1st UK Workshop on Case-Based Reasoning, 1995

Self-Questioning and Experimentation: An Index Vocabulary of Situated Interaction

Ruediger Oehlmann Pete Edwards Derek Sleeman

Department of Computing Science
University of Aberdeen
King's College
ABERDEEN, AB9 2UE,
Scotland, UK.
email: {oehlmann, pedwards, sleeman}@csd.abdn.ac.uk

Abstract

Various cognitive and computational models have addressed the use of previous experience to understand a new domain. In particular, research in case-based reasoning has explored the ideas of retrieving and adapting previous experience in the form of cases. If the cases take the form of plans, the process is referred to as case-based planning. We have developed a computational model of Exploratory Discovery which integrates case-based reasoning and case-based planning. Motivated by results from cognitive science, we incorporated in this model the features of exploration and self-questioning to improve the case-based learning process. This paper will focus on the index vocabulary needed to accomplish the interaction between the case-based generation of self-questions and the case-based experimentation process. The interaction depends on questions asked, answers given, and previously performed experiments. Interaction is therefore situated.

1. Continuous Case-Based Reasoning

When people are exposed to an unfamiliar task such as understanding a new device, they are sometimes reminded of previous experience in other domains. Cognitive models of reminding have evolved in an attempt to explain this phenomenon (e.g., Ross 1989). The concept of re-use of previous experience has been exploited in the paradigm of case-based reasoning where previous experience is usually represented as cases (see Kolodner 1993 for an overview). There is a strong mutual relationship between cognitive models of reminding and case-based reasoning. Perhaps the most crucial issues in case-based reasoning are the retrieval and modification of previous cases, both of which use indexes which describe the content of a case and reflect the current understanding of the reasoner. Often a case contains the description of a problem and its solution. If, in contrast, the case is a plan which contains a sequence of actions, we refer to the process as case-based planning (Hammond 1989).

To date, case-based reasoning and case-based planning have largely been treated as separate research issues. Case-based reasoners do not typically interact with the environment and case-based planners do not typically interact with a case library reflecting a (partial) state of the environment. Furthermore, case-based reasoning and case-based planning have typically been implemented as “one-step” processes: a previous case is retrieved and adapted to the current situation, then the process stops. A solution generated as a result of a case-based reasoning process is not used to initiate a new case-based reasoning process. In contrast, the human reasoning process has been viewed as a continuous “stream of thoughts” (James 1981). When we follow this view, we have to regard generation of questions or experiments as being dependent on generations of previous questions and experiments; i.e. the process of generating a question is a situated interaction with previous episodes of question and experiment generation.

The idea of incorporating plans to facilitate the learning process has been supported by results from cognitive science. Schauble et al. (1991) and Sternberg (1981) suggested that learning is improved if the learner makes use of plans. Additional psychological experiments indicate that human learning can be further improved by the use of self-questions, i.e. questions which are generated and answered by the learner. We have demonstrated elsewhere that self-questions can be used to change the direction of a case-based reasoning process and to avoid reasoning failures (Oehlmann, Edwards, & Sleeman 1994). Therefore question-based reasoning strategies facilitate continuous case-based reasoning.

Most of the work on self-questioning investigates how self-questions can be used to improve reading abilities (see the review of Wong 1985). The influential results of Wong and Jones (1982) demonstrated that the reading comprehension of learning disabled students can be improved by self-questioning. Miyake and Norman (1979) pointed out that the generation of self-questions is constrained. One of these constraints is content knowledge. If prior knowledge is available, it enables the student to formulate a question on a given topic and to evaluate the answer. If, on the other hand, the relevant knowledge is missing, the student's ability to ask questions is restricted. Unfortunately, work which addresses issues of self-questioning does not investigate how question generation is changed by a surprising answer or a situation in which a question cannot be answered. Indirect evidence, however, is given by experiments reported by Schauble et al. (1991). Their subjects had to discover laws, such as Ohm's and Kirchhoff's laws, by self-directed experimentation in a computer circuit laboratory. The results demonstrated that good learners are characterised by two exploratory abilities; they perform more controlled experiments and they rarely use characteristic paths through the space of possible circuits. Although their subjects did not use explicit question strategies, they applied implicit strategies by choosing circuits and measuring voltage.

The psychological results motivate our approach to continuous case-based reasoning based on planning of self-questions and experiments. This approach is referred to as Exploratory Discovery. Whereas cases in the standard case-based reasoning approach describe states, plans are descriptions of processes. An agent has to maintain states as well as processes; therefore case-based reasoning and case-based planning should be integrated. A reasoner should be able to execute plans which modify the environment. Such modifications should be represented in memory as new cases which, in turn, should influence future planning. The planning of self-questions should lead to the planning of experiments and the experimental results should lead to new questions. Therefore interaction between case-based question planning and experimentation planning should sustain a continuous process of reminding.

To use case-based plans, they have to be stored under suitable indexes. It has been argued that a plan index should reflect the content of a given plan (Schank & Osgood 1990). The content of the question, answer, and experimentation plans used in our Exploratory Discovery approach is related to the interaction between self-questioning and experimentation. If we wish to capture the content of this interaction, we have to ask what it means to be able to ask a particular question or to perform a particular experiment. Moreover, we have to ask what it means when a particular piece of information is missing. Answering these questions will lead to a content-theory of interactions between question planning and experimentation planning. Similar content theories which provide appropriate index vocabularies have been developed for story indexing (Schank & Osgood 1990), multi-agent planning, (Goldweic & Hammond 1992), and indexing social situations (Domeshek 1992).

We have investigated the process of Exploratory Discovery in our system IULIAN. In Section 2, we will describe an overview of the system and its data structures. The overview will be followed by an example of continuous reminding based on self-questions and experiments (Section 3). In Section 4, we will describe the index vocabulary which describes the interaction between question, answer, and experimentation planning. Finally in Section 5, we will discuss the vocabulary.

2. The IULIAN System - A Top Level View

In this section, we describe a top-level view of our implementation IULIAN. Figure 1 shows the main modules of the system: question planner, answer planner, experiment planner, hypothesis formation, and question strategy planner. The system attempts to generate an explanation suitable to revise an initial theory. The Question Planner Module accepts a problem description as input, generates a question about the problem, and transfers control to the Answer Planner. If a question cannot be answered, the Question Planner and the Experiment Planner can be used to generate additional questions and experiments which helps the IULIAN system recover from this situation and to provide the knowledge needed to generate the answer. Before an experiment is performed, the Hypothesis Formation module hypothesizes the experimental result. When the actual result is generated, the Hypothesis Formation module determines, when appropriate, an expectation failure as the difference between the hypothesis and the actual result.

If an expectation failure has been detected, the exploration process is initiated. At its simplest, the processes of question and answer generation are based on the Question Planner and the Answer Planner which generate a question about the problem and attempt to answer it. The answer returned should be wrong because if the correct information were known, IULIAN would have generated the correct hypothesis and the expectation failure would not have occurred. If a question cannot be answered, the Question Planner and the Experiment Planner can again be used to generate additional questions and experiments in an attempt to provide the missing knowledge.

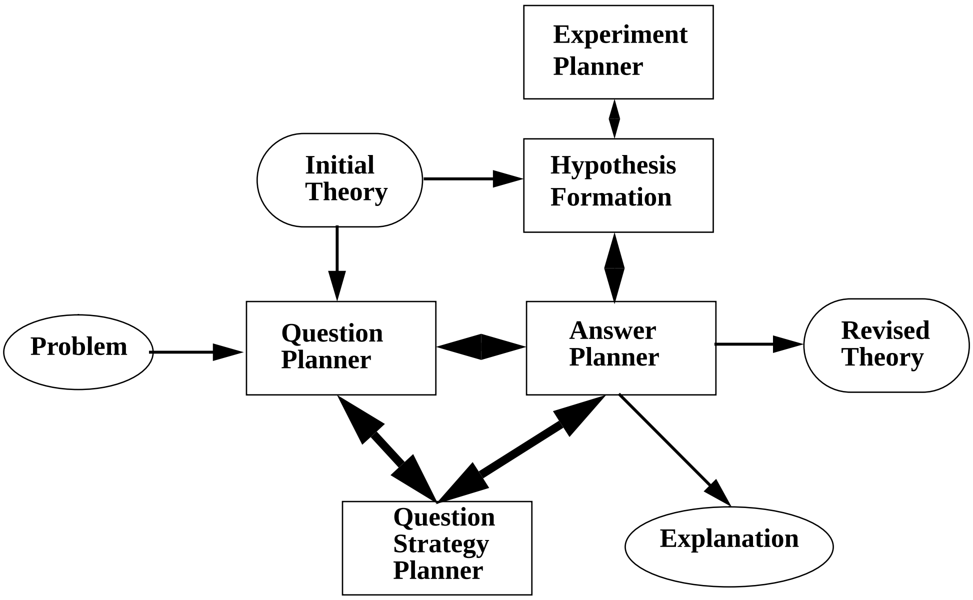

Figure 1: The IULIAN System

All the plans needed to generate questions, answers, and experiments contain indexes which are used to retrieve an appropriate plan in a given situation. The indexes are the main mechanism to control the generation of sequences of questions and answers. In addition, the Question Planner can be used to plan the generation of sequences of questions and answers.

The basic data components of the IULIAN system are experiments, explanations, models, and plans which are used as cases. An experiment consists of two components: an experimental setting (e.g. a description of an electric circuit with battery, lamp, and switch) and the result of an experiment such as the statement that “the lamp is on when the battery is switched on”. Experiments are represented by objects and relations between objects. Objects are represented as Memory Units (MU)[1] which contain an object frame, a context frame and a content frame (Figure 2). The context frame describes the context in which the object occurs represented by a set of relations. The content frame comprises several sets of intentional descriptor values referred to as views. The object frame comprises general information about the object. In addition to experiments, causal models are used to explain experimental results. Causal models have a similar representation to experiments. However, they are stored in a separate library and their objects are viewed as abstract concepts, e.g. the concept “lamp” rather than an actual lamp used in a given experiment (Oehlmann 1992). In addition, causal models use particular relations between concepts to represent causal links. A causal model is linked to the experiment used to generate that model. Question plans are used to apply case-based planning techniques to the generation of single questions. The elementary actions which the planner executes combine question substructures to form a complete question. For example, the question “What does the PIN-WHEEL turn?” can be built by combining the substructures “What “, “does ”, “turn ”, “the ”, and OBJECT1. OBJECT1 is a variable which can be instantiated with the string “PIN-WHEEL ”. A question plan has two main parts: the set of descriptors used for indexing the plan and a sequence of steps, see Figure 3.

The plan is retrieved by matching its index with the current situation; this is characterised by the goals the system pursues in asking the question. Additional slots in the head of each question

[1] MUs are similar to the Universal Index Frame (Schank & Osgood, 1990).

plan contain a list of variable instantiations referred to as bindings and a set of collector slots. The bindings are used to instantiate variables in the step actions. In the collector slots, intermediate results are stored during the question formation process. Each planning step has precondition, goal, and action slots to ensure correct plan execution. If plan execution fails, the usual explanation-based repair mechanisms are employed, see Hammond, Marks, & Converse (1993). These mechanisms use pre-stored repair rules and are based on preconditions and goals.

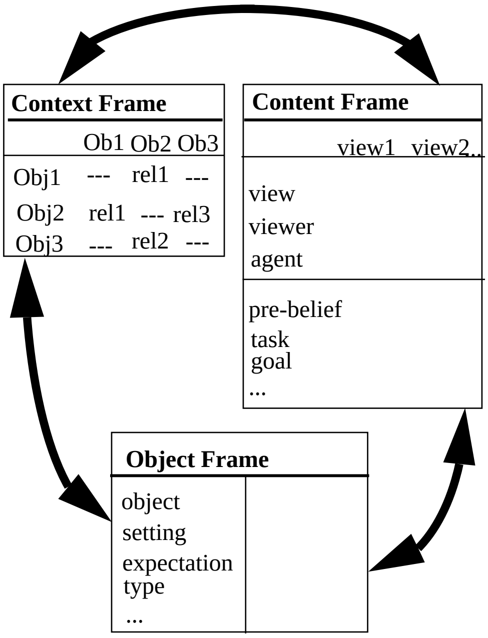

Figure 2: Memory Unit

```
(plan: name identify-effect
   :collector “What does the PIN-WHEEL turn?”
   :collector1
   :collector2
   :collector3
   :bindings ((focus-object1 pin-wheel))
   :question-goals (identify-motion-effect@neighbouring-object)
   :question-strategy-goals (consequence-checking)
   :recovery-goals
   :question-failures
   :expectation-failure
   :experimentation-goal (enable-movement@focus-object
   observe-movement@neighbouring-object)
   :failures
   :steps
      :s-name add-question-word
         :action (add-substructure “What ” collector1)
      :s-name add-question-mark
         :action (add-end-substructure “?” collector1)
      :s-name add-auxillary-verb
         :action (add-substructure “does ” collector2)
      :s-name add-verb
         :action (add-end-substructure “turn ” collector2)
      :s-name transfer-predicate
         :action (transfer-before collector2 “?” collector1)
      :s-name add-article
         :action (add-substructure “the ” collector2)
      :s-name add-noun
         :action (add-end FOCUS-OBJECT1 collector2)
      :s-name transfer-and-insert-subject
         :action (transfer-insert collector2 “turn ” collector1)
```

Figure 3: Question Plan

It is an important advantage of the case-based planning approach that new questions can be learned by modifying previous question plans. Answers are generated in a similar way; however, steps in answer plans may have particular actions which retrieve knowledge from the case-base needed to form an answer. Question strategies are higher level plans which organise the execution of single question and answer plans to generate questions in a particular sequence. The same basic plan structure used for question and answer plans has been employed for experimentation plans, although the index vocabulary differs (Oehlmann, Sleeman, Edwards, 1993). Experimentation plans describe the steps which have to be executed to perform an experiment. The experimental setting and the result of plan execution are stored as a new case

3. An Example of Situated Interactions

Before we describe the details of the index vocabulary, we will discuss the core of an example which illustrates question and experiment generation (Figure 4). The entire example and the questions used during the reasoning process are described in (Oehlmann, Sleeman, & Edwards 1992).

We assume that the IULIAN system receives as input the description of a electric circuit with lamp and closed switch in parallel (target domain). Associated with the description of the circuit (target case) is an experimental (target) plan to build the circuit by connecting the various components and to observe the status of the lamp. The lamp is reported as being off, as a result of this experiment. This result is inconsistent with the system's expectation based on a previous case involving a serial circuit in which the lamp was on. The situation characterised by this expectation failure (partially) matches the index of a question plan for generating a *why* question which would identify an explanation for the expectation failure. IULIAN is able to retrieve the question plan but not an appropriate answer plan, because the explanation has not been stored. This new situation determines an index for retrieving the question strategy ANALOGICAL MAPPING which supports analogical case retrieval between domains. Executing the first step of the question strategy initialises the retrieval and execution of a question plan to generate an additional top level question. The answer plan to this question can be executed and the generated answer comprises a (source) case in the domain of water pipes (source domain). The interplay between the execution of question and answer plans has actually lead to case-retrieval (reminding) across domain boundaries (Oehlmann, Edwards, & Sleeman 1995).

In the source case a plain pipe and a paddle wheel are in parallel. Additionally the observation that the paddle wheel does not turn is stored in the source case, which is associated with both a causal model and the source plan. A summary of the causal model is given below:

I*n order for the paddle wheel to turn, water must flow over it. There is a plain pipe in parallel with the paddle wheel. The smaller the resistance in a given pipe, the greater is the water flow in this pipe. If one of two parallel pipes has a very low resistance and the other one has a very high resistance, most of the water flows through the pipe with low resistance. Since the paddle wheel offers resistance to the flow and the plain pipe does not, all of the water flow goes through the plain pipe. Since there is no water flow over the paddle wheel, the paddle wheel does not move.*

The source model can not be applied to the original electric circuit, because the switch and the plain pipe are not sufficiently similar. Therefore, the source plan is transformed into a new source plan able to generate a new source case which is sufficiently similar to the target case (Oehlmann, Sleeman, & Edwards 1993)

During plan transformation, IULIAN replaces the step which refers to the insertion of a plain pipe by a step which refers to the insertion of a valve. The valve is more similar to the switch in the target domain, because both components are used to pursue the goal “select:flow-interruption/ flow-support”. In addition, the system inserts a step which opens the valve.

Once the transformation process is finished, two additional reasoning stages are needed. First, the reasoner has to collect evidence that the causal model associated with the source case is valid for the transformed source case. Second, the reasoner has to modify the causal model to make it applicable to the target case, and it then has to ensure that the modified causal model is valid for this case.

The first stage can be achieved by modifying the transformed plan (Hammond 1989). This stage has the following effect: in executing the modified plan the water pipe circuit is built by connecting the various components and a small test paddle wheel is placed after the valve. The test paddle wheel allows the planner to test whether the water runs through the valve.

In the second stage, the table of similar objects enables the system to replace all objects from the water pipe domain which appear in the causal model with similar objects from the electric circuit domain.

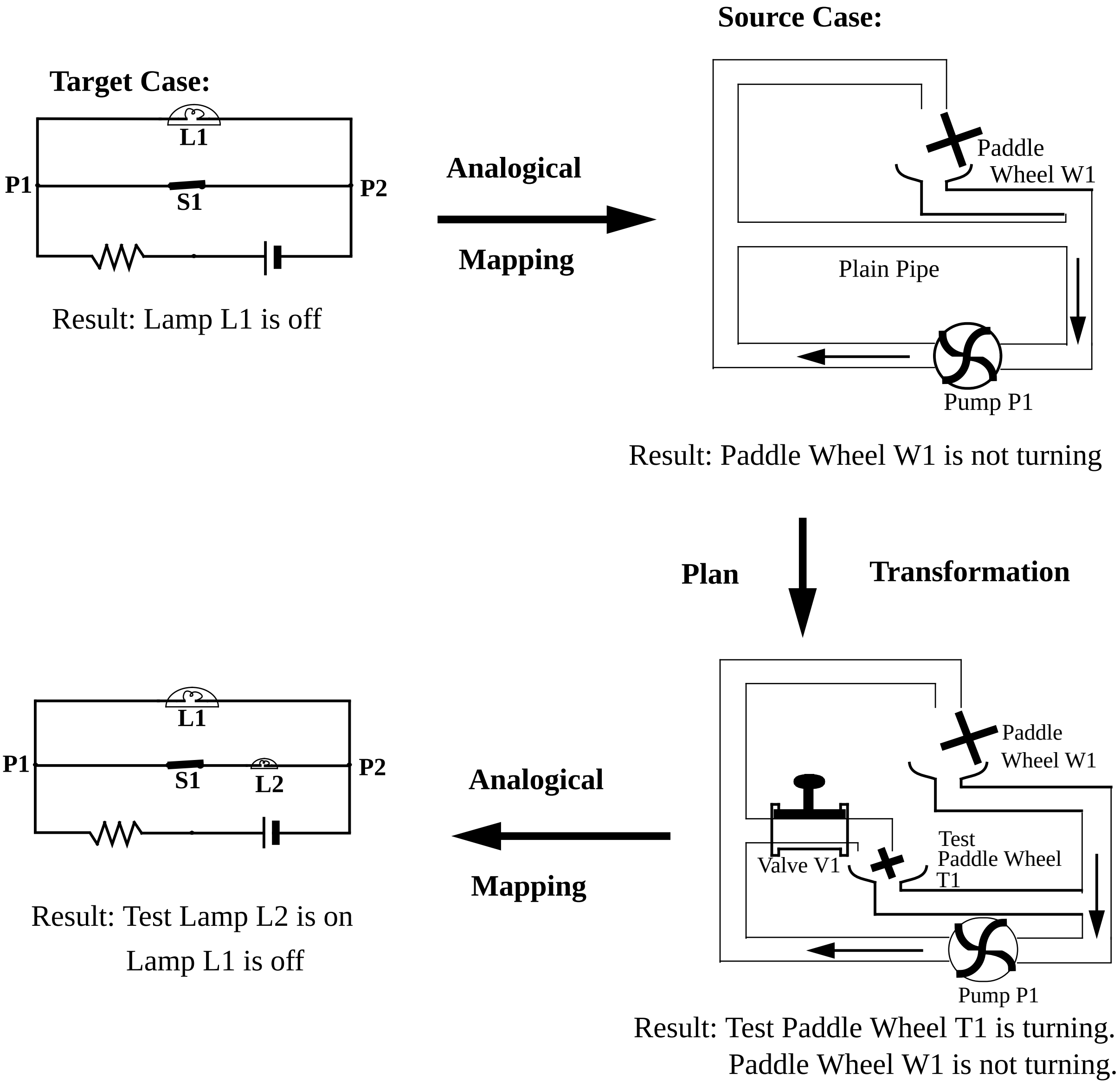

Figure 4: The Electric Circuit Example

The validity of the modified causal model can be tested by modifying the target plan in the same way as the transformed source plan. A step is added to the plan, allowing the planner to insert a test lamp after the switch in the electric circuit in order to test whether the current flows through the path with the switch. The result of plan execution shows that the current flows through the switch rather than through the main lamp and confirms the new causal model.

4. A Vocabulary of Situated Interactions

If we wish to capture the content of interactions between questions and experiments, we have to ask what it means to be able to pose a particular question or to perform a particular experiment. Moreover, we have to ask what it means when a particular piece of information is missing. Questions are often perceived as mechanisms to request information. However, our discussion of self-questioning in Section 1 revealed that questions can be used to guide the reasoning process. Similarly, experiments may have different meanings. An experiment can be performed to confirm or reject a hypothesis. Alternatively, an experiment can be performed with a more inquisitive attitude. The IULIAN system does not know how the environment will behave under a given set of conditions. Therefore, an experiment is performed to investigate it. Often, experiments and questions can be described with a set of descriptors similar to the vocabulary used for story indexing. This means, we can describe a question or an experiment in terms of the goals the system pursues, the plans and tasks to be performed, and the beliefs the system maintains (Oehlmann, Sleeman, & Edwards 1993). However, the goal pursued in asking a self-question can be different from the goal an agent in a story pursues in interacting with other agents. This is particularly true when the question is asked because a previous question could not be answered. In such a situation which we refer to as a *question failure*, a second question may serve as a recovery tool. Also experiments can be used to recover from a *question failure*.

In this Section, we will discuss the issue of question failures in two stages. First, we give an example of a question failure and analyse this failure; we then derive from this analysis a set of index descriptors suitable to retrieve additional question plans, answer plans, and question strategies; all these plans can be used to overcome the failure situation. The index vocabulary developed in these sections will cover various situations in which questions can be combined to form question sequences.

The example in Section3 involved replacing the plain pipe in the water pipe experiment with a valve (Figure 4). However, the system did not know if this replacement was adequate, i.e. if there would be water flow through the open valve. Therefore the following question was generated:

Q1.5: Is there water flow through the pipe with the open valve?

This question could not be answered, because the valve had not been used in a parallel water pipe circuit before. An additional question was needed to focus on the issue of testing the existence of a water flow:

Q1.5.1: What if we insert a small test paddle wheel behind the open valve? Is the test paddle wheel turning?

This question finally led to an experiment which revealed that there was a water flow through the valve. This experimental result allowed the system to generate an answer to Question 1.5.

Hypothesis: The test paddle wheel, T1, is turning.

Experiment: Run an experiment with a test paddle wheel, T1, behind the open valve, V1.

Experimental result: The test paddle wheel, T1, is turning and the main paddle wheel, W1, is not turning.

A1.5: There is water flow through the valve, V1.

Obviously, the main reason for generating a question is to obtain an answer. If an answer cannot be generated because the necessary knowledge is not available, the question fails and the system has to recover from the failure situation. Recovering from a question failure involves the retrieval of either an appropriate single question or an entire question strategy.

In order to support the retrieval of self-questions, an appropriate indexing vocabulary has to be provided. We obtain such a vocabulary by investigating explanations for the failure situation related to Question 1.5 and the recovery strategy related to Question 1.5.1.

1. Why did Question 1.5 fail? It failed because:

a) The question goal the system pursued by asking Question 1.5 can be described as checking if an entity exists which is not part of the system's experience. The entity *water flow* is not known because the system's experience is given in terms of experimental settings such as a system of water pipes, a paddle-wheel, an open valve and a pump and in terms of experimental results such as *paddle-wheel has state turning*. This question goal is not satisfied, because the answer cannot be given.

b) The system is unable to derive the answer to Question 1.5 directly from its causal knowledge.

2. Why has Question 1.5.1 to be asked? It was asked because:

a) The system needs to recover from the failure which occurred when Question 1.5 was not satisfactorily answered.

b) As we stated above, the system has to know about the existence of an entity which is neither part of its experience nor derivable from its causal knowledge. However, if the system has previously generated an explanation, such as *a small test paddle-wheel is turning, because there is water flow through the test paddle-wheel*, then it can match the entity *water flow* in Question 1.5 with the same entity in the explanation. This allows the system to focus on a new entity *paddle-wheel* which can be integrated into Question 1.5.1.

c) In applying a question strategy, the system may attempt to identify an analogy between electric circuits and water pipe circuits (see the example in Section 3). This strategy would not succeed if it were not possible to decide whether the entity *water flow* exists.

d) In Section 2, the system needs to explain the incorrect expectation that the lamp is on although the experiment revealed that the lamp is off. The existence of the water flow, and even more the existence of its analogue the electric current, would support a causal chain between experimental result and experimental setting.

An Indexing Schema for Question Plans

From our discussion in the previous section, we can derive several requirements for indexing question plans. Considering the explanations above, we note that generalisations of these explanations hold for a large number of questions. On the other hand, we expect the details of other question failures to be sufficiently different to allow the system to distinguish between questions and to retrieve the appropriate question.

In addition, we indicated that a question can have several functions. In general, questions have to support the acquisition of additional knowledge, to represent a query about knowledge stored in memory, and to support recovery from situations in which a previous question could not be answered. These requirements are reflected in the indexes of question plans. In the remainder of this section, we will list the descriptors which are used to index questions plans. In addition, we will give short descriptions of the functions these descriptors fulfill.

question-goals: Questions are retrieved by the goals the system attempts to satisfy in asking a given question.

reasoning-goals: In addition to the very concrete **question-goal**, the system might pursue a more general **reasoning-goal**. For instance, the system attempts to identify the state of a paddle wheel under given conditions as part of a general process of analogical mapping. Whereas the **question-goal** focuses on identifying a state, the **reasoning-goal** addresses the entire strategy of analogical mapping.

recovery-goal: The basic strategy to recover from a question failure is the generation of an additional question strategy, an additional single question, and possibly an additional experiment. Whereas a **question-goal** focuses on a possible answer or a possible experiment, the **recovery-**

goal focuses on the question failure and on a means to recover from this failure.

k-resource-need: A possible reason for asking a question is the need for a particular piece of knowledge referred to as a **k-resource-need**[2]. This need can be satisfied by a **k-resource-gain** generated by an answer plan.

k-resource-assumption: Asking a particular question might make sense only under particular circumstances. For example, it is unreasonable to ask why a given lamp is off, when no lamp is used in an experiment. Therefore, the descriptor **k-resource-assumption** specifies assumptions made when asking a particular question.

An Indexing Schema for Answer Plans

Answer plans are indexed by the goals the system pursues in executing the plan and by the planning failures the system attempts to avoid.

answer-goals: Answer goals are used to retrieve an appropriate answer plan.

failures: An answer plan fails when a planning step cannot be executed.

k-resource-gain: If an answer is generated, a particular piece of knowledge is provided. This situation is described by the **k-resource-gain** descriptor which interacts with question plans in two ways. The **k-resource-gain** might satisfy a **k-resource-need**, i.e. the answer was successful, or the **k-resource-gain** might match a **k-resource-assumption**, i.e. it makes sense to generate the question whose plans contains this assumption.

An Indexing Schema for Experimentation Plans

experimentation-goals: In performing an experiment the system pursues various types of goals. In the previous sections we have discussed those goal types which are related to the reasoning process. In addition, the system pursues goals which are directly related to actions, such as

[2] *k-resource* is an abbreviation for *knowledge-resource.*

enabling the current flow. These goals are referred to as **experimentation-goal**s.

activity-goals: This goal type has a function similar to that of **reasoning-goal**s in question plans. In addition to the more concrete **experimentation-goal**, the system might pursue a more general **activity-goal**. For example, the objective of modifying an experiment in order to obtain a new experiment can be expressed by an **activity-goal**. During the modification process, an additional battery could be added to an electric circuit to increase the current. This objective establishes an **experimentation-goal** which is more specific than the general **activity-goal**.

6. Discussion

We have described our Exploratory Discovery system IULIAN which is based on an integration of case-based reasoning and case-based planning. Furthermore, we have argued that our approach of planning the generation of questions and experiments requires a content oriented index vocabulary addressing the interaction between self-questioning and experimentation. This vocabulary enables the system to perform case-based reasoning as a continuous process.

The interactions between question planning and experimentation planning have to address two tasks: filling a knowledge gap and supporting a reasoning or experimentation strategy. For example, Question 1.5 in Section 4 was asked because the reasoner did not know whether there was water flow through the open valve. Therefore the question leads to an experiment; i.e. the index vocabulary has to provide a link between a question which cannot be answered and an experiment.

In contrast, the first question which was asked about the plain pipe experiment in Section 3 was generated to initiate a reasoning strategy. The strategy used in the example is the analogical mapping strategy which belongs to the strategy type c*hanging the focus*. Strategies which change the focus of the reasoning process involve questions about different issues.

The task of filling a knowledge gap involves three sub-tasks: noting that a knowledge gap exists, judging that a question addresses a knowledge gap, and noting that a knowledge gap has been filled. These sub-tasks are addressed by the index descriptor **k-resource-need**, **k-resource-gain**, and **k-resource-assumption**.

Supporting a reasoning strategy is addressed at a local and a global level. The local level guides the generation of the next question, answer, or experiment. The descriptors used at this level are **question-goal**, **answer-goal**, and **experimentation-goal**. The global level guides the generation of sequences of questions, answers, and experiments which realise entire reasoning strategies such as *changing the focus*. The descriptors **reasoning-goal** and **activity-goal** are used at the global level.

In contrast to the tasks discussed so far, the task of recovering from a question failure involves descriptors related to knowledge gaps as well as descriptors related to strategies. For example, the strategy *changing the focus* is used to address a question failure; therefore this strategy is guided by the descriptors **recovery-goal** and **k-resource-gain**. Using the descriptor **recovery-goal** initiates a new strategy, whereas the descriptor **k-resource-gain** indicates when the strategy succeeds and the knowledge gap is filled. The index vocabulary discussed above supports the generation of single questions, answers, and experiments because it enables the system to retrieve the appropriate question, answer, or experimentation plans. Furthermore, the vocabulary supports the retrieval of question strategies and therefore the planned generation of sequences of questions and answers.

References

Domeshek, E. (1992). Do the Right Thing: A Component Theory for Indexing Stories as Social Advice. Ph.D. Thesis, Yale University, New Haven, CT.

Goldweic, P. & Hammond. K. (1992). Multiagent Interactions: A Vocabulary of Engagement. Proceedings of the 14th Annual Conference of the Cognitive Science Society, (pp. 18 - 23).

Hammond, K. (1989). *Case-Based Planning: Viewing Planning as a Memory Task*. New York:

Academic Press.

Hammond, K., Converse, T., & Marks, M. (1993). Toward a Theory of Agency. In: S. Minton (ed.). Machine Learning Methods for Planning (pp. 351-396). San Mateo, CA: Morgan Kaufmann.

James, W. (1981). The Principles of Psychology, 1, Cambridge, MA: Harvard University Press.

Kolodner, J. (1993). *Case-Based Reasoning*. San Mateo, CA: Morgan Kaufmann.

Miyake, N. & Norman, D. (1979) To Ask a Question, One Must Know Enough to Know What is Not Known, *Journal of Verbal Learning and Verbal Behaviour*, 18:357-364.

Oehlmann, R. (1992). Learning Causal Models by Self-Questioning and Experimentation. Proceedings of the AAAI-92 Workshop on Communicating Scientific and Technical Knowledge. (pp. 73-80).

Oehlmann, R., Edwards, P., & Sleeman, D. (1994). Changing the Viewpoint: Re-Indexing by Introspective Questioning. *Proceedings of the 16th Annual Conference of the Cognitive Science Society*, (pp. 675 - 680). Hillsdale, NJ: Lawrence Erlbaum Associates

Oehlmann, R., Edwards, P., & Sleeman, D. (1995). Introspection Planning: Representing Metacognitive Experience. Proceedings of the AAAI Spring Symposium on Representing Mental States and Mechanisms, Cambridge, MA: AAAI-Press, *accepted for publication*.

Oehlmann, R., Sleeman, D. & Edwards, P. (1992). Self-Question and Experimentation in an Exploratory Discovery System, *Proceedings of the ML-92 Workshop on Machine Discovery,* (pp. 41-50).

Oehlmann, R., Sleeman, D., & Edwards, P. (1993). Learning Plan Transformations from Self-Questions: A Memory-Based Approach. *Proceedings of the 11th National Conference on Artificial Intelligence*, (pp. 520-525). Cambridge, MA: AAAI-Press.

Ross, B. (1989). Remindings in Learning and Instruction. In S. Vosniadou & A. Ortony, Similarity and Analogical Reasoning. Cambridge, UK: Cambridge University Press.

Schank, R. & Osgood, R. (1990). A Content Theory of Memory Indexing. Tech. Report 2, The Institute of the Learning Science, Northwestern University.

Schauble, L., Glaser, R., Raghavan, K., & Reiner, M. (1991). Causal Models and Experimentation Strategies in Scientific Reasoning. The Journal of the Learning Sciences, 1:201-238.

Sternberg, R. (1981). Intelligence and Nontrenchment. Journal of Educational Psychology, 73(1):1-16.

Wong, B. & Jones, W. (1982). Increasing Metacomprehension in Learning Disabled and Normally Achieving Students Through Self-Questioning Training. *Learning Disability Quarterly* 5, (pp. 228 - 240).

Wong, B. (1985). Self-Questioning Instructional Research. A Review. *Review of Educational Research, 55*, 227-268