

University of Crete
Computer Science Department

Computer Supported Collaborative Factual
Argumentation and Conflict Resolution

Boutsika Katerina
Master's Thesis

Heraklion, October 2010

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

Σύστημα Συνεργατικής Επιχειρηματολογίας περί Γεγονότων και
Επίλυσης Αντιφάσεων

Εργασία που υποβλήθηκε από την

Κατερίνα Κων. Μπούτσινα

ως μερική εκπλήρωση των απαιτήσεων για την απόκτηση
ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΕΙΔΙΚΕΥΣΗΣ

Συγγραφέας:

Μπούτσινα Κατερίνα, Τμήμα Επιστήμης Υπολογιστών

Εισηγητική Επιτροπή:

Γρηγόρης Αντωνίου, Καθηγητής, Επόπτης

Martin Doerr, Ερευνητής, Μέλος

Δημήτρης Πλεξουσάκης, Καθηγητής, Μέλος

Δεκτή:

Πάνος Τραχανιάς, Καθηγητής
Πρόεδρος Επιτροπής Μεταπτυχιακών Σπουδών
Ηράκλειο, Οκτώβριος 2010

Computer Supported Collaborative Factual Argumentation and Conflict Resolution

Boutsika Katerina

Master's Thesis

Computer Science Department, University of Crete

Abstract

Argumentation is an important human activity concerning many aspects of human life and society such as politics, law, scientific and scholarly discourse. Nowadays, more and more people communicate and publish arguments via the Internet. For this reason, it is of vital importance to build information systems that manage, structure and fully understand human argumentation processes. In recent years, significant efforts have been made in connecting Information Technology with argumentation. However, current argumentation systems are based on models that are either not analytical enough or restricted to formal logic. So far, none of these models connects argumentation with a domain ontology. A small minority of these systems use Semantic Web technologies for analyzing structuring and representing arguments. Furthermore, most of these models blur the logical structure of a composite argument that is believed at some point in time with the temporal order of arguments in the argumentation process.

In this thesis we describe an integrated model for human argumentation in which reasoning may not only consist of falsification or verification but more generally of strengthening or weakening hypotheses. The model includes evolution, composition and revision of arguments and can be connected to a domain ontology. We explain the extensions that were made in order to specialize this model in factual argumentation, i.e., in arguments and counterarguments about propositions concerning material states of affairs in the past. Also, we introduce the extensions that were made in the model in order to be able to defeat arguments that cause inconsistencies. Next, we describe the implementation of the model using the Resource Description Framework Schema (RDFS) semantic language. We present Antilogos, a Semantic Web-based argumentation system for the record and representation of dynamic arguments networks. The aim of the system is to inform the users about the provenance of the registered knowledge in a structured way. Moreover, the system is able

to detect and resolve conflicts between the arguments. Antilogos enables users to create new arguments and also to support, defeat and search propositions of existing arguments. Through Antilogos, users can monitor any state of the knowledge through time and can also understand how this knowledge came up or how was changed. Finally, we demonstrate the use of Antilogos by presenting arguments and counter-arguments of a real published archaeological case study.

Supervisor: Grigoris Antoniou

Professor

Σύστημα Συνεργατικής Επιχειρηματολογίας περί Γεγονότων και

Επίλυσης Αντιφάσεων

Μπούτσικα Κατερίνα

Μεταπτυχιακή Εργασία

Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

Περίληψη

Η επιχειρηματολογία είναι μια σημαντική ανθρώπινη δραστηριότητα που αφορά πολλούς τομείς της ζωής μας και της κοινωνίας μας όπως την πολιτική, τη δικηγορία, τις επιστημονικές και φιλοσοφικές συζητήσεις. Στις μέρες μας, όλο και περισσότεροι άνθρωποι επικοινωνούν και δημοσιεύουν επιχειρήματα μέσω του διαδικτύου. Για το λόγο αυτό, είναι ζωτικής σημασίας η ανάπτυξη πληροφοριακών συστημάτων που να διαχειρίζονται, να δομούν και να καταλαβαίνουν πλήρως την ανθρώπινη επιχειρηματολογία. Τα τελευταία χρόνια έχουν γίνει αξιόλογες προσπάθειες για να συνδεθεί η επιχειρηματολογία με τις τεχνολογίες των πληροφοριών. Παρ' όλα αυτά, τα τωρινά συστήματα επιχειρηματολογίας είναι βασισμένα σε μοντέλα που είτε δεν είναι αρκετά αναλυτικά είτε περιορίζονται στην τυπική λογική. Έως τώρα, κανένα από αυτά τα μοντέλα δεν συνδέει την επιχειρηματολογία με κάποια οντολογία που μοντελοποιεί έναν τομέα. Ελάχιστα από αυτά τα συστήματα χρησιμοποιούν τις τεχνολογίες του σημασιολογικού ιστού για να αναλύσουν, να δομήσουν και να αναπαραστήσουν επιχειρήματα. Επιπλέον, τα περισσότερα από τα μοντέλα συγχέουν την λογική δομή ενός σύνθετου επιχειρήματος το οποίο πιστεύουμε ότι ισχύει σε κάποια χρονική στιγμή με την χρονική σειρά των επιχειρημάτων στην διαδικασία της επιχειρηματολογίας.

Σε αυτή την εργασία περιγράφουμε ένα ολοκληρωμένο μοντέλο για ανθρώπινη επιχειρηματολογία στο οποίο ο συλλογισμός μπορεί να αποτελείται όχι μόνο από την διάψευση ή την εξακρίβωση αλλά γενικότερα από την ενδυνάμωση ή την αποδυνάμωση της υπόθεσης. Το μοντέλο περιλαμβάνει εξέλιξη, σύνθεση και αναθεώρηση επιχειρημάτων και επίσης μπορεί να συνδεθεί με μια οντολογία κάποιου τομέα. Εξηγούμε τις επεκτάσεις που κάναμε προκειμένου να εξειδικεύσουμε αυτό το μοντέλο στην αντικειμενική επιχειρηματολογία, δηλαδή, σε επιχειρήματα και αντεπιχειρήματα πάνω σε προτάσεις που αφορούν καταστάσεις πραγμάτων του συνέβησαν στον παρελθόν. Επιπρόσθετα, εξηγούμε τις επεκτάσεις που έγιναν στο μοντέλο ώστε να είναι σε θέση να ακυρώνει επιχειρήματα τα οποία προκαλούν ασυνέπειες. Στη συνέχεια, περιγράφουμε πως

έγινε η υλοποίηση του μοντέλου στη σημασιολογική γλώσσα RDFS. Παρουσιάζουμε το Antilogos, ένα σημασιολογικό σύστημα επιχειρηματολογίας για την καταγραφή και την αναπαράσταση δυναμικών δικτύων επιχειρημάτων. Σκοπός του συστήματος είναι να πληροφορήσει το χρήστη για την προέλευση της καταχωρημένης γνώσης με έναν δομημένο τρόπο. Επιπλέον, το σύστημα μπορεί να εντοπίζει και να λύνει τις ασυνέπειες μεταξύ των επιχειρημάτων. Το Antilogos επιτρέπει στους χρήστες να δημιουργούν νέα επιχειρήματα, και επίσης να υποστηρίζουν, να αμφισβητούν και να αναζητούν προτάσεις που βρίσκονται στα ήδη υπάρχοντα επιχειρήματα. Μέσω του Antilogos, οι χρήστες μπορούν να παρακολουθήσουν οποιαδήποτε κατάσταση γνώσης κατά τη διάρκεια του χρόνου και να καταλάβουν πώς προέκυψε αυτή η γνώση και πώς άλλαξε. Τέλος, επιδεικνύουμε τη χρήση του Αντιλογος παρουσιάζοντας επιχειρήματα και αντεπιχειρήματα από ένα αληθινό δημοσιευμένο αρχαιολογικό παράδειγμα.

Επόπτης Καθηγητής: Γρηγόρης Αντωνίου
Καθηγητής

Η εργασία αυτή είναι αφιερωμένη στην οικογένειά μου

Ευχαριστίες

Σε αυτό το σημείο θα ήθελα να ευχαριστήσω τον επόπτη μου, καθηγητή κ. Γρηγόρη Αντωνίου που μου έδωσε την ευκαιρία να ασχοληθώ με ένα επίκαιρο και πολύ ενδιαφέρον θέμα καθώς και για την ουσιαστική συμβολή του στην ολοκλήρωση της παρούσας εργασίας. Θέλω επίσης να ευχαριστήσω θερμά και τον δεύτερο επόπτη μου, ερευνητή κ. Martin Doerr, με τον οποίο είχαμε μία άψογη συνεργασία. Με την συνεχή στήριξη και την πολύτιμη επιστημονική του καθοδήγηση σε όλη την διάρκεια της εργασίας αυτής συνέβαλε ουσιαστικά στην ολοκλήρωσή της. Θα ήθελα ακόμα να ευχαριστήσω τον καθηγητή κ. Δημήτρη Πλεξουσάκη για την προθυμία του να συμμετάσχει στην εξεταστική επιτροπή. Επίσης, θέλω να ευχαριστήσω την αρχαιολόγο Αθηνά Κριτσωτάκη για τις συμβουλές και χρήσιμες παρατηρήσεις της.

Ένα μεγάλο ευχαριστώ θέλω να πω στους φίλους μου για τις ευχάριστες στιγμές που περάσαμε μαζί και για την ψυχολογική στήριξη που μου προσέφεραν ακόμη και όταν κάποιοι από αυτούς βρίσκονταν πολλά χιλιόμετρα μακριά! Ιδιαίτερα θέλω να ευχαριστήσω τους παλιούς μου φίλους Εύη, Θάνο, Δέσποινα, Γιώργο Κτιστάκη και Χρίστina, αλλά και τους νεότερους Στέλλα, Γιώργο Μπαργιάννη και Μαρία. Κυρίως όμως θέλω να ευχαριστώ τον Μύρωνα που με υπέφερε και που στάθηκε δίπλα μου σε κάθε όμορφη αλλά και δύσκολη στιγμή, βοηθώντας με να ξεπεράσω κάθε εμπόδιο.

Κλείνοντας, θέλω να εκφράσω την ευγνωμοσύνη μου στους γονείς μου Κώστα και Ντίνα και στον αδερφό μου Δημήτρη για την υπομονή και την ανεξάντλητη υποστήριξή τους όλα αυτά τα χρόνια. Χωρίς την αγάπη και την κατανόησή τους δεν θα είχα καταφέρει να εκπλήρωσω τους στόχους μου. Σας ευχαριστώ πολύ.

Contents

Table of Contents	iii
List of Figures	vi
1 Introduction	1
1.1 Introduction	1
1.2 The problem	2
1.3 Contribution of this thesis	3
1.4 Organization of this thesis	4
2 Related Work	7
2.1 Argumentation Models and Systems	7
2.1.1 Type-structure argumentation models and systems	7
2.1.2 Procedural argumentation models and systems	10
2.2 Summary and Discussion	16
3 An Integrated Model	19
3.1 Generic Concepts	19
3.2 The Model	22
3.2.1 Inference Making	22
3.2.2 Proposition Beliefs	23
3.2.3 Factual Inference Making	24
3.2.4 Inferences	24
3.2.5 Inference Logic	24

3.2.6	Inference Beliefs	25
3.2.7	Inference Defeating	25
3.2.8	Elementary and Composite Inferences	26
3.2.9	Recursive Inferences and Inconsistency	27
3.2.10	Observations and Belief Adoptions	29
3.3	Implementation of the Model in RDF	30
3.3.1	Background in XML, RDF and RDFS	30
3.3.2	Advantages of RDF versus XML	31
3.3.3	Implementation in RDF and RDFS	32
4	Antilogos: a system for factual argumentation based on IAM	35
4.1	Introduction	35
4.2	Assumptions	37
4.3	System Description	38
4.3.1	System Architecture	38
4.3.2	System Platform	40
4.3.2.1	Repository of the system	40
4.3.2.2	Query Language	41
4.3.2.3	Generation of webpages	42
4.4	System Components	43
4.4.1	Storage Component	43
4.4.2	Behavioral Component	43
4.4.2.1	Query Processor	44
4.4.2.2	RDF Generator	44
4.4.2.3	Composite Inference Maker	48
4.4.2.4	Inconsistency controller	53
4.4.3	User Interface	63
4.4.3.1	Register and Login	63
4.4.3.2	Create Issue	63
4.4.3.3	Create an Argument	64
4.4.3.4	Knowledge state through time	65

4.4.3.5	History of a proposition	66
4.4.3.6	Search for a proposition	67
4.5	An Example of use	67
5	Conclusions	83
	Appendix	89
A.1	RDF Source	89

List of Figures

2.1	Toulmin’s argumentation scheme [36]	8
2.2	Scientific document structure according to Logician/SCD format [10].	9
2.3	gIBIS interface ([13])	11
2.4	Araucaria Screenshot [30]	14
3.1	Types of argumentation: Factual Argumentation, Inference making, and Belief Adoption.	20
3.2	Inference Defeating, a special kind of Inference Making.	26
3.3	Elementary Inference Making	27
3.4	Composite Inference.	28
3.5	Belief as result of Observation or Belief Adoption.	29
3.6	Graphical representation of an RDF statement	30
3.7	Is-a relationship between classes	33
4.1	Architecture of the system.	39
4.2	Data flow.	40
4.3	Reification.	48
4.4	An example of a composite inference	51
4.5	First arguments from the history of Alpine iceman	51
4.6	Composite inference from the history of Alpine iceman	52
4.7	Second arguments from the history of Alpine iceman	58
4.8	Example of Inference Defeat	59
4.9	Third arguments from the history of Alpine iceman	60
4.10	Fourth arguments from the history of Alpine iceman	61

4.11	Second example of Inference Defeat	62
4.12	Final arguments from the history of Alpine iceman	62
4.13	Login the system	68
4.14	Form for registration	68
4.15	Home Page	69
4.16	Creation of an issue and an observation	70
4.17	Menus with classes and instances of the domain ontology define the subject of the triple	71
4.18	Menu with predicates of a selected class	71
4.19	Menus with classes and instances of the domain ontology define the object of the triple	72
4.20	An observation in natural language	72
4.21	First state of knowledge	73
4.22	Form of the elementary inference	74
4.23	Elementary inference in natural language	74
4.24	Second state of knowledge	75
4.25	Third state of knowledge	75
4.26	Fourth state of knowledge	76
4.27	Fifth state of knowledge	77
4.28	Sixth state of knowledge	77
4.29	Final state of knowledge	78
4.30	Scroll bar with states of knowledge	79
4.31	Example of a proposition's history	80
4.32	Example of a proposition's history and the arguments that it came up	80
4.33	Example of searching propositions	81

Chapter 1

Introduction

1.1 Introduction

Argumentation is an important human activity concerning many aspects of human life and society. Argumentation can be defined as the process of employing arguments to achieve a particular state of knowledge, a production of arguments that may lead to a conclusion and can be expressed through dialogue. An argument traditionally is the reason, the justification of some conclusion. Aristotle was the first who worked on argumentation with his logical theory. This indicates that argumentation was an important factor already in society. Today, its importance is recognized in a multiplicity of research fields. It is used in every day dialogs and negotiations as well as in civil debates, scientific and scholarly discourse, law and business. Argumentation has also attracted the interest of many researchers in the computer science. The widespread of Internet use and the huge amount of information available on the web made the database technology and the access to digital information integral part of human activities. Internet seems to be the ideal platform for communication and exchange of arguments. In this context, it is of vital importance for building flexible and useful argumentation support systems to fully understand human argumentation processes and to provide a formal model underpin arguments' visualization and also to be able to exchange arguments and structure argumentation.

A lot of systems supporting argumentation have been developed. However, the connection between the Information Technology (IT) and argumentation based on information has

not been achieved sufficiently. None of the current systems connects arguments explicitly with atomic database contents, and none integrates the historical evolution of arguments with composite arguments believed at a certain time. Furthermore, most of these systems are based on relational database structures in order to store and represent arguments or they use inflexible programming languages failing to capture rich ontological concept or to be extensible. On the other hand, Semantic technologies can provide valuable support to store, visualize structure and exchange arguments in distant collaborations. Semantic web languages are very flexible and more appropriate for representing arguments.

In this thesis we aim to provide an abstract model for representing and analyzing arguments that is connected to a domain ontology that describes the possible states of affairs and also makes explicit both the processes of argument making and the states of belief at a particular point in time in a composite inference. The argumentation field is too wide and unexplored to make any claim that we have cover the whole field with this model. Therefore we have specialized the model to factual argumentation, i.e., to arguments and counterarguments about propositions about material states of affairs in the past. As proof of concept we aim to pass from the theory of the model to the implementation. We aim to present an information system built on this model which is based on semantic technologies for the registration and the representation of networks of arguments.

1.2 The problem

More and more people every day take part in on-line discussions in order to improve their knowledge about particular state of affairs. Arguments are important elements for building knowledge and hypothesis. There is a plethora of human arguments in the Web. Hence, it important to find a way to structure all these argument and provide advanced support to manage and manipulate human arguments.

There is a variety of approaches that providing argumentation systems used for different research fields. The problem is that some of these are highly structured and restricted to formal logic (decision making systems) in which reasoning consists only of falsification or verification of hypothesis and others are not analytical enough or even they are blogs or unstructured online forums accessed through websites. There is a need for a model that is

highly structured and offers highly scalability at the same time in which reasoning may not only consist of falsification or verification but more generally of strengthening or weakening hypotheses.

Human argumentation is a dynamic process aims to understand, support and verify knowledge. Modeling human argumentation should shed light on how knowledge in information systems could be better accessed, structured and used for real life research purposes. Another deficiency in the existing systems we found in our investigation is the lack of the connection to an ontology of the domain of discourse. Ontology defines the “possible states of affairs” of things that exist in the application domain [34]. Argumentation belongs to epistemology, the possible ways of knowing. Without ontology, terms and symbols are not defined and can not well be linked to knowledge. There is no explicit connection showing how the argumentation model acts on the domain ontology. Hence we cannot learn from them how scientific argumentation may operate on an information system. The elements of the models we found tend to be phrases in natural language, which can only be interpreted as arguments in conjunction with tacit background knowledge. We also found inconsistencies and confusion about the precise meaning of terms such as *argument*, *opinion*, *position*, *reason*, *ideas*, *premise*, *conclusion*, *issue* etc. The ontological nature of what an “argument” is stays mostly unclear. Many models confuse the argument with a fact used as an argument. Our model interprets arguments as explicit relationship between facts and hence is free of tacit background knowledge for the interpretation of the role of such facts. Finally, there is no previous work that makes explicit what the state of belief in composite inferences at some given point in time is and how it changes.

1.3 Contribution of this thesis

In this work we made a survey of the related work of argumentation systems for several domains (legal, learning, archaeological and computer science domain) and we selected and adopted useful elements from the existing argumentation models and applications. We studied an existing model for human argumentation not restricted to falsification or verification (a practice that is applied in most models of formal logic), but more generally describing processes of strengthening or weakening hypotheses about states of affairs. The model can

be connected to a domain ontology in order to describe these states of affairs. We extended this model in order to focus mainly on factual argumentation about material states of affairs in the past. We did not look at kinds of argumentation about what “should be done” in future. Also we made extensions in order the model to be able to defeat inconsistent arguments. More specifically, we add classes and we made generalizations to others in order to describe correct factual argumentation (see Section 3.1) and the process of inference defeating. We implemented the model and we built an information system based on this model using RDF (Resource Description Framework semantic language) and RDFS (RDF schema). The system aims at the record and the representation of arguments so that a group of people to be able to collaborate. The system supports creation, composition, revision, storage and representation of arguments through time. It enables users to monitor the current state of knowledge as well as that at any other instant in the past. Also, it inform the users about the provenance of the knowledge, how why and when this knowledge came up. We demonstrate the model and the system by analyzing a published archaeological example. The system is a web tool and uses a set of software components such as Sesame [5] RDF repository, Apache Tomcat server, Ajax, JavaScript and Java for the communication between Sesame and JSP pages. Also SPARQL language [6] is used for querying the repository.

1.4 Organization of this thesis

This thesis is organized as follows.

Chapter 2 discusses related work and gives an overview about the existing argumentation models and systems.

In **Chapter 3**, we introduce an integrated argumentation model, namely IAM for factual argumentation and we describe what extensions we made on the basic model and why. Finally, we explain how the argumentation model acts on a domain ontology and how the model makes explicit both the processes of argument making and the states of belief at a particular point in time in a composite inference.

In **Chapter 4**, we present Antilogos, the information system that we implemented based on the IAM as proof of concept. We refer how we restricted IAM for this implementation

and we describe the architecture and the components of the system.

Chapter 5 concludes this thesis and identifies issues that are worth further research.

The appendix comes after the conclusion contains the ontology in RDF code.

Chapter 2

Related Work

2.1 Argumentation Models and Systems

There is a variety of approaches and computer systems for modeling and representing arguments so far. For the purpose of this thesis, related work can be divided in two areas; on one side we investigate models that can be characterized as *declarative argumentation models and systems* - these models are static and focus on analyzing the internal structure of complex arguments (basically comprising of premises and conclusions). On another side, we refer to *argumentative process models and systems* which are dynamic, procedural models that show the evolution of arguments. These “dialectical” models typically analyze the process of constructing arguments and counterarguments during a discussion.

2.1.1 Type-structure argumentation models and systems

A popular form of argument is the Toulmin’s model [36]. Toulmin’s model is focusing on micro-arguments. His argument theory provides a theoretical framework for analyzing the structure of written arguments. Toulmin’s arguments typically consist of six parts. *Claim* is the statement being argued, the conclusion of the argument. *Data* are the facts or evidences which are the foundation for the *Claim*. *Warrants* are the general, hypothetical (and often implicit) logical statements that serve as bridges between the claim and the data. *Qualifiers* are statements that limit the strength of the argument or statements that propose the conditions under which the argument is true. *Rebuttals* are counter-arguments

or statements indicating circumstances when the general argument does not hold true, they are exceptions to the claim. *Backing* is statements that serve to support the warrants [Figure 2.1]. The elements of the model are natural language phrases, and therefore miss, besides others, to distinguish the facts from the belief about the facts. It is hard to make a bridge from the notions of “warrant” and “backing” to any more analytical sort of inference logic, formal or informal. Toulmin’s scheme has been influential on further research. Many scientists based their work on his model and tried to build on it. However, he has only discussed the structure of arguments. He failed to analyze the problem-solving process. Toulmin’s approach does not support reinstatements and does not discuss the evaluation status of arguments.

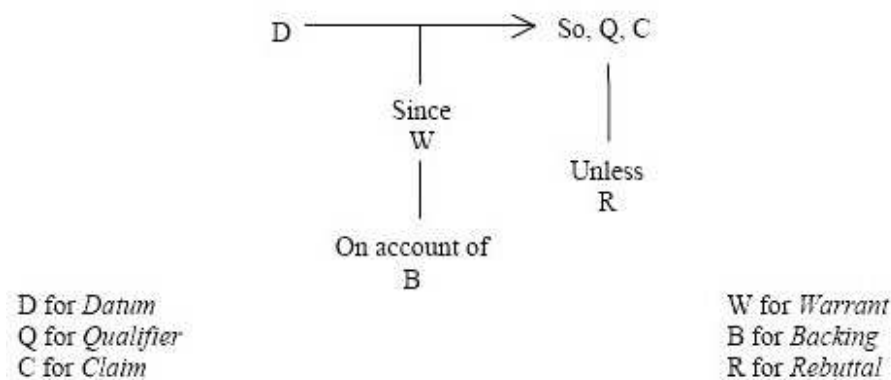


Figure 2.1: Toulmin’s argumentation scheme [36]

The Arkeotek project [15] is a meta-model for retrieving scientific reasoning and structuring documents in order to manage and retrieve scientific knowledge about archaeological publication. The Arkeotek project uses SCD (Scientific Construct Data format) format for structuring documents. It brings out the logico-semantic structure of the archaeological theories. The SCD model is an adaption of the logicist program. According to the “Logicist Analysis” [16], archaeological theories are considered as computational structures made up of a data base which is a set of declarative propositions (P0) relating to object or phenomena and an inference tree expressing the steps as an author goes from one set of propositions (Pi) to another set of propositions (Pj) - conclusions/hypotheses in the argument [14]. This tree can be read bidirectional: empirico-inductive from the data base P0 to conclusions

P_n and hypothetico-deductive from hypotheses P_n to the data base P₀ [Figure 2.2]. This diagram forms a synoptic presentation of the chaining of reasoning steps. Each node represents a proposition that contains a title together with some text for describing data (in initial propositions), or that puts to light the logical operations carried out at each step (interpretative proposition). Interpretative proposition have antecedent propositions, which are their son node in the tree. The meaning of the links between propositions is that all the antecedents of a proposition are the statements required to demonstrate the validity of this proposition.

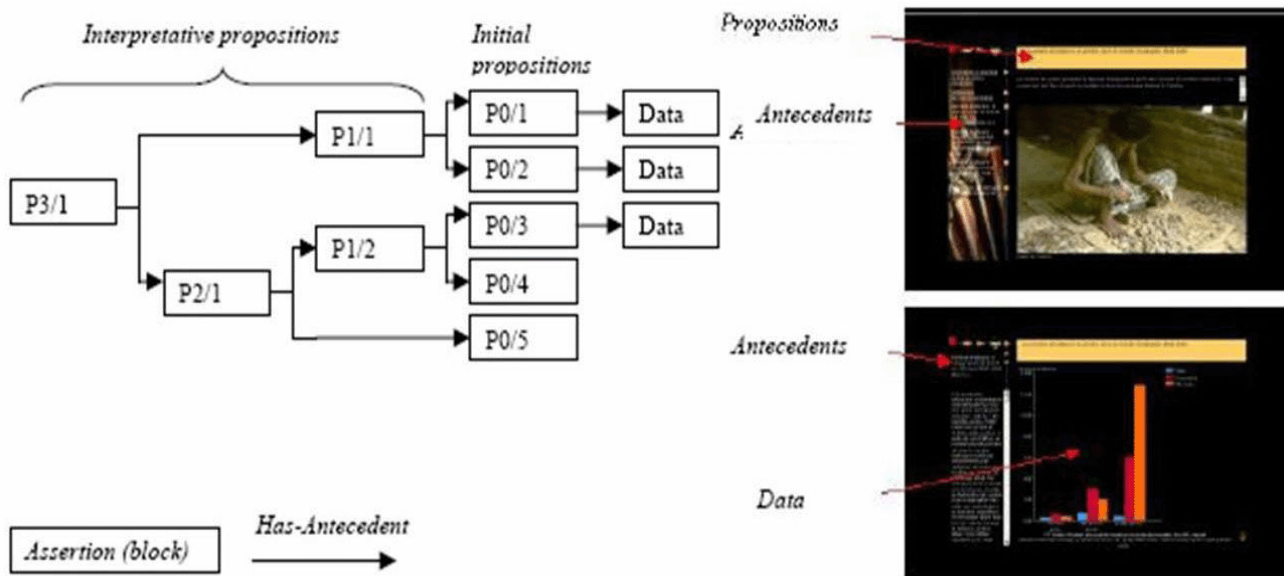


Figure 2.2: Scientific document structure according to Logician/SCD format [10].

To improve the information retrieval the Arkeotek project experiments semantic annotation with a domain ontology. Using the domain meta-model Arkeotek has successfully demonstrated the semantic annotation in order to structure real archaeological publications according to the role of each paragraph in the overall argumentation [9]. The documents are fragmented into propositions (initial or interpretative) and analyzed. The domain ontology is specialized for document annotation. Each paragraph is associated with domain concepts, relations or terms. The structured documents form a model of how facts, hypotheses or data are interpreted to produce new facts and hypotheses. It represents graphically the reasoning underlying scientific constructs and makes the whole publication easier to read

and to understand.

2.1.2 Procedural argumentation models and systems

Other works deal with arguments in communication processes. We distinguish three main categories: the decision making argumentation systems in collaborative environments, the systems that incorporates argumentation in ontology engineering and the systems for visualizing arguments.

Decision making

Decision making systems in collaborative environments used when there is a group of persons or agents collaboratively looking for a decision about a certain topic or problem. Most applications in this category aim for linking arguments with each other using relationships and claims. The gIBIS (graphical IBIS) [13] is such a system. Is a graphical tool that implements a specific method, called Issue Based Information Systems (IBIS) in order to build and browse typed IBIS networks. gIBIS supports collaborative constrictions of these networks by a number of cooperating team members. The IBIS method was developed by Horst Rittel in 1970 [23]. According to Rittel any problem, concern, or question can be an *issue*, and may require discussion (if not agreement) in order for the design to proceed. Each Issue can have many *Positions*. A Position is a statement or assertion which resolves the Issue and they may have one or more *Arguments* which either support that Position or object to it. There are types of issues and several kinds of relationships between them. The IBIS methodology is proposed as a solution to decision problems and relies on a model of problem solving as an argumentative process. IBIS does not support the deduction of new conclusions, since it focuses only the *justification* of an initial central issue. The interface of gIBIS represents the nodes and the relations between them in a graph structure and through a control panel that contains buttons for creating nodes and links. The gIBIS makes use of a relational database as other tools. Relying on databases, could somehow create a kind of rigidity in extending structures. The modification of a database schema usually requires taking care of the table's reference keys and table's structures, requiring the involvement of the database administrators and designers. gIBIS also visualizes arguments. Figure 2.3 illustrates the main interface of gIBIS.

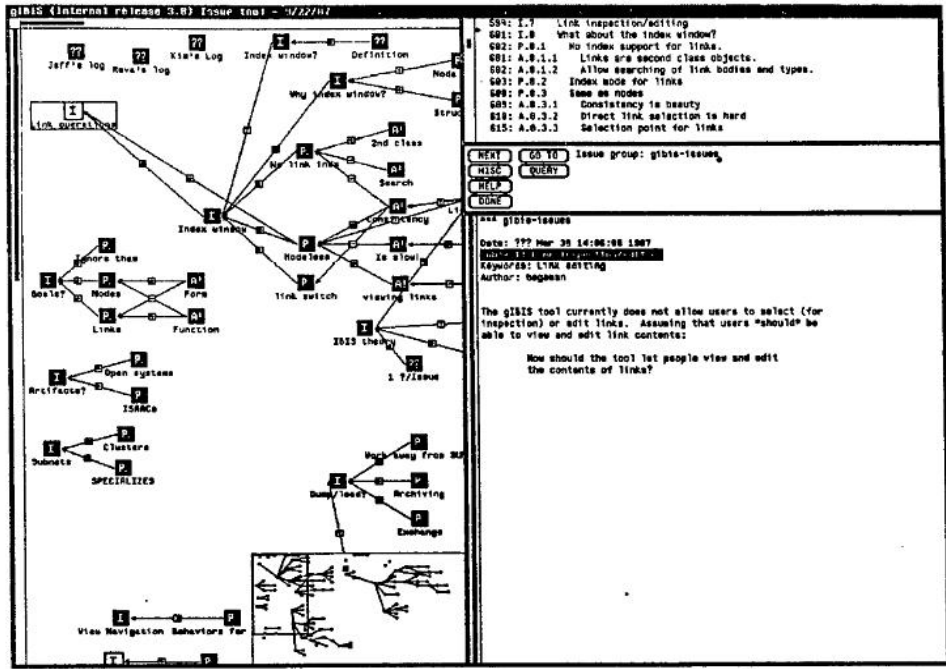


Figure 2.3: gIBIS interface ([13])

A variant of (IBIS) model is ZENO [18], a formal model of argumentation. The Zeno is a Web-based mediating system¹ based on the informal models of Toulmin and Rittel. ZENO provides an issue-based discussion forum where users can structure their arguments and set their preferences among them. The argumentation elements of ZENO's model are *issues*, *positions*, and *arguments* (pro or con). A position is defined by its role and use in a discussion. An advantage of ZENO over the informal Toulmin and IBIS models of argumentation is that provides the possibility to label alternative positions of an issue, whereas IBIS does not show the effect of arguments on the status of positions. It is a dynamic, dialectical model which emphasizes on showing the states of argumentation. Still, since it is issue-based, it has *one specific issue only* to justify. ZENO interface places issues, positions and arguments into a "picture". The index produced by ZENO is stored in a relational database where messages and the argumentation elements within messages are

¹A mediation system is a kind of computer based discussion forum with particular support for argumentation. In addition to the generic functions for viewing, browsing and responding to messages, a mediation system uses a formal model of argumentation to facilitate retrieval, to show and manage dependencies between arguments, to provide heuristic information focusing the discussion on solutions which appear most promising, and to assist human mediators in providing advice about the rights and obligations of the participants in formally regulated decision making procedures.

stored. SQL queries are used to be selected, filtered and sorted them. ZENO was used also by other systems. GEOMED [31] (GEOgraphical MEDIation) is one of them. The GEOMED provides an Internet based support for spatial planning and decision making, like regional or urban planning. Another decision support system based on Zeno is HERMES [21]. HERMES aids decision makers to reach a decision, not only by efficiently structuring the discussion, but also by providing reasoning mechanisms for it. Also, it supports constraint solving and conflict detection features. HERMES tool used in the medical field.

Compendium [32] is a software package supporting IBIS-based dialog mapping. It is a graphical hypermedia tool that enables mapping of meetings and provides a set of templates, methods, and tools that connect people and ideas. The templates are question-oriented that used to question or raise an issue in a subject matter domain. Also it is a system for development metadata codes that are assigned to any concept in the database. A discussion is visualized by different maps, interlinking and connecting the exchanged arguments (Compendium use visualize techniques). Compendium offers enhanced tools such as XML and RDF support, as well as connection to other data stores.

Ontology Engineering

Structured argumentation is also beneficial for ontology engineering. We describe two methodologies that have been developed in this context DILIGENT [28] [35] and HCOME [22].

DILIGENT argumentation Ontology is a project proposed for knowledge management between virtual e-science organizations. It is theoretically based on Rhetorical Structure Theory and IBIS methodology; the DILIGENT ontology includes three main concepts such as *issues*, *ideas* and *arguments*, represented as classes. *Issues* introduce new concepts in the discussion, in the sense of what *should be* in a conceptual model of the ontology - an issue also may refine another issue under discussion (elaborations). *Ideas* refer to how concepts should be represented and formalized in the ontology (as a class, as an instance etc.). *Ideas* respond to *issues* indicating the way of their implementation in the ontology. *Arguments* are related/argued to/on one particular idea or one particular issue. Arguments types such as elaboration, evaluation/justification, alternatives, examples and *counter examples* were identified as the most influential on the creation of an ontology. *Positions* on *issues*, *ideas*

or *arguments* lead to *decisions*. It is a process model which traces types of arguments *exchanged in a discussion* during an ontology building process.

HCOME is a methodology which integrates argumentation and ontology engineering in a distributed setting. It supports the development of ontology versions. Phases of ontology development are performed in three different working spaces: Personal Space, Shared Space and Agreed Space. Users can improvise, merge and manage ontologies in the Personal Space then they can post them to the Shared Space for further discussion with others members. An argumentation dialogue aims to reach an agreement on the aim and scope of a shared ontology. Any group member can raise *issues*, makes *positions*, and states *arguments* for and against these positions concerning a shared ontology. Specifically, the argumentation dialogue follows a version of the IBIS model.

The issue abstraction represents a decision problem. A position is a statement that resolves an issue and can be a posting of a new version of an ontology. The argument either supports or objects a position. HCOME has limited the IBIS model to seven relationships between the abstractions mentioned: zero or more positions or versions of an ontology may provide a solution for an issue raised. Each such version can be supported or objected by zero or more arguments. Also an issue can suggest a new position/version, or an issue can be the generalization or specialization of another issue. Furthermore, an argument can raise an issue. HCOME is a useful approach to integrate different information needs into one consensual ontology; however, it does not analyze or structure the arguments that are used in a discussion.

Visualizing Arguments

Visualization of arguments is important in cases where people are having a certain debate, or in learning environments. They help students to improve their argumentation skills and may be able to understand more about arguments.

Araucaria [30] is a tool of argument diagramming based on the Argumentation Markup Language (AML)[12] formulated in XML. Once an argument has been analysed it can be saved in the portable format AML. Araucaria is used in teaching and studying philosophy for visualizing arguments. Users can build arguments and link them with relations. Arguments building starting from an argumentation text input file that consists of propositions

identified either as a premise or as a conclusion. This practice, generally, can not describe the use of the argument propositions in a different role (since premise and conclusion are not defined and represented as a role, but as a class, a distinct entity). Araucaria supports argumentation schemes, and provides a user-customisable set of schemes with which to analyse arguments.

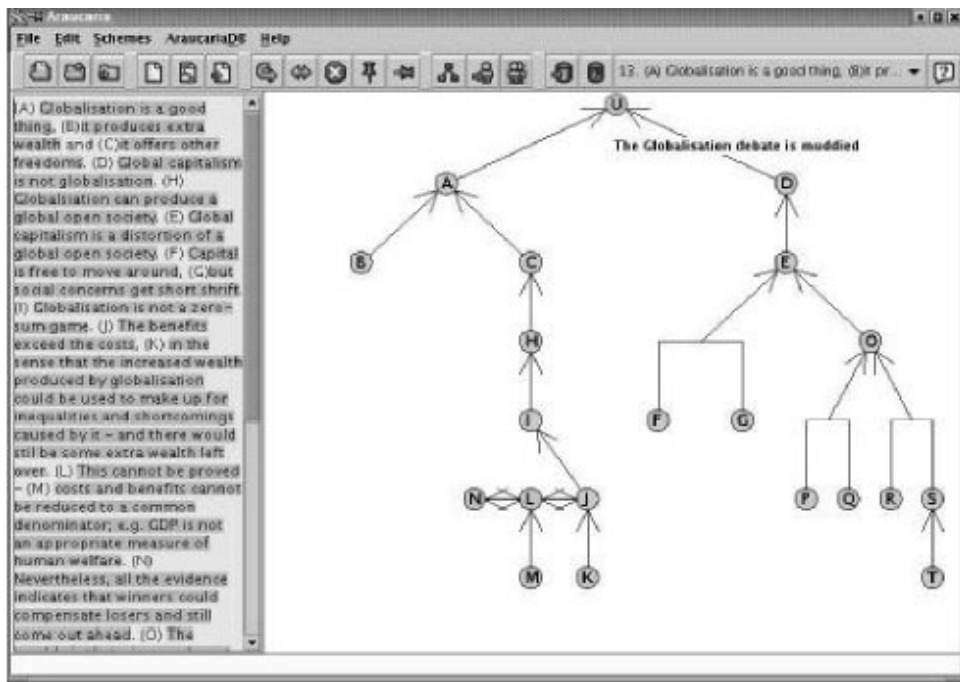


Figure 2.4: Araucaria Screenshot [30]

Figure 2.4 shows what an Araucaria window looks like. The argumentation always has the form of a tree structure. The nodes of the diagram (indicated by alphabetical letters) are formed by highlighting sections of the argument text. Premises can be arranged in serial, convergent or linked structures. Also, relationships from one node to another are possible to be created.

ClaiMaker [24] is another argumentation system that includes visualization of arguments. ClaiMaker is a system for modelling readers' interpretations of the core content of papers. ClaiMaker provides tools to build a Semantic Web representation of the claims and arguments in research papers. It is based on the ScholOnto ontology [33] which can express a number of basic reasoning schemes and relationships between concepts found in scholarly

discourse (e.g. similarity of ideas and taxonomies of concepts).

Another tool for learning domain and visualizing complex structure of an argument is Reason!Able [17]. Reason!Able was developed as an educational tool in order to improve students' critical thinking. It is a software package used to manipulate, annotate and display argument graphs. Reason!Able helps to organize common thought, explains the way of using arguments and produces argument maps in a colored diagrammatic form. Reason!Able provides a workspace within which click and drag operations are used to build and modify hierarchical "tree" structures representing the inferential relationships among the various claims which make up argument. It builds argument structures made of claims, reasons and objections.

A similar software package to Reason!Able developed for the same purposes is Athena [1]. Athena is an easy tool for argument visualization.

TruthMapping [7] is an on-line Web-based argumentation system in which a large number of users take part in who can store their claims and support them with corresponding links. It makes a clear distinction between unsupported premises, which when supported become claims, and provides a way to post rebuttals and responses to each of these. TruthMapping supports arguments that consist only of premises and conclusions. Also, each user can add critique to the premise. Arguments can be linked with each other.

The ArgDF system [29] is a web tool enabling users to author new arguments or manipulate and visualize existing ones. Is the first argumentation tool to adopt a Semantic Web architecture based around the W3C standard Resource Description Framework (RDF) for distributed data modeling and interchange. The ontology used for describing arguments based on the Argument Interchange Format (AIF) [11]. This combination of AIF and RDF is a notable advance. However, while proving the conceptual and technical feasibility of a semantic web orientation for argumentation, it does not yet have a user community, an engaging User Interface.

Other Procedural systems

Argumed-system [38] is argument mediation system for lawyers with a template-based interface. The argumentation theory of the ArguMed-system is an adaptation of Verheij's

CumulA-model [37] a procedural model of argumentation with arguments and counterarguments. It is based on two main assumptions. The first assumption is that argumentation is a process during which arguments are constructed and counterarguments are adduced. The second assumption is that the arguments used in argumentation are *defeasible*, in the sense that whether they justify their conclusion depends on the counterarguments available at a *stage* of the argumentation process. The user gradually constructs arguments, by filling in templates that correspond to common argument patterns. The ArguMed-system has three central *data structures*. The elementary data structure is that of a *statement*. A statement consists of a *sentence* that represents the propositional content of the statement. Statements can be of two types, of *issue-type* and of *assumption-type*. The second data structure is that of an *argument*, which is simply a tree of statements. The third data structure combines *reasons* with their *conclusions*: it consists of two statements, one of which represents a reason, the other the conclusion supported by the reason. They are also contained exceptions (of undercutter-type) that can block the connection between the reason and the conclusion. Unlike IBIS, ArguMed allows for free argumentation: forward argumentation or inference (new conclusions inferred) and backward argumentation or justification (new reasons adduced).

2.2 Summary and Discussion

Summarizing, we detect in two categories of argumentation models:

1. Static models which are focus on analyzing the internal structure of complex arguments.
2. Dynamic, dialectical and procedural models which are show the evolution of arguments.

Almost most of the models we found are obviously successful to monitor a collaborative dialogue and to inform people in a structured way about a discourse. However, they deal with argumentation without targeting the issue of integration between domain ontology and argumentation. The applications of these models use the form of a text box where users can enter any piece of information in order to capture arguments. Ontology defines the

“possible states of affairs” of things that exists. Argumentation belongs to epistemology, the possible ways of knowing. Without ontology, terms and symbols are not defined and can not well be linked to knowledge. Hence, they cannot shed light on how the argumentation may act on knowledge elements in information systems. Further, most of the applications fail to distinguish the structure of a composite argument believed at some point in time from the temporal order of arguments in an argumentation process. The logical sequence of one conclusion being premise for the next conclusion is wrongly taken for the temporal order in which the inferences were found. Indeed, the historical order is widely independent from the logical one, and only accidentally coincides. Therefore they can not support argumentation analysis well. Also, some systems are restricted to formal logic (decision making systems) and they cannot capture the richness of reasoning and others are not analytical enough or unstructured. The structure of argument in some models is base on the traditional premise-conclusion but from a philosophical point of view an argument does not consist just of premises and conclusion it has additional, internal structure. Also other models regard an argument as a statement which is instance of a class named “argument”, “position”, “claim”, “reason”, “objection”, “premise”. These models cannot describe the use of argument proposition in different role. If a statement is declared as “premise” or “reason”, there is the problem that the same statement could be first “conclusion” and then “premise” of the next argument. Finally, the most systems are not benefit from the semantic technologies and they are based on databases and rigid programming languages that make them difficult to be extensible and flexible.

Chapter 3

An Integrated Model

In this chapter, we describe our argumentation model (see [25] for more details). Let us call it “IAM”, Integrated Argumentation Model. The model aims at connecting the epistemological aspects of an argumentation theory with instances of a formal ontology as may be found as data in an information system. It further differentiates between the temporal aspects of argumentation, i.e., the making of an inference as historical (temporal) event and the associated phases of believing, from the synchronic structure of a complex inference chain and its associated belief values, and integrates both. Argument making may motivate other argument making.

Figures 3.1, 3.3, 3.4 ,3.5 illustrate subsequently the model from general to more specific notions. We denote classes by boxes, properties by thin arrows, and IsA relations by thick, grey arrows. We use the CIDOC CRM (ISO21127) [26] as domain ontology which has been proven to adequately describing fundamental aspects of history in the sense and the terms of discrete, human-scale events (in contrast to plant growth, for instance). We refer in the following to CRM concepts with a namespace “crm” followed by the concept identifier used by the CRM, such ”crm:E39 Actor”. We omit the namespace in the figures.

3.1 Generic Concepts

We regard an argument as a composite structure of claims with inferential or evidential relationships to each other. An argument may be a series of statements with intermediate steps providing the transition from the premises to the conclusion, an inference [27]. We

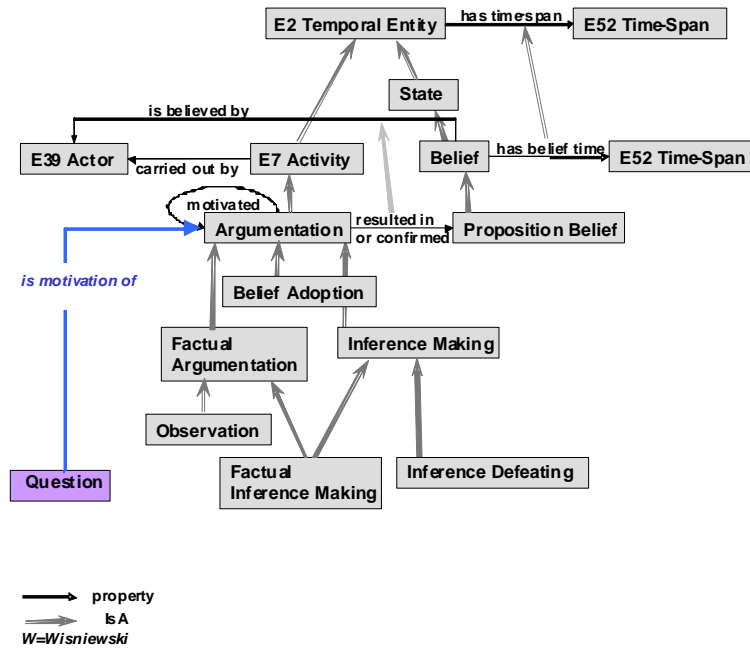


Figure 3.1: Types of argumentation: Factual Argumentation, Inference making, and Belief Adoption.

regard an inference as the application of a set of inference rules applied on some statements producing or motivating other statements; a process of passing from some proposition(s) to another proposition [39]. Premises and conclusion of our arguments are basically statements about beliefs in propositions, rather than the propositions themselves. We regard argumentation as the process of employing arguments to achieve a particular state of knowledge. And we regard knowledge as the justified belief that something is true.

We consider *Belief* (see Figure 3.1) as a mental state that is determined by a (human) *Actor* (crm:E39 Actor), a particular belief value and a proposition, and may exist for some *Time-Span* (crm:E52 Time-Span). So, we regard *Belief* as a *State*. A *State* can be defined as the persistence of a particular value range of the properties of a particular thing or things (such as “having fever”) over a time-span. We regard any change of belief value as initiating a new belief, i.e., the identity of the belief depends on the particular value over a simple time-interval.

We consider *Actor* to be a person or a *group* (sharing believes). We regard that the belief value of an *Actor* in a proposition may increase or decrease, or be *strengthened* and

weakened, based on *confrontation* with evidence or even the pure beliefs of another authority or majority.

Argumentation is a process which “*result in or confirm*” instances *Proposition Beliefs* (see Figure 3.1, and Section 3.2.2), i.e., beliefs in some statement that can objectively be described. We assume that argumentation and its constituents is made public and is comprehensible to the partners of a discourse as in good scholarly and scientific practice, and that it is therefore sufficient to describe belief changes in the discrete steps of publicized arguments.

Argumentation may take on many forms. We regard *factual knowledge*, in contrast to categorical or theoretical knowledge, as the justified belief that a particular state of affairs is true, and we define *Factual Argumentation* as the one handling factual knowledge. Such a belief may be grounded on direct empirical Observation (evidence), *belief* in an *Inference* (Figure 3.3) about facts, or *trust* in a reported belief, i.e., a belief shared with others or better being adopted from others, typically in the form of a literature reference. The latter process we describe as *Belief Adoption*. *Inferences* are produced in processes of *Inference Making* (Figure 3.3), be they factual or others. An Inference may be *Elementary* or *Composite*. In the following, we go only into detailed analysis of argumentation with respect to factual knowledge. As exception, we have to include a non-factual form of inference making, *Inference Defeating*, because factual knowledge may be invalidated not only by new evidence, but also by suggesting “impossible worlds”. The logical resolution of such “inconsistencies” can, to our opinion, be described by “defeating” previous inferences themselves, rather than the facts these inferences used (see below 3.2.7).

In argumentation about reality, belief values may not just be “true” or “false”, but also “unknown”, or any degree of subjective or objective probability that the proposition corresponds to reality, possibility or plausibility. For instance, in some cases, trust in a source may be expressed as the probability of errors inferred from the frequency of errors observed in a representative sample. We know quite well that even direct observation is error prone, as well as the reporting of it. So, in a more general sense, any belief value might be regarded as a sort of probability a proposition to be true or false, and realistic argumentation models should be extensible to probabilistic belief values in a monotonic way. Our model aims at being at least generic enough to comprise both probabilistic beliefs

and three-valued beliefs (true, unknown, false). Sources of errors that may be represented in a probabilistic belief model may be: misobservation, misinterpretation, memory gaps, misreading, writing and spelling errors, misapplication of logic. Our examples will however only deal with three-valued believes.

We finally see a connection to questions, which we regard as generalizations of notions like “problems” and “issues” (we believe that our model could be connected with Wisniewski’s erotetic inferences [39, 40]).

In this thesis we did not developed the whole model from the begging. This model is developed based on previous work¹. In a nut, the contribution of the thesis on the model is:

- The explicit representation of Factual Argumentation. Before this thesis the notion of factual argumentation was only theoretically. There wasn’t any class “Factual Argumentation” and the classes Observation, Belief Adoption and Inference Making were direct subclasses of the class Argumentation.
- The introduction of the class “Inference Defeating” as a special case of the Inference Making and the separation of the Factual Inference Making.
- The generalization of Composite and Elementary inferences. We introduce the class Inference. We want the transition from an Elementary to a Composite Inference to be monotonic. The class Inference allow us not to delete any information when such a transition happens.
- The generalization the class Proposition in order to include Inference.
- The implementation of the final model in RDF encoding

3.2 The Model

3.2.1 Inference Making

The core notion of our model is the process of *Inference Making*, a historical event and human activity (crm:E7 Activity), in which an *Actor* relates two or more **Propositions** as playing the roles of *premise* and *conclusion* of an **Inference** [Figure 3.3]. An instance of

¹See the initial version of the model http://www.csd.uoc.gr/~boutsika/papers/ArgumentModel_v6.doc

Inference Making may derive new true or false propositions as a conclusion, or only *verify* or *falsify* the truth value of a known proposition.

3.2.2 Proposition Beliefs

Propositions are regarded to be statements about things that are, and as such may be true or false . We call *Factual Proposition* a proposition about a particular state of affairs (a “material fact” [20]). An inference depends critically on the assumed truth value of its premises and from these produces truth values for its conclusions. The truth value cannot be integral part of the proposition itself, if we want to represent a discourse in which different Actors assign different truth values at different times to the same propositions, and want to be informed about what is *believed* at any time about a particular proposition - in particular a proposition registered in an information system. Hence the truth values we must handle in our model are actually *Belief Values*.

For the time being, we aim at representing only honest argumentation, i.e., argumentation aiming at increasing the state of knowledge, and hence believing the respective propositions as expressed by the belief values at least for the duration of the argumentation. (This conforms to good scholarly and scientific publication practice). Therefore, the premises and conclusions in our model are not direct relations between an inference and propositions, but relations between an inference and instances of *Proposition Belief* consisting of a set of propositions and a *Belief Value* common to this set. Consequently, a *factual proposition belief* believes factual propositions.

We impose the following constraint: An actor can have only one belief for a proposition (one proposition belief) at a time. The proposition belief of an actor may change over time. In the implementation of our system system, we may assume that the beliefs of any actor are only the ones made known to the system and from the time on they are made known. We do not intend to analyze any Open World issues in this thesis.

3.2.3 Factual Inference Making

Since we do not verify our model with all kinds of inference making, in particular justifying categorical knowledge, we define *Factual Inference Making* as a special case of Inference

Making AND Factual Argumentation. A process of factual inference making may only conclude in *factual proposition beliefs*, rather than other proposition beliefs and must use as premise *factual proposition beliefs*. For the time being, we assume factual propositions to be statements about particulars that can be formulated in terms of a formal ontology. Examples of such statements may be unary or binary predicates, such as ‘Martin’ is a ‘Researcher’ (instance of), ‘Section 3.1’ is part of ‘This Thesis’.

3.2.4 Inferences

We regard an *Inference* as the application of a set of inference rules applied on some statements producing or motivating other statements; a process of passing from some proposition(s) to another proposition [39]. This process can be encoded in a static way, i.e., independent from the time it happened. In that sense, an inference can be seen as an information object, a simple or composite statement in its own right. Therefore we regard it as a subclass of *Proposition*. It is, however, not factual, because it only relates beliefs in reality, but is neutral with respect to their truth, the classical “if...then...”. As it is a proposition, it is subject to a belief, an *Inference Belief*, subclass of *Proposition Belief* (see Section 3.2.6)

3.2.5 Inference Logic

We assume that any scientific or good scholarly inference can be described as correct application of some form of *Inference Logic* that is generally acceptable for serious argumentation in a scientific or scholarly discipline [16, 14]. Examples are the direct application of formal logic, mathematical theories and calculus, formal or informal default reasoning based on default values associated with categories, probabilistic reasoning based mathematical models and assumed or observed frequencies for certain categories, etc.. For the time being, we pack into the concept Inference Logic all sorts of categorical background knowledge and methods employed.

These may be analyzed into much more detail in the future - for instance, one may distinguish general logic and calculus from axioms and empirically founded theories. For the time being we regard inference logic is only relevant that the inference logic does not contain any statement about something particular that is, and as such does not pertain

to any actual state of affairs [19] under consideration. This is necessary order to avoid confusing the inference logic with factual premises and conclusions.

3.2.6 Inference Beliefs

As we learn from experience, an inference may yield or support correct conclusions, but yet be wrong, i.e., a wrong application of logic, calculus or axioms. On the other side, in reasoning forms based on defaults or probability, an inference may be correct and applicable, but the conclusion nevertheless is wrong. Following our principle of honest argumentation we assume that the inference maker believes him/herself at least from the end of the inference making process on that the inference is correct, but may be convinced at any later time that the inference is actually wrong, regardless the truth of the conclusion. I.e., an *Inference Belief* may be independent from the belief in the conclusion of the believed inference. In our model (Figure 3.3), an instance of Inference Making *initiates* a positive *Inference Belief* in the inference made, which must last longer than the time-span of the inference making.

For the time being, we do not consider other belief values than *true* or *not true* for the inference belief. We make the simplification that the inference logic can only be applicable or not applicable to a particular thing, case or situation, but not be right or wrong in its own.

3.2.7 Inference Defeating

Even if we want restrict our model purely to reasoning on facts, we must include the case that an inference per se, regardless the beliefs in premises and conclusions, is questioned, in other words, that an Inference Belief is changed. Otherwise, in a respective argumentation system implementation, an Inference Belief would persist for ever. An inference may be defeated just by a reviewer. Another typical case is, when an inference has produced a conclusion which comes into contradiction with a later observation. Reinvestigating the inference, we may find flaws in the logic or its application.

Inference Defeating is in fact making an inference on another inference (see Figure 3.2), which has its own logic. Since we regard inferences as special cases of propositions (not

factual ones), the Inference Defeating falls into the same pattern of premises and conclusions as other inference making. It must have as conclusion at least one (negative) inference believe, in addition it may refer to factual propositions as premises and conclusions: If an inference is defeated, its effect on its conclusions should be “undone”. We can describe this by turning the belief values of the conclusions of the defeated inference to “unknown”, which we regard as a formalization of “questioning” the proposition. In the implementation of our argumentation system, we may restrict this change only to belief values of conclusions that are not supported by other arguments.

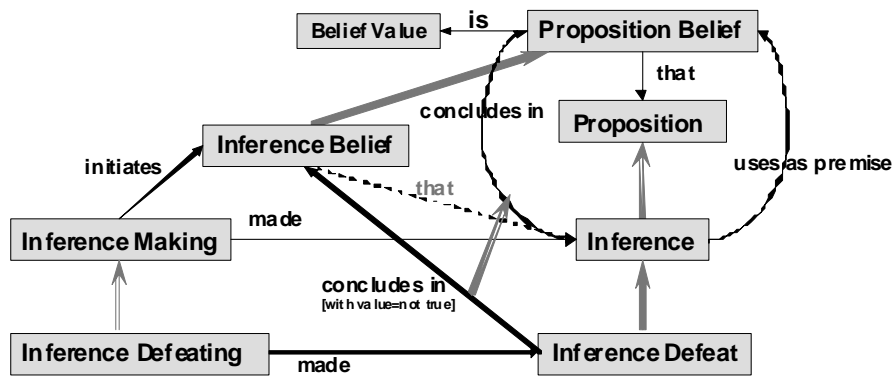


Figure 3.2: Inference Defeating, a special kind of Inference Making.

3.2.8 Elementary and Composite Inferences

We regard two kinds of inferences: the *Elementary Inference* and the *Composite Inference*. The elementary inference (see Figure 3.3) depends directly on the proposition beliefs used as premises and appearing as conclusion, the applied inference logic, and the inference belief. It represents the result of argumentation, in which no intermediate conclusions are given, but the immediate application of the inference logic, explicitly stated or not, is regarded as obvious or trusted to the expert. Applied to factual argumentation, it represents also the cases, in which a further break down into steps with intermediate factual conclusions is not possible.

The composite inference consists of a set of connected elementary inferences, as shown

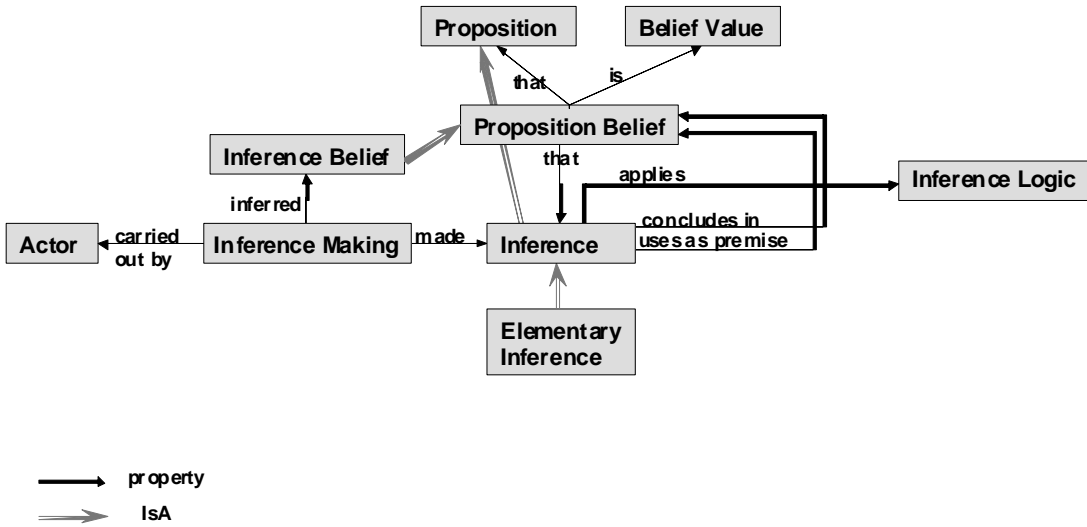


Figure 3.3: Elementary Inference Making

in Figure 3.4. For each of its elementary inferences holds: at least one of the premises must be the conclusion of another of its elementary inferences or at least one of the conclusions must be the premise of another of its elementary inferences, i.e., they form a connected, directed acyclic graph (DAG) with intermediate and final results. All premises not being also conclusions are regarded as *initial premises*. All conclusions not being also premises are *final conclusions*. The *Inference Belief* of the composite inference believes all elementary inferences, i.e. continues or adopts their inference beliefs and, in addition, believes that all premises and conclusions are consistent. The inference logic of the composite is that defining the consistency of the proposition beliefs, e.g., that a proposition cannot be true and false at the same time. Throughout the making of the composite inference, all implied premises and conclusions are believed as in the making of the elementary inferences it combines.

3.2.9 Recursive Inferences and Inconsistency

A special case of inference is not completely covered by the temporal constraints of the above model: An inference may conclude on one or more propositions in the premises themselves. In Section 4.4.2.2 we give an example of an inference which uses as premise a proposition that an event happened at a specific interval of time and concludes that the event did not happened at that specific interval of time. It is impossible to belief both propositions

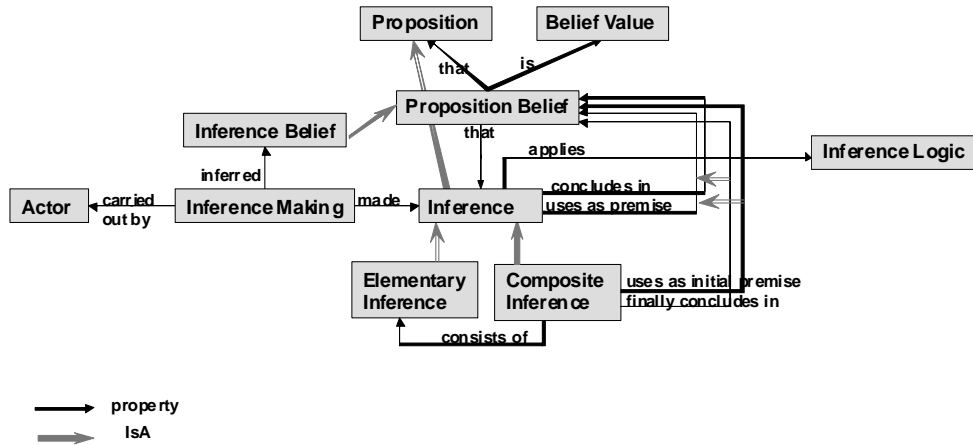


Figure 3.4: Composite Inference.

at the same time. At [25] (Section 5) there is also an example where archaeology researches conclude in a set of temporal sequences was indeed believed, until they found out that they are cyclic in time - which forms an impossible world. In these cases, the inconsistency comes not from a single proposition, but a set of propositions as a whole violates a constraint of nature we believe in. In this case, at least one of the propositions must be wrong, but we do not know which one. Using our model with 3-valued beliefs, we interpret this case in the following way: The inference renders the belief value of all involved propositions as “unknown”, which we regard as a formalization of “questioning” a proposition belief.

In the case of questioning the conclusions of an inference in the course of an inconsistency resolution, the simplest assumption is to question both the inference and the premise, a kind of Inference Defeating based on factual knowledge. Recursive inferences may be elementary or complex. We assume that all inferences revealing inconsistencies can be described in such a way. Revealing inconsistencies is a normal motivation to reexamine the sources of proposition beliefs.

3.2.10 Observations and Belief Adoptions

Observation (see Figure 3.5) is the primary source of knowledge, in the sense of natural sciences, is a kind of human activity: At some Place and within some Time-Span, certain

Physical Things and their behavior are observed, either directly by human sensory impression, or enhanced with tools and measurement devices. Mechanical and manual recordings may serve as additional evidence. Observation result in a factual proposition belief. Observations make the transition between reality and propositions in the form of instances of a formal ontology. Measurements and witnessing of events are special cases of observations. We regard any digital measurement data, regardless their size, as sets of propositions. E.g., a digital image makes propositions about the light emission strengths towards a certain point (the camera lens), in a certain time-span and place. We regard any formal and machine evaluation of observed data as inference making. Software may be used to make complex inferences.

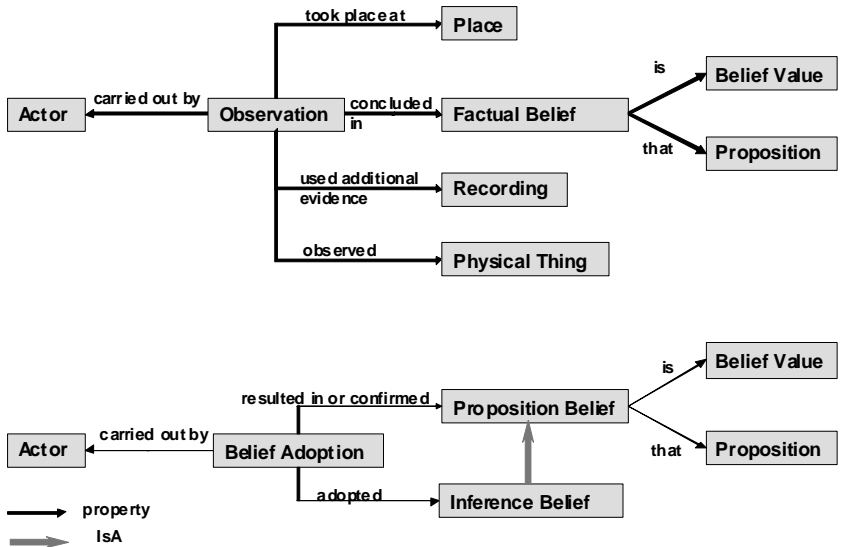


Figure 3.5: Belief as result of Observation or Belief Adoption.

The most frequent source of knowledge is other people. In scholarly and scientific practice, we use to cite the opinions of other scholars or scientists published in recognized media. We describe this process as *Belief Adoption* [Figure 3.5], i.e., an Actor adopts the belief of another Actor about a particular proposition, either taking it over, or strengthening or weakening his/her own belief in it.

3.3 Implementation of the Model in RDF

3.3.1 Background in XML, RDF and RDFS

RDF (Resource Description Framework) is a standard metadata model that is used for description, modelling and exchanging data in the Web. The base element of RDF model is the statements (also known as triples) about resources [8]. An RDF statement has three structural parts: a subject, a predicate, and an object. Resources are represented by Uniform Resource Identifiers (URIs). A URI usually is unique identification key. The subject of RDF statements must actually be a resource. The object could be an RDF literal, that is a string, but an object could also be a resource. And the predicate expresses a relationship between the subject and the object. All RDF predicates must use a namespace to clarify their meaning. Many triples together form an RDF graph. A normative syntax for serializing RDF is RDF/XML. Also, there are other serializations formats. A popular syntax is N3 (Notation 3). Let's consider the statement "Da Vinci Code is written by Dan Brown", Figure 3.6 shows the common graph representation of this RDF statement

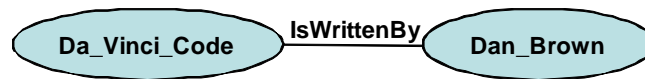


Figure 3.6: Graphical representation of an RDF statement

In this example the object (Dan.Brown) is resource such as the subject (Da.Vinci.Code) but could also be a literal. The RDF/XML representation of this statement is:

```
<rdf:Description about="Da_Vinci_Code" >
  <isWrittenBy>
    <rdf:Description about="Dan_Brown" >
      </rdf:Description>
    </isWrittenBy>
  </rdf:Description>
```

Da Vinci Code is a book and Dan Brown is a writer. The following code specifies the type of Da Vinci code and the type of Dan Brown.


```
<rdf:Description about="Da_Vinci_Code" >
  <rdf:type rdf:resource="Book" />
</rdf:Description>
```

```
<rdf:Description about="Dan_Brown" >
  <rdf:type rdf:resource="Writer" />
</rdf:Description>
```

RDF Schema (RDFS) is a language for knowledge representation. RDFS can be used to describe ontologies. Is a language for defining RDF types and describes taxonomies and hierarchy of classes and properties [8]. It also extends definitions for some of the elements of RDF, for example it sets the domain and range of properties and relates the RDF classes and properties into taxonomies using the RDFS vocabulary. Some of the RDFS futures that are used in building an ontology are *rdfs:Class*, *rdfs:subClassOf*, *rdfs:Property*, *rdfs:subProperty*, *rdfs:label*, *rdfs:comment* and many others.

The Extensible Mark-up Language (XML) is a W3C standard meta-language for describing document structures by tagging parts of documents. The tags are not fixed and they are defined according to the kind of information. XML documents have a structure, content and semantics and consist mainly of nested elements. The structure of an XML document is a tree. In order to describe types of XML documents the XML Document Type Definition (DTD) and XML Schema languages are used.

3.3.2 Advantages of RDF versus XML

The implementation of IAM is based on RDF and RDFS because RDF has more expressive power than XML and also can describes rich semantic information more compendious than XML. RDF is a data model which based on sets of interconnected statements that can successfully represents arguments and their relationships, whereas XML is based on documents. An argument and general a statement can use part of another (e.g. an argument can use a premise or a conclusion of another argument) and many related statements together built up a network of associated statements, a graph. RDF uses graph content,

while XML is based on tree structure that is less flexible for expressing metadata. Storing and representing arguments means making such graphs. In our argumentation system we construct networks of arguments. Thus, RDF graph is very flexible and more appropriate for representation of those networks than XML. RDF is used in cases that the representation of knowledge based on a tree structure is not sufficient. Moreover, for a graph of knowledge, the order of the nodes is not important and this is the main advantage of RDF over XML because when new statements are added dynamically to the repository, need hardly to worry about the order of the statements inserted. On the other hand, updating an XML document requires the maintenance of the tree structure and the node's order. The order of RDF properties does not matter, while it does in XML. Also, the metadata for a resource many times reside outside of the referring RDF document. Contrariwise, nodes that refer to an XML schema are located within the same XML document, in a specific location within the structure of that document. So, XML does not offer external relationships between two or more entities. Also, RDF is for expressing semantic meaning, while XML is for expressing syntactic meaning. As a result, RDF is more powerful, more expressive and offers more efficient representation and querying than XML.

3.3.3 Implementation in RDF and RDFS

In order to implement the IAM in RDF and RDFS we used Protégé [4]. Protégé was developed at Stanford University

We represented is-a relationships between classes and their subclasses of the IAM as well as the relations between classes by using the graphical interface of Protégé. Protégé exports the RDFS code and is platform for building ontologies for Semantic Web.

For example, Figure 3.7 represents that the class “Factual Inference Making” is subclass of the class “Inference Making” and also is subclass of the class “Factual Argumentation” if we assume that “am” is the namespace of our argumentation model, the RDFS code that is exported from Protégé is:

```
<rdfs:Class rdf:about="&am;Factual_Inference_Making" rdfs:label="Factual_Inference_Making">  
  <rdfs:subClassOf rdf:resource="&am;Factual_Argumentation"/>  
  <rdfs:subClassOf rdf:resource="&am;Inference_Making"/>
```

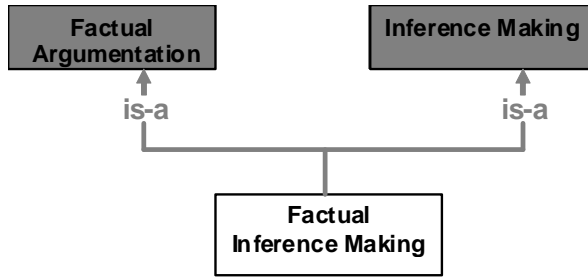


Figure 3.7: Is-a relationship between classes

```
</rdfs:Class>
```

The relations between the classes are represented as class attributes. We used the RDF attributes domain and range in order to implement IAM restrictions on the relations. For example the exporting code from Protégé for the relation “has_time_span” which has domain the class “E2.Temporal_Entity” and range the class “E52.Time-Span” is:

```
<rdf:Property rdf:about="&am;has_time_span" rdfs:label="has_time_span">
  <rdfs:domain rdf:resource="&cidoc;E2.Temporal_Entity"/>
  <rdfs:range rdf:resource="&cidoc;E52.Time-Span"/>
</rdf:Property>
```

As far as the sub-properties is concerned, they can implemented by Protégé too. For example let’s consider the property “has_belief_time”. It is sub-property of the property “has_time_span” and has the class “Belief” as domain and the class “E52.Time-Span”. The RDFS code is:

```
<rdf:Property rdf:about="&am;has_belief_time" rdfs:label="has_belief_time">
  <rdfs:domain rdf:resource="&am;Belief"/>
  <rdfs:range rdf:resource="&cidoc;E52.Time-Span"/>
  <rdfs:subPropertyOf rdf:resource="&am;has_time_span"/>
</rdf:Property>
```

The whole IAM can be found in Appendix.

Chapter 4

Antilogos: a system for factual argumentation based on IAM

4.1 Introduction

In this chapter we describe our system and we figure some screenshots from the interface. The argumentation system we have developed is a semantic web based system built on the IAM model. We name it Antilogos. The system connects the argumentation model represented in Chapter 3 with a domain ontology in order to represents the propositions. The users can add elementary inferences, observations and belief adoptions at any time. Antilogos automates and assists the formulation of the argumentation and its parameters (who, when, proposition parameters, beliefs) as much as possible. The users can monitor the current state of knowledge as well as that at any other instant in the past. A graphical interface presents what is currently believed about a proposition and why, its current belief value, how it is came up, if it is believed by observation or by adoption of a source or if it is believed because of an inference with another proposition (see Figure 4.31). Also, it informs the user about the history of each belief, if the belief has been changed, when it happened and why. Users can query the system by various useful parameters.

We suppose that users have the same knowledge base and the same belief in a proposition each time. However, the belief in a proposition can be changed at any time by any user, exactly as in Wikipedia (<http://en.wikipedia.org/wiki/Wikipedia>). When

a user introduces a new argument, the system checks the global consistency and constructs the largest possible composite inferences. In case that the new inference defeats another inference, the system runs an algorithm that changes all the beliefs that are affected directly or indirectly by the new inference. Basically, it resolves conflicts by turning beliefs to “UNKNOWN”. The latest argument is assumed to be stronger. Thus, the system uses the beliefs of the user’s argumentation to infer either new, different beliefs or the defeat of some inference. In other words, it makes an inference on the inferences. As such, the system acts as another user, and its algorithm and inferences are regarded to be trusted by all users. It implements the notion of global awareness of all facts in the system. Antilogos does not aim to replace the user and does not takes decisions instead of the user (it not decision making system). Antilogos is a system for the record and the representation of arguments. It informs the user about the provenance of the knowledge in order to able to understand who says what and why. In case of conflicts and inconsistencies, Antilogos helps the users by detecting and resolving them.

We studied a real published archaeological example about the natural mummy that became known as “Otzi, the Iceman”, we collected the information and we selected real arguments and counterarguments that have been created through time by the researches which work on the mummy. The principal source for this example was the official site [3]. The system successfully demonstrates this rich set of facts and arguments. The possible states of affairs for the formulation of propositions are defined by the CIDOC CRM ontology [26]. It is a complex, real example of argumentation, which is quite interesting because of the multiple interpretations and archaeological theories stated, published, re-examined and argued over through years; it is a complicated and rich example of argumentation used in a scientific community, showing how different scientific theories based on facts have been developed and changed in time. It comprises many individual facts that contributed to several radical knowledge revision cycles about the social role and the circumstances of death of the mummified person.

The system so far confirmed that the respective discourse can convincingly be represented by the IAM and, under adequate specialization, be turned into an operational information system. It was not the intention to proof its ergonomic adequacy for tracing argumentation, but for testing the completeness and adequacy of the IAM to represent argumentation as

an integrated theory. However, is a tool for human argumentation, the interface is simple and clear and the navigation is easy. The propositions and the arguments are represented in natural language and every user can monitor the discussion and understand from when and why something is believed and who posted it. Various improvements in the interface of the system were conducted after the useful advises of an archeologist which used and navigate the system. The interface does not confuse the user with unrequited information or with information that is targeted to or understandable only form specific group of users.

Section 4.2 describes the assumptions under which the system is implemented and how it restricts the model. In Section 4.3 we present the architecture of the system, the data flow and we analyze our technical choices. Section 4.4 describes the components of the system in depth. We discuss the system functionality through examples an we explain what happens behind the screen and how the system behaves. Finally Section 4.5 introduces an example of the use of the system and illustrates screen shots from the system as the user interacts with it.

4.2 Assumptions

The system works under constraints and rules as they are defined by the IAM. The model has been developed as a theory, not as a functional database schema. It leaves for instance open, when an Actor takes notices of an argument and a belief change. It would be impractical for a first proof of concept to introduce into all communications of users with the system belief adoption activities. Further, the IAM makes no specific choice of a belief value system. Therefore, this implementation has restricted the IAM and its use to the following assumptions:

1. All actors use the system as if they were one person (one group in general agreement).

We assume that every belief and every argument that someone introduce to the system, is acceptable from all the other actors at least until the end of the inference making process. Thus, all Actors act like one single actor. However if an actor believes that the argument is wrong, s/he could make another argument that invalidates the previous one.

2. Every fact in the system is known to everybody. This means that the users are prompted with changes are all aware for every new argument that is uploaded to the system.
3. As in a good discussion round, an actor can only question an argument made known to the system by making another argument. Until now we presume that all actors accept each argument, so there is no “voting” for keeping actors who belief the argument and those who don’t.
4. The belief value system is restricted to “TRUE”, “FALSE” and “UNKNOWN”.
5. The implementation computes a connection of the argumentation to an ontology in order to represent propositions as instances of possible states of affairs. The ontology is configurable and extensible in the system.

4.3 System Description

In this section we give an overview of our system. We describe the architecture of the system and the software technologies and we explain why we decide to choose these technologies.

4.3.1 System Architecture

The architecture of the system was structured in three layers as shown in Figure 4.1. These are as follow:

1. The representation layer: the Web-based user interface, the front-end of the system. The interface is interactive and the content of the most pages changes dynamically. It provides to the users a clear view of the arguments and the operations. The system functionality is invoked with simple user actions, such as buttons, selections and links. In order to develop the web pages we used HTML, JSP and CSS.
2. The behavioural layer: the main components of Antilogos are located in the behavioural layer. This component implements the logic of the system. This layer is developed in Java. In the RDF Generator component transforms the arguments that

a user introduced from the webpage, in RDF statements according to IAM and connects the argument with instances of the domain ontology. The Query Processor component is responsible for building user queries in order to execute them. Also, in this layer many other important functions are executed such as the check for the global consistency, the estimation of the new knowledge state, the detection of conflicts etc.

3. The storage layer: includes the RDF database where the data are kept.

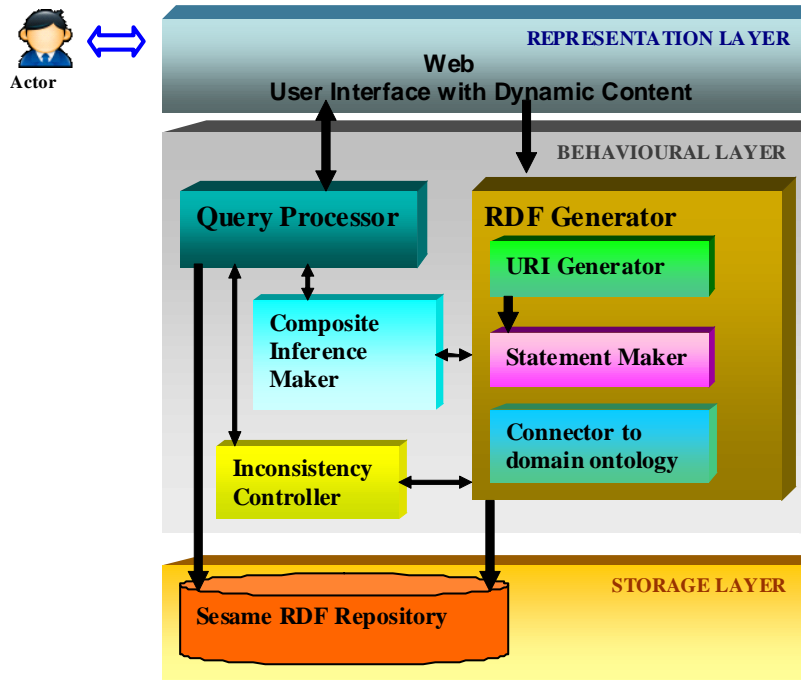


Figure 4.1: Architecture of the system.

To understand the system’s functionality, let us examine the data flow that happens behind the scenes as a user interacts with the Antilogos webpage. Figure 4.2 illustrates what happens when an actor introduce a new argument (black arrows) and what when the system accept another event from the webpages e.g. a request for some data (blue arrows).

As it is represented in Figure 4.2, in the case of the creation of a new argument, the actor inserts to the system the information and the system execute a sequence of processes. First, the generation of unique URIs for the elements of the argument and the creation of the appropriate statements through the java classes are take place. Then, the system checks for the global consistency of the arguments and update the old knowledge state. After that, the

with which client applications (or human users) can communicate over http.

Sesame provides many advantages. We describe some of them in brief. It offers a variety of options for storing RDF data. It can store data in main-memory, on disk, or in a relational database. We chose to use a Native RDF Repository that keeps and retrieves data directly to the disk. A native store it is slower than the in-memory store, since it has to access the disk. However, it is more scalable and efficient for large datasets that do not be kept in memory, since it isn't limited to the size of available memory. Also, Sesame provides a powerful Application Programming Interface (API) which can be used so that Java application can access manipulate and control Sesame RDF repositories. The API includes RDF parsers, writers, and RDF stores and it offers methods to add RDF, extract or delete RDF statements. Moreover, Sesame provides a lot of methods for creating and evaluating queries in various languages (SPARQL, SeRQL, RQL) and offers various output formats for query results (XML, HTML, RDF statements). Sesame also supports named graphs. It manipulates quads through the notion of *context*. It is a way to insert, edit, delete or group set of triples according to one (or more) identifier.

There are also other frameworks for storing and querying RDF, similar to Sesame like Jena [2]. Sesame, unlike Jena provides Workbench, which is a graphical tool to manage a Sesame server, and supports load, query, and explore operations via a user friendly web interface.

4.3.2.2 Query Language

In order to query the repository we used SPARQL [6] query language for RDF. It is an expressive language that is supported by Sesame and also it is a W3C recommendation. SPARQL contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions. SPARQL also supports extensible value testing and constraining queries by source RDF graph. It can query any number of named graphs. The results of SPARQL queries are results sets or RDF graphs. In our case, the most query results are results sets that are saved in several data structured.

A simple SPARQL query consists of the *select* clause that identifies the results variables, and of the where clause that provides the basic graph pattern to match against the data graph. The following example queries the repository to retrieve the all the factual beliefs.

The variable that will appear to the results is “?factualBelief”

```
PREFIX rdfs:< http://www.w3.org/2000/01/rdf-schema# >
PREFIX am:< http://protege.stanford.edu/am# >
PREFIX rdf:< http://www.w3.org/1999/02/22-rdf-syntax-ns# >
```

```
SELECT DISTINCT ?factualBelief
WHERE { ?factualBelief rdf:type am:FactualBelief }
```

Also, there is a way to limit a SPARQL query to a specific context suffice it to modify the query to select only from the context we are interested in. The above query is being modified in order to extract all factual beliefs from a specific context that is defined in *from* clause.

```
PREFIX rdfs:< http://www.w3.org/2000/01/rdf-schema# >
PREFIX am:< http://protege.stanford.edu/am# >
PREFIX rdf:< http://www.w3.org/1999/02/22-rdf-syntax-ns# >
```

```
SELECT DISTINCT ?factualBelief
FROM <context>
WHERE { ?factualBelief rdf:type am:FactualBelief }
```

A number of RDF languages are available as SeRQL and RQL. However they are not standards. Although SeRQL (Sesame RDF Query Language) is quite similar to SPARQL has not tool interoperability like SPARQL (Jena, Redland, 3Store, Sesame). Also, there are some other limitations of SeRQL include the missing of *order by* clause and no support for regular expressions. However, SeRQL has better features than RQL and also supports some operation (as MINUS and nested queries) that SPARQL does not. We had no need for these extra operations of SeRQL and that’s why we used SPARQL.

4.3.2.3 Generation of webpages

We used Java Server Pages (JSP) for generate the webpages. JSP is a server-side language that uses simple tag-based codes inserted into HTML and XML to produce dynamic and interactive web pages. JSP technology has a lot benefits for the developers.

Java Server Pages can be accessed from any web server and they are platform independent, meaning that by all rights they should appear exactly the same on every computer screen, no matter the platform. Moreover, it is provided a large range of API that increases the result oriented functionality. The JSP technology emphasizes the use of reusable components. These components can be combined or manipulated towards developing more purposeful components and page design. There is also a helpful tutorial on Sun's web site (http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/JSPIntro.html) and Java code fragments, called scriptlets, as well as JavaScript and Ajax (Asynchronous JavaScript and XML) can be encapsulated in a JSP page. These reasons and also my previous experience in JPS led me to choose JSP technology

4.4 System Components

This section describes the components that the base system consists of. As mentioned above Antilogos consists of three parts: the storage component, the behavioral component and the representation layer or the User Interface. We will describe the functionality of the system components according to the above classification

4.4.1 Storage Component

The repository of the system, as mentioned earlier, is a Sesame RDF repository and it works on the Apache Tomcat server. It allows uploading and storing RDF and RDFS statements as well as exporting them in different formats (RDF/XML, N3, N-Triples and others). Also, it offers the feature of querying the repository. Sesame is used a lot in the Semantic Web development domain.

4.4.2 Behavioral Component

The behavioral component constitute the mechanism that incorporates all the system intelligence. They are the functional modules that are invoked by user actions or by the interaction with other modules and interact both with the User Interface and with database. The main subcomponents are the Query Processor, the RDF Generator, the Composite inference Maker and the Inconsistency Controller.

4.4.2.1 Query Processor

This is an important component of the system. The query processor turns user's queries into SPARQL queries on the database. Every information displayed to the interface and concerning propositions and arguments that are stored in Sesame is result of the query processor. The system allows to the user to make a variety of queries. The search and the history of a proposition, the whole argument that a proposition came up, the actor and the time of the argument, the state of the knowledge are some of these queries. Query processor stores the queries results sets in data structures and pass them to Java functions and algorithms for processing in order to give back the right results that satisfy the user's request. The result are displayed to the browser. Further, query processor interacts and co-operates with other algorithms and functions of the behavioral component in order to give them the information they need to execute algorithms and functions. For example, the algorithm that checks for the global consistency, the algorithm that is executed for the creation of the composite inferences and the algorithm that detects the inconsistencies retrieve data form the database through query processor.

4.4.2.2 RDF Generator

RDF Generator includes all the procedures that take place in the transformation of the information that the system gets form the user into RDF statements and their storage in Sesame. Also, this module produces and stores all the RDF information that does not come only of the user but also of the same the system namely the construction of the composite inferences and the making of the inference defeat. In order to implement these procedures RDF Generator incorporate the following modules.

Automatic URI Generator. Every instance of a class that is to be uploaded to the repository has to have a unique URI as a key identifier. This module is responsible for the creation of these unique URIs. The system constructs URIs for instances both of the classes of IAM and of the classes of the domain ontology.

We use a concatenation of the model's namespace, the session identifier (ID), the name of the resource class and a string (that consists of the word "Instance" concatenated with a number). A session ID is a unique number that a Web site's server assigns a

specific user for the duration of that user’s visit. Every time a user visits the system’s Web site, a new session ID is assigned. Thus, when the user adds a new argument and a new instance of a class is to be uploaded to the repository, the session ID is retrieved from the server and a unique URI is constructed. For example the URIs for two new factual propositions are:

```
URI factualProposition1URI =
createURI(“am#78B7071E15563D6D9AB8AAF01F2AA1CE_FactualProposition_Instance1”) and
URI factualProposition2URI =
createURI(“am#78B7071E15563D6D9AB8AAF01F2AA1CE_FactualProposition_Instance2”)
```

In the same way URIs for the instances of the domain ontology are constructed. If we regard “crm” is the name space of the CIDOC ontology, a URI for an instance of the class E18.Physical_Thing:

```
URI physicalThing1URI =
createURI(“crm#78B7071E15563D6D9AB8AAF01F2AA1CE_E18.Physical_Thing_Instance1”)
```

RDF generator creates URIs also for the predicates and the resources classes of the instances needed in triples. These URIs are not unique, they are the RDF type of an instance and are the same for all the instance of a class. For example the resource URI for every factual proposition is:

```
URI resourceOfFactualProposition =
createURI(“am#Factual_Proposition”)
and the URI resource for a physical thing instance is:
URI resourceOfPhysicalThing =
createURI(“crm#E18.Physical_Thing”)
```

Statement Maker. Statement Maker gets the outputs of URI Generator in order to make and store RDF statements. After the URI creation the system formulates the incoming arguments according to the data that the user enters, and to the RDF Schema of IAM and also creates and store the propositions according to the domain ontology. The system creates triples that define not only the resource of an instance but also its relation with other instances.

Let assume that a user with username “boutsika” and password “kDtRbB”, inserts an elementary inference that applies the Inference Logic “Thing go off through time”. The system in that case will get the URIs from the URI Generator and will create statements such as the following:

```

URI actorURI = URIGenerator.getUniqueURI();
URI resOfActor = URIGenerator.getResourceClassURI();
URI username = URIGenerator.getResourcePredicateURI();
URI password = URIGenerator.getResourcePredicateURI();
(actorURI, RDF.TYPE, resOfActor);
Literal usernameOfActor = createLiteral("boutsika")
(actorURI, username, usernameOfActor)
Literal passwordOfActor = createLiteral("kDtRbB")
add(actorURI, password, passwordOfActor)

factualInferenceMakingURI = URIGenerator.getUniqueURI()
resourceOfFactualInferenceMaking = URIGenerator.getResourceClassURI()
URI made = URIGenerator.getResourcePredicateURI()
/*.... get all necessary URIs */
(factualInferenceMakingURI, RDF.TYPE, resourceOfFactualInferenceMaking)
(factualInferenceMakingURI, carryOutBy, actorURI)
(factualInferenceMakingURI, made, elementaryInferenceURI)
Literal activityDateTime = createLiteral(getCurrentDateTime)
(factualInferenceMakingURI, hasTimeSpan, activityDateTime)

elementaryInferenceURI = URIGenerator.getUniqueURI()
(elementaryInferenceURI, RDF.TYPE, resourceOfElementaryInference)
(premiseFactualPropositionBeliefURI, RDF.TYPE, resourceOfFactualBelief)
(premiseFactualPropositionURI, RDF.TYPE, resourceOfFactualProposition)
(premiseFactualPropositionBeliefURI, that, premiseFactualPropositionURI )
/*.... get all necessary URIs */

(premiseFactualPropositionBeliefURI, is, premiseBeliefValueURI )
(elementaryInferenceURI, usesAsPremise, premiseFactualPropositionBeliefURI )

(conclusionFactualPropositionBeliefURI, RDF.TYPE, resourceOfFactualBelief)
(conclusionFactualPropositionURI, RDF.TYPE, resourceOfFactualProposition)
(conclusionFactualPropositionBeliefURI, that, conclusionFactualPropositionURI )
/* .... create all necessary triples in this way */

```



```
(conclusionFactualPropositionBeliefURI, is, conclusionsBeliefValueURI )  
(elementaryInferenceURI, concludes_in, conclusionFactualPropositionBeliefURI )
```

```
(InferenceLogicURI, RDF.TYPE, resourceOfInferenceLogic)  
Literal inferenceLogicLiteral = createLiteral(“Thing go off through time”)  
(InferenceLogicURI, hasLiteral, inferenceLogicLiteral)  
(elementaryInferenceURI, applies, InferenceLogicURI )  
/* .... create all necessary triples in this way */
```

In this way the system makes RDF statements. In order to store a whole argument, the system creates a lot of such triples.

In some triples the object is a Literal. This is the sentence (or else the string) that the user entered (e.g as inference Logic).

Connector to domain ontology. As mentioned before, we assume propositions to be statements about particulars that can be formulated in terms of a formal ontology. So, we connected the IAM to instances of an ontology in order to represent propositions as instances of possible states of affairs.

Proposition are triples of the domain ontology. If a new proposition is to be uploaded, the system generates a URI for that proposition, but also creates unique URIs for the selected subject, predicate and object of the domain ontology and creates a triple. Thus, creating a proposition means creating an RDF triple with classes and properties of the domain ontology and connects this triple to the proposition’s URI of the model. In other words means making a statement about a statement. This is known as reification. Figure 4.3 presents how a statement is being object of another statement.

Sesame’s API does not have explicit support for reification. However, it supports named graphs. Sesame offers the statement context mechanism and statements are quads (subject, predicate, object, context). So context is a named graph. We used the context placeholder to record info about each statement of the ontology. The context of an ontology statement is the URI of the proposition. A drawback of RDF reification is that can identify only individual triples and also reifying a single triple takes at least four additional triples that is inefficienta and non-scalable. Contrarily context can identify groups of triples which is more useful and flexible.

In the above example (see Statement Maker in Section 4.4.2.2) the URIs `premiseFactualPropositionURI` and `conclusionFactualPropositionURI` are instances of the class `Factual Proposition`.

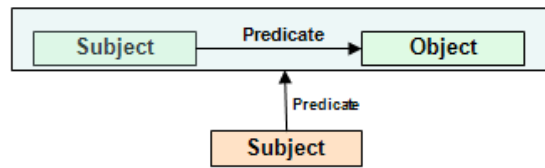


Figure 4.3: Reification.

However, in order to describe the possible state of affairs this module of the system creates also triples concerning instances of the domain ontology. If the factual proposition of the conclusion was “corpse of Otzi has_condition well maintained” and the resource class of the subject (“corpse of Otzi”) was `E18.Physical_Thing`, the resource class of the object (“well maintained”) was `E3.Condition_State` and the predicate was the property `P44F.has_condition` from CIDOC crm the system after the automatic URI generation inter alia would create the following quads:

```
(fromInstanceURI, RDF.TYPE, resourceOfFromClassURI, conclusionFactualPropositionURI)
(toInstanceURI, RDF.TYPE, resourceOfToClassURI, conclusionFactualPropositionURI)
(fromInstanceURI, has_condition, toInstanceURI, conclusionFactualPropositionURI)
```

In this way we achieve the connection of the arguments with the domain ontology. Figures 4.17, 4.18 and 4.19 illustrate how the user creates the new proposition “corpse of Otzi has_condition well maintained” by selecting the subject, the predicate and the object of the domain ontology from the user interface.

4.4.2.3 Composite Inference Maker

Users carry out inferences making that make only elementary inferences. If an elementary inference uses as premise (concludes in) the conclusion (premise) of another elementary inference then the system constructs a composite inference with initial premise the premise of the first (the last) elementary and with final conclusion the conclusion of the last (first) elementary. Thus, the system acts as an actor who carries out composite inferences. When an inference uses as premise or as conclusion the premise or the conclusion of another, we imply the same proposition belief (the same proposition and the same belief value). The beliefs in a composite inference have to be consistent. It is not possible to have a belief which is *TRUE* and *FALSE* at the same time in an inference. A composite inference consists of a lot of elementary inferences and may become very complex. A composite inference becomes a graph of inferences,

a network of valid elementary inferences with consistent beliefs. The Inference Belief of the composite inference believes all elementary inferences and also believes that all premises and conclusions are consistent. The inference logic of the composite is that defining the consistency of the proposition beliefs, e.g., that a proposition cannot be *TRUE* and *FALSE* at the same time. Algorithm 1 shows how the system makes a composite inference.

The algorithm takes as input the elementary inference that the user enters to the system. In lines 11-24, the algorithm searches all valid elementary inferences to find those which have as premise the conclusion of the incoming inference or as conclusion the premise of the incoming inference and also they are consistent. More than one elementary inferences may exist that can be linked to the new elementary and they may belong to one or more composite inferences. The system stores them temporarily in a map data structure in order to construct the largest possible composite inference. In lines 30-49, the algorithm constructs a new composite inference considering both the registered inferences and the incoming one. The algorithm uses the RDF Generator which constructs URI for the new composite inference as well as other RDF statements according to the model (such as the actor that carried out the inference, the time, the inference logic etc.) Iterating the entries of the map, the algorithm retrieves the elementary inferences that found in lines 11-24 and the RDF Generator creates and stores to the database the RDF statements in order to make the new composite to *consist of* these inferences. If these elementary inferences belonged to other composite inferences, the latter are removed since the system has already constructs the new composite containing all of them. Finally the system estimates the *initial premises* and the *final conclusions* of the new composite and the RDF Generator stores them also.

In Figure 4.4 we give an example of a composite inference. In this Figure we omitted the representation of inference making that made the elementary inferences and the composite one. Also we assume that all elementary inferences are consistent and they don't cause impossible world and also that all of them apply an inference logic. We presume that the premise of elementary inference 1 was concluded from an observation or an adoption. In this example when the insertion of elementary inference 2 that uses as premise the conclusion of the elementary inference 1, the system constructs a composite inference which has as initial premise the premise of elementary inference1 and as final conclusion the conclusion of elementary inference 2. When the elementary inference 3 is introduced, the system updates the final conclusion of the composite inference and keeps the same initial premise. Then, elementary inference 4 concludes at the conclusion of elementary inference 1 and uses as premise Premise 4 and the system renews the initial premise and finally the initial premises of the composite are Premise

Algorithm 1 CompositeInfernceMaker(*elementaryInference*)

```

1: /* elementaryInference is the elementary inference that the user introduced */
2:
3: Q ← newQueryProcessor()           ▷ Q is a Query Processor object
4: RDFGenerator ← newRDFGenerator()   ▷ RDFGenerator is an RDF Generator
5:                                       ▷ object
6: map ← newHashMap()                 ▷ HashMap with keys elementary Inferences and
7:                                       ▷ values lists with composite inferences that they belong to
8: premiseList ← Q.getPremisesOfElementaryInference   ▷ the premises of the elementary
9: conclusionList ← Q.getConclusionsOfElementaryInference   ▷ the conclusions of
10:                                       ▷ the elementary
11: for (each premise ∈ premiseList) do
12:   inferencesWithConclusion ← Q.getValidInferencesWithConsistentConclusion(premise)
13:   for (each elementary ∈ inferencesWithConclusion) do
14:     compositesListContainsInference ← Q.getCompositesConsistOf(elementary)
15:     map.put(elementary, compositesListContainsInference)
16:   end for
17: end for
18: for (each conclusion ∈ conclusionList) do
19:   inferencesWithPremise ← Q.getValidInferencesWithConsistentPremise(conclusion)
20:   for (each inference ∈ inferencesWithPremise) do
21:     compositesListContainsInference ← Q.getCompositesConsistOf(elementary)
22:     map.put(elementary, compositesListContainsInference)
23:   end for
24: end for
25:
26: if (map ≠ null) then
27:   /* RDFGenerator creates the URIs and RDF statements for the composite inference*/
28:   newComposite ← RDFGenerator.constructCompositeInference()
29:   newComposite ← RDFGenerator.makesCompositeConsistsOf(elementaryInference)
30:   for (each elementary ∈ map) do
31:     compositeList ← map.get(elementary)
32:
33:     if (compositeList = null) then           ▷ the elementary is not belong to any composite
34:       newComposite ← RDFGenerator.makesCompositeConsistsOf(elementary)
35:
36:     else
37:       for (each composite ∈ compositeList) do
38:         elemInferences ← Q.getElementariesOfComposite(composite)
39:         for (each elemInference ∈ elemInferences) do
40:           newComposite ← RDFGenerator.makesCompositeConsistsOf(elemInference)
41:         end for
42:         removeOldComposite()
43:         initialPrem ← estimateinitialPremises()
44:         finalConc ← estimatefinalConclusions()
45:         newComposite ← RDFGenerator.addsInitialPremiseToComposite(initialPrem)
46:         newComposite ← RDFGenerator.addsInitialFinalConclusionToComposite(finalCon)
47:       end for
48:     end if
49:   end for
50: end if

```

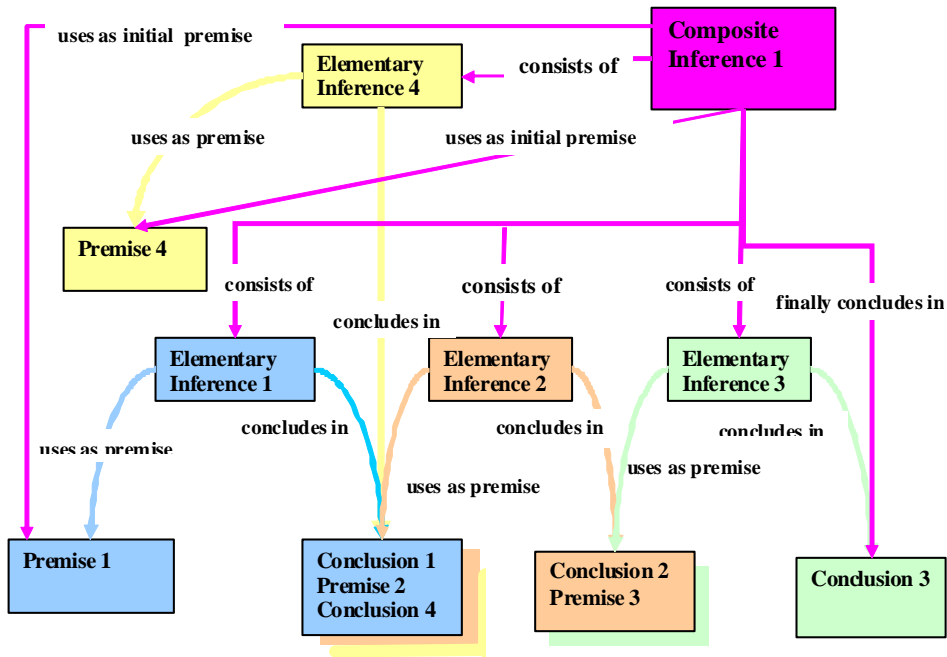


Figure 4.4: An example of a composite inference

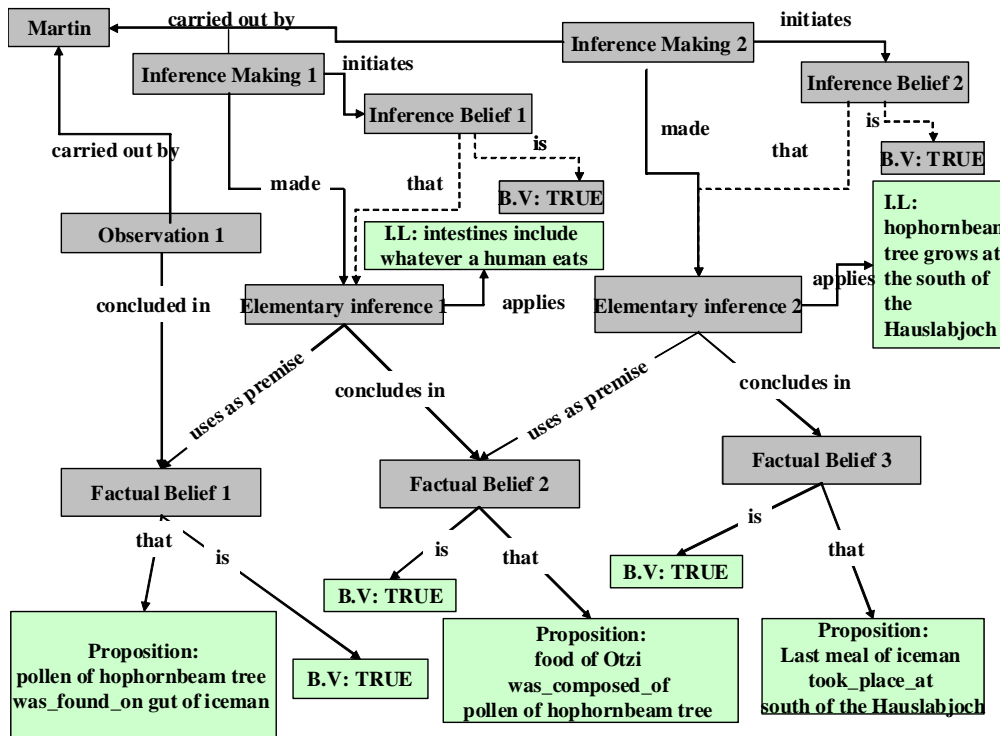


Figure 4.5: First arguments from the history of Alpine iceman

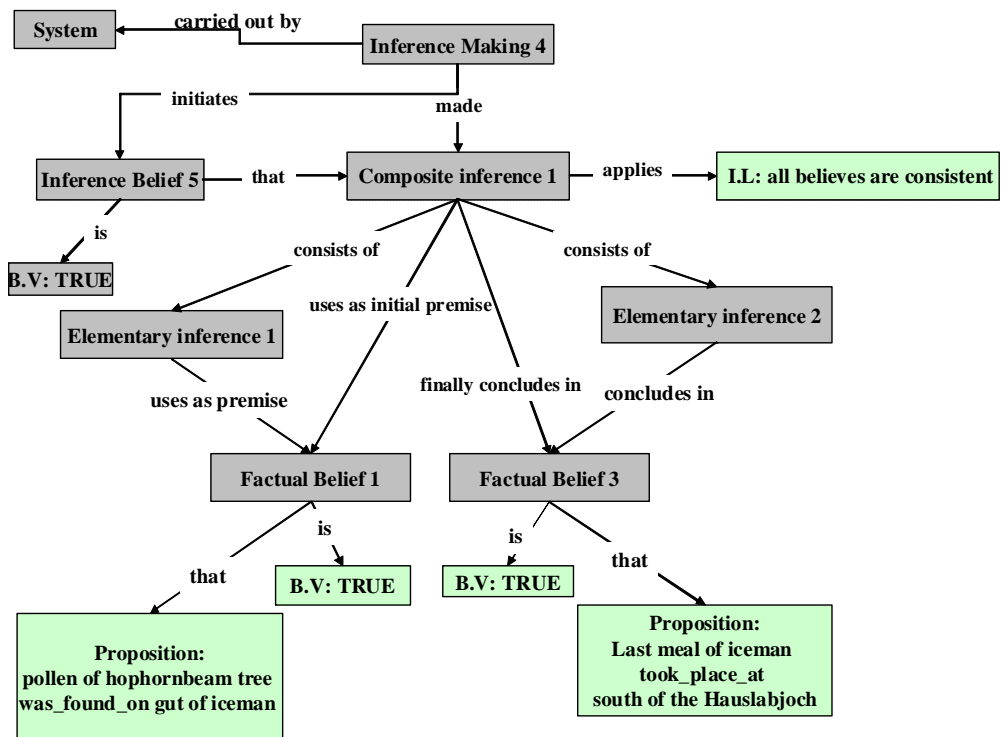


Figure 4.6: Composite inference from the history of Alpine iceman

1 and premise 4 and the finally conclusion in Conclusion 3.

Let's see an example with real facts from the story of Otzi the iceman. Researches using x-rays analysis found what iceman ate before he died. X-rays revealed not only the content of iceman's gut but also how many hours he eta his last meal. His gut has state of a gut after eight hours from meal. Inter alia his gut contains pollen of the hophornbeam tree. So, we have the first argument which is an observation that contains this fact. Form this fact they concluded that food of iceman contained pollen from this tree. The tree hophornbeam grows up at the south of the Hauslabjoch. Thus, the researchers concluded that the last meal of Otzi took place at the south of the Hauslabjoch.

In Figure 4.5 we notice that elementary inference 1 and elementary inference 2 construct together a composite inference since the second uses as premise the conclusion of the first (see Figure 4.6 which depicts this composite).

4.4.2.4 Inconsistency controller

The system is able to detect and resolve inconsistencies. Inconsistencies may be exists when two inferences conclude in a belief with the same associated proposition but with a different belief value or when an inference has produced a conclusion which comes into contradiction with a later observation.

When an inconsistency happens, the system carries out an Inference Defeating. As already mentioned Inference Defeating is an inference on another inferences (see Section 3.2.7). It has conclusions and premises. As conclusion must have at least one (negative) inference belief. The process of an Inference Defeating starts after an inference of a user which may cause inconsistencies or impossible world situations. If an inference is defeated, its effect on its conclusions should be "undone". We can describe this by turning the belief values of the conclusions of the defeated inference to "UNKNOWN", which we regard as a formalization of "questioning" the proposition and we change the belief values of conclusions that are not supported by other arguments.

Algorithm 2 is executed every time that a new argument is introduced in order to check if this argument causes a conflict. If so, the algorithm solves the conflict by defeating recursively all the associated inferences and changing to "UNKNOWN" the beliefs of the conclusions that not supported by other arguments.

Let us give some qualifications in order to apprehend the algorithm:

- Each belief (premise or conclusion) stands for an interval of time whereon we give a *begin date* and an *end date* to each belief. When a new belief is introduced RDF Generator sets as *begin date* the date and time that the introduction happened (this is the date and time that the system has at that time) and no *end date*. When we terminate a belief, we imply that we set *end date* to that belief. Also, every change in the belief value of a belief corresponds to a new belief and thus indicates that the old belief is terminated. A belief value of a belief changes when an actor changes directly its belief value or when the system change its belief value indirectly as a result of reasoning. A terminated belief is not applied.
- Inference Belief of an inference since is also a belief and has *begin date*, *end date* and *Belief Value*. When the user makes an Inference Making it *initiates* an Inference Belief. RDF Generator gives a positive belief value (“TRUE”) to this Inference Belief. As *begin date* we regard the date time that the inference is introduced in the system. When an inference is defeated RDF Generator sets *end date* to its inference belief and creates a new inference belief with negative belief value (“NOT TRUE”) and makes an Inference Defeat that concludes in that belief. Terminate an Inference Belief means set *end date*.
- A valid Inference is an Inference whose Inference Belief has positive value and it has no *end date*.
- A belief has supporters when there are valid inferences that conclude in it.

Algorithm 2 takes as input a list which contains the beliefs that the new argument concludes in, searches all the registered and applied beliefs with the same proposition. If there is an applied belief with same proposition and different belief value, the algorithm implies that it detects an inconsistency and terminates this belief.

Then the algorithm calls algorithms 3 and 4 in order searches and find all the inferences that have the terminated belief as premise and as conclusion respectively. These algorithms change to “UNKNOWN” the beliefs of the conclusions that not supported by other arguments and in case the that conclusion has supporters, the algorithms just terminate the inference belief of the inference with the terminated conclusion.

Algorithm 3 takes as input a list with all the inferences that have as premise the belief that terminated in Algorithm 2. For each inference of the list checks if its premise beliefs are supported by other inferences (line 10).

- If so, the algorithm terminates the inference belief of that inference and creates a new inference belief with negative belief value (lines 13-14). By calling RDF Generator creates a

Algorithm 2 InconsistencyController(conclusionBeliefsList)

```

1: /* conclusionBeliefsList is the list with the new incoming conclusion beliefs */
2:
3:  $Q \leftarrow newQueryProcessor()$ ; ▷  $Q$  is a Query Processor object
4:  $RDFGenerator \leftarrow newRDFGenerator()$ ; ▷  $RDFGenerator$  is an RDF Generator
5: ▷ object
6: for (each  $belief \in conclusionBeliefsList$ ) do
7:    $beliefValue \leftarrow Q.getBeliefValueOfBelief(belief)$ 
8:    $similarBeliefsList \leftarrow Q.getAllBeliefsWithTheSameProposition(belief)$ 
9:
10:  /* similarBeliefsList contains beliefs with the same proposition */
11:  for (each  $simiralBelief \in similarBeliefsList$ ) do
12:     $differentBeliefValue \leftarrow Q.getBeliefValueOfBelief(simiralBelief)$ 
13:    if ( $beliefValue \neq differentBeliefValue$ ) then ▷ Detected inconsistency
14:
15:       $RDFGenerator.terminateBelief(simiralBelief)$ 
16:       $inferecesList1 \leftarrow Q.getValidInferencesConcludesInBelief(simiralBelief)$ 
17:       $invalidateConclusion(inferecesList1)$ 
18:       $inferecesList2 \leftarrow Q.getValidInferencesUsesAsPremiseBelief(simiralBelief)$ 
19:       $invalidPremise(inferecesList2)$ 
20:    end if
21:  end for
22: end for

```

new inference defeat and construct the RDF statement which implies that this inference defeat concludes in that inference belief with negative value.

- If the conclusion belief of the an inference is not supported by others the algorithm terminates this belief and creates new belief with the same proposition and “UNKNOWN” belief value (lines 19-21). Also, terminates the inference belief and creates a new negative one. The inference defeat now concludes in both of new beliefs. Then the algorithm has to check if the terminated conclusion belief is used as premise by other inferences. Thus, the same process is followed recursively until there is no associated inference undefeated (lines 27-28).

Algorithm 4 is similar to 3. Takes as input a list with all the inferences that have as conclusion the belief that terminated in algorithm 2. For each inference checks if the its conclusion are supported by other inferences.

Let’s continue the above example (see Section 4.4.2.3) and add more arguments. After more analysis in his gut found that he had eta red deer meat but since deer don’t leave at Hauslabjoch and they leave northern close to the place that iceman was found. Thus, researchers conclude that iceman could not have taken his last meal at Hauslabjoch. Figure 4.7 illustrate this chain of inferences. In Figure 4.7 researches concluded in a factual belief 5 that has the same proposition with the factual belief 3 (the conclusion of elementary of Figure 4.5) but with

Algorithm 3 invalidatePremises(inferenceList)

```

1:  $Q \leftarrow \text{newQueryProcessor}();$  ▷  $Q$  is a Query Processor object
2:  $RDFGenerator \leftarrow \text{newRDFGenerator}();$  ▷  $RDFGenerator$  is an RDF Generator
3: ▷ object
4:
5: if ( $\text{inferenceList}$  is empty ) then
6:   return
7: end if
8: for (each  $\text{inference} \in \text{inferenceList}$ ) do
9:    $\text{conclusionsList} \leftarrow Q.\text{getConclusionsOfInference}(\text{inference})$ 
10:   $\text{boolean supported} \leftarrow \text{checkIfExistSupporters}(\text{conclusionsList})$ 
11:
12:  if ( $\text{supported} = \text{true}$  ) then
13:     $RDFGenerator.\text{terminateInferenceBeliefOfInference}(\text{inference})$ 
14:     $\text{negativeInferenceBelief} = RDFGenerator.\text{createNewNegativeInfernceBelief}()$ 
15:     $RDFGenerator.\text{inferenceDefeatConcludesIn}(\text{negativeInferenceBelief})$ 
16:
17:  else
18:    for (each  $\text{conclusion} \in \text{conclusionsList}$ ) do
19:       $RDFGenerator.\text{terminateBelief}(\text{conclusion})$ 
20:       $\text{proposition} \leftarrow Q.\text{getPropositionOfBelief}(\text{conclusion})$ 
21:       $\text{newBelief} = RDFGenerator.\text{createNewSimilarBelief}(\text{proposition}, \text{"UNKNOWN"})$ 
22:
23:       $RDFGenerator.\text{terminateInferenceBeliefOfInference}(\text{inference})$ 
24:       $\text{negativeInferenceBelief} = RDFGenerator.\text{createNewNegativeInfernceBelief}()$ 
25:       $RDFGenerator.\text{inferenceDefeatConcludesIn}(\text{negativeInferenceBelief}, \text{newBelief})$ 
26:
27:       $\text{newInferncesList} \leftarrow Q.\text{getValidInferencesUsesAsPremiseBelief}(\text{conclusion})$ 
28:       $\text{invalidPremise}(\text{newInferncesList})$ 
29:    end for
30:  end if
31: end for

```

Algorithm 4 `invalidateConclusion(inferenceList)`

```
1:  $Q \leftarrow \text{newQueryProcessor}()$ ; ▷  $Q$  is a Query Processor object
2:  $RDFGenerator \leftarrow \text{newRDFGenerator}()$ ; ▷  $RDFGenerator$  is an RDF Generator
3: ▷ object
4:
5: if ( $\text{inferenceList}$  is empty ) then
6:   return
7: end if
8: for (each  $\text{inference} \in \text{inferenceList}$ ) do
9:    $\text{premiseList} \leftarrow Q.\text{getPremisesOfInference}(\text{inference})$ 
10:  ▷ search for inferences concludes in premiseList
11:  boolean  $\text{supported} \leftarrow \text{checkIfExistSupporters}(\text{premiseList})$ 
12:  ▷ we except the current inference
13:
14:  if ( $\text{supported} = \text{true}$  ) then
15:     $RDFGenerator.\text{terminateInferenceBeliefOfInference}(\text{inference})$ 
16:     $\text{negativeInferenceBelief} = RDFGenerator.\text{createNewNegativeInfernceBelief}()$ 
17:     $RDFGenerator.\text{inferenceDefeatConcludesIn}(\text{negativeInferenceBelief})$ 
18:
19:  else
20:    for (each  $\text{premise} \in \text{premiseList}$ ) do
21:       $RDFGenerator.\text{terminateBelief}(\text{premise})$ 
22:       $\text{proposition} \leftarrow Q.\text{getPropositionOfBelief}(\text{premise})$ 
23:       $\text{newBelief} = RDFGenerator.\text{createNewSimilarBelief}(\text{proposition}, \text{"UNKNOWN"})$ 
24:
25:       $RDFGenerator.\text{terminateInferenceBeliefOfInference}(\text{inference})$ 
26:       $\text{negativeBelief} = RDFGenerator.\text{createNewNegativeInfernceBelief}()$ 
27:       $RDFGenerator.\text{inferenceDefeatConcludesIn}(\text{negativeBelief}, \text{newBelief})$ 
28:
29:       $\text{inferecesList1} \leftarrow Q.\text{getValidInferencesUsesAsPremiseBelief}(\text{premise})$ 
30:       $\text{invalidPremise}(\text{inferecesList1})$ 
31:    end for
32:  end if
33: end for
```

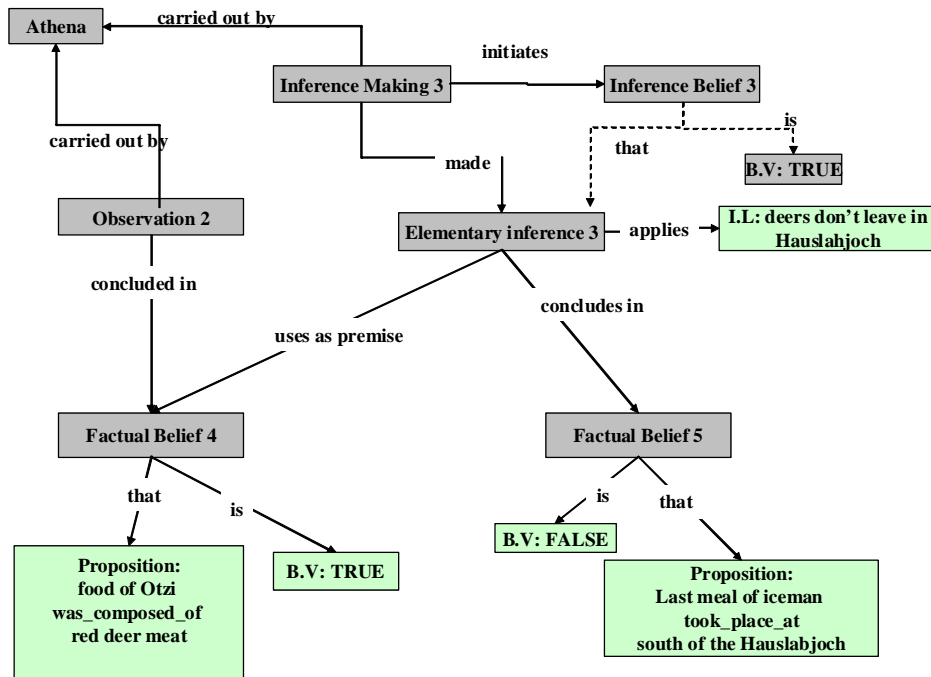


Figure 4.7: Second arguments from the history of Alpine iceman

different belief value. So, here we have a situation that an inference defeats another.

We can describe this by turning the belief values of the conclusions of the defeated inference to “UNKNOWN”. Our algorithm restricts this change only to belief values of conclusions that are not supported by other arguments. Distinctly in the above example, we have to examine if the factual belief that used as premise of elementary inference 2 (Figure 4.5), is supported by other valid elementary inferences (this mean if it is conclusion of other inferences which is valid). The algorithm finds that the premise (factual belief 2) of elementary inference 2 is also conclusion of another inference (elementary inference 1). Factual belief 2 has a supporter, for this reason, the algorithm does not change its belief value. In lieu thereof, it invalidates the inference making 2. Invalidate an inference means setting end date to its inference belief. The result is an inference defeating, another inference making made by the algorithm. This inference uses as premise the factual belief 5 and concludes to an inference belief with value “NOT TRUE”. Figure 4.8 shows graphically this inference.

Another kind of inference Defeating that the system made is when the conclusion of an inference is questioned and the premise does not supported by other inferences (see Section 3.2.9). In this case the system question both the premise and the whole inference. This happens when an inference brings an “impossible world”, to our algorithm this happens when a premise belief

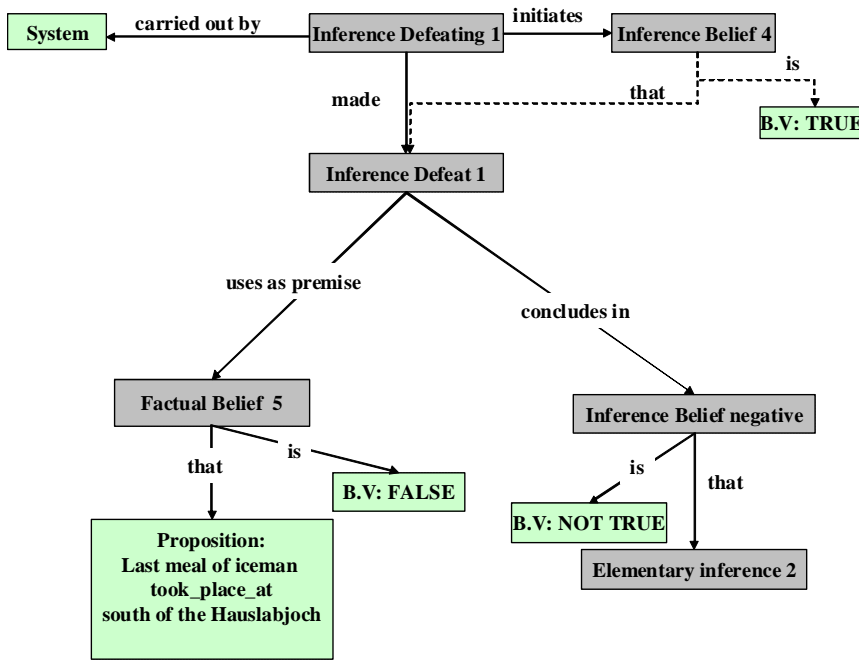


Figure 4.8: Example of Inference Defeat

appearing both in premise and in conclusion with different belief values at the same time.

Now we show graphically (Figures 4.9, 4.10) with an example with arguments from the history of Otzi how could this happens. From the analysis of Otzi’s guts researchers found that he had consumed wheat. Since the harvest of wheat occurs in autumn, they conclude that he died in autumn and thus it is impossible to had consumed a vernal product.

However as already mentioned, they had found from the x-rays analysis that his gut contained also pollen of hophormbeam tree that blooms only in spring. So it is impossible to had died in autumn. In this case we have an “impossible word” situation. We begin with the belief that “death of Otzi has_time-span autumn” is TRUE and we conclude that in the same time “death of Otzi has_time-span autumn” is FALSE. After the elementary inferences 4,5 and 6 in Figures 4.9, 4.10 the system makes a composite inference that consists of the three of them. However, when the elementary inference 7 is introduced it does not be incorporated in the composite inference because its conclusion is not consistent with a previous premise. The system detects this inconsistency turns all the associated values to “UNKNOWN” and invalidates the associated inferences by making an inference defeat. Figure 4.11 presents the results of the algorithm.

And they assumed that he had eta wheat harvested in the year before since grain can be stored

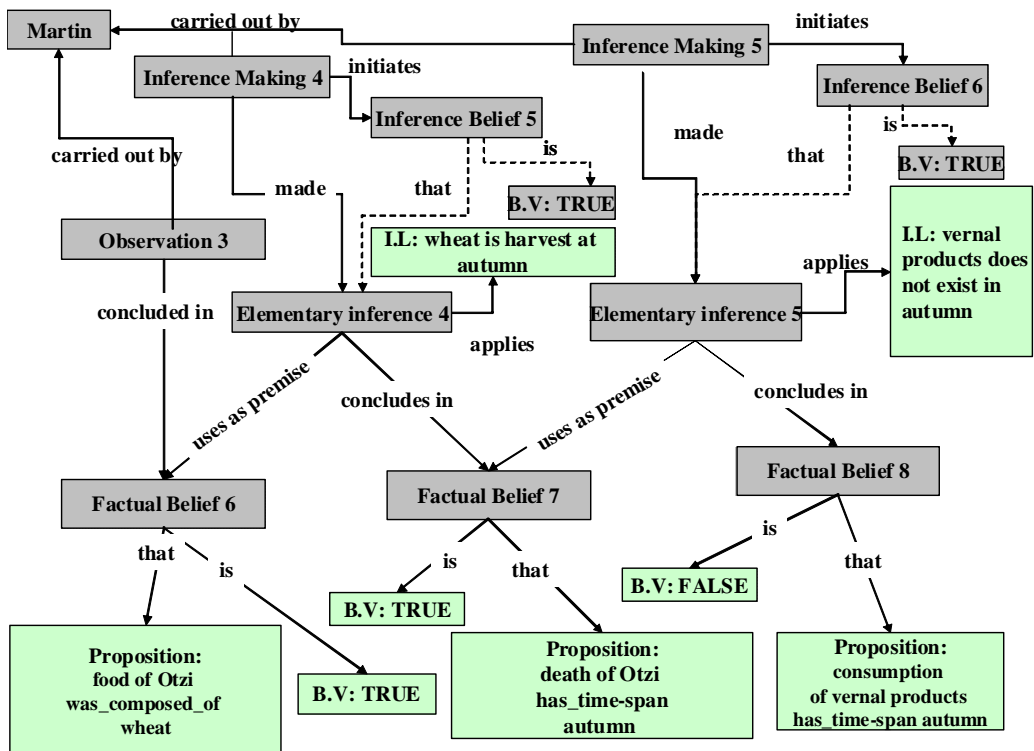


Figure 4.9: Third arguments from the history of Alpine iceman

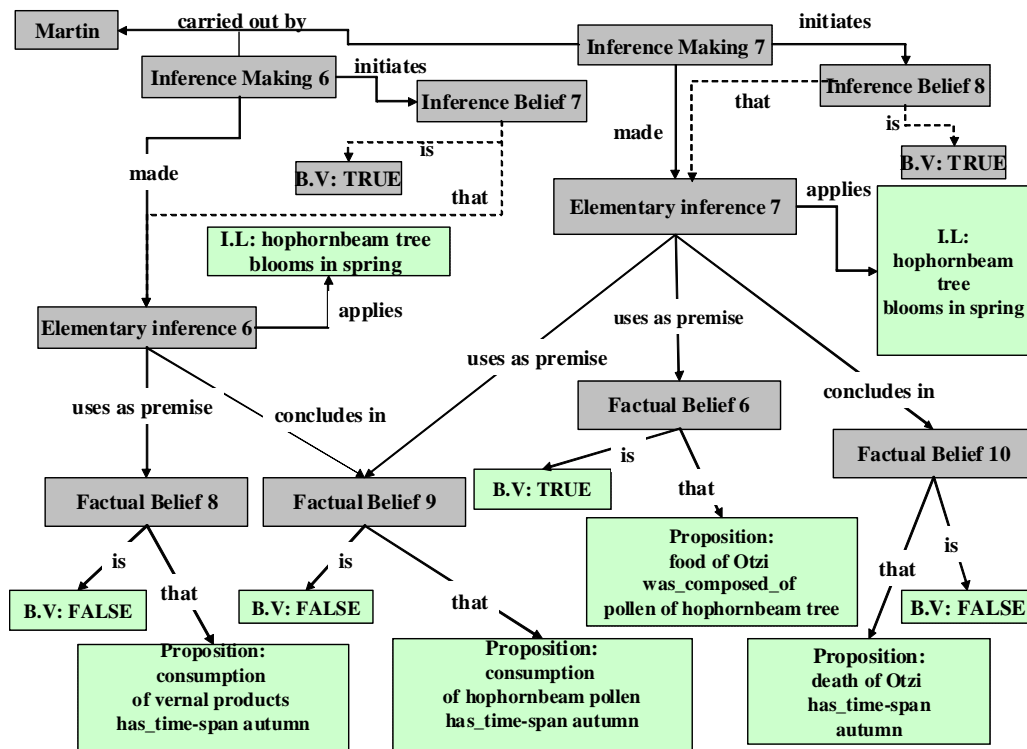


Figure 4.10: Fourth arguments from the history of Alpine iceman

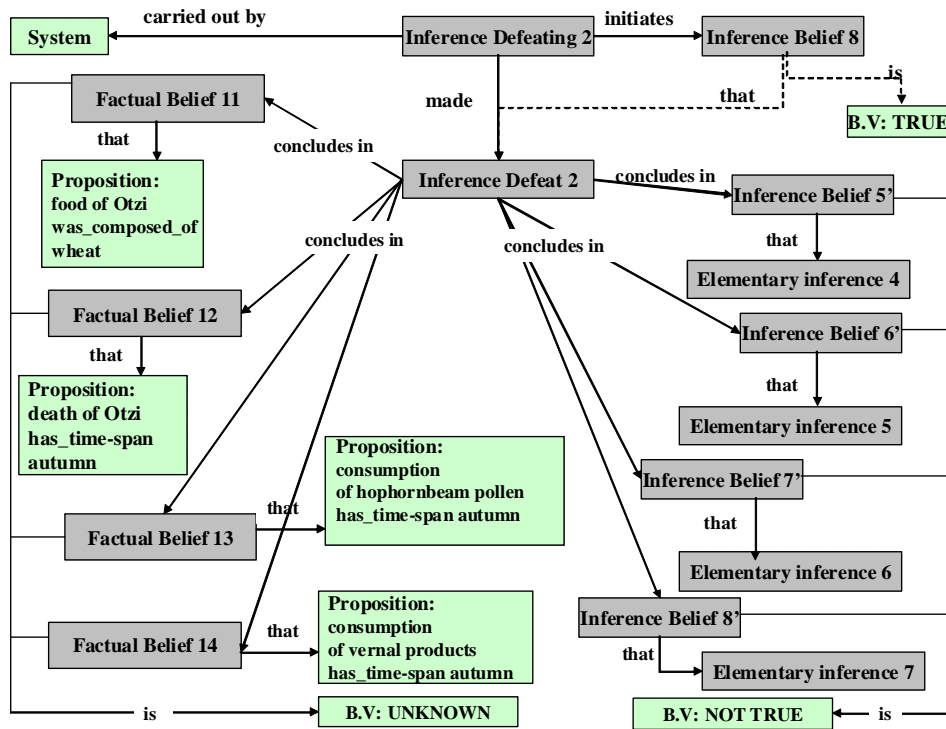


Figure 4.11: Second example of Inference Defeat

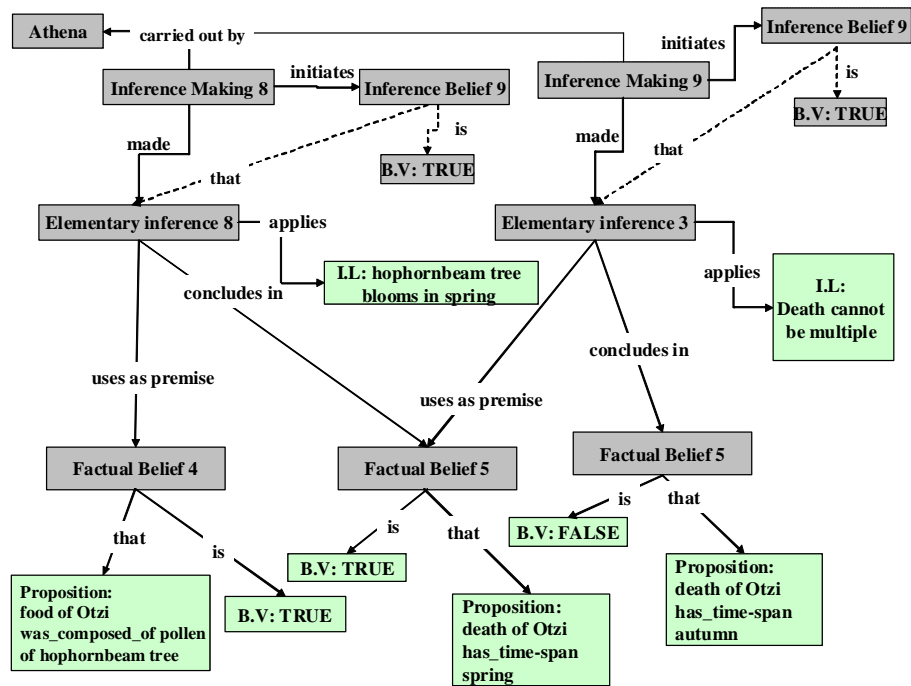


Figure 4.12: Final arguments from the history of Alpine iceman

and he died in spring (see Figure 4.12).

In Section 4.5 we demonstrate this example and we represent screen shots from the system step by step.

4.4.3 User Interface

In this section we describe the components that the user will interact to store and retrieve arguments, to monitor the knowledge states through time, as well as to search for a proposition and its history in the repository. The users will invoke system's functionality with simple actions, such as buttons, selections and links. The description of the components that follows will include the description of their functionality and a list of the functions that the component will provide to the user as actions invoked by available interface.

4.4.3.1 Register and Login

In our system we assume that each time everyone has the same belief. Thus, it seems not so necessary to us to keep data for each user. However, it is useful to know who said something so everyone who wants to use the system has to register. The system provides an interface with an appropriate form in order user enter some personal and contact information (first name, last name, username, password, e-mail and telephone). The system when indicates errors such as not filling in a required field informs with helpful messages the user for them. Every registered user is an instance of the class Actor of IAM. Figure 4.14 in Section 4.5 depicts the register form.

After the registration the user is able to login the system at any time entering his/her username and password in a form (see Figure 4.13 in Section 4.5).

4.4.3.2 Create Issue

For the needs of the implementation we introduce the notion of Issues. Any discussion begins with someone posting an Issue which regarded to be a general problem or question. A new Issue is a String in reality, a title for the discussion such as "History of Otzi the iceman" An issue is motivation of argumentation. An actor can insert an issue and may also post an argument. In the model an Issue is an instance of the class Question and for

the moment we regard that is a title for the arguments. The first argument of the Issue is an Observation or an Adoption. It is make no sense if the first argument is an inference because its premise can not be reliable as they are not facts.

4.4.3.3 Create an Argument

In Antilogos users can create and add new arguments, based on one of three types of argumentation. The types of the arguments are:

1. Inference Making
2. Observation
3. Belief Adoption

At any case, the users select the type of the argument they want to insert and automatically the system displays the appropriate form for the selected type of argument. For each argument users have to make one or more new propositions or even they can choose exiting propositions and associate them with belief values. For the belief values there is also a drop down menu with values “TRUE”, “FALSE” and “UNKNOWN”. In order to make a new proposition in an argument, users have to insert the subject the predicate and the object. The system offers drop down menus for this purpose (see Figure 4.22). Users can also select an proposition form a list of the already existing propositions and reuse it. By doing this, they can support or reject an existing proposition. Also, in this way the arguments can be linked to each other and the system constructs chains of arguments.

Moreover, some other information they are able to insert for each argument. For example, the Inference Logic that is applied to an inference they make. For the inference logic of the inference there is a text field. Alternatively, the user can select older inference logic. The inference logic is a string that is stored as a literal to Sesame. Figure 4.22 (Section 4.5) shows what the system displays when the user selects to add an inference making.

In case of an observation the user can add the place of the observation and the physical thing they observed, if additional evidence are exists the can commit them too. Figure 4.16 (Section 4.5) illustrates the form of an observation. If the user adopts the belief of someone he/she has to introduce the source of the adoption.

Another optional text field is available for inserting a comment about a triple. The comment may be a small text, a note that gives more information for the proposition. n

4.4.3.4 Knowledge state through time

The system holds the state of knowledge at any time. The state of knowledge change every time a new activity is uploaded to the system. From one state of knowledge to another a new argument may be added, one or more inferences may be invalidated and the belief values of some proposition may be changed. Thus, at a point time a proposition may be TRUE and at the next point may be FALSE or UNKNOWN and then an actor may turn it to TRUE again etc. The changes to the belief values influence also the inferences. An inference that is valid may become invalid and then something may be change and the inference gets its previous state and become again valid. Nothing can be deleted and that is right because by deleting old propositions and arguments, a piece of the discussion's history is lost. A lot of times in research we conclude to something that we had negated in past (e.g. before 2 years) and so it is useful to be able to see and hark back the reason that lead us to revise our opinion. Whenever the user wants can request the system to monitor a previous state or the current state. In the interface all the states through time are links in a horizontal scroll bar and each link is the time of the corresponding activity (see Figure 4.30 in Section 4.5)

A state of knowledge at a point is all these propositions that the actors believe at this point, the states of affairs. Apart from the proposition that are believed the system displays the reason why a proposition is believed and also if a proposition was resulted from observation or if it was adopted by a source. Each proposition in this interface is link (see Figure 4.29 in Section 4.5) and by clicking it a pop up window with its history opens. Thus, the system in the begging informs the user for the general knowledge of a proposition and then for the more particular knowledge.

The interface is simple but provides the necessary information to the user in a structure way about what is believed and why. In case of composite inferences the system displays the information in a tree structured. In a composite inference we are interesting for the final conclusions but also for how we conclude to these conclusions, which elementary inferences are included in this composite and which premises of an elementary are conclusions to

another elementary.

4.4.3.5 History of a proposition

Propositions are regarded to be statements about things that may be true or false. In our model every proposition is assigned to a belief value. Different Actors can assign different truth values at different times to the same propositions, and want to be informed about what is believed at any time about a particular proposition. The system allows to the user to be aware of the history of each registered proposition and watch each change of the proposition's belief value through time as well as why happened this change. A belief value of proposition changes directly because an Actor can change it or indirectly because of the system's reasoning. Figure 4.31 (Section 4.5) shows the history of the proposition "death of Otzi has_time-span recently". It is clear the interval of time when the proposition was TRUE and the reason of it and also when the proposition became FALSE and for which reason. Also it is visible if this reason is an observation or another proposition or an adoption of an external resource whereon there are links to these resources on the World Wide Web.

The system also can inform the user about the whole argument from which the proposition came up by clicking on each proposition. Figure 4.32 (Section 4.5) illustrates the history of proposition "death of Otzi has_time-span recently" and the arguments that made this proposition FALSE. The propositions that are conclusions to an inference or they are observation or adoption from a source and also used as premises to another inference they are also links inside the last inference. For example in Figure 4.32 (Section 4.5) the proposition "axe of Otzi has_condition neolithic" is contained in an observation. In the first argument the proposition is also link and if is be clicked the whole argument with all information of the observation will be appeared under the first argument. This is a recursive process which searches the propositions that are premises of the appeared inferences and checks if they are also conclusion of other inferences or if they are observations or adoptions. If so, they become also links otherwise are appear as simple text in the argument. In order to offer this functionality and display dynamically the requested data in the same page, we retrieve data from the server asynchronously in the background by using AJAX (Asynchronous JavaScript and XML). It is an interactive and dynamic interface that helps user to understand how a proposition arose and how a composite inference is made.

4.4.3.6 Search for a proposition

It is easy enough for the users to find arguments and propositions and learn about their history. Antilogos allows to the users to extract arguments and being informed about all propositions that are registered even they don't know anything about them. User does not have to know if the proposition is a premise or a conclusion. At the main menu bar of the system there is a search choice. The content of this page is changes dynamically as new propositions are added to the repository.

In this page is displayed the list with the abstract classes of the domain ontology we use (for our example we use CIDOC CRM). Each class contains a nested list with all of its instances that are stored in the repository and each instance contains another nested list with the predicates that have this instance as domain. Similarly, each predicated has a list with instances that have it as range. The list are displayed in a "tree" form. So the user opens gradually these lists until a proposition (a triple) is came in. These propositions are links and a pop up window like the one of Figure 4.31 (Section 4.5) is popped with the history of this proposition. Thus, the system provides an easy way to inform users about the proposition that they are interesting in and also can monitoring all propositions and arguments. Figure 4.33 (Section 4.5) shows an example of this search. In this figure the user is interesting in finding information about the event of the death of Otzi and more especially about the time of the death.

4.5 An Example of use

In this Section we give an example of use. We show screenshots of all the screens of system step by step. We begin with the login page. Then we picture the registration page. After that we show the home page of the system. And then we shows snapshots of each functionality discussed at Section 4.4.3. The design of the pages is still in its primitive stage.

Figure 4.13 shows the login page. This the first page the user sees. If the user has already being registered, enters the username and password otherwise he/she has to visit the registration page and register (see Figure 4.14) the required information.

After the registration and login the user is redirected to choose one of the main functionalities (Figure 4.15). The main choices are the creation of a new issue, the choice to see



Antilogos
SYSTEM FOR FACTUAL ARGUMENTATION

HOME ALL ISSUES INSTANCE MAKER SEARCH REGISTER LOGIN

Enter your username and password to login [If you are not a member Register](#)

Username:

Password:

Login

Figure 4.13: Login the system



Antilogos
SYSTEM FOR FACTUAL ARGUMENTATION

HOME ALL ISSUES INSTANCE MAKER SEARCH REGISTER LOGIN

Register

Please fill in the following fields.
Fields with (*) are required.

Contact information

* Firstname:

* Lastname:

* Username:

* Password:

* Confirm password:

* Email:

Telephone:

Submit Reset

Figure 4.14: Form for registration

all issue (in order to add and monitor the arguments in one of them) and the search.



Figure 4.15: Home Page

In this example we regarded that we want to make a few arguments about the history of Otzi the iceman, who was he, when and where did he die etc. Especially, we show the arguments that already we have discussed in Section 4.4.2.2 about the year of the death and the then about the place of his last meal and the session he died. Everything started from an observation at Alps. Two tourists Erika and Simon Helmut discovered the corpse of Otzi in well condition.

In Figure 4.16 the user chooses to create the issue “The history of Otzi the iceman” and selected to add the argument with the above facts. The user started to construct the factual proposition that the corpse of Otzi was well maintained.

In order to make a new proposition, the user has to create a triple with classes and relations of the domain ontology. To define the subject of the triple, the user has to select a resource class of the ontology and an instance of this class. After the subject selection, the system fills the drop down menu with the predicates of the selected class and the user has to select one of them. Similar to the subject selection is the object selection. As far as the belief value is concerned, is just a drop down menu with values “TRUE”, “FALSE” and “UNKNOWN”.

Figures 4.17, 4.18 and 4.19 illustrate how the user construct the new proposition “corpse of Otzi has_condition well maintained” by selecting the subject, the predicate and the object

The screenshot shows the 'Antilogos' web application interface. At the top, there is a navigation bar with links: HOME, ALL ISSUES, INSTANCE MAKER, SEARCH, REGISTER, and LOGOUT. The main heading is 'Create an Issue for argumentation' with the user 'boutsika' logged in. Below the heading, there is a form with the following elements:

- Issue Title:** A text input field containing 'The history of Otzi the iceman'.
- Add your own argument based on:** A dropdown menu currently set to 'Observation'.
- Instructions:** A message: 'Inset your observations on the topic. If you want to add more observations, press the button near.' followed by an 'Add more observations' button.
- Place:** An empty text input field.
- Evidence:** An empty text input field.
- Physical thing:** An empty text input field.
- 1) Insert your belief:** A section with several dropdown menus:
 - From class:** '-- Select From Class --'
 - From Instance:** '-- Select From Instance --'
 - Relation:** '-- Select relation --'
 - To Class:** '-- Select to Class --'
 - To Instance:** '-- Select To Instance --'
- Insert a comment:** A text input field.
- Belief Value:** '-- Select --'
- Save:** A button at the bottom of the form.

Figure 4.16: Creation of an issue and an observation

of the domain ontology. In Figure 4.17 the user sees all classes and chooses the class E18.Physical_Thing. Then the next drop down menu fills with the instances of this class and the user selects the instance “corpse of Otzi”

In Figure 4.18 another drop down menu is filled automatically by the system with all the predicates of the selected class.

Finally the user defines the the range class and instance in order to complete the triple. Figure 4.19 illustrates the 2 last steps to complete a new proposition.

As the pictures show the user chose existing instances of the domain class but also the system provides and the creation of new instances. In the drop down menus that contain the instances there is the choice “New Instance” and by selecting it a text field is display in order to write something and the system converts it to an RDF instance.

Instead of selecting subject, predicate and object or even a previous proposition, the system offers a simple editor that the user can use to register his/her beliefs. By clicking the link ”Insert triples (Class#Instance predicate Class#Instance-Belief Value) manually (only for expert user)” a small editor is appeared and the user can write proposition beliefs

4.5. AN EXAMPLE OF USE

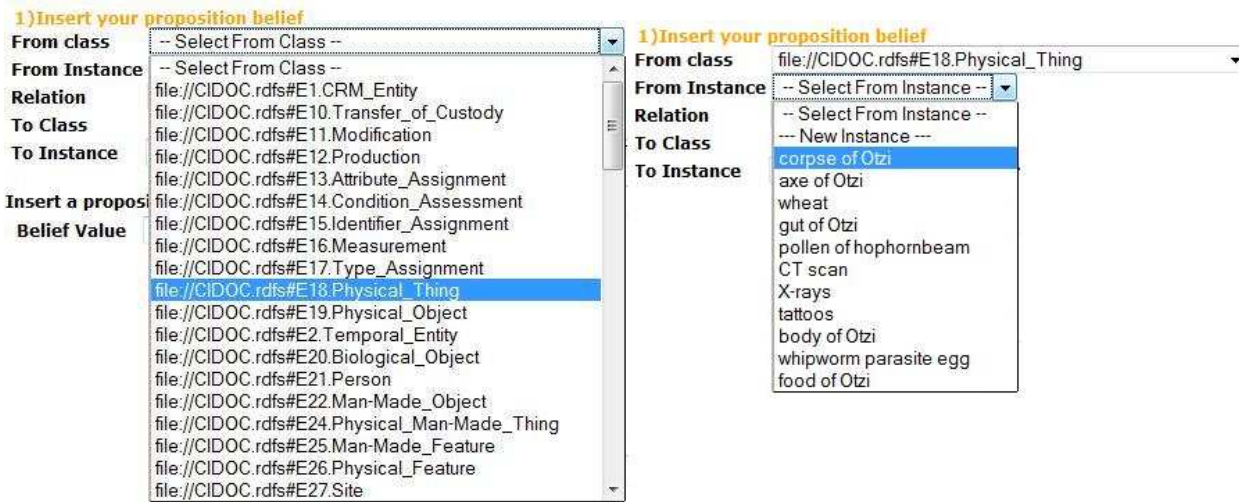


Figure 4.17: Menus with classes and instances of the domain ontology define the subject of the triple

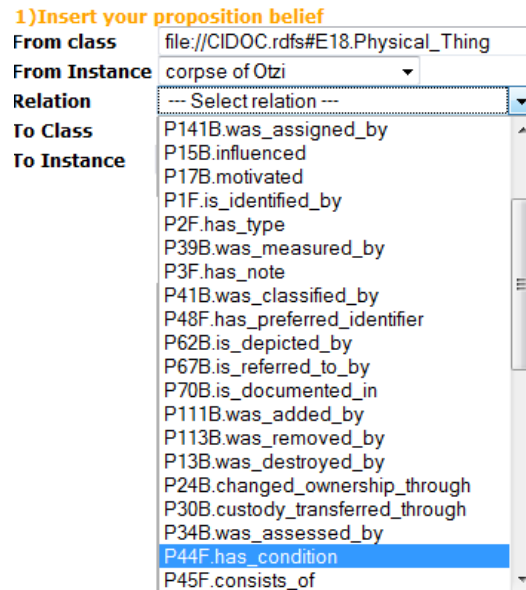


Figure 4.18: Menu with predicates of a selected class

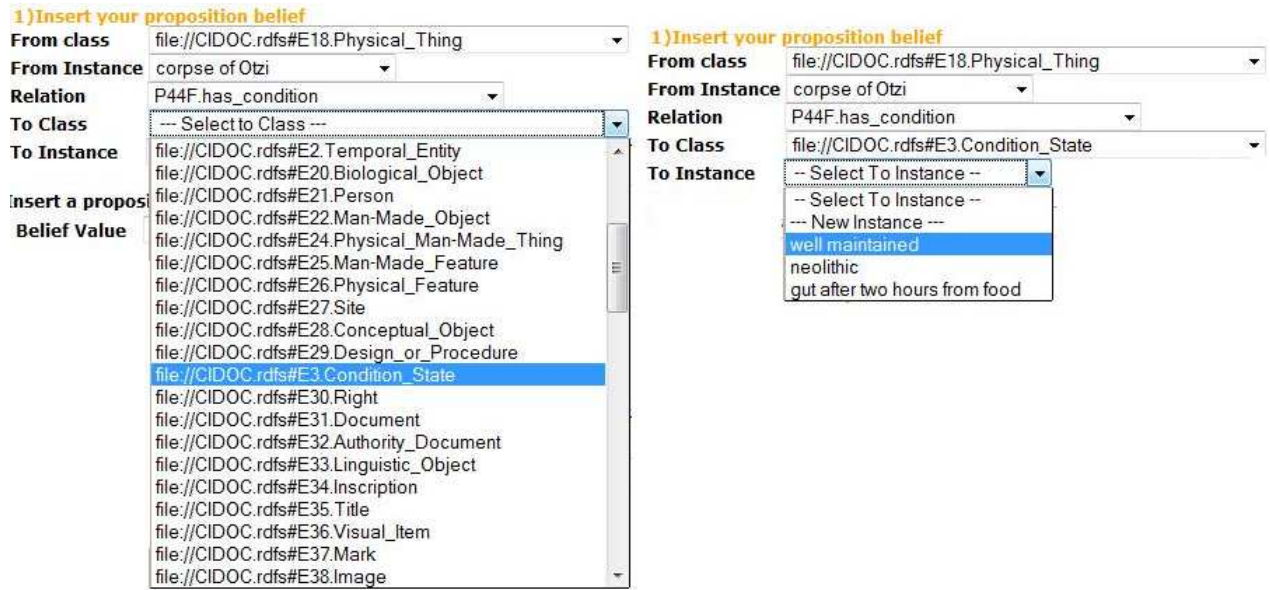


Figure 4.19: Menus with classes and instances of the domain ontology define the object of the triple

in form Class#Instance predicate Class#Instance-Belief Value. The system parsing the user’s input and informs him/her with suitable error messages. However, this editor is quite awkward for the simple user and that is why it is target only on expert users which known the classes and the predicates of the domain ontology (see Figure 4.22).

In the same way we regard that the user entered and the other propositions concerning the place and the actors of the discovery. After the uploading of the argument, the system displays the introduced argument in natural language as show in Figure 4.20 and allows to the user to see the the state of the knowledge after the new argument by clicking in the associated link (“see the updated state of knowledge”) under the displayed argument.

You entered the following argument

Author	Argument	Observation that posted at 2010-06-30T01:26:01
boutsika	By observation, <i>Discovery of Otzi has_time-span September of 1991</i> is TRUE , <i>Discovery of Otzi had_participant Erika Simon</i> is TRUE , <i>Discovery of Otzi had_participant Helmut Simon</i> is TRUE , <i>Discovery of Otzi took_place_at frozen mountains of Alps</i> is TRUE , <i>corpse of Otzi has_condition well maintained</i> is TRUE	

[see the updated state of knowledge](#)

Figure 4.20: An observation in natural language

The state of knowledge at a point is what we believe, the ischium proposition beliefs. Figure 4.21 illustrates the state of knowledge after the introduced observation.

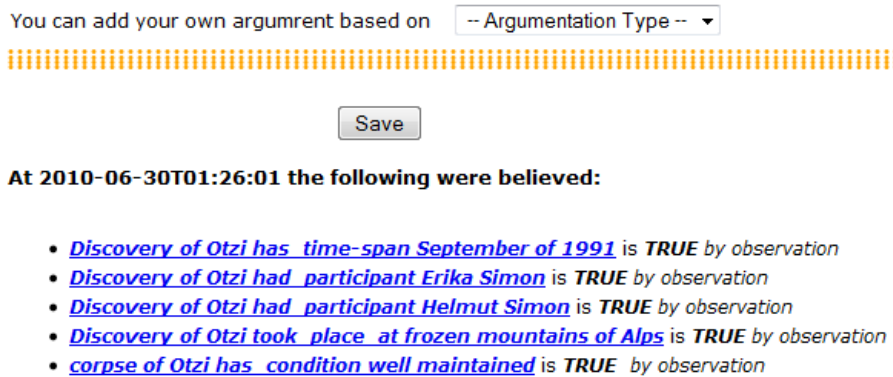


Figure 4.21: First state of knowledge

At first, they believed that the death of Otzi happened recently because his body was in good condition. Thus, in Figure 4.22 the user inserts an elementary inference that uses as premise the proposition “corpse of Otzi has_condition well maintained” and concludes in the new proposition “deth of Otzi has_time-span recently” and applies the inference logic that “thing go off through time”. The proposition of the premise (“corpse of Otzi has_condition well maintained”) exists and the user does not need to construct it again (by selecting subject, predicate and object), he/she reuses it by selecting it from the appropriate drop down menu.

In Figure 4.23 the system display the new argument in natural language. Now the updated state of knowledge is that in Figure 4.24.

Figure 4.24 illustrates the reason why we believe that “deth of Otzi has_time-span recently” it is because (we know that “corpse of Otzi has_condition well maintained” by an observation). Also, mousing over the word “because”, a box is appeared containing the inference logic of the inference (“thing go off through the time”).

Later after inspection in place and they observed Otzi’s staff. His axe was Neolithic. So they add this observation (“axe of Otzi has_condition neolithic”) in the system. After that, researchers used Carbon-14 in order to reveal the year of the death. We assume that a user adopts the belief that Carbon-14 has purpose to reveal the year of the death of a body from <http://en.wikipedia.org/wiki/Carbon-14>. The user accepts as true and adds this

You can add your own argument based on

Inference Logic: or select one

Conclusions

[insert triples \(Class#Instance predicate Class#Instance-Belief Value\) manually \(only for expert user\)](#)

Inset your conclusions on the topic. If you have mare conclusions, press the button near.

1) Insert your proposition belief

From class

From Instance

Relation

To Class

To Instance or select a triple

Insert a proposition about this triple:

Belief Value

Premise

[insert triples \(Class#Instance predicate Class#Instance-Belief Value\) manually \(only for expert user\)](#)

Inset your Premises on the topic. If you have mare premises, press the button near.

1) Insert your proposition belief

From class

From Instance

Relation

To Class

To Instance or select a triple

Insert a proposition about this triple:

Belief Value

Figure 4.22: Form of the elementary inference

You entered the following argument

Author	Argument
boutsika	Inference making that posted at 2010-07-15T16:37:52 Since things go off through the time and it is TRUE that <i>corpse of Otzi has_condition well maintained</i> , we conclude that <i>death of Otzi has_time-span recently</i> is TRUE

Figure 4.23: Elementary inference in natural language

4.5. AN EXAMPLE OF USE

You can add your own argument based on -- Argumentation Type --

things goes off through the time

At 2010-06-30T01:27:01 the following were believed:

- [death of Otzi has time-span recently](#) is **TRUE** because [corpse of Otzi has condition well maintained](#) is **TRUE** by observation
- [Discovery of Otzi has time-span September of 1991](#) is **TRUE** by observation
- [Discovery of Otzi had participant Erika Simon](#) is **TRUE** by observation
- [Discovery of Otzi had participant Helmut Simon](#) is **TRUE** by observation
- [Discovery of Otzi took place at frozen mountains of Alps](#) is **TRUE** by observation
- [corpse of Otzi has condition well maintained](#) is **TRUE** by observation

Figure 4.24: Second state of knowledge

adoption (“Carbon 14 had_specific_purpose the definition of the year of death”) in the system. Carbon-14 revealed that Otzi died before 5310 years ago, at 3300 BC. Thereafter, the user introduce an elementary inference in the system that applies the inference logic that “Carbon-14 is a synchronous technique that calculates the age of things that contains radio-carbon” that uses as premise that the belief “Carbon 14 had_specific_purpose the definition of the year of death” is TRUE and concludes in the belief that “death of Otzi has_time-span 3300BC” is TRUE. Figure 4.25 presents the state of knowledge after the above three arguments.

At 2010-06-30T01:31:09 the following were believed:

- [death of Otzi has time-span recently](#) is **TRUE** because [corpse of Otzi has condition well maintained](#) is **TRUE** by observation
- [death of Otzi has time-span 3300BC](#) is **TRUE** because [Carbon 14 had specific purpose the definition of the year of death](#) is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>
- [Discovery of Otzi has time-span September of 1991](#) is **TRUE** by observation
- [Discovery of Otzi had participant Erika Simon](#) is **TRUE** by observation
- [Discovery of Otzi had participant Helmut Simon](#) is **TRUE** by observation
- [Discovery of Otzi took place at frozen mountains of Alps](#) is **TRUE** by observation
- [corpse of Otzi has condition well maintained](#) is **TRUE** by observation
- [axe of Otzi has condition neolithic](#) is **TRUE** by observation
- [Carbon 14 had specific purpose the definition of the year of death](#) is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>

Figure 4.25: Third state of knowledge

Next, a user while monitoring the knowledge state in Figure 4.25, observes that there are two facts that that it is not possible to stand together. The belief that death of “Otzi has_time-span recently” TRUE and the death of “Otzi has_time-span 3300BC” is TRUE

for different reasons. Although these beliefs are conflicting to each other, the system can't turn the belief value of none of them by reasoning because they use different propositions. As mentioned before we assume that the latest argument is the more stronger and thus we regard stronger the that conclude in the belief that "Otzi has_time-span 3300BC" is TRUE. The user now make another inference that uses as premise that "Otzi has_time-span 3300BC" is TRUE and concludes in "Otzi has_time-span recently" is FALSE. The system now runs the the algorithm 4 presented in Section 4.4.2.4 for inconsistencies because it detects that there are two conclusions with the same proposition but with different belief value. It invalidates the first inference and turns the belief value of its premise ('corpse of Otzi has_condition well maintained') to UNKNOWN because there are no supporters of this belief. So, now the new knowledge state is that in Figure 4.26

At 2010-06-30T01:32:54 the following were believed:

- [death of Otzi has_time-span recently](#) is **FALSE** because [death of Otzi has_time-span 3300BC](#) is **TRUE**
 - [death of Otzi has_time-span 3300BC](#) is **TRUE** because [Carbon 14 had_specific_purpose_the_definition_of_the_year_of_death](#) is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>
- [Discovery of Otzi has_time-span September of 1991](#) is **TRUE** by observation
- [Discovery of Otzi had_participant Erika Simon](#) is **TRUE** by observation
- [Discovery of Otzi had_participant Helmut Simon](#) is **TRUE** by observation
- [Discovery of Otzi took_place_at_frozen_mountains_of_Alps](#) is **TRUE** by observation
- [axe of Otzi has_condition neolithic](#) is **TRUE** by observation
- [Carbon 14 had_specific_purpose_the_definition_of_the_year_of_death](#) is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>
- [corpse of Otzi has_condition well maintained](#) is **UNKNOWN**

Figure 4.26: Fourth state of knowledge

Then, we assume that the user wants to support the belief that "Otzi has_time-span 3300BC" is TRUE and creates another inference using as premise the fact from a previous observation that "axe of Otzi has_condition neolithic" and applying the inference logic that "a synchronous human cannot have Neolithic tools". After that the knowledge state is that in Figure 4.27.


Researches using synchronous techniques found even what Otzi ate before he died and how many hours before his dead. The Figures 4.28, 4.29 illustrates gradually the example with the arguments that is represented in Sections 4.4.2.3 and 4.4.2.4 about the last meal of Otzi. Figure 4.29 illustrates the whole page with the final state of knowledge after the storage of all the above arguments.

4.5. AN EXAMPLE OF USE

At 2010-06-30T01:33:58 the following were believed:

- death of Otzi has time-span recently is **FALSE** because axe of Otzi has condition neolithic is **TRUE** by observation
- death of Otzi has time-span recently is **FALSE** because death of Otzi has time-span 3300BC is **TRUE**
 - death of Otzi has time-span 3300BC is **TRUE** because Carbon 14 had specific purpose the definition of the year of death is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>
- Discovery of Otzi has time-span September of 1991 is **TRUE** by observation
- Discovery of Otzi had participant Erika Simon is **TRUE** by observation
- Discovery of Otzi had participant Helmut Simon is **TRUE** by observation
- Discovery of Otzi took place at frozen mountains of Alps is **TRUE** by observation
- axe of Otzi has condition neolithic is **TRUE** by observation
- Carbon 14 had specific purpose the definition of the year of death is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>
- corpse of Otzi has condition well maintained is **UNKNOWN**

Figure 4.27: Fifth state of knowledge



hophornbeam tree grows at the south of Hauslabjoch

At 2010-08-25T14:22:38 the following are believed:

- last meal of iceman took place at south of the Hauslabjoch is **TRUE** because food of Otzi is composed of pollen of hophornbeam tree is **TRUE**
 - food of Otzi is composed of pollen of hophornbeam tree is **TRUE** because pollen of hophornbeam tree is found on corpse of Otzi is **TRUE** by observation
- death of Otzi has time-span recently is **FALSE** because axe of Otzi has condition neolithic is **TRUE** by observation
- death of Otzi has time-span recently is **FALSE** because death of Otzi has time-span 3300BC is **TRUE**
 - death of Otzi has time-span 3300BC is **TRUE** because Carbon 14 had specific purpose the definition of the year of death is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>
- consumption of hophornbeam pollen has time-span autumn is **FALSE** because consumption of vernal products has time-span autumn is **FALSE**
 - consumption of vernal products has time-span autumn is **FALSE** because Death of Otzi has time-span autumn is **TRUE**
 - Death of Otzi has time-span autumn is **TRUE** because food of Otzi is composed of wheat is **TRUE** by observation
- Discovery of Otzi has time-span September of 1991 is **TRUE** by observation
- Discovery of Otzi had participant Erika Simon is **TRUE** by observation
- Discovery of Otzi had participant Helmut Simon is **TRUE** by observation
- Discovery of Otzi took place at frozen mountains of Alps is **TRUE** by observation
- axe of Otzi has condition neolithic is **TRUE** by observation
- pollen of hophornbeam tree is found on corpse of Otzi is **TRUE** by observation
- food of Otzi is composed of wheat is **TRUE** by observation
- Carbon 14 had specific purpose the definition of the year of death is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>
- corpse of Otzi has condition well maintained is **UNKNOWN**

Figure 4.28: Sixth state of knowledge

4. ANTILOGOS: A SYSTEM FOR FACTUAL ARGUMENTATION BASED ON IAM

The screenshot shows the Antilogos web application interface. At the top, there is a navigation bar with links for HOME, ALL ISSUES, INSTANCE MAKER, SEARCH, REGISTER, and LOGOUT. The user is identified as 'boutsika'. The main content area is titled 'The history of Otzi the iceman' and displays a timeline of knowledge state updates. The selected time point is 2010-06-30T02:31:27. Below the timeline, there is a text input field for adding arguments and a 'Save' button. The main part of the page lists the current knowledge state as of 2010-09-07T20:21:38, showing various facts about Otzi the iceman, such as his death, last meal, discovery, and the condition of his body, with their truth values (TRUE, FALSE, UNKNOWN) and supporting evidence.

The history of Otzi the iceman

Knowledge state of issue 'The history of Otzi the iceman' at

TIME

2010-06-30T03:43:51 2010-06-30T02:42:08 2010-06-30T02:39:24 2010-06-30T02:35:12 2010-06-30T02:33:27 2010-06-30T02:31:27

This issue was created by boutsika

You can add your own argument based on -- Argumentation Type --

Save

At 2010-09-07T20:21:38 the following are believed:

- [death of Otzi has time-span recently](#) is **FALSE** because [axe of Otzi has condition Neolithic](#) is **TRUE** by observation
- [death of Otzi has time-span recently](#) is **FALSE** because [death of Otzi has time-span 3300BC](#) is **TRUE**
 - [death of Otzi has time-span 3300BC](#) is **TRUE** because [Carbon 14 had specific purpose the definition of year of death](#) is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>
- [last meal of Otzi took place at south of the Hauslabioch](#) is **FALSE** because [food of Otzi is composed of red deer meat](#) is **TRUE**
 - [food of Otzi is composed of red deer meat](#) is **TRUE** because [red deer meat is found on gut of Otzi](#) is **TRUE** by observation
- [death of Otzi has time-span autumn](#) is **FALSE** because [death of Otzi has time-span spring](#) is **TRUE**
 - [death of Otzi has time-span spring](#) is **TRUE** because [food of Otzi is composed of pollen of hophornbeam tree](#) is **TRUE**
 - [food of Otzi is composed of pollen of hophornbeam tree](#) is **TRUE** because [pollen of hophornbeam tree is found on gut of Otzi](#) is **TRUE** by observation
- [Discovery of Otzi has time-span September of 19991](#) is **TRUE** by observation
- [Discovery of Otzi had participant Erika Simon](#) is **TRUE** by observation
- [Discovery of Otzi had participant Helmut Simon](#) is **TRUE** by observation
- [Discovery of Otzi took place at frozen mountains of Alps](#) is **TRUE** by observation
- [axe of Otzi has condition Neolithic](#) is **TRUE** by observation
- [pollen of hophornbeam tree is found on gut of Otzi](#) is **TRUE** by observation
- [red deer meat is found on gut of Otzi](#) is **TRUE** by observation
- [Carbon 14 had specific purpose the definition of year of death](#) is **TRUE** by adoption of <http://en.wikipedia.org/wiki/Carbon-14>
- [corpse of Otzi has condition well maintained](#) is **UNKNOWN**
- [food of Otzi is composed of wheat](#) is **UNKNOWN**
- [consumption of vernal products has time-span autumn](#) is **UNKNOWN**
- [consumption of hophornbeam pollen has time-span autumn](#) is **UNKNOWN**

Figure 4.29: Final state of knowledge

At any time the user can monitor the state of knowledge at any previous time. At the top of the page there is the horizontal scroll bar that contains links which represent all states of knowledge through time (see Figure 4.30).



Figure 4.30: Scroll bar with states of knowledge

As we see in the Figures with the pages of knowledge state the user can see what is believed about every proposition, its believe value and why it believed. In these pages the system gives the general idea of the registered arguments. However, the system holds also the history of each proposition. The user is able to be informed about the history of a proposition, about its past. By clicking on a proposition opens a pop up window with the history of it. The system guides the user to passe from the general information to more particular information.

Figure 4.31 shows the history of the proposition “death of Otzi has_time-span recently”. It is clear the interval of time when the proposition was TRUE and the reason of it and also when the proposition became FALSE and for which reason. Also it is visible if this reason is an observation or another proposition or an adoption of an external resource whereon there are links to these resources on the World Wide Web.

In the pop up window the system is able to inform the user can to be informed about more particular information, about the whole argument from which the proposition came up by clicking on each proposition. Figure 4.32 illustrates the history of proposition ”death of Otzi has time-span recently” and the arguments that made this proposition FALSE.

The propositions that are conclusions to an inference or they are observation or adoption from a source and also used as premises to another inference they are also links inside the last inference. For example in Figure 4.32 the proposition “axe of Otzi has_condition neolithic” is contained in an observation. In the first argument it is also link and if is be clicked the whole argument with all information of the observation will be appeared under the first argument.

In order to offer this functionality and display dynamically the requested data in the

History of *death of Otzi has_time-span recently*

[Close and go back at state 2010-07-19T10:17:02](#)

from 2010-06-30T01:31:58 until now
death of Otzi has_time-span recently is FALSE

- *death of Otzi has_time-span recently* is FALSE because *axe of Otzi has condition neolithic* is TRUE by observation
- *death of Otzi has_time-span recently* is FALSE because *death of Otzi has_time-span 3300BC* is TRUE
 - *death of Otzi has_time-span 3300BC* is TRUE because *Carbon 14 had specific purpose the definition of the year of death* is TRUE by adoption of <http://en.wikipedia.org/wiki/Carbon-14>

from 2010-06-30T01:27:01 until 2010-06-30T01:31:58
death of Otzi has_time-span recently is TRUE

- *death of Otzi has_time-span recently* is TRUE because *corpse of Otzi has condition well maintained* is TRUE

Figure 4.31: Example of a proposition's history

History of *death of Otzi has_time-span recently*

[Close and go back at state 2010-07-19T20:29:47](#)

from 2010-06-30T01:31:58 until now
death of Otzi has_time-span recently is FALSE

- *death of Otzi has_time-span recently* is FALSE because *axe of Otzi has condition neolithic* is TRUE by observation
- *death of Otzi has_time-span recently* is FALSE because *death of Otzi has_time-span 3300BC* is TRUE
 - *death of Otzi has_time-span 3300BC* is TRUE because *Carbon 14 had specific purpose the definition of the year of death* is TRUE by adoption of <http://en.wikipedia.org/wiki/Carbon-14>

Author	Argument	Inference making that posted at 2010-06-30T01:31:58
martin	Since a synchronous human cannot have neolithic tools and it is TRUE that <i>axe of Otzi has condition neolithic</i> , we conclude that <i>death of Otzi has_time-span recently</i> is FALSE	

Author	Argument	Inference making that posted at 2010-06-30T01:32:54
martin	Since Carbon-14 is a synchronous technique that calculates the age of things that contains radiocarbon and it is TRUE that <i>death of Otzi has_time-span 3300BC</i> , we conclude that <i>death of Otzi has_time-span recently</i> is FALSE	

from 2010-06-30T01:27:01 until 2010-06-30T01:31:58
death of Otzi has_time-span recently is TRUE

- *death of Otzi has_time-span recently* is TRUE because *corpse of Otzi has condition well maintained* is TRUE

Figure 4.32: Example of a proposition's history and the arguments that it came up

same page, we retrieve data from the server asynchronously in the background by using AJAX (Asynchronous JavaScript and XML). It is an interactive and dynamic interface that helps user to understand how a proposition arose and how a composite inference is made.

Search in the repository is a useful functionality in applications. Users usually want to know something particular without monitoring all proposition or arguments one by one. In Figure 4.33 we assume that a user wants to know everything about the time of the event of Otzi’s death. Gradually the user opens the nested list with instances of the class E5.Event. Then he/she opens the list with the predicates of the instance “death of Otzi”. From all of predicates the user selects to open the list with range instances of the predicate has.time-span. Finally, the user finds four registered propositions about the time of the death. Each proposition is a link and a pop up window like the one of Figure 4.31 is popped with the history of this proposition.



Figure 4.33: Example of searching propositions

Chapter 5

Conclusions

We described a argumentation model that discriminates between the structure of a composite inference at a point in time and the historical process of developing the elements of this structure. This is something that most models failed to describe. They blur the logical sequence of one conclusion being premise for the next conclusion with the historical (temporal) order in which the inferences were found. Indeed, the historical order is widely independent from the logical one, and only accidentally coincides. Further, none of these models connect explicitly to an ontology or even a database schema, and therefore cannot represent the effect of argumentation on the contents of an information system. Many models confuse arguments with the propositions in the premises. Our model introduces a distinction between a proposition and the belief in it and is able to represent human argumentation by describing the different attitudes of participants to the same proposition through time. Also, is one step forward from the classic “premise-conclusion” structure of an argument by introducing the inference logic. We believe that our model can indeed describe scholarly and scientific argumentation. In this work we made extensions to this model in order to concentrate on factual argumentation and we included the case that an inference per se, regardless the beliefs in premises and conclusions, is defeated. Also, We implemented the model using RDF Schema encoding.

We developed Antilogos (described in Chapter 4), an web-based information system based on this model for factual argumentation where a group of people can collaboratively reach to conclusions about possible state of affairs that happened in the past.

The system

- Supports the creation, the evolution, the composition, the revision and the storage of arguments, which are machine readable.
- Supports representation of the arguments in a structure way.
- Provides monitoring of the argumentation process. Antilogos enable users to be informed about the current as well as about any other state of knowledge in past and also provides information about the provenance of this knowledge.
- Detects and resolves conflicts. Inferences of users many times lead to impossible situations, to contradictions between the beliefs. Antilogos helps users to conclude to the right propositions by resolving theses inconsistencies.
- Enables users to query the registered information structures. Users can search and learn the history of a proposition and view the different beliefs about this proposition that different users had at different times.
- Uses RDF, a W3C standard, for expressing and representing networks of arguments and thus it exploits the rich features of the semantic web.
- Anilogos uses Sesame RDF repository, which can be accessed through the web for uploading and querying arguments.

Antilogos can successfully demonstrate a real published archaeological example using CIDOC CRM ontology for describing the possible state of affairs. We believe that it can be used by research community for the record and representation of empirical data about research processes and help them to intergrade the historical sources and the huge amount of the sources.

In future, we're planning to extend IAM in order to support and other kinds of scientific argumentation. Furthermore, we intent to incorporate also probabilistic beliefs to the system. Also, as the data representation will become increasingly important, we need to make more improvements to the interface and to evaluate the usability of the system's GUI by a group of users.

Bibliography

- [1] <http://www.athenasoft.org/>.
- [2] <http://jena.sourceforge.net/>.
- [3] Otzi. <http://www.mummytombs.com/main.otzi.htm>.
- [4] Protege. <http://protege.stanford.edu>.
- [5] Sesame. <http://www.openrdf.com>.
- [6] Sparql. <http://www.w3.org/TR/rdf-sparql-query/>.
- [7] Truthmapping. <http://www.truthmapping.com>, 2006.
- [8] G. Antoniou and F. Van Harmelen. *A semantic web primer*. The MIT Press, 2004.
- [9] N. Aussenac-Gilles. Ontology or meta-model for retrieving scientific reasoning in documents: the Arkeotek project. In *Proc. of the Workshop on Exploring the limits of global models for integration and use of historical and scientific information*, pages 24–25, 2006.
- [10] A. Bayliss and C.B. Ramsey. Pragmatic Bayesians: a decade of integrating radiocarbon dates into chronological models. *Tools for Constructing Chronologies: crossing disciplinary boundaries*, pages 25–41, 2004.
- [11] Carlos Ches nevar, Jarred McGinnis, Sanjay Modgil, Iyad Rahwan, Chris Reed, Guillermo Simari, Matthew South, Gerard Vreeswijk, and Steven Willmott. Towards an argument interchange format. *Knowl. Eng. Rev.*, 21(4):293–316, 2006.
- [12] Glenn Rowe Chris Reed, Raquel Mochales Palau and Marie-Francine Moens. Language resources for studying argument. In Bente Maegaard Joseph Mariani Jan Odjik Stelios Piperidis Daniel Tapias Nicoletta Calzolari (Conference Chair), Khalid Choukri, editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may 2008. European Language Resources Association (ELRA). <http://www.lrec-conf.org/proceedings/lrec2008/>.

- [13] Jeff Conklin and Michael L. Begeman. gibis: a hypertext tool for exploratory policy discussion. *ACM Trans. Inf. Syst.*, 6(4):303–331, 1988.
- [14] J.C. Gardin. Archaeological discourse, conceptual modelling and digitalisation: an interim report of the logicist program. In *CAA*, pages 5–11, 2002.
- [15] J.C. Gardin and V. Roux. The Arkeotek project: a european network of knowledge bases in the archaeology of techniques. *Archeologia e Calcolatori*, (XV):25–40, 2004.
- [16] J.C.I. Gardin. The structure of archaeological theories. *Studies in Modern Archaeology*, 3(3):7–25, 1990.
- [17] Tim Van Gelder. Argument mapping with reason!able. In *The American Philosophical Association Newsletter on Philosophy and Computers*. <http://www.arts.unimelb.edu.au/?tgelder/papers/APA.pdf>, 85:85–90, 2002.
- [18] Thomas F. Gordon and Nikos Karacapilidis. The zeno argumentation framework. In *ICAAIL '97: Proceedings of the 6th international conference on Artificial intelligence and law*, pages 10–18, New York, NY, USA, 1997. ACM.
- [19] Nicola Guarino. Formal ontology and information systems. pages 3–15. IOS Press, 1998.
- [20] Giancarlo Guizzardi, Heinrich Herre, and Gerd Wagner. On the general ontological foundations of conceptual modeling. In *ER '02: Proceedings of the 21st International Conference on Conceptual Modeling*, pages 65–78, London, UK, 2002. Springer-Verlag.
- [21] Nikos Karacapilidis and Dimitris Papadias. Computer supported argumentation and collaborative decision making: the hermes system. *Inf. Syst.*, 26(4):259–277, 2001.
- [22] K. Kotis, G.A. Vouros, and J.P. Alonso. HCOME: A tool-supported methodology for engineering living ontologies. *Semantic Web and Databases*, pages 155–166, 2005.
- [23] W. Kunz and H. Rittel. Issues as elements of information systems. Technical report, Citeseer, 1970.
- [24] Gangmin Li, Victoria Uren, Enrico Motta, Simon Buckingham Shum, and John Domingue. Claimaker: Weaving a semantic web of research papers. In *In Proceedings of the 1st International Semantic Web Conference, Sardinia*, 2002.
- [25] Doerr Martin, Kritsotaki Athina, and Boutsika Katerina. Factual argumentation - a core model for assertions making. *J. Comput. Cult. Herit.*, 2010.
- [26] T. Gill S. Stead N. Crofts, M. Doerr and M. Stiff. Definition of the cidoc conceptual reference model. http://cidoc.ics.forth.gr/docs/cidoc_crm_version_4.2.5a.doc, September 2008.

- [27] T. Parsons. What is an Argument? *The Journal of Philosophy*, 93(4):164–185, 1996.
- [28] H. Sofia Pinto, Steffen Staab, and Christoph Tempich. Diligent: Towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. In *In Proceedings of the 16th European Conference on Artificial Intelligence (ECAI)*, pages 393–397. IOS Press, 2004.
- [29] Iyad Rahwan. Mass argumentation and the semantic web. *Web Semant.*, 6(1):29–37, 2008.
- [30] G. Rowe, F. Macagno, C. Reed, and D. Walton. Araucaria as a tool for diagramming arguments in teaching and studying philosophy. *Teaching Philosophy*, 29(2):111, 2006.
- [31] Barbara Schmidt-Belz, Claus Rinner, and Thomas F. Gordon. Geomed for urban planning - first user experiences. In *GIS '98: Proceedings of the 6th ACM international symposium on Advances in geographic information systems*, pages 82–87, New York, NY, USA, 1998. ACM.
- [32] Albert Selvin, Simon Buckingham Shum, Maarten Sierhuis, Jeff Conklin, Beatrix Zimmermann, Charles Palus, Wilfred Drath, David Horth, John Domingue, Enrico Motta, and Gangmin Li. Compendium: Making meetings into knowledge events. knowledge technologies 2001. In *In: Knowledge Technologies 2001*, pages 4–7, 2001.
- [33] Simon Buckingham Shum, Enrico Motta, and John Domingue. Scholonto: An ontology-based digital library server for research documents and discourse. *International Journal on Digital Libraries*, 3:237–248, 2000.
- [34] J.F. Sowa. *Knowledge representation: logical, philosophical, and computational foundations*. MIT Press, 2000.
- [35] Christoph Tempich, H. Sofia Pinto, York Sure, and Steffen Staab. An argumentation ontology for distributed, loosely-controlled and evolving engineering processes of ontologies (diligent). pages 241–256. Springer, 2005.
- [36] S.E. Toulmin. *The uses of argument*. Cambridge Univ Pr, 2003.
- [37] Bart Verheij. Rules, reasons, arguments. formal studies of argumentation and defeat. <http://www.metajur.unimaas.nl/bart/proefschrift/>, 1996.
- [38] Bart Verheij. Argumed - a template-based argument mediation system for lawyers., 1998.
- [39] A. Wisniewski. Erotetic arguments: A preliminary analysis. *Studia Logica*, 50(2):261–274, 1991.
- [40] A. Wisniewski. Questions and inferences. *Logique et analyse: publication trimestrielle du Centre National Belge de Recherches de Logique*, pages 5–43, 2001.

Appendix

A.1 RDF Source

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:am="http://protege.stanford.edu/am#"
  xmlns:crm="http://www.cidoc-crm.org/rdfs/cidoc_v4.2.rdfs#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#" >
  <rdfs:Class rdf:ID="Argumentation" rdfs:label="Argumentation" >
    <rdfs:subClassOf rdf:resource="crm:E7.Activity" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="Belief" rdfs:label="Belief" >
    <rdfs:subClassOf rdf:resource="am:State" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="Belief_Adoption" rdfs:label="Belief_Adoption" >
    <rdfs:subClassOf rdf:resource="am:Argumentation" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="Belief_Value" rdfs:label="Belief_Value" >
    <rdfs:subClassOf rdf:resource="rdfs:Resource" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="Composite_Inference" rdfs:label="Composite_Inference" >
    <rdfs:subClassOf rdf:resource="am:Factual_Inference" />
  </rdfs:Class>
  <rdfs:Class rdf:ID="Elementary_Inference" rdfs:label="Elementary_Inference" >
    <rdfs:subClassOf rdf:resource="am:Factual_Inference" />
  </rdfs:Class>
```

```

<rdfs:Class rdf:ID="Factual_Belief" rdfs:label="Factual_Belief" >
  <rdfs:subClassOf rdf:resource="#Proposition_Belief" />
</rdfs:Class>
<rdfs:Class rdf:ID="Factual_Inference" rdfs:label="Factual_Inference" >
  <rdfs:subClassOf rdf:resource="am:Inference" />
</rdfs:Class>
<rdfs:Class rdf:ID="Factual_Inference_Making" rdfs:label="Factual_Inference_Making" >
  <rdfs:subClassOf rdf:resource="am:Factual_Argumentation" />
  <rdfs:subClassOf rdf:resource="am:Inference_Making" />
</rdfs:Class>
<rdfs:Class rdf:ID="Factual_Proposition" rdfs:label="Factual_Proposition" >
  <rdfs:subClassOf rdf:resource="am:Proposition" />
</rdfs:Class>
<rdfs:Class rdf:ID="Factual_Argumentation" rdfs:label="Factual_Argumentation" >
  <rdfs:subClassOf rdf:resource="am:Argumentation" />
</rdfs:Class>
<rdfs:Class rdf:ID="Inference" rdfs:label="Inference" >
  <rdfs:subClassOf rdf:resource="#Proposition" />
</rdfs:Class>
<rdfs:Class rdf:ID="Inference_Belief" rdfs:label="Inference_Belief" >
  <rdfs:subClassOf rdf:resource="am:Proposition_Belief" />
</rdfs:Class>
<rdfs:Class rdf:ID="Inference_Defeating" rdfs:label="Inference_Defeating" >
  <rdfs:subClassOf rdf:resource="am:Inference" />
</rdfs:Class>
<rdfs:Class rdf:ID="Inference_Logic" rdfs:label="Inference_Logic" >
  <rdfs:subClassOf rdf:resource="rdfs:Resource" />
</rdfs:Class>
<rdfs:Class rdf:ID="Inference_Defeating" rdfs:label="Inference_Defeating" >
  <rdfs:subClassOf rdf:resource="am:Inference_Making" />
</rdfs:Class>
<rdfs:Class rdf:ID="Inference_Making" rdfs:label="Inference_Making" >
  <rdfs:subClassOf rdf:resource="am:Argumentation" />
</rdfs:Class>
<rdfs:Class rdf:ID="Observation" rdfs:label="Observation" >

```

```

    <rdfs:subClassOf rdf:resource="am:Factual_Argumentation" />
</rdfs:Class>
<rdfs:Class rdf:ID="Proposition" rdfs:label="Proposition">
    <rdfs:subClassOf rdf:resource="rdfs:Resource" />
</rdfs:Class>
<rdfs:Class rdf:ID="Proposition_Belief" rdfs:label="Proposition_Belief">
    <rdfs:subClassOf rdf:resource="am:Belief" />
</rdfs:Class>
<rdfs:Class rdf:ID="Recording" rdfs:label="Recording">
    <rdfs:subClassOf rdf:resource="crm:E73.Information_Object" />
</rdfs:Class>
<rdfs:Class rdf:ID="State" rdfs:label="State">
    <rdfs:subClassOf rdf:resource="crm:E2.Temporal_Entity" />
</rdfs:Class>
<rdf:Property rdf:ID="adopted" rdfs:label="adopted">
    <rdfs:domain rdf:resource="am:Belief_Adoption" />
    <rdfs:range rdf:resource="am:Proposition_Belief" />
</rdf:Property>
<rdfs:Class rdf:ID="Question" rdfs:label="Question">
    <rdfs:subClassOf rdf:resource="rdfs:Resource" />
</rdfs:Class>
<rdf:Property rdf:ID="applies" rdfs:label="applies">
    <rdfs:domain rdf:resource="am:Inference" />
    <rdfs:range rdf:resource="am:Inference_Logic" />
</rdf:Property>
<rdf:Property rdf:ID="is_motivation_of" rdfs:label="is_motivation_of">
    <rdfs:domain rdf:resource="am:Question" />
    <rdfs:range rdf:resource="am:Argumentation" />
</rdf:Property>
<rdf:Property rdf:ID="carry_out_by" rdfs:label="carry_out_by">
    <rdfs:range rdf:resource="crm:E39.Actor" />
    <rdfs:domain rdf:resource="crm:E7.Activity" />
</rdf:Property>
<rdf:Property rdf:ID="concluded_in" rdfs:label="concluded_in">
    <rdfs:range rdf:resource="am:Factual_Belief" />

```

```

    <rdfs:domain rdf:resource="am:Factual_Argumentation" />
    <rdfs:subPropertyOf rdf:resource="am:resulted_in_OR_confirmed" />
</rdf:Property>
<rdf:Property rdf:ID="concludes_in" rdfs:label="concludes_in">
    <rdfs:domain rdf:resource="am:Inference" />
    <rdfs:range rdf:resource="am:Proposition_Belief" />
</rdf:Property>
<rdf:Property rdf:ID="consists_of" rdfs:label="consists_of">
    <rdfs:domain rdf:resource="am:Composite_Inference" />
    <rdfs:range rdf:resource="am:Elementary_Inference" />
</rdf:Property>
<rdf:Property rdf:ID="finally_concludes" rdfs:label="finally_concludes">
    <rdfs:domain rdf:resource="am:Composite_Inference" />
    <rdfs:range rdf:resource="am:Proposition_Belief" />
    <rdfs:subPropertyOf rdf:resource="am:concludes_in" />
</rdf:Property>
<rdf:Property rdf:ID="has_belief_time" rdfs:label="has_belief_time">
    <rdfs:domain rdf:resource="am:Belief" />
    <rdfs:range rdf:resource="crm:E52.Time-Span" />
</rdf:Property>
<rdf:Property rdf:ID="initiates" rdfs:label="initiates">
    <rdfs:range rdf:resource="am:Inference_Belief" />
    <rdfs:domain rdf:resource="am:Inference_Making" />
    <rdfs:subPropertyOf rdf:resource="am:resulted_in_OR_confirmed" />
</rdf:Property>
<rdf:Property rdf:ID="is" rdfs:label="is">
    <rdfs:range rdf:resource="#Belief_Value" />
    <rdfs:domain rdf:resource="#Proposition_Belief" />
</rdf:Property>
<rdf:Property rdf:ID="is_believed_by" rdfs:label="is_believed_by">
    <rdfs:domain rdf:resource="am:Belief" />
    <rdfs:range rdf:resource="crm:E39.Actor" />
</rdf:Property>
<rdf:Property rdf:ID="made" rdfs:label="made">
    <rdfs:range rdf:resource="am:Inference" />

```

```

    <rdfs:domain rdf:resource="am:Inference_Making" />
</rdf:Property>
<rdf:Property rdf:ID="motivated" rdfs:label="motivated">
    <rdfs:range rdf:resource="am:Argumentation" />
    <rdfs:domain rdf:resource="am:Argumentation" />
</rdf:Property>
<rdf:Property rdf:ID="observed" rdfs:label="observed">
    <rdfs:domain rdf:resource="#Observation" />
    <rdfs:range rdf:resource="crm:E18.Physical_Thing" />
</rdf:Property>
<rdf:Property rdf:ID="resulted_in_OR_confirmed" rdfs:label="resulted_in_OR_confirmed">
    <rdfs:domain rdf:resource="am:Argumentation" />
    <rdfs:range rdf:resource="am:Proposition_Belief" />
</rdf:Property>
<rdf:Property rdf:ID="that" rdfs:label="that">
    <rdfs:range rdf:resource="am:Proposition" />
    <rdfs:domain rdf:resource="am:Proposition_Belief" />
</rdf:Property>
<rdf:Property rdf:ID="that_2" rdfs:label="that">
    <rdfs:range rdf:resource="am:Inference" />
    <rdfs:domain rdf:resource="am:Inference_Belief" />
<rdfs:subPropertyOf rdf:resource="am:that" />
</rdf:Property>
<rdf:Property rdf:ID="that_3" rdfs:label="that">
    <rdfs:range rdf:resource="am:Factual_Proposition" />
    <rdfs:domain rdf:resource="am:Factual_Belief" />
<rdfs:subPropertyOf rdf:resource="am:that" />
</rdf:Property>
<rdf:Property rdf:ID="used_additional_evidence" rdfs:label="used_additional_evidence">
    <rdfs:domain rdf:resource="am:Observation" />
    <rdfs:range rdf:resource="am:Recording" />
</rdf:Property>
<rdf:Property rdf:ID="uses_as_initial_premise" rdfs:label="uses_as_initail_premise">
    <rdfs:domain rdf:resource="am:Composite_Inference" />
    <rdfs:range rdf:resource="am:Proposition_Belief" />

```

```
<rdfs:subPropertyOf rdf:resource="am:uses_as_premise" />
</rdf:Property>
<rdf:Property rdf:ID="uses_as_premise" rdfs:label="uses_as_premise">
  <rdfs:domain rdf:resource="am:Inference" />
  <rdfs:range rdf:resource="am:Proposition_Belief" />
</rdf:Property>
<rdf:Property rdf:ID="username" rdfs:label="username">
  <rdfs:domain rdf:resource="crm:E39.Actor" />
  <rdfs:range rdf:resource="rdfs:Literal" />
</rdf:Property>
<rdf:Property rdf:ID="firstname" rdfs:label="firstname">
  <rdfs:domain rdf:resource="crm:E39.Actor" />
  <rdfs:range rdf:resource="rdfs:Literal" />
</rdf:Property>
<rdf:Property rdf:ID="lastname" rdfs:label="lastname">
  <rdfs:domain rdf:resource="crm:E39.Actor" />
  <rdfs:range rdf:resource="rdfs:Literal" />
</rdf:Property>
<rdf:Property rdf:ID="email" rdfs:label="email">
  <rdfs:domain rdf:resource="crm:E39.Actor" />
  <rdfs:range rdf:resource="rdfs:Literal" />
</rdf:Property>
<rdf:Property rdf:ID="telephone" rdfs:label="telephone">
  <rdfs:domain rdf:resource="crm:E39.Actor" />
  <rdfs:range rdf:resource="rdfs:Literal" />
</rdf:Property>
<rdf:Property rdf:ID="end_date" rdfs:label="end_date">
  <rdfs:domain rdf:resource="am:Belief" />
  <rdfs:range rdf:resource="xsd:dateTime" />
</rdf:Property>
<rdf:Property rdf:ID="begin_date" rdfs:label="begin_date">
  <rdfs:domain rdf:resource="am:Belief" />
  <rdfs:range rdf:resource="xsd:dateTime" />
</rdf:Property>
</rdf:RDF>
```