Chapter 2
Modeling the Domain: An Introduction to the Expert Module

Roger Nkambou

University of Québec at Montréal, 201 Du Président-Kennedy Avenue, PK 4150,
Montréal, QC, H2X 3Y7, Canada
Nkambou.roger@uqam.ca

Abstract. Acquiring and representing a domain knowledge model is a challenging problem that has been the subject of much research in the fields of both AI and AIED. This part of the book provides an overview of possible methods and techniques that are used for that purpose. This introductory chapter first presents and discusses the epistemological issue associated with domain knowledge engineering. Second, it briefly presents several knowledge representation languages while considering their expressivity, inferential power, cognitive plausibility and pedagogical emphasis. Lastly, the chapter ends with a presentation of the subsequent chapters in this part of the book.

2.1 Introduction

The purpose of intelligent tutoring systems (ITSs) is to enable learners to acquire knowledge and develop skills in a specific domain. To provide such tutoring services effectively, these systems must be equipped with an explicit representation of the domain knowledge that is the subject of the learning activity. It must also be equipped with the mechanisms by which the representation can be used by the system for reasoning in order to solve problems in the domain.

Acquiring and representing a domain knowledge model is a difficult problem that has been the subject of numerous Artificial Intelligence research projects since research in this field began (Clancey 1985; Brachman and Levesque 2004; Russell and Norvig 2009). Knowledge-based systems and expert systems, in particular, must explicitly represent the knowledge and inferences associated with the expertise in this domain.

Intelligent tutoring systems must also possess a domain-specific expert module that is able to generate and resolve domain problems and provide access to such knowledge in order to facilitate the dissemination (Wenger 1987; Woolf 2008) and acquisition of this knowledge by learners. Hence, developing an explicit model of domain knowledge with sound reasoning mechanisms is an important issue in the field of ITS research. The expert module of an ITS should provide the basis for interpreting learner actions (Corbett et al. 1997). It is therefore important

R. Nkambou et al. (Eds.): Advances in Intelligent Tutoring Systems, SCI 308, pp. 15–32.
springerlink.com

to consider not only the nature and value of the domain knowledge, but also the formalisms used to represent and apply it.

Many solutions have been put forward in an attempt to provide an explicit representation of domain expertise. The solutions presented have been drawn from fields such as philosophy, psychology, AI and education sciences. Philosophy, psychology and education sciences tend to focus on knowledge from an epistemological perspective which is essential for its treatment (Piaget 1997; Gagné 1985; Winograd and Flores 1986; Merrill and Twitchell 1994), whereas AI and cognitive sciences provide solutions that facilitate the expression and computational implementation of knowledge (Anderson 1996; Collins and Quillian 1969; Minsky 1975; Sowa 1984). From the perspective of AIED, it is important to incorporate the above-mentioned approaches so as to effectively meet the requirements associated with the development of a rich knowledge model and the inferential mechanisms associated with ITSs.

The goal of this chapter is to provide a brief overview of the means available for developing the expert module of ITSs. First, we present the epistemological perspective by addressing its importance in the domain knowledge modeling process for the purposes of learning. Next, we explore various languages that are available to represent knowledge and are frequently used to develop domain models for ITSs. We focus on the expressiveness, inferential power and cognitive plausibility of these formalisms and provide examples of the ITSs in which they have been used. We also present two examples of languages which have a strong pedagogical emphasis. We conclude the chapter with an introduction to subsequent chapters in this part of the book, each of which deals with a specific approach to domain modeling.

2.2 The Epistemological Perspective of Domain Knowledge

Epistemology refers to what we know and how we know it. The term “epistemology" is used here according to the definition given by Piaget (1997).Epistemology involves posing fundamental questions about the nature of knowledge (gnoseological considerations), the construction of knowledge (methodological considerations), and the value and validity of knowledge. In our opinion, these considerations are a prerequisite for formalizing knowledge pertaining to a specific domain. In this light, the epistemological perspective is of prime importance when characterizing the nature of knowledge and the inference mechanisms at play in a given domain. This perspective also makes it possible to question several aspects of knowledge: the production modes, the foundations underlying the knowledge in question and the production dynamics. The research carried out by Ramoni et al. (1992) clearly shows that this epistemological perspective is taken into consideration when developing knowledge-based systems. Classical knowledge engineering methodologies involve two distinct levels: the epistemological level and the computational level. The first is the level on which the epistemological analysis is carried out while taking into account the constraints derived from the conceptual structure of the domain knowledge, patterns of inference and tasks to be executed. The second level is the one on which the methods and formalisms must be adopted to formalize these elements. At the epistemological level, the ontology and inference model of a knowledge-based system must be defined. Ontology represents the conceptual model of domain knowledge which focuses on the nature, the properties and the constraints that govern the existence of knowledge. The inference model is the conceptual representation of the nature of the inference structure required to solve a problem or to execute a task by managing the ontology.

When designing ITSs, the nature of the knowledge to be taught or learned should be considered. An important aspect of epistemology consists in defining the nature of the knowledge involved in the learning process. The best known system of classification in the fields of AI and psychology divides domain knowledge into two types: declarative and procedural knowledge. However, more elaborate classification systems exist: Bloom (1975) and Gagne (I985) were among the first educational psychologists to develop clear classifications of knowledge and skills. They assert that different types of knowledge require different types of teaching or instructional methods. Other knowledge-typing schemes were later developed which are more strongly based on modern cognitive theory and are more operational and concrete for the purposes of computational representation. For example, Merrill's Component Display Theory (Merrill 1991) organizes knowledge in a matrix with content type (e.g., fact, concept or procedure) on one axis and performance level (e.g., remember, apply and create) on the other. This matrix scheme is more expressive and intuitive than hierarchical representations such as the one proposed by Gagné. However, epistemology not only involves distinguishing knowledge types, it also includes the quality and role knowledge types play in problem-solving situations. In this sense, Kyllonen and Shute (1988) proposed a multidimensional model which is more complex and which distinguishes knowledge types by means of a hierarchy based on cognitive complexity. This model organizes the knowledge types in relation to the level of the learner's autonomy and the processing speed needed to perform the task. More recently and in a similar manner to Kyllonen and Shute, De Jong and Ferguson-Hessler (1996) stated that, although absolute classification is important, a more pragmatic typology of knowledge should take into account the context in which the knowledge is used. They proposed a "knowledge-in-use" perspective which leads to an epistemological analysis of knowledge that renders ontological types and quality of knowledge.

Although the epistemological perspective of domain knowledge is of some significance in the field of education (as shown in the preceding paragraphs), it rarely surfaces in relation to ITSs and, as a result, has not received any special attention. This may be attributed to the fact that most ITSs have focused on procedural domains having limited scope, which target only the concepts and, more importantly, the tasks that such domains require. However, in the area of learning sciences, there has been a growing interest in the role of epistemology in teaching and learning. Studies suggest that the specific beliefs that teachers have about the nature of knowledge and learning influence their decisions regarding curriculum, pedagogy, and assessment (Schraw and Olafson 2008; Peters and Gray 2006). Moreover, as indicated above, the epistemological perspective of domain knowledge has clearly been a part of the established methodology in the field of knowledge-based systems for several years. Indeed, knowledge-based system methodology generally includes a step involving an epistemological analysis which focuses on the conceptual features of the system's two main components (knowledge about the domain and knowledge about the inference procedures needed to solve a problem) (Ramoni et al. 1992). The epistemological analysis provides a framework containing knowledge-structuring primitives which represent types of concepts, types of knowledge sources, types of structural relationships, such as inheritance, and types of problem-solving strategies (Brachman 1979; Hickman et al. 1989). The expert module of ITSs is similar to a knowledge-based system; accordingly, it should apply the same methodological principle in regard to knowledge representation. Since our goal is not to suggest new practices in the field of ITSs, we will end our discussion of epistemological considerations here. In the next section, we will present methods that allow for the computational implementation of knowledge representation and reasoning.

2.3 Computational Perspective

Epistemological analysis may help in specifying the constraints that are derived from the conceptual structure of the domain knowledge, the patterns of inference, and the tasks to be executed. However, in order to realize a computational model in a domain, certain methods and formalisms must be adopted. In this section, we describe some relevant approaches for the implementation of the expert module in ITSs.

2.3.1 A Historical View of Domain Modules in ITSs

From a historical point of view, three types of models have been identified in connection with ITSs: the black box models, the glass box models and the cognitive models.

2.3.1.1 The Black Box Models

A black box model is a completely inexplicit representation providing only the final results (Nwana 1990). It fundamentally describes problem states differently than those described by the student. A classic example of a black box model system is SOPHIE I (Brown and Burton 1974), a tutoring system for electronic troubleshooting that uses its expert system to evaluate the measurements made by students when troubleshooting a circuit. The expert system comprises a simulated troubleshooting model based on sets of equations. The tutor makes decisions by solving these equations. It can recommend optimal actions for each problem-solving context, but it is up to the student to construct a description of the problem-solving context and his/her rationale for the appropriate action.

2.3.1.2 The Glass Box Models

A glass box model is an intermediate model that reasons in terms of the same domain constructs as the human expert. However, the model reasons with a different control structure than the human expert. In this model, each reasoning step can be inspected. A classic example of a glass box model system is GUIDON (Clancey 1982), a tutoring system for medical diagnosis. This system was built around MYCIN, an expert system for the treatment of bacterial infections. MYCIN consists of several hundred "if-then" rules that probabilistically relate disease states to diagnoses. These rules reference the same symptoms and states that doctors employ in reasoning, but with a radically different control structure, i.e., an exhaustive backward search. During learning sessions, GUIDON compares the student's questions to those which MYCIN would have asked and critiques him/her on this basis.

2.3.1.3 The Cognitive Models

A cognitive model seeks to match representation formalisms and inference mechanisms with human cognition. One of the very important early findings in intelligent tutoring research was the importance of the cognitive fidelity of the domain knowledge module. Cognitive approaches aim to develop a cognitive model of the domain knowledge that mimics the way knowledge is represented in the human mind in order to make ITSs respond to problem-solving situations as the student would (Corbett et al. 1997). This approach, in contrast to the other approaches, has as objective to support cognitively plausible reasoning. In brief, it aims to apply the same style of representation to encode knowledge as that used by the learner.. A positive approach consists of building a system which uses a cognitive architecture such as Adaptive Control of Thought (ACT-R) (Anderson 1996). ACT-R is a theory for simulating and understanding human cognition. ACT-R architecture allows a system to capture in great detail the way humans perceive, think about, and act on the world. Several ITSs have been built using ACT-R (or ACT*, in its early version) production rules, including Algebra Tutor (Singley et al. 1989), Geometry Tutor (Koedinger and Anderson 1990) and LISP Tutor (Corbett and Anderson 1992). It is also generally accepted that tutors representing procedural domain knowledge based on a cognitive analysis of human skill acquisition are also cognitively oriented (Beck et al. 1996). Sherlock (Lesgold et al. 1992) is a good example of such a tutoring system.

2.3.1.4 The Need for Representation Languages

Whether the selected model is a black box model, a glass box model, or a cognitive model, careful consideration must be given to the means (languages) to be used in representing and using knowledge. These means are numerous and the representation thereby obtained may be symbolic (formal or informal), connectionist, or hybrid. As a result, it is not easy for system developers to choose a language. Their selection should take four important points into consideration: the expressivity of the language, the inference capacity of the language, the cognitive plausibility of the language (in terms of language representation, as well as reasoning) and the pedagogical orientation of the language (i.e., the manner in which the specific learning context is considered).

The expressivity of an encoding language is a measure of the range of constructs that can be used to formally, flexibly, explicitly and accurately describe the components of the domain. However, there is a compromise that has to be made between expressivity (what you can say) and complexity (whether the language is computable in real time).

Inference capacity is generally rooted in a formal semantic system and based on an inference procedure. The semantics of a language refers to the fact that the message unambiguously means what it is supposed to mean. For example, consider the language construct “A subconcept of B”: Does this mean that all instances of A are also instances of B, or parts of B, or special kinds of B? Clearly defined and well-understood semantics are essential for sound inference procedures.

Whatever language is chosen, the expert module must guarantee mutual understanding between the actions of the system and those of the learner. In fact, if the student does not understand the system's instruction or if the system cannot interpret the student's behaviour in terms of its own view of the knowledge, tutoring can be compromised (Wenger 1987).

In the following sections, we present a few examples of such languages.

2.3.2 General-Purpose Languages

In artificial intelligence, various languages and notations have been proposed for representing knowledge. These are typically based on logic and mathematics and have easily parsed grammars and well-defined semantics to ease machine processing. A logic generally consists of a syntax, a semantic system and a proof theory. The syntax defines a formal language for the logic, the semantic system specifies the meanings of properly formed expressions, and the proof theory provides a purely formal specification of the notion of correct inferences. In the following sections, we present a brief description of some common general-purpose knowledge representation languages. Further details can be found in classic AI books, such as those by Luger (2009) and Russell & Norvig (2009). Readers can also find in these books other non-classic formalisms, such as fuzzy logic, probabilistic approaches, and connectionist approaches. In general, these languages have no pedagogical emphasis, i.e., they do not make any assumptions regarding the pedagogical application of the knowledge that they represent.

2.3.2.1 Production Rules

Production rules is one of the most popular and widely used knowledge representation language. Early expert systems used production rules as their main knowledge representation language. One such example is MYCIN.

A production rule system consists of three components: a working memory, a rule base and an interpreter. The working memory contains the information that the system has acquired concerning the problem. The rule base contains information that applies to all the problems that the system may be asked to solve. The interpreter solves the control problem, i.e., it determines which rule to execute on each selection-execute cycle.

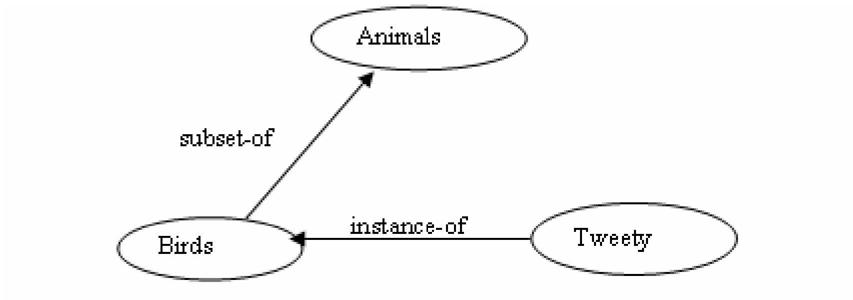

Fig. 2.1 A Simple Example of a Semantic Network

As a knowledge representation language, production rules have many advantages, including their natural expression, modularity, restricted syntax and sound logic basis for making inferences equivalent to First-order Logic (FOL). Production rules are cognitively plausible in terms of representation structures as well as reasoning mechanisms (Anderson 1982). Most cognitive tutors (see Chapter 3) encode the actions of the expert problem-solver as production rules and attempt to determine which rules the student is having difficulty applying.

2.3.2.2 Semantic Networks

An important feature of human memory is the high number of connections or associations between the pieces of information it contains. Semantic networks are one of the knowledge representation languages based on this capacity.

The basic idea of a semantic network representation is very simple: There are two types of primitives: nodes and links or arcs.. Nodes, on the one hand, correspond to objects or classes of objects in the world Links or arcs, on the other hand, are unidirectional connections between nodes that correspond to relationships between these objects. Figure 2.1 shows an example of a semantic net. The basic inference mechanism consists in following inheritant and instance links. To determine whether an object, represented by node A, is a member of a set represented by node B, every link extending upwards from A (is-a and instance link) must be traced to see whether it intersects node B. In order to determine the value of certain properties of an object represented by node A, every link extending upwards from A must be followed (as above) until it intersects a node possessing this property (function link). Many other inferences procedure was proposed including path-based and node-based inferences proposed by Shapiro (1978).

Semantic networks, which correspond to human memory, are cognitively plausible at the structural (representation) level, but not for their reasoning mechanism. In 1969, Collins & Quillian conducted a series of studies to test the psychological plausibility of semantic networks as models for both the organization of memory and human inferencing.

The domain knowledge of SCHOLAR (Carbonell 1970) is that of South America geography. This domain knowledge model is represented using a semantic network whose nodes instantiate geographical objects and concepts. Statements such as “Tell me more about Brazil” simply invoke a retrieval of facts stored in the semantic network. However, the strength of this representation schema lies in its ability to answer questions for which answers are not stored. For example, it is not necessary to store in the semantic network that “Lima is in South America,” provided that the program which interprets the network can make the relevant inference. In other words, the program must know about the attributes concerned, “location” and “capital,” and, in particular, that if X is the capital of Y and Y is located in Z, then X is in Z: This is the rule of inference.

The semantic network form of knowledge representation is especially suitable for describing the taxonomic structure of categories for domain objects and for expressing general statements about the domain of interest. Inheritance and other relationships between such categories can be represented in and derived from subsumptive hierarchies. In contrast, semantic networks are not ideal for representing concrete individuals or data values, such as numbers or strings. Another major problem with semantic networks is the lack of clear semantics for the various network representations (despite the word “semantic”) (Sharples et al. 1989). For example, Figure 2.1 can be interpreted as a representation of a specific bird named Tweety, or it can be interpreted as a representation of some relationship between Tweety, birds and animals.

Many variants of semantic networks have been used in ITS applications, including concept/topic maps (Albert and Steiner 2005; Murray 1998; Garshol 2004; Kumar 2006) and conceptual graphs (Sowa 1984). The latter is presented in the next paragraph.

2.3.2.3 Conceptual Graphs

This formalism (Sowa 1984) is based on semantic networks but is directly linked to the language of first-order predicate logic from which it takes its semantics.

A simple conceptual graph is a bipartite (not necessarily connected) graph composed of concept nodes that represent entities, attributes, states or events, and relation nodes that describe the relationships between these concepts. Each concept node c of a graph G is labelled by a pair *(type(c), ref(c)),* in which *ref(c)* is either the generic marker * corresponding to the existential quantification or an individual marker corresponding to an identifier. M is the set of all individual markers. Each relation node r of a graph G is labelled by a relation type, *type(r),* and associated with a signature identifying the constraints on the types of concepts that may be linked to its arcs in a graph. For example, the conceptual graph given in Figure 2.2 represents the information: “the experiment *E1* carries out an interaction *I1* between *Nisin* and *Listeria Scott A* in *skim milk* and the result is *reduction.*”

Concept types (respectively relation types of the same arity) create a set T_C (resp. T_R) partially ordered by a generalization/specialization relationship $\leq_c$ (resp. $\leq_r$). (T_C, T_R, M) is a lattice defining the *support* upon which conceptual graphs are constructed. A support thus represents the ontological knowledge that provides the ground vocabulary on which the knowledge base is built.

The semantics of conceptual graphs rely on the translation of both conceptual graphs and their support into first-order logic formulas. For instance, the "kind-of" relationships between types in the support are translated into logical implications.

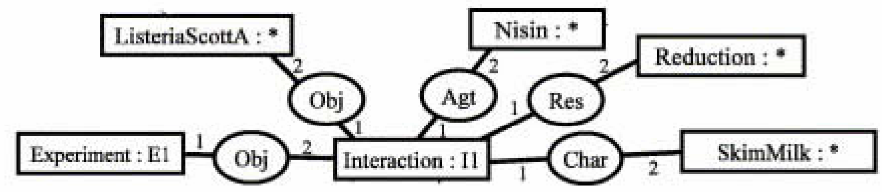

Fig. 2.2 A Sample Conceptual Graph (Adapted from Dibie-Barthélemya, Haemmerléa and Salvatc 2006).

In addition, each conceptual graph has a logical interpretation which is a first-order logic formula, in which each generic marker is associated with a distinct variable, each individual marker with a constant, each concept type with a unary predicate applied to its marker, and each relation type with a predicate applied to the markers of the concept vertices it links. The formula associated with the conceptual graph is then the existential closure of the conjunction of all atoms. For instance, the logical interpretation of the conceptual graph represented in Figure 2.2 is the following:

$$\exists x \exists y \exists z \exists t\ (\text{ListeriaScottA}(x) \wedge \text{Nisin}(y) \wedge \textit{Experiment}(E1) \wedge \text{Interaction}(I1)$$
$$\wedge \text{Reduction}(z) \wedge \text{SkimMilk}(t) \wedge \text{Obj}(I1,x) \wedge \text{Agt}(I1,y) \wedge \text{Obj}(E1,I1)$$
$$\wedge \text{Res}(I1,z) \wedge \text{Char}(I1,t)\ ).$$

The main inference procedure relies on a subsumptive relationship between two conceptual graphs. This relationship is equivalent to the logical implication between the two logical formulas corresponding to these graphs. In addition, a key operation known as projection makes it possible to compute subsumptive relationships between graphs. Reasoning with conceptual graphs is based on the projection which is complete with respect to logical deduction. However, finding a projection between two graphs is an NP-complete problem.

Many extensions of conceptual graphs have been proposed for the purpose of extending conceptual graphs expressivity with sound semantics. Some of these extensions have focused on concept descriptions, a process equivalent to those of conceptual graphs and description logics (Coupey and Faron 1998; Delteil and Faron 2002). For instance, Delteil & Faron (2002) proposed a graph-based concept description language called GDL, which involves a thorough decision-making procedure. It can be used to represent concept descriptions with complex graph patterns as well as negation and disjunction, which leads to an expressive language. GDL combines features of both conceptual graphs and description logics (see subsequent sections).

Conceptual graphs are used in some ITSs. For instance, they were used in STyLE-OLM and in HYLITE+ (Bontcheva and Dimitrova 2004). STyLE-OLM is an interactive learner modeling system that extracts extended models of the learner's cognition. HYLITE+ is a natural language generation system that creates adaptive Web pages based on a learner model. CBITS, a case-based ITS, is another example of a tutoring system that uses conceptual graphs to represent cases (Fung and Kemp 1999).

2.3.2.4 Frame-Based

Frame-based systems (Minsky 1975) are based on the notion of frames or classes which represent collections of instances (the concepts underlying ontology). Each frame has an associated collection of slots or attributes which can be filled by values or other frames. In particular, frames can have a "kind-of" slot which allows for the assertion of a frame taxonomy. This hierarchy may then be used for the inheritance of slots, thereby allowing for a sparse representation. In addition to frames representing concepts, a frame-based representation may also contain instance frames, which represent particular instances.

An example of the frame-based model is Open Knowledge Base Connectivity (OKBC), which defines an API for accessing knowledge representation systems. OKBC also defines most of the concepts found in frame-based systems, object databases and relational databases. The OKBC API (Chaudhri et al. 1998) is defined in a language-independent manner, and implementations exist for Common Lisp, Java and C.

Frames generally provide quite a rich set of language constructs but impose very restrictive constraints on how they can be combined or used to define a class. Moreover, they only support the definition of primitive concepts, and taxonomy must be hand-crafted.

COCA (Major and Reichgelt 1992) is a shell for building ITSs in which domain knowledge is represented using the frame approach with a number of user-defined attributes and attribute values. Attribute values may be primitive data types (e.g., text strings), procedures to be run, or pointers to other frames.

2.3.2.5 Ontology and Description Logics

Ontology is a formal specification of a domain and includes a definition of concepts and the relationships among them. An ontological knowledge base is composed of two parts: a terminology box (TBox), which contains terminology axioms (concepts and role descriptions), and an assertions box (ABox), containing individuals (concept and role instances). The Semantic Web community has developed a formal language for ontology implementation called Web Ontology Language (OWL).

Frames and ontology semantics are suitable for building on Description Logics (DLs). In fact, DLs (Baader et al. 2007) may be seen as a logical reformulation of frames and ontologies and may provide them with a rigorous and strong basis for reasoning. Indeed, DLs provide representation and reasoning languages with precise semantics. They also limit language expressiveness so that they can guarantee tractable inference. For instance, W3C's OWL-DL is based on *SHOIN* expressive description logic which is known to be decidable (Horrocks and Sattler 2007). A major characteristic of a DL is that concepts are defined in terms of descriptions using other roles and concepts. In this way, the model is built up from small pieces in a descriptive way rather than through the assertion of hierarchies. The DL supplies a number of reasoning services. Main inference (reasoning) tasks include subsumption, classification and satisfiability. Subsumption aims to verify whether a concept is a subset of another by comparing their definitions. Classification verifies whether an instance belongs to a concept. Satisfiability involves the consistency of a concept definition. Satisfiability is accomplished by verifying whether the membership criteria are logically satisfiable. It is the most important reasoning function as subsumption may be redefined as a satisfiability problem. The well-known tableau algorithm (Baader and Sattler 2001) is the decision procedure that implements satisfiability. These reasoning services may subsequently be made available to applications that make use of the knowledge represented in the ontology or in the frames.

There may be some drawbacks with more expressive DLs, which make them difficult to use in real world applications. These include the intractability of their satisfiability or subsumption algorithms, and the increase in the computational complexity of reasoning. However, some research results show that efficient and practical implementations of relatively expressive languages are feasible despite their theoretical complexity (Horrocks 1998; Horrocks and Sattler 2007). As DLs have clear semantics, it is possible to use all of the knowledge encapsulated in the ontology to determine whether the ontology is consistent and complete.

Ontology language is purely declarative and not overly expressive for the description of procedural knowledge. However, knowledge generally goes beyond the description of what exists in the world; it also links goals to actions (Newell 1982). In that sense, knowledge has a strongly procedural aspect. This is also true in the context of ITSs, in which learning of procedural knowledge is the main focus. Accordingly, to build an ITS that teaches a procedural task, one must not only specify the experts' conceptualizations of the domain, but also clarify how problem-solving will ideally occur. In fact, to be able to follow the learner's reasoning and to provide relevant suggestions and explanations, such ITSs must have knowledge of the task that is both robust and explicable.

There is no doubt that ontology is a good tool for representing propositional knowledge and providing shared domain descriptions for various purposes. However, it is not enough. If we wish to include problem-solving tasks for which we have enough knowledge to predict that certain solution strategies will be particularly appropriate, then we need to make the strategies explicit. We need to represent problem-solving methods that could form the basis for the procedural components of the domain knowledge base, thereby making the system more understandable and traceable (Brewster and O'Hara 2007). Task ontology may help achieve this. However, the notion of task ontology should be applied prudently. In fact, ontology should only be used declaratively to specify conceptual building blocks that provide the foundation on which the knowledge base is built. In this sense, the role of ontology should be strictly limited to specifying hidden conceptualization and not be used as a tool for the in-depth definition of procedural knowledge and rules. Such a definition should be built on top of the domain ontology. As a result, task ontology aims to provide relevant terminology that is both necessary and sufficient for building problem-solving models in a given domain (Mizoguchi et al. 1995). The problem-solving model (e.g., task decomposition in terms of goals, sub-tasks and control-flow) needs this task ontology to relate the problem-solving steps to the relevant knowledge in the domain ontology. This idea is consistent with the current practice in semantic web, in which there is a clear separation between the ontology layer and the rule layer. Semantic Web Rule Language (SWRL) is a standard language that extends OWL to include Horn-like rules, making it possible to define procedural knowledge linked to the domain ontology as shown in Chi (2010).

In conclusion, there is no standard means of correctly integrating procedural knowledge and linking it to the declarative domain knowledge. What is important is the fact that these two levels of knowledge should be "loosely coupled" to make computation easier (as in ACT-R).

2.3.3 Pedagogically-Oriented Languages

Several other languages with a significant pedagogical emphasis may be used to represent domain knowledge in a less formal manner. These languages are generally associated with a precise characterization of the knowledge as well as the pedagogical functions that enable learning. We present two examples below.

MOT (Paquette 2008) is one example of such a language. It provides the user with four basic types of knowledge units in graphical form: facts, concepts, procedures and principles (see Figure 2.3).

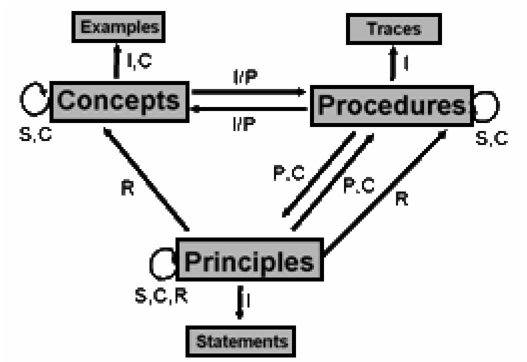

Fig. 2.3 Types of Knowledge in MOT

Knowledge units are implemented as schemas and are connected using six types of links constrained by specific grammar rules. Two examples of these rules are: 1) All abstract knowledge entities (concepts, procedures and principles) may be related by an instantiation link to a set of facts representing individuals called respectively examples, traces and statements. 2) All abstract knowledge entities (concepts, procedures and principles) may be specialized or generalized to other abstract knowledge by means of specialization links. Knowledge units connected by means of links form the domain knowledge model which can be exported into OWL to "become" a formal ontology model. Domain knowledge produced with MOT was used mainly as a semantic basis for learning object semantic referencing.

CREAM language (Nkambou et al. 2001) is another example of a pedagogically- oriented language. It allows for the creation and organization of curriculum elements in three interconnected perspectives: domain, pedagogy and didactic. The domain perspective is represented by a graphical structure of semantic knowledge based on Gagne's taxonomy (1985) and connected by means of semantic links. The pedagogical perspective organizes learning objectives on the basis of Bloom's taxonomy and pre-requisite links. The didactic perspective defines the model of learning resources that supports learning activities. These three perspectives are connected in a complex bi-partite graph that serves as the basis for making tutoring decisions related to content planning and learning resource recommendations. Figure 2.4 shows the basic components in CREAM.

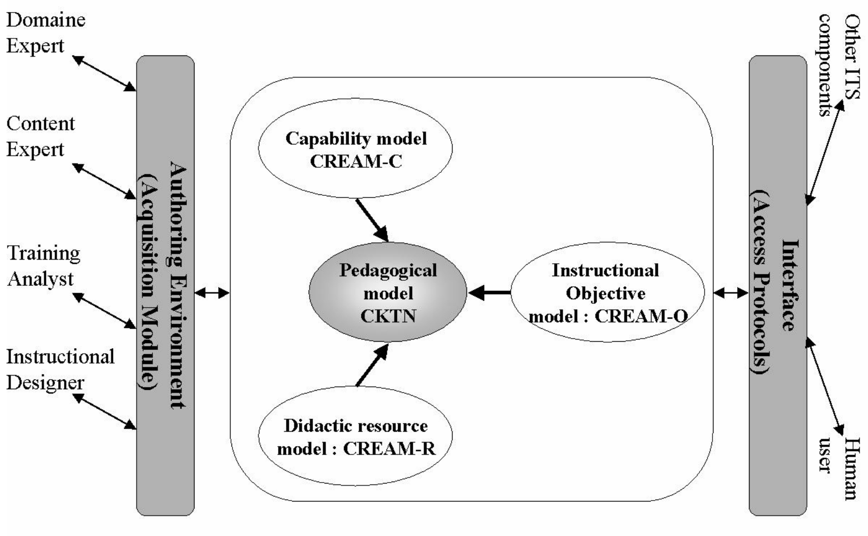

Fig. 2.4 Basic Components in CREAM

The limits of these pedagogically-oriented languages can be attributed to their reduced expressivity as well as their strong pedagogical orientation. The meta-model of the epistemological level is determined in advance (for MOT, see Figure 2.3) with no possibility of modification or extension via other types of objects; conversely, general languages (Conceptual Graphs and ontology) do provide the possibility of determining the epistemological level, e.g., concept and relation types. Moreover, pedagogically-oriented languages that are not based on a precise semantic system are relatively informal in nature. In the case of MOT, even though exporting to an ontological language is possible, it is clear that its limited expressivity is insufficient to describe complex problems (such as defining concepts based on primitive concepts). In addition, the use of unrestrained graphic language does not ensure that the resulting axioms are logically sound and semantically valid. Accordingly, the inference procedure cannot make any guarantees (neither completeness nor completion within a reasonable time). Note that for the two languages cited as examples, cognitive plausibility is not an issue and has no bearing.

For these reasons, it is difficult to see how such languages could be adequate for use in an open ITS. Their relatively meagre semantics do not give rise to any logical foundation on the basis of which inferences may be made regarding the constructed models. However, they may prove to be very effective in certain very specific contexts. For instance, in the case of MOT, the resulting structure serves as a valuable "semantic" base which is available to developers for the purpose of learning design or learning object semantic annotation. In addition, a knowledge model like CREAM can support effective content planning in order to meet specific training needs (Nkambou et al. 1998).

2.4 Conclusion

Although numerous solutions exist to represent domain knowledge and the reasoning mechanisms that operate them, there is no consensus on what can be considered an ideal approach. Even if such a consensus may be difficult to achieve, current approaches need to be analyzed using well-defined criteria. Such analyses could advance ITS developers' knowledge and make their choices easier. In this introductory chapter, we first pointed out the importance of determining the nature and value of domain knowledge and subsequently described some existing solutions for representing the expert module of an ITS in light of four criteria: expressivity, inferential power, cognitive plausibility and pedagogical considerations. We also stated that the AIED community should determine relevant criteria to be considered in selecting a representation language for a given tutoring context.

Furthermore, and in this era of Semantic Web and Web 2.0, ontology can easily be defended as the formalism of choice for several reasons: 1) it offers several levels of expressiveness that can be adapted to needs and inferential constraints; 2) it allows for the integration of multiple views and enables interoperability; and 3) it is built on well-defined semantics resulting from description logic, which provides for sound reasoning mechanisms. However, ontology remains essentially a declarative approach used in the creation of the domain semantic memory. For a procedural learning domain, such semantic memory has to be supplemented by operational and procedural knowledge that refers to its elements. Therefore, another language is needed to build this procedural level on top of the ontology level.

The next three chapters in this part explain specific techniques for representing the expert module. The following two chapters describe two approaches in the field of AIED that are gaining in popularity. First, Chapter 3 presents the "cognitive-tutor" approach which is based on the need for balance between the domain structure and a theory of cognition (in this case, ACT-R). Second, Chapter 4 describes the CBM approach which is based on the need to focus not on an explicit representation of all elements of a domain, but on a definition of a set of principles as constraints that govern the field. One can question the validity of these two approaches in ill-defined domains. Chapter 5 discusses this case and presents appropriate solutions. Finally, given the importance of the ontology-based approach, Chapter 6 describes how ontology can serve as domain knowledge. Chapter 6 also focuses on how such domain ontology can be automatically learned (extracted) from textual learning resources with a very limited solicitation of domain experts.

The current chapter does not address the problem of the acquisition of domain knowledge. Indeed, regardless of the nature of knowledge, the approach or the representation language, the knowledge that the system uses must be acquired. Such knowledge is usually held by human experts. It can also be found in knowledge sources such as documents. Knowledge acquisition is a major concern in the field of AI and several solutions have been proposed, including automatic extraction approaches using machine-learning or data-mining techniques. The other chapters in this part describe how this problem is addressed in the context of ITSs. Specific tools are generally provided for facilitating the domain knowledge acquisition process. For example, CTAT (cf. Chapter 3) provides tools to facilitate the creation of domain knowledge in the form of production rules. ASPIRE (cf. Chapter 4) provides tools that simplify the implementation of constraints in a given domain. Moreover, automatic learning tools are sometimes used for the acquisition of domain ontology (cf. Chapter 6) or procedural knowledge (cf. Chapter 5). Part 4 of this book describes other examples of authoring tools to facilitate the development of ITSs and their expert modules.

References

Albert, D., Steiner, C.M.: Representing domain knowledge by concept caps: How to validate them. In: Okamoto, T., Albert, D., Honda, T., Hesse, F.W. (eds.) The 2nd Joint Workshop of Cognition and Learning through Media-Communication for Advanced e-Learning, Tokyo (2005)

Anderson, J.R.: Acquisition of cognitive skill. Psychological Review 98(4), 369–406 (1982)

Anderson, J.R.: The architecture of cognition. Lawrence Erlbaum Associates Inc., NJ (1996)

Baader, F., Sattler, U.: An overview of Tableau algorithms for description logics. Studia Logica: An International Journal for Symbolic Logic 69(1), 5–40 (2001)

Baader, F., Horrocks, I., Sattler, U.: Description Logics. In: Harmelen, F., Lifschitz, V., Porter, B. (eds.) Handbook of Knowledge Representation. Elsevier, Amsterdam (2007)

Beck, J., Mia, S., Haugsjaa, E.: Applications of AI in education. Crossroads, Special issue on artificial intelligence 3(1), 11–15 (1996)

Bloom, B.S.: Taxonomy of Educational Objectives: the Classification of Educational Goals. McKay D, New York (1975)

Bontcheva, K., Dimitrova, V.: Examining the use of conceptual graphs in adaptive web-based systems that aid terminology learning. International Journal on Artificial Intelligence Tools 13(2), 299–331 (2004)

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

Brachman, R.J., Levesque, H.J.: Knowledge Representation and Reasoning. Morgan Kaufmann, San Francisco (2004) ISBN 978-1-55860-932-7

Brewster, C., O'Hara, K.: Knowledge representation with ontologies: Present challenges - Future possibilities. International Journal of Man-Machine Studies 65(7), 563–568 (2007)

Brown, J.S., Burton, R.R.: SOPHIE: a pragmatic use of artificial intelligence in CAI. In: Proceedings of the 1974 annual ACM conference, vol. 2, pp. 571–579 (1974)

Carbonell, J.: AI in CAI: An artificial intelligence approach to computer- assisted instruction. IEEE Transactions on Man-Machine Systems 11, 190–202 (1970)

Chaudhri, V.K., Farquhar, A., Fikes, R., Karp, P.D., Rice, J.P.: OKBC: A Programmatic Foundation for Knowledge Base Interoperability. In: Proceedings of AAAI, pp. 600–607. AAAI Press, Menlo Park (1998)

Chi, Y.-L.: Rule-based ontological knowledge base for monitoring partners across supply networks. Expert Systems with Applications 37, 1400–1407 (2010)

Clancey, B.: Acquiring, representing and evaluating a competence model of diagnostioc strategy. Technical Report: CS-TR-85-1067, Stanford University (1985)

Clancey, W.J.: Overview of GUIDON. Journal of Computer-Based Instruction 10(1-2), 8–15 (1982)

Collins, A.M., Quillian, R.: Retrieval time from semantic memory. Journal of Verbal Learning and Verbal Behavior 8, 240–247 (1969)

Corbett, A.T., Anderson, J.R.: The LISP intelligent tutoring system: Research in skill acquisition. In: Larkin, J., Chabay, R., Scheftic, C. (eds.) Computer Assisted Instruc-tion and Intelligent Tutoring Systems: Establishing Communication and Collaboration. Erlbaum, Hillsdale (1992)

Corbett, A.T., Koedinger, K.R., Anderson, J.R.: Inteligent Tutoring Systems. In: Helander, T.K., Landauer, P. (eds.) Handbook of Human-Computer Interaction. Elsevier Science, Amsterdam (1997)

Coupey, P., Faron, C.: Towards Correspondences Between Conceptual Graphs and Description Logics. In: Mugnier, M.-L., Chein, M. (eds.) ICCS 1998. LNCS (LNAI), vol. 1453, p. 165. Springer, Heidelberg (1998)

De Jong, T., Ferguson-Hessler, M.G.M.: Types and qualities of knowledge. Educational Psychologist 31(2), 105–113 (1996)

Delteil, A., Faron, C.: A Graph-Based Knowledge Representation Language for Concept Description. In: ECAI 2002, pp. 297–301 (2002)

Fung, P.-W., Kemp, R.H.: On developing case-based tutorial systems with conceptual graphs. In: Tepfenhart, W.M. (ed.) ICCS 1999. LNCS, vol. 1640, pp. 214–229. Springer, Heidelberg (1999)

Gagné, R.M.: The Conditions of Learning, 4th edn. Holt Rinehart and Winston, New York (1985)

Garshol, L.M.: Metadata? Thesauri? Taxonomies? Topic Maps! Making Sense of it all. Journal of Information Science 30(4), 378–391 (2004)

Hickman, F.R., Killin, J.L., Land, L., Mulhall, T., Porter, D., Taylor, R.: Analysis for Knowledgebased Systems. In: A Practical Guide to the KADS Methodology. Ellis Horwood, New York (1989)

Horrocks, I.: Using an Expressive Description Logic: FaCT or Fiction? In: Proc. of the 6th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 1998), pp. 636–647 (1998)

Horrocks, I., Sattler, U.: A Tableau Decision Procedure for SHOIQ. J. of Automated Reasoning 39(3), 249–276 (2007)

Koedinger, K.R., Anderson, J.R.: Abstract planning and perceptual chunks: Elements of expertise in geometry. Cognitive Science 14, 511–550 (1990)

Kumar, A.: Using Enhanced Concept Map for Student Modeling in a Model-Based Programming Tutor. In: Proc. of 19th International FLAIRS Conference on Artificial Intelligence, Melbourne Beach, FL (2006)

Kyllonen, P.C., Shute, V.J.: A Taxonomy of Learning Skills. Brooks Air Force Base, TX, AFHRL Report No. TP-87-39 (1988)

Lesgold, A., Lajoie, S., Bunzo, M., Eggan, G.: SHERLOCK: A coached practice environment for an electronics troubleshooting job. In: Larkin, J., Chabay, R. (eds.) Computer assisted instruction and intelligent tutoring systems: Shared issues and complementary approaches, pp. 201–238. Lawrence Erlbaum Associates, Hillsdale (1992)

Luger, G.F.: Artificial Intelligence: Structures and Strategies for Complex Problem Solving, 6th edn. Addison-Wesley Pearson Education, Boston (2009)

Major, N., Reichgelt, H.: COCA: A shell for intelligent tu-toring systems. In: Frasson, C., McCalla, G.I., Gauthier, G. (eds.) ITS 1992. LNCS, vol. 608, pp. 523–530. Springer, Heidelberg (1992)

Merrill, D., Twitchell, D.: Instructional Design theory. Educational Technology Publication Inc., Englewood Cliff (1994)

Merrill, M.D.: An Introduction to Instructional Transaction Theory. Educational Technology 31, 45–53 (1991)

Minsky, M.: A Framework for Representing Knowledge. In: Winston, P.H. (ed.) The Psychology of Computer Vision. McGraw-Hill, New York (1975)

Mizoguchi, R., Vanwelkenhuysen, J., Ikeda, M.: Task Ontology for Reuse of Problem Solving Knowledge. In: Mars, N.J.I. (ed.) Towards Very Large Knowledge Bases. IOS Press, Amsterdam (1995)

Murray, T.: Authoring Knowledge Based Tutors: Tools for Content, Instructional Strategy, Student Model, and Interface Design. J. of the Learning Sciences 7(1), 5–64 (1998)

Newell, A.: The Knowledge Level. Artificial Intelligence 18(1), 87–127 (1982)

Nkambou, R., Frasson, C., Gauthier, G.: A new approach to ITS-curriculum and course authoring: the authoring environment. Computers and Education 31(2), 105–130 (1998)

Nkambou, R., Frasson, C., Gauthier, G., Rouane, K.: An Authoring Model and Tools for Knowledge Engineering in Intelligent Tutoring Systems. Journal of Interactive Learning Research 12(4), 323–357 (2001)

Nwana, H.: Intelligent Tutoring Systems: an overview. Artificial Intelligence Review 4, 251–277 (1990)

Paquette, G.: Graphical Ontology Modeling Language for Learning Environments. Technology, Instruction., Cognition and Learning 5, 133–168 (2008)

Peters, J.M., Gray, A.: Epistemological perspectives in research on teaching and learning science. Educational Research 5(1), 1–27 (2006)

Piaget, J.: The Principles of Genetic Epistemology. Routledge, London (1997)

Ramoni, M.F., Stefanelli, M., Magnani, L., Barosi, G.: An epistemological framework for medical knowledge-based systems. IEEE Trans. Syst. Man Cybern. 22, 1–14 (1992)

Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach, 3rd edn. Prentice Hall, Upper Saddle River (2009)

Schraw, G.J., Olafson, L.: Assessing Tecahers' Epistemological and Ontological World Views. In: Khine, M.S. (ed.) Knowing, Knowledge and Beliefs, pp. 24–44. Springer, Heidelberg (2008)

Shapiro, S.C.: Path-based and node-based inference in semantic networks. In: Proceedings of the 1978 workshop on Theoretical issues in natural language processing. ACM-SIGART, pp. 219–225 (1978)

Sharples, M., Hogg, D., Hutchison, C., Torrance, S., Young, D.: Computers and Thought: A practical Introduction to Artificial Intelligence. MIT Press, Cambridge (1989)

Singley, M.K., Anderson, J.R., Gevins, J.S., Hoffman, D.: The algebra word problem tutor. Artificial Intelligence and Education, 267–275 (1989)

Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley, Reading (1984)

Sowa, J.F.: Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks Cole Publishing Co., Pacific Grove (2000)

Tennyson, R., Rasch, M.: Linking cognitive learning theory to instructional prescriptions. Instructional Science 17, 369–385 (1988)

Wenger, E.: Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge. Morgan Kaufmann Publishers, Inc., Los Altos (1987)

Woolf, B.: Building Intelligent Interactive Tutors: Student-centered strategies for revolutionizing e-learning. Morgan Kaufmann, San Francisco (2008)

Winograd, T., Fernando, F.: Understanding Computers and Cognition: A New Foundation for Design. Ablex, Norwood (1986)