Chapter 5
Building Intelligent Tutoring Systems for Ill-Defined Domains

Philippe Fournier-Viger[1], Roger Nkambou[1], and Engelbert Mephu Nguifo[2]

[1] Department of Computer Sciences, University of Quebec at Montreal,
201President-Kennedy Avenue, Montreal, Canada, H2X 3Y7
fournier_viger.philippe@courrier.uqam.ca,
nkambou.roger@uqam.ca
[2] Department of Mathematics and Computer Sciences,
Université Blaise-Pascal Clermont 2, BP 125, 63173 Aubière, cedex, France
mephu@isima.fr

Abstract. Domains in which traditional approaches for building tutoring systems are not applicable or do not work well have been termed "ill-defined domains." This chapter provides an updated overview of the problems and solutions for building intelligent tutoring systems for these domains. It adopts a presentation based on the following three complementary and important perspectives: the characteristics of ill-defined domains, the approaches to represent and reason with domain knowledge in these domains, and suitable teaching models. Numerous examples are given throughout the chapter to illustrate the discussion.

5.1 Introduction

In recent years more and more research has focused on building Intelligent Tutoring Systems (ITS) for domains that pose new challenges, i.e., where traditional approaches for building tutoring systems are not applicable or do not work well. Such domains have been termed "ill-defined domains" by the Artificial Intelligence in Education (AIED) community (Aleven 2003; Aleven et al. 2006; Aleven et al. 2007). Research on ill-defined domains has given rise to several solutions. Some are domain-specific, while others tend to be more generic. Despite the numerous papers published on this topic and the three workshops presented at recent conferences (Aleven 2003; Aleven et al. 2006; Aleven et al. 2007). the definition of an "ill-defined domain" is still under debate and none of the proposed solutions are appropriate for all domains (Aleven 2003; Aleven et al. 2006; Aleven et al. 2007). This chapter aims to clarify the definition of an "ill-defined domain," to offer various solutions for building tutoring systems for these domains, and to provide insight into current research. It provides an updated overview of the research on ill-defined domains, the previous synopsis having been published in 2006 (Lynch et al. 2006).

The chapter is organized into four main sections. Sections 2, 3 and 4 correspond to three complementary and important perspectives that need to be considered to build ITSs for ill-defined domains. Section 2 addresses the characteristics of ill-defined domains, which is important to identify in order to categorize solutions for domains with similar features. Sections 3 and 4 next discuss the approaches to represent and reason with domain knowledge in these domains and suitable teaching models. As will later be explained, some approaches and models are more appropriate for certain types of ill-defined domains, which ease considerably the task of building ITSs. Lastly, Section 5 provides a detailed case study of an ITS called CanadarmTutor. This case study illustrates the advantages and limitations of several approaches discussed in the chapter.

5.2 What Is an Ill-Defined Domain?

An "ill-defined domain" or "ill-structured domain" in the context of ITS research is a domain in which traditional approaches for building ITSs are not applicable or do not work well (Aleven 2003; Aleven et al. 2006; Aleven et al. 2007). A domain is ill-defined because its structure or content makes it less suitable for supporting tutoring services. It is important to note that a complex domain is not necessarily an ill-defined domain, although many ill-defined domains are complex. A domain may be complex because it contains numerous knowledge elements and/or relations, and is still well-defined. An example from the field of geography is the name of each country and its capital. This domain is complex because there are hundreds of capitals. However, it is a well-structured domain because the knowledge can be represented simply as a list of pairs.

It is also important to note that the notion of an ill-defined domain is vague. Indeed, there are no clear boundaries between ill-defined and well-defined domains. Rather, there is a continuum ranging from well-defined to ill-defined. To identify ill-defined domains, to compare domains on the basis of their ill-definedness, and to categorize the strategies and approaches for supporting tutoring services in these domains, we first need to identify what characterizes ill-defined domains. The following section addresses this issue.

5.2.1 Characteristics of Ill-Defined Domains

To provide a working definition of an ill-defined domain, Lynch et al. (Aleven et al. 2008) did an extensive review of the research in the field of ITS and of the literature on "ill-structured problems" in artificial intelligence and decision-making in uncertain conditions. Lynch et al. concluded that domains having one or more of the following characteristics are ill-defined (Lynch et al. 2006):

(1) Multiple and controversial solutions: Domains having problems with many controversial solutions and no clear procedures for evaluating a solution are ill-defined. An example of such a domain is the design of entity relationship diagrams from text descriptions. There is a potentially infinite number of diagrams

with respects to a description, but the evaluation process of a diagram is partly subjective. A second example is law argumentation. Many legal arguments can be proposed for a legal case, yet there is not a single right solution even though some solutions may be preferable, depending on certain aspects, such as aesthetics and their success in a courtroom (Lynch et al. 2006). A third example is the domain of ethics. Ethical problems by definition have no right answer.

(2) No complete formal domain theory: Domains that do not have a clear or complete domain theory for determining a problem's outcome and testing its validity are ill-defined. For example, in the domains of music composition and architecture, there are only incomplete theories. In contrast, in the well-defined domain of geometry, there is a single theory applicable to all geometry problems.

(3) Ill-defined task structure: From the perspective of task structure, three domain types have been identified by Lynch et al. The first two are ill-defined. First, "design domains" contain tasks involving the design of new artefacts (e.g., writing a story and composing music). These domains are ill-defined as the goal is novelty, although some general rules or principles can provide guidance. The second main domain type is "analytical domains." Analytical tasks typically require performance analyses of incomplete and potentially incorrect information regarding a changing environment in order to make decisions. For this reason, these domains are also said to be ill-defined. Two examples of analytical tasks are stock trading and medical diagnoses. The third domain type is "problem-solving domains." These domains involve applying a formal theory to solve problems having a clear and complete definition in order to obtain a definite and verifiable answer. Such domains are considered well-defined. Most mathematical problems that do not involve the construction of new knowledge are examples of this domain type (e.g., calculating the volume of a sphere given its radius).

(4) Open-textured concepts: "Open-textured concepts" are abstract concepts that are partially undetermined or do not have absolute definitions. They are problematic when they need to be applied in concrete situations to carry out tasks. Domains, including open-textured concepts, are ill-defined. They include most domains that rely on natural language because words and sentences can have ambiguous meanings. Another example is the domain of law. In this domain, many domain concepts are abstract and can have several interpretations.

(5) Overlapping sub-problems: Domains having complex problems that cannot be divided into smaller independent sub-problems that are easier to solve are also said to be ill-defined. A general example is the problem of building a house. One must chose an appropriate site for construction and a plan for building the house. Since these two tasks are dependent, they cannot be planned separately without compromising the success of the whole task.

5.2.2 Ill-Definedness of Tasks and Domains

Recently, Mitrovic & Weerasinghe (2009) argued that ITS researchers need to consider two dimensions in ill-defined domains: "tasks" and "domains." They stated that both can be "ill-defined" or "well-defined" resulting in four different

combinations of types of tasks/domains. A domain is viewed by Mitrovic & Weerasinghe as declarative domain knowledge or a domain theory that can be used in tasks (Ashley et al. 2002). For example, in the task of writing a story, the domain includes storytelling notions, whereas the task is to write a story.

However, Mitrovic & Weerasinghe did not provide the means to identify an example of an ill-defined domain containing well-defined tasks (Mitrovic and Weerasinghe 2009). Therefore, we argue that the domain dimension is debatable, especially from the point of view of ITS researchers, because the main goal of building an ITS is to provide support to the learner at the level of individual tasks (Woolf 2009). In fact, an ITS designer simply needs to consider the tasks and the subset of directly relevant domain knowledge in order to select appropriate techniques to provide tutoring services. Moreover, since a domain can contain both ill-defined and well-defined tasks, it does not seem relevant to discuss the ill-definedness of a domain as a whole. For example, this is the case for the domain of software engineering, in which an ITS could offer well-defined multiple-choice questions about the properties of UML diagrams, in addition to some ill-defined problems such as designing UML diagrams.

We believe that the confusion of "task" versus "domain" in the proposal of Mitrovic & Weerasinghe originates from the choice of the term "ill-defined domain" in the ITS context by Lynch et al. However, Lynch et al. (Lynch et al. 2006) are very clear to use the word "domain" instead of "problem" simply "to emphasize that the end goal of tutoring is typically general domain knowledge (...), not problem specific answers" and that the distinction between domains and problems "is immaterial." We therefore suggest considering only if a task is ill-defined and in what way it is ill-defined (including the domain knowledge directly used in the task) when choosing domain knowledge modelling and reasoning techniques for supporting tutoring services.

On the other hand, choosing appropriate tasks for teaching a domain of knowledge is, we believe, the responsibility of educational and domain experts. However, one should be aware that some ITS techniques are more appropriate for certain types of tasks and teaching models than others. For this reason, we suggest that educational/domain experts should ask ITS experts to evaluate the feasibility of supporting tutoring services for a given task and of a teaching model before deciding if it should be included in an ITS. Examples of teaching models that can be used in ill-defined domains are presented in Section 4.

5.2.3 Characteristics of Ill-Defined Tasks

To describe how tasks are ill-defined, we suggest using the definition of Simon (Simon 1978) as a complement to the definition of Lynch et al. The definition of Simon is based on the study of human problem-solving and on the research of building artificial problem-solvers. This definition is relevant to the ITS context because solving any exercise offered by an ITS can be viewed as a problem-solving task. Simon stated that a problem is ill-structured if it possesses one or more of the following features (Simon 1978):

(1) The indefinite starting point: The instructions or information necessary for solving the problem are incomplete or vague.
(2) The indefinite ending point: The criterion that determines if the goal has been attained is complex and imprecise. There could be multiple and controversial solutions (Aleven et al. 2008).
(3) Unclear strategies for finding solutions: There are no clear strategies for finding solutions at each step of the problem-solving process.
This definition is formulated in general terms and can thus cover a wide variety of domains. It is also consistent with the definition by Lynch et al. (whose definition was partly inspired by that of Simon) (Lynch et al. 2006). In the next section, we use both definitions to present the principal approaches to represent and reason with domain knowledge in ill-defined domains.

5.3 Representing and Reasoning on Domain Knowledge in Ill-Defined Domains

There are three traditional approaches for representing and reasoning on domain knowledge in problem-solving-based intelligent tutoring systems. This section reviews these approaches and highlights their limitations for ill-defined domains. It then presents two additional approaches for ill-defined domains.

5.3.1 Cognitive Approach and Model-Tracing

The first traditional approach for representing and reasoning on domain knowledge is the cognitive approach and model-tracing (MT). MT-tutors are generally built from cognitive task analysis. The process of cognitive task analysis consists of producing effective problem spaces or task models by observing expert and novice users (Koedinger et al. 1997) using different solving problems strategies.

A task model can be designed for a problem or a problem set. Task models are usually represented as sets of production rules (sometimes structured as a goal-decomposition tree (Woolf 2009)) or as state spaces in which each rule or transition corresponds to an action or an operation to perform a task. Some of the rules/transitions can be tagged as "buggy" or annotated with hints or other didactic information. The most well-known examples of model-tracing tutors are cognitive tutors (Koedinger et al. 1997) (see also chapter 3), which encode the operations for solving a problem as a set of production rules. When a learner performs a task with a cognitive tutor, the latter follows the learner's reasoning by analyzing the rules being applied. This process is called model-tracing The MT paradigm is beneficial because the reasoning processes of the learner can be represented in great detail (in fact, authors of cognitive tutors claim to model the cognitive processes of the learner), and the models obtained can support a wide variety of tutoring services, such as: (1) suggesting to the learner the next steps to take, (2) giving demonstrations; (3) evaluating the knowledge that the learner possesses in terms of the skills that are applied; and (4) inferring learner goals.

Model-tracing tutors are recommended for tasks in which the goal is to evaluate the reasoning process rather than simply determining if the learner attained the correct solution. MT-Tutors generally assume that the starting and ending points of a problem are definite. The main limitation of model-tracing with respect to ill-defined domains is that, for some domains, there are no clear strategies for finding solutions, and it can therefore be difficult to define an explicit task model. Moreover, for complex domains, one would need to determine a large number of rules and solution paths, and designing a set of rules or a state space for a task would be very time-consuming.

A strategy for using the MT paradigm in ill-defined domains is to describe only the well-defined part of a task with a task model (Lynch et al. 2006). This strategy is used, for example, in CanadarmTutor, an ITS used to teach astronauts how to operate a robotic arm [11]. Another strategy for enabling the MT to operate ITSs in ill-defined domains is to use an iterative approach to design task models (Ogan et al. 2006). Initially, a rough model is created by domain experts. The model is then evaluated empirically several times to see if it correctly predicts user-behavior. It is subsequently adjusted until a satisfactory model is attained. This strategy was used to build an ITS which enables the learner to distinguish verb tenses in the French language (Ogan et al. 2006).

5.3.2 Constraint-Based Modeling Approach

The second approach for representing and reasoning on domain knowledge is constraint-based modeling (CBM) (Mitrovic et al. 2007; Mitrovic and weerasinghe 2009). This approach is thoroughly described in chapter 4. It consists of specifying sets of constraints on what is a correct behavior or solution rather than to provide an explicit task model. When the learner violates a constraint during a task, the CBM Tutor diagnoses that an error has been made and provides help to the learner regarding the violated constraint.

Unlike MT-Tutors, CBM-Tutors do not support tutoring services such as to present demonstrations or suggest the next steps to perform to the learner. This is one of the principal limitations of the CBM approach. CBM is recommended for domains in which the goal is to validate states/solutions regardless of the strategies a learner uses to provide solutions (it can be applied to domains in which there are no clear strategies). However, in order for CBM to be applicable, one needs to define relevance and satisfaction conditions that characterize optimal states/solutions for a task. But, designing and selecting a set of constraints is not always easy. Moreover, for some tasks, states/solutions sometimes do not provide enough information to permit the specification of a set of relevant constraints, and, in some cases, a large number of constraints is required because there are too many diverse solutions (Kodaganallur et al. 2006). Another limitation of CBM is that CBM-Tutors do not take into account the solution path leading to a constraint violation. This limitation can have two negative implications. First, the help generated may be inadequate, especially when a solution path differs significantly from the ideal solutions (Woolf 2009). Second, if the reasoning that led to the solution is not

evaluated, the CBM-Tutor may be unable to distinguish that a learner may have intentionally or unintentionally answered a question correctly.

Despite these limitations, CBM has been applied successfully to some ill-defined domains, in particular, to problem-solving and design tasks (see (Mitrovic et al. 2007) for an overview of CBM applications). An example of a CBM-Tutor is KERMIT/EER-Tutor, an ITS used to teach design entity relationship diagrams (Mitrovic et al. 2007). Designing such diagrams is an ill-defined task because it involves creativity (it is a design task), there is no clear strategy for performing it, problems statements are generally ambiguous and for one problem there can be many acceptable and controversial solutions. When applying CBM in KERMIT, approximately 100 constraints were defined and KERMIT's interface was designed to restrict the learner to select terms from problem descriptions to name diagram elements. The latter restriction greatly reduces the scope of possible solutions. It also illustrates a popular design strategy for building ITSs in ill-defined domains, in general, i.e., forcing a structure into a domain so as to transform it into a better defined form. However, a drawback of this strategy, from a pedagogical perspective, is that the learner eventually has to learn to perform tasks in less-structured problem-solving environments (Moritz and Blank 2008).

5.3.3 The Expert System Approach

The third approach for representing and reasoning on domain knowledge consists of integrating an expert system in an ITS (Clancey 1984; Graesser et al. 2000; Kabanza et al. 2005; Moritz and Blank 2008). This approach is very broad because many forms of expert systems can be used, such as rule-based, neural networks, decision trees and case-based reasoning systems. The advantage of this approach is that tailored expert systems are particularly well-suited for some domains, unlike CBM and MT that are general approaches. There are two principal and complimentary means of using an expert system in an ITS.

First, an expert system can be used to generate expert solutions. The ITS can then compare these solutions with learner solutions. It can subsequently use them as demonstrations or to suggest to the learner the problem-solving steps he/she should take. For example, this approach was used in GUIDON (Clancey 1984), an ITS used to teach the learner how to diagnose infectious diseases based on a patient's history of clinical tests. GUIDON relies on MYCIN, a rule-based expert system containing approximately 500 rules to generate expert solutions. These solutions are then compared with the learner's solutions in order to diagnose mistakes, generate demonstrations, and produce mixed initiative dialogues. MYCIN is particularly well-suited for use in an ITS, as the rules in its domain are meaningful to the learner. GUIDON uses this property extensively to present rules to the learner that support or refute his/her reasoning.

The second principal means for using an expert system in an ITS is to compare ideal solutions with learner solutions (Clancey 1984, Graesser et al. 2000). The first example is AutoTutor, (Graesser et al. 2000) an ITS which was applied to several domains, including Newtonian physics and computer literacy. AutoTutor teaches by conducting conversations with learners in natural language. To evaluate the

student's natural language answers, AutoTutor uses Latent Semantic Analysis (LSA) to compute the semantic distance with the expected answers. LSA is a black-box technique in the sense that it cannot explain its "reasoning," in contrast to expert systems such as MYCIN. Despite this limitation, it is very effective in assessing natural language answers.

The second example of an ITS that relies on an expert system to compare learner solutions with ideal solutions is DesignFirst-ITS (Moritz and Blank 2008), an ITS which assists the learner in designing UML diagrams from text descriptions. The design of UML diagrams is an ill-defined task because it is a design task, the solution space is very large, there is no right solution, problem statements are ambiguous, and it uses natural language (Moritz and Blank 2008). The uniqueness of DesignFirst-ITS is that it offers an almost completely unconstrained problem-solving environment with the learner, unlike other ITS for UML, such as Collect-UML, which relies on CBM (Moritz and Blank 2008). DesignFirst-ITS evaluates the step by step construction of UML diagrams by comparing them with templates of acceptable solutions. The matching process is not insignificant as it searches for synonyms, spelling errors, adjective use, etc.

In all of these cases, the expert systems approach provides advanced tutoring services that would be hard to offer for the same domains with the MT or CBM approaches. However, the limitations of the expert system approach are the following: (1) developing or adapting an expert system can be costly and difficult, especially for ill-defined domains; and (2) some expert systems cannot justify their inferences, or provide explanations that are appropriate for learning.

5.3.4 The Partial Task Modeling Approach

The three aforementioned classical approaches for representing and reasoning on domain knowledge can be very time consuming and difficult to apply in some ill-defined domains. As an alternative to these approaches, a promising approach for ill-defined domains is to apply data-mining or machine-learning techniques to automatically learning partial task models from user solutions. The rationale for this approach is that a partial task model could be a satisfactory alternative to an exhaustive task model in domains in which a task model is difficult to define.

This approach was demonstrated in CanadarmTutor (Fournier-Vigier et al. 2009; Kabanza et al. 2005) (mentioned in section 3.1), an ITS for learning to operate Canadarm2, a robotic arm deployed on the international space station which has seven degrees of freedom. In CanadarmTutor, the main learning activity is to move the arm from an initial configuration to a goal configuration in a 3D-simulated environment. To move the arm, the operator must select every minute the three best cameras to view the operation scene. He must next select and perform appropriate joint rotations to move the arm, while avoiding collisions and dangerous configurations. Moving Canadarm2 is an ill-defined task because there is an almost infinite number of solution paths and no simple "legal move generator" to find the best operations to perform at each step.

To allow the system to learn partial task models automatically, CanadarmTutor was modified to record each attempt to solve a problem in a database as a sequence of actions annotated with contextual information, such as the success or failure of the attempt (Fournier-Vigier et al. 2009) Data-mining algorithms were then applied to extract partial solution paths that regularly occur for each problem. The idea is that even if there are many solution paths for a problem, some parts occur frequently, and, thus, could be used to support tutoring services. In CanadarmTutor, exploiting these frequent paths allows for supporting tutoring services that are comparable to what a MT-tutor can provide for a well-defined task (Fournier-Vigier et al. 2009).

The approach of learning partial task models is appropriate for problem-solving tasks in which: 1) the initial state and the goal state are clear; 2) there is a large number of possibilities; 3) there is no clear strategy for finding the best solution; and 4) solution paths can be expressed as sequences of actions. The advantages of this approach are that it does not require any specific background knowledge by the domain expert, and the system can enrich its knowledge base with each new solution. Also, unlike the expert system approach, this approach is based on human solutions. On the other hand, no help is provided to the learner if part of a solution path was previously unexplored. One way to address this limitation is to combine this approach with other approaches, such as CBM or MT (see next subsection). A second limitation of the approach is that it needs to be applied to each problem. However, this may be a compromise worth accepting if a collection of exercises can be set up and administered to many students.

5.3.5 The Hybrid Approach

The last approach for representing and reasoning on domain knowledge is to combine two or more of the aforementioned approaches. The goal of this approach is to combine the advantages of different approaches in order to overcome their limitations for ill-defined tasks. The motivation behind having the hybrid approach is that different approaches can be better suited for different parts of the same ill-defined task so as to offer common or complementary tutoring services. For example, in CanadarmTutor, an expert system (Kabanza et al. 2005), a cognitive model (Fournier-Vigier et al. 2008) and the partial task model approach (Fournier-Vigier et al. 2009) are integrated to provide assistance for different parts of the arm manipulation task. The result is tutoring services which greatly exceed what was possible to offer with each individual approach, for this domain (Fournier-Vigier et al. 2009). According to Lynch et al. (Lynch et al. 2006), the hybrid approach is one of the most promising ways of supporting tutoring services in ill-defined domains.

In conclusion, Section 4.3 describes five approaches for representing and reasoning on domain knowledge in ill-defined domains for ITSs. The description of these five approaches provides a general overview of the techniques available.

5.4 Teaching Models for Building Intelligent Tutoring Systems in Ill-Defined Domains

Section 4 presents the challenge of building ITS in ill-defined domains from a complementary perspective, that is to say, suitable teaching models. Choosing a teaching model for a domain is an important decision because it can considerably facilitate or impede the conception of an ITS.

5.4.1 Structuring Learning around Case Studies

The first effective teaching model for ill-defined domains, which has partial domain theories and in which practitioners may have to draw analogies with past cases (e.g., medical diagnoses and law argumentation), is a model in which the learning is structured around the study of cases.

An example of an ITS which implements this strategy is CATO (Aleven 2003). The goal of CATO is to teach beginning law students the basic skills of developing arguments with cases (Ashley et al. 2002). In particular, it is designed to teach the skill of distinguishing legal cases, which consists in demonstrating that a case is significantly different from another, to suggest that it needs to be decided differently. Identifying positive distinctions among cases for the person invoking the argument is a difficult and ill-defined task as natural language is used and since verifiable and accurate solutions are rarely determined (Ashley et al. 2002). CATO presents good and bad examples with interactive explanations to teach the learner the skills to distinguish among cases, Lynch et al. 2006; Simon 1978. To support these tutoring services, CATO uses an expert system which incorporates a set of cases indexed with legal factors (Aleven 2003). The latter is also used in CATO-Dial (Ashley et al. 2002), a variation of CATO that engages the student in simulated courtroom arguments.

CATO and CATO-Dial implement elaborate tutoring services to support learning with cases and to teach learners how to compare cases. However, this is not the only way to support learning with cases. A simpler strategy exists to enable the learner to analyse or practice one case at a time, such as is the case in GUIDON (Clancey 1984).

5.4.2 Supporting Metacognition

Another teaching model that has successfully been used in ITSs for ill-defined domains is a model which provides metacognitive support to the learner so as to improve his/her ability to acquire knowledge. This model also gives him/her some support to learn domain knowledge. The advantage of this teaching model is that it does not require a detailed model of domain expertise.

An example of an ITS that supports metacognition in an ill-defined domain is the experimental tutoring system designed by Walker et al. (2008) for teaching intercultural competence skills. This domain is ill-defined because explaining

cultural behavior is based on the interpretation of events and language, and, although some rules exist, there is not a complete formal theory (Walker et al. 2008). A student who uses the system by Walker et al. is asked to watch clips of a French movie that relate to immigration. He/she is then required to answer questions about the clips. Lastly, he/she must contribute to a forum discussion. Each time the learner contributes to the forum, the system uses keyword analysis algorithms to evaluate the writing. The evaluation is based on five quality measures, for example, if the writing is on topic and if it shows awareness of different points of views. The ITS then generates advice for improvement. The learner is subsequently required to make at least one revision to the post before publishing it. Although the system has no domain model of what constitutes intercultural competence skills, it fosters learning by promoting good learning behavior (writing posts which satisfy quality measures) (Walker et al. 2008). The system of Walker et al. uses an expert system approach to support metacognition. However, MT and CBM (Mitrovic et al. 2007) can also be used to support metacognition.

5.4.3 Supporting Inquiry Learning

A third teaching model used in ITSs for ill-defined domains is a model which supports inquiry-learning i.e., a constructivist approach to learning (Woolf 2009). Inquiry learning consists of constructing knowledge by discovery, gathering data and testing hypotheses. The role of an ITS for inquiry learning is to support the inquiry process rather than to provide knowledge to the learner. Thus, the ITS does not require a sophisticated domain model. For this reason, inquiry learning is an appropriate teaching model for many ill-defined domains in which there is no clear or complete formal theory. ITSs for inquiry learning can provide different types of support, i.e., controlling the inquiry process, providing help at a metacognitive level concerning what is a good inquiry behavior and evaluating the reasoning of the learner to give tailored advice (Woolf 2009).

An example of an ITS for inquiry learning is Rashi (Dragon et al. 2006), a generic ITS which was applied to several domains, including forestry, human biology, history and geology. Rashi presents cases to the learner and enables him/her to gather data, formulate hypotheses, and develop arguments in order to verify his/her hypotheses. The learner can collect information about a case with data-collecting tools, such as an interview tool or an image explorer tool. He can next construct his/her arguments using the argument-building tool. Rashi supports learning at a metacognitive level, for example, by promoting the consideration of multiple hypotheses, or by encouraging top-down or bottom-up argument construction. Rashi also uses a limited domain model to detect faulty relationships in arguments and to suggest missing relationships.

An interesting feature of Rashi for ill-defined domains is that it allows the learner to enter additional evidence that is not predefined in the domain model to support arguments. The importance of building ITSs that take into account the impact of the learner's background knowledge on his/her reasoning has also been noted by Easterday et al. (2007), in the domain of policy problems. A policy problem consists of judging the likelihood that a policy will lead to a desired outcome

(Easterday et al. 2007). Policy problems are ill-defined because there are no objectively correct answers and no agreed upon strategies for representing problems (Easterday et al. 2007). Easterday et al. observed that novices and experts who solve policy problems sometimes: (1) disregard the overriding evidence because they rely on their background knowledge; and (2) speculate about the outcomes because they are influenced by their background knowledge. To handle these types of errors, Easterday et al. proposed several solutions: (1) to ask the student to reason with the evidence and with his/her background knowledge separately before reasoning with both; (2) to design user interfaces that allow the learner to add his/her background knowledge; and (3) to ask the learner to articulate his/her reasoning and to identify at what point his/her background knowledge intervenes.

Recently, Easterday et al. proposed PolicyWorld (Easterday), an ITS for teaching deliberation for policy problems. PolicyWorld supports deliberation for policy problems by supporting searching for evidences and comprehending them, building interpretation of policy problems as causal diagrams, relating these diagrams to evidences, synthesizing diagrams, and taking decisions based on them. PolicyWorld does not allow the learner to enter his/her background knowledge. However, the tutoring services provided can indirectly handle errors that are caused by the learner's background knowledge. For example, they can check if the learner's beliefs appear consistent with the evidence collected. PolicyWorld uses the hybrid approach to represent and reason on domain knowledge by combining MT, CBM and the expert system approach. It relies on an inquiry-based teaching model.

5.4.4 Using Interactive Narratives

A fourth teaching model that has been used in ITSs for ill-defined domains is a model which offers "interactive narratives." A system that provides interactive narratives puts the learner in stories and enables him/her to make decisions that directly affect the story's direction and/or outcome (Hodhod and Kudenko 2008). This approach is demonstrated in AEINS (Hodhod and Kudenko 2008), an ITS for learning ethics through moral dilemmas. AIENS takes part in narratives in which the learner is faced with moral dilemmas. Making decisions in moral dilemmas is an ill-defined task because there is no right answer. The learner makes the final judgement of what is right or wrong. AEINS gives freedom to the learner to act and make decisions which permits the learner to learn from his/her decisions. When there are dilemmas, AEINS uses the Socratic Method to conduct a dialogue so as to encourage the learner reflect on the implications of the decisions made. AIENS has the capability to generate and adapt narratives spontaneously in order to provide situations that are tailored to each learner. AEINS relies on the MT approach to update a student model.

Learning with interactive narratives is similar to inquiry learning as it is also a form of learning by discovery. However, it differs from inquiry learning in that the learner does not have to gather data, nor to build and test hypotheses in a structured fashion.

5.4.5 Structuring Learning around Collaboration

A fifth teaching model for ill-defined domains is a model which structures learning around collaboration. The goal of this model is to make the student learn by working and exchanging information and ideas with his/her peers. The benefit of this strategy for ill-defined domains is that supporting collaboration can replace the need to build a domain model.

An ITS which supports collaboration can offer various levels of control in interactions. It can guide collaboration, provide advice on how to improve interactions, and display aggregated data regarding interactions so that the participants can decide on appropriate remedial actions (Soller et al. 2005). One example of a collaborative system in ill-defined domains is that of Walker et al. (Walker et al. 2008). As previously mentioned, Walker et al.'s system fosters learning by helping learners make useful contributions to forum discussions.

One approach to collaboration that merits mention is that of creating virtual companions that interact with the learner (Chou et al. 2003). The learner can compete, collaborate and even learn by teaching his/her virtual companions. Although building a virtual companion may not require a detailed domain model, designing a virtual companion can be challenging.

5.4.6 Other Teaching Models

The list of teaching models presented in this section is not exhaustive and many variants of the presented teaching models are possible. The intent of this section is to present a set of less conventional teaching models. But it is also possible to adopt more traditional models. For instance, CanadarmTutor (Fournier-Vigier et al. 2008 and 2009; Kabanza et al. 2005) adopts a problem-solving teaching model in which the principal learning activity is to solve problems akin to most ITSs which teach problem-solving tasks.

5.5 A Case Study: CanadarmTutor

The two previous sections listed several examples of ITSs and discussed the strategies used for representing and reasoning on domain knowledge in ill-defined domains and suitable teaching models. The focus of Section 5 is to look at the case study of CanadarmTutor in more detail. It was deemed appropriate to analyse CanadarmTutor in greater depth due to the fact that there have been several studies on the different approaches for representing and reasoning on domain knowledge with this ITS. This case study, therefore, allows for the comparison of various approaches in the same domain.

5.5.1 CanadarmTutor

CanadarmTutor is an ITS that we have developed (Fournier-Vigier et al. 2008 and 2009; Kabanza et al. 2005) (depicted in Figure 5.1.a). It is a simulation-based

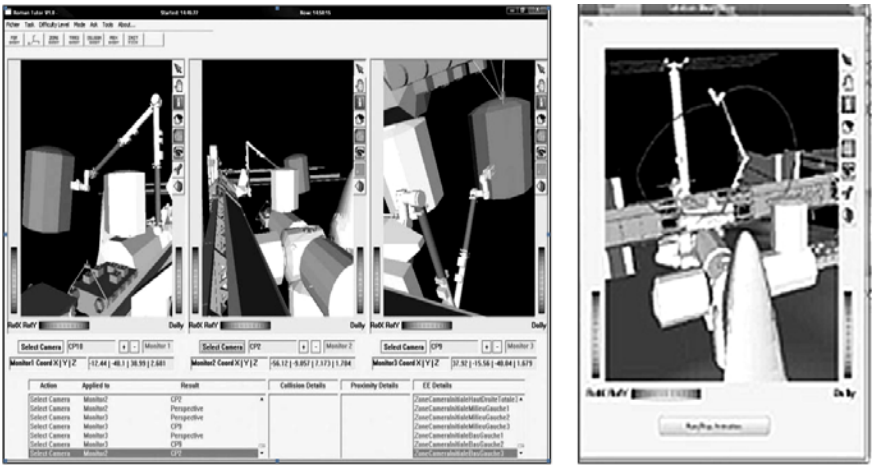

Fig. 5.1 (a) The CanadarmTutor user interface, (b) a path-planner demonstration

tutoring system to teach astronauts how to operate Canadarm2, a robotic arm deployed on the International Space Station (ISS) that provides 7 degrees of freedom. As previously explained, the main learning activity in CanadarmTutor is to move the arm from a given configuration to a predetermined configuration. Operating Canadarm2 is a difficult task since operators do not have a direct view of the scene of operation at the space station and must rely on cameras mounted on the manipulator and on strategic places in the environment where it operates. To move the arm, the operator must select at every moment the best cameras for viewing the scene of operation. Moreover, an operator has to select and perform appropriate joint rotations to move the arm, while avoiding collisions and dangerous configurations. To provide domain expertise to CanadarmTutor, the three following approaches were applied (Fournier-Vigier et al. 2008 and 2009; Kabanza et al. 2005).

5.5.2 The Expert System Approach

Initially, a special path-planner was integrated into CanadarmTutor. The path-planner is based on a probabilistic roadmap approach (Kabanza et al. 2005) and can produce examples of correct and incorrect motions in training. It acts as a domain expert and can calculate the arm's moves avoiding obstacles, and consistent with the best available camera views to achieve a given goal (Kabanza et al. 2005). The path-planner makes it possible to track the learner's solution step- by-step, and to generate demonstrations when necessary (as depicted in Figure 5.1.b). However, the generated paths are not always realistic or easy to follow since they are not based on human experience and cannot provide help on important aspects of the manipulation task, such as selecting cameras and adjusting their parameters. Also, the path-planner cannot support important tutoring services, such as estimating the learner's knowledge gaps, as there is no knowledge or skills representation.

5.5.3 The Model-Tracing Approach

To sustain more effective learning, a cognitive task analysis was performed, and an effective problem space that captures real user-knowledge was defined (Fournier-Vigier et al. 2008). To model the manipulation of Canadarm2, we were inspired by the research on spatial cognition, which states that spatial representations used for complex spatial reasoning are encoded as semantic knowledge (Fournier-Vigier et al. 2008). We used a custom cognitive model including a semantic retrieval mechanism to model the recall of spatial knowledge during the task (Fournier-Vigier et al. 2008). To model the spatial knowledge, we discretized the 3D space into 3D sub spaces called elementary spaces (ES). Spatial knowledge was then encoded as relationships, such as: (1) a camera can see an ES or an ISS module; (2) an ES contains an ISS module; (3) an ES is next to another ES; and (4) a camera is attached to an ISS module. The procedural knowledge of how to move the arm to a goal position was modeled as a loop in which the learner must recall a set of cameras to view the ESs containing the arm, select the cameras, adjust their parameters, retrieve a sequence of ESs to go from the current ES to the goal, and move to the next ES. CanadarmTutor detects all the atomic actions, such as camera changes and entering/leaving an ES. Performing model tracing with the cognitive model allows CanadarmTutor to offer the following six important tutoring services (Fournier-Vigier et al. 2008).

First, the learner can freely explore the task model to learn how to operate the arm. The learner can also consult the declarative knowledge associated with the task model to learn about the properties of the space station, the cameras and Canadarm2. Second, model-tracing allows CanadarmTutor to evaluate the knowledge acquired by the learner during the arm manipulation exercises. After a few exercises, CanadarmTutor builds a detailed learner profile that shows the strengths and weaknesses of the learner in terms of mastered, missing and buggy knowledge. The third tutoring service evaluates the declarative knowledge linked to the task model by asking direct questions, such as "Which camera can be used to view the Node02 ISS module?" The fourth tutoring service provides hints and demonstrations on request during arm manipulation exercises. The next step is suggested by model-tracing. CanadarmTutor can give messages, such as: "If you have finished adjusting the third monitor, then you should start moving the arm." Demonstrations are generated dynamically due to model-tracing. The fifth tutoring service generates personalized exercises based on the student model. During a training session, CanadarmTutor relies on the student model to generate exercises that progressively involve new knowledge and knowledge judged to be not yet mastered by the learner. For instance, CanadarmTutor can suggest an exercise involving a camera that has been rarely used by the learner. The sixth and last tutoring service offers proactive help to the learner. When the proactive help is activated, CanadarmTutor, for example, warns the learner that another camera should be selected if the arm has moved to an area that is not visible by the currently selected cameras. This type of help is particularly appreciated by beginners.

5.5.4 The Automatic Acquisition of Partial Task Models

Although the task model specified by the hand provides a precise cognitive assessment of the learner's knowledge for the main steps of the manipulation task, it does not go into finer details such as how to select joint rotations for moving Canadarm2. Being that Canadarm2 is an arm with seven joints, there are multiple possibilities on how to move the robot to a goal configuration for a given robot manipulation problem. Since one must also consider the safety and the facility of each manoeuvre, it is very difficult to define a "legal move generator" to generate the joint rotations that a person could execute. In fact, some joint manipulations are preferable to others, based on several criteria, which are difficult to determine, such as the view of the arm given by the cameras chosen at a given moment, the relative position of obstacles to the arm, the arm configuration (e.g. avoiding singularities) and the familiarity of the user with certain joints manipulations. It is thus not possible to define a complete and explicit task model for this task (this task is ill-defined according to the definition of Simon (1978)). On the other hand, the path-planner sometimes provides paths that are too complex and difficult to be executed by the user, as the paths are not based on human solutions.

Due to these difficulties, we have applied the fourth approach, the automatic acquisition of partial task models (Fournier-Vigier et al. 2009). This approach is carried out in three phases in CanadarmTutor.

The first phase records the user's solutions when he/she attempts an exercise. In CanadarmTutor, one exercise is to move the arm from an initial configuration to a goal configuration. For each attempt, a *sequence of events* is created in a database. We define an event as a set of actions carried out by the learner that are considered to be simultaneous. In CanadarmTutor, we defined 112 actions that can be recorded, including: (1) selecting a camera; (2) performing an increase or decrease of the pan/tilt/zoom of a camera; and (3) applying a rotation value to one of the seven arm joints. An example of a partial action sequence recorded for a user in CanadarmTutor is <(0, *rotateSP*{2}), (1, *selectCP3*), (2, *panCP2*{4}), (3, *zoomCP2*{2})>, which represents decreasing the rotation value of joint SP by two units, selecting camera CP3, increasing the pan of camera CP2 by four units and then its zoom by two units. Furthermore, each sequence can have annotations called "dimensions" (Fournier-Vigier et al. 2009). Table 5.1 shows an example of a toy database containing six learner plans annotated with five dimensions. In this table, the single letters *a*, *b*, *c*, and *d* denote actions. The first dimension "*Solution state*" indicates if the learner plan is a successful or a buggy solution. In the case of CanadarmTutor, values for this dimension are produced by the tutoring system. The four other dimensions are examples of dimensions that can be added manually. While the dimension "Expertise" denotes the expertise level of the learner who performed a sequence, "Skill_1", "Skill_2" and "Skill_3" indicate if any of the three specific skills were demonstrated by the learner when solving the problem. This example illustrates a five dimensional database. However, any kind of learner information or contextual information can be encoded as a dimension. In CanadarmTutor, we used 10 skills, and the dimensions of "solution state" and "expertise level" to annotate sequences (Fournier-Vigier et al. 2009).

Table 5.1 An example toy database containing 6 user solutions

ID	Dimensions					Sequence of actions
	Solution state	Expertise	Skill_1	Skill_2	Skill_3	
S1	successful	expert	yes	yes	yes	<(0,a),(1,bc)>
S2	successful	novice	no	yes	no	<(0,d) >
S3	buggy	expert	yes	yes	yes	<(0,a),(1,bc)>
S4	buggy	intermediate	no	yes	yes	<(0,a),(1,c), (2,d)>
S5	successful	expert	no	no	yes	<(0,d), (1,c)>
S6	successful	novice	no	no	yes	<(0,c), (1,d)

Table 5.2 Some frequent patterns extracted from the dataset of Table 1 with a minsup of 33 %

ID	Dimensions					Sequence of actions	Support
	Solution State	Expertise	Skill_1	Skill_2	Skill_3		
P1	*	expert	yes	yes	yes	<(0,a)>	33 %
P2	*	*	*	yes	yes	<(0,a)>	50 %
P3	*	expert	yes	yes	yes	<(0,a), (1,b)>	33 %
P4	successful	*	no	*	*	<(0,d)> <(0,c)>	50 %
P5	successful	expert	*	*	yes	<(0,d)>	33 %
P6	successful	novice	no	*	no		33 %

The second phase extracts partial task models from user plans. In order to accomplish this, CanadarmTutor applies a custom sequential pattern mining algorithm that we developed (see (Fournier-Vigier et al. 2009) for more details), which takes a database of user solutions as input and a user-defined threshold called *minsup*. The output is the set of all subsequences that appears in at least *minsup* sequences of the database. When applying the algorithm, it is possible to specify time constraints on subsequences to be discovered (see (Fournier-Vigier et al. 2009) for detailed information). For example, Table 5.2 shows some subsequences (also called patterns) found from the database shown in Table 5.1 with *minsup* = 33 %. When taking pattern P3 into consideration, pattern P3 represents doing action *b* one time unit immediately after action *a*. The pattern P3 appears in sequences S1 and S3 of Table 1. It has thus a support of 33 % or two sequences. Moreover, the annotations for P3 tell us that this pattern was performed by expert users who possess skills "Skill_1", "Skill_2" and "Skill_3" and that P3 was found in plans that failed, as well as plans that succeeded.

The third phase uses the partial task model for supporting tutoring services. We implemented three tutoring services (Fournier-Vigier et al. 2009). The two first tutoring services rely on a plan recognition algorithm, which after each learner action during an exercise, identifies the patterns that best match the last actions performed (see (Fournier-Vigier et al. 2009) for details on the algorithm).

The first tutoring service is to assess the profile of the learner (expertise level, skills, etc.) by looking at the patterns applied. If, for example, 80 % of the time a learner applies patterns with a value "intermediate" for the dimension “expertise,” then CanadarmTutor can assert with confidence that the learner's expertise level is "intermediate." In the same way, CanadarmTutor can diagnose mastered and missing/buggy skills for the user who demonstrated a pattern by looking at the “skills” dimensions of patterns applied (e.g., “Skill_1” in Table 2).

The second tutoring service is to guide the learner. This tutoring service consists of determining the possible actions from the set of patterns and proposing one or more actions to the learner. In CanadarmTutor, this functionality is triggered when the student selects "What should I do next?" in the interface menu. CanadarmTutor then identifies the set of next actions possible according to the matching patterns found by the plan recognition algorithm.

Finally, a tutoring service was implemented in CanadarmTutor to enable the learner to explore patterns so as to discover possible methods to solve problems. Currently, the learner can explore a pattern with an interface that lists the patterns and their annotations and provides sorting and filtering functions (for example to display only patterns leading to success).

5.5.5 The Hybrid Approach

Although learning partial task models from user solutions in CanadarmTutor is useful in helping the learner to manipulate the joints –a task which was impossible with the cognitive model or the path-planner (Fournier-Vigier et al. 2009), no assistance is provided to the learner if part of a solution path has not been explored by other users. Since the three approaches that we applied to CanadarmTutor have advantages and disadvantages, we decided to combine them to create a hybrid model. This hybrid model works as follows.

When the learner performs an exercise with CanadarmTutor, the model-tracer uses the cognitive model to update the student model (as previously explained). This latter contains probabilities that knowledge units defined in the cognitive model are mastered by the learner. When the learner answers the questions asked by CanadarmTutor, the student model is also updated.

When a user completes, either successfully or unsuccessfully, an arm manipulation exercise), the solution is added to the sequence database of user solutions for that exercise. The knowledge mastery levels from the student model are used to annotate the sequence. Other annotations can be added, for example, information regarding the success or failure of the student and manual annotations. When a minimum of a few sequences have been recorded, the data mining algorithm is applied to extract a partial task model for the exercise. The patterns extracted include skills from the cognitive model as annotations.

When CanadarmTutor detects that that learner is following a pattern from the partial task model during an exercise, the annotations of the pattern are also used to update the student model. For example, if a learner applies a pattern common to the learners possessing “skill_1,” the mastery level of “skill_1” in the student model will be increased. As a result, the partial task model is also used to update

the student model (the student model is now shared by the cognitive model and the partial task model).

During a learning session, CanadarmTutor uses the student model to generate exercises that progressively integrate knowledge that is judged to be not yet mastered by the learner. Generated exercises are questions about the cognitive model and its declarative knowledge, or manipulation exercises.

When the learner asks for help regarding the next step of a manipulation exercise, three different types of system assistance is provided. First, the cognitive model gives a general procedure that should be followed to move the arm. For example, the cognitive model might say to the student, "If you have finished adjusting the third monitor, then you should start moving the arm." This assistance is generated by model-tracing with the cognitive model. In the same window, the patterns from the partial task model that match the current user solution are then displayed to the learner. These patterns provide information pertaining to the joint rotations that should be performed to move the arm. If no pattern matches the current learner solution, a demonstration generated by the path-planner is shown to the learner to illustrate a possible solution path. This information is complementary.

The learner can also explore patterns from the partial task models, as explained earlier, which illustrate possible alternatives for solving problems. The learner can also explore the cognitive model to learn the general procedure for moving the arm. In addition, at any time, the learner can ask for demonstrations from the path-planner or the cognitive model.

Lastly, as previously discussed, CanadarmTutor can use the cognitive model to generate proactive help to the learner, such as giving suggestions on selecting cameras.

We recently conducted a preliminary evaluation of the new version of CanadarmTutor with five users who had experience using the previous versions of CanadarmTutor. The study consisted of having each user try the new version for one hour. We then interviewed them regarding their experience using the new version of CanadarmTutor. All users agreed that the new version of CanadarmTutor is more comprehensive in all aspects of the manipulation task than are the previous versions they had used. They unanimously agreed that integrating the three approaches into CanadarmTutor resulted in better tutoring services. We are currently working on enhancing CanadarmTutor with more elaborated pedagogical strategies. We are also preparing an extensive empirical study to evaluate formally how the newest version of CanadarmTutor contributes to fostering learning.

5.6 Conclusion

Research on ill-defined domains presents many challenges to ITS researchers. However, it is an exciting area of research as many domains are unexplored and remain ill-defined. Research on ill-defined domains will undoubtedly result in the creation of ITSs in domains which, until recently, have been overlooked in ITS research.

In this chapter, we provided a synopsis of the problems and solutions for building ITSs for ill-defined domains. Three complementary and important perspectives were taken into consideration when creating ITSs for ill-defined domains: (1) the characteristics of ill-defined domains; (2) the approach for representing and reasoning on domain knowledge; and (3) the teaching models. Throughout the chapter, we presented several examples of ITSs in order to illustrate their different approaches and to highlight some of their advantages and disadvantages. We specifically presented the case-study of CanadarmTutor in order to compare four approaches for representing and reasoning on domain knowledge in the same ITS.

We believe it is important for future researchers, to further investigate domain-specific and general approaches to represent and reason on domain knowledge, since the former can be more effective in specific domains. We also believe that creating hybrid models for representing and reasoning on domain knowledge, such as that used in CanadarmTutor, is a promising approach which needs to be explored in greater depth.

Acknowledgments. Our thanks go to the NSERC (Natural Sciences and Engineering Research Council), and the FQRNT (Fond Québécois de Recherche sur la Nature et les Technologies) for their logistic and financial support.

References

Aleven, V.: Using background knowledge in case-based legal reasoning: a computational model and an intelligent learning environment. Artif. Intell. 150, 183–237 (2003)

Aleven, V., Ashley, K., Lynch, C., Pinkwart, N.: Proc. ITS for Ill-Defined Domains Workshop. ITS 2006 (2006)

Aleven, V., Ashley, K., Lynch, C., Pinkwart, N.: Proc. Workshop on AIED applications in ill-defined domains. AIED 2007, Los Angeles, USA (2007)

Aleven, V., Ashley, K., Lynch, C., Pinkwart, N.: Proc. ITS for Ill-Defined Domains Workshop. ITS 2008, Montreal, Canada (2008)

Ashley, K.D., Desai, R., Levine, J.M.: Teaching Case-Based Argumentation Concepts Using Dialectic Arguments vs. Didactic Explanations. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 585–595. Springer, Heidelberg (2002)

Chou, C.-Y., Chan, T.-W., Lin, C.-J.: Redefining the learning companion: The past, present, and future of educational agents. Comput. and Educ. 40, 255–269 (2003)

Clancey, W.: Use of MYCIN's rules for tutoring. In: Buchanan, B., Shortliffe, E.H. (eds.) Rule-Based Expert Systems. Addison-Wesley, Reading (1984)

Dragon, T., Woolf, B.P., Marshall, D., Murray, T.: Coaching Within a Domain Independent Inquiry Environment. In: Ikeda, M., Ashley, K.D., Chan, T.-W. (eds.) ITS 2006. LNCS, vol. 4053, pp. 144–153. Springer, Heidelberg (2006)

Easterday, M.W.: Policy World: A cognitive game for teaching deliberation. In: Pinkwart, N., McLaren, B. (eds.) Educational technologies for teaching argumentation skills. Bentham Science Publ., Oak Park (in press)

Easterday, M.W., Aleven, V., Scheines, R.: The logic of Babel: Causal reasoning from conflicting sources. In: Proc. ITS for Ill-Defined Domains Workshop. AED 2007, Los Angeles, USA (2007)

Fournier-Viger, P., Nkambou, R., Mayers, A.: Evaluating Spatial Representations and Skills in a Simulator-Based Tutoring System. IEEE Trans. Learn. Technol. 1(1), 63–74 (2008)

Fournier-Viger, P., Nkambou, R., Mephu Nguifo, E.: Exploiting Partial Problem Spaces Learned from Users' Interactions to Provide Key Tutoring Services in Procedural and Ill-Defined Domains. In: Proc. AIED 2009, pp. 383–390. IOS Press, Amsterdam (2009)

Graesser, A., Wiemer-Hastings, P., Wiemer-Hastings, K., Harter, D., Person, N.: Using Latent Semantic Analysis to evaluate the contributions of students in AutoTutor. Interact. Learn, Environ. 8, 149–169 (2000)

Hodhod, R., Kudenko, D.: Interactive Narrative and Intelligent Tutoring for Ethics Domain. In: Proc. ITS for Ill-Defined Domains Workshop. ITS 2008, Montreal, Canada (2008)

Kabanza, F., Nkambou, R., Belghith, K.: Path-Planning for Autonomous Training on Robot Manipulators in Space. In: Proc. 19th Intern. Joint Conf. Artif. Intell., pp. 1729–1731 (2005)

Kodaganallur, V., Weitz, R., Rosenthal, D.: An Assessment of Constraint-Based Tutors: A Response to Mitrovic and Ohlsson's Critique of "A Comparison of Model-Tracing and Constraint-Based Intelligent Tutoring Paradigms. Intern. J. Artif. Intell. Educ. 16, 291–321 (2006)

Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A.: Intelligent Tutoring goes to school in the big city. Intern. J. Artif. Intell. Educ. 8, 30–43 (1997)

Lynch, C., Ashley, K., Aleven, V., Pinkwart, N.: Defining Ill-Defined Domains; A literature survey. In: Proc. Intelligent Tutoring Systems for Ill-Defined Domains Workshop, ITS 2006, pp. 1–10 (2006)

Mitrovic, A., Weerasinghe, A.: Revisiting Ill-Definedness and the Consequences for ITSs. In: Proc. AIED 2009, pp. 375–382 (2009)

Mitrovic, A., Martin, B., Suraweera, P.: Intelligent Tutors for All: The Constraint-Based Approach. IEEE Intell. Syst. 22, 38–45 (2007)

Moritz, S., Blank, G.: Generating and Evaluating Object-Oriented Design for Instructors and Novice Students. In.: Proc. ITS for Ill-Defined Domains Workshop, ITS 2008, Montreal, Canada (2008)

Ogan, A., Wylie, R., Walker, E.: The challenges in adapting traditional techniques for modeling student behavior in ill-defined domains. In: Proc. Intelligent Tutoring Systems for Ill-Defined Domains Workshop (2006)

Simon, H.A.: Information-processing theory of human problem solving. In: Estes, W.K. (ed.) Handbook of learning and cognitive processes (5). Human information

Soller, A., Martinez-Monez, A., Jermann, P., Muehlenbrock, M.: From mirroring to guiding: A review of state of the art technology for supporting collaborative learning. Intern. J. Artif. Intell. Educ. 15(4), 261–290 (2005)

Walker, E., Ogan, A., Aleven, V., Jones, C.: Two Approaches for Providing Adaptive Support for Discussion in an Ill-Defined Domain. In: Proc. ITS for Ill-Defined Domains Workshop, ITS 2008, Montreal, Canada (2008)

Woolf, B.P.: Building Intelligent Interactive Tutors. Student Centered Strategies for revolutionizing e-learning. Morgan Kaufmann Publ., Mass (2009)