Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Technical Memo
TM-93-01

Reconstructive Integrated Explanation of Lathe Production Plans

Otto Kühn, Andreas Birk

March 1993

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 20 80
D-6750 Kaiserslautern, FRG
Tel.: (+49 631) 205-3211/13
Fax: (+49 631) 205-3210

Stuhlsatzenhausweg 3
D-6600 Saarbrücken 11, FRG
Tel.: (+49 681) 302-5252
Fax: (+49 681) 302-5341

Deutsches Forschungszentrum für Künstliche Intelligenz

The German Research Center for Artificial Intelligence (Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI) with sites in Kaiserslautern and Saarbrücken is a non-profit organization which was founded in 1988. The shareholder companies are Atlas Elektronik, Daimler-Benz, Fraunhofer Gesellschaft, GMD, IBM, Insiders, Mannesmann-Kienzle, SEMA Group, and Siemens. Research projects conducted at the DFKI are funded by the German Ministry for Research and Technology, by the shareholder companies, or by other industrial contracts.

The DFKI conducts application-oriented basic research in the field of artificial intelligence and other related subfields of computer science. The overall goal is to construct systems with technical knowledge and common sense which - by using AI methods - implement a problem solution for a selected application area. Currently, there are the following research areas at the DFKI:

- ❑ Intelligent Engineering Systems
- ❑ Intelligent User Interfaces
- ❑ Computer Linguistics
- ❑ Programming Systems
- ❑ Deduction and Multiagent Systems
- ❑ Document Analysis and Office Automation.

The DFKI strives at making its research results available to the scientific community. There exist many contacts to domestic and foreign research institutions, both in academy and industry. The DFKI hosts technology transfer workshops for shareholders and other interested groups in order to inform about the current state of research.

From its beginning, the DFKI has provided an attractive working environment for AI researchers from Germany and from all over the world. The goal is to have a staff of about 100 researchers at the end of the building-up phase.

Friedrich J. Wendl
Director

Reconstructive Integrated Explanation of Lathe Production Plans

Otto Kühn, Andreas Birk

DFKI-TM-93-01

This work has been supported by a grant from The Federal Ministry for Research and Technology (FKZ ITW-8902 C4).

Contents

Reconstructive Integrated Explanation of Lathe Production Plans

Otto Kühn & Andreas Birk

DFKI, Postfach 2080, 6750 Kaiserslautern, Germany
Phone: +49-631-205-3463,
Fax: +49-631-205-3210
email: kuehn@dfki.uni-kl.de

Abstract:

A Reconstructive Explanation tool has been developed and implemented within an integrated knowledge acquisition framework. This tool (RIETO) employs both a formal and an informal knowledge base to construct explanations for individual lathe production plans. RIETO adopts a reconstructive explanation approach [WickThompson92] which does not rely on the problem solving trace constructed by the inference component of the system. Instead it reconstructs possible lines of reasoning which may provide justifications for those aspects of the solution which are questioned by the user. The explanation tool can thus bring to bear all pertinent information which was captured during knowledge acquisition, even if it was not used for actually solving the problem. RIETO can answer 'why?' and 'why not?' questions about different aspects of the production plan, can give justifications for rules and provide all information about a particular topic (e.g. the selection of cutting materials) which is pertinent to the context in which the question is asked by the user.

1. Motivation

Knowledge acquisition and knowledge utilization are mutually dependent and thus should be tightly coordinated throughout the life cycle of a knowledge-based system (KBS). On the one hand, the task to be solved by the KBS determines what knowledge has to be acquired, and an elaborate model of the problem solving (also termed interpretation model or model of expertise) can be used as guidance for knowledge acquisition [BreukerWielinga89]. On the other hand, the KBS should be able to bring to bear all the knowledge that was acquired and that is potentially relevant for the task at hand.

Even when a KBS has been designed to solve only one class of tasks, the knowledge in its knowledge base is accessed and utilized by the knowledge engineer, the domain expert and the end user who want to achieve different goals. The knowledge engineer together with the domain expert perform the knowledge acquisition and extension. Thereby they must access the existing knowledge base in order to determine what knowledge is present and how new knowledge should be integrated. The end user(s) utilize the knowledge base in order to obtain a solution for a particular problem. In addition, users will request an explanation of the suggested problem solution, since they will not trust blindly in the quality of the proposed solution. An explanation of a given problem solution may also be requested by the knowledge engineer and the domain expert during knowledge acquisition [Schmalhofer+91a]. Finally, a knowledge base is usually evolved by validation and exploration [Boley+92]. An overview of the different types of knowledge utilization and the involved users is given in Figure 1.

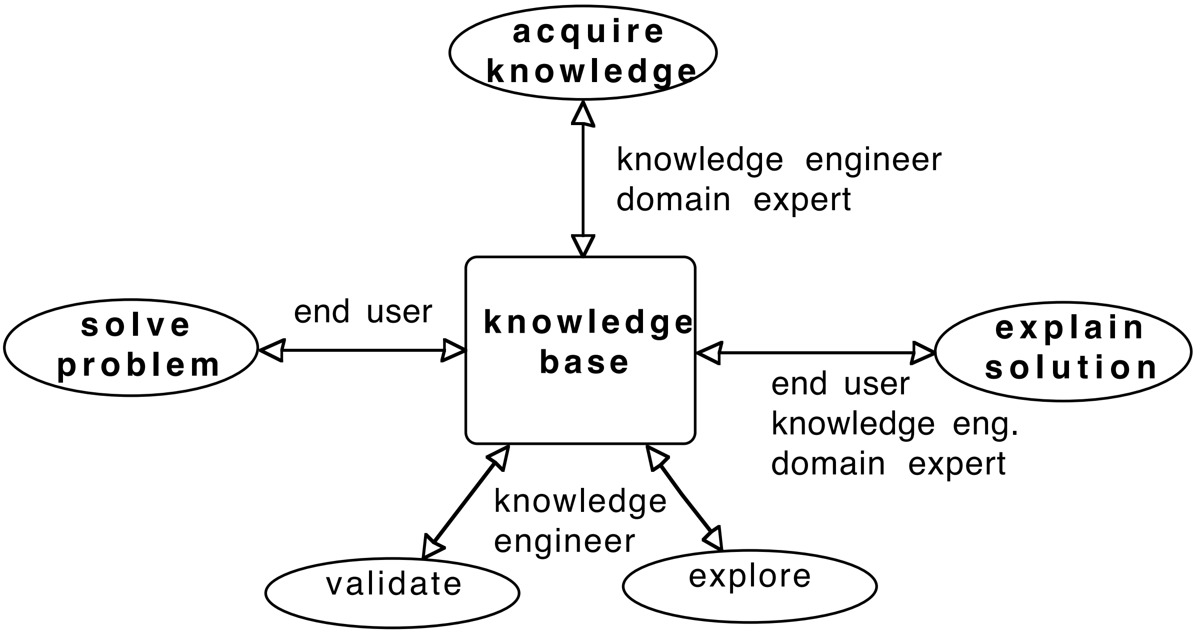

Fig. 1: Different types of knowledge utilization (Ellipses represent different activities, arrow labels indicate who usually performs the activity, the most typical user is listed first).

In the current paper, we will investigate the relationship between knowledge acquisition, problem solving, and the explanation of a problem solution which all operate on a shared knowledge base. We will present an explanation tool which is tightly integrated with the knowledge acquisition component and can thus bring to bear all knowledge which was collected during knowledge acquisition. The proposed explanation tool does not rely on a trace of the problem solving process by which the solution was derived. Instead it reconstructs possible lines of reasoning which may provide justifications for those aspects of the problem solution which are questioned by the user. The explanation tool can thus be employed to obtain justifications for problem solutions which may have been constructed by the user, by the expert, or by the problem solving component of the KBS.

The detachment of the explanation tool from the problem solving component and its close integration with the knowledge acquisition component allows for an independent verification of the proposed problem solution which is based on all knowledge which was captured during knowledge acquisition. Although the primary purpose of the explanation tool is to enhance the user's confidence in the proposed solution, it can also be employed by the knowledge engineer and the domain expert to explore and validate the knowledge base with respect to particular cases.

The remainder of the paper is structured as follows: in section 2, we will give a brief overview of the application domain for which the explanation tool was developed, the adopted knowledge acquisition method, and the structure of the resulting knowledge base. In section 3, we will describe the basic ideas of the reconstructive explanation approach and its relation to the problem-solving component. In section 4, the reconstructive integrated explanation tool will be described in detail and some sample interactions will be presented. Finally, we will discuss the advantages and possible drawbacks of the proposed explanation tool with respect to other work in related areas.

2. Integrated knowledge acquisition for lathe production planning

2.1. The problem

The problem of lathe production planning can be stated as follows:[1]

Given:

- A description of the geometry and the technology of the goal workpiece to be manufactured
- A description of the mold from which the goal workpiece is to be manufactured
- A description of the manufacturing environment, i.e. the lathe machine and the available tools
- A description of the manufacturing context such as lot size and production deadlines

Determine:

- The production plan, i.e. the sequence of chucking and cutting operations by which the goal workpiece can be efficiently manufactured from the mold in the given environment and context

A graphical representation of a lathe production planning problem and its solution is given in Figure 2. The top center of the figure shows the goal workpiece, a drive shaft, overlaid with the mold, a cylinder, from which it is to be manufactured. The black area on the left and the black triangle on the right indicate the chucking fixture by which the mold is held and rotated in the lathe. The numbers 1 through 7 above the mold refer to the sequence of cuts by which material is removed from the mold. The individual cuts of the production plan are specified in more detail in the remainder of Figure 2. For each cut, the cutting tool is specified together with the cutting path and the cutting parameters. For instance, the cutting tool number 1 with the specification "CSSNL 3232 C15 SNGN151016 TO 3030" is applied to remove a part of the outer layer of the cylinder with a cutting speed of v_c = 450 m / min, a cutting feed of f = 0.45 mm/rotation and a cutting depth of a_p = 5 mm.

[1] For a more detailed description see [KuehnSchmalhofer92].

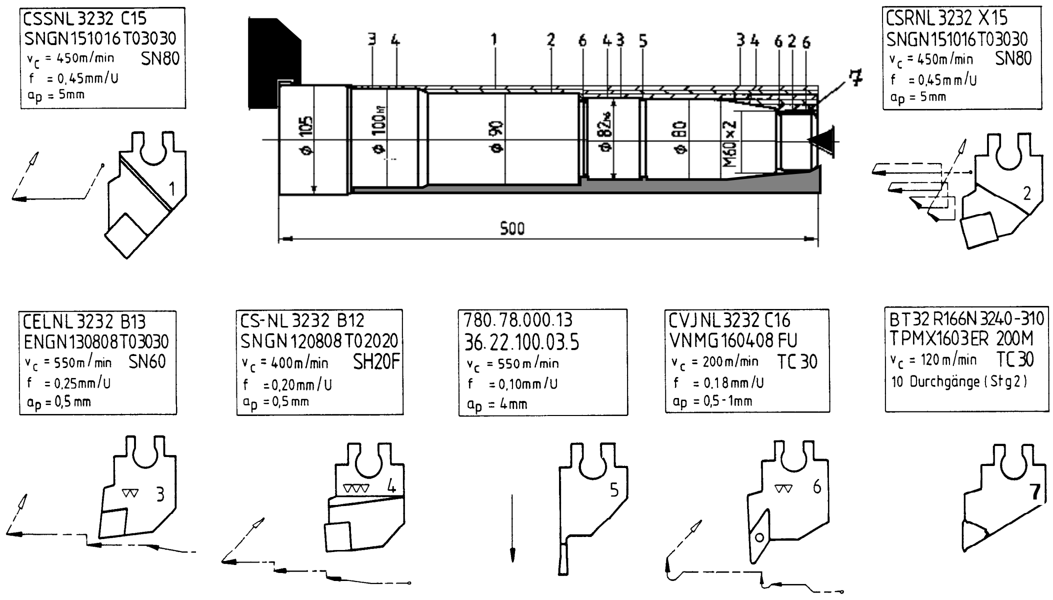

Fig. 2: A typical lathe production planning problem and its solution

When selecting a cutting tool and determining the cutting parameters, a large number of problem features must be taken into account, such as the workpiece geometry, the workpiece material, the cutting material, the surface roughness of the mold, the required precision of the goal workpiece, etc. A good production plan must not only manufacture the specified goal workpiece when executed on the lathe, but must do so at reasonable costs which depend on factors such as tool wear, production time, etc. Lathe production planning is a complex real-world problem which requires both theoretical knowledge (e.g. physical models of the cutting process) as well as practical experience (e.g. which parameter settings worked best in similar conditions). The relevant knowledge can be found in different sources of information which must be combined for the development of the KBS.

2.2. The knowledge acquisition procedure

In order to acquire the required knowledge, an integrated knowledge acquisition procedure was developed which conjointly employes three different sources of information: written documents, libraries of previously solved cases, and human experts. The integrated knowledge acquisition procedure has been previously described in [Schmalhofer+91a] and [Schmalhofer+91b]. We will review here only those aspects of the knowledge acquisition procedure which are relevant for the integration of the explanation tool.

Based on the KADS methodology [Wielinga+92], a model of expertise is first developed which specifies, how the problem is to be solved by the system and what knowledge is required. For lathe production planning, skeletal plan refinement was identified as the most adequate problem solving method

[KühnSchmalhofer92]. According to this method, three basic types of knowledge are required: abstraction rules which relate the concrete problem description to more abstract features, association rules which relate problem features to abstract, or skeletal plans, and refinement rules for the specification of a concrete plan from an abstract plan.

Even though the acquisition of the domain knowledge is guided by the selected problem solving method, the domain knowledge itself is to be represented purely declaratively. This means in particular that the representation of domain knowledge be independent of its use in the specific problem solving method. Whereas a feasibility of a strict separation of domain and problem solving knowledge can be questioned both on theoretical and practical grounds [KühnSchmidt92], it is nevertheless possible to represent the domain knowledge in such a way that it can be reused for several related tasks. Since solving a problem and explaining the problem solution are two related tasks which require overlapping although not identical knowledge, the domain knowledge should in particular be acquired and represented in such a way that it can be employed for both of these tasks.

In the developed integrated knowledge acquisition method, this is achieved by using explanation as a knowledge acquisition technique for combining different sources of information. An expert is asked to explain a prototypical case selected from a filing cabinet in terms of knowledge units extracted from a textbook and his own experiences. Besides offering the advantage of being more economical (the solution of a production planning problem is rather time consuming), this procedure guarantees that all the relevant knowledge both for solving a problem and for explaining the solution are made available during knowledge acquisition. The pieces of relevant knowledge are initially extracted from written documents. Together with the constructed explanations and the experts utterances they are captured and stored in an informal knowledge base. These knowledge units are then stepwise transformed into a formal representation [Schmidt92].

In this knowledge acquisition process, each knowledge unit is documented by different kinds of information which are readily available during knowledge acquisition. This documentation is not only essential for making later revisions or adaptations of the knowledge base, but it can also be exploited by the explanation tool in order to provide the users a better insight into the knowledge of the system, thereby increasing their confidence into the adequacy of the proposed solution.

2.3. The knowledge base

The described knowledge acquisition procedure yields a collection of formal and informal knowledge units which is structured by various links. The entire knowledge base thus constitutes a structured Hypertext. Whereas most elements of this hypertext are natural language descriptions which can be accessed through standard browsing and information retrieval techniques, some elements have a formal representation and can thus in addition be employed for deriving inferences.

Two different types of Hypertext elements or knowledge units can be distinguished on epistemological grounds: object descriptions and rules. An object description is a cluster of propositions which refer to one entity and are seen as being essentially dependent on that entity [GuarinoBoldrin92]. A rule is a proposition which would not naturally be considered as referring to one single entity. For instance, the proposition "For a rough cut the cutting feed is typically greater than 0.3 mm/rot." is readily seen as referring to the entity *rough cut* which is introduced as an object with an attribute *cutting feed*. The proposition "When rough cutting cast iron, a cutting-feed of 0.8 mm/rot should be applied." refers both to the objects rough cut and cast iron and is therefore stored as a rule.

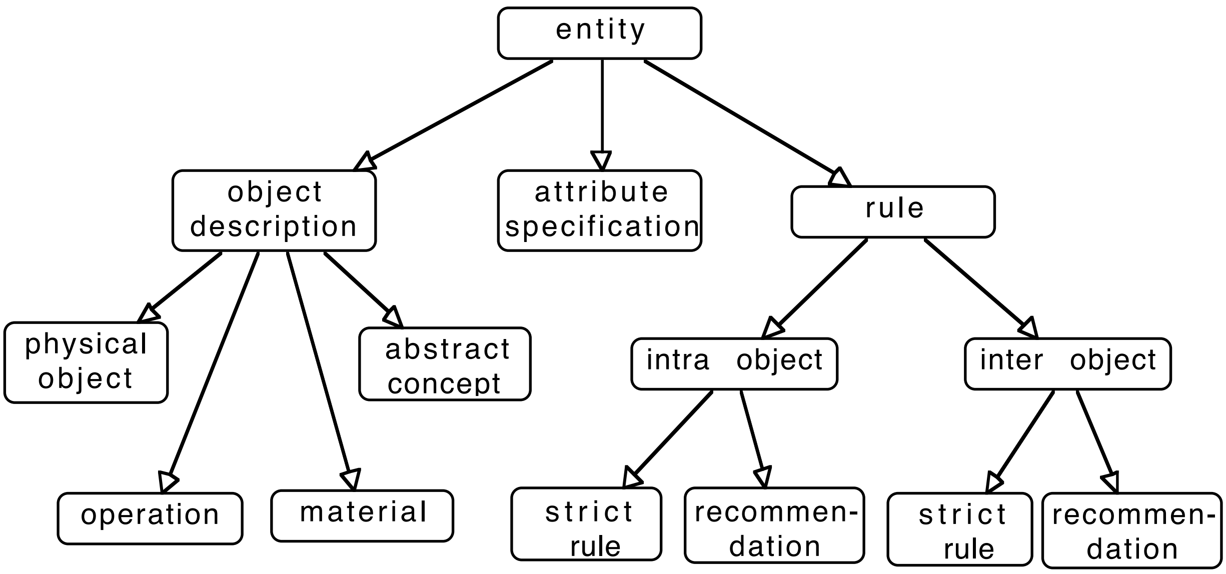

Fig 3: The root of the domain ontology indicating the basic types of knowledge that are distinguished in the knowledge base

Two different types of rules are distinguished based on whether they refer to one object or to several objects. The former, termed intra-object rules, represent relations between attribute values of an object. They can be employed to determine an unknown attribute value of an object from one or more known attribute values of the same object. Typical examples of such rules are abstraction and refinement rules. Inter-object rules establish associations between attributes of different objects. Each of these two types of rules may be either a strict rule which must be satisfied under all circumstances, or a recommendation which should be satisfied whenever possible. Figure 3 shows the root of the domain ontology which indicates the most basic types of knowledge which are distinguished in the KB.

Figure 4 illustrates the organization of the knowledge base which can be seen as a Hypertext-like network of formal and informal knowledge units. Each piece of formal domain knowledge is linked to its informal representation from which it was constructed. By a *reason*-link it may be linked to another (informal) knowledge unit which indicates why a piece of knowledge is considered to be true. The domain meta layer provides additional documentation of a knowledge unit by linking it to the domain ontology, the information source from which it was acquired, and by indicating its application scope. Finally the formal knowledge units are linked to specific inference actions which indicate how they are employed to perform inferences. The combination of individual inference actions for the solution of some given task is described on the task layer, which is not shown in figure 4.

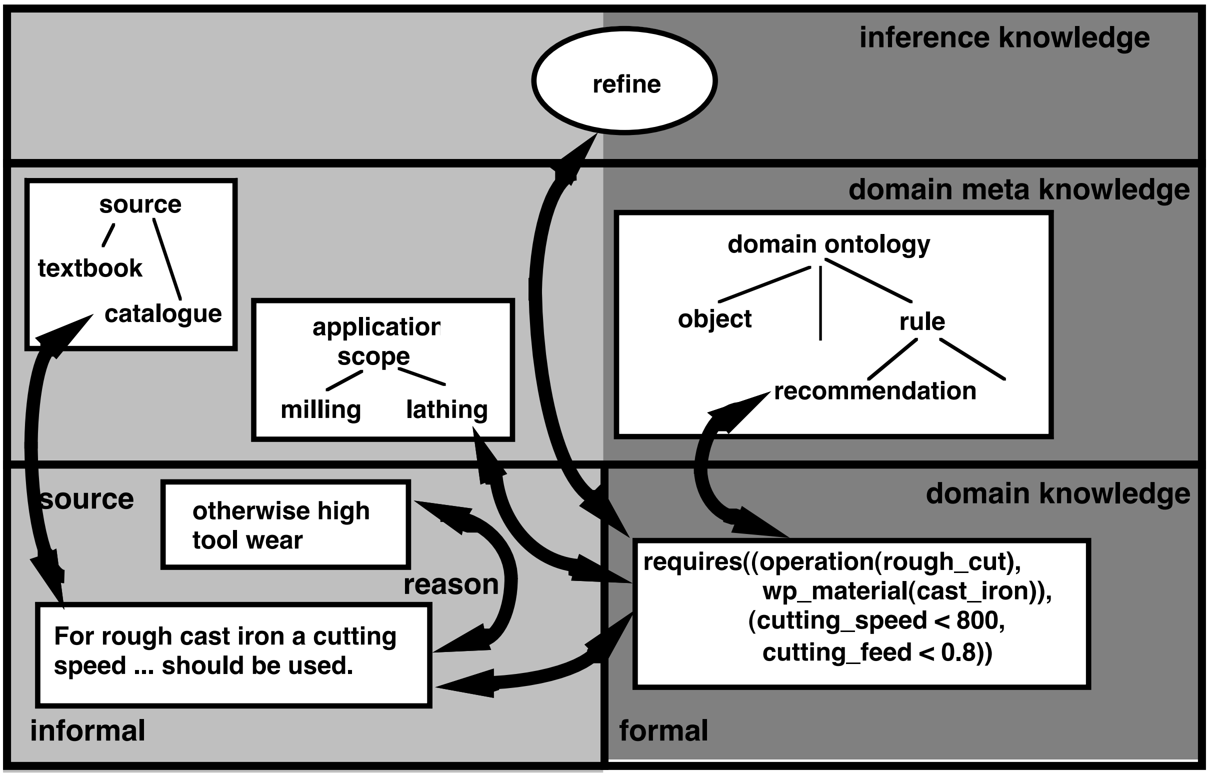

Fig 4: Organization of the knowledge base: A formal domain knowledge unit linked to informal knowledge, domain meta knowledge, and inference action(s)

3. Explanation and problem solving

As already indicated in Figure 1, an explanation of a problem solution may be requested by the knowledge (or system) engineer, the domain expert or the end user. These consultants have quite different explanation needs: The knowledge engineer and possibly also the domain expert will be interested in how the system actually derived the solution in order to verify that it works correctly. Obviously, this can only be shown by a trace-based explanation. The end user and mostly also the domain expert are, however, primarily interested in why the proposed solution is adequate. This can be shown by providing arguments which support (or criticize) the proposed solution. Thereby it may be advisable to completely disregard the actual solution trace since the information it contains has been tailored to computational needs, i.e. for effectively deriving a solution.

[WickThompson92] (p.38) define explanation as "an information processing operation that takes the operation of an information processing system as input and generates a description of that information processing operation as an output". Whereas this definition portrays well the essentials of a trace-based explanation, it gives a rather misleading characterization of reconstructive explanation.

Our view of the explanation problem and its relation to problem solving is depicted in Figure 5.[2] The left side of the figure illustrates the problem solving process which is triggered by a problem description, employes the information stored in the knowledge base, and produces a problem solution. The execution of the problem solving process may be captured in a solution trace.

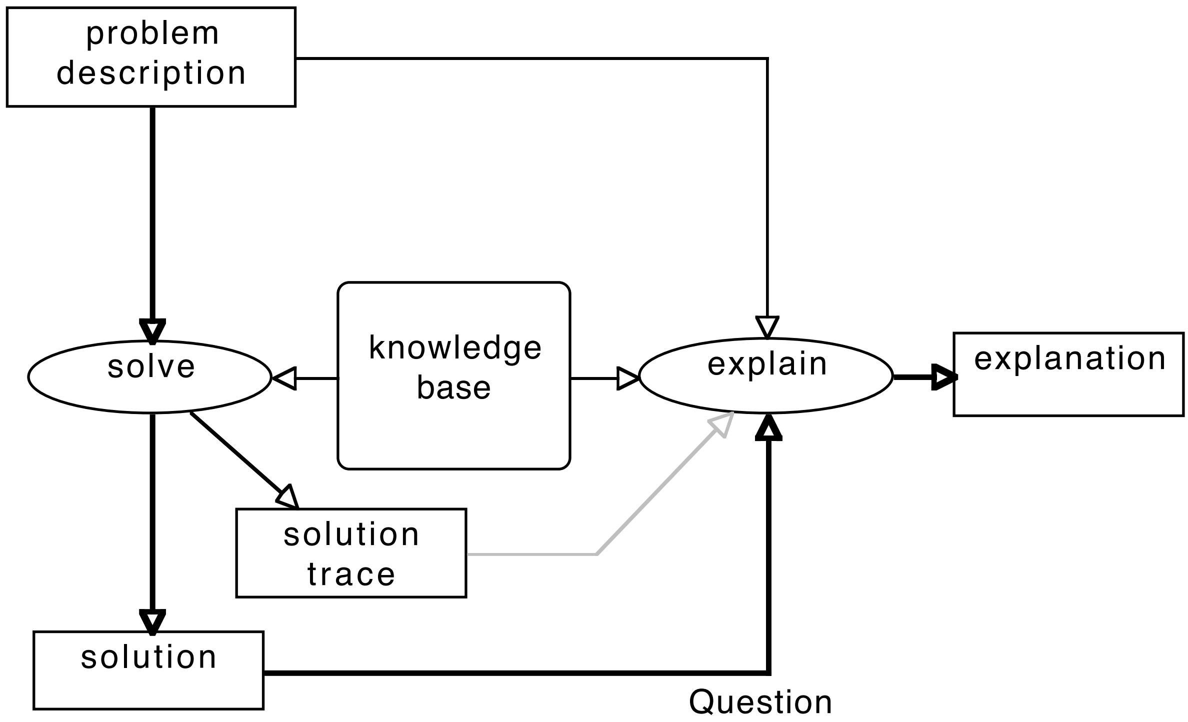

Fig. 5: Problem solving and explanation as information processing (Ellipses represent processes, boxes represent data, arrows indicate input and output, thick arrows highlight the principal input and output.)

The explanation process takes as input the problem description, the knowledge base and the problem solution (and possibly also the solution trace as indicated by a dashed line). As indicated by the thick arrow, the explanation process is typically triggered by a question referring to one particular aspect of the problem solution.

---

[2] This figure is analogous to figure 2 in [WickThompson92; p.39].

After the characterization of the problem solving and the explanation processes, we will now take a closer look on the different data structures which are the inputs and outputs of these processes.

Problem description:
The problem description **P** is a set of singular propositions which can be represented as ground facts in first order logic. For the lathe production planning problem presented in section 2.1 exemplary propositions of the problem description could be given by: "The length of the mold is 500 mm.", "The workpiece material is GGG80.", The workpiece is to be manufactured on a machine Boehringer PNE 480.", etc. A formal representation of these propositions in PROLOG notation is given in [Schmalhofer+91b].[3]

Solution:
The problem solution **S** is also seen as a set of singular propositions. The lathe production plan shown in Figure 2, can be described by propositions such as: "The cutting speed of the first cutting operation is 800 mm/sec.", "The cutting feed of the first cutting operation is 0.8 mm/rot.", etc.

Knowledge base:
The knowledge base has already been described in section 2.3 as a Hypertext-like structure consisting of a collection of formal and informal knowledge units which are structured by various links. If we forget the links and the informal knowledge for the time being, we can logically characterize the knowledge base as a set **K** of universal propositions. The different types of knowledge which were distinguished on epistemological grounds in figure 3, form a partition of **K** into respective subsets.

Solution trace:
The solution trace documents which and in what sequence elements of the knowledge base were employed to construct the solution from the problem description.

Explanation target:
The explanation target **E** is an element of the solution for which the user demands an explanation. The explanation target is specified in a question. An explanation episode will start with a question which asks for a justification of a singular proposition of the problem solution. Other types of questions will be discussed in the next section together with the description of RIETO.

Explanation:
In the theory of science [e.g. Hempel77], a logically correct explanation **X** of an explanation target (or "explanandum") **E** has been conceived as a set **T** of

---

[3]Of course, an object-centered representation would be epistemologically more adequate, since it emphasizes the inherent structures. But such a representation is logically equivalent to a set of ground facts. In the subsequent discussion will stick to the latter view, in order to illustrate the basic principles of reconstructive explanation at the knowledge level irrespective of representational details.

general propositions and a set **A** of singular propositions which together allow the deduction of **E**. Thus $\mathbf{X} = \mathbf{T} \cup A$, and $\mathbf{X} \Rightarrow \mathbf{E}$. In addition, it is usually postulated that an explanation X be minimal, i.e. it should not contain propositions which are redundant for the deduction of E. In order to be nontrivial, an explanation X must not contain the explanandum E.

In order to meet these requirements, both T and A must be nonempty, when E is a singular proposition. On the other hand, A must be empty, when the explanandum E is a general proposition, since no singular proposition can be employed for the deduction of a general proposition. Besides being logically correct, an explanation should also be plausible, evident, and novel.

4. RIETO: a Reconstructive Integrated Explanation Tool

The tool RIETO aims at providing reconstructive explanations for the various characteristics of a given lathe production plan. The explanations are based on the entire body of knowledge and information which was collected during knowledge acquisition. RIETO provides a graphical user interface which allows the user to conveniently ask questions about the given problem solution and to further inquire about pieces of information which were given as answers to previous questions.

RIETO was primarily conceived as an explanation component for the end-user of a knowledge-based system. Similar to traditional explanation tools, RIETO provides argumentative support for specific aspects of a suggested solution which are questioned by the user. In addition, RIETO can present all the knowledge and information which went into the development of the KBS which thus becomes transparent to the end-users. They can make a realistic assessment of the competence of the KBS and thus gain confidence into the quality of the provided solutions.

Since RIETO does not require a problem solving component, it can be used by the knowledge engineer and the domain expert for a focussed examination of the system´s knowledge which is important both during the development of the KBS as well as its subsequent evolution.

4.1. Input to RIETO

As already indicated in Fig. 5, the input to RIETO consists of a knowledge base together with its informal documentation, a given case, (i.e. a problem description together with its solution), and a particular question asked by the user. Whereas the first question of a dialogue must always refer to an aspect of the problem solution, subsequent questions may refer to elements of a previously provided explanation.

An overview of the different types of questions which can be asked to RIETO is given in Table 2 together with concrete examples from the domain of lathe production planning. These types of questions were found to be essential for providing comprehensive explanations which can establish a user´s confidence into the knowledge-based system.

"Why?" questions:
"Why?" questions are the fundamental type of questions for any explanation tool.[4] In RIETO, "Why?" questions provide logically correct and cognitively adequate explanations which satisfy the criteria stated in section 3. A "Why?" question may refer to a singular proposition or to a general proposition. The former may comprise either an attribute value given in the solution or a derived attribute value[5]. The latter may be any knowledge item which was presented in a previous explanation. As already pointed out by Clancey [Clancey83], being able to give further explanation of knowledge units presented in previous explanations, is an important prerequisite for a user-adequate KBS.

"Why not?" questions:
In RIETO, a user may specify an alternative value for an attribute of the presented solution and ask the system for drawbacks of the suggested alternative. Such "Why not?" questions are essential for resolving possible discrepancies between the presented solution and a user's expectations [RoussetSafar87]. Even moderately knowledgeable users will have some expectations about the solution of the problem and they will not feel comfortable with the given solution unless they can be shown why their expectations were not appropriate.

question type	**explanation target**	**example**
why?	solution attribute = value derived attribute = value	cutting_speed = 500 workpiece_material = cast_iron
why?	rule or recommendation	For cast iron the cutting speed should be lower then 600.
why not?	solution attribute = other value	cutting_speed = 800
source?	rule or recommendation	For rough cutting alloyed steel the cutting material SN80 is advantageous.
info?	attribute	cutting_speed = ?

Table 2: Characterization of the different types of questions which can be asked to RIETO.

"Source?" questions:
This type of questions was added to RIETO in order to enhance the transparency of the knowledge base. Similar to "Why?" questions, "Source?" questions provide additional information about rules and recommendations in

---

[4]) "How?" questions are not supported by RIETO since it adheres to a pure reconstructive explanation philosophy.

[5]) Obviously, the system cannot explain attribute values of the problem description, since this would require knowledge beyond the considered task, e.g. design knowledge in addition to planning knowledge.

the knowledge base. Whereas the former provide logical reasons of why the particular piece of knowledge is considered to be correct, the latter provide argumentative support by revealing the origin of the knowledge item, i.e. the bibliographic reference of the document from which it was extracted or the domain expert by whom it was mentioned. Such information usually plays an important role in discussions between human professionals. It should therefore be utilized by the explanation component of a KBS to provide cognitively adequate justifications of proposed solutions.

"Info?" questions:
This type of questions was added to RIETO in order to provide still broader information about an aspect of the problem solution which is questioned by the user. "Info?" questions may refer to an attribute of the problem solution and they can be translated as: "What do you know about determining the value of this attribute for the given problem?". "Info?" questions return all rules and recommendations of the knowledge base which are applicable under the given problem context. By showing the users the richness (or scarcity) of the system's knowledge which is relevant for the given problem they may gain confidence (or justified doubts) about the adequacy of the proposed solution.

RIETO facilitates asking questions by providing a list or tree of the possible explanation target. To ask a question the user simply selects the one of the offered explanation target and clicks a button to indicate the type of question. In order to allow for a more natural dialogue, RIETO always takes into account the context of the current question which thus need not be specified explicitly by the user. For instance, the user can simply ask "Info cutting_speed?" and RIETO automatically interprets this question as referring to that cutting operation of the currently considered case about which the previous question was asked.

4.2. Providing Explanations

We will now describe the principles according to which explanations are provided for the different types of questions without presenting the algorithmic details of the procedures which produce an explanation. Since the answers to all types of questions consist primarily of knowledge units from the integrated knowledge, all we have to do is to specify the criteria according to which the appropriate knowledge units are retrieved. Since "Why?" questions referring to an attribute value and "Why?" questions referring to a piece of knowledge call for different explanations as explained in section 3, they will be treated separately in the subsequent elaborations.

"Why?" questions referring to an attribute value:
As proposed in section 3, explanations for this type of "Why?" questions, can be provided by examining all the different proof trees for the questioned fact (i.e. attribute value), and presenting the most informative one to the user. It has also been argued that the selected explanation should consist of only one general rule R, i.e. T = {R}, together with the facts providing the antecedents

for this rule, i.e. A = {a1 .. an} with each a_i covering exactly one condition of R. The rule R is selected from the knowledge base according to the following criteria which are applied in succession, in order to determine the best piece of knowledge to be used in the explanation.

Criterion 1: Question pertinence. The general knowledge unit presented in the explanation should be related to the asked question in an obvious way so that the user can directly see that it provides an answer to the asked question. In RIETO, all rules and recommendations which contain the questioned attribute in their conclusion are selected according to this criterion. The given attribute value is ignored in this first selection step in order to include also those knowledge units which suggest a different value.

Criterion 2: Problem adequacy. The rule or recommendation used in the explanation must be applicable to the given problem description. This criterion can be tested by constructing proofs for the conditions of the rules based on the given problem description. The constructed proofs provide the antecedents A which are needed for the completion of the explanation.

Criterion 3: Solution consistency. This criterion selects those rules or recommendations which are consistent with the given solution. It thus introduces a bias in favor of the given solution by selecting those knowledge units which support it. Technically, criterion 3 is established by testing that the conclusions specified in the rule or recommendation are indeed satisfied in the given problem solution. For instance, if a recommendation says that the cutting speed should be lower than 800 m/sec and the cutting feed lower than 0.8 mm/rot, it will be tested whether both given attribute values satisfy these specifications. It might be argued that it is sufficient to check for the questioned attribute value, since this guarantees a logically correct explanation X of E so that X entails E. This is true, however, such an explanation X would entail some E' which is consistent with the explanation target but not with the whole solution. By taking into account the complete solution and not only the explanation target, RIETO avoids giving misleading explanations which do not truly support the given problem solution.

Criterion 4: Specifity. The provided explanation should be closely related to the given case. Since a case consists of a problem description and its solution, two types of specifity might be distinguished, namely *problem specifity* and *solution specifity*. A rule or recommendation is the more problem specific, the closer its condition is to the given problem description, and the more solution specific, the closer its conclusion is to the given problem solution. As a measure of closeness the depth or length of the respective proof tree might be employed. In RIETO, the problem specifity of a rule is determined based on the object hierarchies in the knowledge base. In order to assess the solution specifity for numerical attributes a simple comparison of intervals is employed.[6]

Computational considerations: The sequence in which the four criteria were introduced is based mainly on theoretical considerations and is not optimal for actually constructing an explanation. The question pertinence criterion (criterion 1), as defined above, can be easily tested and yields an effective reduction of candidates which must be tested with respect to the remaining criteria. This criterion is therefore applied first in RIETO. Since criterion 2 is computationally expensive, it may be advantageous to first apply criteria 3 and 4.

Presentation of explanation: As an answer to "Why?" questions referring to an attribute value, RIETO presents the stored informal representation(s) of the formal knowledge unit(s) which best satisfy the four criteria. If no knowledge units are found which are both question pertinent and problem adequate (i.e. which satisfy the first two criteria), RIETO tells that it cannot give an explanation. If the solution consistency criterion cannot be satisfied, RIETO informs the user that it cannot justify the suggested attribute value and lists the knowledge units which satisfy criteria 1 and 2 but suggest an alternative value. Finally, if several knowledge units cannot be distinguished with respect to the fourth criterion (i.e. they are equally specific according to the partial order established by the hierarchies), RIETO presents the first explanation found and tells the user that there are more arguments supporting the current solution which may be given on request.

"Why?" questions referring to general knowledge:

As discussed in section 3, these questions call for an explanation of a general proposition or "law". A logically adequate answer to such a question should provide a number of general propositions which entail the questioned proposition. Since RIETO does not operate on a compiled knowledge base but instead uses the knowledge as found in the information sources, there are usually no knowledge units or general propositions which entail other knowledge units in a strict sense. RIETO does therefore not look for knowledge units which entail the explanation target, but instead relies on the informal justifications for knowledge units which were captured during knowledge acquisition and are linked to the formal knowledge units via reason links.

When a "Why?" question is asked about a piece of knowledge, RIETO simply presents those knowledge unit(s) which are linked to the questioned unit via a reason link. If no reason link exists for the questioned knowledge unit, RIETO is unable to give an explanation. Since the reason links established in the knowledge base reflect the justifications given in the source materials which are written for practitioners in the field, it is to be expected that they cover those knowledge units for which an explanation will be required in practice.

---

6) For instance, interval I is considered more specific than interval J, iff I is contained by J, or both intervals intersect and I is shorter than J.

"Why not?" questions:
Since the first two criteria for answering "Why?" questions do not depend on the given attribute value, the criteria of question pertinence and problem adequacy can be also employed to select those knowledge units which are relevant for answering "Why not?" questions. Let K_{QP} be the set of knowledge units which satisfy these criteria. The criterion of solution consistency can now be applied to K_{QP} both for the original attribute value and for the alternative solution suggested by the user. Let K_{QPS} and K_{QPA} be the sets of knowledge units which satisfy the solution consistency criterion for the original solution and the alternative solution respectively.

If K_{QPS} is empty, all arguments which can be constructed by the system favor the alternative solution. The specifity criterion can the be applied to K_{QPA} to determine the most specific arguments. These are then presented to the user together with the notice that the solution alternative suggested by him is indeed better. If, on the other hand, K_{QPA} is empty, the user is presented the most specific arguments from K_{QPS} which support the original solution and contradict the alternative suggested by the user.

If neither K_{QPS} nor K_{QPA} is empty, there are arguments both for the original solution and for the alternative solution suggested by the user. The sets $K_{QPS} \setminus K_{QPA}$, $K_{QPA} \setminus K_{QPS}$ and $K_{QPS} \cap K_{QPA}$ are computed which contain the arguments pro the original solution and con the alternative, pro the alternative solution and con the original, and pro both solutions respectively. All three sets of arguments are presented to the user who is thus enabled to judge the two alternatives based on a broader background of knowledge.

"Source?" questions:
"Source?" questions are answered in RIETO by providing the information stored in the "Source" link of the questioned knowledge unit. For knowledge units acquired from written documents, the bibliographic reference of the source text is shown, whereas for knowledge units acquired from an expert, the name of the expert together with a brief description of the knowledge acquisition context are provided (e.g. "Focussed interview with X.Y about cutting parameters for iron workpieces on 12.1.92."). Additional information about the history of the knowledge unit [KühnSchmidt92] is currently not provided by RIETO, since this information is rather technical and mostly of interest for the knowledge engineer and the domain expert only. A brief account of the history of a knowledge unit (e.g. how did it gain its present form and why was a previous version abandoned) might, however, also be helpful to the end user, since it allows him to assess the effort invested into the development of the KB.

"Info?" questions:
"Info?" questions return all knowledge units which satisfy the first two criteria for "Why?" questions, i.e. question pertinence and problem adequacy. By asking an "Info?" question in addition to a "Why?" question, a user can

thus gain a full account of the system's knowledge which is relevant for determining the value of the questioned attribute in the given problem context.

4.3. A sample session with RIETO

Figure 6 shows the screendump of a sample session with RIETO. The interaction with RIETO proceeds via a window which is subdivided into four areas with different functionalities: the control panel, the dialog protocol, the attribute browser, and the solution grapher.

The control panel provides four buttons that correspond to the different question types supported by the system. Every action performed with RIETO is documented in the dialog protocol. Thus every user action, i.e. selection by mouse click, and in particular every answer of the system in an explanation dialog is displayed in the dialog protocol area of the window.

The attribute browser presents a list of items about which the user might ask further questions. These items refer to a special domain of interest. When explaining a production plan, such a domain of interest may be a particular operation which was selected by a mouse click in the solution grapher.

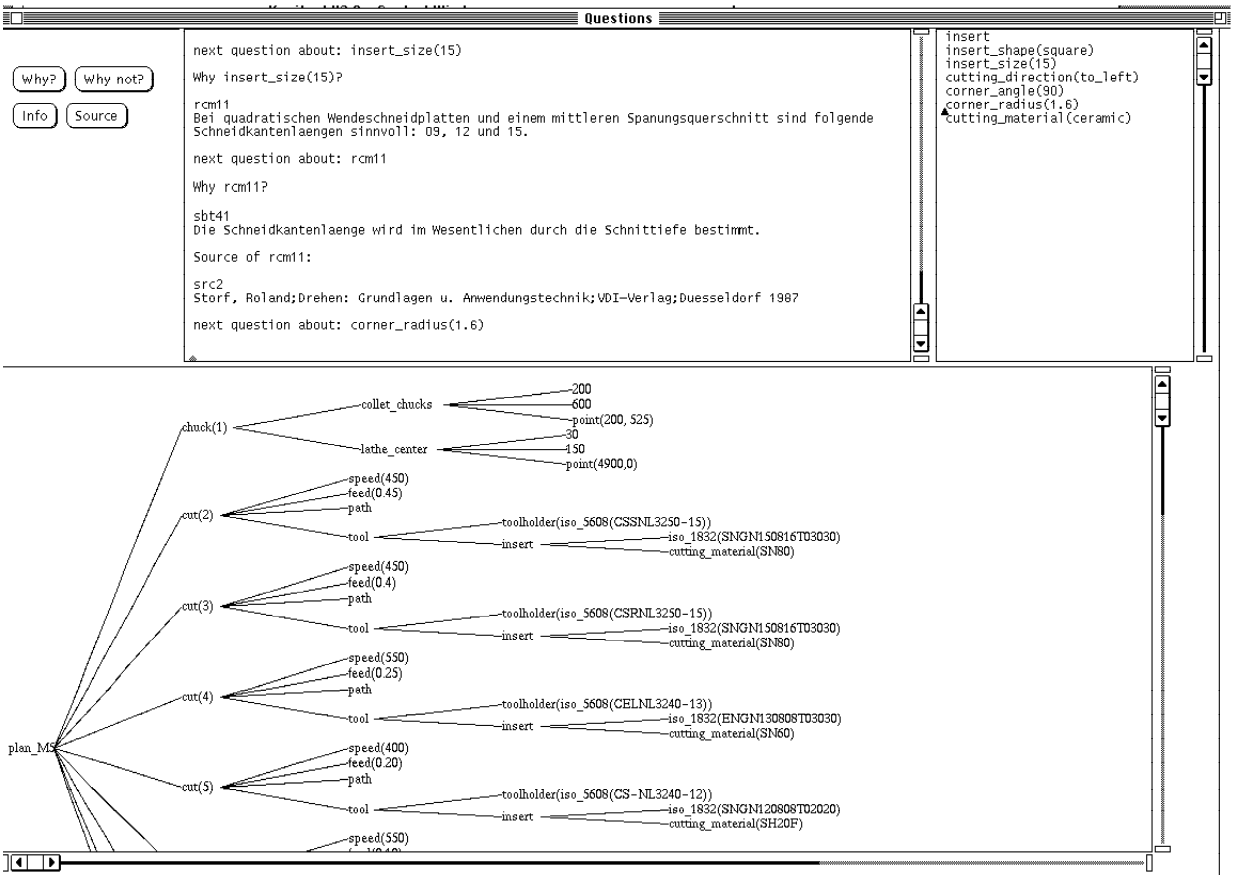

Fig. 6: A sample session with RIETO

The solution grapher shows the entire problem solution in a scrollable window. For the production plans the individual manufacturing operations steps are displayed with their parameters. The value of the cutting tool parameter is itself a complex object whose subparts and their respective attributes are presented in form of a tree.

The dialog session shown in figure 6 was started by clicking at "insert" in the "tool" subtree of "cut(2)" in the solution grapher. This causes the system to focus on the data of this tool's insert and display the data of this section of interest in the attribute browser. The user can select any attribute value pair from the browser to tell RIETO the explanation target. The user clicks on "insert_size(15)", the selected explanation target is prompted in the dialog protocol window. The user then asks a question by clicking the "Why?" button and RIETO returns as an answer the knowledge unit "rcm11" which says that "for cuts with a medium chip section, a square insert of size 15 should be used".

The name of this knowledge unit is displayed in the attribute browser area and can be selected as an explanation target for further question. In the sample session, the user then asks, why "rcm11" is valid. After the answer is displayed, the user wants to know the source of this knowledge unit, and RIETO presents the title of a basic text-book about lathing.

5. Discussion

Before comparing RIETO to similar approaches we will first discuss an apparent objection to a merely reconstructive explanation approach which has been advocated in this paper.

5.1. Doesn´t a merely reconstructive explanation deceive the user?

Whenever the knowledge used for providing an explanation of a problem solution is different from the knowledge which was actually used for constructing the solution, one might argue that the user who asked for an explanation is being deceived. What good is a nice and convincing explanation if there is no guarantee that the displayed knowledge has been applied to the problem at hand and will be applied to similar problems in the future?

This objection is definitely justified for the system developers who want to verify that the acquired knowledge is used correctly by the problem solving component. They must indeed be provided a trace-based explanation which closely reflects the actual solution trace. Most important for the end user, however, is a veracious assessment of the quality of the proposed solution. As has been previously argued, this can best be achieved by a reconstructive explanation which is not biased by and can even criticize the problem solving system.

In order to elucidate this question let us assume that for a given problem there is the optimal solution SO which works best in practice, the solution SP provided by the problem solving system, and the solution SE for which the explanation tool gives the most convincing arguments. Ideally, all three solutions should coincide, i.e. the problem solving component should return the optimal solution, and the explanation component should provide the most supportive arguments for that solution that indeed works best. An explanation tool misleads the user, if SE is rather distant from SO. The same is true for the problem solving component, if SP deviates strongly from SO. The distance between SP and SE is not as important to the end user as their distance to SO.

Since RIETO can focus on one aspect of the problem solution (the selected explanation target), it can bring to bear all relevant knowledge, whereas the problem solving component probably had to ignore some possibly relevant knowledge in order to find a solution at all. It is thus reasonable to assume that SE is qualitatively better than SP even though both inference systems operate on the same knowledge base. A merely reconstructive explanation tool which suggests to the user the better solution SE instead of blindly supporting SP will therefore be apt to be more useful than a trace based explanation tool which is always biased towards SP.

5.2. Comparison to related approaches

[WickThompson92] present an explanation system (REX) which can satisfy the needs of the different audiences by allowing to specify different degrees of coupling between the solution trace and the explanation. RIETO always constructs completely reconstructive explanations. Contrary to REX, RIETO can answer different types of questions which allow the user a better assessment of the quality of the proposed solutions than mere "Why?" questions.

The system XPLAIN [Swartout83] provides justifications of programs by examining the refinement structure created by the automatic programmer. It is similar to RIETO in that it brings to bear additional information which was available at prior stages of system development but is not accessible to the problem solving component. The refinement structures used in XPLAIN are analogous to the integrated knowledge base employed by RIETO. Whereas the former were produced by an automatic programmer and thus contain only formal information, the latter were manually constructed by the knowledge engineer in cooperation with a domain expert and they thus contain much richer and also informal knowledge.

The "Explainable Expert System (EES)" approach advocated by [Neches+85] also stresses the importance of explicitly recording the development history of an expert system and to exploit the stored information for explanation and maintenance.

Since RIETO can also criticize a given solution, it can also be compared to critiquing systems such as JANUS [Fischer89]. Such critiquing system (see [Fischer91] for a survey) are usually embedded in an environment which supports a user in constructing a solution to a given problem. They closely monitor the user's actions and present a critique (and advice) as soon as they detect an error or suboptimal conditions. Whereas the critiquing systems thus help the user to construct a solution in a cooperative problem solving effort, RIETO supports a user in detecting the merits and deficiencies of a given solution.

Acknowledgement

This research was supported by grant (FKZ ITW-8902 C4) from the Federal Ministry for Research and Technology.

We thank our colleagues Dr. Franz Schmalhofer and Gabriele Schmidt for the fruitful discussions, and the students Bidjan Tschaitschian, Marcus Specht, and Friederike Fuhrmann for their help in completing this work.

References

[BreukerWielinga89] Breuker, J. and Wielinga, B. Models of expertise in knowledge acquisition. In *Topics in expert system design, methodologies and tools.* North Holland, Guida, G. and Tasso, C., pp. 265 - 295, Amsterdam, Netherlands, 1989.

[Boley+92] Boley, H., Hanschke, Ph., Hinkelmann, K., Meyer, M., and Richter, M.M.. *VEGA - Knowledge validation and exploration by global analysis.* Project Proposal, German Research Center for Artificial Intelligence, Kaiserslautern, Germany, 1992.

[Clancey83] Clancey, W.J. The epistemology of a rule-based expert system - a Framework for Explanation. *Artificial Intelligence 20*(1983), 215-251.

[Fischer89] Fischer, G., McCall, R., and Morch, A. JANUS: Integrating hypertext with a knowledge-based design environment. In *Proceedings of Hypertext '89,* ACM, New York, 1989, pp. 105-117.

[Fischer91] Fischer, G., Lemke, A.C., Mastaglio, T., and Morch, A.I. The role of critiquing in cooperative problem solving. *ACM Transaction on Information Systems* (1991).

[GuarinoBoldrin92] Guarino, N. and Boldrin, L. Formal ontology for knowledge sharing and reuse. In *Proceedings ECAI-workshop on Knowledge sharing and reuse: ways and means*, Mars, N.J.I., Vienna, 1992.

[Hempel77] Hempel, C. Aspekte wissenschaftlicher Erklärung (Aspects of scientific explanations), Walter de Gruyter, Berlin, 1977.

[KühnSchmalhofer92] Kühn, O. and Schmalhofer, F. Hierarchical Skeletal Plan Refinement Task- and Inference Structures. In *Proceedings 2ndKADS User Meeting,* Bauer, C. and Karbach, W., Siemens☐AG, Munich, 1992.

[KühnSchmidt92] Kühn, O., & Schmidt, G. (1992). Knowledge Acquisition for Sharing and Reuse of Knowledge Bases. Workshop Notes "Knowledge Sharing and Reuse: Ways and Means", 10th European Conference on Artificial Intelligence, Vienna, 3-7 August 1992.

[Neches+85] Neches, R., Swartout, W.R. and Moore, J.D.. Enhanced Maintenance and Explanation of expert Systems Through Explicit Models of Their Development. *IEEE Transactions on Software Engineering*, 11, 1337-1351.

[RoussetSafar87] Rousset, M.C. & Safar, B. Negative and positive explanations in expert systems. *Applied Artificial Intelligence*, 1(1987), 25-38.

[Schmalhofer+91a] Schmalhofer, F., Kühn, O., and Schmidt, G. Integrated knowledge acquisition from text, previously solved cases, and expert memories . *Applied Artificial Intelligence* *5*(1991), 311-337.

[Schmalhofer+91b] Schmalhofer, F., Bergmann, R., Kühn, O., and Schmidt, G. Using an integrated knowledge acquisition method to prepare sophisticated expert plans for the reuse in novel situations. In *Proceedings of the 15th German Workshop on Artificial Intelligence,* Christaller, T., Springer Verlag , Berlin, 1991.

[Schmidt92] Schmidt, G. Knowledge Acquisition from Text in a Complex Domain. In Proceedings of the Fifth International Conference on Industrial & Engineering Applications of Artificial Intelligence and Expert Systems. University□of Paderborn, 1992 (auch DFKI RR-92-24).

[Swartout83] Swartout, W.R. XPLAIN: a System for Creating and Explaining Expert Consulting Programs. *Artificial Intelligence*, 21, 285-325.

[Wielinga+92] Wielinga, B., Schreiber, G., and Breuker, J.A.. KADS: A modelling approach to knowledge engineering. *Knowledge Acquisition.*, 4(1), 1992.

[WickThompson92] Wick, M.R. & Thompson, W.B. 1992. Reconstructive expert system explanation. *Artificial Intelligence*, 54, 33-70.

Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

DFKI
-Bibliothek-
PF 2080
67608 Kaiserslautern
FRG

DFKI Publikationen

Die folgenden DFKI Veröffentlichungen sowie die aktuelle Liste von allen bisher erschienenen Publikationen können von der oben angegebenen Adresse oder per anonymem ftp von ftp.dfki.uni-kl.de (131.246.241.100) unter pub/Publications bezogen werden.
Die Berichte werden, wenn nicht anders gekennzeichnet, kostenlos abgegeben.

DFKI Publications

The following DFKI publications or the list of all published papers so far are obtainable from the above address or via anonymous ftp from ftp.dfki.uni-kl.de (131.246.241.100) under pub/Publications.
The reports are distributed free of charge except if otherwise indicated.

DFKI Research Reports

RR-93-10
Martin Buchheit, Francesco M. Donini, Andrea Schaerf: Decidable Reasoning in Terminological Knowledge Representation Systems
35 pages

RR-93-11
Bernhard Nebel, Hans-Jürgen Bürckert:
Reasoning about Temporal Relations:
A Maximal Tractable Subclass of Allen's Interval Algebra
28 pages

RR-93-12
Pierre Sablayrolles: A Two-Level Semantics for French Expressions of Motion
51 pages

RR-93-13
Franz Baader, Karl Schlechta:
A Semantics for Open Normal Defaults via a Modified Preferential Approach
25 pages

RR-93-14
Joachim Niehren, Andreas Podelski, Ralf Treinen:
Equational and Membership Constraints for Infinite Trees
33 pages

RR-93-15
Frank Berger, Thomas Fehrle, Kristof Klöckner, Volker Schölles, Markus A. Thies, Wolfgang Wahlster: PLUS - Plan-based User Support
Final Project Report
33 pages

RR-93-16
Gert Smolka, Martin Henz, Jörg Würtz: Object-Oriented Concurrent Constraint Programming in Oz
17 pages

RR-93-17
Rolf Backofen:
Regular Path Expressions in Feature Logic
37 pages

RR-93-18
Klaus Schild: Terminological Cycles and the Propositional μ-Calculus
32 pages

RR-93-20
Franz Baader, Bernhard Hollunder:
Embedding Defaults into Terminological Knowledge Representation Formalisms
34 pages

RR-93-22
Manfred Meyer, Jörg Müller:
Weak Looking-Ahead and its Application in Computer-Aided Process Planning
17 pages

RR-93-23
Andreas Dengel, Ottmar Lutzy:
Comparative Study of Connectionist Simulators
20 pages

RR-93-24
Rainer Hoch, Andreas Dengel:
Document Highlighting —
Message Classification in Printed Business Letters
17 pages

RR-93-25
Klaus Fischer, Norbert Kuhn: A DAI Approach to Modeling the Transportation Domain
93 pages

RR-93-26
Jörg P. Müller, Markus Pischel: The Agent Architecture InteRRaP: Concept and Application
99 pages

RR-93-27
Hans-Ulrich Krieger:
Derivation Without Lexical Rules
33 pages

RR-93-28
Hans-Ulrich Krieger, John Nerbonne, Hannes Pirker: Feature-Based Allomorphy
8 pages

RR-93-29
Armin Laux: Representing Belief in Multi-Agent Worlds viaTerminological Logics
35 pages

RR-93-30
Stephen P. Spackman, Elizabeth A. Hinkelman:
Corporate Agents
14 pages

RR-93-31
Elizabeth A. Hinkelman, Stephen P. Spackman:
Abductive Speech Act Recognition, Corporate Agents and the COSMA System
34 pages

RR-93-32
David R. Traum, Elizabeth A. Hinkelman:
Conversation Acts in Task-Oriented Spoken Dialogue
28 pages

RR-93-33
Bernhard Nebel, Jana Koehler:
Plan Reuse versus Plan Generation: A Theoretical and Empirical Analysis
33 pages

RR-93-34
Wolfgang Wahlster:
Verbmobil Translation of Face-To-Face Dialogs
10 pages

RR-93-35
Harold Boley, François Bry, Ulrich Geske (Eds.):
Neuere Entwicklungen der deklarativen KI-Programmierung — *Proceedings*
150 Seiten
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

RR-93-36
Michael M. Richter, Bernd Bachmann, Ansgar Bernardi, Christoph Klauck, Ralf Legleitner, Gabriele Schmidt: Von IDA bis IMCOD: Expertensysteme im CIM-Umfeld
13 Seiten

RR-93-38
Stephan Baumann: Document Recognition of Printed Scores and Transformation into MIDI
24 pages

RR-93-40
Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, Werner Nutt, Andrea Schaerf:
Queries, Rules and Definitions as Epistemic Statements in Concept Languages
23 pages

RR-93-41
Winfried H. Graf: LAYLAB: A Constraint-Based Layout Manager for Multimedia Presentations
9 pages

RR-93-42
Hubert Comon, Ralf Treinen:
The First-Order Theory of Lexicographic Path Orderings is Undecidable
9 pages

RR-93-43
M. Bauer, G. Paul: Logic-based Plan Recognition for Intelligent Help Systems
15 pages

RR-93-44
Martin Buchheit, Manfred A. Jeusfeld, Werner Nutt, Martin Staudt: Subsumption between Queries to Object-Oriented Databases
36 pages

RR-93-45
Rainer Hoch: On Virtual Partitioning of Large Dictionaries for Contextual Post-Processing to Improve Character Recognition
21 pages

RR-93-46
Philipp Hanschke: A Declarative Integration of Terminological, Constraint-based, Data-driven, and Goal-directed Reasoning
81 pages

RR-93-48
Franz Baader, Martin Buchheit, Bernhard Hollunder:
Cardinality Restrictions on Concepts
20 pages

RR-94-01
Elisabeth André, Thomas Rist:
Multimedia Presentations:
The Support of Passive and Active Viewing
15 pages

RR-94-02
Elisabeth André, Thomas Rist:
Von Textgeneratoren zu Intellimedia-Präsentationssystemen
22 Seiten

RR-94-03
Gert Smolka:
A Calculus for Higher-Order Concurrent Constraint Programming with Deep Guards
34 pages

RR-94-05
Franz Schmalhofer,
J.Stuart Aitken, Lyle E. Bourne jr.:
Beyond the Knowledge Level: Descriptions of Rational Behavior for Sharing and Reuse
81 pages

RR-94-06
Dietmar Dengler:
An Adaptive Deductive Planning System
17 pages

RR-94-07
Harold Boley: Finite Domains and Exclusions as First-Class Citizens
25 pages

RR-94-08
Otto Kühn, Björn Höfling: Conserving Corporate Knowledge for Crankshaft Design
17 pages

RR-94-10
Knut Hinkelmann, Helge Hintze:
Computing Cost Estimates for Proof Strategies
22 pages

RR-94-11
Knut Hinkelmann: A Consequence Finding Approach for Feature Recognition in CAPP
18 pages

RR-94-12
Hubert Comon, Ralf Treinen:
Ordering Constraints on Trees
34 pages

RR-94-13
Jana Koehler: Planning from Second Principles —A Logic-based Approach
49 pages

RR-94-14
Harold Boley, Ulrich Buhrmann, Christof Kremer:
Towards a Sharable Knowledge Base on Recyclable Plastics
14 pages

RR-94-15
Winfried H. Graf, Stefan Neurohr: Using Graphical Style and Visibility Constraints for a Meaningful Layout in Visual Programming Interfaces
20 pages

RR-94-16
Gert Smolka: A Foundation for Higher-order Concurrent Constraint Programming
26 pages

RR-94-17
Georg Struth:
Philosophical Logics—A Survey and a Bibliography
56 pages

DFKI Technical Memos

TM-92-04
Jürgen Müller, Jörg Müller, Markus Pischel, Ralf Scheidhauer:
On the Representation of Temporal Knowledge
61 pages

TM-92-05
Franz Schmalhofer, Christoph Globig, Jörg Thoben:
The refitting of plans by a human expert
10 pages

TM-92-06
Otto Kühn, Franz Schmalhofer: Hierarchical skeletal plan refinement: Task- and inference structures
14 pages

TM-92-08
Anne Kilger: Realization of Tree Adjoining Grammars with Unification
27 pages

TM-93-01
Otto Kühn, Andreas Birk: Reconstructive Integrated Explanation of Lathe Production Plans
20 pages

TM-93-02
Pierre Sablayrolles, Achim Schupeta:
Conlfict Resolving Negotiation for COoperative Schedule Management
21 pages

TM-93-03
Harold Boley, Ulrich Buhrmann, Christof Kremer:
Konzeption einer deklarativen Wissensbasis über recyclingrelevante Materialien
11 pages

TM-93-04
Hans-Günther Hein:
Propagation Techniques in WAM-based Architectures — The FIDO-III Approach
105 pages

TM-93-05
Michael Sintek: Indexing PROLOG Procedures into DAGs by Heuristic Classification
64 pages

TM-94-01
Rainer Bleisinger, Klaus-Peter Gores:
Text Skimming as a Part in Paper Document Understanding
14 pages

TM-94-02
Rainer Bleisinger, Berthold Kröll:
Representation of Non-Convex Time Intervals and Propagation of Non-Convex Relations
11 pages

DFKI Documents

D-93-09
Hans-Ulrich Krieger, Ulrich Schäfer:
TDL ExtraLight User's Guide
35 pages

D-93-10
Elizabeth Hinkelman, Markus Vonerden, Christoph Jung: Natural Language Software Registry
(Second Edition)
174 pages

D-93-11
Knut Hinkelmann, Armin Laux (Eds.):
DFKI Workshop on Knowledge Representation Techniques — Proceedings
88 pages

D-93-12
Harold Boley, Klaus Elsbernd, Michael Herfert, Michael Sintek, Werner Stein:
RELFUN Guide: Programming with Relations and Functions Made Easy
86 pages

D-93-14
Manfred Meyer (Ed.): Constraint Processing – Proceedings of the International Workshop at CSAM'93, July 20-21, 1993
264 pages
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

D-93-15
Robert Laux:
Untersuchung maschineller Lernverfahren und heuristischer Methoden im Hinblick auf deren Kombination zur Unterstützung eines Chart-Parsers
86 Seiten

D-93-16
Bernd Bachmann, Ansgar Bernardi, Christoph Klauck, Gabriele Schmidt: Design & KI
74 Seiten

D-93-20
Bernhard Herbig:
Eine homogene Implementierungsebene für einen hybriden Wissensrepräsentationsformalismus
97 Seiten

D-93-21
Dennis Drollinger:
Intelligentes Backtracking in Inferenzsystemen am Beispiel Terminologischer Logiken
53 Seiten

D-93-22
Andreas Abecker:
Implementierung graphischer Benutzungsoberflächen mit Tcl/Tk und Common Lisp
44 Seiten

D-93-24
Brigitte Krenn, Martin Volk:
DiTo-Datenbank: Datendokumentation zu Funktionsverbgefügen und Relativsätzen
66 Seiten

D-93-25
Hans-Jürgen Bürckert, Werner Nutt (Eds.):
Modeling Epistemic Propositions
118 pages
Note: This document is available only for a nominal charge of 25 DM (or 15 US-$).

D-93-26
Frank Peters: Unterstützung des Experten bei der Formalisierung von Textwissen
INFOCOM:
Eine interaktive Formalisierungskomponente
58 Seiten

D-93-27
Rolf Backofen, Hans-Ulrich Krieger, Stephen P. Spackman, Hans Uszkoreit (Eds.):
Report of theEAGLES Workshop on Implemented Formalisms at DFKI, Saarbrücken
110 pages

D-94-01
Josua Boon (Ed.):
DFKI-Publications: The First Four Years
1990 - 1993
75 pages

D-94-02
Markus Steffens: Wissenserhebung und Analyse zum Entwicklungsprozeß eines Druckbehälters aus Faserverbundstoff
90 pages

D-94-03
Franz Schmalhofer: Maschinelles Lernen: Eine kognitionswissenschaftliche Betrachtung
54 pages

D-94-04
Franz Schmalhofer, Ludger van Elst:
Entwicklung von Expertensystemen: Prototypen, Tiefenmodellierung und kooperative Wissensevolution
22 pages

D-94-06
Ulrich Buhrmann:
Erstellung einer deklarativen Wissensbasis über recyclingrelevante Materialien
117 pages

D-94-08
Harald Feibel: IGLOO 1.0 - Eine grafikunterstützte Beweisentwicklungsumgebung
58 Seiten

D-94-07
Claudia Wenzel, Rainer Hoch:
Eine Übersicht über Information Retrieval (IR) und NLP-Verfahren zur Klassifikation von Texten
25 Seiten