Legal theory, sources of law and the semantic web

Boer, A.

DOI
10.3233/978-1-60750-003-2-i

Publication date
2009

Chapter 2

Knowledge Engineering and the Law

2.1. Introduction

Knowledge Engineering is the technique applied by knowledge engineers to build *knowledge based systems* (KBS). Given a knowledge-intensive task normally performed by knowledgeable human problem solvers, the knowledge engineer attempts to model the domain knowledge and problem solving techniques of the human problem solver, and uses this model to implement a KBS capable of performing the knowledge-intensive task.

The KBS in principle replaces the human problem solver capable of performing the task by himself, either completely or – more commonly – by allowing less skilled workers to perform the same task assisted by the KBS. The introduction of a KBS is typically intended to take boring routine tasks out of the hands of scarce experts so that they can spend more time on the hard problems. If KBS are for instance capable of handling the great bulk of tax applications without any human intervention – a stage that has already been reached in a number of countries – the experts in the tax administration will get to spend more time on complicated and suspect cases.

Scarcity of expertise seems inescapable in the legal field in the foreseeable future: any increase in efficiency of the legal system will presumably immediately be canceled by increasing demand. The argument for this observation is simple: if the legal system is the expensive *last resort* solution for maintaining normative order in a society, as is often argued, any reduction in the costs of this solution generates increased demand. If the fielded KBS is cheaper than the people it replaces it is therefore reasonable to expect that the use of KBS increases the quality of jobs in the legal field, while making the legal solutions more accessible.

Effectiveness is however usually the more important reason to field KBS in the legal system: *uniformity* (or *stare decisis*) in legal decision making is a goal in itself in the field of law, and uniformity brings down the overall costs of the legal system by giving people less incentive to reject or appeal against its decisions.

The legal expert's role gradually shifts from decision making to setting uniform decision making policies for the KBS, i.e. to producing (decision making) knowledge instead of decisions. This shift is significant, and is mirrored by a recent shift in attention in knowledge engineering and knowledge management literature (cf. [191]): the shift from attention for the problem solving competence of KBS to attention for the decentralized maintenance of reusable knowledge components.

In this chapter the discipline of knowledge engineering, and its subject matter, is introduced, to help the reader position the observations and judgments in the following chapters in their context of use.

The framework sketched in this chapter is however rather specific for a number of goals. Reasoning and knowledge as conceived in this book are based in the concept of reasoning as *problem solving*, and knowledge as a resource in problem solving. Section 2.2 introduces this view of knowledge, and introduces knowledge components.

Since the quest for reusability puts the question of falsifiability of knowledge on the center stage, the distinction – made in section 2.2.5 – between *ontological* knowledge that can be considered not falsifiable, and other knowledge that is considered falsifiable, receives a lot of attention. The concepts of *ontology*, *abstraction* and *ontological* *stratification* play a key role in this, and are explained in section 2.3.

The concept of problem solving, and the role of falsifiable inference in problem solving, is addressed in section 2.4 about epistemology. In addition a distinction is drawn between knowledge and reasoning, the subject of this book, on the one hand, and information and argumentation on the other hand in section 2.5. This distinction is intended for scoping purposes: although there is a lot one should know about information and argumentation to design good LKBS, this subject the beyond the scope of this book.

2.1.1. Legal Knowledge Based Systems

A *legal knowledge based system* (LKBS) is a KBS that mainly deals with legal subject matter.

Legal knowledge based systems are mostly used for routine tasks, usually by public bodies specialized in a limited number of decision processes constrained by law relating to one topic. I have for instance worked in applied research projects in taxation (e.g. [270, 51, 50]), social security (e.g. [272, 271, 287]), and court (e.g. [60, 64]) settings.

Unfortunately no empirical studies along the lines of [124] have been undertaken specifically on *legal* KBS: there appears to be no hard data about their prevalence, or about how they fare over time. Rare empirical research into the quality of decisions made by LKBS suggests that matching or exceeding human performance is quite feasible (cf. generally [129]). Generalizing from the results of [124] one might speculate that many more than the ones discussed in academic literature qualify as LKBS[1], and that two out of three systems fall out of use within 5 years. In the field of law a system may be discarded after some years because the relevant sources of law have changed so considerably that adapting the LKBS becomes too costly or time-consuming.

LKBS in combination with *Semantic Web* technologies, and information standardization in general, are, under the moniker *eGovernment*, often proposed in political-administrative circles as *potentially* the most effective non-legal solution for reducing the *administrative burden*, the costs of carrying out administrative activities that one would not carry out in the absence of regulation (cf. the bibliography in [7], or the European Commission's view in COM2007/23, p.12, which mentions "intelligent portals").

This administrative burden is often conceived of as a form of economic *deadweight loss*. Note that this deadweight loss is incurred both through taxes spent on administrative processes within the government and through direct administrative costs made by citizens and businesses. Political attention appears to be directed towards the direct costs of businesses: as pointed out in [7] reduction of the administrative burden may simply move the deadweight from businesses to the government instead of reducing it. The justification of this belief in reducing economic deadweight loss through "intelligent portals" is beyond the scope of this book.

While the impact of LKBS on institutional effectiveness and efficiency is hard to judge, in part because of the hidden demand issue noted earlier, LKBS *could* have a

[1]An even more daring speculation: there is probably also no positive relation between output of scientific publications about a system and its real world success.

disruptive, even revolutionary, effect on an inter-institutional level.

IT in general turns out to be a major force in leveling playing fields between governments competing for favour in some areas: we for instance regularly see very unlikely candidates scoring high on *eGovernment* rankings. In [279] we see for instance Equitorial Guinea, Liberia, Sierra Leone, Bhutan, Eritrea, Gabon, and Ethiopia outranking not just wealthier, but also clearly IT-savvy nations such as Japan, France, Sweden, Denmark, and Norway in 2007. The most obvious explanation for these outliers is that these countries invest more (relatively speaking) in eGovernment for competitive reasons.

Competitive use of eGovernment more often than not involves automated decision making processes. Electronic tax filing appears to be a major contributing factor to increased tax competition between countries, as it becomes increasingly easier for companies to declare taxes and duties wherever it is judged to be most advantageous. The LKBS helps to advertise the legal services a country provides, by making them easy and efficient to use. LKBS that rearrange business operations continually to optimize taxes – declared to government administration LKBS – are foreseeable. In general LKBS could make traffic of persons, goods and services over national borders a lot easier, provided that systems can interface with each other, and their inputs and outputs can be easily compared. But this turns out to be a major problem, for reasons described in section 2.2.4.

In the many areas where governments however do not compete with each other for the customer's favour, eGovernment easily becomes merely window dressing instead of a new window: an internet-based portal connected to the same old, slow decision making processes (cf. generally [71, 8, 7]).

The field of legal knowledge engineering specializes in knowledge engineering topics specific or typical to the development of LKBS. Some of these topics are:

Formal sources of law play a central role as written *knowledge sources* in LKBS development and maintenance. In law the sources telling us how to think and how to act are *explicit*. This characteristic seems to make law eminently suitable to KBS, and law is indeed one of the oldest areas of application of KBS (viz. [200]). At the same time, legal constraints on "how to think" do not necessarily make building a system, or formulating generic theories about the design of such systems, *easier*. So-called *tacit knowledge* (cf. [68, 191]) – which is hard to make explicit – of experts plays a larger role in other disciplines. In most KBS projects the required knowledge has to be "discovered" and "acquired": in law it is more readily available (cf. section 3.5.1).
In law, it is also often change in the sources of law, sometimes just a court verdict contrary to the interpretation of the LKBS, that dictates an update of the LKBS. The law continually makes the knowledge it creates explicit in the form of (changes to) the sources of law, and itself sets the deadlines for when it should be used in decision making. This is uncommon in many other application domains, where the represented domain usually must remain constant for fairly long times, or only changes at the instigation of the KBS user[2], to make the deployment of a KBS

[2]The introduction of new components in the product catalogue for instance prompts the intro-

justifiable.

The distinction between prescriptive and descriptive is important in all types of reasoning, but the field of law has turned the specification and categorization of prescriptive knowledge into an art form. Most reasoning tasks require great amounts of invariant, descriptive domain knowledge, and a few goals, constraints, requirements, norms to prescribe what the generic solution looks like. Law is about mostly concerned with the organization of prescriptive knowledge, and borrows descriptive knowledge from other knowledge domains when needed.

The central role of justification is an essential characteristic of law. The procedurally and substantively correct method is more important than the quality of the outcome, for which no independent standard is available. Most academic disciplines apply scientific method to some domain of interest and are primarily interested in correct outcomes, evaluated against some external standard of correctness, regardless of the method followed. The outcome without a proper justification in terms of the prescribed method is of little value in law.

The distinction between facts and claims has led to a wide array of logics, frameworks, and theories dealing with *adversarial* settings where different parties disagree on the facts. Many of these efforts where initiated by people with experience in Computer Science & Law (cf. [236]). A medical KBS will for instance typically never even distinguish between facts and the claims made by the user(s) of the system, and is not build to handle logical conflicts arising from contradictory claims relating to the same facts. We might refer to this as the presumption of *cooperativeness*. In LKBS application areas questionable and contradictory claims about the facts are commonplace.

A legal knowledge representation makes well-motivated representation choices with respect to these topics. In the next section we focus on the nature of these representation choices.

2.2. Knowledge Representation

Legal knowledge engineering aims to express legal knowledge in a language that can be used for communicating and storing what we know, and for making knowledge productive in the form of LKBS. If we express knowledge in such a language we call the activity *knowledge representation*. If a language was designed for representing knowledge we call that language a knowledge representation language. The result of the knowledge representation activity is a knowledge representation or *knowledge base*, depending on context.

Knowledge representation languages can be considered as a special kind of language specifically designed to express knowledge. Some knowledge representations are so frequently used that almost no one recognizes them as being a knowledge representation. Musicians for instance use a knowledge representation language that is centuries old and allows musicians to express and exchange ideas. Mathematicians, physicians, chemists, pharmacologists and others have also introduced a plethora of knowledge

duction of new component descriptions in a parametric design system representations. The reader will recognize this formula from his or her high school period:

$$F = \frac{\delta p}{\delta t}$$

Momentum p measures the motion content of an object, and is the product of an object's mass and velocity. Momentum doubles, for example, when velocity doubles. Momentum and force are related by the fact that force F is the rate at which momentum changes with respect to time. If m is constant – which is often the case in application scenarios – then the following holds:

$$F = \frac{\delta p}{\delta t} = m\frac{\delta v}{\delta t} = ma$$

The genius of great scientists such as Newton is not in determining the mathematical constraints that hold between some existing F and p, but in the description of concrete real world problems in terms of newly invented abstract concepts such as force and momentum – simplifying calculation in the process. Expressions in terms of p and F are meaningful when the reader understands how to recognize what things in the domain of interest these concepts apply to, and how to measure mass m, time t, and velocity v.

The same thing applies in the legal field: legal concepts must eventually be defined in terms of concepts that can be operationalized in the domain of interest, by the people who should understand and apply the law.

A *knowledge representation language* is a language designed for the purpose of knowledge representation. Not all data structures are equal from a reasoning perspective: Representing the knowledge in one way may make the solution simple, while an unfortunate choice of representation may make the solution difficult or obscure.

Long division is for instance much simpler with Arabic numerals than with Latin numerals.

Copernicus' claim to fame is in showing that assuming that the sun does not move is a whole lot easier than assuming that the earth doesn't move, if one wants to predict the movements of celestial bodies. Interestingly, both assumptions are clearly untrue: both sun and earth move relative to the rest of the universe.

Most practical physics problems are easier to solve with Newton's model of physical reality than with Einstein's, even though Einstein's is considered a better fit to reality in marginal cases. Newton's theory is obsolete as an explanation of how the world works, but it still *works* as an instrument.

Different knowledge representation languages are designed for different domains of interest. The field of legal knowledge engineering designs its own languages. The reason for having so many different representation languages is that these languages are tailored for expressing specific relevant aspects in the domain. This special focus gives them their expressive power and helps the knowledge engineer in focusing attention to the right things.

This book limits its attention to knowledge representation used by KBS. The whole of knowledge representations used by some KBS is its *knowledge base*.

Most knowledge representation languages are *logics*. Logical systems are formalizations of the proper methods of reasoning and logical truths are those demonstrable by these correct methods. In this sense logics are a special kind of prescriptive theories of proper reasoning. The key notion in logic is the notion of a valid argument. An argument is valid if and only if, if its premises are true, then its conclusion is true. An argument is sound if and only if it is valid and it has all true premises. The logic cannot tell us whether the premises are true: we are the judges of that.

An automated reasoning method for deciding arguments is *adequate* if it decides that an argument is valid if and only if the argument is valid. It is *complete* if it is adequate for all arguments that the language can express. It is *decidable* if it is possible to decide on the validity of any argument in a finite number of well-defined steps, and *tractable* if a computer can produce such a decision in a reasonable time frame using the reasoning method.

The *expressiveness* of language is the extent to which the language allows us to represent the knowledge we want to express. Knowledge representation languages usually differ in expressiveness, and are geared to some type of problem. Knowledge representation languages do not always differ in expressiveness, even if they are designed for different purposes. Logicians are often of the opinion that two knowledge representation languages that are equally expressive from the logical point of view, are really just the same language written in another way. For knowledge engineers this is not the case: languages designed to be used for representing different things cannot be considered equally, more, or less expressive relative to each other: ordering them on the same expressiveness dimension is a non sequitur.

The claim of logical reasoning is that, whatever truth may be, it provides methods to make sure that conclusions are as true as the premises, if the axioms are true. A logic is *monotonic* if new facts cannot invalidate old conclusions, i.e. change the truth value of a proposition. Monotonic reasoning does not allow for exceptions. If something is believed to be true, we will continue to believe it is true regardless of any new information we gather about the world.

Monotonicity and tractability are very appealing properties for a knowledge representation language to have. The great advantage of monotonic reasoning is that the order in which inferences are carried out does not affect the conclusion of reasoning. In other words, we do not have to worry about ordering effects in the knowledge representation language, the order in which knowledge representations are added together in a knowledge base, or the order in which facts and claims are discovered or supplied. The property of monotonicity, in combination with tractability, has special significance in the context of reusability (see section 2.2.4) and the Semantic Web (see section 3.2). Monotonicity is a good *design principle*, because it helps to decompose systems into independent components, even if it is a bad model of how human beings solve problems.

In the legal field it is has become common to use nonmonotonic logics, because *isomorphism* between the defeasible reasoning policies expressed in the sources of law and the knowledge representation appears to demand it, and because of the essentially adversarial nature of legal argumentation. In section 2.5 I will explain part of the case for defeasibility in law.

Isomorphism can be characterized in different ways, but the most obvious one is that a human being should be able to understand the relation between the knowledge representation and a source of knowledge being represented. Isomorphism does not just apply to for instance the mapping between legal knowledge sources – generally writings – and knowledge items, but also for instance to justification: the human observer should be able to understand how the system reached the conclusion it reached.

Isomorphism as used in legal knowledge engineering is suggestive of the more precise term used in mathematics and the natural sciences. Isomorphic structures are *structurally* identical at a certain chosen level of granularity. This level of granularity may in the case of written sources be the word, the sentence fragment, the sentence, the complete or article, etc.

Isomorphism between the source of law and its knowledge representation (introduced in this sense in [17]) is an objective of this book. Isomorphism with argumentation and justification is a more contentious issue, as section 2.5 explains.

Although most knowledge representation languages are logical languages, their properties cannot be fully reduced to the properties of the associated logic. The language comes with *ontological commitments*, as explained in the next section.

This book does not propose or recommend a specific logical language. As stated in chapter 1, use of the OWL DL language is a design requirement for this project, but this book is about representation choices and their practical consequences for the management of complex, version-managed, and distributed knowledge bases.

2.2.1. Knowledge Representation and Meaning

Reasoning is a process that goes on inside the mind, while most of the things we wish to reason about exist only in the outside world. The knowledge representation is a stand-in for the things that exist in the external world. Operations on and with representations substitute for operations on the real thing, i.e. substitute for direct interaction with the world. Reasoning itself is in part a surrogate for action in the world.

Viewing representations as *surrogates* leads naturally to two important questions.

The first question about any surrogate is its intended identity: what is it a surrogate for? The correspondence between surrogate and the real world referent is a fundamental component of the meaning of the knowledge representation, also sometimes called the *extension* or *extensional semantics*, not to be confused with the operational or denotational semantics of the underlying logic.

The second question is how close is the surrogate to the real thing? This is the *fidelity* of the knowledge representation. Perfect fidelity is in general impossible, because any thing other than the thing itself is necessarily different from the thing itself. Fidelity can however always be increased by investing more effort in the representation.

Note that the identity and fidelity question apply to the things we are reasoning about, while the isomorphism question addressed before applies to the information we choose to use as a source of knowledge.

Knowledge representation serves equally well for intangible, *mental things* such as thoughts, moral debts, and plans as it does for tangible, *physical things* – whose existence can be experienced through the senses – such as gear wheels, centers of gravity, or clouds, allowing humans and KBS to reason about both types of things, and their interaction. Somewhat paradoxically it is even perfectly normal to use something to represent nothing[3].

Abstract things, which should not be confused with mental ones, can of course exist inside the KBS with *perfect fidelity*: Mathematical sets and graphs, for example, can be captured exactly, precisely because they are abstract objects. Abstract objects are objects that exist only because someone declared them: it is in principle possible to state the necessary and sufficient conditions of abstract terms, although one may choose not to do so. There is in fact no appreciable difference between the thing, for instance the set {`1, 2, 3`}, and its representation {`1, 2, 3`}, and no difference between the extension of set theory and its denotational semantics. Something is abstract by virtue of having no extensional meaning different besides its denotational semantics.

One could argue that denotational semantics is a kind of intermediate form between the messy outside world and the mind. Since almost any real world reasoning task will encounter the need to reason about things outside as well as abstract things, imperfect surrogates are however inevitable.

The distinction between mental and abstract things is described in more detail by us in for instance [153, 154]. The notion of the mental thing is borrowed from Dennet's *intentional stance* in [95]: the world of mental things is an alternative route to explaining physical events, applied to the behaviour of human beings and, to a lesser extent, animals as actors. The mental category subsumes epistemic and intentional things: epistemic things are mental things that have to do with (acquiring) knowledge. Intentional things only exist because we intend them. It is of course possible that one represents one's own mental state, and (at least) a KBS can do so with perfect fidelity. Whether humans are good at uncovering their own mental reality through *introspection* is a point of contention (cf. for instance [191, 257]).

The knowledge representation itself, if one for the moment forgets that it is *about* something, behaves like an abstract object, and the properties of abstract objects can be used to predict the unknown properties of the referent. It is for instance possible to use a theoretical disk, with an evenly distributed weight, to predict the center of gravity of a porcelain plate. The real center of gravity of the porcelain plate is however observable: it is the place on which it can be balanced. The belief that the center of gravity is equal to the center of gravity of the idealized object is a *hypothesis*, that can be falsified by observation. This hypothesis is undoubtedly useful, because porcelain plates are very breakable: it is a good idea to first predict where it is before one tries balancing it. This predictive quality of the knowledge representation, through the denotational or operational semantics of the representation, can be called it *intension*, or *intensional semantics*.

One of the attractive features of the law is its similarity to knowledge representation. There is no issue of fidelity with respect to outside referents as long as the processes of applying the law are not taken into account. The fidelity question is reduced to the

[3]In OWL the distinction between `owl:thing` and `owl:nothing` is in fact the most fundamental ontological distinction made.

fidelity of the copy of the sources of law into the knowledge representation language, which is unfortunately generally less expressive than the language the legislator uses. In reality things are of course slightly more complicated.

Law deals with the same fidelity issue as knowledge representation: the correspondence between legally declared reality and social reality can break down. Generally speaking the legislator explicitly decides to leave the interpretation of certain terms used in legislation open to interpretation. In this case legal theory often speaks of *open textured concepts* (cf. [237]). In its broadest sense all physical and mental things described in legislation, its regulated domain, are open textured. In a more limited sense, a concept is called open-textured when the legislator is being *intentionally* vague. Open-texture is for instance used if the legislator believes that it is not practical to try to narrowly define terms because the regulated domain of interest[4] changes too often in unpredictable directions or is hard to pin down in general (cf. [284, 237]). In other cases, the use of open-textured concepts is a compromise, the consequence of disagreement in group decision making.

Two important consequences follow from the inevitability of imperfect surrogates. One consequence is that in describing the world, we must inevitably resort to untruth, by omitting details at least. At a minimum we must omit some of the effectively limitless complexity of the outside world. In addition, our descriptions may introduce artifacts not present in the world.

The most important consequence is that reasoning about the world must eventually reach conclusions that are incorrect, independent of the quality of the reasoning process used, and independent of the quality of the representation used. Logically sound reasoning cannot save us: If the world model is somehow wrong (and it must be) some conclusions will be incorrect, no matter how carefully drawn. A better representation cannot save us: all representations are imperfect and any imperfection can be a source of error.

If all representations are imperfect approximations to reality, each approximation attending to some things and ignoring others, then in selecting any representation we are in the very same act making a set of decisions about how and what to see in the world. Selecting a representation means making a set of *ontological commitments*.

These commitments and their selectiveness are not some unfortunate and incidental side effect of a representation choice. A knowledge representation *is* the realization of a set of ontological commitments. It is unavoidably so because of perfect fidelity is impossible. It is usefully so because judicious selection of commitments provides the opportunity to focus attention on relevant aspects of the world, while ignoring the infinite details. Just like the court, the knowledge representation only admits relevant evidence.

Part of what makes a language a representation language is that it carries extensional *meaning* or extensional semantics, i.e. there is a correspondence between its constructs and things in the external world. That correspondence in turn carries with it *constraints*. While every representation must be implemented in the KBS by some

[4]Note that I use *domain* in two senses throughout this book: the domain being represented by the knowledge representation and the domain being regulated, and therefore represented, by legislation. The domain represented by the knowledge representation includes the relevant legislation.

data structure, the representational property is in the correspondence to something in the world and in the constraint that correspondence imposes.

If you say in the ontology specification of your domain that all cars are vehicles, this is an ontological commitment, meaning that the system may derive that something is a vehicle because it is a car but for instance not that something is a car because it is a vehicle. This is a constraint on the inferences warranted by the knowledge representation. A system that ignores these constraints on the knowledge representation ignores the semantics of the knowledge representation.

In Davis, Shrobe, and Szolovitz (viz. [88]) knowledge representation was characterized as follows:

1. A knowledge representation is a surrogate for the thing itself, used to enable an entity to determine consequences by thinking rather than acting.
2. It is a set of ontological commitments, i.e. an answer to the question: In what terms should I think about the world?
3. It is a fragmentary theory of intelligent reasoning, expressed in terms of three components: (i) the representation's fundamental conception of intelligent reasoning; (ii) the set of inferences the representation sanctions; and (iii) the set of inferences it recommends.
4. It is a medium for pragmatically efficient computation; It organizes information so as to facilitate making the recommended inferences.
5. It is a medium of human expression, i.e. a language in which we say things about the world.

These characterizations position knowledge representation against several dimensions on which the representation can score better or worse; Often trade-offs are involved. Davis, Shrobe, and Szolovitz use the term inference in a generic sense, to mean any way to generate new expressions – or beliefs when talking about a human being – from old ones. Problem solving can somewhat trivially be characterized as making inferences recommended and warranted by the knowledge representation until the the problem is solved. The quality of the knowledge representation is amongst others determined by whether a solution is reached at all, and how many inferences are needed to do it.

Additional standards for determining the quality of a knowledge representation can be for instance found in (cf. [120]).

When considered as a reusable resource for production, a good knowledge representation:

1. will allow the reuse of the same knowledge for different types of problems and with different reasoning methods,
2. is easy to maintain when the outside world changes, and
3. facilitates isomorphism between representation and consulted knowledge sources.

Finally, it should be understood that a KBS, and the knowledge base it depends on, embodies a certain *problem solving competence*, i.e. the competence to solve a certain type of problems (cf. generally [187]). A KBS that was built to, for instance, determine income taxes, need not be competent to determine income. The mere fact that it knows about *income* – i.e. there are descriptions of the domain in its knowledge base involving *income* – does not mean that it *knows enough* about income to help determine it. This distinction is central to the design of KBS: the KBS may embody an ontological commitment to a certain conception of income, but at the same time may lack operational knowledge about determining it.

2.2.2. Limitations of Symbolic Knowledge Representation

A popular objection to knowledge representation – at least of the symbolic or "logical" approach – holds that knowledge representation is not a viable model of human intelligent behaviour at all. Proponents of this thesis deny that the brain can be conceived of as a linear processing machine using "knowledge" stored in a memory, and advocates alternative processing models, for instance neural networks, that offer a better analogy to what we know of how brains really work. This discussion of course relates more to philosophy of artificial intelligence, something the field of legal knowledge engineering has never really been concerned with[5], than to the utility of knowledge engineering in the design of decision support systems and knowledge-based systems.

The neural network approach has not gained currency in the legal field. One of the problematic aspects of this approach is for instance articulated by Kowalczyk (cf. [177]): the neural network approach is fundamentally deficient for many uses because of its inability to justify the results reached. If there is any similarity between neural networks and brains, it is definitely lost on *minds*. The law is not just about the decision, but also about the correct justification of the decision, based on explicit knowledge of the legal sources to be applied, and the features that are and are not to be taken into account. Neural networks are not very good at justifying themselves, and cannot distinguish between spurious and relevant correlations between input and output. They are not transparent, and engineers like transparent systems, not just in law. This limits their usefulness to solving problems that humans do not understand well (such as detecting sea mines in sonar data, "electronic noses", and data mining).

Some have since demonstrated generation of justifications from neural networks (e.g. cf. [128, 16]), and made interesting observations about the conditions for successful application of neural networks and statistical methods to law (cf. [161, 16]). Kowalczyk's point is however not really about the abilities of neural networks, but about a design philosophy of "training" a general-purpose architecture with a set of cases from the past to achieve competence in producing the same output as the legal system does, even though explicit jurisprudential theories aiming at the same thing exist. The inputs and outputs of the system have to be so carefully engineered to be justifiable for application in law that it matter-of-factly becomes symbolic knowledge representation, or as John Sowa put it, on his first encounter with a call for papers on *feature engineering for machine learning*[6]:

> For years, the people who were working with neural nets and other learning systems were hyping the claim that they did away with "knowledge engineers" because they didn't need rules, frames, formulas, etc. But now they discovered that these learning systems don't really learn all by themselves – somebody has to analyze the subject domain and select

[5]A variation on a well known epigram: The philosophy of artificial intelligence is as relevant to artificial intelligence as ornithology is to birds.

[6]Email archive: `http://suo.ieee.org/email/msg12963.html`

some set of "features" for the learning systems to work with. Since they had already got rid of the knowledge engineers, they now discovered a totally new breed of "feature engineers".

Law and knowledge engineering both operate with "folk psychology" cognitive models of human knowledge, intentions, and behaviour that are incorrect according to postmodern critics armed with scientific knowledge from psychology and neurology (viz. [6]). This resort to "folk psychology" is for instance very evident in the ascription of intent to suspects of crimes, based on their observed behaviour, and assumptions about what they must have foreseen as the consequences of their actions, given what they must know. In chapter 4 I will touch on this point again and discuss its relevance – or rather its lack of relevance – for legal knowledge engineering. See for instance [277] for a discussion, in the context of a critical evaluation of the role of expert witnesses in court, of the differences between effective and justifiable decision making in law and empirical science as a quest for the truth.

The law applies a set of assumptions about human behaviour very similar to what knowledge engineer Allen Newell (cf. [209]) called the *knowledge level hypothesis*. The knowledge level permits predicting and understanding behaviour without having an accurate operational model of the processing that is actually being done by the agent. Observers treat an intelligent agent as a system at the knowledge level: they ascribe knowledge and goals, and a variety of other *mental things*, to it. According to Newell knowledge can be defined as "whatever can be ascribed to an agent such that its behaviour can be computed according to the principle of rationality," noting that the principle of rationality means that if an agent has knowledge that one of its possible actions will lead to one of its goals, then the agent will take that action.

Newell's hypothesis does not exclude the possibility that "knowledge" is best captured in some kind of neural networks, but by stressing that knowledge is something that is "ascribed" by the outside observer (i.e. it is an *externalist* and *doxastic* epistemological theory of behaviour) it clearly leads to the conclusion that descriptive fidelity in a neurological sense is much less important than more standard desirable properties of a scientific theory such as:

1. its effectiveness as a predictor;
2. its value as a justification; and the
3. ease of communication of the model.

The third and second functions are clearly better served by a symbolic knowledge representation. Also, by assuming the "principle of rationality" as a constant in the prediction of behaviour, it must necessarily follow that the fact that agents do different things in the same circumstances must be explained in terms of different knowledge bases. The claims the "principle of rationality" makes about human behaviour are in effect quite modest: it makes no assumptions about the processing equipment, and even declares itself invulnerable to falsification by introspective accounts.

In legal knowledge engineering, more specific charges have been made against the logical approach. It has often been claimed that legal reasoning – mostly by theorists from outside the field of legal knowledge engineering – cannot be captured by a logic. These criticisms boil down to three types.

The first one is a mere restatement of what we already noted before: that the real world cannot be captured with perfect fidelity. Any knowledge representation, including legal knowledge representation, will at some point sanction inferences that do not lead to true conclusions. When you are making a new design plan for your garden, the theory that the earth is flat is perfectly viable. But when you are traveling to another continent by ship, the theory breaks down, and has to be replaced by the theory that the earth is round. This is also true for an LKBS (see for instance [201]): even if it works in most cases, at some point the LKBS will make a decision that perhaps a human would not have made because the human is grounded in reality in a way that the LKBS is not.

This criticism is acknowledged in knowledge engineering, and is the main reason for the interest in making explicit the ontological commitments a knowledge-based system is based on. The decision of the LKBS is as much a prediction as for instance a lawyer's assessment of what a court would decide. Another consequence is obviously that the responsibility for maintenance of the LKBS and the liability for the decisions it takes should be critically evaluated when a LKBS is introduced.

The issue of *falsifiability* of the conclusions of the LKBS may also be connected to defeasible reasoning. One should however keep in mind that also an LKBS based on a knowledge representation language that allows for defeasible reasoning will at some point sanction inferences that do not lead to true conclusions. The issue here is one of isomorphism rather than of fidelity of representation.

The second criticism is more specific to legal knowledge representation: legal knowledge representation languages usually prominently represent norms. The usual presentations of deontic logic, the logic of norms, whether axiomatic or denotational, treat *norms* as logical sentences that bear truth-values. A fundamental problem of deontic logics, is to reconstruct it in accordance with the philosophical position that norms prescribe rather than describe, and are neither true nor false. In other words, norms do not have truth values. This observation is generally attributed to mathematician and philosopher Walter Dubislav (viz. [103]).

This criticism may be waved away by noting that concepts, triangles, planets, and sets do not have truth values either. Only the statements *about* them do, by affirming or violating the ontological commitments we made regarding these objects.

One may for instance say that such and such a norm is (or is not) part of (or implied by, consistent with, etc.) such and such a normative code. For example one may say that the British driving code requires vehicles, in normal circumstances, to be driven on the left hand side of roads, while continental European codes requires them to drive on the right side. These statements report on the presence or absence of a norm with a certain effect in a given normative system, and are often taken to be the fundamental representation category in deontic logics. To mark the difference, they are sometimes called *normative propositions* or *normative statements*.

As suggested for example by Bengt Hansson in [136], it is enough to take one's preferred possible worlds semantics for deontic logic and reinterpret its deontic components as relativized to a given normative system to solve the problem. One may also state that such and such situation or behaviour is a violation of a norm, or that this norm is in conflict with that norm, or that some norm directs us to choose this alternative over that alternative in a certain situation. In each case we make a statement about the norm, instead of representing the norm. What we however cannot do is equate some proposition or logical sentence to a norm.

As Makinson notes in [195], most people in AI & Law do accept the theoretical problem, but in practice work goes on as if the distinction had never been heard of:

> In axiomatic presentations of systems of deontic propositional logic, the truth-functional connectives "and", "or" and most spectacularly "not" are routinely applied to items construed as norms, forming compound norms out of elementary ones. But as [103, 163] already observed, if norms lack truth-values then it is not clear what could be meant by such compounding. For example, the negation of a declarative statement is understood to be true if and only if the item negated is false, and false if and only if the latter is true; but we cannot meaningfully formulate such a rule for norms.

He notes an unfortunate consequence of lack of attention for the normative system as a whole, which leads us back to the earlier point about fidelity: the major problem that plagues legal knowledge engineering is that legal knowledge representation fails to *scale* because we cannot reliably break it down into reliable components. Legal knowledge representation usually works for single knowledge-based systems tailored towards solving a specific type of problem, but breaks down if we put the knowledge bases of two knowledge-based systems together in one logical reasoning system. One of the reasons is the lack of attention for how and when normative statements are composed into larger composite normative statements, if it cannot be reliably done with truth-functional connectives, more specifically an implicit conjunction of normative statements.

In this book two design approaches are used to address this problem. The first one is to consider normative statements as statements *about* a norm; The norm is an object in the knowledge domain, and *not* a logical sentence. The second one is a skeptical attitude to composition of norms into normative systems: the violation of a norm in principle stands by itself, unless our knowledge tells us otherwise.

A third criticism is for instance voiced by Moles in [201]: Legal knowledge engineering focuses too intensively on representing the sources of law instead of the meaning of law. This criticism is somewhat addressed in section 3.5.1. The reader however has to keep in mind that this work is indeed only about sources of law.

2.2.3. Problem Solving and Problem Solving Competence

The engineers of a knowledge based system try to provide a specific intended functionality in an efficient manner. They take into account resource limitations on for instance time and memory of the computer, the availability of information and the cost of collecting information, the required performance of the KBS, and make *assumptions* about the institutional context of the problem solving task, about the availability, quality, and completeness of certain kinds of information, about the knowledge and needs of the users of the system, etc.

This focus on functionality and efficiency is not only necessary because knowledge representation is a resource consuming task, but also because otherwise the design process would be underspecified, missing an explicit goal, and there would be no way to determine that the result is acceptable.

Knowledge representation is usually only suitable for a specific *problem type* that the knowledge engineers had in mind. Each problem type supplies its own view of what is important to attend to, and each suggests, conversely, that anything not easily seen in those terms may be ignored.

A KBS designed for buying and selling second-hand cars will generally speaking not be very helpful for someone designing a knowledge-based system for diagnosing and repairing cars or driving cars, even though all three contain descriptions of cars: the required information about cars is very different (cf. [153] for this example).

In terms of the porcelain plate example used earlier: A KBS that helps us to determine the center of gravity of porcelain plates usually does not explicitly communicate its theory on idealized porcelain plates (i.e. as disks with an evenly distributed weight), and has no knowledge at all that would help it plan the production of a porcelain plate.

A *problem solving method* (cf. [62]) supplies a coherent set of assumptions about what to attend to and what to ignore to achieve a certain *epistemic competence* or *problem solving competence*, i.e. to be able to solve a type of problems. Fensel and Straatman in [109] for instance characterize the problem solving method by a functional specification, an operational specification, a cost description, and a set of assumptions about the domain and the available domain knowledge that must be true for the problem-solving method to realize its functionality.

Note that more often than not we make assumptions that are known in advance to be not true, but true often enough to be practical, in order to reach conclusions. This is for instance often the case with the *closed world assumption* (cf. [251, 205]), which states that we know all that is relevant to know, or *frame assumptions* (cf. [86]), which state that no unforeseen outside processes affect the state of some relevant object or situation, and which are often used in planning problems. An assumption specific to legal planning is for instance that other agents do not generally violate the law. This assumption makes for instance driving in busy traffic workable: we do not have to consider all possible ways in which other traffic participants could decide to violate traffic law.

The use of unverifiable assumptions, certainly if they are left implicit, means that a knowledge representation suited for one type of problem – for instance planning – is not automatically suitable for another type of problem – for instance diagnosis, not even if we add relevant knowledge to it.

2.2.4. Reusability and Knowledge Components

There are good reasons to expect that knowledge representation that is neutral with respect to problem solving competence is in principle impossible (cf. generally [76]). For one thing, such a design problem would be underspecified as pointed out before. Secondly, one could argue that knowledge, as understood in the Newellian sense ([209]), simply cannot be disconnected from the problem solving competence it helps to realize: we call some data structures knowledge representation by virtue of the problem solving competence they help to realize when properly understood by a rational agent. This is a necessary consequence of the external position we take towards intelligent agents when we ascribe knowledge to them: the agent first has to show it is competent to solve a certain type of problem before it can be said to be knowledgeable.

The degree of reusability of the knowledge base as a whole has already been fixed by the designers of the system: the knowledge base is reusable only relative to that part of the knowledge base that varies per problem solving session. We can call this variant part *the case*, and the rest the *invariant knowledge base*, or – if no ambiguity exists – simply the knowledge base.

In the case of LKBS, the case usually does not *include* the sources of law, even though these are subject to change. Since the sources of law change, in some domains regularly, and moreover usually change outside of the control of the users of the LKBS, the LKBS maintenance task is both labour intensive and hard to schedule resources for.

Regular maintenance imposes some constraints on the design of LKBS. Firstly, it must be easy to identify *where* the LKBS must be changed in response to a change in the sources of law. This is the problem of *isomorphism* between source of law and representation. Secondly, we want to change them in only one place. Even better is if we can make a single change to *all* deployed LKBS that use the same source of law.

We therefore compose the knowledge base in as far as possible from reusable *knowledge components*, that are maintained at a single point. This is the notion of *knowledge syndication*. The component in effect becomes a generic knowledge base interrogated by multiple specific purpose LKBS, which creates a coordination problem: we must make the change in such a way that all user KBS keep using it correctly.

Focus has changed over time from knowledge as the attainment of truth to knowledge as productive factor (cf. [191]), to be realized as – one-shot or reusable – plans or – reusable – designs for machines[7]. The first perspective is often called the (traditional) *philosophical* perspective, while the second can be characterized as the *economic*, or more generally the *social sciences* perspective. It is this shift that introduced the concept of *tacit knowledge*, the knowledge that bridges the epistemological gap between explicit "theoretical knowledge" that explains certain well-understood aspects of a situation and *skill* on the work floor (cf. for instance [191]), which involves the less well-understood capacity of intelligent *situated action*. The common (mis)conception of tacit knowledge as knowledge that is hard to make explicit – alluded to in section 2.1.1 – is only secondary to this first meaning: if tacit knowledge is a stop-gap concept, referring to those aspects that are not well-understood, it ceases to be applicable as soon as the situation is well-understood.

The study of reusable knowledge components is in essence about knowledge as a factor in the production of yet more knowledge, the next shift in focus as evidenced by the *Semantic Web* vision, discussed in section 3.2. Interestingly, this new shift has led to renewed interest in the philosophical perspective.

While it may be impossible to make a representation problem independent, we can restate the issue to enumerating and describing types of problems and problem

[7][191] uses classical economic terminology: capital, labour, plans, and machines. Plans and machines in this context evoke business processes and conveyor belt-based assembly-lines. Tool, device, or artifact are other possible terms for what I mean here: I interpret this as anything manmade that requires non-trivial knowledge to make and has some *function* or purpose, ranging from computers to hammers and violins.

solving methods, and understanding the systematic relations between them and their components, in the hope that this gives us clear ideas about what constitutes a reusable knowledge component. Attempts to make such an inventory, for knowledge engineering in general, are found in for instance [59, 206]. The other, "philosophical", avenue of exploration, exemplified by top ontologies (viz. [186, 185, 255, 222, 197, 114], simply divorces the search for fundamental truth from problem solving, in the expectation that the truth part is reusable, and is of less direct interest for this discussion.

Breuker in [59] (see also [33, 34]), distinguishes the problem types modeling, planning, design, assignment/scheduling, monitoring, assessment, diagnosis, repair, and remedy, and postulates a number of functional dependencies between them, observing that the output of one problem type is often the input of another one. The problem types impose different epistemological perspectives on the same domain (knowledge).

Design is for instance characterized as the composition of building blocks with a certain functional interface specification into a system specification meeting certain constraints and requirements. Diagnosis takes observations of the behaviour of such a system, that is observed to deviate from its specification, and determines which building block(s) does not behave according to its specification.

Breuker's list of problem types clearly bears the marks of the focus on knowledge as a productive factor, a resource: design, diagnosis, and repair revolve around machines that are supposed to have some function or purpose, while planning, scheduling, and assessment revolve around a plan that attains some goal.

Complementary to the problem types approach are attempts to automatically design problem solving methods from components, as proposed in for instance [264].

The problem types are surprisingly versatile in their mapping to domain knowledge: design applies to such varied things as violins, organizations, computer programs, mathematical proofs, business processes, etc. The mapping is of course not completely free of constraints: you cannot plan a train, but you can plan to take it, or plan its construction.

In the context of legal knowledge engineering, planning (subject to the constraints of the law) and assessment (of cases) appear to be the legal problem types par excellence (as noted in for instance [265, 275]). This distinction is in itself fruitful, as it for instance helps us to realize that the ex ante role of the norm in planning is very different from its ex post role in recognition and assessment after the fact (cf. section 4.7). This type of differentiation of role in problem solving is called the *epistemic role*, or simply the knowledge role, of knowledge items.

It is however also easy to come up with typical design or diagnosis scenarios in the law. In my own work reported in for instance [284], the domain of interest is firstly ship design, subject to legal constraints and requirements, and only secondarily ship handling. One could also consider typical form filling systems (electronic tax filing, legal drafting) to be typical parametric design systems.

Deciding which driver(s) caused a chain collision can for instance be conceived of as a form of diagnosis, if we accept the machine metaphor for traffic flow, where we apply typical diagnostic assumptions such as the *single fault assumption*: i.e. we first consider all hypotheses assuming some non-standard behaviour of one component, while all others are assumed to behave according to specification, and see whether one of these hypotheses explains the evidence. Only when this doesn't yield a conclusion we proceed to the assumption that two components failed, etc. The drivers are the metaphorical components.

The conclusion must be that the field of law is *not* strictly characterized by attention to specific types of problems: how it conceives of legal problems is determined by the nature of the problem and the character of our domain knowledge. We cannot establish a systematic relationship between the *domain* of law and specific problem types, and there is no apparent reason to expect that there is such a relationship.

Separation of reusable knowledge about the domain and reusable knowledge about problem solving, or ontology and epistemology, as characterized in [26, 153, 154], and in this book, is however the first step to bridging the *epistemological gap* and designing reusable knowledge components. This proposed solution comes with a cost: reusing existing components, and designing knowledge components for reuse, is certainly initially more costly than designing a solution for a single problem, and this increased cost must be justified in terms of reusability, maintenance, or scalability.

This distinction between knowledge of the domain and knowledge of problem solving has gained some currency in knowledge engineering in general, but the distinction between the ontological and epistemological aspects of knowledge as a productive factor is not often explicitly made in legal knowledge engineering[8], where the need for separating knowledge by epistemic role is seldom felt. An analysis of legal reasoning in the context of this distinction is therefore in itself worthwhile. Moreover, if law is not characterized by its own problem types, it should be possible to deal with legal reasoning using standard knowledge engineering techniques.

Another important methodological admonition that can be gleaned from this section and the preceding one is that a KBS should explicitly characterize the problem solving competence it is intended to realize in its knowledge base, so that we can at least assess the value of the knowledge it represents as a productive factor beyond the immediate context of use for which it was designed.

Moreover we should distinguish between the porcelain plate and the theoretical disk of section 2.2.3: this distinction involves the concept of abstraction.

2.2.5. Ontology and Epistemology

When considering knowledge representation, we can distinguish issues of ontology – about the *object* of our knowledge – and issues of *epistemology* – about the processes by which knowledge is attained. This is not the traditional philosophical distinction. In fact, the sense in which we use epistemology is closer to what cognitive scientist Christopher Longuet-Higgins[9] named *epistemics* – to distinguish it from epistemology – instead of epistemology proper. The distinction is close to the language level distinction between the extension of the sentence, and its semantics proper, but even here we have to distinguish between the semantics brought in by the ontological commitment and the semantics brought in by what we try to do.

[8][202] is an exception, as well as the work of [153, 154] mentioned earlier.

[9]The first chairman of a "school of epistemics".

In this book I will often use the adjective *epistemic* when discussing particulars (e.g. a hypothesis is an epistemic object), and *epistemological* when discussing universals (e.g. hypotheses have a lower status relative to truth than facts). For ontology I can make no such distinction, as *ontic* would be considered a confusing neologism by the reader.

So far I have mostly discussed the semantics of knowledge representation in relation to ontological commitment and fit with an outside world. Epistemology imposes another criterion: to know something is to have acquired the belief in that something through some process by which knowledge is reliably achieved. Note that epistemic reliability cannot be simply reduced to a statistical criterion (see for instance [202]: p. 42–43); A process that more often than not produces a true answer, does not necessarily discriminate correctly, need not be operating properly, etc.

Epistemology and ontology of course cannot strictly speaking be separated: production of knowledge is obviously inextricably bound to ontological commitments.

The distinction is another one: the separation of the conceptualization of the "outside" world the reasoner is acquiring knowledge about, the *knowledge domain*, from the conceptualization of that part of the mental world of the reasoner that plays a role in acquiring knowledge. From an ontological viewpoint there is nothing that relates bin packing and scheduling lectures to classes and time slots: from an epistemological viewpoint both are assignment problems. Epistemological and ontological knowledge are linked by a non-obvious mapping, usually handmade by the knowledge engineer, that, if certain assumptions are met, gives a reasoner that understands it a certain epistemic competence, i.e. problem solving competence, in the domain covered by the ontology.

The distinction between epistemology and ontology in this sense helps reuse by separating two aspects of knowledge that can be reused under different conditions. Epistemological knowledge can be reused in different knowledge domains if we are solving essentially the same type of problem using the same types of knowledge sources, while the same ontology can be reused for problems directly or indirectly involving the same knowledge domain.

For the subject of this book this means that we are interested in ontological knowledge about the world of law, as expressed in the sources of law. Epistemological knowledge plays a role to the extent that the sources of law themselves model that mental world.

2.3. Ontology

In philosophy *ontology* is the study of *the nature of being, reality, and substance.* The *ontology* as it is usually understood in Knowledge Engineering has a more specific meaning: it is the product resulting from the systematic inventory by knowledge engineers of relevant aspects of a certain knowledge domain. This necessary restriction of attention to *relevant* aspects relates to the ontological commitments of Davis et al. (cf. [153, 154, 88, 67]).

Ontology in the knowledge engineering sense is often defined as "a specification of one's conceptualization of a knowledge domain." (cf. [131]). The specification is usually made in a knowledge representation language, but it is important to distinguish between the specification (which is usually called the ontology in practice) and the conceptualization, which embodies the ontological commitments and of which it is a specification.

Secondly, ontology is also often described as a *shared* conceptualization of a domain, which stresses the prescriptive nature of the ontology as a kind of quasi-contract binding the developers of the KBS and its prospective users, and a vocabulary they can use to communicate with each other. The same concept can also be applied to interoperability between systems. The ontology often also functions as a schema or specification against which messages can be validated. Open standards in IT are for instance often said to have a ontology (cf. generally [53, 182, 245] for this purpose.

The ontology specification usually consists of terms T, denoting a set of things to which the term can be applied, to be specified by terminological axioms of the form $T \subseteq C$ (C is a necessary condition for T) or $T \supseteq C$ (C is a sufficient condition for T) specifying them. In rare cases a term can be *defined* – all necessary and sufficient conditions can be supplied – by an axiom $T \equiv C$; This is most common for abstract terms, for reasons noted earlier. The terms *term* and *concept* are used interchangeably for most practical purposes. I generally prefer term for a named concept: T is a term, while $T_1 \cap T_2$, occurring at one of the sides of a terminological axiom, is a concept (the intersection of the extensions of T_1 and T_2) but not a term. The terms together form a terminology or vocabulary, and in combination with the terminological axioms the ontology.

It is important to note that each and every statement in a knowledge representation carries with it ontological commitments, but that only those statements that are considered *terminological* in character are part of an ontology specification proper.

For instance: that a father is the male parent of a child is ontology, and contingencies such as that John is the father of Jane, that a father of a child is usually married to the mother of that child, or that a parent is entitled to some child benefit from the government, are not.

Another example, from legal theory: that legal responsibility is the liability to be subjected to a sanction of the law is ontology of law, but that one must have done something in some sense to be legally responsible for it is not, at least if we follow Lehmann in [184], or originally Hart and Honoré in [141].

Because the description of the ontological commitments is stored in a knowledge representation language in the form of terminological axioms, and may become part of the knowledge base of a KBS, there is a lot of confusion between knowledge representation, knowledge bases, and ontologies. It is not uncommon to refer to every knowledge representation, or even the physical embodiment of a knowledge base in the form of a file or database, as an ontology.

This has the unfortunate consequence of bringing all kinds of discussions about knowledge representation issues that have no bearing on ontology into the ontology field. There are in the opinion of the present author no such things as fuzzy, Bayesian, or defeasible ontologies, for instance: this is anathema to the notion of shared vocabulary, and we cannot properly characterize the epistemic competence of a KBS that does not commit to its ontology.

Ontology consists of terminological axioms, and axioms are epistemically speaking self-evident: known to be true by understanding its meaning, its intended fit to reality, without proof or possibility of falsification. Inference on terminology alone strictly speaking doesn't produce knowledge: it merely reformulates what is already believed.

To design ontologies, it is good to follow some simple design principles. Firstly, it is bad practice to try to achieve a given epistemic competence in the ontology by searching for a set of sufficient conditions for key terms. Ontologies and terminological reasoning are only intended to help to:

1. rephrase descriptions of things into other terms, to
2. recognize that one description terminologically subsumes another, or
3. that a description is terminologically incoherent.

Axioms should not be added to the ontology specification for other purposes: these propositions have another epistemic role, and contrary to our absolute commitment to terminological axioms we are willing to retract them if they do not fit reality.

It is good practice to insist that any term used by a KBS should be specified in an ontology: the ontology is the main access point into the knowledge base for determining what a certain term means to the KBS. Note that not only individuals, which are entities that have no instances, but also universals, entities that have instances, can in principle be described by terms (cf. [114]). Most ontologies aim to be ontologies of individuals (but unsuccessfully if we follow the argument in [154]).

It is good practice to see to it that all terms in the ontology are related through terminological axioms with other terms. One should expect to find *coherent* terminologies – islands of terms connected to each other by terminological axioms. Single, unconnected terms identified only by waving in the general direction of some external class of referents are meaningless. Coherent terminologies naturally associate with specific (families of non-terminological) theories of the world. Abstract terminologies should always be associated with some theory: this is the very reason for their existence. There is however no reason to expect that terminologies which happen to concur very regularly in descriptions of domains have an *ontological* connection: the reason for the systematic coincidence may be found in epistemic habits.

An important pragmatic aspect in information exchange between human beings is the aspect of quantity: being informative requires that the information one conveys is *relevant* (cf. [227]). One tries to limit oneself to expressing information that adds to the shared knowledge base. Anything that is obvious from discourse *context* will not be made explicit in written information sources, but should be made explicit in the ontology. Consideration of the relation (and the difference) between ontology and *information* yields some additional pieces of useful advice.

When we base an ontology mostly on written sources, and in legal knowledge engineering we are likely to do that, we need to be aware of "hidden" participants in the context of the utterance. One for instance cannot conceive of a parent without a child (i.e. $\texttt{Parent} \equiv \exists\texttt{child}\top$), but information contexts where only someone's status as parent (without explicit consideration of the child) is relevant, are possible. Establishing someone's status as a parent introduces the existence of at least one child *of* that parent in the discourse context, and one may later in the discourse refer to that child.

One could think of metonymy and synecdoche as overzealous reduction of the quantity of information exchanged. Clearly, *metonymy* and *synecdoche* do not belong in an ontology: one does not allow that a Volkswagen is a type of car instead of a car brand, or that someone pays the taxi instead of its driver. This is a very common oversight, and a major source of variety in ontologies about the same subject.

Very importantly, one should be cautious of allowing *metaphor*, even conventional metaphors (cf. [183]), into the ontology, in particular if they pertain to physical entities. Instead one should ask oneself whether the metaphorical use of language indicates the existence of a useful abstraction, as explained in the next section about granularity.

As pointed out by Lakoff et al. in i.a. [183], terminology for certain basic physical phenomena – for instance physical orientation in space (up, down, etc.) – is readily transferred to other things – physical, mental, social, institutional, abstract – in the form of conventional metaphors.

In the case of mental, social, and institutional things we perhaps cannot avoid this use of conventional metaphor (cf. [225]), as it is arguably (an aspect of) the fundamental mechanism by which we construct these concepts in the first place (according to [183]), and therefore part of their fundamental meaning. The transfer of metaphors at least tells us that entities share something important, but – again – this something may simply be an epistemic habit.

In the case of abstract entities, we could argue that time, orientation, containers, fluids, etc. give us the original archetypes for abstract things such as preorders or sets, which can then be applied to other things. The use of metaphorical language can be considered a hint that things share a conventional abstraction. Contrary to [183] I will make no assumptions about the evolution of conventional metaphors and abstractions.

Given the observations about discourse context and about metaphors, one should be ready to accept that there are things generally referred to by the same name that refer to different concepts, so-called homonyms or polysemes. Polysemes are different, but closely related, senses of a concept. Linguists have a rule of thumb for recognizing them: If a word, that does not appear to be an obvious homonym, seems to exhibit *zeugma* when applied in different sentences, it is likely that the contexts these sentences evoke bring out different, but related, senses of the same word. For instance "the state is free to execute its laws and citizens as it sees fit" or "oh, flowers are as common here, Miss Fairfax, as people are in London.". This rule of thumb is less useful than it may seem, since finding the right sentences is hardly a trivial exercise.

Metaphor transfer is one likely explanation of the common occurrence of polysemes. Polysemy and homonymy is generally solved in ontology specifications by declaring such different concepts in different "namespaces", e.g. `law:norm` vs. `measure:norm` vs. `social:norm`, or `finance:check` vs. `chess:check`.

In legal knowledge engineering, ontologies are the primary interface between knowledge engineering and legal theory. This work can be characterized as *computational legal theory*, which studies the same subjects as legal theory but with application in knowledge engineering and knowledge management in mind.

In this book I will roughly follow the distinction made in i.a. [153, 154] between physical, mental, and abstract things. Social things derive from the recognition that you and others have a mental world, the possibility of exchange of information, and the assumption that objects can be shared between mental worlds. Special attention goes to intentional and epistemic things, which are mental things, and institutional things, which are intentional and social in nature. This distinction between a physical, mental, and abstract world is a form of *ontological stratification*, as explained in the next section.

When one speaks of an *ontology of law*, this may mean that it is a specification of specifically legal concepts, or a specification of the relevant aspects of some knowledge domain from the perspective of some type of legal problem solving, or from legal theory. Section 4.2 takes a position on what things are legal or "legally relevant" (to follow a distinction made by Mommers in [202]): legal things are created by the law, while legally relevant things are recognized by it.

Ontologies of law usually contain many, or even mostly, "legally relevant" common sense terms (e.g. [65, 153]): the ontology describes a shared vocabulary, but it is definitely not intended as, or limited to, a jargon dictionary for the legal domain. Ontology of law is more properly characterized as the study of law's conventional abstractions.

2.3.1. Granularity and Abstraction

The complete ontology of a KBS need not be a particularly coherent description of the world. Copernicus' elliptical orbits and Ptolemy's epicycles can harmoniously co-exist, just like the abstraction of a road to a line for navigation, to a surface for driving, and to a volume for repairing potholes.

There is however a difference between the theories of Ptolemy and Copernicus and the three alternative models of the road: the first pair are simply rival explanations, and the second has been judged clearly superior to the first by history, while the three models of the road are all useful perspectives at different levels of granularity depending on what we try to do. Earlier I pointed out that the round earth and the flat earth, and Newton and Einstein's theories also happily co-exist because they have different strengths and weaknesses.

The notion of levels of *granularity* used in description depending on what we are trying to do appears in different guises in different contexts. In knowledge engineering we find for instance Hobbs' work in i.a. [150, 149], who introduced the road example. In science we find for instance the excellent discussion of [84] on *reductionism* and how complexity at one level of granularity of description appears to magically collapse into simplicity at another granularity level of description, for no other apparent reason than that it is a better description of our problem. Dennett's (cf. [95]) intentional, design, and physical stance are another example. Dennett not only discusses the phenomenon of granularity, but also gives a meaningful rule of thumb for ontological stratification into fixed levels of granularity.

Clancey in [81] describes the use of abstraction, aggregation, and refinement as a generic problem-solving strategy.

When we abstract a road to a line, surface, or volume, this is often indicated in natural language by metaphor. The line abstraction for instance gives rise to *intersections*, and we do see this term applied to road networks. A term such as *box junction* necessitates a refinement to a surface. These spatial abstractions are by no means the only ones for a road: another common abstraction is the hydraulic or cardiovascular one (congestion, artery).

It is important to note that shifts of granularity are not performed on the thing itself, but on abstract models of the thing. The road, and mathematical line, surface, and volume remain different things on the ontological level (see figure 2.1): the abstraction *ideally* speaking simulates the original as in the computer science technique of *bisimulation*. Abstraction is the relation between the thing and its abstract model. In the context of bisimulation techniques we can also distinguish overapproximating and underapproximating abstractions, as in [223], and we can for instance distinguish idealizing abstractions to explain the use of prototypes in knowledge representation. The thing itself, the road, remains firmly fixed to the ordinary realm of the senses, of the conventional referents.

Most deontic logics for instance (consciously) position themselves as overapproximating abstractions, explicitly making predictions not always observed in the original thing. Makinson's argument in section 2.2.2 for instance points this out with respect to the composition of norms. In chapter 6 we will sometimes explore the fine line between underapproximation and overapproximation in reasoning about norms. Some kinds of reasoning about norms are after all more defeasible than others.

The notion of simulation, as used here, has some similarity with the notion of isomorphism from section 2.2 and fidelity from 2.2.1. Each of these can be mistaken for an identity relation and cause confusion.

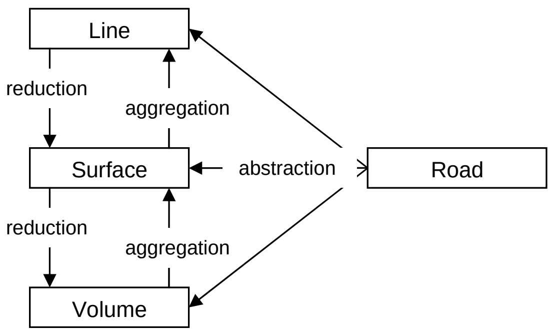

Figure 2.1. Abstraction, aggregation, and refinement.

A refinement moves from a coarser grained abstract model to a finer grained one: it is not the inverse of abstraction, but of aggregation. Abstraction should certainly not be confused with terminological subsumption, the main organizing relation of the ontology, even though it may on occasion be implemented by an operation having the same operational semantics.

Refinement is also not the same as decomposition, although refinement generally involves decomposition (i.e. the refined model has additional parts). Refinements R

are also not by necessity simulations of the aggregated whole A. If it is, we speak of a *perfect refinement* or *reduction* $R \models A$. Refinement is a form of *ampliative reasoning* (to be discussed in section 2.4.1), in the sense that it adds new information not entailed by the problem description. Making predictions about a real thing based on its abstraction is obviously always ampliative.

An unfortunate choice of level of granularity can simply fail to gain traction over the problem. Whether to take a particular stance, is determined by how successful that stance is when applied. Aggregation and abstraction are epistemic principles: whether we will gain traction over a problem by an abstraction move is not a matter of ontology, but of the reliability of the process in the context in which it is used.

Reusable ontologies are clusters of terms that correspond with these levels, stances, perspectives, etc, to be each individually fitted on a situation to gain traction. Good examples are Hobbs' scale, change, cause, plan, and goal in [150]. The notion of a physical object with a location, an orientation, and movement (cf. [153, 183]) is another convincing example. Reusable ontologies sometimes go by the name of *core ontologies*, if we only consider the set of terms, or *core theories*, if we also take into account the non-ontological rules positioning them as a knowledge source.

Different KBS choose different levels of granularity, and a KBS may even switch between levels of granularity in the process of generating hypotheses. This does not imply that the underlying *ontologies* are somehow defeasible, or contextual. It is not an argument for nonmonotonic inference techniques in terminological reasoning. Rules recommending an abstraction, or effecting a shift between abstraction levels, are not ontological: if the decision to abstract eventually introduces incoherency it is the abstraction move that should be questioned. Rules recommending an abstraction move are defeasible.

The decision to *equate* concepts from different terminologies because they concur often or always in a specific KBS setting is obviously equally non-ontological and defeasible, and creates serious maintenance problems.

2.3.2. Ontological Stratification

The discussion of granularity in knowledge engineering relates to a central issue in philosophical ontology: the classification of qualitatively different types of existence (cf. for instance [254]) that will be referred to in this book as *ontological stratification* (following i.a. [55]). Strata are usually envisioned as deposited "on top of" each other, and objects in different strata are "co-located" against the same background, often spatio-temporal, with an intentional relation between things in one stratum to things in another one, and each stratum has separate criteria for existence in that stratum. Things in different strata explain each other in some way, but they do not *reduce* to each other.

Even on the physical level one can distinguish different strata that clearly do not reduce to each other in a meaningful way (cf. generally [84], and [254] for a discussion of phenomenological, microbiological, and physical-chemical apples). The distinction between physical, mental, social, and abstract objects is a typical example of stratification, as are Dennett's physical, design, and intentional stances, and the distinction between ontology and epistemology made here. A well-written inquiry into the nature of legal reality is for instance found in [204]. A more general discussion of the rationale for stratification is however beyond the scope of this book.

The existence of the physical stratum can be demonstrated through the senses, directly or only in carefully managed experiments using special tools. The existence of the mental is presumably based on introspection. The abstract is created by a reasoner and identified by its purpose in reasoning. The existence of the social is grounded in (one's awareness of) recognition by multiple consciousnesses[10].

There is a strong temptation, maybe grounded in Ockhamist reductionism (as [254] believes), and maybe simply in pre-existing epistemic habits[11], to assume that things that are co-located but inhabit different strata are in effect one and the same thing, conceived from different points of view. Obviously, we are part of reality, perceive reality through our experiences, and our experiences are contingent upon our ontological stance towards reality. Coincidence is however different from identity, even if it is often possible to fudge the consequences of the differences.

The *ontological distinction principle* (cf. [55]) prescribes that terms whose meaning is determined by different identity criteria must be disjoint, but we do not have to subscribe to this principle. The following weaker principle does hold without reservation:

Proposition 1. Ontological stratification principle: two concepts that describe objects in different strata do not stand in a terminological entailment relation to each other.

The road (physical thing) and the line (abstract thing) are not necessarily to be treated as disjoint terms: if the KBS only needs road-as-a-line objects, the road and its abstraction *can* be rolled into one, for the sake of simplicity, as long as we accept that predictions based on the line metaphor are correct for the road only within a certain margin of error. Instead of for instance keeping track of the fact that i' is an abstraction of i, and i a reference to some relevant thing in the outside world, one simply assumes $i = i'$, i.e. we make them co-referents on a logical level (even if one would, if pressed, nevertheless insist that they are two different things). Lines, surfaces, and volumes however cannot be equated without introducing inconsistency.

There are many instances of this coincidence issue. Well known is for instance the "Cicero = Tully" (or "Superman = Clark Kent") identity problem (cf. for instance [224]). The agent identified by both Superman and Clark Kent for instance has the ability to fly, but only the social *role* of Superman is associated with flying. Do we need to distinguish the different behavioural expectations we have about the social roles of Superman and Clark Kent? Do we for instance dare to equate "the murder suspect" and "the murderer" in our minds, or are we open to the possibility that "the

[10][198] for instance distinguishes between social objects in the narrow and broad sense, and gives quark and triangle as examples of 'social' in the broad sense because their meaning depends on communities that recognize them. Note however that it is not at all *necessary* for the community to exist for triangle to be a useful concept: it is a perfectly valid and useful abstraction for an individual consciousness. Abstract and social are distinct strata.

[11]perhaps the same ones that lead us astray through metonymy and synecdoche (see section 2.3).

murder suspect" and "the murderer" may turn out to be different people? We know we have to separate them as soon as we encounter ontological incoherence, or bizarre entailments, but in many cases we can equate them without problems.

Chapter 4 will address the issue of why the identity of *institutional* facts and the *constitutive* facts they are grounded in should be conceptually separated, and why action and the task that is performed by the action should be separated. In chapter 5 the issue of why a document as a work, as an expression, as a manifestation, and as a (physical) item should be identified separately, as advocated by [245].

A KBS may hold incompatible conceptualizations of "the same thing" if properly stratified. It is however possible to organize knowledge components well, by clearly separating terms and terminologies on the ontology level into separate strata, without burdening KBS developers with an explosion of spurious abstract entities in the deployed KBS if this is unnecessary. In this book I will generally explicitly separate things for the purpose of clarity wherever needed, without subscribing to the principle that the domain and range of a relationship spanning ontological strata *must* be disjoint.

2.4. Epistemology

The epistemological dimension of knowledge representation accounts for the processes by which we acquire knowledge, and more specifically, how the knowledge representation realizes epistemic competence, i.e. the competence to solve a type of problem, in the hands of a rational agent that understands the knowledge representation. Central to this account is the notion of problem solving.

A problem is an obstacle which makes it difficult to achieve a desired objective, or for some machine to serve its purpose. In a broad sense, a problem exists when an individual becomes aware of a significant difference between what actually is the case and what is intended and expected: to solve the problem he has to bring about a change. The concept of *problem solving* is however rarely explicitly connected to the notion of a problem.

Standard descriptions of intelligent agents posit some three-step think-act-perceive cycle such as in [240]. This simple problem solving cycle, as sketched in figure 2.2, is at the center of the activity of problem solving: intelligent activity consists of monitoring, and comparison of observations to intentions and expectations, finding and understanding obstacles, or problems, characterized as significant deviations from the intended scenario, and planning, or designing, to remove them. The cycle positions two fundamentally different epistemic roles of knowledge:

(know-why) modeling and explaining the situation (cf. for instance [82, 258]), and
(know-how) deciding what to do given the (explanation of the) situation (in accordance with Newell's conception of knowledge in [209]).

This distinction is similar to the distinction between construction (how) and classification (why), or synthesis (how) and analysis (why) (cf. generally [59]).

Presumably we have hit upon a problem every time when deciding what to do next is non-trivial. Very often explaining the situation and deciding what to do can be considered separate phases in problem solving, although one could argue that the optimal course of action should take into account all possible conceptualizations of the situation.

The three-step cycle loosely fits well on Breuker's problem types in [59]. The major difference between this cycle and Breuker's original acyclic proposal is the explicitness of the re-entry from remedy to planning and repair to design. In the case of repair, we can distinguish between repairing the machine in the case of malfunction of the machine relative to the specification, and revising the design if monitoring a simulation, scale model, or prototype shows it does not live up to expectations. Note that we do not only think in order to "act". Lack of satisfaction with one's diagnosis results will for instance generally result in a plan to monitor specific behaviours or even to coerce the occurrence of behaviours that help distinguish between alternative diagnostic hypotheses: in this case we act in the service of refining our explanation of the situation. The most trivial attempt to test a hypothesis is to *ask the user*: many KBS do nothing more than conducting a structured dialog with the user.

Note that most actual KBS do not support a single iteration of the whole cycle, but focus on aspects of it that are better – in the sense of effectiveness or efficiency – performed by a computer than by a human.

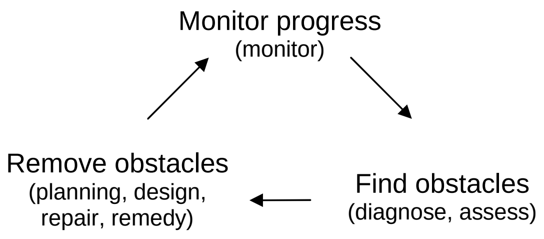

Figure 2.2. Problem solving cycle.

Considering the relation between the notion of "processes by which we acquire knowledge" and the notion of epistemic competence, we hit upon an ontological mismatch. Competence, whether physical, epistemic, or legal, naturally relates to *action*, following a decision, while an *epistemic process* suggests something that happens unintentionally, beyond conscious control. Apparently, we can distinguish between a kind of reasoning that "happens in the background", beyond conscious control, and another part that is consciously and intentionally used or not. We at least have two different vocabularies to apply to reasoning.

In problem solving (method) literature we find both characterizations in terms of function – suggesting an embodied process, a machine, or a design specification for one, that can be used for some purpose – and in terms of competence – more suggestive of decision making and planning. Also to problem solving methods we can apply Dennett's design and intentional stance of [95].

Terminological inference on the ontology is the most obvious candidate as a background process, since we already committed to ontology being not falsifiable within a KBS, perhaps with other such processes we have not explicitly identified, while the interesting part of problem solving involves decision making on the epistemic level.

To solve a problem is to set oneself a task, and one has the epistemic competence to solve the problem if one knows an applicable (problem solving) method for performing the task. The problem solving method is characterized by a set of assumptions about the knowledge domain and the available domain knowledge that function as conditions for its application, a functional specification, and an operational specification (cf. [109]). The functional specification tells us what epistemic competence will be realized, and the operational one how it will be realized.

The terms describing the problem solving method belong to epistemic vocabulary: they exist to describe processes through which one reliably creates knowledge. Assumptions about available knowledge are *autoepistemic*, i.e. they are about epistemic entities, since they involve introspection over what one knows (or knows a reliable acquisition process for).

Sources of law obviously do not specify problem solving methods; At best they sometimes give instructions about the application of rules that presume certain things about the use of rules in the mind of the reasoner, and about available knowledge. To position these instructions, some explanation of my perspective on problem solving, and in particular defeasibility, is needed.

2.4.1. Ampliative Reasoning and Entrenchment

Problem solving methods generally are said to involve *abduction*, or inference to the best explanation. Inference to the best explanation is a basic method of reasoning in which one *chooses*, from a space of *hypotheses*, the hypothesis that would, *if* true, best explain the relevant evidence. Abductive reasoning is an account of *ampliative reasoning*: it adds information that was not already implicitly available in the premises. It is often described as a dual of deduction, as abduction allows the precondition ϕ of $\phi \models \psi$ to be inferred from the consequence ψ, and it is thus clearly non-deductive. But not every inversion of a deduction is obviously also a meaningful abduction; Making abductions is an epistemic decision making process, and one needs a good plan for that. Good strategy depends on what problem one is trying to solve.

The generic problem solving method par excellence, Generate & Test, is considered a generic abductive framework; Most problem solving methods in literature (cf. [76]) postulate additional epistemic assumptions or knowledge sources for generating hypotheses in the best possible order or cut down the portion of the search place that needs to be explored.

In addition there are many generic theories of default or nonmonotonic reasoning that can considered accounts of ampliative reasoning. A shortcoming of these generic approaches is that they do not account for how ampliative inferences help to solve a problem, as opposed to problem solving method literature which does make this connection.

Parametric design is an example of a problem type that has a well-studied set of ampliative problem solving methods associated to it (see for instance [75, 207, 281, 280, 108]). Read these if the idea of a problem solving method remains unclear.

An abduction problem can be characterized as a tuple of sets of propositions[12] in conjunction (T, A, S), where T is some background theory, S the situation to explain, and A a set of abducibles. A subset $E \subseteq A$ is an (abductive) explanation of S in a certain knowledge state if we believe that:

1. $T \cup E \models S$
2. $T \cup S \not\models \neg E$
3. $T \cup E \cup S$ is consistent[13]

The function of the abducibles is in essence to specify a set of propositions that may be assumed instead of (deductively) proved. We are willing to retract abducibles if they do not fit reality. There is a nice proverb about the role of these *assumptions*: "the wish is the father of the thought". We do not make an assumption just because we believe it is true, but because it believing it *brings us closer to solving our problem.* The optimal strategy for explaining a situation therefore depends on the problem setting.

E is minimal if no proper subset E' of E is an explanation of S. One simple way to characterize *the best explanation* is as the disjunction of all minimal explanations. Typically one however wants to be able to explicitly choose between the members of this set of minimal explanations, i.e. to order them by (epistemic) preference, if explanations may be falsified at a later point in time by additional information.

Abduction problems can be conceived of in deductive frameworks in terms of model generation or satisfiability. The concept of abduction has been related in literature to various approaches to *defeasibility* such as belief revision and truth maintenance systems, default and other nonmonotonic logics, and to negation as failure in logic programming (cf. for instance [214] for an overview).

The *hypothesis* is an epistemic entity, a provisional idea that can be falsified by a test against evidence. A hypothesis has greater *explanatory power* if it provides greater opportunity for its own falsification. In a knowledge engineering setting, the objective is in my view generally to generate concrete plans, designs, arguments, or explanations, that:

1. are ontologically coherent, i.e. are not inconsistent with, are a valid model of, the terminological axioms;
2. pass some adequacy test, meet constraints and requirements, achieve some objective, etc;
3. can be executed, used, built, etc; generally, are at the right level of detail to be acted on; and
4. can be monitored by comparing intended or expected events against observable events to determine whether one encounters a new problem.

The characterization of abduction problems helps as a general framework for understanding how the first two objectives are met. The last two criteria are generally filled in implicitly in KBS, and refer to the required explanatory power. The KBS works

[12] I intend proposition in a general sense here: whatever it is that is expressed by a sentence that can be evaluated to true or false.

[13] Note that A need not be internally consistent, and it is not a given that E is.

towards a level of granularity and detail that is deemed suitable by the designers of the KBS. The simplest method of achieving this is to ensure that the reasoner has no more knowledge available than the knowledge required to produce an explanation on the right level of granularity and detail: by ensuring that reasoner has no knowledge that is irrelevant to the problem at hand, it is ensured that the reasoner does not perform irrelevant inferences. The set of abducibles is therefore usually carefully handcrafted to result in explanations on exactly the right level of detail.

The characterization of abduction problems given here fits better with the role of knowledge as explaining the situation than the role of knowledge as a resource for deciding what to do next: this is a result of referring to S as a situation, instead of, for instance, a plan or a design. To explain how a chess board can contain eight queens that do not attack each other is to solve the problem of placing the eight queens. S is sometimes conceived of as a goal, but this is merely confusing.

There is however a fundamental difference between explaining the situation and deciding what to do next, and it is most easily described in terms of decision making, of choosing between alternatives. Choice between alternatives reveals – from the external point of view – or is motivated by – from the internal point of view – a *preference* ordering. There are two fundamentally different ordering principles for generating and testing hypothesis:

1. The first ordering principle, modeled by *epistemic entrenchment* (cf. [117, 1, 208]), orders hypotheses on epistemic quality, i.e. the likelihood that they retain descriptive fidelity, or the order in which we adopt and abandon hypotheses to come to a solution. This is a preference relation on the *epistemic level*, which finds its rationale in an assessment of how reliable the process is by which it the hypotheses were generated. This ordering principle is the one modeled by defeasibility and nonmonotonicity in logic and knowledge representation.
2. The second ordering principle only comes into play when we consider reasoning directed towards a decision to do something: it models the order in which we adopt and abandon plans and designs. This is a preference on the *intentional level*, a utility measure on plans, designs, etc. In analogy with epistemic entrenchment, we may call this (with a neologism) *intention entrenchment*.

Example 1. I for instance prefer winning a lottery over losing a lottery, but I prefer *believing* that I will lose a lottery over believing that I will win a lottery, since this is likely to save me money in the long run. Too strong a preference for favourable outcomes will lead to unrealistic plans and designs.

The two different methods to order hypotheses find their origin in two different decision making processes. Epistemic level decision making – dominant in prediction, postdiction, diagnosis, and assessment – deals with explaining and evaluating what is happening, or *what to think*. Intentional level decision making – dominant in design, planning, and scheduling – deals with *what to do* and is normally speaking relative to the problem we are solving.

The distinction also plays a prominent role in classifying the epistemic role of rules found in sources of law. The two ordering principles are relevant to a number of legal theoretical concepts discussed in the next chapters. The norm is for instance based in *deontic choice*. This appeals to the intentional level. Deontic logics also vary in the assumptions they make about this ordering relation in decision making. A notion like applicability on the other hand is to be understood in the context of epistemic-level decision-making.

Choice between alternatives and the composition of preference relations therefore plays an important role in the following chapters, both on the intentional and on the epistemic level, because of its central importance in explaining reasoning. Preference relations do not, however, very often explicitly play a role in KBS.

Managing both orderings in problem solving at the same time is a complex task. The best known even-handed approach for it is the *expected utility hypothesis* central to decision theory (cf. generally [137, 240]): if certain (not very plausible) assumptions are met, then utility and probability of alternatives can be composed into a single expected utility function that orders alternatives and can be used to choose between them. In knowledge engineering, expected utility and probability theory in general play a minor role. Expected utility is a “useful abstraction” in the sense discussed in section 2.3.1, but also one that has little cognitive plausibility and comes with very clear limits in applicability.

Since knowledge about generating hypotheses and knowledge about testing hypotheses are in practical KBS settings rarely in balance, both phases generally occur: even if we can compare any two alternatives to determine which one is preferred, we are not necessarily able to generate them in that order. We may not even be able to generate hypotheses consistent with the background theory: many practical approaches to abductive reasoning require filtering of the generated hypotheses. This creates a lot of variation in the knowledge we may have about the preference relations, and a lot of variation in approaches in knowledge engineering for generating and testing hypotheses.

The assumed mathematical properties of the preference order relations employed have a big influence on the efficiency of problem solving. It for instance makes a difference whether we recognize an adequate score immediately, i.e. we are dealing with a *scale* and can safely discard the hypotheses not yet seen because we have hit a sufficiently high or the highest score on the scale. If we can generate hypotheses in the correct order, then the test is reduced to confirming, for the KBS in general, that no better hypothesis exists than the one generated. This is a good reason to make assumptions, for instance the ones underlying expected utility, even if one knows they are occasionally falsified in practice.

The notion of a preference ordering to explain choice is not entirely straightforward: the hard problem of knowledge representation – and of decision making – is generally to enumerate the sensible alternatives. As pointed out by for instance Sagoff in [241] preference as revealed by choices appears empirically well-founded in the notion of choice, but the choice problem is in the end as unobservable as the preference. The following joke explains the issue well:

> Two graduate students overtook Professor Paul Samuelson as he walked. “There’s a beggar at the corner,” they told him, “who, when offered the choice between fifty cents and a dollar, always takes the fifty cents.” Samuelson replied, “He’s irrational.”
> Samuelson decided to see for himself. “In my left hand, I have fifty cents; in my right, one

> dollar; you may have whichever one you prefer," Samuelson said to the beggar. "I'll take the fifty cents," the man answered without hesitation.
> After giving him the two quarters, Samuelson asked, "Don't you understand that a dollar is worth twice as much as fifty cents?" "Of course I do." "Then why did you take the fifty cents?" "Had I taken the dollar," the beggar replied, "economists wouldn't troop down here every day to offer me the choice."

The set of alternatives ascribed to the beggar and chosen from by the professor and the set of alternatives chosen from by the beggar are not the same set: the professor failed to generate the right explanation of the behaviour of the beggar.

Both proposed preference orders suffer from serious cognitive plausibility problems, also recognized in law. As for instance pointed out by Mommers in [202] the idea of the autonomous court of law presupposes *doxastic voluntarism*, i.e. the stance that one can consciously choose whether or not to believe something. Amsterdam and Bruner in [6] attack this very proposition, based on evidence from psychological experiments, but the legal system cannot function without this presumption. The same may obviously be said of behaviour compliant with norms: the legislator and the court of law must assume that human beings are capable of consciously integrating the law into their behaviour, and it must ascribe choices between a concrete set of alternatives to them.

Law cannot function without the presumption that the sources of law can influence choices between alternatives, and that these choices are observable from the outside.

Preference relations also have the redeeming feature that they characterize underlying problem solving strategies in a simple way, and link them directly to available logics and algorithms, regardless of whether they are cognitively very plausible descriptions of intelligent behaviour.

In chapters 4 and 6 I often explain how the law's commands are integrated into behaviour in terms of choices and preferences, for no better reason than that legal theory also tends to defer the issue to *rational choice theory* (cf. Hansson in [137] for an overview of choice theory).

2.4.2. Epistemic Things and Epistemic Roles

Problem solving strategy is conceived of in terms of processes and decision problems involving epistemic things. Epistemic things are mental things: a mind "has" them, and the representation refers to them. If the "mind" is the KBS's, mental and abstract collapse into one for practical purposes; Let's call this the *internal* perspective. Since the mind makes its own epistemic things, it may in principle also be able to maintain perfect fidelity. For the KBS, a proposition and its representation in the form of a data structure are one and the same thing. This is not the case with hypotheses about what somebody else knows; This is the *external* perspective: the epistemic object is *ascribed* to someone else. The KBS may need a theory of both, and it does not necessarily use the same theory for the internal mental world and external mental worlds.

Of particular importance to knowledge engineering are the epistemic entities I will refer to as *doxastic*[14] or propositional attitudes: these express mental attitudes towards, or the epistemic role or status of, propositions. These form the metaphorical

[14] i.e. pertaining to beliefs.

bridge between our conceptualization of the mental world and the knowledge representation language that expresses it.

Searle's speech act theory (cf. [248]), one of the dominant perspectives on language, is founded in a strict distinction between the proposition expressed by a speech act, and the speech act's meaning as an act, grounded in a theory of action. We can make a similarly strict distinction between propositions and the epistemic objects that contain them. Problem solving methods, the abductive framework sketched before, and logics in general, can then be conceptualized as theories of epistemic process or action.

Logical entailment $\phi \models \psi$ is a relation on propositions, independent of their epistemic role or status as a belief. The *belief*, a simple proposition container, is the fundamental building block of the mind from a doxastic perspective. Commitment to the existence of individual beliefs with propositional content justifies reification and manipulation of the sentences of the knowledge representation language as if they were beliefs. This allows me to make epistemic level statements such as "believing ϕ is better than believing ψ" or "ϕ is a terminological axiom". An account of reasoning is a account of sets of beliefs held in a progression of knowledge states.

Epistemic terms are found everywhere: hypotheses, factors, evidence, symptoms, indicators, problems, solutions, etc (cf. [26]). As pointed out before (see 2.3, about metonymy), epistemic vocabulary should not be confused with the domain ontology: flue is not a "subtype" of symptom in the medical domain, but often has the *epistemic role* of symptom in medical diagnosis.

Epistemic vocabulary often depends on whether we take an *ex ante* or *ex post* perspective: we hypothesize about actions or events in the past or future, or we reason about designs and plans before or after we start executing or using them. Ex ante *prediction* and ex post *postdiction*, or, often, simply, explanation, are basic abduction problems, and do not connect to action.

Breuker's problem types in [59] can also be classified on this dimension: planning, design, modeling, parametric design, and scheduling are conceptualized from an ex ante perspective, while monitoring, diagnosis, and assessment are conceptualized ex post. Remedies and repairs should be understood as re-entries into planning, but described from an ex post perspective. Breuker's problem types are conceptualized in the context of decision making and action.

One of the complicating factors of law, it that it also explicitly uses epistemic terms with complicated semantics, to be used from both internal and external perspectives. We need to explain the LKBS's operations in terms of them because proper justification, with reference to the sources of law, is a central issue in law. Accounts of reasoning given in legislation and legal theory often lead us away from the approach that would be chosen for the same type of reasoning problem in other knowledge engineering fields.

Take as an example for instance the doctrine of *mens rea*, the guilty mind, which distinguishes between intention (direct or oblique), knowledge, recklessness, and negligence[15]: all of these are epistemic modalities, to be ascribed to a suspect, and together they form a simple theory of the functioning of the mind. These concepts will however

[15]Or the comparable Dutch distinction between oogmerk, opzet, voorwaardelijk opzet.

not be found in automated planning literature. As I will explain in chapter 4 theories about norms in law are also strongly influenced by the distinction between the ex ante and ex post point of view.

Summarizing, it makes a difference whether we are taking an internal or external perspective to epistemology. The internal perspective is the one we take to describe automated reasoning by the KBS. The law takes an external perspective to epistemology when it ascribes intent or knowledge to agents. The way we describe reasoning depends on our position relative to the thing we are reasoning about (ex ante, ex post), whether we are dealing with a machine or plan, and whether we are making a decision or simply trying to understand something. In any of these contexts, we can make more and less plausible inferences, and when we are making a decision we choose between better and worse alternatives.

Finally we get to the relationship between epistemic objects and the things they are about. It is very common to simply equate the things we are reasoning about with the variables and constants of our logical language, while epistemic roles characterize the sentences we can create in the language: epistemic role is a qualification of a reified proposition. The language used in this book will be introduced in section 3.2. This brings us to the next level: the level of language and information. By making knowledge explicit, we turn it into information.

2.5. Argumentation and Information

More often than not, knowledge based systems do not reason in order to decide what to "do", but simply tell the conclusion of reasoning, and its explanation and justification, to the user. In [59] the explanation and justification are even considered essential parts of a solution.

Explanation and *justification* are two quite different things. When you explain something, you do not have to take a position on whether it is good or bad. When you justify something, you try to explain why something is good. They belong in different vocabularies – one for problem solving and one for decision making. Generally speaking one solves a problem, and explains the solution, as opposed to making a decision in a situation, and justifying the decision. Decision making involves a preference ordering of alternative outcomes, and generally involves a design or plan, while explanation applies to any kind of problem solving and generally focuses on epistemic entrenchment; Hence the earlier characterization of abductive reasoning as "inference to the best *explanation*".

Since, as noted in the previous section, epistemic entrenchment can also be conceived of as rational decision making on the epistemic level, it is possible to recast explanation into decision making terms. One can for instance *explain* how one arrived at a taxable income, or *justify* one's income tax claim.

Explanations and justifications are constructed from arguments. An argument is typically a *claim*, or "thesis", expressing a proposition, backed up with evidence that supports the claim.

Argumentation is primarily an *information* issue, and not an ontological or epistemological one, although there are obvious connections between good problem solving strategy and good argumentation.

If one engages in discourse one exchanges *information*, in the form of propositions. By doing so, on commits oneself to these propositions; they become part of a shared discourse context, i.e. the parties in the discourse, including yourself, know you have claimed these propositions are true and they expect that you also believe that propositions they believe are entailed by it – according to them – are true (cf. generally [227]). The same logical entailment that operates on sets of beliefs can also be applied to this shared discourse context, but the discourse context has no direct relation to belief sets: it is perfectly possible to commit oneself to propositions one does not believe are true, and the totality of exchanged propositions obviously need not be consistent.

Information is a message received and understood. Information is a quality of a message from a sender to one or more receivers. Information is always about something: it is the proposition represented by an act of representation. It assumes the existence of a common language understood by the sender and at least one of the receivers: it is that which would be communicated by a message if it were sent from a sender to a receiver capable of understanding the message, i.e. that knows what the message is about.

Since the existence of a definite sender is required, information cannot be directly extracted from an environment, e.g., through observation or measurement. Knowledge on the other hand can be. Knowledge and information are different things. Observations and measurements do *become* information by representing them and "sending" them to some receiver capable of understanding them.

The study of information exchange is different from the ontology and epistemological considerations that determine what the information is about. As pointed out in section 2.3, information exchange is ruled by principles (such as for instance the Gricean maxims of quality, quantity, relation, and manner; cf. generally [227]) and tactical considerations (cf. for instance [230]) that are foreign to ontology and epistemology. Presentation of argumentation in court, or any other setting for that matter, is not a direct representation of the underlying legal reasoning, but a competitive game. It is in fact a separate planning problem for the involved parties (as for instance explicitly pointed out in [256]).

The KBS can be conceived of as a machine that sends and receives, in accordance with a predefined plan, information conformant to a predefined schema. In typical KBS use cases a human provides a formalized problem description to a computer, and receives a solution and explanation to interpret. It is however good to realize that the roles in the transaction can also be reversed: the computer asks a person or a large number of people to solve a problem, then collects, interprets, and integrates their partial solutions. This idea – *human-based computation* – aims to optimize the use of differences in abilities between humans and computer agents, and is already used for image and video rating and labeling on the World Wide Web on a voluntary basis. It is obviously not always the KBS that plays the role of knowledge source: KBS will increasingly collect knowledge from "the wisdom of the crowd".

If a KBS engages in a structured dialog, i.e. it doesn't just answer questions, but also poses them, it implicitly creates a division of labour between itself and its human user(s). Implicitly this division of labour either distributes a burden of proof, or embodies assumptions about 'who knows what'.

The plan that determines the structure of the dialog between KBS and user is very often a derivative product of problem solving strategy or the logic used. In some cases ([57] comes to mind) the argumentation, or explanation, for a certain thesis is explicitly equated with the *proof tree* produced by the problem solving process. For instance [272] describes our own approach to compiling proof trees out of general rule sets in an application intended to help clients explain a request for legal assistance in legal terms.

Although there are additional considerations to take into account for good argumentation, it is obvious that explicit argumentation must closely follow the problem solving strategy; Firstly, the argument made in section 2.2.2 would be meaningless if no such close relationship existed, and secondly the audience, presumably equipped with similar symbol processing machinery, is supposed to *understand* the message. Obviously, *doxastic* epistemological theories are in a sense always based on the presumption that argumentation reveals the working of the underlying epistemic process.

The abductive problem setting sketched in the previous section for instance clearly suggests an interaction form: if possible explanations are sets of abducibles, it makes sense to translate the abducibles into questions to the KBS user in the order the KBS runs into them. A slightly more refined approach asks for the most discriminating abducible first, i.e. an abducible that cuts our set of possible explanations into to equal halves. See for instance [259] for a useful general theory of asking relevant questions.

While the information and argumentation level are very interesting to discuss in relation to knowledge and reasoning, we need to keep in mind that knowledge and information, and reasoning and argumentation, are indeed different things. When one implements a KBS one does need to consider the interaction between KBS and users, but the reasoning process does not need to directly mirror its explanation, its justification, or any kind of related argumentation encountered in the field. In this book both the human computer interface of the KBS and legal argumentation play almost no role in the explanation of knowledge representation decisions. The perspective taken is strictly that of one *mind* solving legal problems.

2.5.1. Adversarial Dialogs

A mode of message passing that deserves special attention is the dialectical or adversarial one, often referred to as (legal) argumentation in the narrow sense, where the participants exchange arguments and counter-arguments for a thesis or antithesis. These argumentation games are often ruled by a procedure and assigned burdens of proof and information duties. This is most apparent in court cases, which function as a more or less level playing field for argumentation. Many other legal exchanges of claims have a similar form, but with one dominant side who makes the rules and for instance a supplicant who claims entitlement to a benefit.

Because the court is the prototypical setting of the legal system, it has a large influence on the way legal problem solving is conceptualized in legal theory. In computational legal theory we find a large school of thought that places adversarial argumentation on the center stage (cf. generally i.a. [278, 122, 230]). Defeasibility and other such violent metaphors characterize this approach. When one models an adversarial dialog *setting*, i.e the discourse context instead of the content of the minds of the participants, then some of the observations made earlier no longer apply: since the participants in argumentation are not committed to agreeing with each other on any subject, even ontology can become a subject of discussion and is therefore *defeasible*, etc.

Since case law is obviously one of the most important knowledge sources for LKBS, there is an argument to be made for modeling legal argumentation, the procedures that guide it, and the "good moves" that are made as we find them, instead of trying to distill coherent epistemological "deep structures" (cf. [93]) underlying them. This is a point explicitly made for instance by Gordon in [35]. Since issues of ontology and epistemology are rarely explicitly addressed in case law, they appear relatively unimportant.

The relative rarity of arguments about ontology can however also be interpreted in favour of the importance of ontology and epistemology in knowledge engineering as I pointed out in [30]. Deconstructivist Schlag explains in [247] that the games of law can be played (and won) even if one doesnt know what the ball (the law) one plays with looks like: no one engaged in "doing law" will raise the question. Superficially this supports the view that ontology is unimportant. Schlag (cf. [247]) proposes that the following rhetorical hierarchy is what actually guides legal practice:

1. Do not confront an ontological question if it can be handled as an epistemic question.
2. Do not confront an epistemic question if it can be handled as a normative question.
3. Do not confront a normative question if it can be handled as a technical question.

Ontological questions question the truth of terminological axioms, and the ontological inferences based on them. Based on the examples given in [247], it is clear that epistemic simply means (indirectly) questioning (the reliability of) a non-terminological inference made, from certain premises that function as evidence and that are not themselves questioned, to arrive at a thesis. Normative questions address whether something is allowed or disallowed, good or bad, etc. In each of these cases the inference is in law generally based on a rule, and it is either the existence of the rule, the accuracy of the description of the rule, or the applicability of the rule to the case under scrutiny that is questioned.

Technical simply appears to mean "relating to the (truth of the) facts of the case". It concerns questioning those propositions, not (yet)established by an explicit argument from premises, whose truth is left to common sense. These propositions do not generally have the character of a rule.

This rhetorical hierarchy is relevant to legal knowledge engineering because because it exposes a theory about *epistemic entrenchment* that normally remains implicit. The hierarchy tells us which propositions to put forward and which (parts) of the opponent's propositions to attack. It tells us that explicitly attacking terminological axioms is the weakest move one could make, in court and, I would propose, outside it. Questioning ontology and the quality of epistemic processes directly attacks the very notion that a shared discourse context exists and that we can perform logical entailment on it. Moving to these levels, certainly the ontological one, is bound to provoke irritation.

This hierarchy has practical application in legal knowledge engineering. The following list tentatively orders the four levels mentioned by Schlag by decreasing reusability in law, and therefore by decreasing usefulness as a *reusable* knowledge component:

1. Ontology
2. Epistemology
3. Normativity
4. Technique

Ontology is highly resistant to change, and is assumed to be *not* falsifiable at all in KBS. The rhetorical hierarchy corroborates that design choice. Even ontology is in the end subject to questioning, at least in its sense of being a shared understanding between discourse participants, but certainly not in a routine decision making context. Falsification of an ontological axiom is rather a reason for knowledge engineers to review the decision problem solved by the KBS altogether.

Epistemology and normativity can be interpreted as relating to the what to think (epistemology) and what to do (normativity) distinctions of section 2.4.1. Schlag makes a clear choice here, by making rules guiding interpretation of a situation more resistant to being questioned than rules guiding decision making. Also this is in line with section 2.4.1: interpretation of the situation (or finding obstacles in terms of section 2.4) is indeed logically prior to taking action in it (removing obstacles).

In practical KBS both are often not falsifiable: the KBS, whose knowledge base is expressed in a knowledge representation language that exhibits the property of monotonicity, reflects one single unambiguous interpretation of the knowledge domain relative to a specific problem. In AI & Law theory one however also finds many proposals that consider all legal reasoning falsifiable. An intermediate form also exists; In for instance the framework of [265] (but also my own work in [286]) the possibility of falsification of normative qualifications *is* taken into account, but situation recognition is assumed to be a mere matter of translating the case into the terms of the ontology. The possibility of falsifiability of the interpretation of the situation is not considered. One does not easily find proposals that are counter-examples to the hierarchy proposed by Schlag.

The defeasibility of arguments in a discourse with more than one participant is a fundamentally different issue from defeasibility of rules in reasoning, when reasoning is equated with problem solving. Ontology is not defeasible in problem solving. In argumentation it may be, but even the tactical rules for argumentation proposed by Schlag acknowledge that questioning ontology is a rare and often even ill-adviced move.

2.6. Conclusions

Reasoning and knowledge as conceived in this book are based in the concept of reasoning as *problem solving*, and knowledge as a resource in problem solving. Knowledge directly relates to descriptive *fidelity* and to *problem solving competence*, and only indirectly – after one has committed to a specific doxastic theory of reasoning – to information and its structure.

In this book therefore both the human-computer interface of the KBS and legal argumentation play only a minor role in the explanation of knowledge representation decisions. The perspective taken is strictly that of one *mind* solving legal problems, and the knowledge ascribed to it to explain its apparent problem solving competence.

On the other hand we need to be aware that one cannot avoid assuming the existence of some interesting relationship between beliefs in reasoning and arguments in argumentation, and more generally, that information reveals something about the processes that created it and use it – even if some intelligent “decoding” is necessary. The use of a knowledge representation language of course presumes some commitments to an interpretation of the mental world in terms of structured information. In this book, we are committed to the description logic OWL DL.

The objective of this book is however hard to align with the notion of knowledge as problem solving: reusability of knowledge components that represent a source of law – not a problem or task – must be increased by keeping considerations of problem solving competence at arm’s length. Separation of reusable knowledge about the domain and reusable knowledge about problem solving, or ontology and epistemology, as characterized in [26, 153, 154] is the first step to designing reusable knowledge components.

For the subject of this book this means that we are interested in ontological knowledge about the world of law, *as expressed* in the sources of law. Epistemological knowledge plays a role to the extent that the sources of law *themselves* model that mental world.

The sources of law will however generally use epistemological concepts and constructions different from those used in knowledge engineering, while one of our goals is a general compatibility with general knowledge engineering work on problem types like planning, assessment, etc. In chapter 4 a survey of relevant legal concepts and patterns is made, with the aim of divorcing epistemological background – that often just explains the main functions of legal constructs from certain especially relevant points of view – from the ontological essentials that are represented. The resulting representation of *legal rules*, to be developed in chapter 5, is often very simple and straightforward.

Section 2.3 identified some information-level issues that complicate the design of ontologies: polysemy, metonymy, synecdoche, etc. When one tries to distill ontological knowledge from written text, these complications should also be kept in mind: one cannot sincerely take a written text “literally”.

Section 2.3 about ontology makes a distinction between concrete objects that are a surrogate for something out there (mental or physical), and abstract objects that only take their meaning from what you can do with them on an epistemological level.

Contrary to physical and mental concepts, abstract ones can usually be fully defined (with necessary and sufficient conditions). This distinction between a physical, mental, and abstract world is an example of *ontological stratification*, the classification of qualitatively different types of existence (cf. for instance [254, 55]).

Strata are usually envisioned as deposited "on top of" each other, and objects in different strata are "co-located" against the same background, with an intentional relation between things in one stratum to things in another one. This identification between structures in different strata, and the perception that they are layered on top of eachother tells us something important about epistemology.

The common application of ontological strata deposits one or more abstract layers on top of a concrete one. The move from a concrete thing to an abstract one identified with it, is *abstraction*. Abstracting a concrete thing is done in order to use the associated abstract operational theory to explain something about the concrete thing. By doing so one temporarily makes the abstract thing a surrogate of the concrete thing. "Scientific style" theories recommend an abstraction to an abstract, mathematical thing: the inferences that can be made inside the theory after this abstraction is made are not considered *hypothetical*, but rather the rules telling us when to abstract[16].

Section 2.4.1 made a distinction between non-ampliative, deductive reasoning, and ampliative, hypothetical reasoning. The realization of non-trivial problem solving competence invariably involves ampliative reasoning. A premise for the rest of this book, to be applied to the law, is that the direction in which we hypothesize involves a change of ontological stratum. Only for rules introducing new objects in another ontological stratum defeasibility plays a role. The conceptualization of an ontological stratum is by definition not defeasible within the problem solving process: instead one may switch between conceptualizations in order to "gain traction" over concrete reality.

If ampliative reasoning involves defeasibility, then there are alternative hypotheses (actions, situations, beliefs) to choose from and a preference relation orders the alternatives. Ampliative reasoning is therefore often characterized as a decision problem –involving reasoning actions – while non-ampliative reasoning is a process. Very often a KBS manages two interacting decision making problems at the same time: an epistemic level one, on beliefs, to address defeasibility, and an intention-level one, on actions, or the situations one expects as a result of actions. Knowledge of these preference relations again will play a role only to the extent that the sources of law *themselves* model it: no attempt will be made to model cognitively plausible or logically convenient choice principles that make problems solvable. Chapter 3 provides us with some basic instruments for defeasibility. In chapter 6 I will have to appeal to the intention-level preference ordering to explain *normative order*.

The following chapters will attempt to stratify law. Special attention will go in chapter 4 to intention, a mental thing, and institutional things, which are social in nature.

When one is representing (knowledge about) sources of law, *isomorphism* between

[16]E.g. apply $F = ma$ if you can determine a candidate m and a, but only if it is safe to assume that m remains constant. Consider for instance the application of the formula to a snow ball thrown from a building or rolling from a slope.

the source and its representation is of course of great importance. It is an explicitly stated objective of this book. Isomorphism in the sense used in section 2.2 means that there is some structural coincidence between the source of law and its representation.

This structural coincidence is in this book however not based on identifying text fragments in the source of law with logical propositions in the knowledge representation language. Makinson and Dubislav (in section 2.2.2) rightly criticize this identification for norms, but this argument in my view also extends to legal rules, or any other things "contained" by the source of law. Instead there is a structure of objects in the knowledge domain that represents the meaningful objects found in the source of law, and the logical propositions that we qualify as knowledge are *about* these objects. This also involves a move between ontological strata.

This approach to representation and isomorphism will be worked out in chapter 5.