists of the following the steps:



Table 5.2: Possible aspects for categorisation of ancient artefacts

	$p_1$	$p_2$	$p_3$	Total
<i>Material</i>	70	90	50	210
<i>Chronology</i>	20	10	20	60
<i>Use</i>	10		10	20
<i>Origin</i>			15	15
<i>Colour</i>			5	5

- Step 1: Collect a set of potential attributes for categorisation suggested by group members.
- Step 2: Calculate a ranked list of proposed attributes using NGT.
- Step 3: Ask members to propose new subcategories for the current node in the category concerning this attribute.
- Step 4: Determine whether new subcategory will be adopted or not using simple MVS.
- Step 5: For each subcategory, repeat steps 2 - 5 until no more subcategory can be found.

For example, suppose we only have a top-level root category called “Ancient artefact”. If there are four members, a ranked list (as shown in Table 5.2) for the possible attributes suggested by team members can be obtained through NGT. In this case, *Material* is considered as the most important aspect.

In this case, *Material* is considered as the most important aspect. Then we accepted this aspect and then asked members to propose possible subcategory with respect to *Material*. Each proposed subcategory for the (e.g. Bronze object) will be determined by polling result using majority voting system – there will be a simple “**approve** or **denied**” question for every option. For example, among the proposed subcategories below,  $A_1$ ,  $A_2$ ,  $A_3$  are likely to be approved.

- $A_1$  adding subclass *Bronze object*?
- $A_2$  adding subclass *Ironwork*?
- $A_3$  adding subclass *Clay-based object*?
- $A_4$  adding subclass *Plastic object*?

Similarly, we could ask members to propose a category regard to *Chronology*.

The proposed ontology-based categorisation framework allows for the crucial feature that allows an object to associate with multiple classes. For example, a person might be classified as man by gender or youngest by age group.

As discussed earlier, the advantages of having some expressive ontologies for modelling categories is their ability to represent complex class and property hierarchy. The ontology-based data model also supports multiple inheritance, which is very useful in multi-dimensional classification - the model also makes it possible to create a single instance of multiple classes.

### Category subsumption reasoning

Because the categorization system modelled in CCOnt is hierarchical, we will need to consider indirect RDFS class-subclass subsumption relations when recording user inputs — assume object  $o$  has been labelled as an instance of class  $C_1$  with a degree of uncertainty  $CF$  by a user. The same assumption also applies to the transitive closure of its superclasses  $C_2...C_n$ , with same or a higher degree of certainty. For example, if one believes that  $c$  is a horse then it implies  $c$  is an animal because *Horse* is a subclass of *Person*. The system generates inferred statements derived from an original statement and assigns the lower-bound to CF uncertainty. The procedures for generating implicit statements and corresponding degree of uncertainty are described in following the SWRL-style [81] deductive reasoning rules:

```
[SubClassTransitiveRule:
  (?c1 owl:subClassOf ?c2) (?c2 owl:subClassOf ?c3) ->
  (?c1 owl:subClassOf ?c3)]
```

```

//transitive rules
[InferredStatementRule:
(?c1 owl:subClassOf ?c2) (?t rdf:type Assertion)
(?u rdf:type User) (?t claimedBy ?u) (?t hasSubject o)
(?t hasPredicate rdf:type) (?t hasObject ?c1)
(?t hasCF ?cf) ->
  (?t2 rdf:type Assertion) (?t2 claimedBy ?u) (?t2 hasSubject o)
  (?t2 hasPredicate rdf:type) (?t2 hasObject ?c2)
  (?t2 hasCF_lowerbond ?cf)]
//inferred statement and uncertainty

```

Reasoning rules are written in the form of “antecedents  $\rightarrow$  consequent”. The first rule *SubClassTransitiveRule* defines the transitive closure of the subclass relation while the latter one *InferredStatementRule* assigns the lower-bound to uncertainty *CF*.

#### 5.2.2.4 Manually Classify Sample Data Sets

The manual classification results are subject to individual’s understanding and interpretation. The decisions usually grounded in the evidence based on a combination of several factors. Although it is not always straightforward to see these criteria, we can identify a list of possible attributes that are relevant to the assessment. Therefore when collecting “instance-of” assertion. It is important to keep records of attribute groups in relation to the criteria applied. Attributes can be numeric or nominal. For example, to identify whether an image is a spider (arachnids), person *p1* classified the image as a spider based on the fact that it has eight legs but has no wings, person *p2* also classified it as a spider but based on the spider web shown next to it. We may deduce these properties such as *hasNumberOfLegs*, *hasWing* are more likely to be relevant in the first case, while in the latter case, the property *canSpinWeb* has an effect on the user’s decision-making process. During the annotation process, annotators should provide the following information:

- (1) “instance-of” statement with respect to the target class. Given an object *o* and a set of categories  $C = \{C_1 \dots C_n\}$  based on the method in section

5.2.2.3, an annotator should map  $o$  to a subset of  $C$ , but any two  $C_x$  in the subset must not fall within the same root-to-leaf path.

- **(2) associated degree of uncertainty.** For every statement in (1), provide a certainty factor  $cF$  ranging from 0 to 1, which indicates the confidence in the certainty of such statement.
- **(3) relevant attributes with respect to the criteria.** Given an object  $o$  with a set of attributes  $A = \{V_1 \dots V_n\}$ , an annotator should indicate a sub set  $A'$  ( $A' \subset A$ ), where  $A'$  are relevant attribute values that led to the conclusion in (1).

To allow non-technical end-users to select classification criteria and attributes in an intuitive way, the graphical query we proposed can also be used to select relevant attributes graphically.

### 5.2.3 Criteria Weighting for Single Decision-maker

In section 5.2.2.3, we discussed the process to obtain a ranked list of criteria from a group of people. Given a particular question, one single person's classification decision is also based on multiple criteria.

It is important to learn what led to the final decision to build an intelligent classifier by learning manual classification result (how various criteria are weighted by every individual). In this context, a variation of Analytic Hierarchy Process (AHP) is used to aggregate multiple classification criteria. For one classification decision-maker, a pairwise comparison matrix is used to define the relative importance of attributes (associated with a given criteria). Here we adopt the "nine-point intensity scale" introduced in [92] as shown in Table 5.3 to express the degree of preference between classification criteria, which is very similar to what people used in surveys.

In the context of a real-world scenario, a group of archaeologists unearthed an unknown clay-based disc object from an excavation site. Assume the following information regarding this uncategorised object are available including laboratory analysis results and other measurements:

Table 5.3: nine-point intensity scale for pairwise comparison [82]

Preference on pairwise comparison of attributes	Score
attribute $A_x$ is equally important as $A_y$	1
attribute $A_x$ is moderately important than $A_y$	3
attribute $A_x$ is strongly more important than $A_y$	5
attribute $A_x$ is very strongly more important than $A_y$	7
attribute $A_x$ is extremely more important than $A_y$	9

- $A_1$ : Shape
- $A_2$ : Decorative motif (e.g. cross, stamps)
- $A_3$ : Microscopic analysis of rock and mineral inclusion
- $A_4$ : Organic residue
- $A_5$ : Weight (g)
- $A_6$ : Thickness (cm)

The group intends to put this object into a correct category based on what it could have been used for (e.g. as cookware, weaving tool or hunting weapon). A pairwise matrix could be used to express relative importance of attributes among  $A_1 \dots A_6$  as shown in matrix 5.4

$$\begin{bmatrix}
 1 & A_1/A_2 & A_1/A_3 & A_1/A_4 & A_1/A_5 & A_1/A_6 \\
 A_2/A_1 & 1 & A_2/A_3 & A_2/A_4 & A_2/A_5 & A_2/A_6 \\
 A_3/A_1 & A_3/A_2 & 1 & A_3/A_4 & A_3/A_5 & A_3/A_6 \\
 A_4/A_1 & A_4/A_2 & A_4/A_3 & 1 & A_4/A_5 & A_4/A_6 \\
 A_5/A_1 & A_5/A_2 & A_5/A_3 & A_5/A_4 & 1 & A_5/A_6 \\
 A_6/A_1 & A_6/A_2 & A_6/A_3 & A_6/A_4 & A_6/A_5 & 1
 \end{bmatrix} = \begin{bmatrix}
 1 & 3 & 5 & 9 & 5 & 7 \\
 1/3 & 1 & 7 & 7 & 3 & 5 \\
 1/5 & 1/7 & 1 & 1/3 & 1/5 & 1/5 \\
 1/9 & 1/7 & 1/3 & 1 & 1/7 & 3 \\
 1/5 & 1/3 & 3 & 1/3 & 1 & 5 \\
 1/7 & 1/5 & 5 & 1/3 & 1/5 & 1
 \end{bmatrix} \quad (5.4)$$

Suppose one member put this object under category *Weaving tool*. Apart from the conclusion, this person should also provide a pairwise comparison matrix as shown in

5.4 to indicate the importance of different attributes concerning this decision according to their perception.

$$\begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \\ A_5 \\ A_6 \end{bmatrix} = \begin{bmatrix} (1 \times 3 \times 5 \times 9 \times 5 \times 7)^{1/6} = 4.10 \\ (1/3 \times 1 \times 7 \times 7 \times 3 \times 5)^{1/6} = 2.50 \\ (1/5 \times 1/7 \times 1 \times 1/3 \times 1/5 \times 1/5)^{1/6} = 0.27 \\ (1/9 \times 1/7 \times 1/3 \times 1 \times 1/7 \times 3)^{1/6} = 0.36 \\ (1/5 \times 1/3 \times 3 \times 1/3 \times 1 \times 5)^{1/6} = 1.24 \\ (1/7 \times 1/5 \times 5 \times 1/3 \times 1/5 \times 1)^{1/6} = 0.46 \end{bmatrix} = \begin{bmatrix} 0.45 \\ 0.28 \\ 0.03 \\ 0.04 \\ 0.13 \\ 0.05 \end{bmatrix} \quad (5.5)$$

Therefore, we can see in this case, *Shape* and *Decorative motif* are considered by this member as the most important attributes in respect to this classification decision.

### 5.2.3.1 Combining Uncertainties from Multiple Decision-makers

When individual made the assumption about an object's category, there is always some uncertainty involved. As discussed earlier in Chapter 3.1.2, certainty factor can be used to represent the degree of uncertainty regarding an assumption. A certainty factor (CF)[67] is a number (-1.0 1.0) that indicates how certain the person was when making an assumption regarding what categories an object might belong to. Positive certainty means the user generally agrees that the assumption is true according to his/her understanding. In contrast, negative certainty is used to express disagreement. Our Collaborative Classification Ontology (CCOnt) extends triple graph to include certainty factor in a collaborative environment so that the uncertainties of the classification decision is captured in the model.

In a multi-user collaborative environment, a mechanism is needed to combine multiple members' uncertainties. If more than one user express their uncertainty over the same assumption, a parallel function (5.6) can be used to compute the overall-uncertainty.

Suppose two members  $p_1, p_2$  make the same assumption  $A$  over object  $O$ 's category  $C$ , but with different degree of uncertainty  $(CF_1, CF_2)$ . The situation can be described using uncertain RDF triples.

A Likert-type scale as shown in Table 5.2.3.1 is used to help the decision-maker to

Table 5.4: Linguistic terms and related certainty factor for describing uncertainty

Degree of uncertainty	certainty factor (CF)
strongly agree	0.9
mostly agree	0.6
slightly agree	0.3
no opinion	0
slightly disagree	-0.3
mostly disagree	-0.6
Strongly disagree	-0.9

express their degree of uncertainty in respect to the assumption. Intermediate values (e.g. 0.5, 0.45) can be used to represent the compromises between different degrees.

$A: O \text{ rdf:type } C$  ( $O$  is an instance of  $C$ )

$P_1$  claimed  $A$  with degree of uncertainty  $CF = 0.5$

$P_2$  claimed  $A$  with degree of uncertainty  $CF = 0.6$

To obtain an overall-uncertainty over assumption  $A$ , we need to combine  $P_1$  and  $P_2$ 's opinions using parallel function described in Equation 5.6.

$$CF = \begin{cases} CF_1 + CF_2(1 - CF_1), & \text{if } CF_1, CF_2 \geq 0, \\ CF_1 + CF_2(1 + CF_1), & \text{if } CF_1, CF_2 < 0 \\ \frac{CF_1 + CF_2}{1 - \min(|CF_1|, |CF_2|)}, & \text{otherwise} \end{cases} \quad (5.6)$$

For example, given  $CF_1 = 0.5$  and  $CF_2 = 0.6$ , we have

$$CF = 0.5 + 0.6 \times (1 - 0.5) = 0.8 \quad (5.7)$$

As we can see, the over-all uncertainty factor is not the arithmetic mean of all uncertainties from all sources. An assumption will have a higher positive overall uncertainty if people tend to claim the same things with reasonable confidence.

### 5.2.3.2 User Credibility Extraction

On the other hand, a classification statement is more likely to be true reputable users endorse it. Apart from general user profile such as age, educational background, work experience, the credibility of someone in a particular field could possibly be deduced by looking at judgements previously made by this user. As discussed in [68] [69] [1], we could use a reputation model to extract user credibility from the annotation. In general, if statements made by a user are accepted by a community of people sharing a common interest, it is likely this user has a higher reputation in this domain. Certainty factors  $CF$  can then be used in conjunction with user credibility as Certainty-credibility factors ( $CCF$ ) to reflect the truthfulness and reliability of statements.

Unlike most reputation models, which only consider member's reputation as a global property regardless of its context, the proposed framework uses a domain-specific reputation model. The detail about the model and the credibility extraction process has been discussed in section 3.1.3.

### 5.2.4 Training Modified Naive-bayes Classifier

So far we have presented the approaches to gain some insight into the categorisation through annotation and how to store the information. These easy steps are usually manual as these should usually be done for a sample of objects. Once a sample is classified, we can use this to train an automatic classifier. Suppose  $U = \{u_1 \dots u_r\}$  is a group of users participated in the training. There are  $n$  target classes,  $C = \{C_1 \dots C_n\}$ ,  $o$  is an object with attribute values  $V = \{v_1 \dots v_m\}$ , which describe measured values of  $m$  features  $A = \{a_1 \dots a_m\}$  respectively. The aim of the training process is to build a classifier to predict the most likely classes for  $o$ , given each attribute  $a_1 \dots a_m$  has a specified value  $v_1 \dots v_m$ .

There are several variations of Naive-Bayes classifier, but at the time of writing, most of these classifiers are more concerned about handling complex data structures rather than incorporating collective-intelligence learning into the training process. Many researchers in this area seem less concerned about uncertainty in human decision-making, user credibility and other factors that may affect the accuracy of the



classification results on the training data set. Therefore, we propose a modified version of the Naive-Bayes classifier for a collaborative team, which considers various uncertainties involving multiple criteria that might affect team members' decision-making process.

The Naive-Bayes classifier classifies instances based on attribute values distribution. The classifier will return the conditional probability distribution  $P(v_1 \dots v_m | C_x)$  of an un-classified item  $o$  being a member of each class  $C_x$ , based on existing statements in the training dataset, in conjunction with uncertainty and credentials obtained from the collaborative annotation stage. In the end, the class with the highest probability will most likely be the best fit. To reduce the computational complexity, our underlining assumption is that the attribute variables are independent of each other. It is not always guaranteed to produce best results, but the Naive-Bayes algorithm could still achieve reasonable performance in practice even if variables are dependent [93].

According to Naive-Bayes probabilistic model, the posterior probability is proportional to the product of the prior probability measure and the likelihood function  $P(a_1 \dots a_n | C_x)$ , which can be written as follows [94]:

$$posterior(C_x) \propto P(C_x) \prod_{i=1}^n P(a_i | C_x) \quad (5.8)$$

In a flat categorization, the class with the highest posterior probability will be chosen. The classifier can be expressed as in Equation (5.9). But because our categorisation system is hierarchical, if subsumption inferencing is considered, the root node in the class tree will have the highest probability. So instead of predicting only one class, the algorithm should return the probability for each node in a root-to-leaf path of a class tree. We will discuss the detail in section 5.2.5.

$$classify(v, ..v_m) = \underset{c}{\operatorname{argmax}} P(C = C_x) \prod_{i=1}^m P(a_i = v_i | C = C_x) \quad (5.9)$$

An important question is how to aggregate multiple certainty factors from different users over the same statement. We define  $cF(u, t)$  as the certainty factor  $CF$  [57]

associated with the RDF triple statement  $t = \{subject, predicate, object\}$  claimed by user  $u$ . This can be used to express “instanceOf” assertion. For example, “user  $u$  claims  $o$  is an instance of class  $C$  with the degree of uncertainty  $CF$ ” can be described as  $cF(u, t = \{o, instanceOf, C\})$ . Assuming users  $u_1$  and  $u_2$  have positive certainty factors  $CF(u_1, t)$  and  $CF(u_2, t)$  respectively over the same statement  $t$ , we can combine  $CF(u_1, t)$  and  $CF(u_2, t)$  into an overall certainty factor  $CF(t)$  using the parallel function [67] in equation (5.10).

$$CF(t) = cF(u_1, t) + cF(u_2, t)(1 - cF(u_1, t)) \quad (5.10)$$

The prior probability distribution  $P(C = C_x)$  can be calculated based on the frequency in the training set using equation (5.11)

$$P(C = C_x) = \frac{\sum_{t' \in T'} CF(t')}{\sum_{t \in T} CF(t)} \quad (5.11)$$

where  $CF(t)$  is the overall certainty factor  $CF$  associated with class assertion  $t$  claimed by its contributors.  $T$  and  $T'$  are two sets of class assertion triple statements, where  $T = \{t | t.object \in C, t.predicate = instanceOf\}$  and  $T' = \{t | t.object = C_x, t.predicate = instanceOf\}$ ,  $C_x \in C$  is one specific class within the target classes set  $C$ .

A class attribute can be a nominal variable, numeric variable or a binary relationship that links to another object. Attribute types can be discrete or continuous depending on the categories and characteristics of different types of property in the ontology (DataType property/ Object property). For continuous and discrete attributes, the probability of  $o$  being an instance of class  $C_x$  given attribute  $a_i = v_i$  can be measured by the probability density functions (*pdf*) in Equation (5.12) and (5.15) respectively. Unlike probability, which always ranges between 0 to 1, a probability density function can acquire values greater than 1.

For continuous quantitative attributes, the posterior probability is calculated by the Normal (Gaussian) [95] probability density function in (5.12), where  $\mu$  is the mean

of values in attribute  $a_i$  in relation with class  $C_x$  and  $\sigma^2$  is the variance of values in attribute  $a_i$  associated with  $C_x$ . When calculating  $\mu$  and  $\sigma$ , we will only take attribute values into account when the attribute is considered as a relevant assessment criterion by some user during the annotation process.

$$P(a = v|C = C_x) = \frac{1}{\sigma_a \sqrt{2\pi}} e^{-\frac{1}{2} \left( \frac{v - \mu_a}{\sigma_a} \right)^2} \quad (5.12)$$

Not all assumptions contribute equally to the final average because each attribute-value pair is associated with a certainty factor ( $CF$ ) [1], which represents the relative reliability of said assertion. Our modified probability density function uses  $CF$ -weighted mean and variance rather than the simpler average (arithmetic mean). The respective equations are (5.13) and (5.14):

$$\mu_a = \frac{\sum_{t \in T_a, u \in U} CF(u, t) \cdot v_{u,t} \cdot k_{u,t}}{\sum_{t \in T, u \in U} CF(u, t) \cdot k_{u,t}} \quad (5.13)$$

$$\sigma_a^2 = \frac{\sum_{t \in T_a, u \in U} CF(u, t) (v_{u,t} - \mu)^2}{\sum_{t \in T, u \in U} CF(u, t)} \quad (5.14)$$

Let  $t = (o, a, v)$  be a tuple representing the fact that property  $a$  of item  $o$  has a measured value of  $v$ .  $T$  is a set of distinct  $t$  in the training set and  $T_a$  is a subset of  $T$  that concern attribute  $a$ . We denote by  $cF(u, t)$  a user  $u$ 's certainty factor over an RDF assertion  $t$ . In (5.13),  $v_{u,t}$  is an estimated value of attribute  $a$  associated with object  $o$  measured by user  $u$  where  $t = (o, a, v)$ . Variable  $k_{u,t}$  is the relevance factor which takes on a value of one when a user  $u$  states that this attribute is relevant to classification and 0 otherwise.

$$P(a = v|c) = \frac{|N_{a,v}| + 1}{|N_c| + m} \quad (5.15)$$

A discrete probability density function (5.15) is used for nominal, binary and ordinal attributes, where  $|N_{a,v}|$  refers to how many times attribute  $a = v$  occurs in class  $C$ . If we take the certainty factor into account, the number of occurrences of a nominal attribute  $a = v$  across a training set can be calculated using  $|N_{a,v}| = \sum_{t \in T'} CF(t)$ , where  $t \in T' = \{t | t.object = v, t.predicate = a\}$  is a subset of all assignments with respect to attribute  $a$ .  $CF(t)$  is the overall certainty factor over the statement  $t = (o, a, v)$  rated by all users. The steps that combine multiple certainty factors and user credibility factors were discussed in [1].  $|N_c|$  is the number of objects in the training set that have been classified as instances of  $C$ . When combined with  $CF(t)$ , it can be written as  $|N_c| = \sum_{t \in T} CF(t)$ . However, it might be the case that a specific attribute value does not occur at all in the training data set, which will result in a zero probability. In this case, we use Laplace smoothing to avoid zero probability estimates, where  $m$  is the count of distinct values of attribute  $a$ .

### 5.2.5 Applying the Classifier

The final step is to apply the classifier we trained as discussed in Chapter 5.2.4 to predict which of the classes are best suited for an unclassified object. Because our ontology-based categorization system is based on a hierarchical model, users might label objects at different levels of abstraction, for an unclassified object  $o$  with attributes list  $a_1 \dots a_n$ , our prediction result is not a single class with the highest probability, but a set of the most likely classes it might belong to. Considering the category as a weighted tree and every class  $C_x$  is a node of the tree, whose weight is the posterior probability  $P(a_1 \dots a_n | C_x)$  as explained in equation (5.11).  $w_1 \dots w_n$  is the sum of node weights of all possible root-to-leaf paths  $p_1 \dots p_n$ . Algorithm (4) describes the steps to obtain a list of possible root-to-leaf paths  $p_1 \dots p_n$  with the highest probability at the top. The recursive function *getHeaviestWeightedPaths* uses a simple depth-first search strategy to explore all possible paths and sort the array with insertion sort. Be aware the weight of a node is not the sum of all descendant leaves and branches because reasoning rules have been used to cover the class-subclass subsumption relations during annotation process we discussed in section 5.2.2.

**Algorithm 4** Calculate probability-weighted root-to-leaf paths

---

```

1: pathlist: Array
2: function GETHEAVIESTWEIGHTEDPATHS(node, path, length)
3:   if n is Empty then
4:     calculate node.pb from (5.12) or (5.15)
5:     path[length].pb  $\leftarrow$  node.pb; path[length].id  $\leftarrow$  node.id
6:     length  $\leftarrow$  length + 1
7:   end if
8:   if n is leaf then pathlist.add(path)
9:     SORTPATH(pathlist);
10:  else
11:    for each c  $\in$  node.child
12:      GETHEAVIESTWEIGHTEDPATHS(c, path, length)
13:    end if
14: end function
15: function SORTPATH(path)
16:   for i = 1  $\rightarrow$  (pathlist.length - 1) do
17:     e  $\leftarrow$  pathlist[i]
18:     ipos  $\leftarrow$  i
19:     while ipos > 0 and pathlist[ipos - 1].pb < e.pb do
20:       pathlist[ipos] = pathlist[ipos - 1]
21:       ipos = ipos - 1
22:     end while
23:     pathlist[i] = e
24:   end for
25: end function

```

---



# Application and Evaluation

---

This chapter will first give an overview of existing data models and methods in comparison with the proposed approach and then discuss the advantages and shortcomings of it. Then it will introduce several applications developed for a focus group based on the methodology discussed earlier in Chapter 3, 4 and 5. In the end, the chapter will discuss a series of experiments that have been conducted to evaluate the performance and usability of the proposed models and methods.

## 6.1 Evaluation and Comparison: KR and Uncertainty Models

Many of the existing Knowledge Representation (KR) models are based on first-order logic or description logic, either in the form of facts, rules or ontology. Several existing KR models are capable of representing uncertain data. Most of the models are based on three formal logics: (1) Fuzzy logic (2) Bayesian probability and (3) Certainty factor. Fuzzy logic is about the degree of truthfulness. Bayesian probability is about making predictions based on the current state of knowledge. However, what we want to describe is the level of uncertainty of a certain statement entirely based on a given individual's own judgment and interpretation - in other words, the confidence level with regard to the assumption. Many of these uncertainty models are not aimed at providing supports for recording statements given by individuals in a collaborative team. (Abbreviations used in Table 6.1 are described in Table 6.2).

Table 6.1: A summary of KR/Uncertainty models

KR models	Formal Logic	Syntax	Collaborative Support
Prolog	FOL	FR	S
Fuzzy Prolog	FOL FL	FR	S
OWL-2 Fuzzy DL	DL	ONT	S
RuleML	FL	R	
PR-OWL	BP FOL	ONT	S
OWL URW3-XG	DL	ONT	S
Uncertainty Ontology	UNI		
Collaborative Annotation	DL	ONT	M
Ontology	CF	R	

## 6.2 Evaluation and Comparison: Querying and Ranking Methods

Table 6.3 gives an overview of some of the querying techniques such as XQuery/IR [96], Transact-SQL [97], SQWRL [98], SPARQL 1.1 [77], SPARQL-DL [78], R2DB [99] and our proposed querying method. The table compares different querying techniques regarding (1) support for fuzzy search (2) personalization of query criteria (3) aggregating individual preferences and (4) reasoning support.

Several fuzzy extensions of the various query languages are developed. XQuery/IR is an extension to XQuery in which an additional rank operator is integrated into XQuery; Transact-SQL is a proprietary extension to SQL [100], in Transact-SQL, weight matches in full-text search is supported by CONTAINSTABLE clause. CON-



Table 6.2: The abbreviations used in Table 6.1

Formal Logic	FOL	First-order Logic
	DL	Description Logic
	FL	Fuzzy Logic
	BP	Bayesian Probability
	CF	Certainty Factor
Syntax	ONT	Ontology-based
	R	Rule-based
	UNI	Unspecified <sup>1</sup>
Collaborative Support	S	Single-user mode
	RP	Multi-user moodel with integrated reputation model

TAINSTABLE clause creates an intermediate table with a RANK column which uses different values to indicate how well a row matches the criteria. These two querying techniques are used only for text-based full-text fuzzy search. SPARQL is an SQL-like text-based query language for querying RDF. It is the foundation of our proposed framework but SPARQL itself does not natively support OWL, neither does it support weight matches or deductive reasoning. As a result, SPARQL needs to be used in conjunction with DL reasoner and SWRL rules to achieve the desired results. SPARQL-DL [78] is a subset of SPARQL that has a well-defined OWL-DL based semantics. It is more expressive than SPARQL 1.1 and some DL-based query languages but it still does not support weighted matches. SQWRL is an SWRL (Semantic Web Rule Language) based Prolog-like language for querying OWL ontologies. In contrast to OWL's open world assumption, SQWRL only works on known individuals and the result of the query will not be written back to the ontologies [98]. SQWRL also does not support weighted matches and currently it is still not possible to query over relations. DL-query [101] is a language for querying OWL ontology based on well-defined OWL Manchester syntax and it does not support weighted matches. Another research outcome that is related to this research is R2DB [99], a framework for ranked path queries over weighted RDF graphs called R2DF [102]. In terms of reasoning supports, many OWL-based querying languages allow DL reasoner to be used for checking the consistency of the ontology

Table 6.3: A summary of structured query languages for weighted graph

Structured Query Techniques	Syntax	Weighted graph/fuzzy data	Personalized criteria	Collaborative search	Reasoning supports
XQuery/IR	XQuery extension	Y	N	N	N
Transact-SQL	SQL-style	Y	N	N	N
SQWRL	DL-based SWRL extension	N	N	N	OWL-DL SWRL [81]
DL-Query	DL-based	N	N	N	OWL-DL
SPARQL 1.1	RDF-based	N	N	N	SWRL
SPARQL-DL	RDF-based DL-based	N	N	N	OWL-DL
R2DB	DL-based	Y	N	N	OWL2
Proposed querying framework (Chapter 4.1)	RDF-based SPARQL 1.1 extension	Y	Y	Y	OWL Custom rules

and simple SWRL rules can be used to achieve basic deductive reasoning. However, the majority of the query languages are designed mainly with a single user in mind. They are not intended to be used for more than multiple users to interact in a collaborative environment. It is important to note that several existing query methods have developed extensions to achieve weighted matches but very limited supports are provided for weighted query criteria. Most existing query techniques are not designed for multiple query makers. Regarding inferencing, many OWL-based methods have preliminary support for OWL/SWRL hybrid reasoning though there are a few limitations. DL reasoning is primarily used for checking OWL consistency and building inferred class hierarchy, which is usually not the primary goal of RDF searching; SRWL

plays an major role in the deductive reasoning though the language has its limitations. For example, SWRL does not support predicate as a variable.

Compared to these existing querying techniques listing above, the proposed querying framework is based on graph pattern weighted matching and it naturally supports weight graph. The framework enables individual query maker to customise criteria weight for each triple pattern and aggregated individual preferences. The framework also incorporates SWRL-style rules but with enhanced deductive reasoning support. For example, the framework allows predicate as a variable which currently not supported in SWRL.

### **6.3 Collaborative Image Annotation Framework for Human Representation**

A web-based prototype application of the framework has been implemented in Java. Figure 6.2 illustrates the architecture of the system. The application consists of several major parts. A web-based annotation interface allows users to annotate images on the web. An annotator can describe concepts and relationships by constructing a set of trusted triple statements, with the help of an extendable WordNet lexical database and a remote OpenCalais[103] service. WordNet groups the synonymous words, provides a precise definition of terms and defines the semantic relation between these synonym sets (e.g. hypernym hierarchies). It works in conjunction with the OpenCalais service to help the user to identify the possible topics (category or domain) of an image. Besides, an annotator can also specify the original degree of certainty about an assumption to express his or her opinion on it. A screenshot of the interface is shown in Figure 6.1. For every triple statement, the overall user certainty and credibility factor will be computed by a certainty calculator and a user credibility calculator respectively. Afterward, these will be combined into a composite user certainty-credibility factor used for ranking search results. The query engine can retrieve annotation data from the trusted triple repository via an Access Layer using SPARQL queries and a native OWL API. The system also utilises both description logics (DL) and deductive rule-based reasoning for inferencing. The search screen is currently a textbox that allows

entry of a query and the results show a ranked list of images that match the given query – both not very exciting to look at in a screenshot.

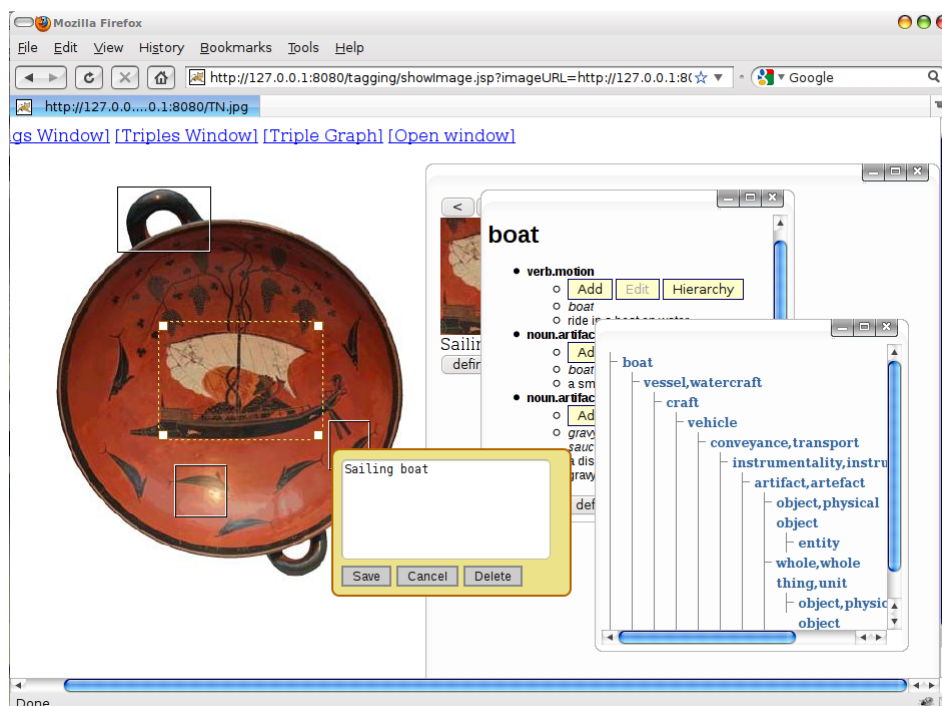


Figure 6.1: A screenshot of the prototype implementation

At the moment we are evaluating our approach with users from the archaeology community working on an archaeological project: “Tracing Networks: Craft Traditions in the Ancient Mediterranean and Beyond” [12]. The proposed graph-based querying technique can be used to search annotated data. We will discuss the experiments conducted in sections 6.4.3 and 6.4.2.

## 6.4 Evaluation: Graph-based Query Language

We have conducted a series of experimental evaluations regarding the usability and expressivity of the proposed graphical query builder on several different occasions: (1) A practical lab session held in conjunction with “Ontology in Archaeology” workshop at the University of Leiden mostly focuses on the usability and learnability. (2) An experiment on querying Loom-weight dataset in the Tracing Networks project.

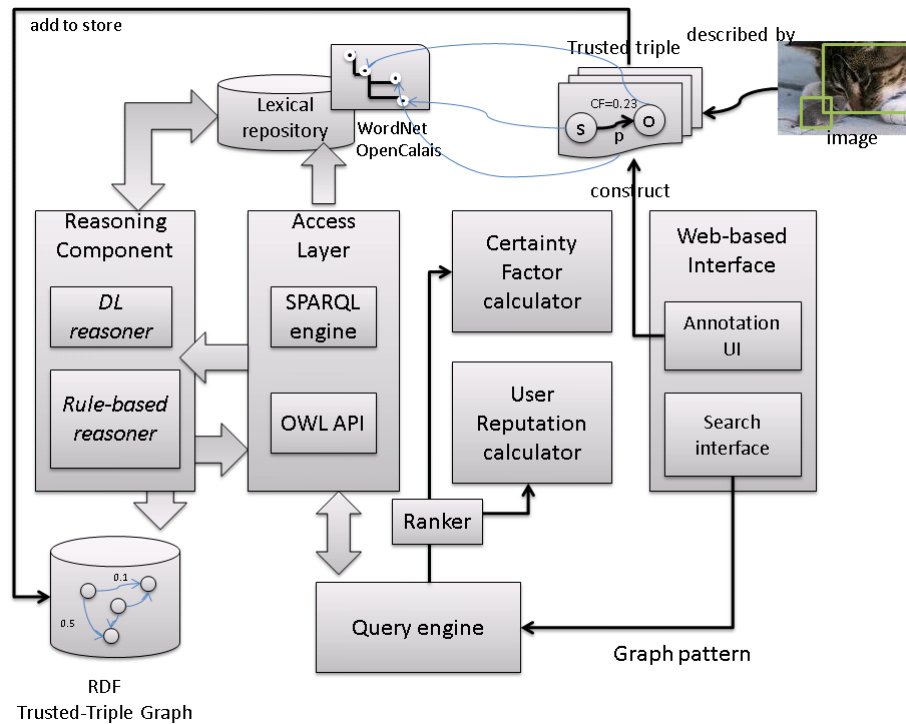


Figure 6.2: System architecture overview

### 6.4.1 Usability and Learnability

Figure 6.3 illustrates the trade-off between simplicity and expressivity. Table 6.4 lists four different techniques for querying. Method (A) only consists of one single text box, such “Google-style” search interface is very simple and easy to use but offers very limited expressivity. Method (B) uses a complex search form, however, “Library-style” advanced search is not flexible enough, sometimes overly complicated. Method (C) is to use native query languages such as SQL, HQL or SPARQL. They are expressive formal specification languages but it would be a challenging task for non-IT professional to write a validated query without proper training.

In Tracing Networks programme, we conducted a survey on the use of database and query techniques used by people working in the cultural heritage sector. The focus group in Tracing Networks programme consists of 13 participants, all of them are archaeologists specialising in their fields. Members of the focus group are ordinary non-technical end-users, none of them has background in computer science or engineering.

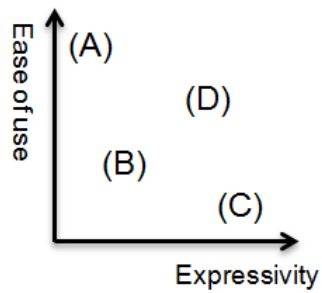


Figure 6.3: Search interface: Simplicity vs Expressivity

Table 6.4: Comparison of different querying mechanisms

Type	UI compoments	Example	Ease of use	Expressivity
(A)	Single textbox	Google	easy	very limited
(B)	Filter-based	Library Advanced search	intermediate	limited
(C)	Native query	SQL, SPARQL	hard	very high
(D)	Graph-based	Our proposed framework	relatively easy	high

Table 6.5 shows various software products and primary query technique used by the focus group.

Table 6.5: Focus group: datasets and primary searching techniques

Dataset	Software product	Primary searching techniques
Human representation	MS Access	built-in MS Access query
Cooking pots	MS Excel spreadsheet	built-in filter
Loom-weights	Microsoft Access	built-in MS Access query
Tiryns (excavations)	FileMarker Pro	built-in form-based search

The survey shows that many end-users would prefer lightweight standalone desktop RDBs or simple spreadsheets. These commercial products are mature and robust, many of them provide simple but handy built-in query tools. However, some problems arise during the face-to-face interviews with archaeologists regarding the usability of these tools.

Currently, users can only perform limited string matching search using built-in

query tools, but an experienced IT technician is required to assist with more complex queries. Most users want to be able to write more sophisticated queries on their own without additional resources or seeking extra help from IT technicians. Although many form-based search interfaces provide some query templates, ordinary users still have difficulties understanding the database schema before writing more sophisticated queries. Most queries involve multiple tables, but ordinary users still find it difficult to understanding primary-foreign key constraints. Even for a skilled user, writing a validated query requires a lot of time and effort.

#### **Survey - traditional form-based vs proposed graph-based search**

We performed a survey on the ease of use and learnability of the proposed graph-based querying method in a one-day workshop on 9 May 2014 at the Faculty of Archaeology, University of Leiden. The detail of the workshop can be found here: [http://www.tracingnetworks.ac.uk/ontologies\\_program.pdf](http://www.tracingnetworks.ac.uk/ontologies_program.pdf).

A total number of 22 people from the University of Leiden with different backgrounds in Archaeology (from classical Mediterranean to pre/post-colonial Caribbean studies, anthropological interests to museum and meta-data interests) attended the workshop. Many of the participants were working with large and complex archaeological datasets. Some of them had some basic knowledge of relational database and some previous experience with spreadsheets (e.g. Microsoft Access/Excel). However, none of the participants had any prior knowledge of ontologies or any structured query languages.

Throughout our participation in discussion, it appears that retrieving information without prior knowledge of the underlying query language syntax is one of the major obstacles that prevents many archaeologists from conducting the analysis of their dataset.

After a brief introductory session on the ontological database (45 minutes), we gave a tutorial on our graphical query language <sup>2</sup>. The participants started learning through practical exercises. A few sample queries as listed in Table 6.6 were shown to the participants to demonstrate the capabilities of the graphical query language:

In the experiment, participants were asked to complete a list of tasks using two

---

<sup>2</sup>slides used in the tutorial: [http://www.tracingnetworks.ac.uk/Leiden\\_workshop.pdf](http://www.tracingnetworks.ac.uk/Leiden_workshop.pdf)

Table 6.6: Queries used in tutorial session

Query	Demonstrate how to ..
Q1	relate instances via intermediate nodes
Q2	draw basic graph pattern
Q3	draw aggregation function in a query
Q4	draw complex graph pattern involving binary logical operators
Q5	draw pattern to query relationship over the graph
Q6	search for "instance of" relations

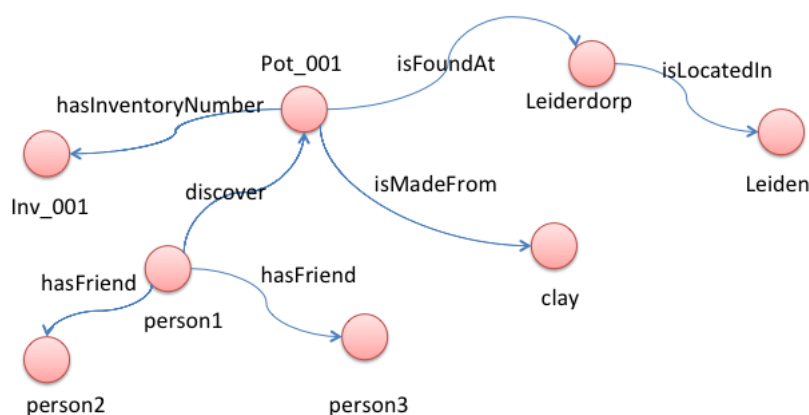


Figure 6.4: RDFS graph used for query tasks

different querying techniques: (1) given a spreadsheet or small relational database (MS Excel/Access), use a built-in filter or form-based query interface to perform T1-T6 as described in Table 6.7. (2) Given an equivalent of the dataset in RDF graph as shown in Figure 6.5, draw a graph pattern for each task in Table 6.7. The evaluation result is summarised in Table 6.8.

In this experiment, triples could be weighted differently and they had significant influence over the relevance ranking of the results. For example, in T3, clay-based object can be expressed using a pattern consisting of several triples:

- s1: *?artefact rdf:type tn:artefact*
- s2: *?artefact rdf:type tn:DrinkingVessel*



Table 6.7: Query tasks for usability evaluation

Task	Description
T1	What is the latitude and longitude of Leiden?
T2	Who excavated <i>Pot_001</i> ? List all his/her friends
T3	List all clay-based objects found in Leiden (inc. suburban)
T4	Show the inventory numbers of all artefacts
T5	Show the percentage of different objects in each category
T6	Show everything (persons, sites etc.) linked to <i>Pot_001</i> and their relationships

- s3: *?artefact tn:isMadeFrom tn:Clay*

As discussed in section 3.1.2, any statements in the form of triples are always associated with some degrees of uncertainty. Users can rank the triple patterns and assign a relative weight to each. Weighted criteria matrix is used in conjunction with certainty factors for ranking. For example, pattern *s2* matched an unknown object that looks like a fragment from a cup (e.g.  $CF = 0.7$ ). An archaeologist specialised in tableware can prioritise *s2* while *s3* might be considered more important by someone who is a pottery expert. In this case, objects of type *DrinkingVessel* were more likely to appear at the top of the search result. Furthermore, adjusting the threshold of *rdf:type* property in conjunction with the *GROUP BY* clause in T5 will have a significant impact on the search result. Because aggregation function *COUNT* will be applied to each group and the members of every group is determined by a statement *?object rdf:type ?class*, adjusting the weight of *rdf:type* will increase or decrease the size of the group for each *GROUP BY* clause. Hence the result may differ for this reason.

Given the data provided in this experiments, graph-based approach shows some advantages over form-based filtering approach in terms of usability. For example (1) despite T1 being achievable through both methods, half of the participants still failed to complete T1 using a form-based filter. (2) form-based filtering technique works to a considerable degree for some straight-forward queries such as T4 but it is incapable of performing a more complicated task such as T5, which involves the use of aggregation function. (3) Some participants were able to complete the first part of T2 using a

Table 6.8: Task achieve rates by individuals - (Table 6.8)

Task	Completion rates	
	Form-based filter	Graph-based query
T1	45.4 %	90.0 %
T2	36.3 %	81.8 %
T3	50.0 %	95.4 %
T4	90.0 %	90.0 %
T5	13.6 %	68.18 %
T6	0 %	59 %

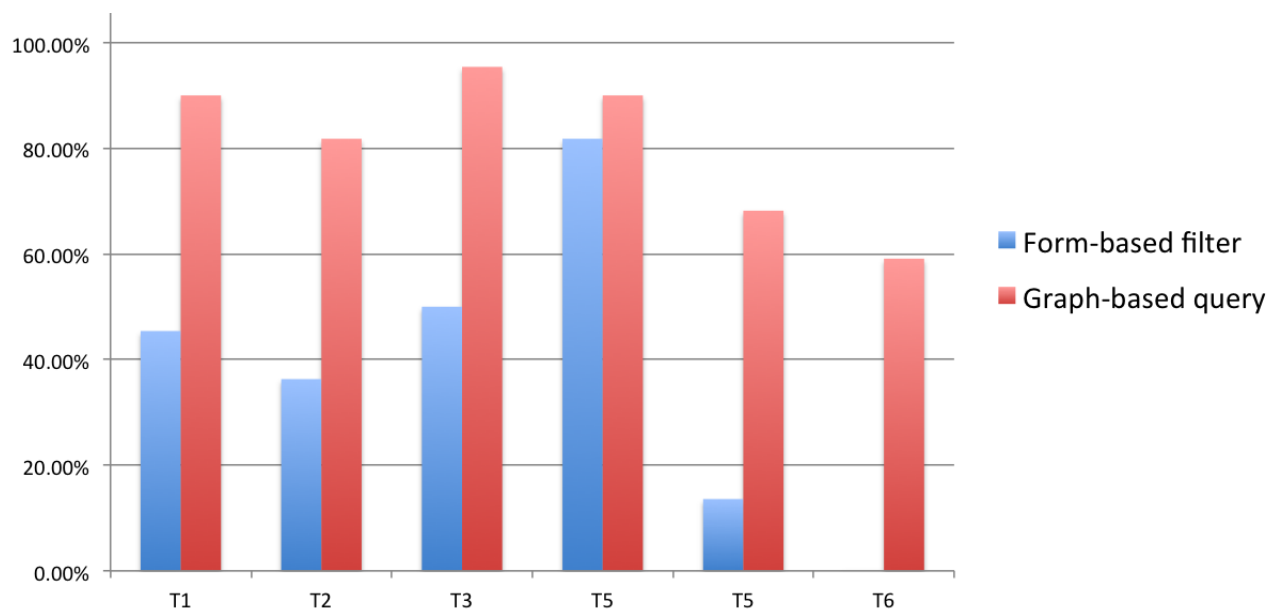


Figure 6.5: Comparison of task completion rates by individuals using form-based filter and graph-based querying approaches

form-based filter but failed to write a join query to search across multiple tables. In contract to this, many felt drawing triple pattern easier. (4) T3: none of the participants managed to obtain the objects excavated from the suburban areas of Leiden (e.g. leidendrop) through a form-based filtering because the approach does not support transitive closure reasoning in *islocatedIn* relation. (5) T5: the majority of the participants were struggling to build a query containing GROUP BY and aggregation function us-

ing form-based interface. (6) T6: none of the participants was able to complete this task using a form-based filter because it asks for individuals (inc. person, objects) that are somehow associated with *Pot\_001* through some link which is not explicitly specified. However, more than half of the participants were able to complete T6 by drawing a graph pattern with the predicate as a variable. Given the data provided in this particular case, graph-based approach shows some advantages over form-based methods.

### 6.4.2 User Experience Assessment

Another assessment of user satisfaction with the proposed collaborative searching techniques was also conducted. We divided participants into six groups and each group consists of three or four people sharing one desktop PC. Participants were asked to give a possible graph-based query solution to T1-T6 as listed in Table 6.7. The result of the survey will be discussed in section 4.3.

**Personalised search as an individual:** each group member customised the query by assigning different weights for triple patterns. As individual query maker, members can express their preferences over different query criteria (triple patterns), so that results can be custom tailored for individuals. (Note: the relative importances of the variable nodes derived from the query topology discussed earlier in section 4.3.2 was also considered).

**Collaborative search as a team:** for each task in table 6.7, every group had to build a query collectively where the collective preferences for every triple pattern can be calculated using Equation 4.4. After the lab session, we conducted a survey on the level of satisfaction of users with three graph-based querying options available to them (1) original query without weight settings (2) personalised query with weighted criteria (3) query collectively created by all members of the group. When asking the question: which graph-based queries best matches your needs and expectations? Most participants agreed that option 2 was the best matched while option 3 scored second as it requires a compromise among group members. Option 1 was the least favourable

option.

Table 6.9: User satisfaction survey: which graph-based queries best matches users' needs and expectations

Level of satisfaction	Types of query	Original	Personalised	Collective
First choice		0	21	1
second choice		0	1	21
third choice		22	0	0

### 6.4.3 Adaptability and Expressiveness

Another experiment was conducted with a volunteer, who is currently an archaeological officer in the Superintendency of Archaeological Heritage of Piemonte and Museum of Egyptian Antiquities, to evaluate the adaptability and expressiveness of proposed graph-based querying technique. The volunteer's research primarily focuses the analysis of loom-weights across the Mediterranean from Late Bronze Age to 3rd century BCE, the exchange of shapes, decorations, craft knowledge and personal or group behaviours within the larger social and economic interactions [104].

Applying graph-based querying technique to real-world problems requires adaptability. The aim of this experiment is to investigate the adaptability of the proposed methodology and evaluate the expressiveness of the querying technique in a real-life situation. The idea is to examine how many of the questions listed below can be answered by the volunteer on his own after completing a one-hour private tutoring session. An ontological data model for loom-weights was designed and implemented in OWL-DL (Appendix A.2). A previously built loom-weight database is mapped and transferred into this ontology-based model using the method discussed in section 3.2. Acting as a non-technical query maker, the volunteer provided us with a list of questions he would like to ask:

- (1) Which is the most attested type of loom weight (conical, pyramidal, etc.) in percentage across all sites?

- (2) List all loom-weights whose dimension ( $\text{height} > 0.1$ ), display related decoration techniques and motifs.
- (3) Which techniques were used in the decoration of loomweights found in Monte Sannace.
- (4) What is the distribution of these different types in relation to the nature of the site (household context, tombs, place of cult etc.)
- (5) List all 3rd Century BC loom-weights weight between 50g to 70g (assuming time span might be recorded in different formats "3rd Century BC", "First half of the 3rd Century BC" or absolute chronological dating such as "200-250BC", etc.)
- (6) List all loom-weights dated between 600BC and 400BC.
- (7) List loom-weights with similar decorative motifs found at Gravira.
- (8) Show all loom-weights that fall into 2 Century BC (dated back to a period that overlapped with 200BC-100BC)
- (9) What is the distribution of these different types in relation to the sites? i.e. which are the different percentages of the types (discoid, etc.) in the different sites?
- (10) What is the distribution of these different types in relation with the time span?
- (11) What is the range of the weight in comparison with the different types? (discoid, pyramidal etc.)
- (12) Is there any relation between two different dimensional parameters, for instance weight versus diameter in the discoid loom weights and weight versus height in the pyramidal loom weights, using only the exact values?
- (13) What is the distribution of the type of decoration? (stamp, stamp on relief, graffito, etc.) in relation to different types (conical, discoid, etc.)

- (14) What is the distribution of a single decoration (for instance Gorgoneion on relief, two heads facing each other on relief and rosette impressed) in different sites?

Table 6.10: Evaluating volunteer’s ability to draw queries independently with respect to questions in Table 6.10. [Abbr.: **A**: Achieved; **A(R)**: Achieved with reasoning rules applied; **PA** Partially achieved ; **PA(R)** Partially achieved with reasoning rules applied; **N**: Not achieved].

Result	Query ID
A	1, 2, 3, 4
A(R)	5, 6, 8, 10
PA	9, 13, 14
PA(R)	7
N	11,12

After one-hour private tutoring session on graph-based query <sup>3</sup>, the volunteer was asked to use the graph-based query to solve the above questions. The adaptability and expressiveness of proposed graph-based querying technique can be measured by analysing the observational data.

Table 6.10 shows the volunteer’s result. Overall, 12 out of 14 questions in the list were answered or partially answered with GROUP BY clause (e.g. Query 1,3,4) as well as conditional filtering (e.g. Query 2). Furthermore, the volunteer was able to answer questions 5, 6, 8 and 10 with the aid of predefined chronological reasoning rules. The volunteer on his own after training, was able to draw queries involving GROUP BY. In TN ontology, an artefact is always associated with an instance of Time-span, Question 8 can be answered using a query in conjunction with generic chronological reasoning rules. (Note: generic chronological reasoning rules were created beforehand in consultation with the archaeological domain experts and tester before the experiment)

The volunteer partially achieved Question 9, 13 and 14. These questions were

---

<sup>3</sup>Tutorial used in this session can be found at [http://www.tracingnetworks.ac.uk/Leiden\\_workshop.pdf](http://www.tracingnetworks.ac.uk/Leiden_workshop.pdf) (page 63-91)

intended to show correlations between two variables in different groups. Currently, it is not practical to answer each question using one single query, but it is possible to write a parameterised template and run it for multiple times.

In Question 7, the volunteer assumed there was a chain of *has\_similar\_motif* properties hence a reasoning rule was applied to enable the transitive closure of *has\_similar\_motif* relations. However, the inferences for transitivity in *has\_similar\_motif* relation can be weakened, that is to say, such relation is not strictly transitive. For example, if object A and object B has a similar decorative motif, object B and object C has a similar decorative motif then it implies that A and C might have similar decorative motif. The length of the transitive chain can be weakened as the number of iterations increase. For this reason, the result of the query created by the volunteer was not entirely correct.

The volunteer was not able to draw the queries to answer questions 11 and 12. Question 11 asked about the range of weights. Although currently it is not possible to retrieve a range of values using proposed graph-based query nor the underlying SPARQL, it is possible to answer this question the other way around. Instead of asking the ranges of weight for a certain type of loom-weight (e.g. discoid), the query maker could draw an alternative graph pattern to answer the question from a different perspective such as : What is the distribution of different types of loom-weights whose weight falls into a certain range (e.g. 70 - 90 g).

Question 12 requires non-trivial statistical correlation analysis. For this reason, the tester found the task difficult to complete using one single graph pattern. (However, it is possible to achieve using aggregation function and GROUP BY clause and present the result in a tree map using the prototype developed in section 6.3)

### Summary

The feedback on the graphical query interface received from the workshop participants and the volunteer was generally positive [105]. Many had said that the graphical query interface offered an entirely different searching experience comparing to the traditional method and many felt that the query tool was more capable and flexible than what they had initially thought. Although it did require necessary mind shifts from a table-style keyword filtering to triple-oriented pattern matching. The vast majority of

the participants believed the graphical search tool developed in this research has great potential, and many of them have also shown a keen interest in the use of ontology for data archiving and statistical analysis [106].

## 6.5 Evaluation of Proposed Categorisation Method

### 6.5.1 Semi-automatic Sense Detection for Ancient Art

A prototype implementation has been developed to evaluate the effectiveness and accuracy of the algorithm. *CCOnt* has been modelled as an OWL-2 ontology using Protégé editor. Experiments have been carried out with several datasets from the Tracing Networks programme [12]. We use data from these datasets: (1) Metaponto Loomweight dataset and (2) Human representation database and (3) House of Menander (Pompeii) database in section 4.1.1. Loomweight database has been used for usability and learnability evaluation as discussed earlier in section 6.4; Human representation database was used in the assessment of the semi-automatic classification; House of Menander (Pompeii) database was used in BDRT workshop [107] to showcase the graphical querying interface.

As we mentioned earlier, a key objective of this of this interdisciplinary research project is to work in close collaboration with archaeologists, to model the Tracing Networks Ontology (Appendix A.2) for the archaeology community. Tracing Networks Ontology functions as a meta-model to support classification and analysis of heterogeneous data in different sub-projects. . As the ontology becomes more mature and robust, it can be shared, re-used across teams of varying sizes in the future. Collaborative Classification Ontology (CCOnt) represents the upper ontology for all sub-domain-specific ontologies, jointly with an extended CIDCO-DRM [40] ontology. Excavation data and archive are represented in an ontological model instead of tables and records in a relational database.

In a sub-project – “Translating art and craft: Human representations, identities and social relations in the Late Bronze and Iron Age of Central Europe”, archaeologists want to detect different types of visual scenes (e.g hunting, boxing, dancing etc) appearing in the human representation database based on what types of human figures, animals,



objects appeared in the images as well as their characteristics. For example, the pictures below in Figure (6.6) depict two human figures (a) a rider on a horse-like animal and (b) a woman with a pottery vessel on her head.

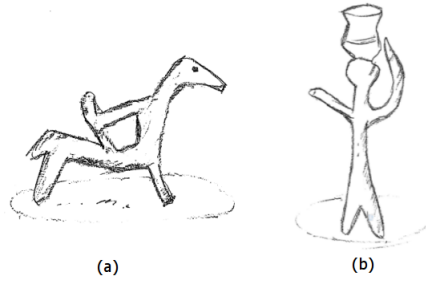


Figure 6.6: Depiction of human figures in Late Bronze and Iron Age

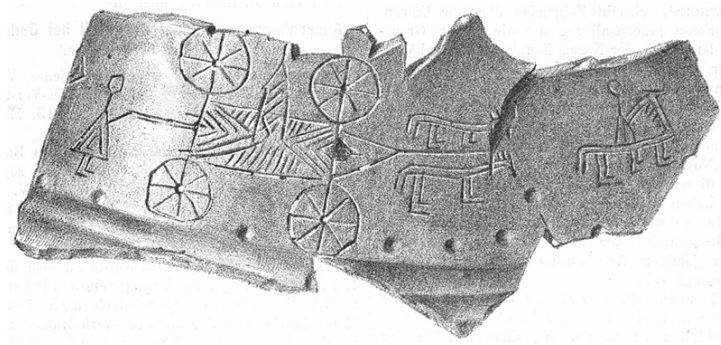


Figure 6.7: Ceramic vessel found at Sopron-Várhely, Hungary depicting a waggon ride scene

If human figures, animals, objects and their relationships match a certain pattern, they compose a scene. For example, if the picture contains one four-legged horse-like animal and one male figure with his arm stretched out straight grabbing reins, it is likely to be identified as a “waggon ride” scene. If the picture contains four men carrying shields with their left hands and facing the same direction, it might be a scene of “warriors march”. The details of how to annotate elements and relationships with uncertainty factors are explained in Chapter 3.

A query builder has been implemented to help archaeologist to retrieve information and label relevant attributes from a complex ontology. Instead of writing a query by

hand, the application allows archaeologists to draw queries and reasoning patterns as graphs using a diagramming tool (yEd). The system transforms the graphs patterns into SPARQL queries, which are executed. The result of the search is exported in ARFF format.

An experimental evaluation of this collaborative classification method was conducted using the datasets from the human representation database. The category hierarchy used in this experiment was generated following the procedure described in section 5.2. An image annotation and matching tool equipped with the modified Naive-Bayes classifier as described in section 5.2 was implemented. The core part of the algorithm is written in Java. Images in the Human representation database are stored as PNG files where their relative paths and certain-credibility values for the trusted triples were kept in OWL 2 model as described in section 3.1.1. Some frameworks or libraries such as WEKA and OpenCV were also used in the implementation.

The evaluation consists of three steps: (1) request a group of archaeologists to annotate images (assuming the total number of images is  $N$ ) in the human representation database (2) select a random sample of images ( $TN$ ) to train the classifier (3) apply the classifier to classify the remaining scenes and compare the predicted result with manual classification results suggested by archaeologists. In the experiment, we examined 2945 scenes instances consisting of approximately 160,000 triple statements created by a group of archaeologists. Table 6.11 lists the total number of successfully classified instances. We treated auto-classification of an artefact as successful if manually tagged category falls into the heaviest path that we predicted. As our result of prediction is a list of root-to-leaf paths with the most heavily weighted path at the top (this experiment only considers the first and the second heaviest paths). *Rate1* shows the success rate for the first heaviest path and *Rate2* shows the success rate for the first or the second path. Figure 6.8 shows statistical analysis of classification results in relation to the size of the training set.

As we can see, larger training sample sizes generally lead to higher classification accuracy. However, a training sample of relatively small size can still achieve reasonable performance. The evaluation result shows this approach is capable of processing a large number of instances based on a relatively small training set. In this experiment,

Table 6.11: Comparison of classification accuracy in human scene detection (N: total instances; P1: correctly classified instances in Path 1; P2: correctly classified instances in Path 1 and Path 2; TN: Training sample size)

<i>N</i> total instances	<i>P1</i> correctly classified	<i>P2</i> correctly classified	<i>TN</i> training set size	<i>Rate 1</i>	<i>Rate 2</i>
80	71	76	2865	88.75%	95.00%
225	181	206	2720	80.44%	91.55%
790	559	663	2155	70.76%	83.92%
1005	750	834	1940	74.62%	82.98%
1201	914	997	1744	75.53%	83.01%
1455	1065	1203	1490	73.20%	82.68%
1671	1202	1297	1274	74.33%	77.61%

despite the fact that not all attributes are completely independent of each other (for example, property *hasDressCode* and *hasGender* are not independent of each other), the algorithm still produces reasonable classification result.

Figure 6.8 shows that the classification became more accurate as the sample size increases. For instances, when 2865 images were used for training the classifier, 76 out of 80 images were correctly classified (Rate 2). This method saw a success rate of approximately 95% when the training set was about 35 times larger in size than testing set. The algorithm had the lowest accuracy rate (approx. 75%) when its training set and the testing set are similar in size.

The feedback received from the principal investigator of the human representation project <sup>4</sup> is positive. The user, who is an expert in human representations in the late bronze and iron age, has suggested that the semi-automatic classification technique developed in this research "has achieved a considerable degree of accuracy" in scene detection compared to manual classification. The feedback from other users was mostly positive though there is still room for improvement. Some have already shown their

<sup>4</sup>Translating art and craft: human representations, identities and social relations in the late bronze and iron age of central europe [109]

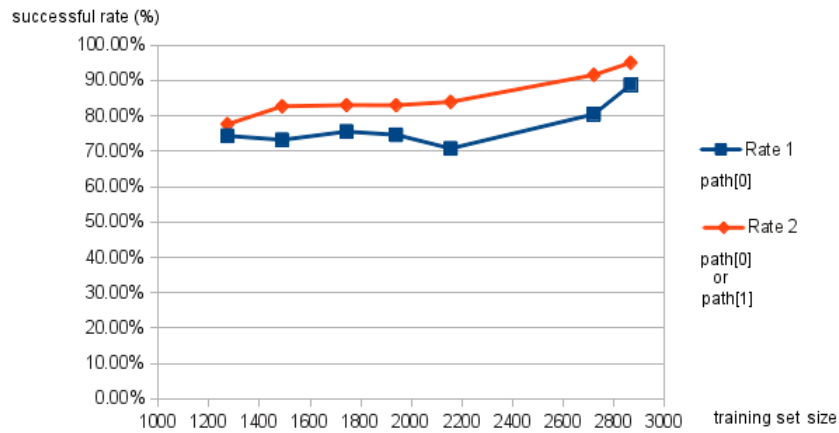


Figure 6.8: Classification accuracy in relation with the size of the training set

interest in BDRT workshop [107]. In addition to classifying unlabelled instances, another possible area of this application is to verify data integrity - in such a case, all instances have been manually classified, the entire dataset can be used as the training set to check its consistency to help reduce human errors.

## Conclusions and Future Work

---

This thesis presented several aspects of the research regarding collaborative annotation, search and categorisation. First of all, this thesis discusses how semantic web technologies could be applied to provide a more expressive and flexible way for annotating objects comparing to traditional text-based tagging. An ontology-based data model has been created to describe arbitrary statements along with their degree of uncertainty. At the same time, the annotator's domain-specific credibility can be recorded and measured. Besides, an ECA-based transformation approach was proposed to convert data stored in the relational database to the proposed ontology-based data model. In the next step, a visual querying technique was presented. The method is based on graph pattern matching and can be used in conjunction with the rule-based reasoning for searching data sets. Furthermore, how personalised and a collaborative search can be achieved while considering individual query maker's perception of the relative importance of different criteria was discussed. In the end, this thesis investigates the feasibility of applying various classification techniques to help categorise annotated objects.

Several proof-of-concept prototypes have been implemented, a series of experiments were carried out. A focus group consisting of archaeologists and historians working in the cultural heritage sector was invited to help evaluate the usability of the proposed approach. The results of the experiments show the proposed data model could provide a flexible structure to record annotations, while representing uncertainty and credibility. The proposed graph-based searching technique also demonstrated its ease of use and expressiveness, which enables collaborative team members to create a complex query collectively in a more convenient manner; the proposed classification method also

showed reasonably good performance with real-world data sets.

## 7.1 Summary of Research

In summary, this thesis presented the research in some key areas of specific focus including (1) Collaborative semantic annotation, in particular how to describe arbitrary statements, incorporate uncertainty and measure user credibility. (2) Developing an intuitive query and reasoning framework for the collaborative working environment, and (3) Collaborative categorisation based on user-supplied annotations. In the end, non-technical volunteers working in the archaeology domain were enlisted to evaluate the usability and user experience of the proposed approaches. The main research contributions in this thesis are listed below:

- Designed and implemented an ontology-based data model for collaborative annotation. The model allows uncertainty of the statements as well as users' domain-specific reputations to be represented.
- Developed a method for transforming legacy relation database to the proposed data model.
- Proposed an intuitive graphical querying method with rule-based reasoning support to enable individuals or a collaborative team to search semantically enriched content, considering factors including the relative importance of criteria derived from preferences and the nature of the query pattern.
- Proposed a semi-automated categorisation framework to help classify semantically annotated content, taking into account different criteria involved in the user's decision-making process.
- Evaluated the proposed methodologies in real-world situations with a focus group consisting of archaeologists and historians.

## 7.2 Future work

Despite much work has been done, many things could be improved. The main directions for future research include:

- **Extending the model to accommodate other factors**

At the moment the domain-specific reputation of a user is entirely based on the user's activities conducted, other factors such as educational backgrounds, expertises, previous experiences are not considered. A possible direction would be to extend the data model and provide the capability for storing such information and take into account these factors using Analytic Hierarchy Process [111]. Analytic Hierarchy Process (AHP) is a formal method for multi-criteria decision problems. In AHP, factors are arranged in a hierarchy to determine their priorities and impacts on the decision making process. Factors that share common characteristics or are considered to be more important are represented at the higher levels while others are nodes further down the tree. A possible future direction is to explore the possibility of adopting AHP to combine various factors that are currently not considered in the model.

- **Supporting alternative knowledge-base**

The current system uses WordNet as its knowledge-base, but it is possible to use alternative repository such as DBpedia. In this case, a universal connector should be developed to allow the system to use various knowledge-bases. Investigating the feasibility of generating machine-understandable annotated data through Natural Language Process (NLP) is another possible direction for future research.

- **Supporting new OWL 2 constructs in DOTL**

Regarding Database to Ontology Transformation Language (DOTL) discussed in section 3.2, a few things could be done in the future:

- The current version of DOTL was intended for OWL1. The next step would be to extend DOTL to support OWL 2 features and Profiles such as OWL

QL, OWL2 EL, OWL2 RL. For example, to support more property and class constraints, anonymous classes, property qualified cardinality restrictions, reflexive, irreflexive, and asymmetric properties, property chain inclusion and simple meta-modeling capabilities.

- Another possible direction will be to develop a DSL (Domain Specific Language) for mapping by extending the standard OWL Functional Syntax/OWL Manchester Syntax. In terms of implementation, the underlying techniques used for ORM (Object-Relation Mapping) could be vastly improved by using latest JPA (Java Persistence API).

- **Refining graph-based query and reasoning techniques**

The proposed graph-based query and reasoning technique still have some restrictions. Some of the limitations are due to the limited expressivity of the underlying query language like predicates cannot have additional properties attached to it. But there are several possible directions for further work without extending the SPARQL specification. For example:

- (1) Defining additional notations to support more SPARQL 1.1 constructs and aggregation functions. SPARQL ASK and CONSTRUCT queries are currently not supported, and it provides supports for a limited number of aggregate functions (e.g. COUNT). The model could be extended to support them as well as more complex query statements such as nested WHERE clause, LIMIT, OFFSET, etc.
- (2) Extending the current specification to represent graph operator OPTIONAL with a different type of edge (currently only UNION operator is supported).
- (3) When there are alternatives in a pattern (e.g. when using UNION operator), solutions from multiple possibilities need to be combined. In this case, the process to specify the relative importance of triples is tedious, and the strategy to calculate topology-based weightings becomes very complicated.



- (4) Currently the graph-based deductive rules do not natively support probabilistic reasoning while it is possible to add certainty factors to rules and adopt a probabilistic approach in this case.
- (5) The current collaborative search method mainly uses the rational model in the collaborative decision making process, everyone in the group has equal contribution when building a query collectively. However, in the real world, this might be affected by the organisational structure of the team, and members of the group do not always share a common objective, some members might try to persuade others to adopt their viewpoints. It is known as "political model". An interest direction would be to see how political model can be adopted.
- (6) Investigating the feasibility of implementing a tableau-based satisfiability algorithm that supports uncertainty-credibility factors.

- **Improving collaborative classification approach**

The evaluation result in section 6.5.1 shows that the proposed semi-automated collaborative classification approach demonstrated reasonable performance. There are several possible directions to extend the research presented in this thesis:

- (1) Although the result of unsupervised clustering in the experiment has not achieved the expected results, one future research direction would be to investigate the possibility of combining Lexical database and manual classification to generate sensible category hierarchy. Despite the fact that the proposed DBSCAN algorithm did not generate desirable category in this particular case, it has the potential for further development in many other areas. It is worth looking at the feasibility of combining the semantic distance measurement method with other unsupervised classification approaches such as Support Vector Machine (SVM). For example, to help the researchers in social science, a project has been proposed to develop a component for monitoring and analysing news. Another direction is to develop a platform for

large-scale tracking and the analysis of real-time twitter data containing specific keywords or hashtags. The platform should provide ongoing monitoring of tweets. The methods discussed in section 5.1 will be used for text mining. The author has been in contact with Dr Veltri Giuseppe from the Department of Media and Communication, University of Leicester to discuss a possible grant proposal.

- (2) The current categorisation approach we used returns a ranked list of the possible root-to-leaf paths with the highest probability at the top. For this reason, the nodes closer to the root usually have heavier weights but they are the most general concepts. The proposed classification method narrows down the catalogue and returns the most desirable root-to-leave path but in practical, it does not necessarily mean the path should always end at the lowest level in the hierarchy.

### 7.2.1 Application: Mobile Social Annotation for Museums

A prototype social annotation app for museum collections was developed as part of this research. Some experiments have been carried out through lots of work still needs to be done.

Museums have evolved and inevitably entered the digital age as new technologies have been used in various ways within culture and heritage sector in the last decade. Multimedia exhibitions have turned many museums into more accessible and interactive places for general public. While providing a great experience in sharing knowledge, offering more informative tours to visitors, most interactive museum exhibitions were essentially one-way information channels. Museums use guest book to collect feedback but messages written in the guest book were usually not specific to an object. There are insufficient social interactions among visitors and also there are very limited ways to collect visitor feedback. Even such channels exist, people find it very hard to make use of these data.

An ontology derived from CCOnt (section 5.2.2) was used in conjunction with smartphones and QR codes [110], to provide a structured data model for visitors to ex-

press their opinions, share knowledge and take part in scientific surveys by categorising artefacts according to their understanding. Information obtained can be used to help museum curators categorise collections. The model also offers the possibility to record different interpretations regarding artefacts and allow tracking of visitor movement and behavioural patterns.

On 23 April 2013, an experiment was carried out at the British Academy (London) [11]. The majority of the visitors coming to this exhibition were archaeologists and archaeological enthusiasts. QR codes were attached to all objects, and visitors use a mobile web application to annotate the artefacts in the exhibition. The system allows browsing of additional information of the object by scanning the QR code attached, and it lets users to annotate and classify objects with a pre-defined ontology-based vocabulary. The mobile application is web-based, implemented in JQueryMobile which works on most mainstream mobile platforms, so visitors do not need to download or install any apps before their visits. When a visitor scans the QR code, a page will display some general information about the object. The visitor can then decide whether or not to take part in a scientific survey. For each artefact, the mobile app asked visitors for their opinions on which category the object belongs to, in the form of multiple choice question - a typical question would be to ask users to classify the clay-based object (Figure 7.1) based on its function. We give visitors an opportunity to categorise objects as well as allow them to indicate relevant attributes that were used as evidence to support the judgment (e.g. this item turns out to be a pyramid loom weight used for textile weaving, the conclusion was made based on its shape and the position of holes, etc.) We introduced a slider so that visitors can express the degree of uncertainty by adjusting the certainty factor associated with the choice. Data collected are used to train the classifier, and then we applied the classifier to other museum collections to see if the system produces reasonable classification results. Also, two tablet kiosks were set up in the exhibition hall for visitors without smartphones. The mobile app on has aroused visitors' interest, and many believed this app has a great potential, which could bridge the gap between visitors and archaeologists. It could help museum curators to categorise collections which were not on display. One of the future tasks would be to continue developing this app to explore other areas for any potential use.

### 7.2.2 Application: Graphical Modelling Tool for DOTL

A more intuitive visual editor for DOTL transformation language can be developed using Eclipse GMF (Eclipse Graphical Modeling Framework). The idea is compile current DOTL grammar to generate EMF metamodel for DOTL, then link such EMF (Eclipse Modelling Framework) metamodel to a GMF model. The aim of this project is to develop a visual editor as Eclipse plugin for DOTL. The editor should allow users to defined RDB to Ontology mapping graphically as opposed to current text-based rules. The editor will integrate some new features from [33]. For example, one of the future tasks would be to incorporate HQL (Hibernate Query Language) into DOTL let users create a highly customised logical view from some Java objects. Embedding an object-oriented query language into DOTL will make the transformation more easily tailored to needs.

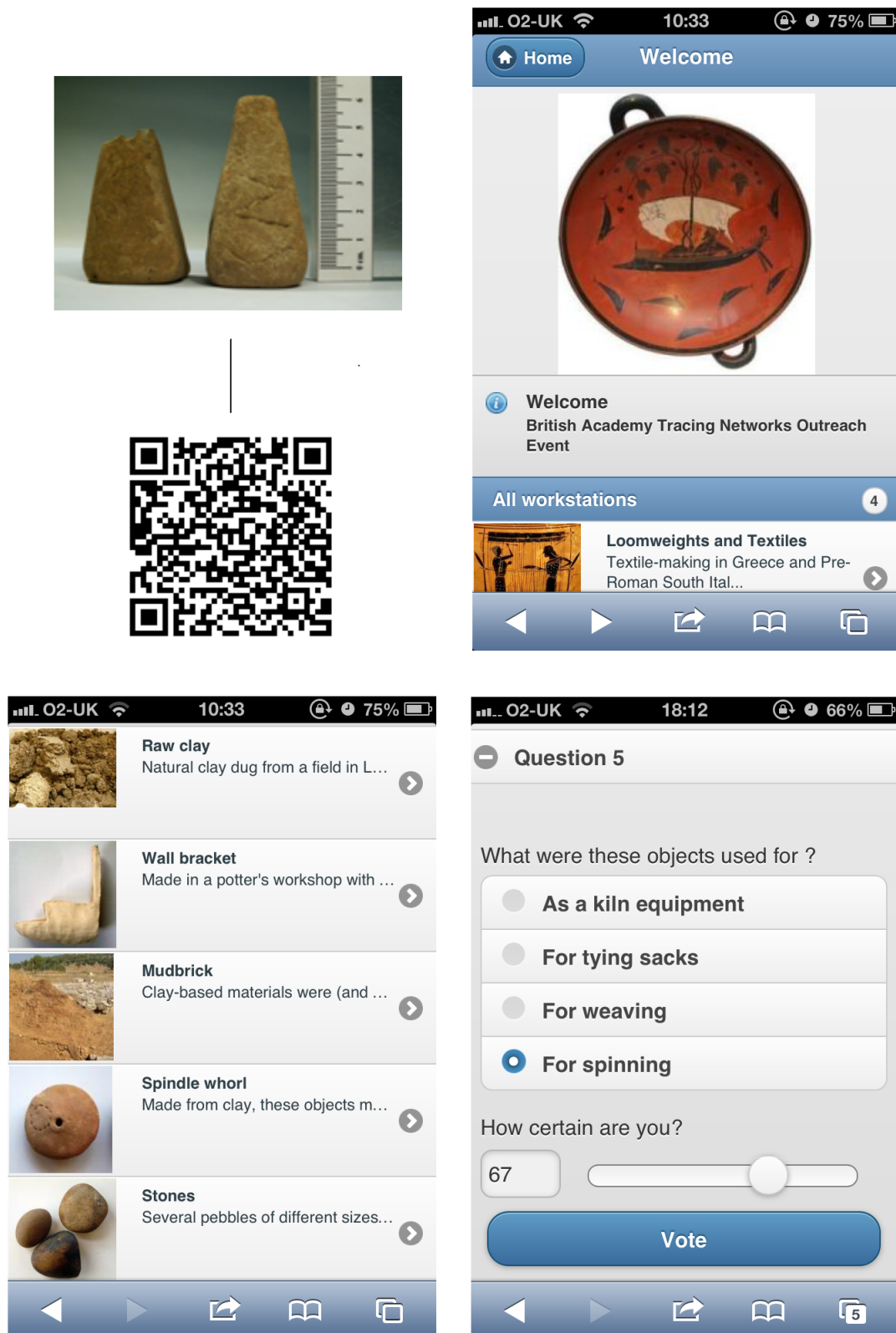


Figure 7.1: British Academy Exhibition: QR code and web-based smartphone app for collaborative object categorisation



# Acknowledgement

I would like to thank my supervisor Dr. **Stephan Reiff-Marganiec** for his continual support and the advices he has given me in the writing of this report.

I wish to personally thank the following for their contributions to my inspiration and knowledge and support in developing this work:

## **Tracing Networks Programme members:**

### *Department of Computer Science*

- José Fiadeiro
- Emilio Tuosto
- Laura Bocchi
- Effie Lai-Chong Law
- Monika Solanki

### *School of Archaeology and Ancient History*

- Pim Allison
- Elisa Alonso Lopez
- Colin Haselgrove
- Alessandro Quercia
- Katharina Rebay-Salisbury
- Sara Strack
- Andrea Roppa
- Leo Webley

- Ian Whitbread

*Department of Museum Studies*

- Ann Brysbaert

- Melissa Veters

*University of Liverpool*

- Lin Foxhall

*University of Exeter*

- Anthony Harding

- Marion Uckelmann

*Brown University*

- Peter van Dommelen



# Publications

- [1] Y. Hong and S. Reiff-Marganiec. Towards a collaborative framework for image annotation and search. In Proceedings of Advanced Information Systems Engineering Workshops CAiSE 2011, pages 564–574. Springer LNBIP, 2011.
- [2] Y. Hong and M.Solanki. "SEA: A Framework for Interactive Querying, Visualisation and Statistical Analysis of Linked Archaeological Datasets" In Proceeding of CAA the 39th Annual Conference of Computer Applications and Quantitative Methods in Archaeology, 2011
- [3] K. T. Pathan, S. Reiff-Marganiec and Y. Hong. Mapping for Software Sensors in the Context-aware Systems Environment. PICom 2011 (9th IEEE International Conference on Pervasive Intelligence and Computing). IEEE, 2011
- [4] Yi Hong, Monika Solanki, Lin Foxhall, and Alessandro Quercia. Fusion of Cultures. A Framework for Transforming Archaeological Databases to Linked Ontological Datasets. In Proceedings of the 38th Annual Conference on Computer Applications and Quantitative Methods in Archaeology, Granada, Spain, April 2010 edited by F. Contreras, M. Farjas and F.J. Melero. ISBN 9781407311081.
- [5] H.Q. Yu, Y. Hong, "Graph Transformation for the Semantic Web: Queries and Inference rules", 4th International Conference on Graph Transformation, Doctoral Symposium Section, LNCS (2008).
- [6] Context Aware Collaborative Working Environments. Reiff-Marganiec, S., Dustdar, S., Y. Hong., Yu, H.Q., and et al., In IK Ibrahim (ed): Handbook of Research on Mobile Multimedia: Second Edition. IGR. ISBN 978-1-60566-046-2. (2009)

[7] H. Truong, S. Dustdar, D. Baggio, H.Q. Yu, Y. Hong, et al., "inContext: a Pervasive and Collaborative Working Environment for Emerging Team Forms", The 2008 Symposium on Applications and the Internet SAINT-2008 Co-located with COMPSAC2008 IEEE-CS Conference on Computer Software and Applications , Turku, FINLAND, (2008).

[8] L. Bocchi, Y. Hong, A. Lopes, J. Fiadeiro From BPEL to SRML: a formal transformational approach. In: M. Dumas, R. Heckel (eds) Web Services and Formal Methods. LNCS, vol 4937. Springer, Berlin Heidelberg, pp 92–107 (2007)

[9] H.Q. Yu, Y. Hong, R. Heckel and S. Reiff-Marganiec, "Context-sensitive Team Formation: Towards Model-Based Context Reasoning and Update", In proceedings of Sixth International and Interdisciplinary Conference on Modeling and Using Context, Doctoral Symposium Section, Roskilde University, Denmark, (2007).

# Appendix

---

## A.1 Terminology for Archaeologists

### **Category**

A category (also called class) is a set of objects having some property or attribute in common and differentiated from others by kind, type or quality. In our definition, objects are organised into hierarchical categories in a tree structure, that means category can have sub-categories, which can have sub-sub categories. For example, category early 20th century might contains three sub-categories 1910s, 1920s and 1930s.

### **Object**

An uncategorised item that has certain data attribute values. An item can belong to more than one category. For example, a person can be classified as male by gender, and at the same time as British by nationality.

### **Attributes (Properties)**

An attribute takes a value and is associated with an object. A property is a member variable of a class. An attribute could be nominal or numeric. For example, gender and age are attributes of any living organisms, the former one is a nominal value while the later is a numeric value.

**Classification decision-maker**

An individual who groups objects into categories. The selection of categories involve multiple aspects and evaluation criteria, and these criteria might have different importance degrees based on the person's own viewpoints and perspectives.

## A.2 Tracing Networks Ontology (OWL-DL)

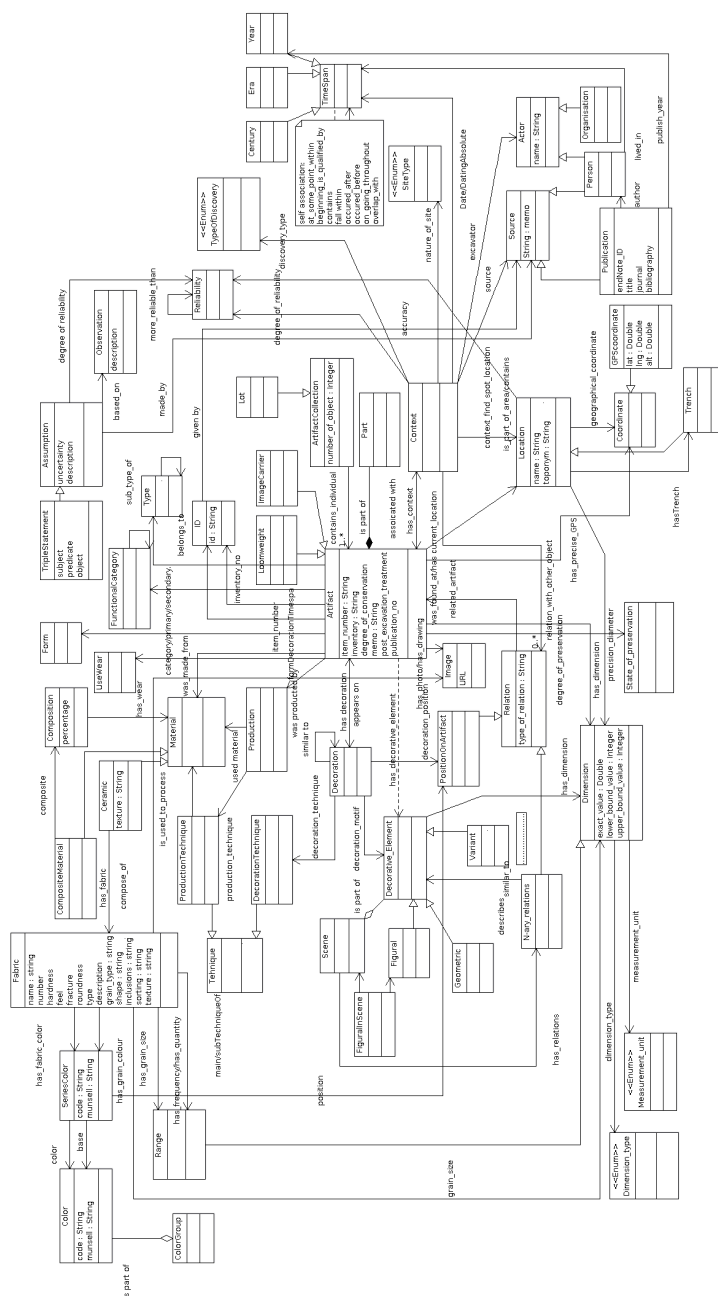


Figure A.1: Tracing Networks Ontology (OWL-DL)

### A.3 Worked Example of Credibility Measurement

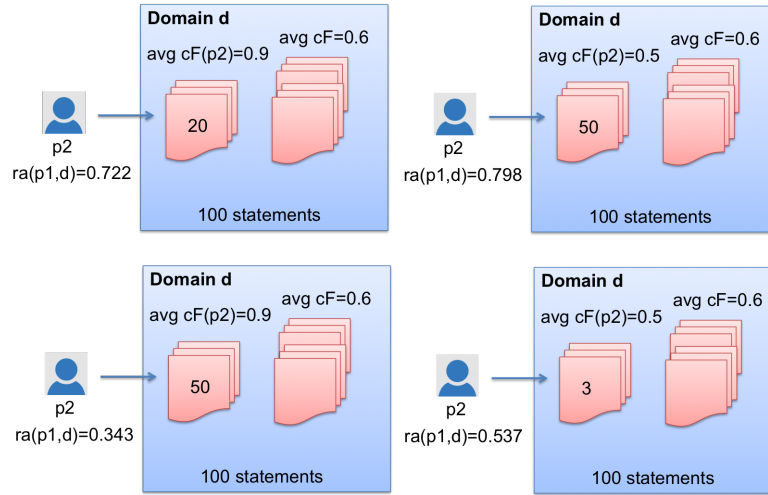


Figure A.2: Worked Example of Credibility Measurement (User-Activity based)

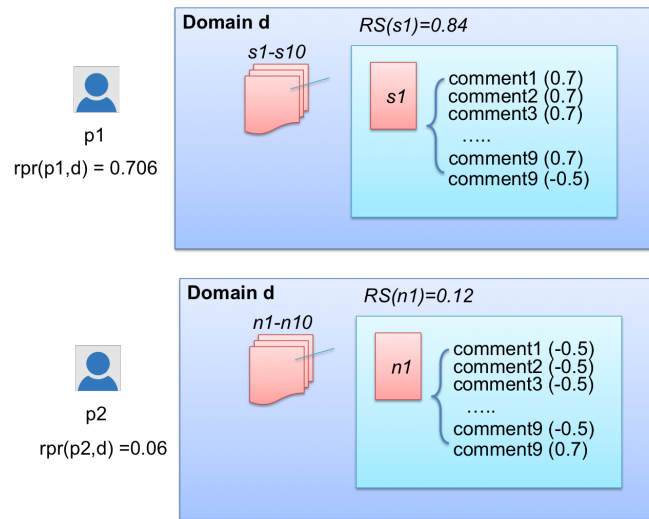


Figure A.3: Worked Example of Credibility Measurement (Peer-view based)

## A.4 Artifact Gallery



Figure A.4: A two-hole loomweight with imprinted meander and wave design.





# Bibliography

- [1] Yi Hong and Stephan Reiff-Marganiec. Towards a Collaborative Framework for Image Annotation and Search. In Camille Salinesi and Oscar Pastor, editors, *Advanced Information Systems Engineering Workshops*, volume 83 of *Lecture Notes in Business Information Processing*, pages 564–574. Springer Berlin Heidelberg, 2011.
- [2] Yi Hong, Monika Solanki, Alessandro Quercia, and Lin Foxhall. *Fusion of Cultures. Proceedings of the 38th Annual Conference on Computer Applications and Quantitative Methods in Archaeology (CAA2010), Granada, Spain, April 2010*. Archaeopress, 2010. ISBN 9781407311081.
- [3] Hong Qing Yu and Yi Hong. Graph Transformation for the Semantic Web: Queries and Inference Rules. In *Proceedings of the 4th international conference on Graph Transformations*, ICGT '08, pages 511–513, Berlin, Heidelberg, 2008. Springer-Verlag.
- [4] Hong Qing Yu, Yi Hong, Reiko Heckel, and Stephan Reiff Marganiec. Proceedings of CONTEXT 2007, context-sensitive Team Formation: Towards Model-based Context Reasoning and Update, 2007.
- [5] Yi Hong and Monika Solanki. SEA: A Framework for Interactive Querying, Visualisation and Statistical Analysis of Linked Archaeological Datasets. In *39th Annual Conference on Computer Applications and Quantitative Methods in Archaeology*, 2011.
- [6] In *Tracing Networks: Investigating Networks of Knowledge in Antiquity and the Digital Age*. Oxford Press, 2016.
- [7] Digital Humanities Colloquium. `sws.geonames.org`, July 2011.
- [8] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, May 2001.

- [9] Web Ontology Language. [www.w3.org/TR/owl-guide/](http://www.w3.org/TR/owl-guide/), December 2013.
- [10] Yi Hong, Stephan Reiff-Marganiec, Katharina Rebay-Salisbury, and Lin Foxhall. Truthfulness and reliability in collaborative image annotation. In *Digital Economy Conference (poster)*. Newcastle, UK, 2011.
- [11] British Academy Exhibition - Networking: Past and Present. [http://www.britac.ac.uk/events/2013/Networking\\_past\\_and\\_present.cfm](http://www.britac.ac.uk/events/2013/Networking_past_and_present.cfm), 2013.
- [12] Tracing Networks: Craft Traditions in the Ancient Mediterranean and Beyond. [www.tracingnetworks.ac.uk](http://www.tracingnetworks.ac.uk), August 2009.
- [13] Hak-Lae Kim, Simon Scerri, John Breslin, Stefan Decker, and Hong-Gee Kim. The State of the Art in Tag Ontologies: A Semantic Model for Tagging and Folksonomies. In *International Conference on Dublin Core and Metadata Applications*, Berlin, Germany, 2008.
- [14] Andrea Marchetti, Maurizio Tesconi, Francesco Ronzano, Marco Rosella, and Salvatore Minutoli. Semkey: A semantic collaborative tagging system. In *Proc. WWW 2007 Workshop on Tagging and Metadata for Social Information Organization*, Banff, Canada, May 2007.
- [15] Lawrence Reeve. Survey of Semantic Annotation Platforms. In *Proceedings of the 2005 ACM Symposium on Applied Computing*, pages 1634–1638. ACM Press, 2005.
- [16] S. H. Walker and D. B. Duncan. Estimation of the probability of an event as a function of several independent variables. *Biometrika*, 54(1):167–179, June 1967.
- [17] Ann Brysbaert. Cross-craft interaction in the cross-cultural context of the late bronze age east mediterranean. <http://www.universiteitleiden.nl/en/research/research-projects/archaeology>, 2014.
- [18] S. B. Kotsiantis. Supervised Machine Learning: A Review of Classification Techniques. *Informatika* 31:249–268, 2007.

- [19] Alexander Budanitsky and Graeme Hirst. Evaluating WordNet-based Measures of Lexical Semantic Relatedness. *Comput. Linguist.*, 32(1):13–47, March 2006.
- [20] Giuseppe De Giacomo and Maurizio Lenzerini. TBox and ABox Reasoning in Expressive Description Logics. In *In Proc. of KR-96*, pages 316–327. Morgan Kaufmann, 1996.
- [21] F. Baader. Description Logic Terminology. In Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors, *The Description Logic Handbook: Theory, Implementation, and Applications*, pages 485–495. Cambridge University Press, 2003.
- [22] Sofia Angeletou, Marta Sabou, Lucia Specia, and Enrico Motta. Bridging the gap between folksonomies and the semantic web: An experience report. In *Bridging the Gap between Semantic Web and Web 2.0 (SemNet 2007)*, pages 30–43, 2007.
- [23] Cohere: A prototype for contested collective intelligence, year=2012. <http://cohere.open.ac.uk>.
- [24] Anna De Liddo and Simon Buckingham Shum. The evidence hub: harnessing the collective intelligence of communities to build evidence-based knowledge. In *Large Scale Ideation and Deliberation Workshop*, 2013. 6th International Conference on Communities and Technologies.
- [25] Maria Maleshkova, Carlos Pedrinaci, and John Domingue. SWEET: Supporting the creation of semantic RESTful service descriptions. In *8th International Semantic Web Conference (ISWC 2009)*, 2009. Proceedings of the 3rd International SMR2 2009 Workshop on Service Matchmaking and Resource Retrieval in the Semantic Web, collocated with the 8th International Semantic Web Conference (ISWC-2009), Washington DC, USA, October 25, 2009. Edited by Alain Léger, Tiziana Margaria, David Martin, Massimo Paolucci CEUR Workshop Proceedings, vol.525.

- [26] Arthur Stuttt Jitu Patel. Beyond classification: the use of artificial intelligence techniques for the interpretation of archaeological data. In *1989 Annual Conference on Computer Applications and Quantitative Methods in Archaeology*, 1989.
- [27] Zooniverse: People-Powered Research. <https://www.zooniverse.org/>.
- [28] J. Domingue and E. Motta. Planetonto: from news publishing to integrated knowledge management support. *IEEE Intelligent Systems and their Applications, IEEE*, 15(3):26–32, May 2000.
- [29] OCML: Operational Conceptual Modelling Language. <http://technologies.kmi.open.ac.uk/ocml/>.
- [30] Aqualog: a potable question-answering system based on NLP and ontology. <http://technologies.kmi.open.ac.uk/aqualog/>.
- [31] Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, Luciano Serafini, and Heiner Stuckenschmidt. *Contextualizing Ontologies*, pages 164–179. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.
- [32] A direct mapping of relational data to rdf, w3c recommendation 27 september 2012. <https://www.w3.org/TR/rdb-direct-mapping/>, 2013.
- [33] R2rml: Rdb to rdf mapping language, w3c recommendation 27 september 2012. <https://www.w3.org/TR/r2rml/>, 2013.
- [34] C.Timurhan Sungur, Christoph Dorn, Schahram Dustdar, and Frank Leymann. Transforming collaboration structures into deployable informal processes. In Philipp Cimiano, Flavius Frasincar, Geert-Jan Houben, and Daniel Schwabe, editors, *Engineering the Web in the Big Data Era*, volume 9114 of *Lecture Notes in Computer Science*, pages 231–250. Springer International Publishing, 2015.
- [35] Trena M. Paulus \*. Collaborative and Cooperative Approaches to Online Group Work: The Impact of Task Type. *Distance Education*, 26(1):111–125, 2005.
- [36] Hong Qing Yu, Yi Hong, Reiko Heckel, and Stephan Reiff-Marganiec. Context-sensitive team formation: Towards model- based context reasoning and update.

- [37] Boris Katz, Gary C. Borchardt, and Sue Felshin. Natural Language Annotations for Question Answering. In *FLAIRS Conference*, pages 303–306, 2006.
- [38] RDF - Resource Description Framework. <http://http://www.w3.org/RDF/>, Feb 2004.
- [39] OWL - Resource Description Framework. <http://www.w3.org/TR/owl-features/>, Feb 2004.
- [40] M. Doerr. The CIDOC CRM - An Ontological Approach to Semantic Interoperability of Metadata. *AI Magazine*, 24:2003, 2003.
- [41] J. David Schloen. Archaeological data models and web publication using xml. *Computers and the Humanities*, 35(2):123–152, 2001.
- [42] J.D. Richards. Archaeology, e-publication and the semantic web. *Antiquity*, 80(310):970–979, 2006.
- [43] Michael K. Bergman. White Paper: The Deep Web. Surfacing Hidden Value. *The Journal of Electronic Publishing*, 7(1):online, Aug. 2001.
- [44] Linked Data : Connect Distributed Data Across the Web. [www.linkeddata.org](http://www.linkeddata.org), August 2009.
- [45] Satya S. Sahoo, Wolfgang Halb, Sebastian Hellmann, Kingsley Idehen, Ted Thibodeau Jr, Sören Auer, Juan Sequeda, and Ahmed Ezzat. A Survey of Current Approaches for Mapping of Relational Databases to RDF, 01 2009.
- [46] Christian Bizer and Andy Seaborne. D2RQ - Treating Non-RDF Databases as Virtual RDF Graphs. In *ISWC2004 (posters)*, November 2004.
- [47] Orri Erling and Ivan Mikhailov. RDF support in the Virtuoso DBMS. volume P-113 of *GI-Edition - Lecture Notes in Informatics (LNI)*, ISSN 1617-5468. Bonner Köllen Verlag, September 2007.
- [48] Jesús Barrasa, Óscar Corcho, and Asunción Gómez-pérez. R2O, an ‘Extensible and Semantically based Database-to-Ontology Mapping Language. In *in In*

- Proceedings of the 2nd Workshop on Semantic Web and Databases(SWDB2004)*, pages 1069–1070. Springer, 2004.
- [49] Ceri Binding, Keith May, and Douglas Tudhope. Semantic interoperability in archaeological datasets: Data mapping and extraction via the cidoc crm. In Birte Christensen-Dalsgaard, Donatella Castelli, Bolette Ammitzbøll Jurik, and Joan Lippincott, editors, *ECDL*, volume 5173 of *Lecture Notes in Computer Science*, pages 280–290. Springer, 2008.
- [50] Leif Isaksen. De-engineering the Semantic Web: Linking Archaeological Data. <http://www.digitalclassicist.org/wip/wip2009-08li.html>, July 2009.
- [51] Ora Lassila, Ralph R. Swick, World Wide, and Web Consortium. Resource Description Framework (RDF) Model and Syntax Specification, 1998.
- [52] James Bailey, Alexandra Poulovassilis, and Peter T. Wood. An event-condition-action language for xml. In David Lassner, Dave De Roure, and Arun Iyengar, editors, *WWW*, pages 486–495. ACM, 2002.
- [53] Red Hat Middleware: Relational Persistence for Java and .NET. [www.hibernate.org](http://www.hibernate.org), August 2009.
- [54] DBpedia: Extract structured content from the information created as part of the wikipedia project. <http://www.dbpedia.org>, August 2013.
- [55] Geoname. <http://sws.geonames.org>, August 2013.
- [56] Google Patents US 6208339 B1- User-Interactive Data Entry Display System with Entry Fields Having Distinctive and Changeable Autocomplete . <https://www.google.com/patents/US6208339>, 2003.
- [57] Uncertainty Reasoning for the World Wide Web. <http://www.w3.org/2005/Incubator/urw3/XGR-urw3-20080331/>, March 2008.
- [58] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory*,

- Implementation, and Applications*. Cambridge University Press, New York, NY, USA, 2003.
- [59] Umberto Straccia. A Fuzzy Description Logic for the Semantic Web. In *FUZZY LOGIC AND THE SEMANTIC WEB, CAPTURING INTELLIGENCE, CHAPTER 4*, pages 167–181. Elsevier, 2005.
- [60] Paulo Cesar G. da Costa, Kathryn B. Laskey, and Kenneth J. Laskey. PR-OWL: A Bayesian Ontology Language for the Semantic Web. In Paulo Cesar G. da Costa, Claudia d’Amato, Nicola Fanizzi, Kathryn B. Laskey, Kenneth J. Laskey, Thomas Lukasiewicz, Matthias Nickles, and Michael Pool, editors, *Uncertainty Reasoning for the Semantic Web I*, volume 5327 of *Lecture Notes in Computer Science*, pages 88–107. Springer Berlin Heidelberg, 2008.
- [61] Jordi Sabater and Carles Sierra. Review on Computational Trust and Reputation Models. *Artif. Intell. Rev.*, 24(1):33–60, September 2005.
- [62] Franz Baader and Ulrike Sattler. Tableau Algorithms for Description Logics. *Studia Logica*, 69:2001, 2000.
- [63] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995.
- [64] Thorsten Joachims. Text Categorization with Support Vector Machines: Learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Machine Learning: ECML-98*, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142. Springer Berlin Heidelberg, 1998.
- [65] Corinna Cortes and Vladimir Vapnik. Support-Vector Networks. *Machine Learning*, 20(3):273–297, 1995.
- [66] Michael M. Stark and Richard F. Riesenfeld. WordNet: An Electronic Lexical Database. In *Proceedings of 11th Eurographics Workshop on Rendering*. MIT Press, 1998.
- [67] David E. Heckerman and Edward H. Shortliffe. From Certainty Factors to Belief Networks. *Artificial Intelligence in Medicine*, 4(1):35 – 52, 1992.

- [68] Anna Gutowska and Kevan Buckley. A computational distributed reputation model for b2c e-commerce. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 03*, pages 72–76, Washington, DC, USA, 2008. IEEE Computer Society.
- [69] Jinhyung Cho, Kwiseok Kwon, and Yongtae Park. Q-rater: A collaborative reputation system based on source credibility theory. *Expert Systems with Applications*, 36(2, Part 2):3751–3760, 2009.
- [70] Giovanni Sartori. The theory of democracy revisited. 1987.
- [71] Susan M. Mudambi and David Schuff. What makes a helpful online review? a study of customer reviews on amazon.com. *MIS Q.*, 34(1):185–200, March 2010.
- [72] The D2RQ Platform - Treating Non-RDF Databases as Virtual RDF Graphs. <http://www4.wiwiiss.fu-berlin.de/bizer/d2rq/>, August 2009.
- [73] Justas Trinkunas and Olegas Vasilecas. In *CompSysTech '07: Proceedings of the 2007 international conference on Computer systems and technologies*, 2007.
- [74] Openlink Virtuoso. <http://sourceforge.net/projects/virtuoso/>, August 2006.
- [75] Richard C. Gronback. *Eclipse Modeling Project: A Domain-Specific Language (DSL) Toolkit*. Addison-Wesley Professional, 1 edition, 2009.
- [76] yEd Graph Editor. <https://www.yworks.com/>, 2015.
- [77] SPARQL Query Language for RDF. <http://www.w3.org/TR/rdf-sparql-query>, Jan 2008.
- [78] Evren Sirin and Bijan Parsia. Sparql-dl: Sparql query for owl-dl. In *In 3rd OWL Experiences and Directions Workshop (OWLED-2007)*, 2007.
- [79] B. McBride. Jena: A semantic web toolkit. *IEEE Internet Computing*, 6(6):55–59, Nov 2002.



- [80] Sparql query results xml format (second edition), w3c recommendation 21 march 2013. <https://www.w3.org/TR/rdf-sparql-XMLres/>, 2014.
- [81] SWRL: A Semantic Web Rule Language Combining OWL and RuleML. <http://www.w3.org/Submission/SWRL/>, Jan 2008.
- [82] Jie Lu, Guangquan Zhang, and Da Ruan. *Multi-objective Group Decision Making: Methods, Software and Applications With Fuzzy Set Techniques*. Imperial College Press, London, UK, UK, 2007.
- [83] Oxford English Dictionary Online, 2nd edition. <http://www.oed.com/>, July 2003.
- [84] Edward L. Keenan and Bernard Comrie. Noun phrase accessibility and universal grammar. *Linguistic Inquiry*, 8(1):pp. 63–99, 1977.
- [85] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117, April 1998.
- [86] Claudia Leacock, George A. Miller, and Martin Chodorow. Using corpus statistics and wordnet relations for sense identification. *Comput. Linguist.*, 24(1):147–165, March 1998.
- [87] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244, 1990.
- [88] Hans-Peter Kriegel and Martin Pfeifle. Density-based clustering of ain data. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 672–677, New York, NY, USA, 2005. ACM.
- [89] Martin Ester, Hans peter Kriegel, Jörg S, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

- [90] Collaborative Protege Editor. <http://protegewiki.stanford.edu/wiki/Collaborative> March 2013.
- [91] Bruce L Golden, Edward A Wasil, and Patrick T Harker. *Analytic hierarchy process*, volume 113. Springer, 2003.
- [92] Paul K. Yoon, Ching-Lai Hwang, and Kwangsun Yoon. *Multiple Attribute Decision Making: An Introduction (Quantitative Applications in the Social Sciences)*. Sage Pubn Inc, March 1995.
- [93] Thorsten Joachims. *Learning to Classify Text Using Support Vector Machines. Methods, Theory and Algorithms*, volume 668 of *The Springer International Series in Engineering and Computer Science*. Springer, 2002.
- [94] David Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In Claire Nédellec and Céline Rouveirol, editors, *Machine Learning: ECML-98*, volume 1398 of *Lecture Notes in Computer Science*, pages 4–15. Springer Berlin / Heidelberg, 1998. 10.1007/BFb0026666.
- [95] George H. John and Pat Langley. Estimating continuous distributions in bayesian classifiers. pages 338–345, 1995.
- [96] Jan-Marco Bremer and Michael Gertz. Xquery/ir: Integrating xml document and data retrieval. In *In Proceedings of the 5th International Workshop on the Web and Databases (WebDB)*, pages 1–6, 2002.
- [97] Full-Text Search and Semantic Search Functions (Transact-SQL). <https://msdn.microsoft.com/en-us/library/ms189760.aspx>, 2014.
- [98] M. J. O'Connor and A. K. Das. SQWRL: a Query Language for OWL. In *OWL: Experiences and Directions (OWLED), Fifth International Workshop*, 2009.
- [99] Songling Liu, J.P. Cedenro, K.S. Candan, M.L. Sapino, Shengyu Huang, and Xinsheng Li. R2db: A system for querying and visualizing weighted rdf graphs. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pages 1313–1316, April 2012.

- 
- [100] Transact-SQL Reference (Database Engine). <https://msdn.microsoft.com/en-us/library/bb510741.aspx>, 2014.
- [101] Protege User Guide: DL Query). <http://protegewiki.stanford.edu/wiki/DLQueryTab>, 2014.
- [102] Juan P. Cedeño and K. Selçuk Candan. R2df framework for ranked path queries over weighted rdf graphs. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*, WIMS '11, pages 40:1–40:12, New York, NY, USA, 2011. ACM.
- [103] Calais: Connect Everything. <http://www.opencalais.com/>, 2011.
- [104] University of Leicester, School of Archeology and Ancient History, Staff list). <http://www2.le.ac.uk/departments/archaeology/people/quercia>, 2016.
- [105] Rebay-Salisbury K. Hong Y. et al. Foxhall, L. Tracing Networks: Technological Knowledge, Cultural Contact and Knowledge Exchange in the Ancient Mediterranean and Beyond. *New Worlds from Old Texts*, pages 281–300.
- [106] Rebay-Salisbury K. Hong Y. et al. Foxhall, L. Tracing Networks Annual Report (2013). Technical report, 2013.
- [107] AHRC Research Network workshop: Big Data on the Roman Table (27 Sep 2015). <http://www2.le.ac.uk/departments/archaeology/people/academics/allison/research/big-data-roman-table/Workshop->.
- [108] Katharina Rebay-Salisbury, Institut für Orientalische und Europäische Archäologie). <http://www.orea.oeaw.ac.at/rebay-salisbury.html>, 2016.
- [109] Tracing Networks: Translating art and craft: Human representations, identities and social relations in the Late Bronze and Iron Age of Central Europe. [http://www.tracingnetworks.ac.uk/content/web/human\\_representations.jsp](http://www.tracingnetworks.ac.uk/content/web/human_representations.jsp).

- [110] Technical report: QR Code Essentials. <http://www.nacs.org/>, 2011.
- [111] T.L. Saaty. How to make a decision: the analytic hierarchy process. *European journal of operational research*, 48(1):9–26, 1990.