A Method for Conceptualising Legal Domains: An Example from the Dutch Unemployment Benefits Act

*Pepijn Visser, Trevor Bench-Capon and *Jaap van den Herik*

Department of Computer Science
University of Liverpool
P.O. Box 147
L69 3BX Liverpool
United Kingdom
Tel: (+44) 151 794 3792
Email: {pepijn, tbc}@csc.liv.ac.uk

* Department of Law and Computer Science
University of Leiden
P.O. Box 9521
2300 RA Leiden
The Netherlands
Tel: (+31) 71 527 7837
Email: jfricd@ruljur.leidenuniv.nl

There has been much talk of the need to build intermediate models of the expertise required preparatory to constructing a knowledge-based system in the legal domain. Such models offer advantages for verification, validation, maintenance and reuse. As yet, however, few such models have been reported at a useful level of detail. In this paper we describe a method for conceptualising legal domains as well as its application to a substantial fragment of the Dutch Unemployment Benefits Act (DUBA).

We first discuss the intermediate models (called expertise models), then present a three-stage method for their construction, drawing on the CommonKADS work in knowledge acquisition, conceptual models of statute law, and the KANT method of knowledge analysis. Subsequently, we describe how these techniques were applied to the DUBA, and provide detailed examples of the resulting model. Finally, conclusions on the framework and guidelines are given as well as means of recording and presenting the various design choices.

1. Introduction

As experience with constructing legal knowledge-based systems (LKBS) has grown, it has become increasingly evident that their disciplined construction requires that we first build a model of the knowledge to be incorporated at a level higher than that of executable code. Originally there were hopes that the declarative nature of the languages used to build LKBS would enable a transition from source knowledge, whether in the form of texts or elicited from an expert, to executable code with no intervening steps. But these hopes proved false; transformation into executable code requires a number of decisions to be made: selection, interpretation, choice of entities and attributes, level of abstraction and the like, and it is important that these decisions be recorded explicitly, rather than needing to be reconstructed from the code. The need for proper design and documentation is present for any properly-engineered system, whether a LKBS or a conventional system. It is now usual to attempt to produce intermediate models of the systems (*cf.* the idea of building a knowledge-level model (Newell, 1982) preparatory to building a symbol-level model). In the CommonKADS method (Breuker and Van de Velde, 1994), which inspires much of the work described in this paper, the intermediate model is termed the "expertise model". Advantages of an expertise model include:

- *Verification and Validation* (V&V): For a full discussion of the role of such an intermediate model in these important processes, see Vermesan and Bench-Capon (1995). Here we will say only that the kind of V&V that can be performed on the actual representation reduces to a set of checks for internal coherence of the model; identifying redundant rules, dead-end rules, sets of conditions that cannot co-obtain in the model, and the like. While these "anomalies" can identify possible errors in the model, these errors must always to be referred back to the expert to see if there really is a problem. In the absence of an explicit intermediate model, therefore, appeal has to be made to the model implicit in the head of the expert. Since this is implicit, the model cannot be guaranteed to be itself consistent and coherent. Thus an explicit model is desirable as providing an objective standard against which the V&V activities can be performed.

- *Maintenance*: The case for an intermediate model to support this activity is made in works such as Coenen and Bench-Capon (1993). Essentially the argument here emphasises the particular importance of adaptive maintenance as against corrective and perfective maintenance for LKBS. This being so, it is typically changes to the source knowledge which require changes to the system, and thus it is essential that there is a very clear relationship between the sources and the code to be maintained. In the executable model it is often impossible to sustain this relationship. Moreover, when maintaining the system, the maintainer must have access to the design choices of the sort mentioned in the first paragraph of this paper, if the maintenance work is to be consistent with these choices. Again, these choices are better made explicitly available than inferred from the code itself.

- *Reuse*: This is Visser's (1995a) main concern. Since there is a very great investment in constructing a knowledge base for a LKBS it is important that as much mileage as possible be obtained from it. Thus, we want to reuse the knowledge base constructed for one application in another application of the same domain. With regard to executable code, this is often very difficult, since the knowledge will have been selected, represented and optimised for a particular task, assuming that it is to support that task as efficiently as possible. The aim in building the intermediate model is to provide a task- neutral representation, which can then be customised if necessary to support particular tasks, effectively separating the reusable knowledge from the task- specific knowledge.

Although now widely seen as desirable, there have been as yet few detailed attempts to build substantial models of this sort in the legal domain reported. Only through such experience will the problems and a method for attacking them emerge. In order to make some experience available, in this paper we describe the construction of an expertise model for a particular piece of legislation; Articles 1 through 29 of the Dutch Unemployment Benefits Act (DUBA). In doing this we introduce a three-stage method found to be effective, and draw attention to some of the particular problems and issues that arose.

A key stage in the development of an expertise model is deciding on the entities and relations to be distinguished in the model. In general, such decisions can be made from different perspectives, resulting in different conceptualisations of the domain (and hence in different expertise models). Following Gruber (1992) we refer to an explicit conceptualisation of a domain as an *ontology*. The idea is to provide a knowledge level description of the terminology that eventually will be used to express the domain knowledge (Van Heijst, 1995). The method introduced in this paper can be seen as an ontological analysis in which we gradually work towards a (formal) ontology for the domain under consideration.

In the remainder of the paper, we first describe the domain (section 2), then the method (section 3). Next we discuss each stage of the method by applying it to the DUBA (section 4), we provide a brief discussion on the ontology and its use (section 5), and finally we make some concluding remarks (section 6).

2. The Domain: Dutch Unemployment Benefits Act (DUBA)

The particular Act on which we focus is the Dutch Unemployment Benefits Act, as issued in November 1986. This legislation is thought to be representative of the sort of legislation for which LKBSs have been constructed. It contains several different types of norms and facts, makes use of time related concepts, and has a limited amount of open texture, relating to the notion of "suitable employment".

The Act comprises eleven chapters, each divided into several parts, which divide into paragraphs, which in turn are divided into articles, which divide further into sections and subsections. Altogether there are 137 articles. Here we concentrate only on the first two chapters, and in particular articles 1 through 29. This part addresses three issues:

- Descriptions of legal concepts (articles 1 - 14): these articles set out some of the central concepts used by the Act including the notions of *employee, relationship of employment, employer, and wage per day.*
- Requirements for entitlement to unemployment benefit (articles 15 - 21): these articles set out the conditions that must be satisfied if unemployment benefit is to be awarded. Chief amongst these are that the employee must be unemployed, and must have worked for 26 weeks in the last 12 months. Also contained here are certain exceptions to entitlement, and the conditions for determining when the entitlement starts and ends.
- Effectuation of entitlement (articles 22 - 29): these articles set out what must be done if a person who is entitled to the benefit is actually to receive it. For example the employee must not become culpably unemployed, and must be making a sufficient effort to obtain suitable employment. The articles also describe the sanctions that can be applied if these conditions are not met, including reducing the amount of the benefit, or withholding it in full.

The open-textured concept of *suitable employment* has been widely discussed, and now there is a reasonable consensus on its meaning. In particular de Wildt (1993) presents a model derived from previous decisions which can be used to simulate decisions, essentially based on the identification of six categories of factors, some pro and some con, which can be considered together to come to a decision. We follow this model in our work. Some factors are decisive: thus if the wage offered is below the minimum, the employment cannot be considered suitable, whereas others, such as the educational level of the employee, are only influences on the decision, and must be assigned some kind of weight when considered with other similar factors.

As can be seen from this brief description, the selected fragment of legislation is substantial, relatively complete in that it can form the basis of several tasks suitable for a LKBS, and representative of the sort of legislation which forms the basis of many projects.

3. The Method

In building our model we drew principally on three different pieces of research; CommonKADS (Aamodt *et al.*, 1992; Wielinga *et al.*, 1992), which supplies the overall idea of an expertise model and what it contains; a theory of frame-based conceptual models of statute law (Van Kralingen, 1995), used to supply a generic ontology of the legal domain; and KANT (Bench-Capon and Coenen, 1992), used to determine the vocabulary with which to instantiate the domain model. We discuss each of these strands in this section.

3.1. The Expertise Model of CommonKADS

The expertise model in CommonKADS consists of two parts: *application knowledge* and *problem-solving knowledge*. Below, we deal with each in turn.

3.1.1. Application Knowledge

A principal attraction of the CommonKADS expertise model is that it enforces a clear separation between control knowledge and domain knowledge, requiring independent descriptions of each. This is vital for our purposes since the compilation of domain and control knowledge is the major drawback of symbol- level models. The application knowledge is meant to include all the knowledge required for solving problems in the domain. Within this, three categories are distinguished:

(1) Task Knowledge
(2) Inference Knowledge
(3) Domain Knowledge.

Task knowledge, which together with the inference knowledge makes up the *control knowledge* of the expertise model, specifies how the problem- solving goals can be achieved. Tasks may be *primitive tasks*, or *composite tasks*, built up from primitive tasks. Primitive tasks are linked onto inferences. Each task has a *task definition* which specifies what the task can bring about, and a *task body* which describes the steps by which this can be achieved.

The inference knowledge specifies the primitive reasoning steps that can be performed. Each inference establishes a relation between entities drawn from the domain knowledge (see below). These are the only knowledge entities that can directly use the domain knowledge.

Domain knowledge comprises three components (Wielinga *et al*., 1992); a *domain ontology*; a *case model*; and the *domain models*. The domain ontology is an explicit conceptualisation of the domain (Gruber, 1992). It defines the vocabulary, typology and taxonomy with which the case and domain models can be specified. Within CommonKADS there is a conceptual modelling language, CML, which provides three constructs: objects (atomic and constructed), relations, and value sets.

The case model is a description of the dynamic data related to a case, and includes the initial case data, available before the reasoning commences, intermediate data, derived during the reasoning process, and the case solution, which is the ultimate result of the reasoning process. All of these are specified by reference to the domain ontology.

The domain models are coherent sets of expressions, again specified using the domain ontology, which describe the domain from a particular perspective, such as its causal relations, or its components.

3.1.2. Problem-Solving Knowledge

This part of the expertise model comprises problem-solving methods and strategic knowledge. Problem-solving methods specify the candidates which could be used to satisfy a certain class of task definitions. Strategic knowledge is used to select a relevant method from the available problem- solving methods. This part of the expertise model is less fully elaborated in the writings on CommonKADS. Partly as a result of this lack of clear guidelines we did not, in fact, specify this part of the expertise model, and hence our expertise model of the DUBA consists only of the application knowledge: task, inference and domain knowledge.

3.2. Van Kralingen's Conceptual Frame Structures

CommonKADS is a domain-independent methodology, and in consequence, the method does not supply very much detailed support for particular domains. In particular the ontology needs to be extended for the domain under consideration. Some work has been done on this in law (Valente, 1995), but it is not sufficiently established to be immediately useful to us. We therefore required a set of domain specific structures for modelling the legal domain. For this we turned to Van Kralingen's work (1995).

Van Kralingen identifies three basic entities to be found in legal systems: *norms*, *acts*, and *concept descriptions*. Each of these are provided with a frame structure specified at a conceptual level. The work seemed suitable for our purposes since it provides a means of structuring the domain in a way customised to law and it provides detailed schemes - firmly based on legal theory - of the entities relevant to our domain. The frame structures are useful both to guide the knowledge engineer in acquiring knowledge, and to provide a framework enabling the knowledge acquired to be assimilated. The slots of the frames identified show what information is required, and hence will identify gaps during the knowledge elicitation process, so as to drive further elicitation. When the knowledge is obtained, it is used to complete the identified structures. We will briefly discuss each of the three basic entities in turn.

3.2.1. Norm Frames

Norms are the general rules, standards and principles of behaviour that subjects of law are enjoined to comply with. They are statements to the effects that something ought to, or may, or can be, done. Norm frames comprise the following slots:

Norm identifier:	The norm identifier (used as a point of reference for the norm).
Norm type:	The norm type (norm of conduct or norm of competence).
Promulgation:	Promulgation (the source of the norm).
Scope:	The scope (the range of application of the norm).
Conditions of application:	The conditions of application (the circumstances under which the norm is applicable).
Subject:	The norm subject (the person or persons to whom the norm is addressed).
Legal modality:	The legal modality (ought, ought not, may or can).
Act identifier:	The act identifier (used as a reference to a separate act description).

3.2.2. Act Frames

Acts represent the dynamic aspects which effect changes in the state of the world. There are two types of acts: events and processes. Events represent an instantaneous change between two states, while processes have duration. Act frames comprise the following slots:

Act identifier:	The act identifier (used as a point of reference for the act).
Promulgation:	The promulgation (the source of the act description).
Scope:	The scope (the range of application of the act description).
Agent:	The agent (an individual, a set of individuals, an aggregate or a conglomerate).
Act type:	The act type (both basic acts, and acts that have been specified elsewhere can be used).
Means:	The modality of means (material objects used in the act or sub acts; e.g., a gun).
Manner:	The modality of manner (the way in which objects have been used or sub acts have been performed) (e.g., aggressively).
Temporal aspects:	The temporal aspects (an absolute time specification; e.g., on the first of August, on Sundays, at night, etc, but not: during a fire, after the King dies, etc).
Spatial aspects:	The spatial aspects (a specification of the location where the act takes place; e.g., in the Netherlands, in Leiden, on a train).
Circumstances:	The circumstantial aspects (a description of the circumstances under which the act takes place; e.g., during a war).
Cause:	The cause of the action (a specification of the reason(s) to perform the action, e.g., revenge).
Aim:	The aim of the action (the goal visualised by the agent; e.g., with a view to unlawfully appropriate an object).
Intentionality:	The intentionality of an action (the state of mind of the agent; e.g., voluntary).
Final state:	The final state (the results and consequences of an action; e.g., the death of the victim).

3.2.3. Concept Frames

Concept frames deal with the meanings of the concepts found in the domain. They may be definitions or deeming provisions and can be used to determine definitively the meaning of a notion, either by, as in the case of the former, providing necessary and sufficient conditions, or, as in the case of the latter, establishing a legal fiction. Another type of concept is the factor, which may either establish a sufficient condition, or indicate some contribution to the applicability of the concept, as discussed above. Finally we may have meta concepts which are provisions governing the application of other provisions. Concept frames comprise the following slots:

Concept: The concept to be described.
Concept type: The concept type (definition, deeming provision, factor, or meta).
Priority: The weight assigned to a factor.
Promulgation: The promulgation (the source of the concept description).
Scope: The scope (the range of application of the concept description).
Conditions: The conditions under which a concept is applicable.
Instances: An enumeration of instances of the concept.

3.3. Instantiating the Domain Models using KANT

Van Kralingen (1995) has provided us with an explicit conceptualisation of the domain, which we use as the basis of our domain ontology. What is now required is a way of instantiating these frame structures so as to produce the domain models. To do this we must identify the specific vocabulary for our detailed domain: the predicates and constants that we will use. The method used to achieve this is based on the Knowledge ANalysis Tool (KANT) as described by Bench-Capon and Coenen (1992).

The key elements taken from this method are the creation of the Freestyle KANT structures, and the creation of a class hierarchy representing the logical types of the domain, their attributes and the values that these attributes can take. The Freestyle structures are created in three steps:

1. Pass through the source documents recording the entities found there, the attributes ascribed to them and any tests that are applied to them
2. Collect the attributes and tests for each entity into a "Tests On Objects" (TOO) structure. In this step we effectively identify the various data, initial, intermediate and final, which are required by the case model.
3. From the TOO structure derive the set of Entity-Attribute-Value Triples required in the domain. These triples represent the propositional functions required to express the rules and data of the domain.

The next step is to organise these Entity-Attribute-Value triples into a hierarchy of logical types. At this point we finalise the attributes that entities can have, and the possible values they can take. This is important both as a way of tightening up on the definitions that we use, and of identifying some implicit knowledge. For instance, the only explicit test on the attribute "sex" to be found in the legislation may be to see if someone is female. It may, however, be important to record that this attribute can also take the value "male", and no other value; or that it can take the values "male" or "neuter" and no others. One of these choices will be appropriate in the context, and needs to be recorded.

At this point we will have a tightly defined vocabulary for the domain which will enable us to instantiate the frame structures.

4. Applying the Method to the DUBA

In this section we discuss how the method described above was applied to the DUBA. We describe the process by considering each of the three techniques introduced. As we go we will highlight the particular problems and issues that arose, and say how they were resolved. This section will contain a high degree of detail: we believe this is necessary both to give an impression of the depth to which an analysis must go, and because it is at the most detailed level that the sharpest tests for the method are presented.

4.1. Identification of Domain Knowledge

The key first stage is to identify the knowledge that we wish to include as domain knowledge and to distinguish it from the knowledge we wish to include as control knowledge. In particular we must examine the legal knowledge contained in our chosen fragment of the DUBA to consider how it is best apportioned between these two knowledge types.

An obvious answer, traditional when modelling legislation, is to consider all the information in the legislation to be domain knowledge: after all the legislation is, in some sense, the domain of the application. Closer inspection of what is actually done, however, reveals that matters are less simple. In the seminal paper on the formalisation of the British Nationality Act (Sergot *et al.*, 1986), not all the legislation was in fact formalised. The parts not formalised:

> "consist mostly of descriptions of amendments to other Acts, repeals, offenses and proceedings related to them, and decisions involving exercise of description by the Secretary of State".

Thus even in this work, which set out to be a formalisation of the Act, some of the legislation is found not to require inclusion as domain knowledge. If we are to propose a systematic methodological approach to formalising legislation, however, it is necessary to characterise the legislation that should be excluded from the domain knowledge, so that its inclusion or exclusion is not simply a matter of judgement by the modeller. Even more important is that we identify knowledge that is properly excluded from the domain knowledge, but which requires inclusion as control knowledge.

In what follows we argue that the legislation does contain information best regarded as control knowledge, and how this may be recognised.

4.1.1. Defining Control Knowledge

Solving a problem is performed by a series of problem-solving actions (*e.g.*, invoking a procedure, deriving new information from the knowledge base, acquiring data). In general, at each point in the problem-solving process a choice of problem-solving actions is available. Following Davis (1980), we distinguish three phases with respect to the invocation of a (problem-solving) action[1]:

(1) *identification of candidate actions*: (collecting the set of actions that could be performed);

(2) *conflict resolution*: (selecting an action from the set of possible actions);

(3) *execution*: (the execution of the selected action).

The problem of selecting an action in the conflict-resolution phase is called the control problem (*cf.* Hayes-Roth, 1988, p.505). Providing a strategy for making a selection is referred to as solving the control problem; the knowledge used to solve the control problem is called control knowledge (Clancey, 1986, p.4). We note, that if the action can be executed by the system itself (instead of by other agents), then control knowledge is an instance of meta-knowledge. As a consequence, reasoning about such control problems is meta-level reasoning (*cf.* Davis, 1980).

To identify control knowledge in legal texts, we have to identify problem-solving actions. Moreover, there should be knowledge about selecting a problem-solving action from a set of possible problem-solving actions. Following Franken (1988) we assume legal texts to consist of legal rules (legal rules can be any kind of text fragment, such as norms, articles, sections, concept descriptions). A problem-solving action is assumed to be the application of such a legal rule (*e.g.*, drawing a conclusion using this legal rule), or the examination of a set of legal rules. We can now state that the identification of control knowledge in legal texts involves identifying legal rules as well as the knowledge about selecting a legal rule from a set of legal rules. In other words, we focus on those control problems in which there is a choice between the application of different legal rules.

4.1.2. Control Knowledge in Legal Texts

Our discussion of control knowledge in legal texts concentrates on what we propose to call normative control knowledge, denoting legal knowledge which in some way obligates the user of the legal text (*e.g.*, a judge) to solve control problems in a predefined way. For instance, to sentence a suspect, it is required that the charge is proven (obligating the judge first to determine whether the charge is proven). We do not discuss forms of control knowledge in which the user of the text may choose how to solve the control problem

---

[1] We use a slightly different terminology. In particular, we avoid the term knowledge source because this term was also used to denote primitive inferences in KADS-I. Davis (1980) uses the term in a more general way (*e.g.*, procedure, inference rule, theorem).

without there being a legal obligation to do it in that particular way (for such control problems see, for instance: Van Kralingen and Visser, 1991; Groendijk and Oskamp, 1993). Below, we discuss two types of normative control knowledge in legal texts, namely control knowledge in:

(1) procedural norms of competence, and
(2) conflict-resolution knowledge.

4.1.3. Procedural Norms of Competence

The first type of control knowledge in legal texts is contained in procedural norms of competence (Van Kralingen, 1995). These norms state such things as who is empowered to perform certain actions, which preconditions must be satisfied before an action can be applied, and the like. We note that norms guiding the behaviour of people in society have different normative functions (*e.g.*, Hart, 1961; Brouwer, 1990) and that one of these normative functions deals with the control of the creation and application of other legal rules. These norms are norms of competence; they detail the manner and form in which the conferred power is to be exercised (Hart, 1961, p.28; Van Kralingen, 1995). The procedural norm of competence (henceforth: PNC) is a special kind of these norms, namely, the norms of competence which control the creation and the application of other legal rules by prescribing legal procedures. Some of these procedural norms of competence comprise control knowledge. This is argued below.

A PNC defines one or more intermediate steps which a norm subject has to take in order to reach a certain legal goal. Possibly, one or more of the intermediate steps can be achieved only through the application of (other) legal rules (*e.g.*, deriving intermediate legal conclusions). This possibility leads to the distinction between:

(a) object-level PNCs, and
(b) meta-level PNCs.

Below, we define these types of norms and illustrate that the second type of norm, in contrast to the first type of norm, comprises control knowledge.

(a) *object-level PNCs* are PNCs that can be performed without invoking other legal rules. For instance, when the Social Insurance Board wants to change some directives, then according to article 125 of the DUBA, an intermediate step should be the announcement of the change in an official newspaper. Although these object-level norms describe a flow of control through possible actions, the actions do not involve the invocation of legal rules. In other words, the set of possible problem-solving actions does not contain an action that requires the invocation of a legal rule. Therefore, this kind of PNC does not comprise control knowledge according to our definition, since the satisfaction of the precondition can be recorded as a fact, rather than needing to be deduced from the other legal rules (and so, they say nothing about the order in which other legal rules should be applied).

(b) *meta-level PNCs* are PNCs that do require other legal rules to be invoked. An example can be found in the Dutch law of criminal procedure. In article 350 of this statute it is stated that the following succession of questions needs to be answered before a person can be sentenced:

(350-1) are the facts described in the charge proved?
(350-2) do the facts described in the charge yield an offence?
(350-3) is the offender punishable? and
(350-4) what punishment or measure has to be imposed?

As will be clear some of these intermediate steps require other legal rules to be applied (*e.g.*, 350-2 requires examination of the offenses described in the Dutch Criminal Laws). Because the problem-solving action that invokes article 350 involves invoking other legal rules, the article is considered to contain control knowledge.

If we make an expertise model of a statute with PNCs, then these norms should be specified. Object-level PNCs do not comprise control knowledge, and, hence, can be specified as domain knowledge. With respect to the meta-level PNCs we are faced with a dilemma. On the one hand, norms derived from legal texts must be specified as domain knowledge. On the other hand, these norms comprise control knowledge, since they impose an order in which things must be done, and should therefore be specified as control knowledge (in terms of a CommonKADS expertise model). Note, that whenever a meta-level PNC is specified as control knowledge, the norm cannot be breached by the system. The system just enforces the norms by performing the prescribed problem-solving actions. This can both be a desirable feature (*e.g.*, in case the system has to determine one's entitlement) as well as a non-desirable feature (*e.g.*, in case the system should be able to monitor whether an institution complies with legal procedures).

Our dilemma has a pragmatic solution with respect to the DUBA in that meta-level PNCs do not occur in this statute. We note that the DUBA does contain several PNCs, for instance, in articles 111 through 127. All these norms are in fact object-level PNCs because the intermediate steps defined in these norms do not require other rules to be invoked (that is, not in our fragment of the DUBA). Hence, we find a practical solution for our dilemma: our legal texts do not contain control knowledge in the form of (meta-level) PNCs. However, in the general case, the question will need resolution, since our pragmatic solution will not always be available.

4.1.4. Conflict-resolution Knowledge

The second form of control knowledge in legal texts is found in what we refer to as (legal) conflict-resolution knowledge. By this we mean the legal knowledge used to discriminate between conflicting conclusions derived from different legal rules. Here, we illustrate that with respect to this knowledge the same dilemma arises as with procedural norms of competence. Although the knowledge is for a large part contained in legal texts it is to be considered control knowledge according to the definition given above.

We discuss the dilemma by first defining more precisely what is meant by a conflict and by conflict-resolution knowledge. After this, we illustrate that conflict-resolution knowledge is contained in legal texts. In particular, we show the potential conflicts that may arise from the DUBA and the knowledge used to solve these conflicts. This leads to a distinction of conflict- resolution knowledge into two parts. Then, we show that conflict-resolution knowledge is an instance of control knowledge, thus yielding a dilemma. Finally, we resolve the dilemma by choosing what part of the conflict-resolution knowledge is modelled as domain knowledge and what part is modelled as control knowledge in the expertise model.

4.1.4.1. Defining a Conflict and Conflict-resolution Knowledge

A common technique in drafting legislation is to state a general rule, and then refine the rule by stating some exceptions which lead to a different conclusion. Were both the general rule and the specific rule to be applied, a contradiction would result. In fact, no contradiction arises in practice, since only one of the rules will be applied. However, the potential for a conflict arises and we need conflict-resolution knowledge to determine which rule is to be applied, so that the conflict can be avoided.

Here, we define a conflict as a state in which there are two conclusions which, from a legal point of view, cannot be or remain true during the rest of the reasoning process (*cf.* Aarnio, 1987, p.113). According to this definition, the conclusion from a rule, and the conclusion (from another rule) that the former rule is not applicable is considered a conflict. Also, there is a conflict if two rules give incompatible values for an attribute, e.g., ‘twelve years of imprisonment’ and ‘fifteen years of imprisonment’ (example from Prakken, 1993, p.30).

Conflicts can be dealt with by preferring one of the conclusions. This requires knowledge on the preference relations between the conclusions (or on the rules that were used to derive the conclusions). This knowledge is, for instance, required to determine that if there is a conclusion that a rule is not applicable then this conclusion is to be preferred over the conclusion of that rule. The knowledge that is used to solve conflicts, that is, to prefer one of the conclusions, is here referred to as conflict-resolution knowledge.

4.1.4.2. Conflicts and Conflict-resolution Knowledge in the DUBA

Legal texts are stated in such a way that conflicts (as we defined them) frequently occur (*cf.* Thornton, 1987). Because most of these conflicts are anticipated by the drafter of the legal text (*e.g.*, exceptions to rules) there apparently must be conflict-resolution knowledge available in some form. So far, we did not elaborate on the appearance of this conflict-resolution knowledge in the legal texts. Below, we illustrate that conflict-resolution knowledge is partly contained in legal texts. We examine the DUBA for (potential) conflicts and discuss the related conflict-resolution knowledge. We do not consider conflicts that can arise as a result of the introduction of new (fragments) of ruling (such as conflicts between older and newer versions of legal rules).

To identify conflicts in the DUBA we use the definition given above. We identify pairs of legal rules of which their conclusions cannot be or remain true from a legal point of view. These pairs are listed below. For reasons of brevity we list only one example of pairs of legal rules that cause each of the conflicts (other pairs of rules that yield the same conflicting conclusions are left out). This yields the following six categories (we assume these conflicts are the only conflicts that can occur):

(c1) conclusion ‘employee’ (derived, for instance, from article 3 section 1) and conclusion ‘not an employee’ (derived, for instance, from article 3, section 2);
(c2) conclusion ‘relationship of employment’ (derived from article 4 section 1) and conclusion ‘article 4 section 1 is not applicable’ (derived from article 4 section 2);
(c3) conclusion ‘relationship of employment’ (derived, for instance, from article 4) and conclusion ‘no relationship of employment’ (derived, for instance, from article 6);
(c4) conclusion ‘entitled to benefit’ (derived from article 17) and conclusion ‘article 17 is not applicable’ (derived from article 18);
(c5) conclusion ‘entitled to benefit (derived, for instance, from article 17) and conclusion ‘not entitled to benefit’ (derived from, for instance, article 19);
(c6) conclusion ‘suitable employment’ (derived from, for instance, an employees employment history (as meant in De Wildt 1993, p.37), and conclusion ‘not suitable employment’ (derived from, for instance, the status of the work offered as meant in De Wildt (1993, p.37)).

The conflict-resolution knowledge with respect to these conflicts is partly contained in the legal texts. We illustrate this. If we study the conflicts listed we can distinguish different kinds of conflict-resolution knowledge. In conflicts (c2) and (c4) the rules involved in the conflict explicitly state which of the two rules involved in a conflict is preferred; the one rule explicitly states the other rule not to be applicable. Obviously, this information is contained in the legal text. Beside this information, the conflict-resolution knowledge comprises a principle stating that the conclusion of the inapplicable rule should be withdrawn, unless it can be justified by some other, applicable, rule. In this example, we consider the conflict-resolution knowledge to comprise two parts, namely (a) the information that a certain rule renders another rule inapplicable, and (b) a general principle stating that a rule that renders another rule inapplicable should be preferred over the rule that is rendered inapplicable.

With respect to conflicts (c1), (c3) and (c5) we find no explicit statement about how the conflicts have to be solved. This does not imply that there is no solution to these conflicts (they are not considered to be undecided conflicts, see: Prakken, 1993). Because the conflicts arise between a general rule and an exception there is an unequivocal solution to the conflict: when both legal rules are applicable the exception is preferred over the general rule. Which legal rule is the exception and which legal rule is the general rule is not stated explicitly, but it is assumed that the user of the text is able to derive this information from the legal text together with knowledge of the structure of legal texts, common sense, and general understanding of natural language that enables one of the rules to be recognised as the more specific rule. For example, in legal texts, exceptions usually are stated directly after the general rule. Here, the conflict-resolution knowledge is considered to comprise: (a) the information that a certain rule is a general rule, and that another rule is an exception to that rule, and (b) a general principle that states that the conclusion of the exception should be preferred over the conclusion of the general rule (the well-known principle Lex Specialis Derogat Legi Generali).

Conflict (c6) differs from the other conflicts in that the legal rules involved in the conflict do not provide explicit or implicit information about how to solve the conflict. This is because here the legal rules do not stand on their own, they cannot be viewed in isolation of all other legal rules that relate to the same concept. We note that this contrasts with the conflicts (c1) through (c5) which can be solved without considering other legal rules. A conclusion on the notion of suitability only contributes to the conclusion on the conflict. This conclusion is drawn on the basis of the sum of all weights of conclusions pro and contra suitability. The weight of each separate conclusion depends on the weight of the legal rule. This information is associated with the legal rule and contained in the legal text (De Wildt, 1993). Again, the conflict-resolution knowledge comprises two parts: (a) the weights of the rules involved in the conflict, and (b) a general principle stating that the conclusion with the highest total weight should be preferred over the other conclusion.

In summary, we conclude that in all conflicts, (c1) through (c6), the conflict-resolution knowledge comprises two parts: (a) some information about the rules involved in the conflict, and (b) a general principle that applies this information to prefer one of the rules (or conclusions) in the conflict. Moreover, we conclude that the first part of the conflict-resolution knowledge, the information about the rules involved in the conflict, is either contained in legal texts, or can directly be derived from the texts. Finally, we conclude that the second part of conflict- resolution knowledge, the general principle that is applied to solve the conflicts, is not specified with the legal text. The principle is implicit and assumed by the drafter to be known by whoever is interpreting the legislation.

4.1.4.3. Conflict-resolution Knowledge is Control knowledge

We have illustrated that conflict-resolution knowledge is partly contained in legal texts. Here, we argue that this conflict-resolution knowledge is an instance of control knowledge. To see this, we reconsider our definitions of control knowledge and conflicts. Assume we have two problem-solving actions a1 and a2. Problem-solving action a1 is the application of a rule r1 and a2 is the application of a rule r2. The conclusions of these rules, conclusions c1 and c2, are in conflict. Because only one of these conclusions can be preserved, one conclusion has to be preferred. This preferred conclusion is used for making further inferences during the problem-solving process. From a different perspective, we may state that only one of the rules r1 or r2 can be applied successfully. If so, the process of preferring one conclusion is an instance of the control problem in which the set of possible actions is {a1, a2}. For this reason, we consider the problem of preferring a conclusion as a control problem, and the knowledge used in this process as control knowledge (*cf.* Tan and Treur, 1991).

The conclusions derived pose the same dilemma as discussed before. On the one hand, we illustrated that conflict-resolution knowledge is partly contained in legal texts, and this suggests that we should model this part of the conflict-resolution knowledge as domain knowledge (in the expertise model). On the other hand, we concluded that conflict-resolution knowledge is control knowledge, which suggests that we should model this knowledge as control knowledge (in the expertise model).

To cope with the dilemma, we have to decide upon the question how to specify conflict-resolution knowledge. Roughly, there are three options to deal with conflicts in a formal specification[2]:

(1) compiling out conflicts,
(2) making the inference engine deal with conflicts, and
(3) performing explicit meta-level reasoning about conflicts.

We choose a solution in between the second and the third option. From the conflict-resolution knowledge, we specify the general principles as control knowledge (in the expertise model), and the information about the rules as domain knowledge (also in the expertise model). Thus, information such as explicit statements concerning the applicability, weights and promulgations are specified as domain knowledge. The way this information is used is explicitly modelled (in task specifications) as control knowledge, while the information used by these rule-selecting tasks is modelled as domain knowledge.

---

[2] In Prakken (1993) a more elaborate discussion of options is provided.

4.1.5. Identification of Structured Entities: Norms, Acts and Concepts

The domain knowledge in an expertise model is structured as a set of domain models (*cf.* Breuker and Van de Velde, 1994, p.22). The most important domain models in our domain specification represent the norms, acts and concept descriptions in the legal domain. In this subsection, we elaborate on these structured entities in the DUBA. We focus on the conceptual framework as defined by Van Kralingen (1995) and identify what frames have to be created for norms, acts, and concept descriptions, respectively.

4.1.5.1. Norms

A norm is a statement to the effect that some action ought, ought not, must, or can be done. We already stated that the DUBA comprises norms of different nature (*e.g.*, norms of competence, and norms of conduct). We here elaborate on the norms in the DUBA.

In our expertise model we recorded all the norms found in the chosen fragment of the DUBA. However, when building an application, we must make a selection of these norms, since some of the norms are irrelevant with respect to the problem-solving tasks we selected for implementation (assessment and planning to receive the benefit). Most of the norms considered irrelevant deal with amendments. In building the system we assume the ‘legal contents’ of the DUBA to be static (no amendments are made to the statute during the problem-solving tasks, although of course the system will require maintenance to keep the knowledge up to date as the legislation changes). Considering the DUBA several norms concern amendments. These norms deal with the way (new or more detailed) regulations have to be issued, and with exceptional circumstances in which some agents (*e.g.*, a Minister) may overrule the existing regulations in the statute. Article 12 section 2, for instance, states that the Minister of Social Services and Employment may, in contrast to what is stated in articles 9 and 10, appoint someone as an employer. Assuming the ‘legal contents’ of the statute to be static, implies that all norms that state how amendments should be issued are not relevant for our specification[3].

We note that most norms that deal with amendments can be recognised by their norm subject. This is because making amendments to regulations is reserved for specific agents. In the DUBA we distinguish ten agents who are subject to a norm:

(1) the employee,
(2) the employer,
(3) the industrial insurance board,
(4) the Minister of Social Services and Employment,
(5) the Minister of Defence,
(6) the Minister of the Interior,
(7) the Minister of Education and Science,
(8) the Social Insurance Board,
(9) employment offices, and
(10) the General Unemployment Fund.

Six of these agents (agents (4) through (8) and (10)) only play a role in norms that deal with amendments. As a consequence the set of relevant norms considered here is restricted to those directed at the agents employee, employer, the industrial insurance board and employment offices. From the norms that remain we select 12 norms which we assume to be representative for the DUBA. These norms are directed at the industrial insurance board and the employee.

Norms with the industrial insurance board as norm subject:

(n1) the industrial insurance board can reduce a benefit if the employee is not fully available to perform labour or if the employee performs new labour since his benefit has been determined (article

---

[3] Although we assume the legal contents to be static, at any given time the Minister will have appointed some set of people as employers. This is specified using extensional concept descriptions (see, below).

20);

(n2) the industrial insurance board can terminate a benefit if the employee is not fully available to perform labour or if the employee performs new labour since his benefit has been determined (article 20);

(n3) the industrial insurance board ought to supply a benefit whenever an employee is entitled (article 22);

(n4) the industrial insurance board ought to notify the employee of their decision with respect to the benefit (article 22);

(n5) the industrial insurance board can reduce a benefit when the employee does not meet his or her obligations laid down in articles 24 and 26 (article 27);

(n6) the industrial insurance board can terminate a benefit when the employee does not meet his or her obligations laid down in articles 24 and 26 (article 27);

(n7) the industrial insurance board can temporarily reduce a benefit when the employee does not meet his or her obligations laid down in articles 24 and 26 (article 27);

(n8) the industrial insurance board can temporarily terminate a benefit when the employee does not meet his or her obligations laid down in articles 24 and 26 (article 27);

Norms with the employee as norm subject:

(n9) the employee ought not become culpably unemployed (article 24);

(n10) the employee ought to accept suitable employment (article 24);

(n11) the employee ought to submit a request in order to obtain benefits (article 26, section 1, paragraph b);

(n12) the employee ought to register at the industrial insurance board and keep this registration up to date (article 26, section 1, paragraph d).

4.1.5.2. Acts

A legal system guides the behaviour of persons and institutions in society. This behaviour is expressed by means of acts that are forbidden, permitted, and obligated. In this subsection we discuss the acts in the DUBA that will be specified in the domain specification.

Acts are specific instances of the more general category of occurrences, the latter being the category of all things that occur in the world (Allen, 1984, p.123). In the DUBA different occurrences can be distinguished, all of which can be shown to play a role in the entitlement of an employee to unemployment benefits. Before we list the occurrences in the DUBA we discuss six categories of occurrences. In particular, we distinguish between (a) acts and non-acts, (b) events and processes, and (c) institutional and physical acts.

(a) *acts and non-acts*

Occurrences can be initiated by agents. An example is the submission of a request for unemployment benefits. Also, there are occurrences that are (usually) not initiated by agents, such as snow fall or high water. We h