ELSEVIER Artificial Intelligence in Medicine 7 (1995) 227–255

Artificial Intelligence in Medicine

A case study in ontology library construction

Gertjan van Heijst [a,*], Sabina Falasconi [b], Ameen Abu-Hanna [a], Guus Schreiber [a], Mario Stefanelli [b]

[a] *University of Amsterdam, Social Science Informatics, Roetersstraat 15, NL-1018 WB Amsterdam, The Netherlands*

[b] *University of Pavia, Medical Informatics, Via Abbiategrasso 209, I-27100, Pavia, Italy*

Accepted 13 January 1995

Abstract

The goal of our work is to facilitate the development of medical knowledge-based systems by providing a library of reusable ontologies. The availability of such a library reduces the amount of knowledge acquisition required to create knowledge bases of new applications, and makes it easier to connect a knowledge-based system to existing data bases. This article presents a case study in constructing such a library. The emphasis is on studying the principles that underly the internal structure of the library as well as on the process of constructing and using the library. We envision that, in the future, application ontologies can be constructed by the selection and refinement of generic ontologies and domain ontologies from such a library.

Keywords: Knowledge-based systems; Library; Ontology; Reusability

1. Introduction

Many authors in the field of knowledge acquisition for knowledge-based systems (KBS) have emphasized the importance of reusable components to reduce the effort required for KBS development (e.g. [17]). Two main types of components have been identified as potentially sharable and reusable: (i) problem-solving methods, abstract descriptions of the steps that must be taken to perform a particular task, and (ii) domain ontologies, abstract descriptions of the structure of domain knowledge in a particular field. Most present approaches share the view

[*] Corresponding author. E-mail: gertjan@swi.psy.uva.nl

that reuse of these components is facilitated by the use of KBS architectures that keep the problem-solving methods and the domain ontologies separated. Fig. 1 shows an example of a KBS that is organized according to these principles.

Until recently, most researchers in the field have concentrated on the domain-independent specification of problem-solving methods [3,14,30,36]. The availability of abstract models of the methods to perform a real-world task has proven to be very useful for knowledge acquisition. Since the model of a method determines which knowledge is required to perform a particular task, it can be used to direct the knowledge-acquisition process. This is often called the model-driven approach to knowledge acquisition. For example, MOLE [6], a knowledge-acquisition tool for systems that use the cover-and-differentiate problem-solving method, uses its knowledge of the domain knowledge requirements for this method to focus the knowledge-acquisition dialogue. The early successes of the model-driven approach to knowledge acquisition have inspired other researchers to investigate other problem-solving methods [31], the configuration of problem-solving methods from smaller building blocks [11,22], and the formalization of such building blocks [1]. The article of Smith et al. [28] in this issue describes experiences with reusable problem-solving methods for medical applications.

The specification of reusable domain ontologies has received much less attention in the literature. Whereas KBS developers nowadays have a good chance to find appropriate, or at least usable, problem-solving methods for their applications in the literature, it is unlikely that they will find reusable domain ontologies. The reluctance to take up the challenge of creating libraries of reusable domain ontologies is, in our view, due to two reasons: the *hugeness problem* and the *interaction problem*. The hugeness problem concerns the overwhelming amount of knowledge there is in the world. This makes the construction of a library of reusable domain ontologies a daunting exercise. The interaction problem, formulated by Bylander and Chandrasekaran [4], states that domain knowledge cannot be represented independent of assumptions of how it will be used in reasoning.

Although these problems provide severe impediments for the development of libraries of reusable domain ontologies, the potential gains are high: collecting domain knowledge is by far the most cumbersome and time-consuming step in the knowledge-engineering process. In this article, a number of hypotheses about the nature of medical domain knowledge are put forward, from which principles are derived for organizing a library in such a way that the hugeness problem and the interaction problem remain manageable. In short, these principles are that (i) there is a relatively small set of basic concepts that are reusable across many medical domains and tasks, (ii) medical subdomains have domain-specific concepts that are often specializations of the basic medical concepts, and (iii) many problem-solving methods require additional concepts that are specific for that method.

The presented principles are illustrated with examples from a case study in constructing a library of reusable medical ontologies that was performed in the context of the GAMES-II project, a project funded by the European Community that aims to develop a methodology for constructing medical KBS. The article is organized as follows. In Section 2, a classification of different types of ontologies is discussed. Section 3 describes the role of an ontology library within the broader framework of the GAMES approach to knowledge engineering. Section 4 describes the organizational principles that the library is based on, thereby showing how the hugeness problem and the interaction problem can be addressed. Section 5 shows how these principles are used to build an initial library and Section 6 shows how such a library facilitates KBS development. Section 7 presents some preliminary conclusions and points at future research issues.

2. Ontologies

The term "ontology" is often used in recent AI literature, and not always in the same way. To avoid confusion, we present our interpretation of the term here, together with a typology of different kinds of ontologies that can be distinguished.

According to Gruber [9] an ontology is a "specification of a conceptualization". It defines the vocabulary of a domain and constraints on the use of terms in the vocabulary. Ontologies, can be classified according to two dimensions: the amount and type of structure of the conceptualization and the subject of the conceptualization. With respect to the first dimension, three categories are distinguished:

- *Terminological ontologies* such as lexicons, specify the terms that are used to represent knowledge in domain of discourse. An examples of such an ontology in the medical field is the semantic network in UMLS (Unified Medical Language System [13]).
- *Information ontologies* which specify the record structure of databases. Conceptual schemata of databases are an example of this class of ontologies. Level 1 of the PEN & PAD model [24], a framework for modeling medical records of patients, is a typical example of such an ontology in the medical field. At this level, the model provides a framework for recording the basic observations of patients, but it makes no distinction between symptoms, signs, treatments etc.
- *Knowledge modeling ontologies* specifying conceptualizations of the *structure* of the knowledge. Compared to information ontologies, which usually have a flat structure, knowledge modeling ontologies have a richer internal structure. Further, these ontologies are often tuned to a particular use of the knowledge that they describe. Within the context of KBS development, knowledge modeling ontologies are the ontologies that we are mostly interested in. The level 2 description of the above-mentioned PEN & PAD model is an example of a knowledge modeling ontology in the medical field. At this level, the level 1 observations are grouped to describe the decision-making process.

The other dimension on which ontologies can be differentiated is the subject of the conceptualization. In the reported study, four categories were distinguished on this dimension: (i) domain ontologies, (ii) generic ontologies, (iii) representation ontologies and (iv) application ontologies. [1]

[1] In [7], two additional categories are discerned: *task ontologies* and *method ontologies*. The nature of these types of ontologies is outside the scope of this paper.

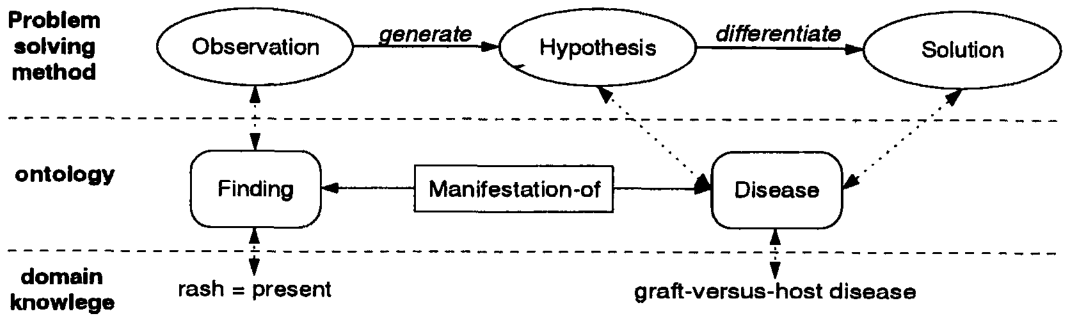

Fig. 1. The relation between problem-solving methods, ontologies and domain knowledge, illustrated for the generate-and-differentiate problem-solving method and a simple conceptualization of a medical domain. The ellipses in the problem-solving method represent knowledge roles, domain-independent labels that specify the role of domain expressions in the reasoning process. The arrows between the roles and the domain classes, depicted as rounded boxes, represent role limitations: only instances of the class disease may play the role of hypothesis in the reasoning process.

- *Domain ontologies* express conceptualizations that are specific for particular application areas. As indicated in Fig. 1, current knowledge engineering methodologies make an explicit distinction between *domain ontologies* and *domain knowledge*. Whereas the domain knowledge refers to particular states of affairs in a certain domain (e.g. chest pain is a manifestation of atherosclerosis), the domain ontology puts constraints on the structure of domain knowledge expressions (e.g. diseases have findings as manifestations).
- *Generic ontologies* are similar to domain ontologies, but the concepts that they define are considered to be generic across many fields. Typically, generic ontologies define concepts like state, event, process, action, component etc. The concepts in domain ontologies are often defined as specializations of concepts in generic ontologies. Of course, the borderline between generic ontologies and domain ontologies is vague, because there is no exhaustive enumeration of domains and their conceptualizations. However, the distinction is intuitively meaningful and is useful for building libraries for other domains.
- *Representation ontologies* explicate the conceptualizations that underly knowledge representation formalisms. They are supposed to be *neutral* with respect to world entities [10]. That is, they provide a representational framework without making claims about the world. Domain ontologies and generic ontologies are described using the primitives provided by representation ontologies. An example in this category is the *Frame Ontology*, which is used in Ontolingua [9].
- *Application ontologies* are a special category of ontologies. Application ontologies contain all the definitions that are needed to model the knowledge that is required for a particular application. Application ontologies are not necessarily reusable themselves. They may be obtained by selecting theories from the ontology library, which are then fine-tuned for the particular application. We use the term application ontology in a similar way as in PROTÉGÉ-II (Tu et al. [29], this issue).

The library presented in Section 4 consists of generic ontologies and domain ontologies of the knowledge modeling type.

3. The role of an ontology library in knowledge engineering

The goal of this article is to facilitate the development of medical KBS by providing a library of reusable domain ontologies. The underlying assumption is that the availability of such a library will reduce the amount of knowledge acquisition required to fill the knowledge bases of new applications, and make it easier to connect the KBS to existing servers. To fulfill such a role, the library must fit into the framework of a knowledge engineering methodology. As mentioned in Section 1, for our work this framework is provided by the GAMES approach.

GAMES views the knowledge engineering process as the construction of two models: the *epistemological model*, which is a model of the knowledge that is required to perform a particular task, and the *computational model*, which is a

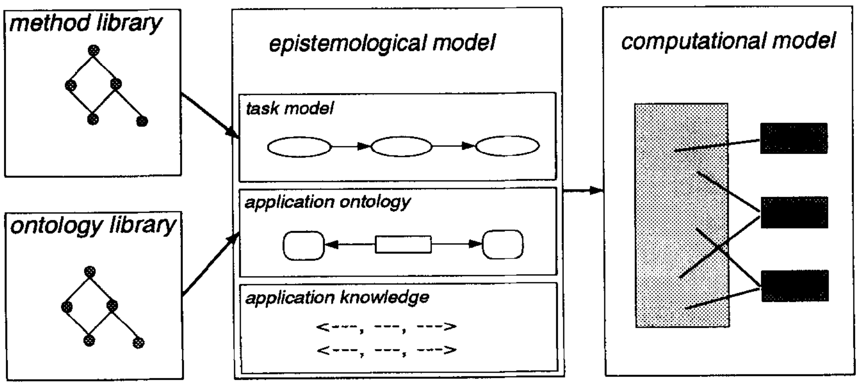

Fig. 2. The role of the ontology library within the GAMES approach.

model of the data structures and algorithms required to have a computer performing that task. The epistemological model is comprised of three parts: (i) the task model, which describes the problem-solving steps that are needed to perform the task, (ii) the application ontology, a description of the structure of the domain knowledge that is required to perform the task, and (iii) the application knowledge, which describes the actual "states of affairs" in the domain. Fig. 2 shows the different models in the GAMES approach [33].

There are two reasons for the explicit distinction between the application ontology and the application knowledge [16]. Firstly, the construction of an application ontology requires another kind of expertise than instantiation of the application ontology. Whereas the construction of the application ontology requires expertise in knowledge modeling, and must therefore be done by a knowledge engineer, instantiation can often be done by a domain expert. Secondly, the application ontology and the application knowledge are reusable under different circumstances.

The role of the ontology library is to facilitate the construction of application ontologies. These application ontologies are then used for three different purposes: (i) they are used to facilitate the acquisition of application knowledge, (ii) they are used to integrate the KBS with existing servers in the application environment, and (iii) they are used to select appropriate data structures in the computational model. The acquisition of application knowledge is facilitated because the application ontology can be used by knowledge-acquisition tools to communicate with domain experts in domain-specific terminology, when collecting the domain knowledge. Integration with external servers is facilitated because the intensional description of the contents of the knowledge base in the application ontology allows specification of mappings between the domain knowledge and the data structures of external servers without actually referring to the domain knowledge.

4. Organization of the library

This section describes the structuring principles that our library is based on. In short, these principles are that (i) there are some general categories of medical knowledge that are fundamental to all kinds of medical reasoning, (ii) in many application domains, there are additional ontological distinctions that are specific for that domain, and (iii) the use of specific reasoning methods may require additional method-specific ontological distinctions. Based on these principles, the library is partitioned into two regions: a core library, which contains definitions of the general medical categories, and a peripheral part, which contains definitions of the domain and method-specific extensions. Section 4.2 describes the core library and in Section 4.3 the peripheral parts are explained. Before turning to a more elaborate description of these parts, first some general issues in library construction are addressed.

4.1. Issues in library construction

Language. Ontologies need to be specified in a language. A number of languages have been proposed as candidates for such a language (e.g. small MODEL [29], CML [26]) but it is not entirely clear at the moment which requirements such a language should satisfy. The presented library is developed with Ontolingua [9]. An Ontolingua ontology consists of a number of *definitions*, collections of labeled sentences that constrain the use of a term. Four kinds of definitions are distinguished: classes, relations, functions and instances. Definitions can be grouped into *theories*, collections of definitions that are somehow related. Theories can include other theories, which means that all the definitions in the included theory are also available in the including theory. Thus, the theory is the main modularity construct available, and is therefore the principal building block of the library that is described below.

Modularity. A key to successful library organization is modularity. A modular organization is one that organizes units in modules so that the cohesion within modules is maximal, while the interaction between modules is minimal. In the context of the presented ontology library, the units are definitions and the modules are theories. There are numerous possible cohesion criteria. Which of these are useful in this context depends on the intended use of the library.

The main intended use of the library is to support the construction of application ontologies. Therefore, definitions that are likely to be used in the same application ontologies should be put together into one theory. There are three features that determine which definitions are needed for an application ontology: (i) the *medical subdomain* that the application should reason about, (ii) the *method* that the application uses to perform a (sub-)task, and (iii) the ontological requirements of the *external servers* that the application will be connected to. The first two features speak for themselves: applications in the domain of cardiac diseases use (at least partially) other knowledge than that used by applications in the domain of bacterial diseases; similarly, applications that *diagnose* cancer are likely to use different knowledge than do applications that *plan* cancer therapy.

The third feature needs some further elaboration. It has often been noted that for medical KBS to be put into routine use, it is necessary to integrate them with conventional databases that are already being used in hospitals (e.g. [32]). In hospitals one may find patient databases, containing patient records, drug databases etc. To support such integration, the library structure should facilitate the specification of mappings from the application ontology onto elements of external database schemata, and thus facilitate the realization of interfaces between the systems. In order to support these mappings it is useful to center theories around entities that have direct counterparts in these external servers. The identification of such entities is an exercise in reverse engineering. In this sense, the construction of the library is reminiscent of the way that the semantic network in UMLS has been constructed. The concept frames in this semantic network are used as an interlingua between clinical vocabularies as QMR [15] and HELP PTXT [21]. As knowledge-based systems are likely to be connected to such systems, the library should contain theories that can easily be mapped onto the ontological distinctions they make.

Alternative definitions. It is important to stress that the library is not intended as *the* ontology of medical knowledge; the definitions are not claimed to capture the essence of knowledge categories in some platonic sense. Instead, the definitions should be viewed as conceptualizations that have been proven useful for solving medical problems, either by human experts or by computers. A consequence of this pragmatic point of view is that it is sometimes necessary to allow for alternative, or even inconsistent, definitions of a concept in the library. For example, an often used concept in medical reasoning is "causality". Since this concept is reusable across many applications, it is an obvious candidate for inclusion in the library. However, the history of philosophy shows that it is extremely difficult to come up with a satisfying definition of causality. When we look at medical reasoning, it seems that a number of alternative conceptualizations are being used. For example, in some cases both the cause and the effect roles of the ***causes*** relation are constrained to be physiological states, while in other cases they need to be events. The temporal aspects of the concept may vary as well; in some cases the relation between cause and effect is immediate, while in others there may be a delay. Because these alternative conceptualizations are useful in medical reasoning, we have chosen to allow multiple definitions of the same concept, leaving the decision of which conceptualization is appropriate in a particular context to the library user.

The need for a higher order language. The requirements of a modular organization and multiple concept definitions make it necessary to allow higher order expressions in the ontology specification. The principle of modularity requires that the more generic aspects of a concept are defined in a core library theory, while the more domain- or method-specific aspects of those concepts are defined in a more peripheral theory. To take the previous example, assume that in a core theory, ***causes*** is defined as a binary predicate that takes states as arguments:

causes(< ***state1*** > , < ***state2*** >)

For some method in some domain, the definition of the causal relation needs to be augmented with a notion of time delay. The typical first-order solution to do this would be to add a third parameter to ***causes***:

causes(< ***state1*** > , < ***state2*** > , < ***delay*** >)

However, it is clear that the introduction of an extra parameter violates the earlier mentioned minimal interaction principle, and thus the principle of modularity. The addition of the time delay parameter leads to the destruction of the internal structure of the generic definition of ***causes***, with the result that all the definitions that rely on the definition of ***causes*** need to be updated as well. To avoid this, the domain- and task-specific specializations must be specified by means of higher order expressions, such as the following, where ***causes-tuple*** refers to a tuple in the extension of the causes relation:

causes-with-delay(< ***causes-tuple*** > , < ***delay*** >)

Unfortunately, allowing higher orders introduces some well known difficulties. Firstly, higher order languages are not decidable, thus it is impossible to have a system that can prove the internal consistency of the ontological theories. Secondly, the use of a higher order language introduces the risk of self referential sentences and the paradoxes that they give rise to. However, since the language will be used for library construction, and not for reasoning, we allow the modularity argument to prevail.

4.2. Basic categories of medical domain knowledge

This section describes the core part of the library, which contains definitions that are considered reusable across many medical domains and medical tasks. Fig. 3 shows a part of the theory structure of this section of the library, in the form of a *theory inclusion graph*. The nodes in the graph represent ontological theories, and the edges denote inclusion relations. Each arrow points from an including theory to an included theory. When a theory includes another theory, this means that all the definitions in the included theory are also available in the including theory.

4.2.1. Criteria for partitioning definitions

The decisions about the partitioning of definitions into theories are based on two considerations which we describe further below: (i) the definitions are to be centered around some "natural categories", and (ii) the number of theory inclusions must be kept to a minimum.

Center definitions around natural categories. The main criterion for partitioning the definitions into theories is based on the observation that there are some, but not too many, basic categories of medical knowledge. These categories are natural in

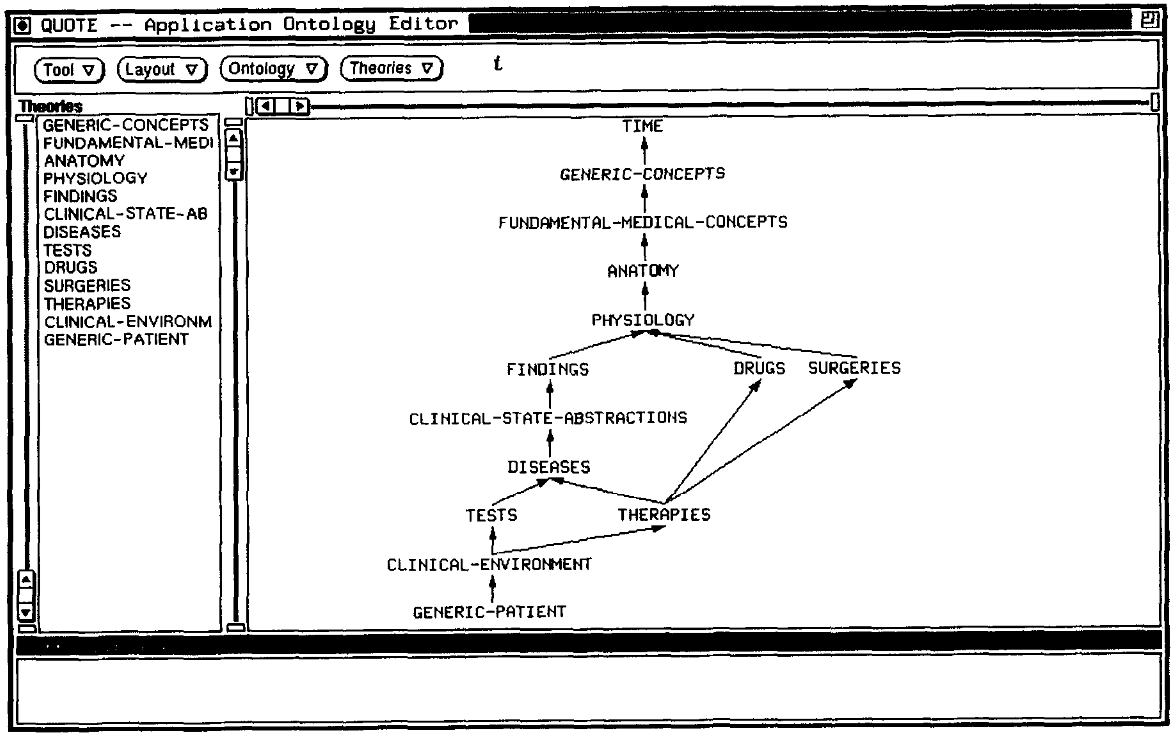

Fig. 3. Theory inclusion graph of the theories defining the basic categories of medical knowledge. This figure shows a graphical editor that was built for handling the construction of an application ontology from a set of Ontolingua theories. Each node in the graph represents a theory, with its own set of definitions.

the sense of Rosch [25], in that they reflect a social consensus that exists in the medical community. Examples of natural categories in the medical domain are concepts such as ***patient, disease, therapy*** etc. These concepts provide a coherent body of terminology that allows medical professionals from different specialties to communicate. These categories recur in almost all medical literature, and they often provide starting points for information analyses for software development.

The natural categories are used as anchor points for modularizing the core library. For instance, the theory diseases is centered around the concept of ***disease***, which is represented as an Ontolingua class. On the instances of this class several relations are defined. These definitions, such as ***disease-etiology*** and ***disease-location***, are also located in the diseases theory, since they have no meaning independent of the meaning of disease. The current organization of the domain theories, as shown in Fig. 3, is based on the knowledge categories that are discerned in a number of existing expert systems (e.g. M-KAT [12] and ABEL [18]).

Minimization of the number of inclusions. An agent that commits to a particular theory necessarily also commits to the theories included by that theory. Therefore, organizing the theories in such a way that a theory includes few other theories, reduces the overhead of committing to that theory and allows a more flexible use of the library. Therefore, the second criterion used to partition the definitions into theories is that the number of inclusion links must be kept to a minimum. A theory must include, directly or indirectly, the minimal set of theories that it presumes. For example, ***disease***, which is defined in diseases, is a subclass of ***clinical-process***, which is defined in fundamental-medical-concepts. Therefore, it is necessary that diseases includes fundamental-medical-concepts.

As depicted in Fig. 3, two indirect inclusion paths connect clinical-environment, defining concepts related to the context in which medical activities take place, to diseases. The classes ***therapy*** and ***test*** are defined in separate theories, enabling external agents to commit to one of the theories without committing to the other. However, because both theories include diseases, all agents committing to one of the two theories must commit to the same definition of diseases. For this reason it is important to avoid ontological overcommitment. In the core part of the library only general characteristics of the concepts should be defined, more specific characteristics should be defined as domain- or method-specific extensions in the peripheral areas of the library.

4.2.2. Contents of the core library

Table 1 contains short descriptions of some of the theories in the library core ontology as shown in Fig. 3. [2] Fig. 4 shows an example Ontolingua definition, namely that of the class ***observable*** which is defined in the theory findings. The sentence labeled as ***:axiom-def*** expresses that ***observable*** is a subclass of ***human-body-state-variable***, which is defined in the theory fundamental-medical-concepts.

[2] The full set of library definitions is given in [7].

Table 1
Characterization of some theories in the core library as shown in Fig. 3

Theory	Characterization of contents
generic-concepts	Defines basic notions such as ***system, process, action*** from an "engineering" point of view. For example, a system is conceptualized as a collection of interconnected components characterized by states and processes.
fundamental-medical-concepts	Contains definitions of basic notions useful for medical knowledge representation, such as ***human-body*** and ***medical-agent***. The definitions in this theory specialize notions defined in generic-concepts. For example, ***human-body*** is a subclass of the class ***system***, i.e. it is conceptualized as a class of complex entities describable through states and concerned with physiological or pathological (e.g. clinical) processes.
anatomy	Define ontological categories such as ***physiological-process***, ***anatomical-part*** and ***organ*** that are generally used in medical contexts. The definitions are mostly based on [18].
physiology	
findings	Define and classify respectively observable findings, conceptualized as values assumed by time variables indicating the clinical state of a patient, drugs and surgical interventions. They can be thought of as the "information ontologies" underlying the patient medical record structure.
drugs	
surgeries	
clinical-state-abstractions	Defines concepts for representing clinical states in compact ways, for instance, to synthesize a set of patient findings. This theory defines, for example, (i) ***qualitative-clinical-state-abstraction*** expressed using symbolic values such as "low" or "high", and (ii) ***quantitative-clinical-state-abstraction*** expressed using numerical values (e.g. a measure such as the body surface computed from body weight and height).
diseases	Defines a ***disease*** as a clinical process whose evolution can be described through finding or clinical abstraction values over time, and tries to define taxonomies, used commonly in medical practice, based on diseases characteristics such as time evolution characteristics (e.g. "acute", "chronic"), etiology and location.

The ***:axiom-constraints*** sentence defines four possible subclasses of ***observable***. The difference between ***:axiom-def*** and ***:axiom-constraints*** sentences is that the former are considered to be definitional (see [8] for an explanation). The terms, ***subclass-of*** and ***subclass-partition*** are defined in the Frame ontology.

The principle behind the core definitions is that these should be minimal. For example, stating that an observable is associated with a quantitative value set (the possible values of the observable are numbers) would be an ontological overcommitment, as this is not likely to hold for every application. Therefore, such a qualification should be defined as a (probably method-specific) extension.

4.3. Method- and domain-specific extensions

The categories described in the previous section are considered basic, in the sense that they are more or less standard across medical tasks and medical domains, and form a generally agreed upon body of terminology in the medical

```
(define-class OBSERVABLE (?observable)
  "An observable is a state-variable whose values can - contextually -
indicate pathological or physiological states which can be observed.
They can be classified according to the way they are obtained."

  :AXIOM-DEF (subclass-of observable human-body-state-variable)
  :AXIOM-CONSTRAINTS (subclass-partition observable
                                          (set-of sign
                                                  symptom
                                                  laboratory-observable
                                                  special-investigation)))
```

Fig. 4. The Ontolingua definition of the notion of "observable". Ontolingua definitions consist of the name of the defined concept, a number of instance variables, and sets of labeled sentences. The sentence labeled as ***:axiom-def*** defines that ***observable*** is a subclass of ***human-body-state-variable***, which is defined in fundamental-medical-concepts. The ***:axiom-constraints*** sentence defines four possible subclasses of ***observable***.

field. We have already mentioned that this set of theories, while relatively small, still allows for alternative definitions. In the core part of the library, the definitions are very general, in the sense that they allow for further specialization according to application-specific requirements. We will now describe the more application-dependent parts of the library. Applications may vary on two attributes: (i) the domains that they reason about, and (ii) the tasks that they perform and the methods that they use.

Reuse of domain-specific concepts across domains. At first glance, the reuse of domain-specific concepts across domains seems a contradiction in terms. However, domain-specificity is not a dichotomy: some concepts are obviously more domain specific than are others. For example, the concept of "fungal skin infection" is more specific than that of "dermatological disease", while both are specific to the domain of dermatology. This observation is made concrete by assigning an attribute of *domain-specificity* to each concept. This attribute indicates to what extent a concept is specific to a domain or to a set of domains. To decide on the domain-specificity of concepts, a taxonomy of medical specialties is used. Each of the nodes in this taxonomy represents a medical subdomain that may be used as a value for the domain-specificity attribute of a concept. When a concept has a particular domain as its domain-specificity value, it is specific for that domain, but it is reusable across all its subdomains. The domain-specificity of a concept is *not* the same as its level of abstraction; abstract concepts can be very domain specific and concrete concepts can be very generic.

The domain taxonomy will be used for the retrieval of definitions that are likely to be used in a particular application. Since future medical KBS applications are likely to be embedded in existing hospital information systems, and therefore must be integrated in the existing organizational structure of the hospitals, the taxonomy should reflect this structure. In other words, the domain taxonomy is derived from medical practice. Example elements of the taxonomy are disciplines such as immunology, pathology, internal medicine and its specializations, etc.

Of course, the organization of medical practice varies between countries. Therefore, the structure of peripheral parts of the library is to a certain extent *situated*. This is another motivation for distinguishing between a "universal" core library and situated extensions of that core.

Reuse of method-specific concepts across methods and tasks. According to the interaction problem, the way that knowledge is represented is necessarily highly interwoven with the way that that knowledge will be used in reasoning. Therefore, it is difficult to reuse knowledge that is defined with a particular method in mind, for another method. Taken literally, the interaction problem precludes the reuse of concepts across methods. In this section it will be argued that the interaction problem does not hold to the same extent for every concept, and it will be shown that the degree of *method-specificity* of concepts can be used as an index to organize the ontology library.

It has been argued elsewhere [23] that there are three fundamental tasks in medical reasoning: diagnosis, therapy planning and patient monitoring. Furthermore, it has been shown that at least two of these generic tasks can easily be modeled as instantiations of one inference model: the *select and test* model, or STModel. Fig. 5 shows the instantiation of this model for medical diagnosis. The model, which is based on the work of the philosopher Peirce [19], distinguishes four fundamental reasoning steps: data abstraction, abduction of hypotheses, deduction of predictions, and inductive verification of the predictions. Furthermore, the model distinguishes two additional activities: deciding in which order the hypotheses will be tested (ranking) and requesting new data.

Each of the inference steps in the STModel can be realized through a number of methods, and each of these methods may have specific ontological requirements. For example, the abduction of hypotheses from patient findings can be done by interpreting direct associations between findings and diseases. This method thus has the ontological requirement that such associations exist. The following production rule is an illustration of this kind of abduction:

IF ***chest − pain = present*** AND

sustained − pain = yes

THEN ***Myocardial − infarct = probable***

In some systems that perform abduction by direct associations, the associations are qualified with *certainty factors*, representing the likelihood that the disease is

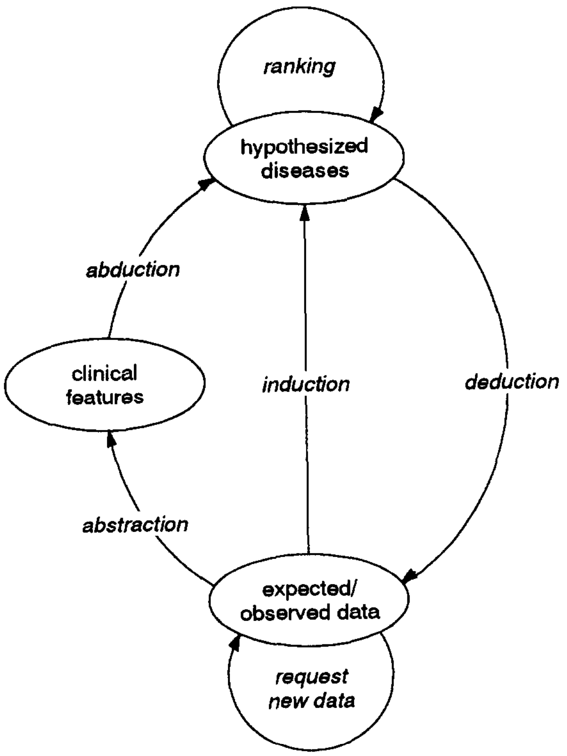

Fig. 5. The generic STModel, instantiated for medical diagnosis.

the cause of the findings. This is, for example, the case in MYCIN [27]. Using this method thus introduces another ontological requirement.

Alternatively, the diseases that may cause a particular finding could be found by tracing pathways in causal networks — a method which requires the existence of *causal connections* in the domain. For specific methods, the causal links in such networks may need further qualification. For instance, CHECK [5], a system for abductive diagnosis, makes a distinction between necessary causal connections and possible causal connections. Another example of this is provided by causal probabilistic networks, where the causal relations are quantified through probability distributions.

Like the domains in the medical field, the methods that are employed in medical reasoning can be organized in a taxonomy. Descending this taxonomy introduces additional ontological commitments. Fig. 6 shows a part of the method taxonomy for abducting diseases from findings in medical diagnosis. The concepts of disease and finding, which are used by all methods for medical abduction, are defined in the core library. The ***manifestation-of*** relation, which models direct associations between findings and diseases, is specific for methods that are specializations of method "abduction by direct associations between findings and diseases" (method 2.1 in Fig. 6). Further specializations of these methods may

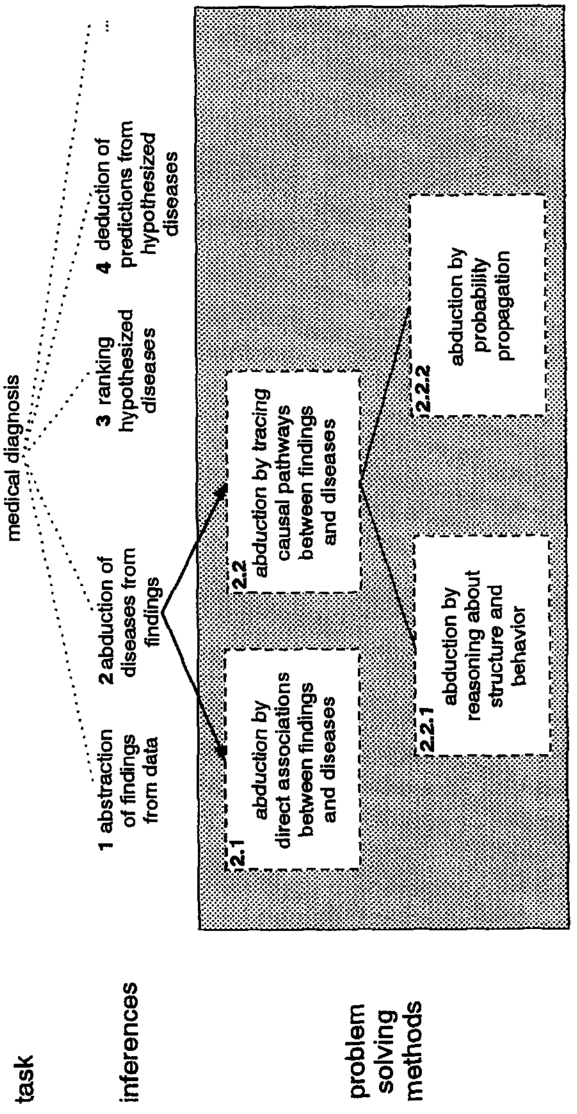
task
inferences
problem solving methods
medical diagnosis
1 abstraction of findings from data
2 abduction of diseases from findings
3 ranking hypothesized diseases
4 deduction of predictions from hypothesized diseases
...
2.1 abduction by direct associations between findings and diseases
2.2 abduction by tracing causal pathways between findings and diseases
2.2.1 abduction by reasoning about structure and behavior
2.2.2 abduction by probability propagation

Fig. 6. Partial taxonomy of methods used in medical diagnosis.

require additional ontological commitments, such as the existence of certainty factors or evoking strengths for these direct associations.

The level of the method taxonomy where an ontological requirement is introduced, is an indicator for the method-specificity of the corresponding concept. In the same way that the domain taxonomy is used to associate each concept with a domain-specificity attribute, every concept is also assigned a method-specificity value.

4.4. *Structure of the library*

Section 4.2 identified some basic categories of medical knowledge that are assumed to be reusable across all medical domains and medical tasks. These categories form the core of the current library, and they are organized in theories according to the criteria mentioned earlier.

Two attributes determine the degree of reusability of a concept: the domain-specificity and the method-specificity. For the definitions in the core part of the library, these attributes are not discriminating, as they are intended to be reusable across most medical domains and most medical tasks. However, this is not the case for the definitions in the extended part. By making the value of concepts on these attributes explicit, it is possible to determine to *what extent* and *under which circumstances* these concepts can be reused. Furthermore, since concepts that have the same values on both attributes are likely to be applicable under the same circumstances, they should be stored in one theory. In this way the attributes provide a scheme for modularization.

For every combination of a node from the domain taxonomy and a node from the method taxonomy, there can be a theory in the library in principle. This theory contains all the definitions that are specific for the method and the domain, but that are reusable across all the specializations of the method and all subdomains of the domain. For instance, the theory "abduction by tracing causal pathways between findings and diseases in the domain of cardiac diseases" would contain all the definitions that are specific for that method in that domain (e.g. oxygen-supply and ischemia), but it would not specify that there are probability distributions that describe the nature of the causal connection between these states, since these are specific for one particular specialization of the causal-tracing method (see Fig. 6). The theory would also not contain a definition of pathophysiological states, since this notion is reusable across a wider range of domains than cardiology. Therefore this concept is defined in the core library.

5. Building the library

The previous section described the principles of organizing the library of medical ontologies. Of course, the library must still be filled. Because this involves a large amount of work, only a prototype library is being developed in the GAMES-II project. Rather than aiming at completeness, the project concentrates on formulating standardized procedures for adding new definitions to the library. The availability of standardized procedures will make it easier for others to augment the library and it will enable the development of tools for semi-automatic library maintenance. The currently used procedure consists of four steps: (i) take an existing medical AI application, (ii) describe the ontology and the problem-solving methods of the system, (iii) score the definitions in the ontology on the domain-specificity and method-specificity attributes, and (iv) put the definitions in the appropriate theories.

5.1. Start with an existing application

The definitions that are most likely to be usable for medical KBS development are the definitions that are already employed in existing systems. Therefore, the initial library is based on analyses of such systems. In this article CASNET [34] will be used as an example. CASNET allows the representation of causal associational networks that describe processes of diseases and has been used to make a causal model for glaucoma.

CASNET was chosen as an example for several reasons. Firstly, as a representation language it provides general building blocks for various potential medical applications. In other words, its underlying domain ontology is a good candidate for reuse across domains. Secondly, in addition to this general causal network ontology CASNET provides somewhat idiosyncratic ontological distinctions required by CASNET's reasoning methods. This combination of properties makes CASNET an attractive illustration for building the core library and its method-specific extensions. As CASNET uses a general representation language, it is unlikely that its analysis will reveal domain-specific extensions.

5.2. Describing the ontology

It is often the case that existing medical KBS do not have an explicit description of the underlying domain ontology. In these cases, it is up to the library builders to define such an ontology. In order to prevent multiple definitions of concepts in the library, the application described is first scored on the domain-specificity and the method-specificity attributes. As described in Section 4, the values of these attributes are derived from the corresponding taxonomies. When the application is scored on these attributes it is possible to retrieve the definitions that are already available in the library. If the indexing is correct, the retrieved definitions are likely to be needed for the application under consideration. In most cases, the retrieved definitions are not sufficient to model the application's domain knowledge. This means that the library builder will have to define the additional classes, relations and functions required for the application.

CASNET's application ontology . Applications built with CASNET have an explicit representation of a network, the nodes of which represent pathophysiological states. States describe events that are deviations from the normal course of events. The links in the network represent causal relations between the states. States are labeled with a confirmation status, which must be one of ***confirmed, denied,*** or ***uncertain***. The evidence for the confirmation status of a state comes from patient observations. A specific state of the network is interpreted in terms of diseases in various states of progression, using classification tables.

Fig. 7 presents parts of the reconstructed application ontology of CASNET in the form of an *ontological semantic network* [2]. The classes shown with a black background belong to the parts of the application ontology that are retrieved from — or should be stored in — the core library. For example, ***patho-physiological-state, observation*** and ***disease*** belong to this category. The classes with a gray background in Fig. 7 belong to the part of application ontology that is specific for the particular problem-solving method that CASNET uses, and which will be explained below. Relations are shown in a similar way. For example, while the relation ***evidence-for*** between an observation and a pathophysiological state belongs to the core library; the ***confidence*** relation is a method-specific extension.

CASNET's problem-solving methods . The analysis of CASNET's problem-solving methods is based on the STModel (see Fig. 5). Fig. 8 shows the methods that perform the abduction inference (Fig. 8a) and the ranking inference (Fig. 8b). The figure also shows the method-oriented parts of the application ontology, and the way they are related to the methods. The methods also refer to the method-independent parts of the application ontology of course, but these references are omitted in the figure.

The abduction task (Fig. 8a) is implemented by the method "Abduction by tracing causal pathways between findings and diseases", which is comprised of three primitive procedures. The first of these uses the evidence links between observations and states and their associated confidence measures to compute the confidence measure of the state. The second procedure then labels the states with ***confirmed, denied*** or ***undetermined*** by applying a threshold on the confidence measure of the states. Finally, the third procedure classifies paths of labeled states (with no denied states) as diseases.

The problem-solving method that CASNET uses for the of the ranking inference (Fig. 8b) consists of two procedures: weighing the evidence for the hypothesized diseases and ranking them according to the weight of the evidences. The weighing procedure consists of three steps, which all use the strengths of the causal relations between the states. The total weight of a state is the maximum between the forward and inverse weights. The forward weight of a state summarizes the weight of the evidences coming from all the causes of that state. The inverse weight summarizes the weight of the evidences coming from the effects of the state. When the status of a state is ***undetermined***, the starting state's a-priori frequency is used for the calculation of its forward weight. The procedure that

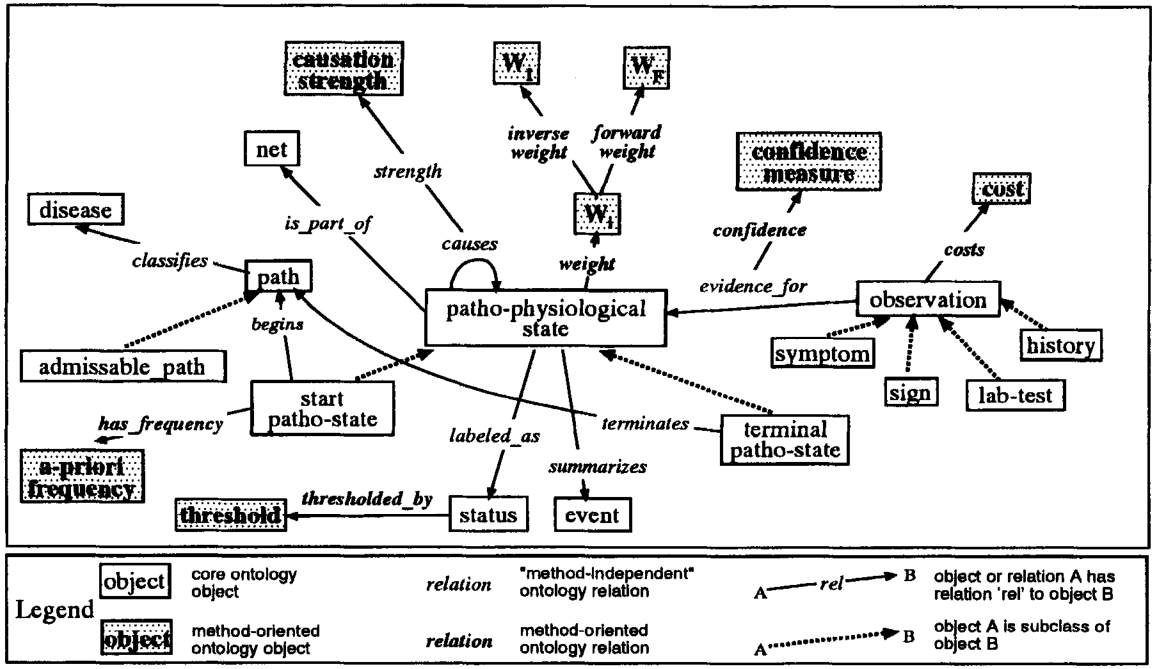

Fig. 7. The application ontology of CASNET represented as an ontological semantic network.

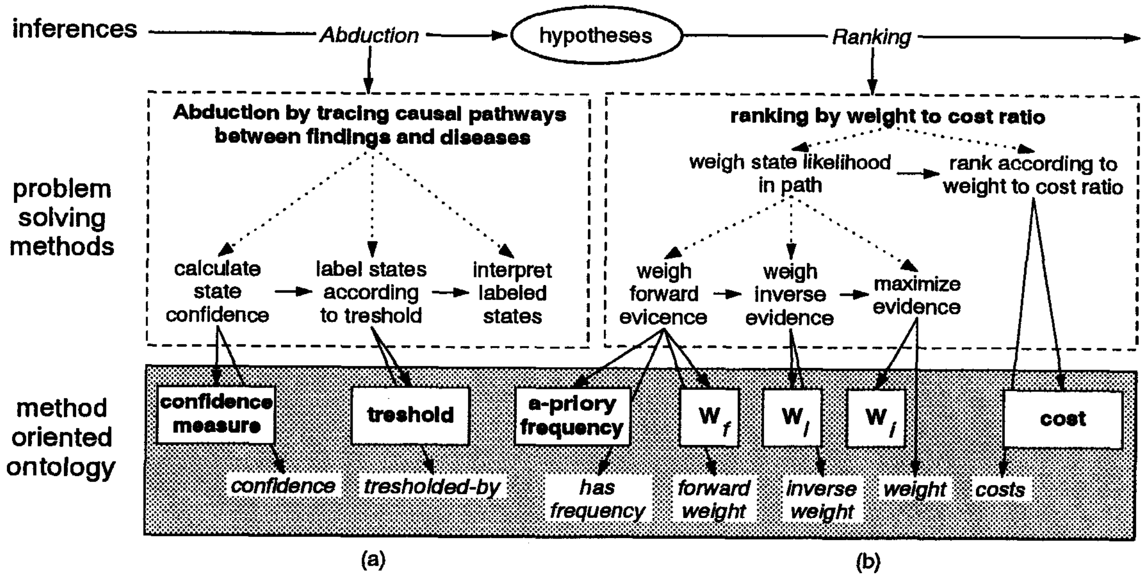

Fig. 8. The problem-solving methods used in CASNET for abduction of hypotheses (a) and ranking of hypotheses (b).

ranks the states (hypotheses) uses the ratio of the weight of the hypothesis and the costs of testing that hypothesis.

5.3. Scoring and storing the definitions

When the ontology of the system has been specified, the newly defined concepts must be indexed and stored in the library. In the case where the core library is largely complete, this is not difficult. All the newly defined concepts are then method or domain specific, and must be stored in the peripheral part of the library. For example, the notion of ***confidence***, which is specific for CASNET's problem-solving method, should be stored in a theory confidence-based-abduction. In the case where the core library is also incomplete, the indexing is more difficult. Now the library builder has to decide whether the definition represents a basic category of medical knowledge, or that it is a method- or domain-specific extension. The procedure to follow in this situation is based on the principle that the concepts in the core library are supposed to be reusable across many tasks and domains. If the library builder estimates that this is true for a concept under consideration, it is stored in the core library, otherwise it is considered as an extension. Of course, the subjective estimates of the library builder are not error-proof, but at the moment this is the only method available. One of the assumptions that underly this approach to library construction is that there are only few truly basic categories of medical knowledge, so that it is likely that the current core library is already more or less complete. Once the library is being used for KBS development, statistics about the actual reuse of concepts can be used to improve the indexing of the library concepts.

6. Using the library

This section presents a scenario that illustrates how the library can be used to construct a part of the application ontology of a KBS. The new application supports ambulance dispatchers in their decision whether to send an ambulance after an emergency call. The (hypothetical) scenario is based on a case study to reconstruct parts of the FREECALL system [20] using the ontology library.

In Section 4 it was mentioned that there are three factors that determine which definitions are needed in an application ontology: (i) the domain, (ii) the task and the methods, and (iii) the external servers that are to be connected to the KBS. Therefore, these are determined first.

The system under development will be used as an on-line assistant for ambulance dispatchers, suggesting diagnoses or further questions to ask to the caller. Because people call emergency units of hospitals for many different reasons, the system should have knowledge about a multitude of medical domains. However, for the sake of simplicity, only cardiac diseases will be taken into account in this example.

Because FREECALL tries to determine the origin of the complaints of the caller, the task of the system is classified as diagnosis. In Section 4.3, it was argued that diagnosis can be considered as an iterative process of abstracting findings, abducting hypotheses, and deducing expectations, where each of the inferences may be realized by different methods. This example will concentrate on the abductive inference of the STModel.

To determine which method-specific extensions must be included in the application ontology, it must be decided which method will be used for the abductive inference. Usually, method selection depends heavily on the characteristics of the available domain knowledge. However, to keep the example simple it is assumed that, in this case, it is possible to select an appropriate method without inspection of the available domain knowledge. The method that is selected is "tracing causal pathways between findings and diseases" which is partially illustrated in Fig. 8a.

The external devices that are to communicate with the KBS are the third factor that influences the contents of the application ontology. In our example, the ambulance dispatchers have a patient database at their disposal, which can be used to retrieve the medical history of a caller. In the case of complaints about chest pain, an ambulance should always be sent if the caller has had an earlier myocardial infarct. In the current non-automated practice, this is determined by asking the caller, since entering a database query takes more time. However, when the name and the address of the caller are already recorded, such a query can be performed by the KBS without bothering the caller or the dispatcher. The dispatcher then only needs to ask the caller about a previous infarct, if no information was found in the database. To facilitate the integration of the KBS

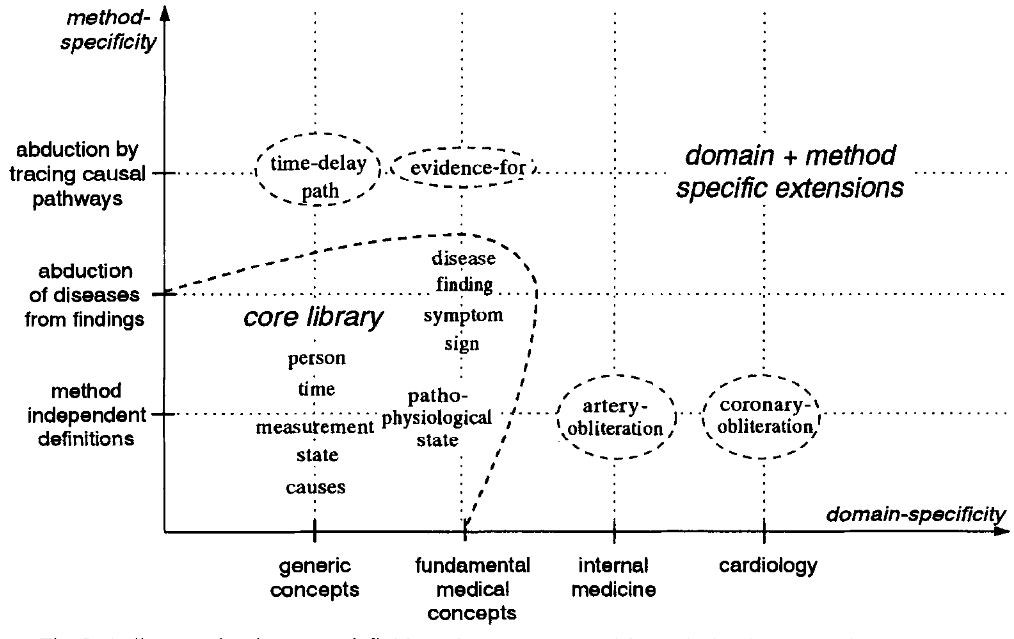

Fig. 9. A diagram showing some definitions that are suggested for inclusion in the application ontology of the ambulance dispatchers assistant. The positions in the diagram reflect the locations of the definitions in the library.

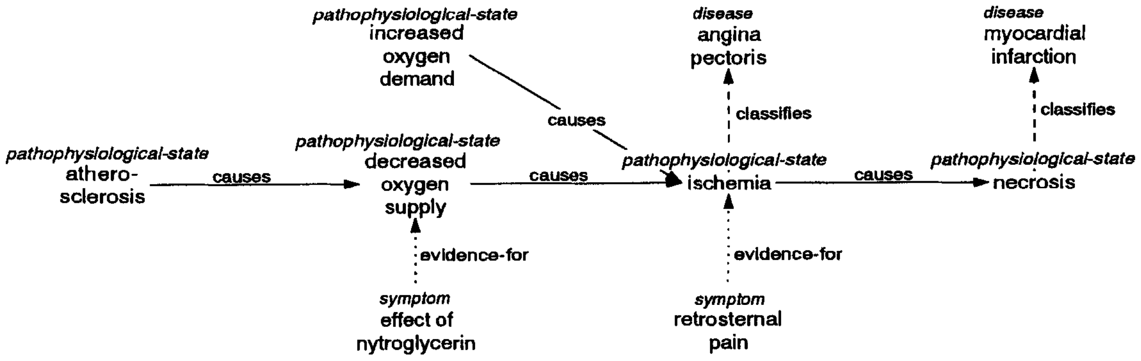

Fig. 10. A causal network in the domain of cardiology.

with the patient database, the application ontology should include concepts that can easily be mapped onto the record structure of the database.

To summarize, the system being developed does abduction by tracing causal pathways in the domain of cardiology, and it will be connected to a patient database that contains records with the names, addresses and clinical histories of previous patients. These qualifications can be used as cues to retrieve a number of definitions from the ontology library that are likely to be useful for the application ontology under construction. It must be emphasized that the library does not produce complete application ontologies, it only suggests likely candidates for inclusion. Which of the suggested concepts will be included in the final ontology must be decided by the knowledge engineer.

For a particular application ontology and a particular reasoning step in the STModel, the reusability characteristics of the definitions can be illustrated in a *reusability diagram*. Fig. 9 shows such a diagram for the abductive reasoning step in the ambulance dispatchers assistant. The domain-specificity axis of the diagram is constructed by starting from the specific domain in the domain taxonomy, and then moving upwards through the domain taxonomy. Each of the parent nodes in the taxonomy is used as a value on the domain-specificity dimension. The method-specificity axis is constructed in a similar way, using the method taxonomy.

The region at the lower left part of the diagram contains the definitions that are retrieved from the core part of the library, which was described in Section 4.2. The definitions that are both method independent and generic are retrieved from the theory generic-concepts. For the other definitions in this region, the positions in the reusability diagram do not reflect from which theories they originate. In general, most of the definitions in the core library are suggested for inclusion, and the knowledge engineer must decide which of these are applicable. For example, the concept ***patient-history*** is included to make sure that the system can be connected to the patient database. This connection is realized by an explicit mapping between the description of the internal structure of a ***patient-history*** which is part of the definition, and the conceptual schema of the database.

The other region in the reusability diagram contains the definitions that are specific for the particular subdomain and method. For example, the relation ***evidence-for*** is added to the application ontology because the abductive method uses this relation to infer pathophysiological states from patient findings. Definitions that are grouped by a dashed bubble in the diagram are retrieved from the same theory.

Once the application ontology is constructed, it can be used to guide the knowledge-acquisition process, ensuring that the elicited knowledge does not violate the ontological constraints. Fig. 10 shows a part of the knowledge base of the ambulance dispatchers assistant.

7. Discussion

The starting point of the work presented here is the observation that, although the potential merits of reusable libraries are widely recognized, no such libraries are available today. They could provide building blocks for an application ontology, which is a specification of all the ontological distinctions that are required to perform a particular task in a particular domain. Two reasons were identified to explain the unavailability of such a library: the hugeness problem and the interaction problem.

This article presents an analysis of these problems in the context of medical knowledge, and suggests ways to make them manageable. In short, the interaction problem is addressed by the introduction of a method-specificity attribute for concepts, based on a classification of problem-solving methods. To the hugeness problem there are two aspects: the large number of concepts makes *building* the library a daunting exercise, and it also complicates *retrieving* the appropriate concepts when they are needed. Because of the former, we do not attempt to build a full library, but instead concentrate on the formulation of procedures for augmenting an initial library. The latter aspect is addressed by the introduction of a domain-specificity attribute, similar to the method-specificity attribute. Based on this analysis and an analysis of the intended use of the library, three principles have been identified that can be used to impose a structure on an ontology library: organizing concepts according to (i) natural categories, (ii) problem-solving methods, and (iii) domain division in practice. The first of these principles advocates structuring ontologies of medical knowledge according to "topics" that recur often in medical practice. These general categories are located in the core part of the library. The importance of this organizational principle is that it provides anchors for the more specialized concepts in the other part of the library, thereby ensuring that concepts that are defined in different ways for different methods or subdomains, have at least some common ground. The second principle says that problem-solving methods should be used as an index for the ontological distinctions that they introduce. This facilitates the construction of application ontologies, because it is easy to find out which domain concepts are required for a particular problem-solving method. The third principle suggests that domain concepts that are specific to a particular branch in medical practice, should be indexed by that subdomain. This facilitates the construction of application ontologies because it can be used to suggest concepts that are specific for problem-solving in that domain, and it also suggests what kinds of external knowledge will be available in the runtime environment of the KBS.

A crucial underlying assumption of this work is that it is indeed possible to score medical applications within the framework presented. The justification of this assumption depends on how well the medical field fits in the mold provided by the above-mentioned principles. For the applications that we have studied so far, this assumption seems to be reasonable. The assumption is also likely to hold for applications built using the now prevailing "knowledge modeling" KBS development paradigm. These approaches provide meta-level descriptions of a knowledge base that can be used for indexing. In particular, applications built through explicit mappings between domain, method-specific and application ontologies, such as advocated by the CommonKADS methodology [35] will lend themselves to library construction and use.

The PROTÉGÉ-II approach [29], makes similar distinctions, and it is to be expected that libraries of the kind described in this article will be usable within that framework as well. In the PROTÉGÉ-II framework, application ontologies are developed by augmenting a domain ontology with method-specific distinctions. Then, explicit mappings are defined between the application ontology and a method ontology. The method ontologies, which are specific for the methods in PROTÉGÉ-II's method library, are reusable, and so is the domain ontology. The use of the method-specificity index in the GAMES library could be viewed as a way of explicating when which method-specific distinctions should be added to the domain ontology.

The presented work is closely related to efforts to standardize medical terminology. In Section 4.1 it was already mentioned that the GAMES library could be related to the semantic network of the UMLS. The GAMES library would benefit from such a mapping because the automatic classification of terms would make it easier to locate concepts in the library. On the other hand, the GAMES library would impose a richer structure on the semantic network, which would make it easier to connect the knowledge in the UMLS to problem-solving methods. To investigate the feasibility of such an approach, we are currently carrying out an experiment to connect the ontology library with a terminology server which is developed in the European GALEN project.

Perhaps the most important practical contribution of the work presented here is that it provides a starting point for experimentation. So far, the usefulness of ontology libraries for knowledge engineering has been a subject of speculation. By building such a library, and also by using it, we hope to gather practical experience. It might well be that the current principles will turn out to be insufficient. Furthermore, it is very likely that there are many other problems with the construction of libraries of reusable ontologies besides the hugeness problem and the interaction problem. However, to get more insight into the nature of these problems, it is necessary to gain experience with using such a library. The purpose of this library is to provide an environment for such experimentation.

Acknowledgements

We have profited from discussions with Manfred Aben, Joost Breuker, Frank van Harmelen, Giordano Lanzola and Bob Wielinga. We are grateful to Mark Musen, Lynda Hardman and two anonymous reviewers for their comments on an earlier version of this paper.

The research reported here was carried out in the course of the GAMES-II project. This project is partially funded by the AIM Programme of the Commission of the European Communities as project number A2034. The partners in this project are SAGO (Florence, Italy), Foundation of Research and Technology (Crete, Greece), Geneva University Hospital (Switzerland), the University of Amsterdam (The Netherlands), University College of London (UK), the University of Pavia (Italy) and the University of Ulm (Germany).

This paper reflects the opinions of the authors and not necessarily those of the consortium.

References

[1] M. Aben, Formally specifying re-usable knowledge model components, *Knowledge Acquisition* 5 (1993) 119–141.

[2] A. Abu-Hanna, Multiple domain models in diagnostic reasoning. PhD thesis, University of Amsterdam, Amsterdam, 1994.

[3] J.A. Breuker and W. Van de Velde, eds., *The CommonKADS Library for Expertise Modelling* (IOS Press, Amsterdam, The Netherlands, 1994).

[4] T. Bylander and B. Chandrasekaran, Generic tasks in knowledge-based reasoning: the right level of abstraction for knowledge acquisition, in: B.R. Gaines and J.H. Boose, eds., *Knowledge Acquisition for Knowledge-based Systems*, volume 1 (Academic Press, London, 1988) 65–77.

[5] L. Console, L. Portinale, D.T. Dupré and P. Torasso, Combining heuristic reasoning with causal reasoning in diagnostic problem solving, in: J.M. David, J.P. Krivine and R. Simmons, eds., *Second Generation Expert Systems* (Springer-Verlag, Berlin, Germany, 1993) 46–68.

[6] L. Eshelman, MOLE: a knowledge-acquisition tool for cover-and-differentiate systems, in: S. Marcus, ed., *Automating Knowledge Acquisition for Expert Systems* (Kluwer, Boston, 1988) 37–80.

[7] S. Falasconi, Ontological foundations of knowledge-based systems in medicine. Master's thesis, University of Pavia, Italy, 1993 (in Italian).

[8] T.R. Gruber, Ontolingua: a mechanism to support portable ontologies, version 3.0. Technical Report, Knowledge Systems Laboratory, Stanford University, California, 1992.

[9] T.R. Gruber, A translation approach to portable ontology specifications. *Knowledge Acquisition* 5 (1993) 199–220.

[10] N. Guarino and L. Boldrin, Ontological requirements for knowledge sharing. Paper presented at the IJCAI Workshop for Knowledge Sharing and Information Interchange, Chambery, France, 1993.

[11] G. Klinker, C. Bhola, G. Dallemagne, D. Marques and J. McDermott, Usable and reusable programming constructs. *Knowledge Acquisition* 3 (1991) 117–136.

[12] G. Lanzola and M. Stefanelli, A specialized framework for medical knowledge-based systems. *Computers and Biomed. Res.* 25 (1992) 351–365.

[13] D.A.B. Lindberg, B.L. Humphreys and A.T. McCray, The unified medical language system. *Methods of Information in Med.* 32 (1993) 281–291.

[14] S. Marcus, ed., *Automatic Knowledge Acquisition for Expert Systems* (Kluwer, Boston, 1988).

[15] R.A. Miller, M.A. McNeill, S.M. Challinor, F.E. Masarie Jr. and J.D. Myers, The INTERNIST-1/quick medical reference project — status report. *West. J. Med.* 145 (1986) 816.

[16] M.A. Musen, *Automated Generation of Model-Based Knowledge-Acquisition Tools.* Research Notes in Artificial Intelligence (Pitman, London, 1989).

[17] M.A. Musen, Overcoming the limitations of role-limiting methods, editorial special issue. *Knowledge Acquisition* 4(1) (1992) 163–168.

[18] R.S. Patil, Causal Representation of Patient Illness for Electrolyte and Acid–Base Diagnosis. PhD thesis, Laboratory for Computer Science, MIT, 1981.

[19] C.S. Peirce, *Collected Papers of Charles Saunders Peirce, volume 2, Elements of Logic* (Harvard University Press, Cambridge MA, 1932).

[20] W.M. Post, R.W. Koster, V. Zocca and M. Sramek, Cooperative medical problem solving, in: *Proceedings of the 4th Conference on Artificial Intelligence in Medicine Europe AIME-93*, Munich, 1993.

[21] T.A. Pryor, R.M. Gardner, P.D. Clayton and H.R. Warner, The HELP system, *J. Med. Syst.* 7 (1983) 87.

[22] A.R. Puerta, J. Egar, S.W. Tu and M.A. Musen, A multiple-method shell for the automatic generation of knowledge acquisition tools. *Knowledge Acquisition* 4 (1992) 171–196.

[23] M. Ramoni, M. Stefanelli, G. Barosi and L. Magnani, An epistemological framework for medical knowledge-based systems, *IEEE Trans. Systems, Man and Cybernetics* 22 (1992) 1361–1375.

[24] A.L. Rector, W.A. Nowlan, S. Kay, C.A. Goble and T.J. Howkins, A framework for modelling the electronic medical record, *Methods of Information in Med.* 32 (1993) 109–119.

[25] E. Rosch, Natural categories, *Cognitive Psychol.* 4 (1973).

[26] A.Th. Schreiber, B.J. Wielinga, J.M. Akkermans, W. Van de Velde and A. Anjewierden, CML: the CommonKADS conceptual modelling language, in: L. Steels, A.Th. Schreiber and W. Van de Velde, eds., *A Future for Knowledge Acquisition. Proceedings of the 8th European Knowledge Acquisition Workshop EKAW'94* (Springer-Verlag, Berlin, 1994) 1–25.

[27] E.H. Shortliffe, *Computer-Based Medical Consultations: MYCIN* (American-Elsevier, New York, 1979).

[28] J.W. Smith, A. Bayazitoglu, K.A. Johnson, N.K. Arma and T.R. Johnson, One framework, two systems: flexible abductive methods in the problem-space paradigm applied to antibody identification and biopsy interpretation, *Artificial Intelligence in Med.* 7 (1995) 201–225.

[29] S.W. Tu, H. Eriksson, J.H. Gennari, Y. Shahar and M.A. Musen, Ontology-based configuration of problem-solving methods and generation of knowledge acquisition tools: the application of PROTÉGÉ-II to protocol-based decision support, *Artificial Intelligence in Med.* 7 (1995) 257–290.

[30] S.W. Tu, M.G. Kahn, M.A. Musen, J.C. Ferguson, E.H. Shortliffe and L.M. Fagan, Episodic skeletal-plan refinement based on temporal data. *Communications of the ACM* 32(12) (1989) 1439–1455.

[31] A. Valente and C. Löckenhoff, Organization as guidance: a library of assessment models, in: *Proceedings of the Seventh European Knowledge Acquisition Workshop (EKAW'93)* (1993) 243–262.

[32] J. van Bemmel, Criteria for the acceptance of decision support systems by clinicians, in: S. Andreassen, R. Engelbrecht and J. Wyatt, eds., *Proceedings of the 4th Conference on Artificial Intelligence in Medicine Europe, 3–6 October 1993, Munich*, volume 10 of *Studies in Health Technology and Informatics* (IOS Press, Amsterdam, The Netherlands, 1993) 7–10.

[33] G. van Heijst, G. Lanzola, A.Th. Schreiber and M. Stefanelli, Foundations for a methodology for medical KBS development, *Knowledge Acquisition* 6 (1994) 1–39.

[34] S.M. Weiss, C.A. Kulikowski, S. Amarel and A. Safir, A model-based method for computer-aided medical decision making, in: W.J. Clancey and E.H. Shortliffe, eds., *Readings in Medical Artificial Intelligence, the First Decade* (Addison Wesley, 1984).

[35] B.J. Wielinga and A.Th. Schreiber, Reusable and shareable knowledge bases: a European perspective, in: *Proceedings International Conference on Building and Sharing of Very Large-Scaled Knowledge Bases '93* (Japan Information Processing Development Center, Tokyo, Japan, 1993) 103–115.

[36] B.J. Wielinga, A.Th. Schreiber and J.A. Breuker, KADS: a modelling approach to knowledge engineering, *Knowledge Acquisition* 4(1) (1992) 5–53. Special issue "The KADS Approach to Knowledge Engineering". Reprinted in: B. Buchanan and D. Wilkins, eds., *Readings in Knowledge Acquisition and Learning* (Morgan Kaufmann, San Mateo, California, 1992) 92–116.