Imperial College of Science, Technology and Medicine
(University of London)
Department of Computing

Naive Metaphysics

Merging Strawson's Theory of Individuals with Parsons' Theory of Thematic Roles as a Basis for Multiagent Semantics

by

Schneider L. N.

Submitted in partial fulfilment
of the requirements for the MSc
Degree in Computing Science of the
University of London and for the
Diploma of Imperial College of
Science, Technology and Medicine.

September 2001

DEO
ET
ARTI

Abstract

The aim of this thesis is the development of a minimal semantic ontology merging intuitions from Strawson's theory of individuals and Parsons' theory of events and thematic roles. This ontology as a set of top-level conceptual distinctions is shown to be the foundation for the semantic subcategorisation of verbs and the basic logical structure of natural language sentences. The minimal ontology proposed in this thesis also underlies the shared semantics in a multi-agent system involving human as well as software agents interacting with each other at least partially via natural language communication. As an experimental test of our semantic theory we implement a multi-agent system in the form of a client-server architecture, involving reasoning software agents capable of parsing, proving or disproving natural language sentences sent to them as challenges by human agents operating the client interface. This proof of concept demonstrates how communication in a multi-agent system can rely on a shared minimal semantic ontology of the type we have described.

Acknowledgements

In the first place I would like to thank my supervisor, Jim Cunningham, not only for his helpful practical advice and his sense for minutia, but also for his humour and enormous patience. Then I would like to express my gratitude to Keith Clark, my second marker, as well as to Frank Kriwaczek for their competent teaching without which I would be a worse Prolog programmer than I am. I very heartily thank Silvana Zappacosta for her enduring friendliness and unfailing helpfulness. I am also deeply grateful to my personal tutor Daniel Rueckert who encouraged me more than once 'to get on with it'.

Special thanks to my brother Eric and to Chris Sansom for proofreading and moral support. It goes without saying that they are not to be held responsible for any linguistic mistakes still lurking on the pages of this thesis.

Finally I acknowledge my deep personal endebtment to Georges Heinrich and Ulrike Kohl who encouraged me to take on this course at Imperial. I owe them one of the most eventful and enriching years of my life.

Contents

Introduction

0.1 Subject Matter and Objectives

An introduction is a compilation of commonplaces. But commonplaces are ideal starting points. Here are two leading us into the heart of the subject:

Agents are in the world. They have to realise their goals in a particular environment and in order to achieve their objectives, they have to adapt to that environment. Thus agents have to *know* the world they are in. Knowledge is adequate belief, an accurate *model* of the world. An agent's model of the world is its *ontology*.

Agents are with other agents. They have to realize their goals in the company of other agents, human or not, and in order to achieve their objectives, they have to cooperate with each other. Cooperation is behaviour coordination through communication on the basis of a specific language. The semantics of an the agent's language is defined by its model. The intended model of an agent's language is the model of the world the agent is in, in other words: the agent's ontology.

These two commonplaces justify the assertion that ontology is the basis for agent semantics. To be more precise: the basis for multi-agent semantics is an ontology shared by the members of a multi-agent system.

The aim of this thesis is to develop a minimal ontology, a set of concepts that can serve as a basis for specifying the semantics i.e. the logical form of agent messages. Its main inspiration comes from the semantical analysis of natural language, as well as philosophical accounts of the commonsense view of the world.

At the heart of our argumentation lies Terence Parsons' [47] account of the semantics of verbs and adjectives in terms of underlying states/events and parts played in them by other individuals. Obviously such an account involves a description of types of "occurrences" (i.e. states and events) as well as a specification of "occurrence"-related roles and their possible "fillers".

An ideal companion to such an enquiry is a theory of basic particular-types. Now just such a theory has been presented by Peter Strawson [60, 61]. Thus there is some hope that merging both approaches could result in an ontology serving as a basis for multi-agent semantics.

The semantic ontology proposed here may be put to two uses: to define the fundamental concepts necessary for agents to communicate and to reason, as well as to contribute to the computational analysis of natural language. The two uses can combined in a multi-agent system involving humans and thus recurring to natural language communication. An implementation that illustrates both uses will be presented in the last chapter as a proof of principle.

0.2 Background and Justification

Quite a lot of ontology work has already been undertaken in the AI community. However, most of this research has been purely implementation-oriented, while systematic theoretical approaches have been scarce [69]. The Nave Physics movement [37] e.g., despite acknowledging the primacy of commonsense conceptualisation and reasoning, did very little to give a overall reasoned account of basic categories which may form the conceptual framework to be used by an intelligent agent [26].

The KR community (e.g. Gruber [22, 23] and Uschold [66, 67, 39]) gives practical motivation and guidelines for industrial ontology design, but is silent on the theoretical side, too.

However, there are exceptions. Indeed, the "formal ontology" of Guarino and Welty [69, 36] is an impressive synthesis of philosophical, logical and computational lines of thought. At the heart of their account is a formal theory of properties, which allows to characterise the semantical nature of KR predicates and, on the basis of that characterisation, to define constraints on IS-A links in taxonomies, thus giving clear formal criteria for consistent and computationally tractable ontologies.

Nonetheless, there might be some analytical work left for us to do. On the one hand the methodology of Guarino/Welty is still a work in progress that may be worthwhile assessing and modifying in the light of philosophical logic, e.g. the related work of Peter Strawson. On the other hand, if we apply Guarino's/Welty's methodology to an ontology combining Strawson's and Parsons' intuitions, we may be able to refine the latter in a reasoned way.

But what is more important, some distinctions drawn by philosophical ontology have never found their way into AI. Strawson's position on the prim-

itiveness of the concept of a person and the differentiation between (mere) bodies and persons turns out to be of crucial importance to the analysis of thematic roles, especially the multiple subject-roles.

Our work ends with the perspective of a computational ontology, that has emancipated itself from philosophy just as logic did at the beginning of the last century. We might well see the emergence of a new discipline, a "Naive Metaphysics" (to make an innocent pun), that combines commonsense ontologies and computational semantics.

0.3 Plan of the thesis

This thesis is divided into four chapters:

Chapter 1 is a brief introduction into the subject matter of ontologies for multi-agent systems. It mainly reviews basic notions about agents and ontologies and tries to relate both in a systematic and documented way.

Chapter 2 introduces the concept of a formal ontology according to Guarino as well as its methodology and main theoretical tools. The meta-properties used by Guarino and Welty to classify KR predicates according to their semantical nature are briefly reviewed. Finally we discuss the notion of ontological commitment that is related to the problem of how to restrict a KR language's interpretation to its intended model.

Chapter 3 details Guarino's / Welty's formal theory of properties, with some modifications inspired by Strawson. The semantic meta-features as well as the respective subsumption restrictions imposed by them are applied in the elaboration of a minimal semantic ontology merging and transforming Strawson's descriptive metaphysics of individuals and Parsons' account of thematic roles. The distinction between persons and non-persons is shown to be fundamental both for a basic account of particulars and for an analysis of thematic roles.

Chapter 4 applies the semantic ontology set up in the previous chapter in a basic client-server implementation of a reasoning agent able to parse natural language sentences and to prove their assertions in a database of facts representing the agent's beliefs. Agent communication occurs between the (human-operated) client and the server threads spawned off to deal with the client's queries. Our minimal ontology is used at the level of language parsing as well as at the level of the structuring of the database that represents the shared ontology of client and server

threads. The system is implemented in Qu-Prolog, a distributed and concurrent version of Prolog developed Peter Robinson (University of Queensland) and Pr. Keith Clark (Imperial College).

Chapter 1

Ontologies and Multiagent Systems

1.1 Multiagent Systems

1.1.1 Agents: an informal account

Wooldridge [71],p. 29, defines an *agent* as a "computer system that is *situated* in some *environment*, and that is capable of *autonomous action* in this environment in order to meet its design objectives". The agent interacts with its environment basically in two ways (cf. also [56], p. 31):

1. it gathers sensory input from, i.e. *perceives* the environment through *sensors*;

2. it influences the environment by output *actions* through *effectors*.

By *autonomy* we mean that agents have control not only over their own behaviour, but also over their *internal state*; in other words, agents should be able to act independently of humans and other computer systems [70]. We require that an agent's actions should be determined not only by its built-in knowledge, which has been programmed by a human, but also by its experience; one can say that agents are only autonomous insofar as their behaviour is based on their sensory input. Clearly, there is a strong connection between autonomy and *flexibility*: an agent that exclusively acts on pre-programmed assumptions is only be efficient in those cases where the environment happens to conform to these assumptions; hence a lack of flexibility ([56], p. 35).

The particularity of agents become more evident when compared to the computational concept of an *object*. Objects are conceived of as *encapsulating*

a state and as being able to apply *methods* on this state, as well as communicating by message passing ([71], p.31). First of all, agents differ from objects insofar as they display a greater autonomy than objects, especially by making decisions on actions to be performed. Second, agents are capable of a flexible behaviour, while objects do not provide for flexibility of method invocation or application. Thirdly, multiagent systems are essentially multi-threaded, each agent having at least one thread of control ([71], pp. 35-36).

Intelligent Agents

Intelligent or *rational agents* can be defined in various ways, depending on what the respective author thinks *intelligence* is. Russell and Norvig propose an analysis of intelligence in terms of performance or efficiency. Supposing that a performance measure of the degree of success has been agreed on, an *ideal rational agent* is an agent that, for each sequence of percepts or sensory inputs, should always perform the action that maximises the agent's performance measure, taking into account the evidence offered by the percept sequence as well as the agent's built-in knowledge about its environment ([56], p. 33). Intelligent behaviour would then be optimally adaptive behaviour in a changing environment.

Wooldridge ([71], p. 32, and [70]) opts for the criterion of *flexibility* and defines an intelligent agent as an agent that is able to perform *flexible* autonomous actions in order to achieve its design objectives. According to Wooldridge *flexibility* involves:

1. *reactivity*: intelligent agents are able to respond quickly to changes in their environment;

2. *pro-activeness*: their behaviour are goal-oriented, enabling them to *take the initiative* in order to achieve their goals;

3. *social ability*: intelligent agents have the capability to *interact* with other agents, i.e. to *communicate*.

As already mentioned, flexibility and autonomy are intimately related: proactiveness is essential to independent behaviour, while autonomy is the precondition of the capacity to adapt to the environment. Furthermore, flexibility is clearly performance-enhancing: a behaviour can only be efficient in a given environment, if it is able to respond to changes in the latter.

As agents are embedded in an environment, it follows that knowledge about the world and its structure is a prerequisite for their operation. Both conceptions of intelligence mentioned above involve world knowledge as a

central element, either as a criterion in terms of which to evaluate efficiency, or as a basis for a shared understanding or meaning in communication. We will return to that point later.

1.1.2 A Formal Framework for Intelligent Agents

A Basic Agent Typology

The intuitive notions discussed above can be detailed in a more formal way ([71], pp. 36-41)[1]. An agent's environment is a set of *environment states* $S = \{s_1, s_2, \ldots\}$; we suppose that the environment is in one of the states $s_1, s_2, \ldots$ at a time. We call the *effectoric capability* of an agent the set of actions A $= \{a_1, a_2, \ldots\}$, which the agent can perform. An agent can thus be simply defined as the function:

$action : S^* \rightarrow A$

which maps sequences of environment states to actions. This captures the idea that an agent chooses his actions according to his current experiences, which are represented as sequences of environment states the agent has so far registered. The behaviour of an environment is represented as the function:

$env : S \times A \rightarrow \wp(S)$

given a current environment state s and an action a, *env*(s,a) is a set of environment states possibly resulting of a in s; in case all values of *env* are singletons, the environment is *deterministic*, otherwise *indeterministic*.

A *purely reactive agent* is an agent that chooses its actions without taking into account its experiences, with the exception of the last situation (environment state) encountered. Such an agent that only lives in the present, so to speak, can be represented thus:

$action : S \rightarrow A$

According to the initial intuitive definition of an agent, however, we have to analyse an agent's decision function further and distinguish between the *perception* and *action* subsystems. Thus, the agent's faculty to observe its environment and the agent's decision making have to be modelled by two distinct functions, *see* and *action*:

$see : S \rightarrow P$

$action : P^* \rightarrow A$

where P is the set of percepts which represent the outputs of the *see* function; the decision function is no longer based on the actual environment states, but on the agent's perception of the latter. Two environment states are *percep-*

[1] We use the formalism of basic set theory: given two sets S and P, we note 'S^*' the set of sequences of elements of S, '$\wp(S)$' the powerset of S and '$S \times P$' the cartesian product of S and P.

tually equivalent, $s_1 \equiv s_2$ iff $see(s_1) = see(s_2)$. Obviously $\equiv$ is an equivalence relation, whose set of equivalence classes is a partition of its domain. The fewer these equivalence classes are, the smaller is the number of indistinct percepts in relation to the number of perceived environment states and the less 'resolving' the agent's perception is. If the number of equivalence classes of $\equiv$ is equal to 1, then the agent cannot be said to have a distinctive perception at all of its environment. In case the set of equivalence classes of $\equiv$ is equipotent to the set of environment states (i.e. when both have the same cardinality), then the agent's perception is perfect and the agent is in the state of *omniscience*.

Finally, we would like to capture the intuition that agents *maintain an internal state*. This internal state pertains an internal data structure, which holds information about the environment state, in other words, world knowledge. The idea is that actions are at least partially based on the internal state[2]:

$action : I \rightarrow A$

where emphI is the set of an agent's successive internal states. An additional function *next* updates the internal state of the agent by taking into account the agent's current percept and state.

$next : I \times P \rightarrow I$

Agents with state execute the following cycle: given an initial state i_0, and a percept of an environment state s generated by the function *see*, the *next* function updates the internal state, which is set to $next(i_0, see(\mathrm{s}))$. On the basis of the new internal state, an action $action(next(i_0, see(\mathrm{s})))$ is selected and the cycle begins anew.

Logic-Based Agents

The classic type of agent architecture, as it has been developed in traditional *symbolic* AI, are so-called *logic-based agent architectures*; the latter are a particularly convenient framework for (distributed) logic programming applications like the one which represents the implementational part of this thesis. In these architectures, decision making is essentially logical deduction. A logic-based agent's program, which represents its decision making strategy, is basically as a logical theory, while the action selection function reduces to a proof procedure ([71], p. 47, and [70]).

A simple example of a logic-based agents are so-called *deliberate* agents ([71], p. 43-44). The internal state of deliberate agents is basically a database consisting of first-order logic formulae describing the characteristics of the

[2]The *see* function remains the same.

agent's environment. In other words, the database represents the world knowledge of the agent and corresponds to a set of beliefs in a human mind.

The set of databases $D = \{\Delta_0, \Delta_1, \ldots\}$ is the powerset of the set of first-order logic sentences L. The internal state of a deliberate agent is a member of D. The decision making strategy is represented by a set of *deduction* or *inference rules* ρ; '$\Delta \vdash_\rho \phi$' stands for 'the formula ϕ is provable from the database Δ'.

There is no change as to the perception function; it should be noted, however, that the percepts of a deliberate agent are symbolic by nature, and are represented, like the agent's beliefs, as first order logic formulae ([71], p. 47):

$see = S \rightarrow P$

The *next* function is specified as to the nature of the internal state:

$next = D \times P \rightarrow D$

next takes as arguments a database and a percept and generates a new database. As to the agents decision function:

$action = D \rightarrow A$

it completely determined by the set of deduction rules ρ in the following way, given that the current state of the agent is the database Δ and the formula ϕ represents the action a:

1. *if* $\Delta \vdash_\rho \phi$, *then* $action(\Delta) = a$
2. *else if* $\neg(\Delta \vdash_\rho \neg\phi)$, *then* $action(\Delta) = a$
3. *else* $action(\Delta) = noop$

In other words, action a is return by *action* in case a is deducible from or compatible with database Δ; otherwise, *noop*, a default action indicating that no decision could be taken on the basis of the current state and percept, is returned.

Logic-based agents are reportedly not very adapted to rapidly changing environments, as their operation is based on the principle of *calculational rationality*, which consists in two assumptions:

1. the decision should be rational with respect to the situation when the deliberation process started, and
2. the environment should remain the same during the process of decision making.

Clearly, these weak constraints on the decision function do not guarantee that the respective action taken is still adequate to an environment which

may well have changed while the agent has been deliberating. Logic programs are very tractable from the semantical point of view, but not reputed to be computationally efficient ([71], p. 47). However, the development of distributed logic programming languages like Qu-Prolog [57] may render this appraisal obsolete.

1.2 Agent Communication

1.2.1 Communication and Coordination

One of the characteristics of intelligent agents mentioned above is social ability (cf. p. 2). The reason for including this requirement is that agents are rarely operating in isolation; as computing systems become increasingly interconnected through networks, the environment of computational agents normally includes other agents pursuing their own goals ([38], p.79).

The fact that multiple agents share the same environment entails the necessity to reduce resource contention, avoid deadlock and observe security conditions. The proper way to do this is coordination, which can happen either by cooperation or negotiation, depending on whether the participating agents are nonantagonistic or competitive or at least self-interested. Coordination of actions and behaviour enables agents as well as agent societies to be more efficient in the pursuit of their individual and common goals ([38], p.83).

The ability of an agent to communicate is basically its ability to send or receive messages over a communications network. There are essentially two messages types: assertions and queries; from the point of view of agents as knowledge bases, these message types correspond to the basic KB operations "tell" and "ask". Other types of messages relate to the set-up and regulation of the communication.

An agent should at least have the ability to accept information which in the simplest case has been communicated by another agent via an assertion. A *passive* agent should be able to a) accept a query and b) to send a reply to the sender through an assertion. An *active* agent must be able to send both queries and assertions. These capabilities allow an agent to exert some sort of control over other agents by making them accept and reply to queries. An *peer* agent is one which in a given dialogue with other agents assumes both a passive and an active role ([38], p.85).

Agents exchange and understand messages on the basis of communication protocols ([38], p. 79). Communication protocols can either be binary, involving one sender and one receiver respectively (unicast), or n-ary, where

one sender communicates with several receivers (broad- or multicast). Each of these protocols is based on the following parameters ([38], pp. 86-87):

1. the sender,
2. the receiver(s),
3. the communication language,
4. encoding/decoding functions for the enclosed message, as well as
5. the actions to be taken of the receiver(s).

1.2.2 Meaning: Pragmatics and Semantics

Agent communication has been actually modelled on human communication and the specification of agent communication protocols has drawn on *speech act theory* initiated by J. L. Austin [1] and developed by J. R. Searle [58] ([38], p. 87). The theory's basic intuition is that human utterances should be regarded as *actions* such as requests, assertions, commands or commitments. When someone says, "I thank you for your help", she does not merely state that she is thankful to the person to whom she speaks, but by making this utterance, she performs the very act which is intended to make this statement true.

According to Searle, a speech act has four aspects:

1. *locution*: the articulation of sound patterns bearing phonological and grammatical features;
2. *proposition*: the factual content of the speech act, which can be either true or false;
3. *illocution*: the intended communicative action;
4. *perlocution*: the effect on the hearer(s) resulting from the locution

In our example, the locution is the production of the phonological and grammatical structures corresponding to the sentence "I thank you for your help", the meaning of which is the speech act's proposition. The illocution in our case is the act of thanking the hearer for her help and the perlocution to the effect of the thanking on the person spoken to, e.g. flattering or embarrassing her.

The communicative intention represented by the illocution is the *illocutionary force*, or *performative*; Searle distinguishes five general types of performatives:

Assertives commit the speaker to the truth of the expressed proposition. Examples are assertions, statements, claims, hypotheses, descriptions, suggestions.

Directives are attempts by the speaker to get the hearer to commit himself to some future course of action. Examples are requests, commands, challenges, pieces of advice, warnings, permissions.

Commissives commit the speaker to some future course of action. Examples are offers, promises, pledges, threats, vows (oaths).

Expressives express the speaker's state of mind or attitude. Examples are thankings, complaints, apologies, greetings, congratulations.

Declaratives bring a change in (social) reality; they presuppose extralinguistic institutions for their performance (e.g. the Church for baptisms, law courts for penal sentences). Examples are marriages, baptisms, christenings, arrests, sentences.

A performative represents an attitude towards a proposition. The illocution can thus be regarded as the envelope of the proposition as message, as it specifies the type or force of the message whose content is the proposition. Davidson [15] analyses a sentence used to make a speech act as two distinct assertions:

1. the *indicative core* of the performative

2. the *mood setter* which asserts that a specific illocutionary force is "attached" to the indicative core

E.g. the performative "Close the door !" can be rewritten as "My next sentence is directive in force." followed by "You will close the door." The meanings of both indicative core and mood setter are fully specified by the conditions under which they are true or false. This is consistent with the fact that in logic-based architectures performatives ('tell(loves(hamlet,ophelia))') are truth-functionally evaluated in the same way as their contents.

Thus speech acts contain two meaning levels roughly corresponding to the pragmatics/semantics distinction. Traditionally, pragmatics is concerned with the use of utterances in communication, semantics with the dimensions of truth and reference ([38], pp. 84-85). The analysis of Davidson suggests that this distinction is not so clear cut as it seems, as some "pragmatic" aspects of speech act sentences might be analysed in truth-functional, hence semantic terms after all. Nevertheless we will retain the term "semantics" for

designating the meaning of the message (i.e. the indicative core) as opposed to the "pragmatics" or meaning of the illocution (i.e. the mood-setter).

Agent interaction and communication rests on the fundamental separation of the semantics of the communication protocol (the set of rules specifying valid message types and allowing agents to understand them) and the semantics of the enclosed messages. The former is always domain-independent as all agents have to share it ([38], p. 88). When we speak of "multi-agent semantics" in the context of this thesis, we actually mean primarily the semantics of the messages which are the indicative core, so to speak, of the respective agent communicative acts.

1.2.3 The Agent Communication Language KQML

It may be worthwhile to have a closer look at one widespread agent communication protocol, the Knowledge Query and Manipulation Language (KQML), which was developed as part of the DARPA Knowledge Sharing Initiative ([18], [40]). KQML specifies three types of performatives which can be used by agents to share knowledge [40]:

Discourse performatives are actually closest to speech acts in the linguistic sense; they are used in a information and knowledge exchange type of conversation between agents (e.g. ask-if,tell,...).

Intervention and Mechanics of conversation performatives are used to intervene in a current agent dialogue, either to terminate it prematurely (error, sorry) or to override the default protocol (e.g. standby, ready,...).

Facilitation and Networking performatives are not speech acts in the strict sense; their role is to allow agents to find other agents that can handle their queries (forward, broker-one,...).

KQML-performatives are defined as ASCII-strings in Lisp-like Polish notation; their parameters are indexed by keywords called *parameter names*. Each parameter name is preceded by a colon and followed by the corresponding *parameter value*. Here is a simple example of such a structure:

```
(KQML performative
                    :sender     <word>
                    :receiver   <word>
                    :language   <word>
                    :ontology   <word>
                    :content    <expression>
                    ...)
```

Some parameters as `:sender` or `receiver` are used for message passing, while `:language`, `:ontology` and `:content` together define the semantics of the enclosed message. `:language` specifies the language in which the message is coded (in contrast to the language of agent communication), `:content` is the field corresponding to the message (which might be just another KQML performative) and, last but not least, `:ontology` stands for the ontology used in the message.

In this context, "ontology" means a "vocabulary" of terms, together with class taxonomies and relationships. The fact that ontology is an explicitly specified parameter for the semantics of messages in a standard agent communication protocol just reflects the acknowledged role of ontologies in multiagent semantics.

1.3 Ontologies

1.3.1 Ontology as a Basis for Multiagent Semantics

So far we have seen two aspects of agent architectures in which ontology seems to be directly involved: (world) knowledge representation (cf. p. 5) and agent communication (cf. p. 10). Both aspects are related insofar as they both represent aspects of understanding in the widest sense.

As Guarino [27] points out, in the last decade there has been a shift from a functional to a modelling view of knowledge in AI. Traditionally, the chief criterion for evaluating knowledge has not been truth, but functional utility: under the condition that an agent's goals are observable, the soundness of its world knowledge can be judged according to the efficiency of its goal-driven behaviour. In the modelling view, knowledge is primarily understood in terms of the classical concept of truth as correspondence to the facts: knowledge should somehow reflect the structure of the agent's environment independently of the agent's current task. The aim of the modelling activity is not to elaborate techniques for assessing behavioural rationality, but the specification of a mathematical structure composed of a domain of individuals and a set of classes and relations defined over that domain in order to interpret and truth-functionally evaluate an agent's beliefs, which are viewed as propositions capable of being true or false.

Domain analysis as a task-independent inquiry is also relevant for agent communication. Entire societies of agents differing from each other have to interact and communicate in ever more various ways and situations; thus coordination is made easier in as much as the agents' knowledge truthfully reflects the environment of the multiagent system. Furthermore, knowledge

is shareable only insofar as it is objective i.e. true.

It would seem that knowledge can only be interesting by itself if it actually corresponds to the reality / environment beyond the agent's representation, in other words, if knowledge is actually true. According to Guarino [27], the philosophical discipline of ontology as the study of the objective structures of the world may thus be relevant for the task of modelling knowledge acquisition and representation. In the knowledge sharing community, however, "ontology" still primarily means "a specific top-level knowledge base".

Linking the two aspects of domain knowledge and communication together, we can say that ontology is a basis for multiagent semantics. According to Davidson [14], this claim can be justified as follows: Successful communication between agents rests on the fact that the exchanged messages are to a large extent true. The truth of agent messages (viewed as propositions embedded in illocutions) is mainly determined by how adequately they represent the facts of the agents' environment. In other terms, the conditions under which the embedded propositions are true can be specified by what can be the case (or not) in the world, i.e. by indicating the possible objects, classes and relationships in the agents' environment. Now, in a truth-functional perspective, the truth conditions of propositions constitute their meaning. Thus (multi-) agent semantics is ultimately based on an ontology as a domain theory, a formal description of the entities and structures of reality.

Frank [20] argues that, as ontologies can help to construct multiagent systems, computational models of agents and agent communities may contribute to understand and evaluate ontologies. Computational models of agents and their environment make it possible to simulate their interactions systematically. In the context of these simulations, ontologies can be implemented in such a way as to integrate in the same computational framework the model of the reality/environment and the model of its representation. Thus it is possible to evaluate ontologies in concrete multiagent applications, and to compare their respective advantages and trade-offs.

1.3.2 Ontology as a Shared Understanding

From the point of view of knowledge representation, an ontology is essentially a shared understanding (Gruber [22] and Uschold [66, 67, 39]). An ontology represents an abstract and simplified worldview that is shared by the members of a multi-agent system. Such a worldview can be regarded as a set of concepts (entities, properties, events) supposed to exist in a certain domain, together with the respective definitions and inter-relationships, a *conceptualisation* in other words. Formalising the knowledge about a given

domain turns the set of objects in that domain into the universe of discourse of the language in which the formalisation is embedded. The entities in the universe of discourse and their inter-relationships are mapped onto the vocabulary which is used to represent the domain knowledge. In formal terms, an ontology is basically the statement of a logical theory (Gruber [23]).

Ontologies exhibit a great variety pertaining to formality and genericity [66, 67]. Highly generic ontologies, so-called *upper-level models*, serve to structure wide areas of human knowledge, such as natural language understanding. Ontologies of lower genericity are obviously designed with specific applications in mind and are therefore commonly referred to as *application ontologies*.

There is also a wide range of possible subject matters of ontologies, which can be roughly grouped together in three categories [67]:

1. subjects such as medicine, geography or botanic, considered separately from the tasks and problems associated to them;

2. the subject matter of problem solving,

3. the subject matter of languages for knowledge representation.

Ontologies whose topic falls into the first category are referred to as *domain* ontologies. *Task*, *method*, or *problem solving* ontologies deal obviously with matters of the second category, while subjects of the third are thematised by *representation* or *meta-ontologies*.

The kind of ontology required as a basis for multi-agent semantics (p. 11) must obviously be highly general in order to have sufficient explanatory power for generating the semantic structures that underly the communication of intelligent agents. This is certainly the case as far as human-machine interaction using natural language is concerned. According to our previously stated taxonomies, an ontology underlying a multi-agent seamtics is a *high-level meta-ontology*, a reasoned classification of top-level categories involved in speech and thought.

Chapter 2

Tools for a Computational Ontology

2.1 Towards a Naive Metaphysics

2.1.1 The Idea of a Formal Ontology

Ontology as the philosophical discipline that deals with the formal *a priori* nature of reality may contribute in a significant manner to tackle computational problems of knowledge representation and multiagent semantics. However, as Guarino [27] observes, there is an evident lack of interest in traditional AI for such aspects as ontology and conceptual modelling. This seems to be due to the necessity of a strong *interdisciplinary* approach while studying these problems, as not only a familiarity with the formal instruments of logic and computing science is required, but also a certain affinity to philosophical conceptual analysis and commonsense reasoning as well as a solid background in linguistics.

Guarino [27] expresses his regret as to the narrow view on the subject matter of ontology as it has been exhibited by even those research trends in AI which somehow attempt to focus on the commonsense modelling of the world, amongst others the "Naive Physics" current (Hayes [37]). Indeed, it seems that ontology research generally responds to immediate implementational problems, leaving more or less completely out of account the philosophical contributions to the domains of commonsense world representation and natural language. Hayes [37] e.g. proposes to formalize the naive worldview with concepts chosen ad hoc and considers as both improbable and undesirable a systematic account of ordinary language concepts within a framework of a limited number of primitive notions. Thus, a formal ontology as a computational discipline is still to be established.

Before we specify the exact subject matter of this discipline, it may be useful to make some preliminary remarks as to the distinction between epistemology and ontology. Epistemology is concerned with the form, i.e. the organisation and structure of knowledge thought of as essentially being a set of propositions. Thus epistemology reduces mainly to the study of the form of (empirical) propositions, with the aim of elucidating the inference process. Ontology, by contrast, deals with the structure of the world apart from its representation in knowledge. This distinction has been drawn by researchers in AI since its very beginnings, not the least by Hayes [37], who stressed the necessity to concentrate not merely on knowledge form, but also on knowledge content, in order to be able to achieve a task-independent analysis of concepts.

Formal ontology, according to Guarino [27], is the systematic description of the *forms* of entities in the world or, to be more precise, the theory of *a priori* distinctions:

1. among the elements of reality (objects, events,etc.),

2. among the meta-level categories used to reconstruct reality (concepts, properties, roles, etc.).

The distinction between *formal* and *material ontology* corresponds to that between *formal* and *material relations*. Under the supposition that a given domain has been partitioned into a set of disjoint primitive subdomains (or categories), formal relations can be defined as characterising the connections and differences between these subdomains, while material relations specify particular subdomains in more detail. Obviously, formal ontology deals exclusively with formal relations such as identity or dependence [21].

Formal ontology is based to a large extent on achievements in analytical philosophy, especially on the programme of a 'descriptive metaphysics' initiated by Sir Peter Strawson. Thus, as the analytical school has given birth to speech act theory and the study of agent communication, it may be once more a source of fruitful intuitions that could develop into a "naive metaphysics" as a commonsense ontology for multiagent semantics.

2.1.2 Strawson's 'Descriptive Metaphysics'

Strawson's [60, 61] draft of a theory of individuals is an ontology in the sense of a basic classification of the entities in the world. This theory is an instance of "descriptive" in contrast to "revisionary" metaphysics. Descriptive metaphysics focuses on the description of the conceptual scheme used in everyday thought and speech about the world. Revisionary metaphysics aims

at criticising and improving the overall conceptual structure of commonsense ([60], p. 9). Obviously, revisionary metaphysics presupposes its descriptive counterpart, as the reconstruction of and improvement on the commonsense view it proposes can only be assessed on the background of the very basic conceptual background it criticises [6].

Descriptive metaphysics rests on the assumption that there is a fixed set of central classifications that represent the "commonplaces of the least refined thinking" at the heart of the human "conceptual equipment" ([60], p. 10). The subject matter of descriptive metaphysics is the *conceptual framework* according to which we perceive and describe the world we live in. Descriptive metaphysics captures the elementary intuitions concerning the ways agents interact with their environment and how they conceptualise it.

The fundamental elements of our conceptual framework are to be sought in the very pre-theoretical, everyday notions which must be mastered, before the more sophisticated theoretical constructions can be endeavoured. An overall, basic feature of the conceptual scheme that is of interest in ontology consists in the use of a range of everyday concepts (e.g. *apple,desk,glass,...*) which may be classified as an abstract type (e.g. *physical object*) according to certain commonalities. A concept or concept-type is of fundamental and non-contingent nature for the conceptual scheme with which we are actually equipped iff it is a necessary characteristic of this conceptual scheme to contain that concept (as it may be the case for *body* and *event*).

To sum up, then: the ultimate aim of descriptive metaphysics is to reveal the structure of interrelated basic concepts or concept-types which constitute necessary overall features of the conceptual framework on the basis of which we as agents *de facto* operate in everyday practice ([61], pp. 23-24).

2.1.3 The Ontological Level

In order to situate ontology in the context of research on knowledge-based systems, Guarino [26, 27] adopts and extends Brachman's [2] classification of knowledge representation levels according to the primitives they put at the user's disposal. There are altogether six different representation levels in knowledge-based systems, of which the so-called ontological level plays a pivotal role by restricting the leeway for the user's interpretation:

Implementational level Primitives at this level are just memory cells and pointers, which are the basic constituents of data structures which are yet uninterpreted.

Logical level This is the level of formalisation. Its primitives are propositions, predicates, functors and logical operators, whose semantics can

be specified in terms of the entities and relations in a chosen domain. However, first order logic is ontologically neutral in the sense that it makes no assumptions as to the domain(s) in which its formulae can be satisfied. The interpretation of logical formulae is unrestricted and arbitrary.

Epistemological level Epistemological primitives have a knowledge structuring function as they allow to detail which types of concepts and conceptual relations may exist in a KR formalism. The idea of a concept or conceptual relation is defined independently from the specific cognitive content. However, interpretation is still fairly unconstrained.

Ontological level Ontological commitments linked to the linguistic primitives are made explicit, either by restricting their semantics or by adding meaning postulates to the language itself. Primitives become thus ontological relations which have to satisfy meaning postulates; the freeway for interpretation is limited by specifying the (formal) meaning of the fundamental categories used to describe the world. A KR formalism is said to be *ontologically adequate*, iff either the language has enough expressive power to formulate the meaning postulates for its primitives, to "tag", so to speak, its primitives with their respective ontological category, or if there are ways to restrict the semantics of its primitives.

Conceptual level At this level primitives are specific concepts and conceptual relations with a definite cognitive content. They constitute a repository of basic notions to categorise the domain. The restrictions at the ontological level can be further refined, interpretation is however subjective and left to the discretion of the user, in the limits imposed at the higher levels.

Linguistic level The KR system's "surface" is language, whose nouns, adjectives and verbs express the basic concepts. Again, interpretation is subjective, insofar it is communication-oriented and context-sensitive.

To give a concrete example: Suppose one would like to express the fact that some elderly people are students. The representation at the logical level would be "$\exists$x (Elderly(x) $\wedge$ Student(x))", a first-order formula satisfiable in an arbitrary enumerable domain. At the epistemological level, one would try to determine which of the two predicates stands for a type or a role; probably, one would classify *Elderly* as a concept and *Student* as a role. The ontological presuppositions involved in the meaning of types and roles

Level	Primitives	Interpretation	Main feature
Implementational	Memory cells, pointers	-	-
Logical	Predicates	Arbitrary	Formalisation
Epistemological	Structuring relations	Arbitrary	Structure
Ontological	*Ontological relations*	*Constrained*	*Meaning*
Conceptual	Cognitive primitives	Subjective	Conceptualisation
Linguistic	Linguistic terms	Subjective	Language dependency

Table 2.1: Classification of KR primitives (after [26, 27])

are stated explicitly at the ontological level; they receive a standard formal interpretation. Thus the use of *Elderly* and *Student* would be restricted, as the the user would not be free to use *Student* as a type or *Elderly* as a role. At the conceptual level, both would be applied as specific concepts, and be expressed by specific nouns and verbs at the linguistic level.

2.2 The Methodology of Formal Ontology

2.2.1 IS-A overloading

According to Guarino [30, 31] and Bouaud [3], ontologies are built around taxonomies, sets of terms structured by partial ordering relations referred to as *IS-A* or *subsumption*. Given two unary predicates A and B, A IS-A B iff $I(A) \subseteq I(B)$, where I is the interpretation function which assigns to each term in the respective language an element or subset in a domain.

As Guarino [30, 31] points out, most ontologies suffer from the fact that lexical relations between terms do not consistently correspond to ontological relations between classes in the domain and vice-versa. One of the reasons for this discrepancy is that multiple inheritance is commonly used to reflect polysemy, which results in an overloading of the IS-A relation. There are five main categories of IS-A overloading:

1. *Confusion of senses:* e.g.: "A window is both an artifact and a place" (Microkosmos); different meanings of a word are merged in one single set that inherits from several supersets.

2. *Reduction of sense:* e.g.: "A physical object is an amount of matter" (Pangloss); the IS-A link reflects just one aspect of the overall sense of a term.

3. *Overgeneralisation:* e.g.: "A place is a physical object; An amount of matter is a physical object" (WordNet). In this case, a category subsumes heterogeneous subcategories.

4. *Suspect type-to-role link:* e.g.: "A person is both a living being and a causal agent" (WordNet); types (*person, living being*) and roles (*causal agent*) are mixed together.

5. *Confusion of taxonomic roles:* This happens when every unary property of a set is specified in terms of a superset to inherit from. No distinction is made between nodes that fulfill a central organising taxonomic role and those that designate an individual property. Such properties should actually not been included in the taxonomy, but only be indicated by attributes.

IS-A overloading can be avoided if IS-A links are restricted to nodes which have similar formal properties, foremost similar identity criteria [30, 31, 36]. In such a way a straightforward correspondence between conceptual taxonomy and ontological structure is established, endowing semantics with all the clarity needed. Thus the specification of formal relations such as identity and dependence, which is the task of formal ontology, is essential for the design of knowledge-based systems.

2.2.2 Tools for a Formal Ontology

Formal ontology as the systematic study of *a priori* ontological distinctions on the basis of formal relations and properties, presupposes, according to Guarino [29, 30, 31] (updated by [34, 35, 36, 69]), the following main auxiliary theories or theoretical tools :

Theory of parts and wholes: the specification of the parthood relation and the formal property of *unity*. Also called *mereology*, the theory deals with the questions of what it is to be a part of some entity and how parts connect together to constitute a whole. The central notion is *unity*, which expresses the fact that parts of a given whole are linked together and are distinguished from other entities in the world through a *unifying condition*. Research in this area is mainly inspired by the seminal work of Simons [59].

Theory of identity: the specification of the formal relation of *identity*. The latter expresses the fact that two instances of the same class can be distinguished or reckoned to be the same on the basis of an *identifying condition*. Obviously identity criteria are fundamental for this issue. An identity criterion (IC) for a property P is the relation I_p which satisfies the condition: $Px \wedge Py \wedge I_P xy \rightarrow x=y$. A property P, for which this relation I_P can be defined, is said to *carry an IC* for the

entities which instantiate P (a more formal definition will be presented in the next chapter). The conceptualisation of the world determines which classes are ascribed an IC. For the domain of everyday life, one can assume that IC's are universally shared by all humans, insofar they presuppose the spatio-temporal framework intrinsic to human perception (Strawson [60]).
Related to the issue of identification are two other problems:

- *identity through change* - how an individual undergoing changes can be still reckoned to be the same entity;
- *re-identification* - how the same particular identified at one occasion can be reidentified in another situation, if it has not been observed continuously between the two moments.

Theory of essence: the specification of the formal properties of essentiality or *rigidity* of properties. The main issue dealt with is whether a given property is necessary or accidental to some or all of its instances, whether its attribution holds in all or only some of the possible worlds. This theory is basically applied modal logic.

Theory of dependence: the specification of different kinds of *dependence* between particulars or particular types (classes). Following Simons [59], Guarino adopts a strong concept of ontological dependence involving mereological notions, while Strawson [60] introduced the weaker formal property of identification-dependence (cf. infra).

In the context of this thesis, we will actually concentrate on the three last aspects of formal ontology, leaving aside the issue of unity. Actually, Guarino's basic classification of properties (as to be found in [34, 69]) does not make use of this *a priori* criterion. Furthermore, for our purposes the formal relations of essence, identity and dependence (which we will propose to define in Strawson's sense) are quite sufficient. The exact mathematical definitions will be presented an discussed in the next chapter.

Before we will continue to develop on the subject of some formal properties, we would like to mention a set of principles for taxonomy structuring proposed by Bouaud a. o. [3]. Even if not referred to by Guarino, these rules summarize rather well the basic intuitions behind the formal specifications in the next chapter. They have been mainly inspired by de Saussure's [10] structuralist methodology for linguistics :

P1: Similarity Principle A child node must have the same category as its parent node. In other terms, all children of a node must have a

common meaning. The necessary condition for being the child of a node is to be of the parent's type. E.g. events, states and processes are all occurrences.

P2: Specificity Principle The specific difference or distinctive property of a child is a necessary and sufficient condition for its existence in the subtree. E.g. Being an occurrence that lasts indeterminately is a complete specification of the concept of *state*.

P3: Opposition Principle Siblings relate to each other in a system of oppositions; all children of a node must be pairwise incompatible. (The application to our example is obvious.)

P4: Unique Semantic Axis Principle Principles P2 and P3 can be superseded by the requirement that all subcategories of a given concept must differ from their parent on a common axis or dimension, each child bearing a unique value defined in that dimension. E.g. the subtypes of *occurrences* can be seen to represent distinct values on the dimension of *completion in time* or *aspect*.

P1 - P4 guarantee that the whole taxonomy constitutes a tree with a unique root, a structure that is computationally very tractable.

2.2.3 Identification, reidentification and dependence

At this stage It may be helpful to become familiar with some central *a priori* concepts such as identity and dependence before attacking their mathematical definitions. Strawson's intuitive explanations of how agents identify entities (including themselves) in their environment and how this reflects basic structures of everyday speech and thought about the world are not only a good basis for the somehow straining formal analysis to follow, but illuminates pragmatic criteria which can be fruitfully applied in the study of basic particular types.

Identification and ontological priority

In the context of communication, there is a hearer and a speaker perspective of identification. A speaker who uses a singular expression in order to refer to a particular is said to make an identifying reference to that particular. A hearer who knows which particular a speaker is identifyingly referring to is said to be able to identify that particular. Finally, a speaker who refers identifyingly to a particular can be said to identify the latter if the hearer

can identify it (Strawson [60], pp. 15-16). If I say to you: "A cat is on the mat" in the presence of a feline lying on your doormat, I make an identifying reference to the animal with the noun phrase "a cat". If you can single out the cat on your doormat on the basis of my identifying reference to it, you are able to identify that feline (in the hearer sense) and I have felicitously identified the latter (in the speaker sense).

Clearly, the possibility to identify the particulars of a certain type is a necessary condition for including this class in our ontology. Furthermore it may exhibit a particularity of our conceptual equipment that the identification of some classes of particulars is supervenient on that of another type of particulars, but not vice-versa (e.g. persons and their feelings, objects and their changes). In other words, if instances of a particular-type B can only be identified by previously identifying instances of a particular-type A (and not the other way round), then it is a characteristic of our conceptual framework that the possibility to talk about B-particulars presupposes the possibility of the discourse about A-particulars. In such a case, we say that A-particulars are ontologically prior to B-particulars. Particular-types showing no identification-dependence on any other class of individuals can be regarded as basic or fundamental ([60], pp. 16-17). In the next chapter, we will review the evidence for bodies and persons being basic particular-types.

Ontological priority is defined in terms of identification-dependence only and does not imply that B-particulars could be reduced to A-particulars. Basic particulars are basic in terms of identification, not of existence. Strawson's concept of identification-dependence is not to be confused with Guarino's notion of ontological dependence [36, 69].

Empirical conditions for successful identification

Hearer identification can be regarded as successful in case the hearer knows that the particular being referred to by the speaker is the same as some particular about which he knows an identifying fact other than being referred to by the speaker. An identifying fact about a particular is a state of affairs which holds for that particular only. In other words, the hearer should be able, if required, to produce a description of that particular in terms other than those related to the speakers reference to it ([60], p. 23). In order to be able to identify the cat on the mat, you have to know some distinctive fact about the feline apart from my referring to it with the noun phrase "a cat"; e.g. you might have to be able to describe the colours and shades of its coat.

There are two types of hearer-identification to be distinguished. So-called (story-) relative identification is the identification of a particular relative to a set of particulars which is itself only identified as belonging to the domain

of entities being talked about by the speaker. It is an identification inside of a speakers story, not within history. E.g. if I talk to you about a particular cat I lived with at a certain time and I casually mention "it's mother", you might be able to relate, merely within my story, the particular mentioned (the mother) to the object I'm talking about all the time (the cat I have lived with). By contrast, full or demonstrative identification requires the hearer to be able to directly locate the particular referred to by the speaker. Under an empiricist premiss, this means that the hearer must be able to single out that particular by perceiving or having perceived it. Thus demonstrative identification involves the element of perception or experience; it is relative to the range of particulars present to the sensory perception of the hearer ([60], pp. 18-19). In our previous example, you are required to locate the cat on the mat, having the opportunity to look around and to inquire the whereabouts of your pet or mat.

Clearly, even though a particular may not be immediately demonstratively identifiable, it can be rendered so by using a description which links it to a different particular which is itself liable to be identified in this way. Story-relative identification obviously rests mediately on the demonstrative identification of the storys speaker. So every identifying description involves in one way or another demonstrative identification ([60], pp. 21-22).

Space-time as the fundamental framework for identification

Demonstrative identification itself must rely on a system of relations which encompasses all particulars that can be directly located, i.e. that are accessible to immediate perception, a system in which each of them can be assigned a unique place. For objects of perception, this system of relations is typically that of spatial and temporal relations .

As users of this system, we have our own place in it, that we can demonstratively identify as a common point of reference and from which we extend the axes of the spatial and temporal dimensions. Thus, we are able to locate and to identify each particular either directly or indirectly (via another particular) by a description relating it to that common reference point in a unique way ([60], pp. 22-23).

Suppose for a moment that there was be a way to describe a particular in an identifying manner without relating it to any element in the spatio-temporal framework. This would mean that the particular in question was be completely isolated and cut off from the rest of our systematised body of knowledge. Clearly, that particular would have no role to play inside of our conceptual framework and, being thus practically irrelevant for our knowledge about the world, would therefore be as good as non-existent ([60], p. 28).

Due to its pervasiveness and comprehensiveness, the system of spatial and temporal relations is uniquely convenient to act as a general framework in which our individuating thought about particulars can be embedded and structurated. Each particular can be assigned a unique position in that system or is such that it can only be identified in relation to a particular which can be located spatially and temporally. Not only is there no other set of relations which is equally qualified to serve as a framework for identification, it can even be claimed that each way of identifyingly describing a particular involves directly or indirectly spatio-temporal characteristics ([60], pp. 25-26). Or, to put it Guarino's terms, every identity condition ultimately rests on the dimensions of space and time. Thus it seems that it is a spatial and temporal structure is a necessary feature of empirical reality ([60], pp. 29).

Reidentification

But being endowed with a system of spatio-temporally interrelated particulars not only involves being able to identify a certain particular at one occasion, but also to identify a particular observed on or described with respect to one occasion as being identical to a particular perceived on or described with respect to another occasion. In addition to simple speaker-hearer or referential identification something more is required for agents to interact with their environment, namely re-identification.

Obviously, the possibility of re-identification, the existence of criteria to that effect, is a necessary condition for our having the idea of continuously existing particulars ([60], p. 31). Whatever our criteria for re-identification are, they must cater for the fact that we do not perceive the spatial framework as a whole, that we do not even observe a part of it continuously and that our relative position in it changes. As there are discontinuities and limits to the perception of (gradual) change of spatial boundaries and relations, we have to rely largely on so-called qualitative recurrences, on the repeated observation of the same patterns or configurations of particulars ([60], pp. 32-33). This reliance is a necessary condition for our having the idea of a spatio-temporal framework of reference, which is itself a intrinsic characteristic of the conceptual equipment we actually have and use.

Suppose, one would doubt the existence of criteria of reidentification in the absence of unrestricted observation of change. This would lead to adopting a new space time framework for each uninterrupted stretch of observation. But in this case, the actual cause of the doubt would disappear too, insofar there would be no all-encompassing system of reference in which alone the problem of reidentification can occur ([60], p.35). Thus the reliance on qualitative recurrences for the purposes of reidentification is rational and so is

the presence of a spatio-temporal framework of reference in our conceptual scheme.

2.3 Ontological Commitment

2.3.1 Ontological Commitment as Model Restriction

The concept of ontological commitment has been introduced first in philosophy before it has been adopted in the KR community. It originated in philosophical and mathematical logic and is used to refer to how the use of singular terms in sentences of the indicative mood actually presupposes or postulates the existence of the things referred to. This is especially important in mathematical logic that addresses, amongst other issues, the question of how much is implied as a mathematical reality by the axioms of number theory or set theory.

Quine [48, 50] e.g. regards (bound) variables as the ultimate device of reference; they can be thought of as ranging over a domain of individuals, the universe of discourse. Following Herbrand, one could put it like this: a (bound) variable in a monadic predicate refers *en masse* to every constant in a domain D, whose substitution for that variable holds a sentence true in D. Consequently, Quine defined "existence" as "being the value of a bound variable", thus linking the use of (bound) variables ranging over a certain universe of discourse U to an ontological commitment to the individuals of the domain U.

For the KR community, Gruber [23] defined an ontological commitment as an agreement to use a common terminology specification. This specification has the form of an axiom set, consistency to which is essentially an ontological commitment. However, this strictly syntactic view excludes the possibility that two different vocabularies can actually correspond to the same ontology.

Guarino [24, 25] therefore prefers a semantical conception of ontological commitment nearer to the original philosophical acceptation. However, he considers the Quinean notion of ontological commitment as too weak. Indeed, as already stated, predicate logic is essentially neutral from the ontological point of view. This means that the number of possible models of a theory stated in first order logic exceeds that of the *intended* ones, which actually correspond to the realm of facts the theory is supposed to have as its subject matter. An ontological commitment should thus be seen as a means to constrain the leeway of interpretation of a formal theory.
The formalisation of ontological commitments happens at the *ontological*

level, where the meaning of a (linguistic) KR primitive is objectively determined (cf. p. 16). This formalisation amounts to specifying the intended meaning of the vocabulary of a first-order language by imposing constraints on the choice of possible models. By explicitly stating the ontological status of the modelling primitives and their formal relationships, an ontological commitment acts as a mapping between the respective language and the formal structure which can be regarded as its ontology.

The ontological commitment pertaining to a theory T stated in a first-order language L has to be expressed in terms of a language L' containing L as its subset. Indeed, the meta-language L' is basically the language L extended by the formal means of modal logic and model theory.

2.3.2 A Modal Account of Ontological Commitments

In [24, 25], Guarino gives a strict formal definition of the concept of ontological commitment. This formalisation is actually used as the basis for a classification of unary predicates that has been partially superseded by later attempts of the same author (e.g. in [35, 36]). However, the theory is still useful as a specification of the notion of ontological commitment.

Let L be a first-order language with signature Σ = <K,R>, where K is a set of constants and R a finite set of predicates. D is the intended domain of L and M* the set of the possible models M = <D,I> of L, where I is an interpretation of L's constants and predicates in D. By adding the modal operators $\Diamond$ and $\Box$ to L, one gets its *modal extension* L_m.

Definition 2.1 *A* constant-domain rigid model *of L_m based on D is a structure $S = <W,\mathrm{R},D,\ \mathrm{F}_K,\mathrm{F}_R>$, W being a set of possible worlds,* R *a binary (accessibility) relation defined on W,* F_K *a function assigning each constant of L an element in D and* F_R *a mapping which attributes to each possible world in W and predicate of L in R a relation defined on D [25, 19].*

The accessibility relation R is conceived by Guarino as an *ontological compatibility* relation. Two worlds are said to be ontologically compatible iff they contain alternative states of affairs that do not contradict the *a priori* properties of the domain. A world in which a certain bough is golden is compatible with one in which the same bough is green, but is incompatible with a world in which the same particular is not a bough, insofar as "being a bough" is a property carrying an identity condition. R is supposed to be an equivalence relation, thus allowing the application of the strongest modal logic, S5.

Definition 2.2 *A* compatibility model *for L_m on the basis of D is simply a constant-domain rigid model for L_m on D, where* R *is required to be the relation of ontological compatibility.*

Finally, we adopt the definition of "ontological commitment" given in [27]:

Definition 2.3 *An* ontological commitment *for a language L of first-order predicate logic based on a domain D is a set C of compatibility models for the modal extension of L, L_m, on the basis of D.*

As the ontological compatibility is an equivalence relation, an ontological commitment can be stated within modal theory S5.

Definition 2.4 *A formula ϕ of L_m is valid in the ontological commitment C iff ϕ is valid in each model in C.*

Indeed, in [24, 25], Guarino restricts the term "ontological commitment" to a set of compatibility models for a mereological extension of L. But this is just one possible modal extension, as for some KB applications other formal tools may be required, as e.g. temporal or epistemic logic. We prefer to stick to Guarino's initial definition and to refer to other conceptualisations as ontological commitments in the extended sense.

A KR formalism for which an ontological commitment has been defined can be guaranteed to be ontologically complete, insofar it is possible to restrict the semantics of its primitives, especially with respect to the subsumption relation, in a systematic way. As we shall see in the next chapter, the semantics of IS-A can be consistently regulated via meaning constraints motivated by the *a priori* classification of properties on the basis of formal ontological features.

Chapter 3

Outline of a Naive Metaphysics

3.1 A Formal Theory of Properties

3.1.1 Goals and Tools of Analysis

Research context

The aim of this section is to present a set of meta-properties allowing to specify the ontological features of the (unary) predicates of a given first-order language L_0. This specification can serve as a basis for a classification of properties which in its turn may be used to formulate constraints for the IS-A or subsumption relation. Our main purpose for adopting and adapting these formal considerations is to acquire an array of tools to assess, criticise and modify a minimal set of categories which can serve as a basis for multi-agent semantics in the sense of the logical form of assertive agent communication messages.

We review and complement a proposal of Guarino and Welty, as presented most recently in [69] and [36], based on their previous work, e.g. [32, 33, 34, 35]. The formal analysis provided by them is mainly inspired by the works of Strawson [60], Lowe [42] and Simons [59]. To a large extent, this section will be a paraphrase, with some occasional modifications and additions.

As already mentioned in the last chapter, there are three meta-properties that are of interest for us: essence, identity and dependence. The two former are mainly relevant for the specification of subsumption constraints, on the basis of which we will criticise Parsons' account of thematic roles. The notion of (identification-) dependence will be used (after Strawson's example) as a tool for isolating so-called basic particular-types. The results of both lines of thought will be integrated in a naive metaphysics in the sense of a minimal ontology.

Logical Instruments

The object or modelling language L_0 as well as its meta-language L_1 is assumed to be a language of first-order predicate logic. For each (unary) predicate ϕ of L_0, L_1 contains a name designating ϕ; meta-properties of L_0-predicates are L_1-predicates which take L_0-predicate names as arguments. A set of *reflection rules* assign to each meta-property μ in L_1 an axiom scheme α of L_0 such that μ holds of a L_0-predicate iff α is true in L_0. Thus, an ascent to second-order predicate logic can be avoided.

The formalisation is based on first-order logic plus identity, extended by a basic temporal logic (predicates can have a time variable as an additional parameter) and a quantified S5 modal logic with Barcan Formulas. The domain of quantification is assumed to be the same in each possible world and includes all the objects which can be possibly thought of. This makes it necessary to introduce the notion of the *actual existence* of an individual x at a particular time t in our world, written "$E(x,t)$". The following definition will be used to exclude properties which are trivially (un)instantiated:

Definition 3.1 *A property ϕ is* discriminatory *iff ϕ possibly holds of some entity while possibly not holding of another entity:*

$$\Diamond \exists x\, \phi(x) \wedge \Diamond \exists x\, \neg\phi(x)$$

In other terms, discriminatory properties are properties whose instantiation is contingent, but not impossible. From now on, "property" will be used as synonymous to "discriminatory property" throughout the rest of this thesis. The formal definition of the subsumption relation reads thus:

Definition 3.2 *A property ψ subsumes a property ϕ ($\psi \preceq \phi$) iff every instance of ϕ is necessarily an instance of ψ :*

$$\Box \forall x\, \phi(x) \rightarrow \psi(x)$$

E.g. *Tree* is subsumed by *Plant*. Metaproperties will be notated as boldface letters preceded by the tags -, + and $\sim$. A property ϕ exemplifying a metaproperty **M** is written $\phi^{\mathbf{M}}$. (Cf. [69])

3.1.2 Rigidity, Identity and Dependence

Rigidity

Definition 3.3 *A property ϕ is* essential *to a particular* a *iff ϕ necessarily always holds for* a *(Lowe [42]):*

$$\Box \forall t\, \phi(a,t)$$

E.g. *Photosynthesize* is an essential property of plants.

Definition 3.4 *A property ϕ is* rigid *iff it is necessarily essential to all the entities it holds of:*

$$\Box \forall xt\,(\phi(x,t) \rightarrow \Box \forall t'\,\phi(x,t'))$$

E.g. *Plant* is a rigid property, while *Living* (speaking - alas not only - of plants) is not.

Definition 3.5 *A property ϕ is* non-rigid *iff ϕ is not essential to some instance:*

$$\Diamond \exists x\,(\phi(x,t) \wedge \Diamond \exists t'\,\neg\phi(x,t'))$$

Definition 3.6 *A property ϕ is* anti-rigid *iff, for every instance* x *of ϕ, ϕ is not essential to* x*:*

$$\Box\,(\forall xt\,\phi(x,t) \rightarrow \Diamond \exists t'\,\neg\phi(x,t'))$$

E.g. *Having foliage* is anti-rigid with regard to broad-leaved trees, which lose their foliage either seasonally or as a symptom of some disease.

Rigid properties are tagged with **+R**, non-rigid ones with **-R** and anti-rigid ones with **∼R**. (Cf. [36])

Identity

We have seen that, in order to identify an individual at a certain instant t, we have to be able to attribute it a certain distinctive or identifying empirical property. As Guarino remarks, an identifying property for a particular-class ϕ can be viewed as a relation between instances of ϕ and certain identifying characteristics; e.g. a person's body can be identified via its fingerprints.

Characteristics can be parts, qualities, or other entities related to the particular in question in a unique way. As an individual cannot be its own characteristic, a characteristic relation is irreflexive. We adopt the following notations:

Definition 3.7 $\chi(x,z,t) =_{def}$ "*x has the characteristic z at time t*"

Definition 3.8 *The* sameness-formula *for a characteristic relation χ is*

$$\Sigma\chi(x,y,t,t') =_{def} \forall z\,(\chi(x,z,t) \leftrightarrow \chi(y,z,t'))$$

asserting that individuals share the same characteristics at (possibly) different moments.

We introduce the concept of signature in order to capture two constraints on identity conditions mentioned somehow disparately in [36]:

Definition 3.9 *A characteristic relation χ of a property ϕ is* signature *for ϕ iff the conjunction of the two following propositions holds:*

(1) $\Diamond \exists xy \neg(\Sigma\chi(x,y,t,t') \leftrightarrow x = y)$

(2) $\Box\,(\phi(x,t) \rightarrow \exists z\, \chi(x,z,t))$

(1) is the *non-triviality constraint* for χ; (2) guarantees that there are characteristics for each instance of ϕ.

The sameness-formula $\Sigma\chi$ of a signature can now be used as an identity condition (IC) for the instances of the respective property. A distinction has to be made between *local* and *global* ICs.

Local ICs hold only of two individuals if both are instances of a given property, while *global* ICs also hold of two particulars if only one is an instance of the property in question. An example of a local IC is the sameness of wing patterns for the non-rigid property *Butterfly*, which is not defined when one of the individuals to be compared instantiates the equally non-rigid property *Caterpillar*.

Definition 3.10 *For each property ϕ and signature χ of ϕ:*

1. ϕ carries a local identity condition $\Sigma\chi$ *iff, necessarily:*
   $E(x,t) \wedge E(y,t') \wedge \phi(x,t) \wedge \phi(y,t') \rightarrow (\Sigma\chi(x,y,t,t') \leftrightarrow x = y)$

2. ϕ carries a global identity condition $\Sigma\chi$ *iff, necessarily:*
   $E(x,t) \wedge E(y,t') \wedge \phi(x,t) \rightarrow (\Sigma\chi(x,y,t,t') \leftrightarrow x = y)$

$\Sigma\chi$ is a synchronic IC iff t=t' and a diachronic IC otherwise.

In order to be able to tackle inheritance of ICs through taxonomies, one has to differentiate between *carrying* and *supplying* an IC.

Definition 3.11 *A property ϕ is a* sortal *iff it carries a local or a global IC. (Strawson [60])*

Definition 3.12 *Let Φ be the set of properties represented in an ontology; a property ϕ* supplies *a local or global IC $\Sigma\chi$ relatively to Φ iff:*

1. *ϕ carries $\Sigma\chi$, and*

2. *not every property in Φ which directly subsumes ϕ carries $\Sigma\chi$.*

Definition 3.13 *A property ϕ is a* type *iff it supplies a global IC.*

Sortals are tagged with **+I**, non-sortals accordingly with **-I**. On the other hand, types are tagged with **+G**, non-types with **-G**. (Cf. [36])

Dependence

In [35, 69], Guarino and Welty adopt a concept of ontological dependence inspired by Simons [59]: a property ϕ is said to be externally dependent on a property ψ iff for every instance x of ϕ there is necessarily an instance of ψ that is neither a part nor a constituent of x. This notion involves the mereological relations of parthood and constituency which are not discussed in this thesis.

We prefer to incorporate the weaker property of identification-dependence proposed by Strawson [60] (cf. chap. 2), that in our opinion is much more convenient for ontological analyses:

Definition 3.14 *A particular x is* dependent *on a particular y iff there is a sortal σ with IC $\Sigma\chi$ such that:*

1. *σ is an essential property of x*
2. *$\chi(x, y, t)$, i.e. y is a characteristic of x under χ.*

E.g. a thought is dependent on the person it is a state of, insofar the IC of the sortal *Thought* involves the thinking person as a characteristic.

Definition 3.15 *A property ϕ is* dependent *on a property ψ iff each instance of ϕ is dependent on an instance of ψ.*

E.g. the property *Thought* is dependent on the property *Person*. Note that it is not necessary for ϕ or ψ to carry or provide themselves an IC, as the dependence may be mediated by a sortal not identical to ϕ or ψ. Properties are marked with **+D** if they are dependent, otherwise with **-D**.

As already reviewed in the previous chapter, according to Strawson [60] a property of particulars is basic iff it is not dependent on any other property of individuals. However, this condition is much too strong in our opinion. Indeed, it is doubtful if there are any properties that are not identification-dependent on others. That is why we prefer the following:

Definition 3.16 *Let Φ be the set of properties represented in an ontology; A subset Ψ of Φ is* basic *relatively to of Φ iff*

1. *each $\phi \in \Phi \setminus \Psi$ is dependent on some $\psi \in \Psi$*
2. *no $\psi \in \Psi$ is dependent on some $\phi \in \Phi \setminus \Psi$*

where $\Phi \setminus \Psi$ is the difference set of Φ and Ψ.

Definition 3.17 *A property is* basic *iff it is an element of a basic set of properties.*

So if a basic property happens to be dependent on some other property, the latter must be basic, too.

3.1.3 A Well-founded Ontology

Principles of Well-foundedness

Definition 3.18 *A set Φ of properties defined over a domain D is a* well-founded ontology *iff Φ satisfies the following four principles :*

Sortal Individuation Principle *Every element of D is necessarily an instance of some sortal in Φ ("No entity without identity" - [69] after Quine [53]).*

Sortal Expandability Principle *Two identical elements of D are necessarily instances of a sortal which carries the IC they meet ([36] after Lowe [42]).*

Unique Sortal Principle *If two elements in D satisfy an IC, there exists necessarily a unique sortal supplying it [36].*

Grounding Principle *All types in Φ are subsumed by types which are basic relatively to Φ (Strawson [60]).*

The concept of a well-founded ontology introduced extends that implicitly used in [36, 69], mainly by adding the Grounding Principle inspired by Strawson. An important consequence is the following theorem:

Theorem 3.1 *In a well-founded ontology, types are rigid. (Guarino and Welty [36])*

Proof of Theorem 3.1 Suppose a certain type ϕ with IC $\Sigma\chi$ is not rigid. Then there are two identical entities that meet the IC $\Sigma\chi$, such that ϕ does not necessarily always hold of both of them. But according to the Unique Sortal Principle there must be a unique sortal ψ supplying $\Sigma\chi$. By hypothesis, it is ϕ that supplies $\Sigma\chi$, thus $\psi = \phi$. Therefore, ϕ must be rigid, which leads to a contradiction.

From Theorem 3.1 and the Sortal Individuation Principle we conclude:

Theorem 3.2 *In a well-founded ontology, every element of the domain instantiates a type.*

Together with the Grounding Principle, Theorem 3.2 implies:

Theorem 3.3 *In a well-founded ontology, each element of the domain instantiates a basic type.*

Types are the backbone of an ontology. In order to supply global ICs, they must be rigid. In other words, types stand for the unchanging properties that specify the essence of an entity insofar they supply ICs to it. (Cf. [36])

Subsumption Constraints

Thus each property in an ontology can be formally characterised as to rigidity (**+R, -R, ∼R**), identity conditions (**+G, -G, +I, -I**) and dependence (**+D, -D**). In "tangled" subsumption hierarchies, this means a possible multiple inheritance of ICs. Now the *a priori* logical meta- properties of kinds constrain the extent as to how far ICs can be inherited. Guarino and Welty [36] state that IC inheritance is ruled by the following principle:

Identity Disjointness Constraint (IDC) Properties carrying incompatible identity conditions are necessarily disjoint. (after Lowe [42])

Indeed, carrying an IC is obviously an essential property and essential properties that are incompatible are necessarily disjoint. Thus it is often sufficient to inspect the essential properties related to kinds in order to decide whether a subsumption link is possible or not. The IDC entails the following special subsumption constraints:

$$\neg(\psi^{+R} \preceq \phi^{-R})$$
$$\neg(\psi^{+I} \preceq \phi^{-I})$$
$$\neg(\psi^{+D} \preceq \phi^{-D})$$

E.g. $Person^{+R,+I}$ cannot possibly subsume $Agent^{-R,-I}$. (Cf. [36])

A Formal Taxonomy of Properties

Finally, the array of meta-properties allows the classification of the kinds of being expressible in an ontology according to general categories. As Guarino points out, there are 24 possibilities of meta-property combinations, which reduce, due to subsumption and incompatibility considerations, to a set of eight categories. Despite the fact that our concept of dependence is weaker than that of Guarino, we may safely suppose that it is implied by his. Thus Guarino's classification (cf. Table 3.1 [69]) can be adopted as such. [1]

This classification can be translated into a taxonomic structure, which conforms to the principles stated by Bouaud a.o. [3] presented in the last chapter. At the root, a distinction is made between *sortals* and *non-sortals*, along the dimension **+/-I**. Riding the dichotomy are *roles*, marked with **-R** (non-rigidity) and **+D** (dependence); they subdivide in *formal roles* (**-I**) and *material roles* (**+I**). Non-sortals are categories, attributions or formal roles. Sortals subcategorise *rigid* (**+R**) and *non-rigid* (**-R**) sortals; the latter can be refined as being *anti-rigid* (**∼R**) (cf. Figure 3.1 [69]).

[1] Except a minor re-interpretation of the **+O**-tag as **+G**-tag required by [36].

G	I	R	D	Kind of Property	Sortal
+	+	+	+/−	Type	yes
−	+	+	+/−	Quasi-type	yes
−	+	∼	+	Material role	yes
−	+	∼	−	Phased sortal	yes
−	+	−	+/−	Mixin	yes
−	−	+	+/−	Category	no
−	−	∼	+	Formal role	no
−	−	∼	−	Attribution	no
−	−	−	+/−	"	"
+	+/−	∼/−		*undefined*	

Table 3.1: Combinations of meta-properties in a well-founded ontology (after Guarino and Welty [69])

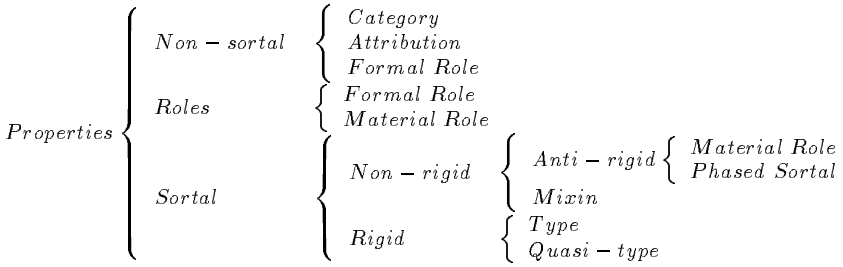

Figure 3.1: Taxonomy of properties in a well-founded ontology (after Guarino and Welty [69])

Categories are rigid properties not carrying any IC. As they cannot be subsumed by sortals, they constitute commonly the top-level distinctions in an ontology, used for a preliminary partitioning of the domain. Examples are *Entity*, *Particular*, *Universal*, etc.

Types are rigid sortals supplying global ICs. They represent the invariant distinctive features of elements in the domain and constitute therefore the core of an ontology. Types immediately subsumed by categories are also called top-types. Examples are *Book*, *Body*, *Person*, etc.

Quasi-Types are rigid sortals not supplying global ICs. They are often used to organise the domain elements already identified by types in

order to introduce refining distinctions. Examples are *State*, *Process*, *Event*, etc.

Formal Roles are anti-rigid and dependent properties carrying no IC. They are generally lambda-abstractions of relations between elements of the domain (e.g. $\lambda x\, agent(DiscoveringAmerica, x)$, often expressing parts played of entities in events. Examples are *Agent*, *Theme*, etc.

Material Roles are also anti-rigid and dependent, however they inherit ICs from some type. They express roles which have already been restricted to certain sortals, and can also associated with some concretely nameable event. E.g. *Spouse* is subsumed by *Person* and relates to a *marriage* event.

Phased Sortals are independent, anti-rigid and carry local ICs. They represent temporal phases of the entities they hold of, obviously individuals that undergo radical changes related to some of their (local) ICs in regular and discrete periods of time. In order that the respective entity still be conceivable as an identifiable object, these changes of local ICs should not affect the global ICs rendering the entity distinctive. Examples are *Butterfly* and *Caterpillar*.

Attributions are either non-rigid or anti-rigid and independent non-sortals. They normally express values of attributes or qualities (*color*, *polarity*). Examples are *Green*, *Plus*, etc.

Mixins are non-rigid sortals. Basically they represent disjunctions or conjunctions of rigid or non-rigid kinds. Examples are *Married or Happy*, *Person and Performer*, etc.

The *backbone taxonomy* of an ontology consists of the three classes of rigid properties in our classification: *categories* organise the universe of discourse into subdomains, *types* provide distinctive features for identifying domain elements and *quasi-types* provide more refined classifications of already identified entities. (Cf [35])

3.2 A Minimal Semantic Ontology

3.2.1 How to Guarino-Welty a Strawson-Parsons

As announced in the introduction, our main goal is to develop a minimal ontology, a set of concepts (types and roles) which can serve as a basis for specifying the semantics i.e. logical form of (assertive) agent messages. At the heart of the following argumentation is Terence Parsons' [47] account of the semantics of verbs and adjectives in terms of underlying states/events and parts played in them by other individuals. We will combine this account with Peter Strawson's [60] theory of basic particular-types.

This is not new *per se*, as e.g. Guarino [29, 30, 31] has proposed a minimal ontology of particulars based on Strawson's ideas. However, we think that there is still some analytical work left for us to do. Some distinctions drawn by philosophical authors have never found their way into AI ontologies. For example Strawson's insistence on the primitiveness of the concept of persons may seem rather exotic at first sight, but it will turn out to be of crucial importance to the analysis of thematic roles, especially the multiple subject-roles.

On the other hand, if we apply the methodology of Guarino/Welty to both Strawson's and Parsons' accounts, we cannot but notice a number of category mistakes to be redressed. The notion of dependence will be a criterion to re-assess Strawson's treatment of the relation between objects and events. In fact, a common assumption which Strawson has not questioned is that events are somehow parasitic on physical objects. But if one has a closer look at the conditions of identification for both particular types, one discovers a mutual identification-dependence corroborated by everyday language (as we shall see later on). Furthermore, the classification of properties summarised in Figure 3.1 will make us reconsider radically Parsons' list of fundamental thematic roles. Indeed, it will turn out that we have to do with a mixture of formal and material roles. Maybe some confusions will have been disentangled at the end of this chapter.

3.2.2 Substances and Occurrences

Occurrences: States and Events

The semantics of the lexical categories *Verb* and *Adjective* seems to imply the existence of "temporal" particular-types (Parsons [47], p. 4, pp. 186-187):

1. verbs represent sortals of *states* or *events*;

2. adjectives express sortals of *states*.

We choose to regroup states and events under the convenient label *occurrences*, which might be more natural than Parsons' *eventualities* ([47], p. 20) or Guarino's/Simons' *occurrents* ([31], [59]).

The idea that the semantics of verbs is to be analysed in terms of an implicit existential quantification over underlying occurrences has been championed in recent times by Donald Davidson [11]. According to this account, a sentence like: "Paul goes to London" would have the logical form:

(E) $\exists$e (Going(e) $\wedge$ Agent(e,Paul) $\wedge$ Motion(e,London))

i.e. the verb "goes" would not represent a relation involving Paul and London, but an event (action) of a certain type (*Going*), to which Paul and London would stand in certain relations (they are fillers of so-called "thematic roles", as we shall see later on).

Parsons ([47],pp. 14-19, 198-200) summarises the evidence in favour of underlying occurrences:

The Logic of Verb Modifiers Sentences involving verb modifiers seem to be logically interrelated because of these modifiers; e.g.

A Paul goes to London by train.

B Paul goes to London.

C Paul goes by train.

D Paul goes.

A implies B and C, while D is implied by the first three sentences. These logical dependences become explainable if we assume that each of them can be analyzed in the manner of E:

A' $\exists$e (Going(e) $\wedge$ Agent(e,Paul) $\wedge$ Motion(e,London) $\wedge$
$\exists$y (Train(y) $\wedge$ Instrument(e,y)))

B' $\exists$e (Going(e) $\wedge$ Agent(e,Paul) $\wedge$ Motion(e,London))

C' $\exists$e (Going(e) $\wedge$ Agent(e,Paul) $\wedge$
$\exists$y (Train(y) $\wedge$ Instrument(e,y)))

D' $\exists$e (Going(e) $\wedge$ Agent(e,Paul))

The Logic of Perceptual Idioms Consider the two following sentences:

F Mary believes Paul goes to London.

G Mary sees Paul going to London.

F can be interpreted as expressing the fact that Mary believes the *proposition* that Paul goes to London:

F' believes(Mary,goes(Paul,London))

However, a similar reading seems to be excluded for G; this becomes clear if we consider the following possible context: Mary does not know that the man is Paul and that the place he is going to is London. In this context, G just implies that Mary sees the *event* of someone - whom we call Paul - going to a certain place - which we call London. G is typical of a whole range of similar sentences involving a verb of perception and an infinitive phrase.

Implicit and Explicit Speech about Occurrences Quantification over underlying occurrences can be used to explain why sentences that contain explicit references to occurrences can have the same meaning as sentences that do not, e.g.:

H After their singing of the Te Deum, they went home.

I After they had sung the Te Deum, they went home.

Another example for states:

J Paul's being bored made him go to London.

K That Paul was bored made him go to London.

Explicit Quantification over Occurrences Closely related to that is the fact that sentences with explicit quantification over occurrences may be premisses of sentences without, e.g.:

J Each of my listenings of that record makes me happy.

K I listen to that record.

L I am happy.

L is obviously entailed by K and L. Again it would seem that the best explanation for this phenomenon is the implicit quantification over occurrences in sentences like K and L. Here is another example for states:

M In every wretchedness, you are brave.

N You are wretched.

O You are brave.

Substances and Occurrences as Basic Particular-Types

Traditionally, there are two types of particulars which qualify for being basic:

- ◇ spatially extended objects, sometimes also called "bodies"; we refer to them neutrally as *substances* following the scholasticist tradition;
- ◇ states and events, which we group together as *occurrences*.

Strawson [60] adopts the commonly held view that substances or bodies are basic, while states and events are not, but dependent on substances with regard to their identification.

His argumentation is as follows: identification ultimately relies on the spatio- temporal framework as a system of reference. I.e. in order to identify particulars they are directly or indirectly located in space and time. Now, the particulars that represent the nodes of that system of spatial and temporal relations must themselves be spatial and temporal, extensive in space and persistent in time. Furthermore, these particulars must be accessible to human sensory perception; as the latter is, as we have seen, limited, these objects must also have sufficient richness and stability of features to allow for reidentification based on qualitative recurrences.

Strawson argues that the only objects which could make up such a framework of reference are material bodies or particulars possessing material bodies. Thus it would seem that material objects are basic particulars with regard to our conceptual equipment ([60],p. 39). Events and states seem to be dependent on the category of material bodies or entities possessing material bodies ([60],pp. 45-46). The main reason for this is that while material bodies are rich and diverse enough in spatial and in temporal features, events, being sufficiently fine-grained in time, are by definition relatively poorly featured in space. Thus they cannot form a homogeneous system of reference on their own; more sophisticated talk about events must rest on the framework provided by material bodies ([60], pp. 53-54).

We agree with Strawson as far as occurrences ultimately rely on substances as their spatial characteristics. But is it true that substances are completely autonomous with respect to their spatial and temporal placement ? Indeed, we get a rather different picture, if we have a closer look at everyday language usage:

1. The book is on the shelf.
2. We married in St. Nicholas'.
3. When I was ill, I read a lot.

4. Four years ago, that was very fashionable.

5. Before asking questions, read the manual.

6. A Renaissance painter.

7. A 13th-century manuscript.

Examples 1 and 2 show how substances (1) and occurrences (2) take substances as their spatial location. (3), (4) and (5) demonstrate how occurrences can be related to other occurrences as their temporal location. The last two examples are the surprising ones: when we want to locate substances temporally, we actually refer to events or groups of events as their respective temporal characteristic, and not to substances.

Ordinary language usage shows that in order to locate something spatially, we look for a substance as its spatial characteristic. But in order to locate something temporally, we refer to occurrences, states and events, as temporal characteristics. The fundamental framework of reference, the grid of spatio-temporal dimensions, is not held together by substances alone. They seem to be, contrary to what Strawson affirms, relatively poor in temporal features. Thus, as much as there are spatial and temporal dimensions and relations, there are two types of particulars that sustain our framework of reference: substances as characteristics in space and occurrences as characteristics in time.

Thus it is a *a priori* feature of our conceptual equipment that *both* substances and occurrences are basic particular-types: apart from their mutual dependence, neither substances nor occurrences are dependent on other types of particulars, while all other types can be assumed to be ultimately dependent on (but not necessarily reducible to) them.

And we get rid of some parasitic entities in the same go. Neither Strawson [60] nor Guarino [31, 30, 29, 28] seem to see any problems with spatial and temporal locations as separate particulars. But is there a reason for adding special placeholders in spatio-temporal links, when we already have found the actual bearers of space-time-relations in substances and occurrences themselves ? Following Ockham's wise motto "Entia non sunt multiplicanda sine necessitate", we would be well advised to reject them.

One could argue that sentences like the following point towards the existence of mere places and times:

1. You might get insane *at Imperial College.*

2. The Caravaggio is *in the next room.*

3. *In 1969* we finally landed on the Moon.

4. I will finish my thesis *in Autumn*.

However, an alternative reading of these sentences would interpret "Imperial College", "next room" or "the Moon" as (sets of) substances, while "1969" and "Autumn"or "illness" could be seen as referring to (sets of) occurrences. We should not forget that we measure space by comparing physical objects and time by tracked states or events like the daily rotation of the earth, the phases of the moon, etc. Basically, places and times can be regarded as abstractions from concretely existing substances and occurrences.

3.2.3 Persons

Mental occurrences and the concept of a person

Another fundamental feature of our conceptual equipment is the distinction between mental or private events/states on the one hand, and material or public occurrences that may become objects of mental events or states on the other hand. Ultimately, this distinction is based on that between the experiencer and the world. Indeed, mental occurrences are identification-dependent on persons. This seems to indicate that the concept of an experiencing, thinking and acting person has a special status inside of our conceptual scheme ([60], p.87). Or to put it in other terms: Our conceptual equipment is such that it posits the existence of two sub-types of substances, namely persons vs. non-persons or (mere) bodies.

P-predicates and how to ascribe them

There are two sorts of predicates which we ascribe to ourselves as persons:

M-predicates are such as to be ascribable to (mere) bodies: physical features like shape, weight, colouring, spatial and temporal position.

P-predicates are the category to which belong sensations and feelings, actions and intentions, perceptions and memories, thoughts and judgements, skills and competences, etc. Not every P-predicate consists in a mere mental occurrence, but it can be safely asserted that each of them implies, in one way or another, the existence of a mental occurrence on the side of the particular it is ascribed to.

The two questions which come immediately to mind are the following ([60], p.90 and pp.104-105):

a) Why are P-predicates ascribed at all ?

b) Why are P-predicates ascribed to the same thing as M-predicates ?

Now nobody would see a problem in assigning M-predicates to ones body. But should P-predicates then be ascribed to our body, too ? It seems obvious that there is an intimate connection between a person and her body. Indeed, for each person there exists strictly one body that has a subtle and varied causal relation to the different sorts of perceptions this person is likely to have. However, this unique association between a person and her body does not explain why P-predicates should be ascribed to any subject at all and why they should be ascribed to the same object M-predicates are attributed to, namely the persons physical body. All the mentioned facts do not give any reason why the concept of a person should belong at all to our conceptual scheme. ([60], pp.92-94)

Persons as a basic particular-type

Formal semantics is based on the fact that, if there are conditions at all for ascribing a certain predicate to an element in the domain over which the predicate is defined, then these conditions must be applicable indiscriminately to every particular in that domain. Thus one cannot attribute P-predicates to oneself unless one is able to ascribe them to other individuals as well. Otherwise one could ascribe P-predicates only to oneself, i.e. to nobody at all, as there would be no other potential bearer of mental predicates to differentiate from. This implies that one must be able to identify other individuals as "owners" of mental or "private" occurrences. ([60], p.100).

Unfortunately, taking the idea of a pure subject of experience, a mere consciousness, as logically primitive, leads to unsolvable problems. Such mere minds could only be singled out with the help of mental i.e. private occurrences. But by going by mental or private occurrences alone there would be no question of ascribing them to anything else than oneself, i.e., as we have already seen, to no particular at all. For the same reason, considering persons as consisting of two different subjects of predication, one for M-predicates, i.e. a body, on the one hand, and one for P-predicates, i.e. a consciousness, on the other, would be also problematic. Pure consciousness can only be treated as secondary, non-primitive concept, and the particulars falling under that header are obviously identification-dependent on persons ([60],pp.102-103).

Thus it seems necessary to treat the concept of person as primitive, where person is to be understood in the sense of a particular which both M- and P-predicates can be attributed to. The two questions stated at the beginning

of this section are actually linked together in the sense that P-predicates can only be attributed at all if there is a type of particulars which both P- and M-predicates can be ascribed to. And the answer to both questions is that persons are basic particulars in the conceptual scheme we actually use. And a person is neither an animated body nor an embodied anima, but the unanalysable subject of both M- and P-predication ([60], pp.101-103).

The type *Person* is also prior to the type *Body*, in the sense that the latter is *definable* as the negate of the former. This does not exclude that both types are basic inasmuch as they both are kinds of substances. But it does seem that the philosopher Martin Buber [5] has been right in arguing that the concept of (interrelating) persons underlies and permeates the way how we experience and conceive the world.

The dichotomy of persons and (mere) bodies cannot be reduced to the presence or absence of a particular component (e.g. a "spirit") within the individual in question, as Strawson has made clear. Following Buber, we could say that what distinguishes persons from bodies is mutuality, a dialogical way of interrelating ([5],p. 65).

We can summarize all we have learnt about basic particular-types in Figure 3.2; note that all properties that are minimally required for a working semantic ontology are written as lambda-abstractions:

$$Particular-Types \begin{cases} \lambda x\, Substance(x) & \begin{cases} \lambda x\, Person(x) \\ \lambda x\, Body(x) = \lambda x \neg Person(x) \end{cases} \\ \lambda x\, Occurrence(x) & \begin{cases} \lambda e\, State(e) \\ \lambda e\, Event(e) \end{cases} \begin{cases} \lambda e\, Private(e) \\ \lambda e\, Public(e) \end{cases} \end{cases}$$

Figure 3.2: Basic Particular-Types

3.2.4 Roles

The origin: Tesnière's Dependency Grammar

The idea that the meaning of natural language sentences might be analysed in terms of underlying events has been around in linguistics long before it made its way into philosophy and logic. It is actually the driving intuition behind Lucien Tesnière's Dependency Grammar [64, 65], which has been the first theoretical account of syntactic structures in modern linguistics. According

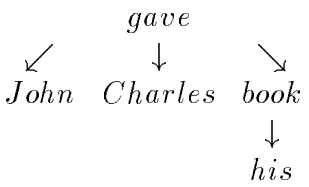

Figure 3.3: A simple English sentence after Tesnière

to Tesnière, a typical English sentence can be viewed as being "built around" its main verb, which acts as the root of a dependence tree of words. E.g. the phrase "John gave Charles his book" can be analysed as shown in Figure 3.3.

The dependency structure can be interpreted as representing an ontological structure involving an occurrence (expressed by the verb) and one or several entities relating in various ways to that occurrence. Consequently, Tesnière classifies the subtrees below the verbal root into two overall semantic/syntactic categories ([65], chap. 48):

Complements ("actants") , subject and object(s) in traditional grammar: entities actually participating in the occurrence;

Supplements ("circumstants") : adverbials in traditional grammar: circumstances of the occurrence (e.g. "to London", "on Monday").

Complements can be subdivided again according to the relation their "referent" bears to the event or state denoted by the verb ([65], chap. 51):

First Complement , traditionally the "subject" (of active phrases): denotes the origin of the occurrence;

Second Complement , traditionally the "direct object" (of active phrases): denotes the entity which the occurrence "happens to";

Third Complement , traditionally the "indirect object": denotes an entity which does participate in, but is not affected by the occurrence.

Verbs are characterised by the number and sorts of complements they "gouvern"; Tesnière calls this syntactic property of verbs "valence". It is the valence of the main verb which determines the core structure of the sentence which can be augmented optionally by one or several supplements. A verb can take up to three complements; including the zero-complement case, that

Verb Class	Complements	Instance	Example sentence
avalent	-	"rain"	"It rains in Britain"
monovalent	1.	"smell"	"The rose smells lovely"
copula + adj.	1.	"be red"	"The rose is red"
bivalent	1., 2.	"see"	"Mary sees the beautiful rose"
trivalent	1., 2., 3.	"give"	"John gives Mary a rose"

Table 3.2: Verb Classes after Tesnière

gives us four possible types: avalent, monovalent, bivalent and trivalent verbs ([65], chap. 97 ff).

Traditional grammars include a further verb-dependent grammatical category apart from subject, object and adverbial, namely the *(predicative) complement*, which we will simply call "predicative", e.g.:

1. The house is *old.*

2. John is *a bright scholar.*

3. I name this ship "the HMS Endemic".

4. Jane thought him *very rude.*

The italicised phrases of examples 1 and 2 are regarded to add information concerning the *subject* (*the house, John*), while those of examples 3 and 4 are seen as contributing somehow to the meaning of an *object* (*this ship, him*). That is why the former are called "subject-predicatives" and the latter "object-predicatives".

Tesnière does not consider predicatives as genuine complements of the verb, but treats them as being a part of the verb ([65], chap. 66-68). This is a somewhat contentious analysis. Regarding subject-predicatives, we retain the idea of treating the "copula-cum-adjective" (example 1) as a special case of a monovalent verb; the verb "to be" in the construction "to be + NP" (c.f. example 2) will be analysed as a bivalent full verb, which takes the predicative NP as its second complement. But we will treat this problem in more detail in chapter 4, where we will present sample formalisations of verb meanings. As to object-predicatives, we will see in the next section that they require a semantical analysis that goes beyond the mere subcategorisation of the verb.

Parsons' account of Thematic Roles

Tesnière's intuition has been adopted in Fillmore's Case Grammar [17] that introduced the concept of "thematic roles". A thorough theoretical account of thematic roles and the semantics of verbs (and adjectives) in terms of underlying occurrences has finally been proposed by Terence Parsons [47].

Different component NPs of a sentence can be assigned a "thematic role" (or "θ-role") consistent with the part their "referents" play in the occurrences expressed by the verb ([47], pp. 72-73), as shown in Table 3.3 ([47], pp. 73-76, with splitting of the *Instrument*-role as suggested at pp. 77-78).

θ-Role	Meaning	Category (Verb)	Example sentence
Agent	Person causing the event	Subject (active) by + NP (passive)	*John* writes a book. The book is signed *by John*.
Theme	Entity affected by the event ; Entity in the state	Direct Object (bivalent) Subject (monovalent) Subject (copula+adj.)	Mary reads a *book*. *Mary* blushed at his sight. *Mary* is bashful.
Goal	Addressee,	Indirect Object, to + NP (trivalent)	John gives *Mary* a rose. Anna writes a letter *to Mary*.
Benefactive	Entity to whose benefit the event occurs	Indirect Object, for + NP (trivalent)	Mary gave *Anne* a party. John signs a book *for Mary*.
Experiencer	Person the event is an experience of	Subject (active)	*Mary* sees a rose. *John* thinks about Mary.
Instrument	Thing the event is accomplished with	with + NP	John opens the letter *with a knife*.
Performer	Thing causing the event	Subject (active)	*The knife* opened the letter.

Table 3.3: Thematic Roles after Parsons

As already emphasised by Tesnière, each verb subcategorises a specific list of thematic roles. This list varies from verb to verb, but there seems to be a universal rule for subcategorisation, namely:

Each verb takes a Theme. ([47], p. 80)

Indeed, as an immediate consequence, the grammatical subject of sentences gouverned by monovalent (i.e. intransitive) verbs is a Theme, as in:

Mary blushed at his compliment.

Thus, the grammatical subject of monovalent verbs can take on several roles:

Mary arrived late. — *Agent-Theme*

The statue stands at the middle of the square. — *Performer-Theme*

John slept until afternoon. — *Experiencer-Theme*

In general, Parsons' account provides for the possibility of NPs with multiple thematic roles ([47], pp. 80-82).

The requirement that each verb should take at least a Theme creates a problem concerning avalent or "impersonal" verbs like *to rain*. One way to

deal with it is to acknowledge avalent verbs as an exception from the general rule. But we could also assume that avalent verbs take indeed an implicit Theme, namely the context of utterance or simply the world. E.g. "it rained yesterday" could be read as stating about the world as Theme the fact that it rained a day ago. However, we are not prejudiced in favour of any of these alternatives.

In Parsons' statement of the theory of θ-roles, Tesnière's Third Complement is split into the *Goal* and the *Benefactive*. *Goal* in the sense of the Addressee is not to be confused with the adverbials *Direction* ("Jack went *to London.*") nor *Objective* ("Anne smiled *to calm the baby*").

A peculiarity is the inclusion of the adverbial *Instrument* in a list of complements. This seems to be rather incoherent and can only be explained by the fact that this label is commonly also applied to "impersonal" origins of events or implicit instruments. In fact, Parsons proposes to regroup the latter under another role which he calls "Performer", with the obvious danger of confusing it with the *Agent*-role ([47], pp. 77-78).

The main problem however, is the fact that there are three subject-related roles: *Agent*, *Experiencer* and *Performer*. It is apparent that this trichotomy does not pertain to the mere relation of an entity to an event or state. Rather we have here additionally a blend of two oppositions:

1. *Person vs. Non-Person*, i.e. *Agent* and *Experiencer* vs. *Performer*,
2. *Public vs Private Occurrence*, i.e. *Agent* and *Performer* vs. *Experiencer*.

While the first opposition should be clear, it might be worthwhile to add some words concerning the second distinction, that between public (material) and private (mental) occurrences. Indeed, traditionally the *Experiencer* is associated with the grammatical subject of verbs expressing mental events or states (*to know, to see, to feel, to sleep*). While *Experiencer* seems to relate to the bearer of a private occurrence, *Agent* can be safely regarded as the person initiating (not necessarily always intentionally) a public state or event. However, one must always keep in mind that "public" and "private" are defined *in relation to the subject in question.* "Private" particulars are, as far as *external* observers of the subject's behaviour are concerned, "public" in the sense of being ascribable through behavioural criteria by persons other than their 'owner". So "to think" refers to a private occurrence (from the perspective of the subject), while "to encourage" expresses a public event, even though it is directed towards another person's state of mind.

Like Tesnière, Parsons does not include predicatives in this account of thematic roles. Relating to object-predicates, Parsons seems to consider at

least the greater part of them as indicating a state or event resulting from the event expressed by the main verb (e.g. "Robert painted the door green" would have to be read as "The state of the door being green is caused by the event of John painting it"). However, the theory of so-called "causatives" and "inchoatives" proposed by Parsons ([47], chap. 6) as a framework for the explanation of such linguistic phenomena is beyond the scope of the present thesis.

A New Theory of Thematic Roles

All these observations and distinctions lead us to an alternative theory of thematic roles that clarifies the classical account on the basis Guarino's / Welty's ontological classification of properties and refines it using the particular-types reviewed in section 3.1.

Indeed, after the short review of Parsons' detail of θ-roles, one cannot but agree with Guarino's and Welty's opinion that a rigorous specification of thematic roles is still to be delivered [35]. We follow their advice to define thematic relations as formal roles structuring role taxonomies and to avoid subsuming to them properties of other classes (such as material roles or mixins).

We distinguish between complement roles and supplement roles, the former being mandatorily subcategorised by the verb, the latter optionally modifying it. The complement roles correspond to Tesnière's first, second and third complements respectively, with the differentiation pertaining to the indirect object adopted from Parsons (cf. Table 3.4). To avoid confusion with the adverbials *Direction* and *Objective*, *Goal* is renamed *Addressee*. *Instrument* is relegated to the category of supplemental roles, which corresponds to Parsons' adverbials and Tesnière's supplements.

Regarding supplement roles, we have not much to contribute. Each grammar contains another classification of adverbials; Table 3.5 presents, as one of many alternatives, our typology that is partly inspired by Parsons ([47], pp. 269-270). While *Instrumental*, *Manner*, *Locative* and *Motion* correspond generally to substances constraining the occurrence expressed by the verb, *Temporal*, *Purpose*, *Reason/Cause*, *Consequence* and *Concession* denote commonly other states or events as circumstances of the occurrence. The leftover cases *Agent*, *Experiencer* and *Performer* can be defined using our four elementary complement rules and the basic particular-types.

First, we introduce two material roles, *Performer* and *Cause* that are the *Origin* restricted to the types *Person* and *Body* respectively. *Performer* and *Cause* are anti-rigid and dependent, but inherit ICs from *Person* or *Body*. In lambda-notation, their definitions are as follows:

Complements	Meaning	Category (Verb)	Example sentence
Origin	Entity causing the event	Subject (active)	*John* writes a book. *A stone* hits the window.
		by + NP (passive)	The book is signed *by John*. The window was hit *by a stone*.
Theme	Entity affected by the event ; Entity in the state	Direct Object (bivalent)	Mary reads a *book*.
		Subject (monovalent)	*Mary* blushed at his sight.
		Subject (copula+adj.)	*Mary* is bashful.
Addressee	Entity the event is directed to	Indirect Object, to + NP (trivalent)	John gives a rose *to Mary*. Mary gives water *to her flowers*.
Benefactive	Entity to whose benefit the event occurs	Indirect Object, for + NP (trivalent)	Mary gave *Anne* a party. John signs a book *for Mary*.

Table 3.4: Complement Roles

Supplements	Example sentence
Instrumental	John writes his letters *with a pen*.
Manner	Mary arranges the flowers *with great care*.
Locative	The apple tree is *in the garden*.
Motion	The carriage rattled *towards South Kensington*.
Temporal	It rained *while we strolled through Hyde Park*.
Purpose	George runs *to catch the train*.
Reason/Cause	*Because Anne is intelligent*, she will succeed.
Consequence	Mary is so happy *that she will not mind*.
Concession	George is relaxed *despite his being late*.

Table 3.5: Supplement Roles (Adverbials) after Parsons

Definition 3.19 $\lambda x\, Performer(o,x) =_{def} \lambda x\, Origin(o,x) \leftarrow Person(x)$

Definition 3.20 $\lambda x\, Cause(o,x) =_{def} \lambda x\, Origin(o,x) \leftarrow Body(x)$

where o is an occurrence parameter. Indeed, *Performer* is preferably to be used for persons, while *Cause* is a convenient term for a non-personal origin of an occurrence. Note that *Performer* does not imply any intentionality.

Second, *Agent* and *Experiencer* can be derived from the *Performer*-role, by specifying the occurrence as public or private relatively to the referent of the verb's subject. Being a special cases of the *Performer*-role both are appropriately classified as material roles.

Definition 3.21 $\lambda x\, Agent(o,x) =_{def} \lambda x\, Performer(o,x) \leftarrow Public(o)$

Definition 3.22 $\lambda x\, Experiencer(o,x) =_{def} \lambda x\, Performer(o,x) \leftarrow Private(o)$

Definitions 3.19 - 3.22 emphasise again the relevance of seemingly speculative distinctions such as *Person vs. Body* and *Private vs. Public* for natural language understanding and processing.

3.2.5 Putting it all together

The whole (in a) picture

Finally, we can summarise our reflections on a minimal semantic ontology in Figure 3.4. Our taxonomy involves just types and roles; types relate to what we think to be basic particulars, while roles are constrained to thematic roles in the widest sense.

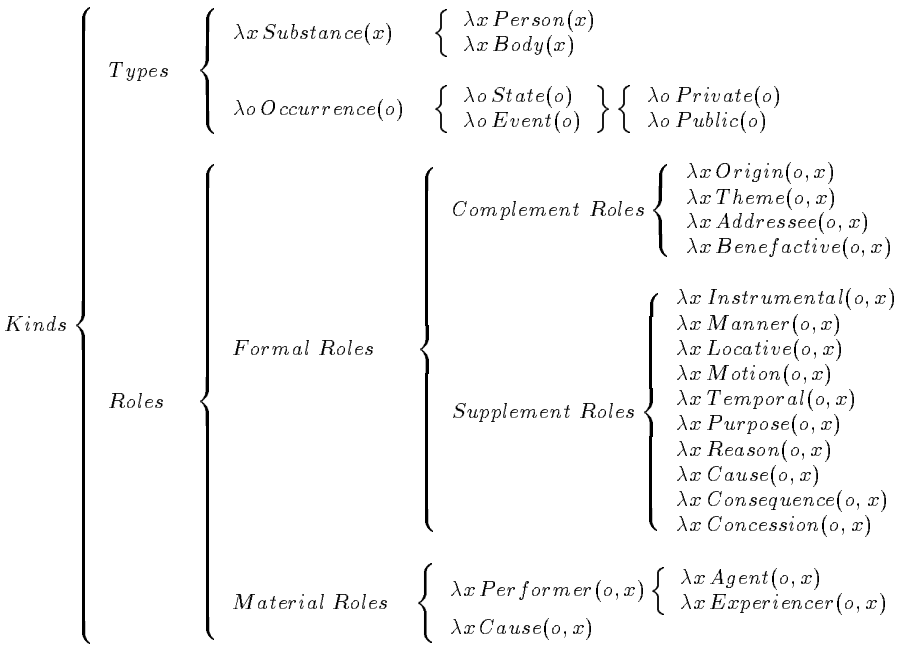

Figure 3.4: A Minimal Semantic Ontology

And that's it. How this taxonomy will be applied in the formalisation of verbs and their representation in a lexicon are questions better to be addressed in the next chapter, where we will present a practical example.

Chapter 4

A Proof of Concept

4.1 Outline of the System

4.1.1 Objective of the Implementation

The discussion of the previous chapters is to be corroborated by the implementation of a reasoning agent as a server capable of processing natural language queries from multiple human operated clients. This is the most basic form of a multi-agent system illustrating our ideas.

The main goal of our implementation is a proof of concept pertaining to the connection between ontology and semantics. Our aim is to demonstrate how the fundamental ontological distinctions presented in Figure 3.4 can be a basis for multi-agent semantics.

Semantics is concerned with the logical form of (declarative) agent messages, while pragmatics describes the communication primitives for agent interaction. A formal ontology of particulars offers the conceptual elements necessary for the analysis of logical form. In the context of a multi-agent system semantics relates to a shared understanding: meaning must be intersubjective, i.e. the declarative content of a message, its proposition, must be the same under each agent's interpretation. This is only possible if every agent analyses the proposition, details its logical form, on the basis of the same set of fundamental concepts. Thus a shared ontology is the precondition of a shared, i.e. a multi-agent semantics.

Hence a proof of concept is required to illustrate how the members of a multi-agent system achieve a common interpretation of propositions based on a shared set of categories, i.e. a shared ontology. Now, the minimal ontology developed in the last chapter largely rests on intuitions originating in natural language research. Indeed, these intuitions pertain to the conceptual framework of human agents. Thus the idea to integrate the aspect of the

interaction between human and non-human agents into the example implementation suggests itself immediately, insofar as it could directly be shown to rely on an ontology shared by the members of a multi-agent system involving human and non-human agents. Hence, the interrelation between a minimal ontology and a multi-agent semantics will be illustrated with respect to natural language based communication, that rests on the fact that non-human agents are programmed in a way to share the human agents' fundamental conceptual distinctions.

Moreover, understanding propositions (message contents) not only involves parsing, but also the ability to act on the information they contain. The fundamental way in which propositional content may become the trigger and the object of agent behaviour is reasoning or symbolic computation. Reasoning is ultimately the derivation or refutation of assertions on the basis of a set of beliefs categorised with the help of a (shared) ontology.

Putting all these ideas together, we can justify the implementation of a simple reasoning software agent capable of interacting with multiple human agents, understanding and assessing the truth value of their assertions, because it shares their semantico-ontological framework of concepts.

4.1.2 Structure of the Implementation

Concretely, the reasoning agent should be able to engage humans in a game of challenges and answers: opponents send natural language assertions to be parsed, proved or disproved, the agent justifying its answers by indicating the respective logical form (meaning) or proof established on the basis of a semantics/ontology shared with the human opponent.

The optimal way to realise such a system is to implement it as a client-server architecture, the server being the reasoning agent and the client(s) operated by the human opponents. The server should spawn off a new thread for each client, thus allowing peer-to-peer communication; thus multi-agency is not merely implemented by pairing off a single program with a single human, but actually involves multi-threading and inter-thread communication.

All these functionalities are provided by a modular system consisting of six components, the code of which is to be found in the appendix:

Parser A logic grammar translating into action the semantical analysis of verbs and sentences in terms of underlying occurrences and occurrence-related roles; for the sake of clarity, we choose a fairly conventional definite clause grammar complying with the principle of a unification-based grammar [parser.ql] ;

Lexicon [lexicon.ql] ;

Prover A meta-interpreter applied by the reasoner to evaluate the logical form of natural language assertions against its knowledge base (see below); proofs are constructed as nested lists of premisses [prover.ql] ;

Knowledge Base : A database of facts and rules representing the beliefs of the reasoner about a certain history of events and states; the knowledge base involves a set of primary concepts and rules shared with a human user and can be regarded as the (Herbrand) domain in which the queries are to be evaluated [kb.ql] ;

Reasoner (Server and Client) : The client-server architecture integrating all previous components; it includes the actual reasoning agent as the server and the client operated by the user [reasoner_server.ql, reasoner_client.ql] ;

Input-Out Utilities [io.ql]

Reasoner server and client are implemented in Qu-Prolog 6.0, a concurrent and distributed extension of standard Prolog, that has been developed by Peter Robinson (Queensland) and Keith Clark (Imperial College), mainly for the purposes of symbolic computation and agent applications [7, 57]. Qu-Prolog's multi-threading and inter-thread communication facilities make it indeed an ideal environment for the implementation of multi-agent systems.

All other components are deliberately written in standard Prolog for the sake of portability. In principle, they should be able to run as stand-alone programs under most Prolog versions. That is why many elegant solutions provided by Qu-Prolog (of which we are very well aware) have not been taken into account for other issues than multi-threading.

4.2 Parser and Lexicon

4.2.1 Syntactic and semantic foundations

Formal semantics rests on the idea that meaning is compositional, i.e. that the meaning of each part of a natural language sentence (except those basic elements that belong to the lexicon) is composed of the meanings of its (syntactic) components.

The notion of meaning compositionality and the mathematical instruments to describe it have originated in the semantics of formalised languages initiated mainly by Alfred Tarski [62]. In logic formalisms the concept of meaning compositionality translates into that of truth functionality, the idea

that the truth conditions of a logical proposition are a function of its formal structure. This approach has been transferred in the domain of formal linguistics and philosophy of natural language by Richard Montague [44, 45] and Donald Davidson [13] respectively.

Compositionality of meaning as truth conditions is quite trivially implemented in Prolog in the form of a parser translating an English sentence into a Prolog clause as it decomposes this sentence into its constituents. The sentence's translation into Prolog can be easily evaluated using Prolog's built-in resolution mechanism.

As the formalisation of verbs and adjectives we propose (cf. next subsection) implies a certain complexity of analysis, it might be advisable not to use a too sophisticated, if more efficient, parser, as the intricacies of its operation are likely to obfuscate the already rather detailed semantic representation. Moreover, the subject of this thesis is not the technicalities of natural language parsing *per se*. That is why we stick to a conventional definite clause grammar integrating elements of a unification-based grammar, as outlined by Covington ([9], chap. 5).

A *definite clause grammar* (DCG) analyses a natural language sentence as a hierarchy of constituent phrases starting with the sentence at the top and ending with the "words" as the basic elements of the lexicon at the bottom. *Phrase structure (PS) rules* describe how syntactic units can be decomposed into their immediate constituents. Each syntactic unit to which a PS rule can be applied represents a syntactic *category* ([9], [8], chap. 9, [4], chap. 21). A *unification-based grammar* is a grammar that represents grammatical information (like *Case*, *Agreement*, etc.) as features and values (e.g. case : nominative) and assigns values to features through unification only ([9], p.111). In a unification-based DCG, each category is assigned a specific *feature structure*; in our grammar, we adopt Covington's notation:

feature_1:value_1..feature_2:value_2.. [...] ..feature_N:value_N

A specification of the features used in our implementation will be given in the presentation of our fragment of English at the end of this section.

The semantic representation using lambda abstractions is also inspired by Covington ([9], chap. 7), but has to be adapted to the meaning formalisation detailed in the next subsection. Lambda notation goes straight into Prolog: '$\lambda y \lambda x formula(x, y)$' is rendered as '`Y^X^formula(X,Y)`'. This representation is used in Prolog with the aim of extracting those variables from the body of the formula that are supposed to unify with other explicitly specified expressions. These ideas will become clearer by going through the following example from our implementation:

```
s1(Sem,force:Mood..pers:Person) -->
        np((X^Sco)^Sem,pers:Person),
        vp(X^Sco,force:Mood..pers:Person).
```

The sentence category `s1` has the features `force` (affirmative or negative mood) and `pers` (`\verb0,1—`), indicating whether its main verb is impersonal or not. Both features are inherited by the verb phrase, as they are needed to determine whether the main verb is negated or not, impersonal or not. The noun phrase inherits the `pers` feature only. As the parser operates top-down / left-right, the value of `pers` is instantiated by parsing the NP, depending on whether the NP is the dummy subject `it` of impersonal verbs or not. The value of `pers` for the VP is instantiated by unification with the respective value in the NP.

As the NP is parsed, `X` in `(X^Sco)^Sem` (the NP's meaning) becomes instantiated; accordingly, `X` in `X^Sco`, the VP's meaning unifies with it and becomes instantiated to the same value. As the VP is parsed, the component `Sco` of its meaning `X^Sco` is instantiated and the corresponding variable in the NP's meaning unifies with the latter, thus instantiating it to the value of `Sco`. At the end, both `X` and `Sco` are instantiated in `Sem`, the sentence's meaning.

4.2.2 Formalisation of verb and adjective meanings

Before sketching the fragment of English tractable by the parser, it may be worthwhile to show how the predicates in Figure 3.4 can be applied to detail the formal semantics of the different verb classes. As in our implementation, we ignore the subtleties of tense and aspect. We use the following attributes:

1. *SUBJ* : subject-related role (*Origin, Performer, Cause, Agent, Experiencer*),
2. *SENSE* : occurrence class (e.g. raining(o), dancing(o), etc.)

SENSE is to be specified in the knowledge base according to whether it is a class of public or private occurrences. Concerning SUBJ, we adopt the convention that it should always take the most specific value, as some verbs may explicitly require an *Origin* of a certain particular-type (e.g. verbs expressing mental occurrences must go with a *Performer*). The rules defining subsumption links between roles are also contained in the knowledge base.

Avalent verbs are (undogmatically) supposed to take as their implicit *Theme* the world that contains the event they describe. The simplest way to render this idea in logic is to let the logical constant *true* represent the world as the Great Fact:

Avalent verb meaning

$\lambda o\,(SENSE(o) \wedge Theme(o, true))$

Example *to rain*

$\lambda o\,(Raining(o) \wedge Theme(o, true))$

Monovalent verbs (including predicative adjectives) take as *Theme* their grammatical subject. The latter may take also take on SUBJ, so that this verb class exhibits the greatest variety in role combinations.

Monovalent verb meaning

$\lambda x\,\lambda o\,(SENSE(o) \wedge Theme(o, x))$

Example *to hesitate*

$\lambda x\,\lambda o\,(Hesitating(o) \wedge Experiencer(o, x) \wedge Theme(o, x))$

A special case of monovalent verbs is predicative adjectives (e.g.: "John *is wise*"), where the verb "to be" only acts as a dummy verb or copula. We will analyse adjectives in general as expressing states, of which the NPs they modify are *Themes*.

Adjective meaning

$\lambda x\,\lambda o\,(SENSE(o) \wedge Theme(o, x))$

Example *thoughtful*

$\lambda x\,\lambda o\,(Thoughtfulness(o) \wedge Theme(o, x))$

Bivalent verbs take an *Origin* and a *Theme*, both being different NPs.

Bivalent verb meaning

$\lambda y\,\lambda x\,\lambda o\,(SENSE(o) \wedge SUBJ(o, x) \wedge Theme(o, y))$

Example *to hit*

$\lambda y\,\lambda x\,\lambda o\,(Hitting(o) \wedge Origin(o, x) \wedge Theme(o, y))$

A special case of a bivalent verb is the verb "to be" (and maybe other verbs) before a predicative NP. It might strain the linguistic intuitions a little bit, but we propose to regard "to be" in this case as a full verb taking the "predicative" as a *Theme*. The underlying occurrence we assume to be the (reified) state of *being identical with something*, represented by the atom *true*. This could be put forward as an argument why the *Theme* of "to be" is in the nominative, like the *Origin*, and not in the accusative. But we acknowledge that our analysis is rather contentious.

$\lambda y\,\lambda x\,\lambda o\,(Being(true) \wedge Origin(true, x) \wedge Theme(true, y))$

Trivalent verbs take as an additional role either a *Benefactive* or a *Goal*.

Trivalent verb meaning

$\lambda z\, \lambda y\, \lambda x\, \lambda o\, (SENSE(o) \wedge SUBJ(o,x) \wedge Theme(o,y) \wedge Goal(o,z))$

$\lambda z\, \lambda y\, \lambda x\, \lambda o\, (SENSE(o) \wedge SUBJ(o,x) \wedge Theme(o,y) \wedge Benefactive(o,z))$

Example *to give*

$\lambda z\, \lambda y\, \lambda x\, \lambda o\, (Giving(o) \wedge Agent(o,x) \wedge Theme(o,y) \wedge Goal(o,z))$

4.2.3 A Fragment of English

We will now briefly characterise the different categories in the lexicon and the parser (grammar), highlighting, where necessary, those aspects that are crucial for the understanding of the code in the appendix. Note that we select a fragment of English just large enough to illustrate our ideas about verb meanings and semantic roles. For a proof of concept, this is all that is needed.

The Lexicon

The lexicon contains the means of expression necessary for couching queries to a knowledge base about a small set of temporally and causally related occurrences. It consists of 6 categories:

Verbs. The representation of verb meanings detailed above translates easily into Prolog. Syntactically, verbs have three features: `conj`, `pers` and `subcat`. `conj` determines whether the verb is finite (`fin`) or infinite (`infin`), `pers` defines it as personal or impersonal (`0,1`) and `subcat` has as value the respective number of complements (`0,1,2,3`) . The thematic roles stated explicitly in the lexicon are: `ag` (*Agent*), `exp` (*Experiencer*), `cause` (*Cause*), `th` (*Theme*), `addr` (*Addressee*) and `ben` (*Benefactive*); in our example, *Origin* and *Performer* are only represented in the knowledge base and not in the lexicon, as the lexical entries about verbs have to be as specific as possible with respect to the subject role. Note that for the sake of simplicity, we ignore the subtleties of tense and aspect; the only temporal form is the past tense. For the same reasons, agreement can be ignored, as there are only singular verb forms in our lexicon.

Adjectives Everything has been said about this category, except that it bears no grammatical features in our implementation.

Prepositions. These terms introduce adverbials and express relations between the occurrence denoted by the main verb and various entities, both substances and occurrences. We distinguish between nominal and sentential prepositions; their only feature `subcat` takes either the value `np` or `s1` accordingly. Prepositions must have access to the occurrence variable of the main verb and sentential prepositions additionally to the occurrence variable of the subordinate clause's verb. This means that the occurrence variable must remain unbound up to the level of the sentence, which has far-reaching consequences, as we shall see.

Determiners. The fragment contains universal, existential, negated existential ('no') quantifiers and the definite descriptor ('the'). The descriptor replaces all other means of direct reference, as e.g. anaphora, that are not contained in our fragment. The negative existential quantifier is mainly used to formalise the meaning of negative verb forms. As the occurrence variable of the main verb remains unbound until the very top of the parse tree, the conventional logical form of determiners has to be modified by adding an explicit reference to that variable, e.g.:

```
det((X^Res)^(X^O^Sco)^O^all:(X,Res,Sco)) --> [every].
```

where O is the occurrence variable of the main verb; O has to be specified for the scope as well as for the overall sentence, in order to secure unification. The eccentric notation `Det:(X,Restrictor,Scope)` merely helps us to get around much more awkward 'univ'-ing. Also note, by the way, that our fragment of English only recognises singular determiners, e.g. `every` instead of `all`. Determiners bear no syntactic features.

Nouns and Proper Names No changes are to be imposed on those categories, whose formalisation follows the convention:

```
n(X^sun(X)) --> [sun].
pn((marcel^Sco)^Sco) --> [marcel].
```

Nouns and proper names bear no grammatical features (in our implementation).

Additionally, both lexicon and grammar contain semantically 'empty' terms like the copula `is`, the auxiliary `did`, the 'dummy' proper name `it`, etc. .

At this stage, a very critical reader familiar with Qu-Prolog might object that we did not use its built-in solutions for lambda-notation and quantification. We are well aware of these advanced features, but for the sake of

portability decided not to make use of them. Furthermore, we believe that a formalisation in pure standard Prolog enhances the clarity of the demonstration that could be otherwise jeopardised by the idiosyncrasies of a special-use dialect.

The Grammar

The fragment that can be generated by the grammar in `parser.ql` only includes sentences with one main clause and possibly one or more subordinate clauses introduced by sentential prepositions or the sentential conjunction 'that' (for verbs taking a sentence as *Theme*). As the fully documented code is to be found in the appendix, we can restrict ourselves to some informal remarks concerning the different syntactic categories:

Sentences. In order for the occurrence variable of the main verb to remain accessible to sentential prepositions, it obviously has to remain unbound up to the top of the parse tree. Hence, we have to distinguish two sentence categories, `s` and `s1`, the latter containing an unbound occurrence variable, as in:

```
O^exists:(X,city(X),(beauty(O),th(O,X)))
O^(wise(O),th(O,joan))
```

while the top sentence category, `s`, is characterised by the fact that this occurrence variable is bound, as in

```
exists:(X,city(X),exists:(O,(beauty(O),th(O,X)),true))
exists:(O,(wise(O),th(O,joan)),true)
```

Note that this necessitates quantifying inside of the scope of a determiner. Furthermore, as the standard form of quantified clauses is `Det:(X,Restrictor,Scope)`, the dummy scope `true` has to be adjoined.

It is sentences of category `s` that are taken as *Themes* by verbs introducing an *oratio obliqua* or indirect speech. Being the uppermost category, they bear no features. Sentences `s1` introduce the features `pers` and `force`, specifying whether the main verb is personal or impersonal, affirmative or negative. Affirmative verbs imply an existential quantification of the occurrence variable, negative verbs a negated existential quantification.

Noun phrases. These consist either of a single proper name or a determiner followed by a noun phrase containing a noun preceded by one or more adjectives. NP's bear the sole feature `pers`, determining whether they are instances of the dummy subject `it` or not. Note also that before an adjective's meaning is combined with that of the noun it modifies, its state variable must be bound by an existential quantifier.

Verb phrases In order to account for the semantical phenomena related to verb phrases, we assume 4 categories, namely `vp`, `v2`, `v1` and `v0`.

1. `vp` is the highest verb-related category. It optionally merges `v2` with a prepositional phrase and bears the top most features `pers` and `force`. Note that the distinction between personal and impersonal verbs relates in our grammar to that between avalent and n-valent verbs.
2. `v2` introduces the subcategorisation feature `subcat`. It incorporates the direct objects of trivalent verbs.
3. `v1` is the level at which direct objects of bivalent verbs or indirect objects of trivalent verbs are integrated.
4. `v0`, finally, corresponds to the (possibly negated) full verb with its auxiliaries, or the predicative adjective preceded by the (negated) copula.

Prepositional Phrases. This category also bears a `subcat` feature taking values `np` (for NPs) or `s1` (for sentences `s1`) respectively. In order to avoid syntactic ambiguities, we restrict our fragment so as to include only sentences in which 'nominal' PrepPs precede 'sentential' PrepPs. Nominal PrepPs are placed together without any conjunction, while sentential PrepPs are joined by the conjunction `and`. Thus we can distinguish a new sentential adverbial modifying the main verb from a subordinate clause of the most recently parsed supplement. The most important thing about PrepPs is that they require the subordinate clause's occurrence variable bound according to its main verb's mood, making it sometimes necessary to quantify inside of the scope of an overall determiner.

These remarks should be enough to enable the reader to understand the code of the parser.

4.3 Knowledge Base and Prover

4.3.1 The Knowledge Base

The knowledge base is the reasoner's set of beliefs and incorporates the shared ontology as the taxonomy of top-level distinctions. We say 'shared' because we believe that these distinctions belong to the fundamental conceptual framework of the human agents who operate the clients that post queries to the reasoner. It is against this database of clauses that the human opponent's challenges in the form of sentences to parse, prove or disprove can be evaluated and their logical form and/or proof be constructed as a justification for the reasoning agent's (positive) answers. The components of the knowledge base are:

Rules for subject roles Only the basic θ-roles are represented directly in the knowledge base. *Performer, Cause, Agent* and *Experiencer* are defined in terms of these fundamental thematic roles on the one hand and the types *Person, Body, Public* and *Private (occurrence).* This is the heart of the semantic ontology shared by the human agent and the reasoning software agent.

Type declarations These rules and facts specify occurrence-classes and substances according to the distinctions *Public* vs. *Private* or *Person* vs. *Body* respectively.

Rules about relations between occurrences Indeed, not every preposition's meaning is represented directly in the knowledge base; the meanings of the prepositions `when, before, after` and `because` are derived:

```
when(O,O1) :- at(O,O1) ; at(O1,O).    % simultaneity

before(O,O1) :- after(O1,O).          % anteriority

after(O,O1) :- post(O,O1).            % posteriority
after(O,O1) :- post(O,O2), after(O2,O1).

reason(O,O1) :- why(O,O1).            % reason
reason(O,O1) :- why(O,O2), reason(O2,O1).
```

Eternal Occurrences are such as to be simultaneous to every occurrence registered in the knowledge base, which goes into Prolog as `at(O,_)`

where `O` is an occurrence variable. A special case is the *state of being identical with something* that is the meaning of "to be" as a full verb and is represented by the atom `true`. This state is such that each substance is its *Theme* and *Origin*:

```
being(true).
th(true,_).
or(true,X) :- th(true,X).
at(true,_).
```

Temporary Occurrences are occurrences that are not eternal. To be more precise: temporary occurrences are such as to be posterior and/or anterior to other occurrences. We suppose a strict ordering of states and events, expressed by the undefined predicate `post`. Clearly, there must be an occurrence `O` such that `post(O1,O)` is not contained in the knowledge base, i.e. an occurrence that is posterior to no other state or event.

The representation of facts about states and events in the knowledge base is partly inspired by Covington ([9], p.251). For example, the proposition that 'Joan wrote a letter with a pen in London' is formalised in this manner:

```
letter(billet).
pen(geha).
...
writing(o(24)).
or(o(24),joan).
th(o(24),billet).
instr(o(24),geha).
in(o(24),london).
```

Some care must be given to the representation of facts about occurrences involving propositions as *Themes*. Indeed, one has to choose the logical form the parser would assign to the sentence in question. Equally, queries must respect the exact wording of the sentence in indirect speech: equivalent formulations will not do. This is no weakness of our representation, but a general fact about opaque contexts, where substitutivity *salva veritate* of synonymous expressions is not given (Quine [51]). This should be fairly obvious for verbs expressing a state of mind concerning a proposition, a *propositional attititude*, like *to know, to believe, to think, to regret*, etc. For example it is possible that Anna knows that Joan likes Marcel, but does not know that the

musician likes the poet, because she is ignorant of Joan being a musician and Marcel a poet. Thus the fact that 'Anna knew that Joan liked Marcel' would have to go into Prolog like this:

```
knowing(o(22)).
or(o(22),anna).
th(o(22),
   exists:(A,(liking(A),exp(A,joan),th(A,marcel)),true)).
at(o(22),_).
```

4.3.2 The Prover

The prover (`prover.ql`) is a Prolog meta-interpreter capable of building up proof trees as nested lists of premisses. This is certainly not the most efficient solution, but has the advantage of using Prolog's symbolic manipulation capacities that make it an ideal platform for meta-programming (Bratko [4], p. 612).

A *meta-program* is a program whose data are other programs. A *meta-interpreter* is a meta-program for a certain language L implemented in L itself ([4], p. 612). A Prolog meta-interpreter evaluates a goal on the basis of a program, in our case: of a knowledge base. A basic meta-interpreter for pure Prolog would be ([4], p. 614):

```
prove(true).
prove((Goal1,Goal2)) :-
     prove(Goal1),
     prove(Goal2).
prove(Goal) :-
     clause(Goal,Body),
     prove(Body).
```

where `clause(Head,Body)` succeeds if there is a clause with head `Head` and body `Body`. Obviously, such a meta-interpreter does not offer more services than the Prolog interpreter. Added value comes from capabilities such as the one to construct proof trees ([4], p. 613).

There has been no need to invent anything new, as there are already plenty of meta-interpreters around (e.g. that in Bratko [4], p. 617). We decided to extend a version presented by Frank Kriwaczek in his Prolog Programming course at Imperial College, by adding some functionalities related to the interpretation of the four determiners available in our fragment of English. Most of these additions should be self-explanatory. We only remark that the descriptor `the` is interpreted as a quantifier of unique instantiation

('at most one'), according to the venerable analysis of Bertrand Russell [55] (see also [49], pp. 146 ff) that treats propositions like

ιx (king_of_france(x) $\wedge$ bald(x))

as existential assertions of the form

$\exists!$x (king_of_france(x) $\wedge$ bald(x)) ,

where $\exists!x$ means 'there is only one x'.

In order to display proof trees generated as nested lists by "our" prover, we need a utility to pretty-print such structures. Again, it is not necessary to reinvent the wheel, as such a program is described in Clocksin and Mellish ([8], p. 96). We adopt a similar utility presented by Frank Kriwaczek. It basically displays the premisses below the conclusion with an indentation. Some operation examples at the end of this chapter will illustrate what the result looks like.

4.4 The Reasoner: Server and Client

4.4.1 Multithreading and Communication

The client-server architecture of the reasoning agent is easily implemented in a language offering the facilities for agent programming. Qu-Prolog 6.0 supports the creation, naming, controlling and deletion of multiple Prolog computation threads that share the same code and dynamic database, but independently execute different goals. Moreover, Qu-Prolog provides for communication between threads of possibly different processes running on possibly distinct machines [57, 7].

We briefly describe the multi-threading predicates used in our examples (for further details please refer to the Qu-Prolog 6,0 Reference Guide [54]). `thread_set_symbol(Name)` causes the symbolic name of the current thread to be set to `Name`. `thread_fork_anonymous(Thread, Goal)` creates an unnamed thread with ID `Thread` (a unique integer automatically assigned to the thread at its creation) and with goal `Goal`. `thread_exit` causes the current thread to terminate.

Communication between Qu-Prolog threads can be implemented in various ways, the most common being based on the *Interprocess Agent Communication Model* (ICM) of McCabe [43], a specification for sending and receiving messages between symbolically named threads of likewise named processes. Processes prospectively using ICM communication have their name registered with a communication server (CS) that may or may not run on the same machine. In Qu-Prolog, a process is registered with the name `Name` by starting it with the `-A Name` switch. This symbolic name is used by other

registered processes or threads to send their messages to the process; it is also part of the sender address contained in messages sent by the process to other processes or threads. The communication server is basically a message router; it receives messages from registered processes and forwards them either to their final destination or to another communication server [57, 54].

Qu-Prolog's ICM support consists of two layers, of which we will only use the highest one. At this level, addresses have the general form

```
ThreadID:ProcessName@MachineName
```

where `ProcessName` or `MachineName` can be dropped in case the communicating threads are of the same process or running on the same machine [57].

The communication primitives used in our implementation are the following. `Msg ->> Address` sends message `Msg` to address `Address`, while `Msg <<- Address` reads message `Msg` with sender address `Address` from the incoming message queue; in a Prolog program, this clause blocks until `Msg` is received. `Msg <<= Address` is used to search the thread's incoming message queue for a message that unifies with `Msg` and `Address`. The structure `message_choice(Alternatives)`, where `Alternatives` stands for:

```
Term1 -> Goal1; Term2 -> Goal2; ... ; TermN -> GoalN
```

each `TermI` being commonly a message pattern of the form `Msg <<- Addr`, inspects every incoming message as to whether it fits one of these patterns. The first message unifying with the uppermost pattern (in the order of appearance) triggers the respective Goal. If no 'fitting' message is found, the call suspends until further messages are received [54].

4.4.2 Implementing a Simple Reasoning Agent

Our implementation of the reasoner server and client [`reasoner_server.ql`, `reasoner_client.ql`] is almost a transposition of Robinson's and Clark's example of a Linda server and client [57]. The originality of our approach lies in the integration of this client-server model with its specific client-triggered threading and connection handshake into a totally different context, namely that of a multi-agent system involving human and non-human participants based on natural language communciation.

We only present the rough outline of each component's code that can be consulted in the appendix of this thesis.

Reasoner Server

The top routine of the reasoner server (that has to be run as an executable) loads the programs for the parser, the lexicon and the knowledge base, names

the main thread and goes into a loop, waiting for client requests to process. Through multi-threading the reasoner can deal simultaneously with several human operated clients by spawning off multiple threads or agents communicating and interacting with the human users.

```
main(_) :-
        [parser,prover,lexicon,kb],
        thread_set_symbol(reasoner_server_thread),
        main_loop.

main_loop :-
        repeat,
        connect <<- RtAddr,
        thread_fork_anonymous(_, reasoner_thread(RtAddr)),
        fail.
```

In case a `connect` message from a client has been received, the server forks off a new anonymous thread setting its goal to a routine that finishes the connection handshake initiated by the client and goes into a loop processing the clients requests to parse or prove sentences or to disconnect.

```
reasoner_thread(RtAddr) :-
        connected ->> RtAddr,
        thread_loop(RtAddr).
```

Note that the server passes to its child thread the return address `RtAddr` of the client, which is used for all further interactions. The server thread's loop consists of a `message_choice` structure. The messages `parse(Sentence)` and `prove(Sentence)` are dealt with by executing the respective parser and prover routines and sending the result (logical form and/or proof as nested list of premisses) to the client. If the message `disconnect` is received from the client, the child thread terminates.

Reasoner Client Support

The reasoner client support puts four commands at the human operator's disposal: `reasoner_connect`, `reasoner_disconnect`, `reasoner_parse` and `reasoner_prove`.

`reasoner_connect` sends a connection request to the main thread of the reasoner server. As soon as a `connected` message from the forked server thread is received, its address is recorded so that the client's queries will go to the competent server thread. The thread ID of the client is stored too in order to allow several clients within the same process.

```
reasoner_connect :-
        connect ->> reasoner_server_thread:reasoner_server_process,
        connected <<= A,
        thread_tid(TID),
        assert(idaddr(TID,A)).
```

`reasoner_disconnect` retracts the stored server thread address from the client's dynamic database and sends a `disconnect` message to its peer, causing the latter to terminate.

```
reasoner_disconnect :-
        thread_tid(TID),
        retract(idaddr(TID,A)),
        disconnect ->> A.
```

`reasoner_parse` and `reasoner_prove` get an input sentence from the human user and send it to the forked server thread, waiting until the latter has replied with the logical form and/or proof in order to display them to the operator. If a message from the thread specifying a parse or proof failure is received, the user is notified of it.

4.4.3 How everything fits together

By spawning off a new thread or agent at each client's request, the reasoner server is at the heart of a multi-agent system of communicating human and non-human peers. The content of the humans' messages are declarative natural language sentences whose meaning, i.e. logical form, mirrors the everyday conceptual framework of intelligent primates.

Shared understanding is made possible by the fact that the software agents have the same ontology, i.e. set of fundamental conceptual distinctions, as their human partners. This ontology is the basis for the semantics of the natural language fragment used by the humans to communicate with their non-human peers. Sharing this ontology as part of their knowledge base, the software agents have the capability of parsing and proving, i.e. of understanding and reasoning upon the assertions submitted to them.

The logical form and proof computed by a reasoning agent of our system reflects the semantic and ontological intuitions of the human operators and can thus be used by them to get a clearer grasp of the subtleties pertaining to their underlying and often unconscious assumptions and notional differentiations. Indeed, as we shall see in the next section, with the help of software agents it is possible to unravel semantic structures of a considerable complexity behind even the simplest human utterances.

4.5 System Operation

4.5.1 Starting up

The file `reasoner_server.ql` has first to be compiled:

```
qc -o reasoner_server reasoner_server.ql
```

then the executable has to be launched with the `-A` switch in order to name it as `reasoner_server_process`:

```
reasoner_server -A reasoner_server_process
```

After that, Qu-Prolog is started also with a naming switch:

```
qp -A client_process
```

After the GUI of the Qu-Prolog process has been opened, the program for the reasoner client can be loaded:

```
Qu-Prolog Version 6.0

| ?- [reasoner_client].

yes
```

Finally we connect to the server:

```
| ?- reasoner_connect.

yes
```

4.5.2 Parsing

We start with some simple examples of parsing; the last one illustrates the behaviour of the system in the case of a parsing failure.

```
| ?- reasoner_parse.
> it rained in london

exists : (_36E , (((raining(_36E) , th(_36E, true)) , in(_36E, london)) , true))
yes

| ?- reasoner_parse.
> the sun shone in paris
```

```
the : (_3A4 , (sun(_3A4) , exists : (_3B5 , (((shining(_3B5) ,
(cause(_3B5, _3A4) , th(_3B5, _3A4))) , in(_3B5, paris)) , true))))
yes

| ?- reasoner_parse.
> marcel liked joan

exists : (_349 , ((liking(_349) , (exp(_349, marcel) , th(_349, joan))) ,
true))
yes

| ?- reasoner_parse.
> marcel liketh joan
ungrammatical
yes
```

4.5.3 Simple proofs

And we continue with some equally simple proofs. The last example shows the system's reaction to proof failures.

```
| ?- reasoner_prove.
> a musical woman was beautiful

exists : (_41A , ((woman(_41A) , exists : (_42F , ((musicality(_42F) ,
th(_42F, _41A)) , true))) , exists : (_448 , ((beauty(_448) , th(_448, _41A)) ,
true))))

     woman(joan)
     musicality(o(11))
     th(o(11), joan)
     beauty(o(13))
     th(o(13), joan)

yes

| ?- reasoner_prove.
> joan liked marcel

exists : (_34A , ((liking(_34A) , (exp(_34A, joan) , th(_34A, marcel))) , true))

     liking(o(20))
     exp(o(20), joan)
          perf(o(20), joan)
               or(o(20), joan)
               person(joan)
          private(o(20))
               liking(o(20))
     th(o(20), marcel)

yes

| ?- reasoner_prove.
> anna knew that joan liked marcel

exists : (_452 , ((knowing(_452) , (exp(_452, anna) , th(_452, exists : (_473 ,
((liking(_473) , (exp(_473, joan) , th(_473, marcel))) , true))))) , true))
```

```
     knowing(o(22))
     exp(o(22), anna)
          perf(o(22), anna)
               or(o(22), anna)
               person(anna)
          private(o(22))
               knowing(o(22))
     th(o(22), exists:(_4F6,((liking(_4F6) ,(exp(_4F6,joan),th(_4F6,marcel))),true)))

yes

| ?- reasoner_prove.
> joan was a tall beautiful musical woman
unprovable
yes
```

4.5.4 Advanced Proofs

Here are some more challenging examples of proofs. The whole point in including these outputs is merely to illustrate what the system's operation looks like and how far it can be pushed. In particular the last example is quite revealing of the considerable power of this still rather simple implementation. There is no need to understand all the details.

```
| ?- reasoner_prove.
> marcel gave joan a rose in a large city

exists : (_4CE , ((exists : (_4E0 , (rose(_4E0) , (giving(_4CE) , (ag(_4CE, marcel) ,
(addr(_4CE, joan) , th(_4CE, _4E0)))))) , exists : (_50C , ((city(_50C) ,
exists : (_521 , ((being_large(_521) , th(_521, _50C)) , true))) , in(_4CE, _50C)))) ,
true))

     rose(rosa)
     giving(o(32))
     ag(o(32), marcel)
          perf(o(32), marcel)
               or(o(32), marcel)
               person(marcel)
          public(o(32))
               giving(o(32))
     addr(o(32), joan)
     th(o(32), rosa)
     city(london)
     being_large(o(4))
     th(o(4), london)
     in(o(32), london)

yes

| ?- reasoner_prove.
> anna encouraged the beautiful woman so that the beautiful woman invited marcel

exists : (_77A , ((the : (_78C , ((woman(_78C) , exists : (_7A1 , ((beauty(_7A1) ,
th(_7A1, _78C)) , true))) , (encouraging(_77A) , (ag(_77A, anna) , th(_77A, _78C))))) ,
the : (_7CD , ((woman(_7CD) , exists : (_7E2 , ((beauty(_7E2) , th(_7E2, _7CD)) , true))) ,
exists : (_7FB , (((inviting(_7FB) , (ag(_7FB, _7CD) , th(_7FB, marcel))) ,
```

```
purpose(_77A, _7FB)) , true))))) , true))

     woman(joan)
     beauty(o(13))
     th(o(13), joan)
     encouraging(o(23))
     ag(o(23), anna)
          perf(o(23), anna)
               or(o(23), anna)
               person(anna)
          public(o(23))
               encouraging(o(23))
     th(o(23), joan)
     woman(joan)
     beauty(o(13))
     th(o(13), joan)
     inviting(o(26))
     ag(o(26), joan)
          perf(o(26), joan)
               or(o(26), joan)
               person(joan)
          public(o(26))
               inviting(o(26))
     th(o(26), marcel)
     purpose(o(23), o(26))

yes

| ?- reasoner_prove.
> marcel was moved because marcel read the letter and because joan invited marcel

exists : (_786 , (((being_moved(_786) , th(_786, marcel)) ,
(the : (_7A7 , (letter(_7A7) , exists : (_7B8 , (((reading(_7B8) , (exp(_7B8, marcel) ,
th(_7B8, _7A7))) , reason(_786, _7B8)) , true)))) , exists : (_7E1 , ((inviting(_7E1) ,
(ag(_7E1, joan) , th(_7E1, marcel))) , reason(_786, _7E1))))) , true))

     being_moved(o(28))
     th(o(28), marcel)
     letter(billet)
     reading(o(27))
     exp(o(27), marcel)
          perf(o(27), marcel)
               or(o(27), marcel)
               person(marcel)
          private(o(27))
               reading(o(27))
     th(o(27), billet)
     reason(o(28), o(27))
          why(o(28), o(27))
     inviting(o(26))
     ag(o(26), joan)
          perf(o(26), joan)
               or(o(26), joan)
               person(joan)
          public(o(26))
               inviting(o(26))
     th(o(26), marcel)
     reason(o(28), o(26))
          why(o(28), o(26))

yes
```

```
| ?- reasoner_prove.
> marcel visited the beautiful woman because anna encouraged the beautiful woman
so that the beautiful woman invited marcel

exists : (_A5F , ((the : (_A71 , ((woman(_A71) , exists : (_A86 , ((beauty(_A86) ,
th(_A86, _A71)) , true))) , (visiting(_A5F) , (ag(_A5F, marcel) , th(_A5F, _A71))))) ,
exists : (_AB2 , ((the : (_AC4 , ((woman(_AC4) , exists : (_AD9 , ((beauty(_AD9) ,
th(_AD9, _AC4)) , true))) , (encouraging(_AB2) , (ag(_AB2, anna) , th(_AB2, _AC4))))) ,
the : (_B05 , ((woman(_B05) , exists : (_B1A , ((beauty(_B1A) , th(_B1A, _B05)) , true))) ,
exists : (_B33 , (((inviting(_B33) , (ag(_B33, _B05) , th(_B33, marcel))) ,
purpose(_AB2, _B33)) , true))))) , reason(_A5F, _AB2)))) , true))

     woman(joan)
     beauty(o(13))
     th(o(13), joan)
     visiting(o(31))
     ag(o(31), marcel)
          perf(o(31), marcel)
               or(o(31), marcel)
               person(marcel)
          public(o(31))
               visiting(o(31))
     th(o(31), joan)
     woman(joan)
     beauty(o(13))
     th(o(13), joan)
     encouraging(o(23))
     ag(o(23), anna)
          perf(o(23), anna)
               or(o(23), anna)
               person(anna)
          public(o(23))
               encouraging(o(23))
     th(o(23), joan)
     woman(joan)
     beauty(o(13))
     th(o(13), joan)
     inviting(o(26))
     ag(o(26), joan)
          perf(o(26), joan)
               or(o(26), joan)
               person(joan)
          public(o(26))
               inviting(o(26))
     th(o(26), marcel)
     purpose(o(23), o(26))
     reason(o(31), o(23))
          why(o(31), o(28))
          reason(o(28), o(23))
               why(o(28), o(27))
               reason(o(27), o(23))
                    why(o(27), o(25))
                    reason(o(25), o(23))
                         why(o(25), o(23))

yes
```

4.5.5 Closing the session

In order to close the session with the reasoner server, we send a `disconnect` message to the peer server thread .

```
| ?- reasoner_disconnect.

yes
```

We went through all these details because we think that future readers that are not particularly familiar with Qu-Prolog should at least get a flavour of how the system is to be operated.

Chapter 5

Conclusions

5.1 Achievements

5.1.1 Summary of Results

The aim of this thesis has been the development of a minimal semantic ontology merging intuitions from Strawson's theory of individuals and Parsons' theory of events and thematic roles. This ontology as a set of top-level conceptual distinctions is shown to be the foundation for the semantic subcategorisation of verbs and the basic logical structure of natural language sentences. The minimal ontology proposed in this thesis also underlies the shared semantics in a multi-agent system involving human as well as software agents interacting with each other at least partially via natural language communication. Indeed, software agents participating in such a system are required to share the fundamental conceptual distinctions of their human partners in order to process their messages.

The theoretical research leading to this new ontology can only be scientifically sound if based on experimentation. Empirical evidence corroborating the semantic theory established in the first three chapters comes from a small, but altogether persuasive implementation demonstrating how communication in a multi-agent system can rely on a minimal semantic ontology of the type we have described.

5.1.2 Achievements in detail

Theoretical developments

On the theoretical level, we can claim to have attempted and, as far as we know, achieved the following innovations:

1. We have partially adapted for our own purposes a formal theory of properties proposed by Guarino and Welty [35, 36, 69], mainly by dropping their concept of notional dependence in favour of the less constraining idea of referential dependence originating in the work of Strawson [60]. This alternative conception of dependence has been integrated into the existing theoretical framework of Guarino and Welty, thus leading to a modification compatible with the original proposal of a formal theory of properties.

2. This transformed modal account of unary properties has been systematically applied in the development of a new minimal ontology integrating and modifying Strawson's theory of individuals and Davidson's/Parsons' theory of events. The main features of this set of fundamental conceptual distinctions are:

   - the assumption that both substances (objects) and occurrences (states and events) are basic to our conceptual scheme;
   - the acknowledgement of persons as a basic particular-type;
   - the idea that the distinction between public (material) and private (mental) events is basic in the commonsense view of the world

3. These ontological distinctions, unraveled in the course of the discussion of both Strawson's and Parsons' work, and in the light of a modified formal theory of properties after Guarino and Strawson, are shown to be fundamental for the analysis of thematic roles and the semantic subcategorisation of English verbs. The main achievement of this work is the development of a new taxonomy of thematic roles and a reinterpretation of subject roles in particular, distinguishing between persons and non-persons on the one hand and private and public events on the other.

4. Finally, this new theory of thematic roles is applied to the formalisation of meaning of verbs and adjectives, as shown in the last chapter.

Implementation

On the practical level, we have implemented a demonstration in the form of a client-server architecture for a multi-agent system involving humans and reasoning software agents using as a central element of their interaction and communication a shared ontology complying with the top-level conceptual distinctions discussed in this work. While being less original than the theoretical part, the implementational component of this project meets the purposes

of a proof of principle. The following elements can justify a certain claim to independence, if not originality:

1. While the parser is based on standard natural language processing techniques, fundamental changes, if not innovations, have been necessary to treat variables ranging over occurrences, leading to, amongst other things, the splitting up of the sentence category in a phrase structure grammar. The bonus is a deeper understanding of the underlying semantics of verbs and sentences as well as of the relation of main clauses to subordinate clauses acting as adverbials of the main verb. The assumption of underlying events and parts played in them by various entities, both substances and occurrences, can be used to explain all these linguistic facts, as outlined by Parsons [47].

2. The server-client model is not original per se; it has been outlined by Robinson and Clark [57] for the purposes of emulating the Linda model of communication. Its transposition to the domain of natural language processing and reasoning can be seen as an autonomous incorporation of existing engineering concepts.

5.2 Future Work

We cannot but be aware of the fact that we have only scratched the surface of the issue of ontologies in multi-agent systems. Here is a perfunctory list of open questions we have come across during our research on this subject:

1. While we believe to have thoroughly clarified the taxonomy of thematic roles, we only have marginally addressed the issue of predicatives. While subject predicatives can be more or less awkwardly catered for in our account, object predicatives have been left out. Following Parsons [47], one may assume that these phenomena can only be treated in a theory of causatives. Such a theory is beyond the scope of this thesis, but may be worthwile exploring in the future.

2. Tesnière [65] proposes a contentious account of word categories in terms of grammatical categories, so that a complement would always be a noun, a supplement an adverb etc. This leads to a merger of morphology and grammar and presupposes a rather complex mechanism of word category change, as a morphemes word category shifts with its syntactical role. It would be interesting to put these ideas to a test, because if they were sound, they could imply that the basic ontology of

substances and occurrences permeates not only syntax and semantics, but also morphology.

3. We have not taken into account the problems of tense and aspect either and treated the problem of how occurrences temporally relate to other events or states only at the level of temporal adverbials. Parsons [47] develops a theory of tense and aspect formulated in terms of occurrence types and properties only. Leith and Cunningham [41] however propose an interval tense logic that can account for these phenomena without any reference to states or events. This could constitute a strong argument against occurrences as fundamental particular-types. Obviously there is a need to respond to this challenge.

4. The principle of a semantics based on occurrences would have to be applied in a large-scale natural language parser before one could definitively pronounce on the computational benefits of such a semantics.

5. Finally, more evidence is needed for supporting the claim that the minimal semantic ontology summarising the human conceptual framework has any computational interest outside of natural language communication and agent systems relying at least partly on it. Such evidence can only come from more experimentation, i.e. more tests through implementations.

These are only some of the issues we think have to be clarified in the future. But they cannot be expanded on within the scope of this thesis.

Code of the Implementation

The Parser

```
/*
 *
 *  PARSER
 *
 *  to be used with the Lexicon (lexicon.ql)
 *
 */



% Operator declarations
:- op(100,xfy,':').
:- op(101,xfy,'..').


/* SENTENCE CATEGORIES */

% (Complete) Sentence. Results from binding the occurrence variable
% of the main verb in a phrase of category s1. Depending on the mood
% (affirmative,negative), an existential or negated existential
% quantifier is applied to the occurrence variable.

% Note also that the meaning of s1 may be already a quantified
% proposition, so that one has to quantify the occurrence
% variable inside the scope of the overall quantifier.

% When the occurrence variable is bound, a dummy scope (true) has
% to be added, as the standard logical form of quantified sentences
% (that the prover is programmed to process) is:
% Quantifier:(Variable,Restrictor,Scope).

s(Complete) -->
        s1(Incomplete,force:Mood..pers:Person),
        {
          (
                (
                        Incomplete = 0^Det:(Var,Res,Sco),
                        Complete = Det:(Var,Res,Quant:(0,Sco,true))
                )
                ;
                (

                        Incomplete = 0^Sem,
                        Complete = Quant:(0,Sem,true)
                )
          ),
          (
                (
                        Mood = aff,
                        Quant = exists
                )
                ;
                (
                        Mood = neg,
                        Quant = no
                )
          )
        }.
```

```
% (Incomplete) Sentence. Because prepositions express relations between
% occurrences and other entities (see below), event/state variables must
% remain unbound up to highest possible level.
% s1 splits in a noun phrase and a verb phrase. The NP inherits the
% person feature, indicating (0,1) if the main verb is impersonal or not.
% The verb phrase inherits both the person and the force (mood) feature.

s1(Sem,force:Mood..pers:Person) -->
        np((X^Sco)^Sem,pers:Person),
        vp(X^Sco,force:Mood..pers:Person).



/* NOUN PHRASES */


% A NP np can be a single NP n1 or a determiner (quantifier)
% followed by an np.

np(Sem,pers:Person) --> n1(Sem,pers:Person).
np(Sem,pers:1) -->
        det((X^Res)^Sem),
        n1(X^Res,pers:1).


% An impersonal n1 (pers = 0) is simply the dummy
% subject ``it''

n1(Sem,pers:0) --> dn(Sem).


% A personal n1 can be either a proper name, a noun, or
% an adjective followed by a noun phrase. Note that the
% meaning of an adjective is a state; therefore, the
% respective state variable must be bound, before the
% adjective's and the noun phrase's meaning are
% combined.

n1(Sem,pers:1) --> pn(Sem).
n1(Sem,pers:1) --> n(Sem).
n1(X^(Base,exists:(S,Comp,true)),pers:1) -->
        adj(X^S^Comp),
        n1(X^Base,pers:1).


/* VERB PHRASES */

% In order to describe verb phrases adequately, we
% have to assume 4 categories (vp,v2,v1 and v0),
% according to the grammatical possibilities of
% combinations with prepositional phrases and
% noun phrases.

% 1. v0 represents the (possibly negated) full verb with its
%    auxiliaries, or a predicative adjective preceded by the
%    (negated) copula.
% 2. v1 is the level at which direct objects of bivalent
%    verbs or indirect objects of trivalent verbs are integrated.
% 3. v2 introduces the subcategorisation feature (subcat).
```

```
%    It also incorporates the direct objects of trivalent
%    verbs.
% 4. vp is the highest verb-related category. It optionally
%    merges v2 with a prepositional phrase.

% The subcategorisation feature has the following values:
%
% 0 - avalent
% 1 - monovalent
% 2 - bivalent
% 3 - trivalent


% A verb phrase vp is either a single verb phrase v2 or a
% verb phrase v2 followed by a prepositional phrase.
% The category v2 has an additional feature (subcat)
% indicating the valency of the verb.

vp(Sem,force:Mood..pers:Person) -->
        v2(Sem,force:Mood..pers:Person..subcat:Args).
vp(X^E^(Pred,Adv),force:Mood..pers:Person) -->
        v2(X^E^Pred,force:Mood..pers:Person..subcat:Args),
        pp(E^Adv).


% A verb phrase v2 is either a single verb phrase v1
% or a single verb phrase v1 followed by a noun
% phrase, if the verb is trivalent (the NP is the
% verb's Theme or direct object).

v2(Sem,force:Mood..pers:Person..subcat:Args) -->
        v1(Sem,force:Mood..pers:Person..subcat:Args).
        {Args = 0;Args = 1;Args = 2}.
v2(X^Pred,force:Mood..pers:Person..subcat:3) -->
        v1(Z^X^Sco,force:Mood..pers:Person..subcat:3),
        np((Z^Sco)^Pred,pers:1).


% A verb phrae v1 is a verb phrase v0 for avalent
% or monovalent verbs

v1(Sem,force:Mood..pers:Person..subcat:Args) -->
        v0(Sem,force:Mood..pers:Person..subcat:Args),
        {Args = 0; Args = 1}.

% A verb phrase v1 is a verb phrase v0 followed by a
% noun phrase (direct object of bivalent verbs, indirect
% object of trivalent verbs).

v1(X^Pred,force:Mood..pers:Person..subcat:2) -->
        v0(Y^X^Sco,force:Mood..pers:Person..subcat:2),
        np((Y^Sco)^Pred,pers:1).

% A verb phrase v1 is a verb phrase v0 followed by the
% sentential conjunction ``that'', followed by a complete
% sentence (in indirect speech, thus quantifying in or
% substitution by equivalent sentences is not possible).
% This is the case of bivalent verbs taking a sentence as
% Theme.
```

```
v1(X^Pred,force:Mood..pers:Person..subcat:2) -->
        v0(Sem^X^Pred,force:Mood..pers:Person..subcat:2),
        s_conj,
s(Sem).

% A verb phrase v1 is a verb phrase v0 followed by a
% noun phrase (Benefactive or Addressee of trivalent verbs).

v1(Z^X^Pred,force:Mood..pers:Person..subcat:3) -->
        v0(Z^Y^X^Sco,force:Mood..pers:Person..subcat:3),
        np((Y^Sco)^Pred,pers:1).


% A verb phrase v0 is a (negated) copula followed by
% an adjective. The copula does not add anyting to the
% meaning of the v0, except that it marks the mood.

v0(Sem,force:aff..pers:Person..subcat:1) -->
        cop,
        adj(Sem).
v0(Sem,force:neg..pers:Person..subcat:1) -->
        cop,
        neg,
        adj(Sem).

% A verb phrase v0 is a (negated) full verb. If
% negated, the verb is infinite.

v0(Sem,force:aff..pers:Person..subcat:Args) -->
        v(Sem,conj:fin..pers:Person..subcat:Args),
        {Args = 0;Args = 1;Args = 2;Args = 3}.
v0(Sem,force:neg..pers:Person..subcat:Args) -->
        aux,
        neg,
        v(Sem,conj:infin..pers:Person..subcat:Args),
        {Args = 0;Args = 1;Args = 2;Args = 3}.

% Finally, ``to be'' as a (negated) full verb is
% followed (and not preceded) by its negation.

v0(Sem,force:neg..pers:Person..subcat:2) -->
        v(Sem,conj:fin..pers:Person..subcat:2),
        neg.


% Auxiliary verbs
cop --> [was].
aux --> [did].

% Negation
neg --> [not].

% Sentential conjunction
s_conj --> [that].



/* PREPOSITIONAL PHRASES */

% Prepositions introduce noun phrases or sentences as
% adverbials; they express relations between occurrences and
% other entities, substances or occurrences. Prepositions,
```

```
% especially sentential ones, must have access to the
% occurrence variable of the verb's lambda form.

% To avoid syntactical ambiguities, we restrict our fragment
% of English to a sublanguage where nominal prepositions
% precede sentential prepositions.

% Again, there are several layers of incorporation:
% pp, p1 and p.


% A prep phrase pp is either a single prep phrase
% p1 or a nominal p1 followed by a sentential p1.

pp(O^Pred) --> p1(O^Pred,subcat:Phrase).
pp(O^(NP,S1)) -->
        p1(O^NP,subcat:np),
        p1(O^S1,subcat:s1).

% A prep phrase p1 is a single prep phrase p or a
% nominal/sentential p followed a p1 of the same
% category. In case of sentential p1, the single
% prep phrases p are separated by the conjunction
% 'and'. This restriction avoids syntactic
% ambiguities.

p1(O^Pred,subcat:Phrase) --> p(O^Pred,subcat:Phrase).
p1(O^(Comp,Base),subcat:NP) -->
        p(O^Comp,subcat:NP),
        p1(O^Base,subcat:NP).
p1(O^(Comp,Base),subcat:s1) -->
        p(O^Comp,subcat:s1),
        p_conj,
        p1(O^Base,subcat:s1).


% Sentential prep phrases consist of a preposition
% followed by a sentence. The occurrence variable
% of the subordinate phrase must be bound, and
% again this happens according to its mood.
% As in the case of the top category 's', a
% possible overall quantification of s1 must be taken
% account (necessity to quantify inside the scope
% of another quantifier).

p(O^Det:(Var,Res,Quant:(O1,(Sco,Adv),true)),subcat:s1) -->
        prep(O^O1^Adv,subcat:s1),
        s1(O1^Det:(Var,Res,Sco),force:Mood..pers:Person),
        {
           Mood = aff, Quant = exists
        ;
           Mood = neg, Quant = no
        }.

p(O^Quant:(O1,Pred,Adv),subcat:s1) -->
        prep(O^O1^Adv,subcat:s1),
        s1(O1^Pred,force:Mood..pers:Person),
        {
           Mood = aff, Quant = exists
        ;
           Mood = neg, Quant = no
        }.
```

```
% Finally, a nominal prper phrase p is a nominal
% preposition followed by a (personal) noun phrase.

p(O^Sem,subcat:np) -->
        prep(O^X^Pred,subcat:np),
        np((X^O^Pred)^O1^Sem,pers:1).


% The conjunction 'and' is used to join prep
% phrases p together.

p_conj --> [and].
```

The Lexicon

```
/*
 *
 *  LEXICON
 *
 *  to be used with the Parser (parser.ql)
 *  and the Knowledge Base (kb.ql)
 *
 *
 */



/*  VERBS  */

% The following thematic roles are used in the
% specification of verb meanings:
%
% th    - theme
% ag    - agent
% exp   - experiencer
% cause - cause
% addr  - addressee
% ben   - benefactive
%
% origin and performer are represented in the
% knowledge base only.

% Avalent verb

v(X^0^(raining(0),th(0,X)),conj:fin..pers:0..subcat:0) --> [rained].
v(X^0^(raining(0),th(0,X)),conj:infin..pers:0..subcat:0) --> [rain].


% Monovalent verbs

v(X^0^(shining(0),cause(0,X),th(0,X)),conj:fin..pers:1..subcat:0) --> [shone].
v(X^0^(shining(0),cause(0,X),th(0,X)),conj:infin..pers:1..subcat:0) --> [shine].

v(X^0^(going(0),or(0,X),th(0,X)),conj:fin..pers:1..subcat:0) --> [went].
v(X^0^(going(0),or(0,X),th(0,X)),conj:infin..pers:1..subcat:0) --> [go].

v(X^0^(living(0),th(0,X)),conj:fin..pers:1..subcat:0) --> [lived].
v(X^0^(living(0),th(0,X)),conj:infin..pers:1..subcat:0) --> [live].


% Bivalent verbs

v(Y^X^0^(being(0),or(0,X),th(0,Y)),conj:fin..pers:1..subcat:2) --> [was].

v(Y^X^0^(visiting(0),ag(0,X),th(0,Y)),conj:fin..pers:1..subcat:2) --> [visited].
v(Y^X^0^(visiting(0),ag(0,X),th(0,Y)),conj:infin..pers:1..subcat:2) --> [visit].

v(Y^X^0^(inviting(0),ag(0,X),th(0,Y)),conj:fin..pers:1..subcat:2) --> [invited].
v(Y^X^0^(inviting(0),ag(0,X),th(0,Y)),conj:infin..pers:1..subcat:2) --> [invite].

v(Y^X^0^(writing(0),ag(0,X),th(0,Y)),conj:fin..pers:1..subcat:2) --> [wrote].
v(Y^X^0^(writing(0),ag(0,X),th(0,Y)),conj:infin..pers:1..subcat:2) --> [write].
```

```
v(Y^X^O^(encouraging(O),ag(O,X),th(O,Y)),conj:fin..pers:1..subcat:2) --> [encouraged].
v(Y^X^O^(encouraging(O),ag(O,X),th(O,Y)),conj:infin..pers:1..subcat:2) --> [encourage].

v(Y^X^O^(liking(O),exp(O,X),th(O,Y)),conj:fin..pers:1..subcat:2) --> [liked].
v(Y^X^O^(liking(O),exp(O,X),th(O,Y)),conj:infin..pers:1..subcat:2) --> [like].

v(Y^X^O^(knowing(O),exp(O,X),th(O,Y)),conj:fin..pers:1..subcat:2) --> [knew].
v(Y^X^O^(knowing(O),exp(O,X),th(O,Y)),conj:infin..pers:1..subcat:2) --> [know].

v(Y^X^O^(reading(O),exp(O,X),th(O,Y)),conj:fin..pers:1..subcat:2) --> [read].
v(Y^X^O^(reading(O),exp(O,X),th(O,Y)),conj:infin..pers:1..subcat:2) --> [read].


% Trivalent verbs

v(Z^Y^X^O^(giving(O),ag(O,X),addr(O,Y),th(O,Z)),conj:fin..pers:1..subcat:3) --> [gave].
v(Z^Y^X^O^(giving(O),ag(O,X),addr(O,Y),th(O,Z)),conj:infin..pers:1..subcat:3) --> [give].

v(Z^Y^X^O^(sending(O),ag(O,X),addr(O,Y),th(O,Z)),conj:fin..pers:1..subcat:3) --> [sent].
v(Z^Y^X^O^(sending(O),ag(O,X),addr(O,Y),th(O,Z)),conj:infin..pers:1..subcat:3) --> [send].

v(Z^Y^X^O^(picking(O),ag(O,X),ben(O,Y),th(O,Z)),conj:fin..pers:1..subcat:3) --> [picked].
v(Z^Y^X^O^(picking(O),ag(O,X),ben(O,Y),th(O,Z)),conj:infin..pers:1..subcat:3) --> [pick].


/*  DETERMINERS  */

det((X^Res)^(X^O^Sco)^O^all:(X,Res,Sco)) --> [every].
det((X^Res)^(X^O^Sco)^O^exists:(X,Res,Sco)) --> [a].
det((X^Res)^(X^O^Sco)^O^exists:(X,Res,Sco)) --> [an].
det((X^Res)^(X^O^Sco)^O^the:(X,Res,Sco)) --> [the].
det((X^Res)^(X^O^Sco)^O^no:(X,Res,Sco)) --> [no].


/*  PREPOSITIONS  */

prep(O^O1^when(O,O1),subcat:s1) --> [when].
prep(O^O1^after(O,O1),subcat:s1) --> [after].
prep(O^O1^before(O,O1),subcat:s1) --> [before].
prep(O^O1^reason(O,O1),subcat:s1) --> [because].
prep(O^O1^purpose(O,O1),subcat:s1) --> [so,that].

prep(O^X^in(O,X),subcat:np) --> [in].
prep(O^X^from(O,X),subcat:np) --> [from].
prep(O^X^to(O,X),subcat:np) --> [to].
prep(O^X^instr(O,X),subcat:np) --> [with].


/*  NOUNS  */

n(X^man(X)) --> [man].
n(X^woman(X)) --> [woman].
n(X^person(X)) --> [person].
n(X^body(X)) --> [body].
n(X^city(X)) --> [city].
n(X^letter(X)) --> [letter].
n(X^pen(X)) --> [pen].
n(X^rose(X)) --> [rose].
n(X^sun(X)) --> [sun].
```

```
/*  ADJECTIVES  */

adj(X^O^(being_old(O),th(O,X))) --> [old].
adj(X^O^(being_large(O),th(O,X))) --> [large].
adj(X^O^(musicality(O),th(O,X))) --> [musical].
adj(X^O^(literacy(O),th(O,X))) --> [literate].
adj(X^O^(wisdom(O),th(O,X))) --> [wise].
adj(X^O^(sensitivity(O),th(O,X))) --> [sensitive].
adj(X^O^(being_moved(O),th(O,X))) --> [moved].
adj(X^O^(beauty(O),th(O,X))) --> [beautiful].
adj(X^O^(being_tall(O),th(O,X))) --> [tall].


/*  PROPER NAMES  */

pn((marcel^Sco)^Sco) --> [marcel].
pn((joan^Sco)^Sco) --> [joan].
pn((anna^Sco)^Sco) --> [anna].
pn((paris^Sco)^Sco) --> [paris].
pn((london^Sco)^Sco) --> [london].
pn((rome^Sco)^Sco) --> [rome].
pn((billet^Sco)^Sco) --> [billet].
pn((rosa^Sco)^Sco) --> [rosa].
pn((geha^Sco)^Sco) --> [geha].
pn((sol^Sco)^Sco) --> [sol].


/*  DUMMY NAME  */

dn((true^Sco)^Sco) --> [it].
```

The Knowledge Base

```
/*
 *
 * KNOWLEDGE BASE
 *
 * to be used with the Lexicon (lexicon.ql)
 * and the Prover (prover.ql)
 *
 */


/* RULES FOR SUBJECT-ROLES */

% The only thematic roles represented as basic predicates
% in the knowledge base are: Origin(or), Theme(th),
% Addressee (addr) and Benefactive (ben).
% Hence, Performer (perf), Cause (cause), Agent (ag) and
% Experiencer (exp) have to be defined in terms of the basic
% theta-roles and the types 'person', 'body', 'public'
% and 'private'.

perf(O,X) :- or(O,X), person(X).
cause(O,X) :- or(O,X), body(X).

ag(O,X) :- perf(O,X), public(O).
exp(O,X) :- perf(O,X), private(O).



/* TYPE DECLARATIONS */


/* Occurrences */

% Public occurrences

public(O) :-
        raining(O) ; shining(O) ; living(O) ; going(O) ;
        being(O) ; visiting(O) ; encouraging(O) ; inviting(O) ;
writing(O) ; giving(O) ; sending(O) ; picking(O) ;
        beauty(O) ; being_tall(O) ; being_old(O) ; being_large(O).

% Private occurrences

private(O) :-
knowing(O) ; liking(O) ; reading(O) ; wisdom(O) ;
        sensitivity(O) ;  musicality(O) ; literacy(O) ;
        being_moved(O).


/* Substances */

% Persons

person(marcel).
person(joan).
person(anna).
```

```
% Bodies

body(paris).
body(london).
body(rome).

body(billet).
body(rosa).
body(geha).
body(sol).


% Subtypes

man(marcel).
woman(joan).
woman(anna).

city(london).
city(paris).
city(rome).

letter(billet).
rose(rosa).
pen(geha).
sun(sol).



/*  WORLD FACTS  */

/* Rules about relations between occurrences */

% The meanings of some prepositions are not represented
% directly in the knowledge base and have to be defined
% in terms of fundamental relations between occurrences.

when(O,O1) :- at(O,O1) ; at(O1,O).        % simultaneity

before(O,O1) :- after(O1,O).              % anteriority

after(O,O1) :- post(O,O1).                % posteriority
after(O,O1) :- post(O,O2), after(O2,O1).

reason(O,O1) :- why(O,O1).                % reason, cause
reason(O,O1) :- why(O,O2), reason(O2,O1).


/* Eternal Occurences */

% Eternal occurrences are simultaneous to any
% other occurrence (in Prolog: at(O,_)).


% The state of being identical with sth. ;
% meaning of ``to be'' as a full verb.

being(true).
th(true,_).
or(true,X) :- th(true,X).
at(true,_).
```

```
% It rained in london

raining(o(1)).
th(o(1),true).
in(o(1),london).
at(o(1),_).


% The sun shone in paris

shining(o(2)).
or(o(2),sol).
th(o(2),sol).
in(o(2),paris).
at(o(2),_).


% The sun shone in rome

shining(o(3)).
or(o(3),sol).
th(o(3),sol).
in(o(3),rome).
at(o(3),_).

% london was large

being_large(o(4)).
th(o(4),london).
in(o(4),_).
at(o(4),_).

% paris was beautiful

beauty(o(5)).
th(o(5),paris).
in(o(5),_).
at(o(5),_).

% rome was old

being_old(o(6)).
th(o(6),rome).
in(o(6),_).
at(o(6),_).


% marcel was literate

literacy(o(7)).
th(o(7),marcel).
in(o(7),_).
at(o(7),_).

% marcel was sensitive

sensitivity(o(8)).
th(o(8),marcel).
in(o(8),_).
at(o(8),_).
```

```
% marcel was tall

being_tall(o(9)).
th(o(9),marcel).
in(o(9),_).
at(o(9),_).

% marcel lived in paris

living(o(10)).
th(o(10),marcel).
in(o(10),paris).
at(o(10),_).


% joan was musical

musicality(o(11)).
th(o(11),joan).
in(o(11),_).
at(o(11),_).

% joan was wise

wisdom(o(12)).
th(o(12),joan).
in(o(12),_).
at(o(12),_).

% joan was beautiful

beauty(o(13)).
th(o(13),joan).
in(o(13),_).
at(o(13),_).

% joan lived in london

living(o(14)).
th(o(14),joan).
in(o(14),london).
at(o(14),_).


% anna was musical

musicality(o(15)).
th(o(15),anna).
in(o(15),_).
at(o(15),_).
```

```
% anna was sensitive

sensitivity(o(16)).
th(o(16),anna).
in(o(16),_).
at(o(16),_).

% anna was tall

being_tall(o(17)).
th(o(17),anna).
in(o(17),_).
at(o(17),_).

% anna lived in rome

living(o(18)).
th(o(18),anna).
in(o(18),rome).
at(o(18),_).


% marcel liked joan because she was musical
% and all that

liking(o(19)).
or(o(19),marcel).
th(o(19),joan).
in(o(19),_).
at(o(19),_).
why(o(19),o(11)).
why(o(19),o(12)).
why(o(19),o(13)).

% joan liked marcel because he was literate
% and all that

liking(o(20)).
or(o(20),joan).
th(o(20),marcel).
in(o(17),_).
at(o(20),_).
why(o(20),o(7)).
why(o(20),o(8)).
why(o(20),o(9)).

% anna knew that marcel liked joan

knowing(o(21)).
or(o(21),anna).
th(o(21),exists:(A,(liking(A),exp(A,marcel),th(A,joan)),true)).
in(o(21),_).
at(o(21),_).

% anna knew that joan liked marcel

knowing(o(22)).
or(o(22),anna).
th(o(22),exists:(A,(liking(A),exp(A,joan),th(A,marcel)),true)).
in(o(22),_).
at(o(22),_).
```

```
/*  Temporary occurrences  */

% anna encouraged joan because anna knew that joan liked marcel (o(20))
% and marcel liked joan (o(19)), so that joan invited marcel (o(26)).

encouraging(o(23)).
or(o(23),anna).
th(o(23),joan).
in(o(23),rome).
why(o(23),o(19)).
why(o(23),o(20)).
purpose(o(23),o(26)).

% joan wrote a letter with a pen so that she sent it to
% to marcel (o(25)), and invited him (o(26)),
% because anna encouraged her (o(23)) and because she
% liked marcel (o(20)).

writing(o(24)).
or(o(24),joan).
th(o(24),billet).
instr(o(24),geha).
in(o(24),london).
post(o(24),o(23)).
why(o(24),o(23)).
why(o(24),o(20)).
purpose(o(24),o(25)).
purpose(o(24),o(26)).

% joan sent marcel a letter so that she invited him to
% london (o(26)), because anna encouraged her (o(23)) and
% because she liked marcel (o(20)).

sending(o(25)).
or(o(25),joan).
th(o(25),billet).
addr(o(25),marcel).
in(o(25),london).
post(o(25),o(24)).
why(o(25),o(23)).
why(o(25),o(20)).
purpose(o(25),o(26)).

% joan invited marcel to london him because she sent him
% a letter (o(25)).

inviting(o(26)).
or(o(26),joan).
th(o(26),marcel).
to(o(26),london).
in(o(26),london).
post(o(26),o(25)).
why(o(26),o(25)).

% marcel read the letter because joan sent him it (o(25)).

reading(o(27)).
or(o(27),marcel).
th(o(27),billet).
in(o(27),paris).
```

```
post(o(27),o(26)).
why(o(27),o(25)).

% marcel was moved because he read the
% letter (o(27)), and because joan invited him (o(26)).

being_moved(o(28)).
th(o(28),marcel).
post(o(28),o(27)).
why(o(28),o(27)).
why(o(28),o(26)).

% marcel picked joan a rose so that he gave her it (o(32))
% because he was moved (o(28)).

picking(o(29)).
or(o(29),marcel).
th(o(29),rosa).
ben(o(29),joan).
in(o(29),paris).
post(o(29),o(27)).
why(o(29),o(28)).
purpose(o(29),o(32)).

% marcel went from paris to london so that he visited joan
% (o(31)), and gave her the rose (o(32)), because he was moved
% (o(28)).

going(o(30)).
or(o(30),marcel).
th(o(30),marcel).
from(o(30),paris).
to(o(30),london).
post(o(30),o(29)).
why(o(30),o(28)).
purpose(o(30),o(31)).
purpose(o(30),o(32)).

% marcel visited joan, so that he gave her the rose,
% because he was moved.

visiting(o(31)).
or(o(31),marcel).
th(o(31),joan).
in(o(31),london).
post(o(31),o(30)).
why(o(31),o(28)).
purpose(o(31),o(32)).

% marcel gave joan the rose, because he picked it
% for her (o(29)).

giving(o(32)).
or(o(32),marcel).
th(o(32),rosa).
addr(o(32),joan).
in(o(32),london).
post(o(32),o(31)).
why(o(32),o(29)).
```

The Prover

```
/*
 *
 * A Prolog meta-interpreter
 *
 * courtesy Frank Kriwaczek, Imperial College, London
 *
 * adapted for quantifiers by the author
 *
 * to be used with the knowledge base (kb.ql)
 * and the reasoner server (reasoner_server.ql)
 *
 */

%
% prove(Clause,ProofTree) attempts to prove Clause, building
% up ProofTree, a nested list of premisses.
%

prove(\+ Goal, [\+ Goal]) :- !,           % negation
        \+ prove(Goal, Proof).
prove(Goal,[Goal]) :-                     % built-in
        built_in(Goal),
        call(Goal).
prove((Goal,Goals),CombProof) :- !,       % conjunction
        prove(Goal,Proof),
        prove(Goals,Proofs),
        append(Proof,Proofs,CombProof).
prove((Goal;Goals),CombProof) :- !,       % disjunction
        (
        prove(Goal,Proof),
        CombProof = Proof
        ;
        prove(Goals,Proofs),
        CombProof = Proofs
        ).
prove(all:(_,Goal,Goals),Proof) :-        % universal quantifier
        prove(\+ (Goal, \+ Goals), Proof).
prove(exists:(_,Goal,Goals),Proof) :-     % existential quantifier
        prove((Goal,Goals),Proof).
prove(the:(X,Goal,Goals),Proof) :-        % descriptor, ``at most one''
        findall(X,prove(Goal,_),List),
        sort(List,Results),
        length(Results,1),
        prove((Goal,Goals),Proof).
prove(no:(_,Goal,Goals),Proof) :-         % negated existence
        prove(\+(Goal,Goals),Proof).
prove(Goal,[Goal,Proof]) :-               % clause (fact or rule)
        clause(Goal,Conditions),
        prove(Conditions,Proof).

built_in(true).                           % built-in expressions
built_in(X = Y).
```

The Reasoner Server

```
/*
 *
 * REASONER SERVER
 *
 * after the client-server architecture for the Linda model
 * by Peter Robinson (University of Queensland)
 * and Keith Clark (Imperial College)
 *
 * To be used with the reasoner client (reasoner_client.ql)
 *
 * After compilation, the executable has to be started with
 * ``-A reasoner_server_process'' in order to name it.
 *
 */


%
% Start the Reasoner server
%
main(_) :-
        [parser,prover,lexicon,kb],                  % load components
        thread_set_symbol(reasoner_server_thread),   % name the main thread
        main_loop.

%
% loop processing client connect messages by
% forking a thread to deal with the client's queries.
%
main_loop :-
        repeat,
        connect <<- RtAddr,                          % wait for a connect message
        thread_fork_anonymous(_, reasoner_thread(RtAddr)),
        fail.

%
% The initial goal of the forked thread: finishes the connection
% handshake with the client and goes into the message processing
% loop.
%
reasoner_thread(A) :-
        connected ->> A,                             % finish the handshake
        thread_loop(A).
```

```
%
% The forked thread's message processing loop.
%
thread_loop(A) :-
        repeat,
        message_choice
        (
           parse(S) <<- A ->        % parse message received from client
                (   s(Sem,S,[])     % parse sentence
                    ->
                        parsed(Sem) ->> A   % result sent to client
                    ;
                        fail ->> A          % failure notification
                )
            ;
            prove(S) <<- A ->        % prove message received from client
                (   s(Sem,S,[])                         % parse the sentence
                    ->
                        ( (copy_term(Sem,Form),
                           prove(Form,Proof))           % prove sentence meaning
                        ->
                                proved(Sem,Proof) ->> A  % meaning and proof sent to client
                        ;
                                prove_fail ->> A         % proof failure notification
                        )
                    ;
                        parse_fail ->> A                % parsing failure notification
                )
            ;
            disconnect <<- A ->          % if disconnect message is received
                        thread_exit      % terminate the thread.
        ),
        fail.
```

The Reasoner Client

```
/*
 *
 * REASONER CLIENT
 *
 *
 * after the client-server architecture for the Linda model
 * by Peter Robinson (University of Queensland)
 * and Keith Clark (Imperial College)
 *
 * to be used with the reasoner server (reasoner_server.ql)
 *
 */


%
% Load input-output library
%

:- consult(io).

%
% reasoner_connect sends a connection request to the main thread
% of the reasoner server. As soon as a connected message from the
% forked server thread is received, its address is recorded so that
% the client's queries are sent to the competent server thread.
% The thread ID of the client is  stored too in order to allow
% several clients within the same process.
%

reasoner_connect :-
        connect ->> reasoner_server_thread:reasoner_server_process,
        connected <<= A,
        thread_tid(TID),
        assert(idaddr(TID,A)).

%
% disconnect from the reasoner server and remove the stored
% server thread address.
%

reasoner_disconnect :-
        thread_tid(TID),
        retract(idaddr(TID,A)),
        disconnect ->> A.
```

```
%
% send a user input sentence to be parsed to the server
% thread and display the result sent back by the peer;
% notify the user in case of a failure.
%

reasoner_parse :-
        read_list(S),       % get user input sentence
        thread_tid(TID),
        idaddr(TID,A),
        parse(S) ->> A,     % send parse request to server
        M <<= A,            % wait till reply in message queue
        (
          M = parsed(Sem),  % success: result returned
          nl,write(Sem)
        ;
          M = fail,         % failure notification to user
          write('ungrammatical')
        ).


%
% send a user input sentence to be proved to the server thread and
% display logical form and proof tree sent back by the peer ;
% notify the user about parsing or proof failures
%

reasoner_prove :-
        read_list(S),             % get user input sentence
        thread_tid(TID),
        idaddr(TID,A),
        prove(S) ->> A,           % send proof request to server
        M <<= A,                  % wait until reply in message queue
        (
          M = proved(Sem,Proof),  % success: display meaning and proof
          nl,
          write(Sem),
          nl,nl,
          ppr(Proof)              % pretty-print proof
        ;
          M = parse_fail,         % parse failure
          write('ungrammatical')
        ;
          M = prove_fail,         % proof failure
          write('unprovable')
        ).
```

Input-Output Utilities

```
/*
 *
 * INPUT-OUTPUT ROUTINES
 *
 * to be used with the Reasoner client (reasoner_client.ql)
 *
 */

%
% 'Public-domain' utility to read in a line and transform
% it into a list of atoms (words)
%

read_list(L) :-
        write('> '),
        get_line(CL),
        wordlist(L,CL,[]).

wordlist([X|Y]) --> word(X), whitespace, wordlist(Y).
wordlist([X]) --> whitespace, wordlist(X).
wordlist([X]) --> word(X).
wordlist([X]) --> word(X), whitespace.

word(W) --> charlist(X), {name(W,X)}.

charlist([X|Y]) --> chr(X), charlist(Y).
charlist([X]) --> chr(X).

chr(X) --> [X], {X>=48}.

whitespace --> whsp, whitespace.
whitespace --> whsp.

whsp --> [X], {X<48}.


%
% Utility to 'pretty-print' nested lists
%
% Courtesy of Frank Kriwaczek, Imperial College
%

ppr(A) :-
pp(A,0).

pp([],N) :- !.
pp([A|B],N) :- !,
        N1 is N + 1,
        pp(A,N1),
        pp(B,N).
pp(true,N) :- !.
pp(A,N) :-
        M is 5 * N,
        tab(M),
        write(A),
        nl.
```

Bibliography

[1] Austin, J. L. 1962. *How to do Things with Words.* Clarendon, Oxford.

[2] Brachman, R. 1979. On the Epistemological Status of Semantic Networks. In N.V. Findler (ed.), *Associative Networks: Representation and Use of Knowledge by Computers.* Academic Press.

[3] Bouaud, J., Bachimont, B., Charlet, J. and Zweigenbaum, P. 1995. Methodological Principles for Structuring an "Ontology". *Proceedings of the IJCAI 95 Workshop on Basic Ontological Issues in Knowledge Sharing.*

[4] Bratko, I. 2001. *Prolog Programming for Artificial Intelligence.* 3rd edition. Addison-Wesley, London, New York.

[5] Buber, M. 1997 (1962). *Das Dialogische Prinzip.* Lambert Schneider, Heidelberg.

[6] Carrara, M. and Varzi, A. 1999. Ontological Commitment and Reconstruction. *Proceedings of the Third European Conference of Analytic Philosophy.* To appear in *Erkenntnis.*

[7] Clark, K., Robinson, P. and Hagen, R. 1999. Multi-Threading and Message Communication in Qu-Prolog. Technical Report 99-41, Software Verification Research Center, University of Queensland.

[8] Clocksin, W. and Mellish, C. 1997. *Programming in Prolog.* 4th edition. Springer, Berlin, New York.

[9] Covington, M. 1994. *Natural Language Processing for Prolog Programmers.* Prentice Hall, Upper Saddle River/NJ.

[10] de Saussure, F. 1985. *Cours de Linguistique Générale.* Payot, Paris.

[11] Davidson, D. 1980a. The Logical Form of Action Sentences. In [12], pp. 105-148.

[12] Davidson, D. 1980. *Essays on Actions and Events.* Oxford University Press, Oxford.

[13] Davidson, D. 1984b. Truth and Meaning. In [16] pp. 17-36.

[14] Davidson, D. 1984a. Moods and Performances. In [16] pp. 109-121.

[15] Davidson, D. 1984b. The Method of Truth in Metaphysics. In [16] pp. 199-214.

[16] Davidson, D. 1984. *Inquiries into Truth and Interpretation.* Clarendon, Oxford.

[17] Fillmore, C. J. 1968. The Case for Case. In E. Bach, R. T. Harms (eds.), *Universals in Linguistic Theory*, London: 1-88

[18] Finin, T., Weber, J., et alios 1994. Specification of the KQML Agent Communication Language. The DARPA Knowledge Sharing Initiative, External Interfaces Working Group. Available at: http://logic.stanford.edu/sharing/papers/kqml.ps .

[19] Fitting, M. 1993. Basic Modal Logic. In D. Gabbay, C. Hogger (eds), *Handbook of Logic in Artificial Intelligence and Logic Programming.* Clarendon, Oxford.

[20] Frank, A. 2001 (to appear). Computational Models for Ontology. In *Proceedings of FOIS 2001, Int. Conference on Formal Ontology in Information Systems.*

[21] Gangemi, A., Guarino, N., Masolo, C., and Oltramari, A. 2001. Understanding top-level ontological distinctions. *Proceedings of IJCAI 2001 workshop on Ontologies and Information Sharing.*

[22] Gruber, T. 1991. The Role of Common Ontology in Achieving Sharable, Reusable Knowledge Bases. *Principles of Knowledge Reprersentation and Reasoning: Proceedings of the Second International Conference.* Morgan Kaufmann, San Mateo/CA.

[23] Gruber, T. 1995. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human and Coputer Studies. Special Issue: Formal Ontology, Conceptual Analysis and Knwoledge Representation.* Kluwer Academic Publishers. Also available as Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University.

[24] Guarino, N., Carrara, M. and Giaretta,P. 1994. An Ontology of Meta-Level Categories. In D. J., E. Sandewall and P. Torasso (eds.), *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR94).* Morgan Kaufmann, San Mateo/CA: 270-280. .

[25] Guarino, N., Carrara, M., and Giaretta, P. 1994. Formalizing Ontological Commitment. In *Proceedings of National Conference on Artificial Intelligence (AAAI-94).* Seattle, Morgan Kaufmann: 560-567.

[26] Guarino, N. 1994. The Ontological Level. In R. Casati, B. Smith and G. White (eds.),*Philosophy and the Cognitive Sciences.* Hölder-Pichler-Tempsky, Vienna: 443-456.

[27] Guarino, N. 1995. Formal Ontology, Conceptual Analysis and Knowledge Representation. *International Journal of Human and Coputer Studies 43(5/6). Special Issue: Formal Ontology, Conceptual Analysis and Knwoledge Representation.* Kluwer Academic Publishers: 625-640.

[28] Guarino, N., Borgo, S. and Masolo, C. 1996. Towards an ontological theory of physical objects. *Computational Engineering in System Applications (CESA 96).*

[29] Guarino, N. 1997. Some Organizing Principles for a Top-Level Ontology. {emphLADSEB-CNR Int. Rep. 02/97 V.3.0-August 1997.

[30] Guarino, N. 1998. Some Ontological Principles for Designing Upper Level Lexical Resources. In A. Rubio, N. Gallardo, R. Castro and A. Tejada (eds.), *Proceedings of First International Conference on Language Resources and Evaluation.* ELRA - European Language Resources Association, Granada, Spain: 527-534.

[31] Guarino, N. 1999. The Role of Identity Conditions in Ontology Design. In C. Freksa and D. M. Mark (eds.), *Spatial Information Theory - Cognitive and Computational Foundations of Geographic Information Science. Proceedings of International Conference COSIT '99.* Springer Verlag, Berlin: 221-234.

[32] Guarino, N. and Welty, C. 2000. Identity, Unity, and Individuality: Towards a Formal Toolkit for Ontological Analysis. In H. Werner (ed.) *ECAI-2000: The European Conference on Artificial Intelligence.* IOS Press, Berlin, Germany: 219-223.

[33] Guarino, N. and Welty, C. 2000. Ontological Analysis of Taxonomic Relationships. In A. Länder and V. Storey (eds.), *Proceedings of ER-2000: The 19th International Conference on Conceptual Modeling.* Springer Verlag.

[34] Guarino, N. and Welty, C. 2000. Towards a methodology for ontology-based model engineering. *Proceedings of the ECOOP-2000 Workshop on Model Engineering.* Cannes, France.

[35] Guarino, N. and Welty, C. 2000. A Formal Ontology of Properties. In R. Dieng and O. Corby (eds.), *Knowledge Engineering and Knowledge Management: Methods, Models and Tools. 12th International Conference, EKAW2000.* Springer Verlag: 97-112.

[36] Guarino, N. and Welty, C. 2001. Identity and Subsumption. LADSEB-CNR Internal Report 01/2001. In Green, R. (ed.), *Semantic Relations*, Kluwer Academic Publishers.

[37] Hayes, P. 1985. The Second Naive Physics Manifesto. In J. R. Hobbs and R. C. Moore (eds), *Formal Theories of the Commonsense World.* Ablex, Norwood/NJ.

[38] Huhns, M. and Stephens, L. 1999. Multiagent Systems and Societies of Agents. In [68]: pp. 79-120.

[39] Jasper, R. and Uschold, M. 1999. A Framework for Understanding and Classifying Ontology Applications. *Proceedings of the IJCAI-99 ontology workshop.*

[40] Labrou, Y. and Finin, T. 1997. A Proposal for a new KQML Specification. Technical Report TR CS-97-03, Computer Science and Electrical Engineering Department, University of Maryland, Baltimore.

[41] Leith, M. and Cunningham, J. 2001. Aspect and Interval Tense Logic. *Linguistics and Philosophy 24*: pp. 331-381.

[42] Lowe, E.J. 1989. *Kinds of Being. A Study in Individuation, Identity and Logic of Sortal Terms.* Basil Blackwell, Oxford.

[43] McCabe, F. 1999. *ICM Reference Manual.* Fujitsu Labs of America. Available at: http://www.nar.fla.com/icm/manual.html .

[44] Montague, R. 1974a. English as a Formal Language. In [46]: pp. 188-221.

[45] Montague, R. 1974b. Universal Grammar. In [46]: pp. 222-246.

[46] Montague, R. , Thomason, R.H (ed.) 1974. *Formal Philosophy: Selected Papers of Richard Montague.* Yale University Press.

[47] Parsons, T. 1990. *Events in the Semantics of English: A Study in Subatomic Semantics.* MIT Press, Cambridge/MA.

[48] Quine, W.V. 1950. *Methods of Logic.* Holt, New York.

[49] Quine, W.V. 1951. *Mathematical Logic.* Revised edition. Harvard University Press, Cambridge/MA., London.

[50] Quine, W.V. 1961a. Logic and the Reification of Universals. In [52], pp. 102-129.

[51] Quine, W.V. 1962b. Reference and Modality. In [52], pp. 139-159.

[52] Quine, W.V. 1961. *From a Logical Point of View. Nine Logico-Philosophical Essays.* Harvard University Press, Cambridge/MA.

[53] Quine, W.V. 1969. *Ontological Relativity and Other Essays.* Columbia University Press, New York.

[54] *Qu-Prolog 6.0 Reference Manual.* Software Verification Research Center, University of Queensland. Available at: http://www.doc.ic.ac.uk/lab/lab_area/mac/qp6.0/doc/manual/ .

[55] Russell, B. 1905. On denoting. In *Mind 14*: pp. 479-493.

[56] Russell, S. and Norvig, P. 1995. *Artificial Intelligence. A Modern Approach.* Prentice Hall, Englewood Cliffs,NJ.

[57] Robinson, P. 2000. Qu-Prolog 6.0 User Guide. Technical Report 00-20, Software Verification Research Center, University of Queensland.

[58] Searle, J.R. 1969. *Speech Acts: An Essay in the Philosophy of Language.* Cambridge University Press, Cambridge.

[59] Simons, P. 1987. *Parts. A Study in Ontology.* Clarendon, Oxford.

[60] Strawson, P. 1959. *Individuals. An Essay in Descriptive Methaphysics.* Routledge, London.

[61] Strawson, P. 1992. *Analysis and Metaphysics. An Introduction to Philosophy.* Oxford University Press, Oxford.

[62] Tarski, A. 1956a. The Concept of Truth in Formalized Languages. In [63]: pp. 151-278.

[63] Tarski, A. 1956. *Logic, Semantics, Metamathematics.* Clarendon Press, Oxford.

[64] Tesnière, L. 1953. *Esquisse d' une syntaxe structurale.* Klincksieck, Paris.

[65] Tesnière, L. 1959. *Éléments de syntaxe structurale.* Klincksieck, Paris.

[66] Uschold,M. and Gruninger,M. 1996a. Ontologies: Principles, methods and applications. *Knowledge Engineering Review, 13(1).* Also available as AIAI-TR-196 from AIAI, The University of Edinburgh.

[67] Uschold,M. and Gruninger,M. 1996b. Building Ontologies: Towards a Unified Methodology. *Proceedings of Expert Systems '96.* Also available as AIAI-TR-197 from AIAI, The University of Endinburgh.

[68] Weiss, G. (ed.) 1999. *Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence.* MIT Press, Cambridge,MA/London.

[69] Welty, C., Guarino, N. 2001. Supporting Ontological Analysis of Taxonomic Relationships. To appear in *Data and Knowledge Engineering*, September 2001.

[70] Wooldridge, M. and Jennings, N. 1995. Intelligent Agents: Theory and Practice. *Knowledge Engineering Review.*

[71] Wooldridge, M. 1999. Intelligent Agents. In [68]: pp. 27-77.