31

N91-21944

COMPUTER-ASSISTED KNOWLEDGE ACQUISITION FOR HYPERMEDIA SYSTEMS

Kurt Steuck
Air Force Human Resources Laboratory

Abstract

The majority of this paper describes how procedural and declarative knowledge can be used to set up the structure or "web" of a hypermedia environment. The Air Force Human Resources Laboratory (AFHRL) has developed an automated knowledge acquisition tool (AKAT) that helps a knowledge engineer elicit and represent an expert's knowledge involved in performing procedural tasks. The tool represents both procedural and prerequisite, declarative knowledge that supports each activity performed by the expert. This knowledge is output and subsequently read by a hypertext scripting language to generate the links between blank, but labeled cards. Each step of the expert's activity and each piece of supporting declarative knowledge is set up as an empty node. An instructional developer can then enter detailed instructional material concerning each step and declarative knowledge into these empty nodes. The paper ends by describing other research that facilitates the translation of knowledge from one form into a form more readily useable by computerized systems. (Slide 2.)

Background

The Intelligent Systems Branch of the Training Systems Division of the Air Force Human Resources Laboratory (AFHRL) is continuing the research and development of intelligent training technologies. We investigate the application of artificial intelligence principles to compter-based training to produce training systems that behave intelligently. We have on-going efforts in the development of intelligent tutors in the operation, maintenance, and repair of complex physical devices, such as console operations and maintenance of the leading edge F-15 wing. We also have efforts which explore the application of artificial neural networks and other machine learning technologies to the modeling of a student as he or she progesses through training. A new effort will explore the application of virtual reality technologies to technical training. (Slide 3.)

Problem

Knowledge is available to instructional developers in forms that cannnot immediately be use for training or job-aiding. That knowledge is in our expert's heads, in paper form in technical documents and procedural guides, and in schematics. The problem is that we need to get that knowledge in a form that we can use in computer-based instruction, hypermedia, expert systems. We need to be able to translate, convert, or transform the knowledge from one form to a more readily useable form efficiently. Our approach in addressing this problem is to design and develop computerized tools that help us reconstruct existing knowledge to be used by computerized training systems. (Slide 4.)

The goal of this paper is to describe related technologies that convert knowledge form one form into different end products. One technology helps a training developer elicit and represent an expert's knowledge in performaing procedural tasks. The output of this tool can then be feed into a hypermedia environment to structure the "web" of information in much the same way as the expert described it to the training developer. The same knowledge can be feed into the second technology for use in other forms of job-aiding and training systems. The second technology is a natural language processor that takes procedural knowledge existing in technical manuals and converts it into expert system rules. To date, this sytem has been implemted in electronic domains to generate CLIPS rules.

Automated Knowledge Acquisition

Our first scenario is one in which critical domain knowledge resides in an expert, but is needed in for the development of an intelligent tutoring system or some other form of computer-based training system. Our approach is to design and develop computerized KA tools which automate knowledge elicitation, formalization, and representation processes. These tools are used to elicit knowledge from subject matter experts (SMEs) in analyzing or decomposing difficult tasks in the expert's domain. (Slide 5.)

The set of techniques used to elicit knowledge from a SME is referred to as knowledge acquisition (KA) or knowledge engineering. In KA for the development of training systems, a knowledge engineer, usually an instructional or cognitive psychologist, verbally elicits important domain knowledge from a subject matter expert (SME). This is a time consuming and difficult process, because the knowledge engineer typically is not intimately familiar with the problems, terminology, or problem solving techniques in the subject domain. Furthermore, the SME may experience difficulty in verbalizing relevant domain knowledge due to a lack of experience at describing the domain, the unavailability of proceduralized knowledge, or low motivation.

The objective of the Automated Knowledge Acquisition (AKA) effort is to design and develop tools which automate knowledge elicitation and representation. Successful automation would allow a SME and knowledge engineer to rapidly enter information directly into the computer without the aid of a computer programmer. A well designed interface would allow for authors to possess minimal computer skills and reduce some of the problems encountered by an expert and knowledge engineer during KA activities. The benefits of automating the representation of an expert's problem solving activities, events, and supporting information would reduce the time and resources required for KA. For instance, paper documentation is eliminated, formalization of the knowledge is less intensive, communication of the information to a computer programmer is not needed, and it does not require SME to fit his/her knowledge into a form constrained by the knowledge engineer.

Hypermedia Automated Knowledge Structure Acquisition Tool

The first technolgy to be described allows us to elicit procedural knowledge from an expert and represent it as the structure of a hypermedia system. (Slide 6.)

Overview of Steps

1. Start with classic expert scenario: the expert has years of experience, his/her knowledge is not documented, he/she may not be able to train many novices, and he/she may not be able to verbalize the domain information very well.

2. Together the knowledge engineer and the domain expert build a representation of the steps, subsequent events, and the supporting knowledge required to accomplish each of the steps.

3. The task description and the knowledge required to perform each step of the task is output in a frame representation.

4. The next step is to read the frames into the hypermedia system with a scripting language inherent in the hypermedia package. In our example, we used Hyperpad and its scripting language. We did not need to write any external code, such as C or Pascal, to accomplish the translation of the AKAT output into the hypermedia web. The knowledge represented in the procedural structure is reconstructed as the structure of the web. It is not the content per se that is important, but rather the structure of the knowledge. The scripting language was written to produce labeled, blank nodes or cards and their inks to next step or event and the prerequisite knowledge elicited in AKAT.

5. The training developer then enters more detailed instruction into the nodes for presentation to domain novices. A training developer then can interview the expert to fill in the cards with details of each step in the procedure.

Each of these steps will be covered in more detail in the next set of slides.

Automated Knowledge Acquisition (Interface)

The interface uses icons to represent different aspects of an expert's procedural knowledge and the scenarios or events that result when the expert performs each step. As an example one step is "Determine Impact on Load and/or Execute Plans" and the subsequent events are "Impact on Load Only," "Impact on Execute Only," "Impact on Both." (Slide 7.)

How is this done? The SME can create, move, or delete icons to represent the problem solving steps and the resulting events using a mouse and keyboard. The user creates this procedural net in a manner that reflects his/her own methods of completing the targeted task. After representing the procedures, the SME and KE fill in a node editor prerequisite knowledge related to each step. The SME can add prerequisite knowledge consisting of concepts, facts, procedures, and rules. For example, two concepts required in order to correctly perform the "Determine Impact on Plans" step are "Load Plans" and "Execute Plans." (Slide 8.)

AKAT Frame Output

After domain expertise has been elicited and verified, the knowledge can be output in a frame representation. The frame includes the name of the step, its children (i.e., the resulting events), and all of the related facts, concepts, rules, and procedures. (Slide 9.)

Hypermedia Knowledge Structure

The information contained in this frame is read by the scripting language to form the structure of the hypermedia web. It is not the content of any one node, but rather a set of nodes. One node is set up for each piece of information. One for each step, one for each of the resulting events, and so on. The links between the nodes are based on the category or type of information related to the node. (Slide 10.)

For example, the node "Determine Impact on Plans" has links to the events "Impact on Load Plans Only," Impact on Execute Plans Only," and "Impact on Both." Links to other nodes are based on the related concepts, facts, rules, and principles.

Hypermedia Nodes

Each node formed by the scripting language has several parts automatically created. Each node has a title derived from the name of the procedural step, links to the events, and links to supporting knowledge. Each node is initially blank. The training developer and omain expert can then systematically develop detailed instruction for each node. (Slide 11.)

The result of all this is a hyper environment that can be used for initial training or refresher training. A student can navigate (I had to use the word at least once in this paper) the web for procedural content or procede in a very comprehensive manner mixing visits to nodes containing procedural and supporting knowledge.

Expanded Approach

The AKAT to hypermedia translation is not the only approach we have taken in addressing the problem of having knowledge in different formats. AKAT was initially developed for converting procedural and supporting knowledge into LISP code for direct import into an intelligent tutoring system. This capability was designed to use the knowledge as part of the expert module of the ITS. We later, however, realized that we could dump out the knowledge in rule format. The preceding event set up the conditions for which a step was to be performed. This was easily represented as "IF <an event was true>, THEN <perform the step>." We modified AKAT to reproduce the net of steps and events as a set of IF <event>, Then <step> rules. (Slide 12.)

More importantly, another technology developed for transforming knowledge is a system called the Textual Automated Reduction System (TARS). The goal of the TARS effort was to develop a natural language processor that would read technical manuals (tech orders, TOs) and produce well-formed expert system rules. TARS reads constrained natural language text, reduces it to a regularized English form, and then matches the regularized statements to expert system rule templates. The knowledge in the regularized statements is then output in expert system rules, in this case, CLIPS. (Slides 13 and 14.)

We have also used this technology to convert an expert's problem solving steps elicited in AKAT to CLIPS rules. An intermediate form of the procedural knowledge (IF-THEN rules) was output by a variation of AKAT and translated by TARS. The outcome is that we can load the rule base into the CLIPS environment for use as the expert system of an ITS or use as a job-aid.

Summary

The problem we have been addressing is that knowledge exists in many forms and is sometimes needed in different formats. Our approach is to develop technologies that facilitate the translation of knowledge in order to produce computerized training or job-aiding systems. We have investigated ways of eliciting and representing undocumented expert's procedural knowledge for use in hypermedia, intelligent tutors, and indirectly expert systems. We have also explored NLP for reconstructing experts' procedural knowledge or knowledge embedded in procedural guides into a representation useable by ITS or expert systems. The goal is to have the capability to develop computerized training or job-aiding systems more efficiently. (Slide 15.)

COMPUTER-ASSISTED KNOWLEDGE ACQUISITION FOR HYPERMEDIA

AIR FORCE SYSTEMS COMMAND

DR KURT STEUCK

AF HUMAN RESOURCES LAB

4 DEC 90

OUTLINE

- BACKGROUND
- PROBLEM
- AUTOMATED KNOWLEDGE ACQUISITION TOOLS
- HYPERMEDIA
- EXPANDED APPROACH
- SUMMARY

SLIDE 2

BACKGROUND

- MISSION: RESEARCH AND DEVELOP INTELLIGENT TRAINING TECHNOLOGIES
  - INTELLIGENT TUTORING SYSTEMS
  - KNOWLEDGE ACQUISITION TOOLS
  - HYPERMEDIA
  - ARTIFICIAL NEURAL NETWORKS
  - VIRTUAL WORLDS

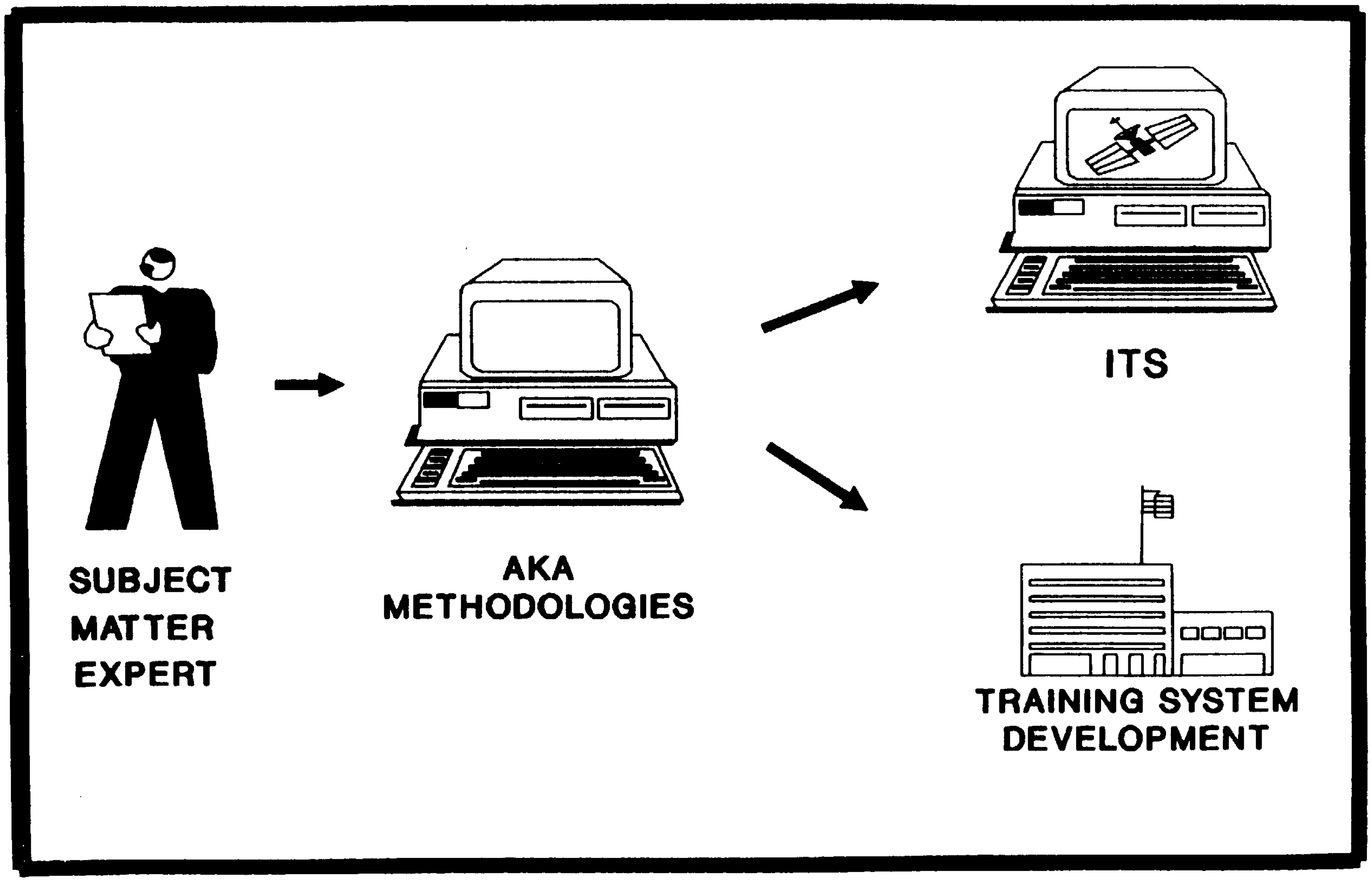
PROBLEM

- KNOWLEDGE EXISTS IN DIFFERENT FORMS
  - IN DOMAIN EXPERTS
  - IN TECHNICAL DOCUMENTS
  - DIAGRAMS, PICTURES, ETC

- KNOWLEDGE NEEDED FOR:
  - INTELLIGENT TUTORING SYSTEMS
  - HYPERMEDIA
  - TRADITIONAL COMPUTER-BASED TRAINING
  - EXPERT SYSTEMS

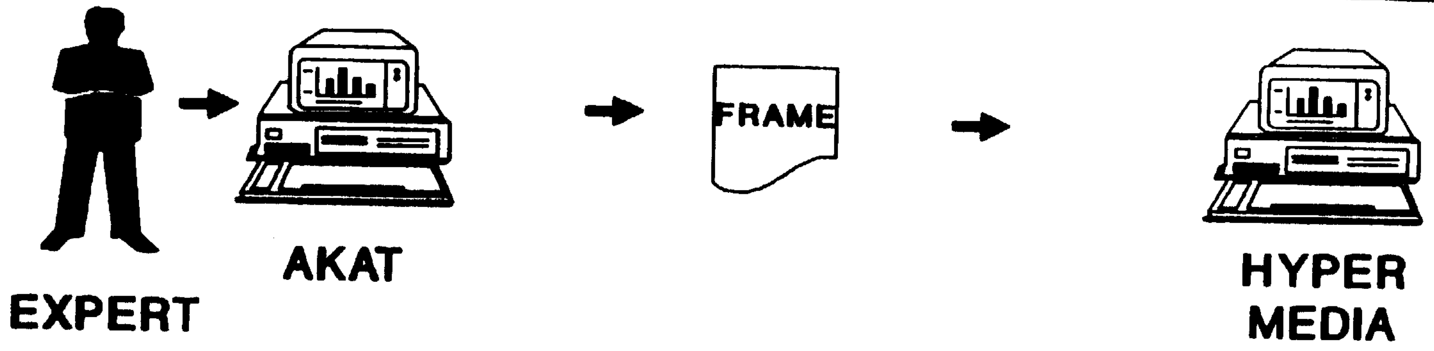
SLIDE 4

AUTOMATED KNOWLEDGE ACQUISITION



SLIDE 5

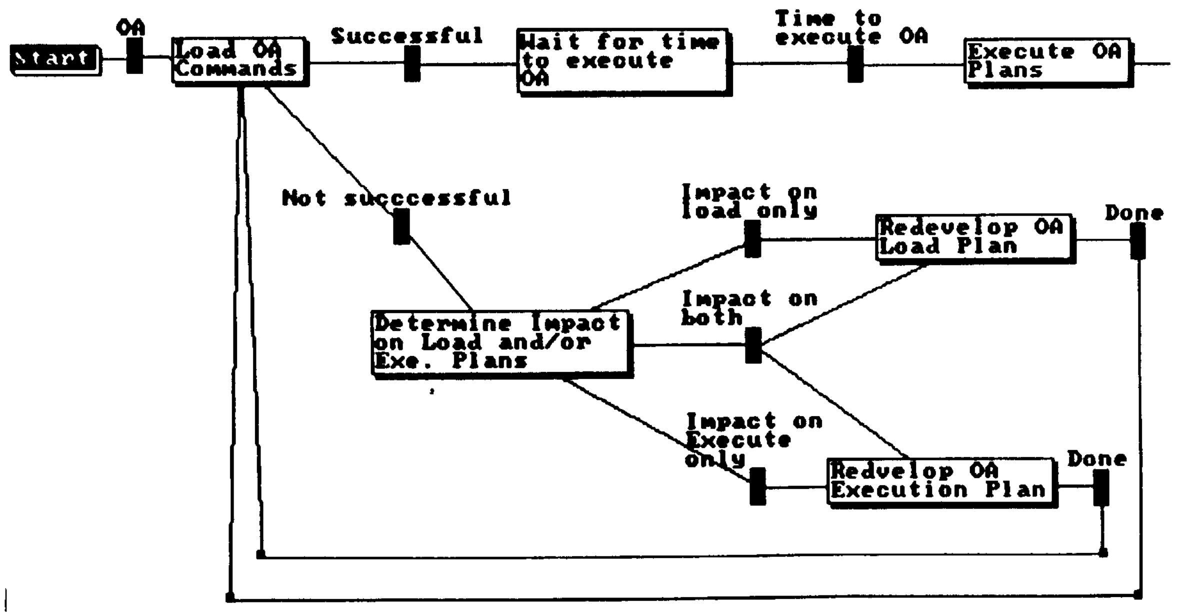
KNOWLEDGE ACQUISITION AND HYPERMEDIA



STEPS

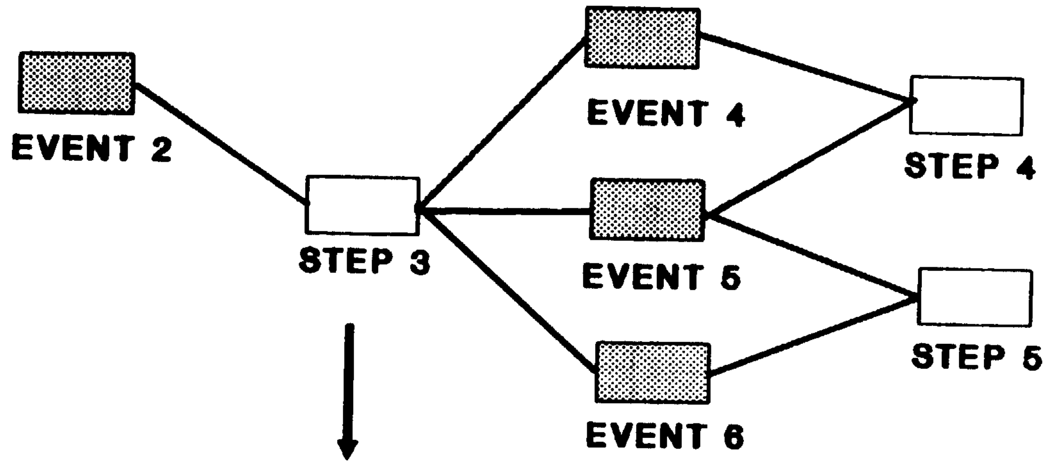
1. KNOWLEDGE ENGINEER INTERVIEWS EXPERT
2. REPRESENT KNOWLEDGE WITH AKAT
3. OUTPUT KNOWLEDGE IN FRAME
4. READ FRAME WITH SCRIPTING LANGUAGE
5. TRAINING DEVELOPER ENTERS INSTRUCTION

AUTOMATED KNOWLEDGE ACQUISITION



SLIDE 7

AUTOMATED KNOWLEDGE ACQUISITION



KNOWLEDGE FOR DETERMINE IMPACT ON PLANS	
CONCEPTS FACTS PROCEDURES RULES	1. LOAD PLANS 2. EXECUTE PLANS

SLIDE 8

AKAT FRAME OUTPUT

STEP: DETERMINE IMPACT ON PLANS

EVENT: IMPACT ON LOAD ONLY

EVENT: IMPACT ON EXECUTE ONLY

EVENT: IMPACT ON BOTH

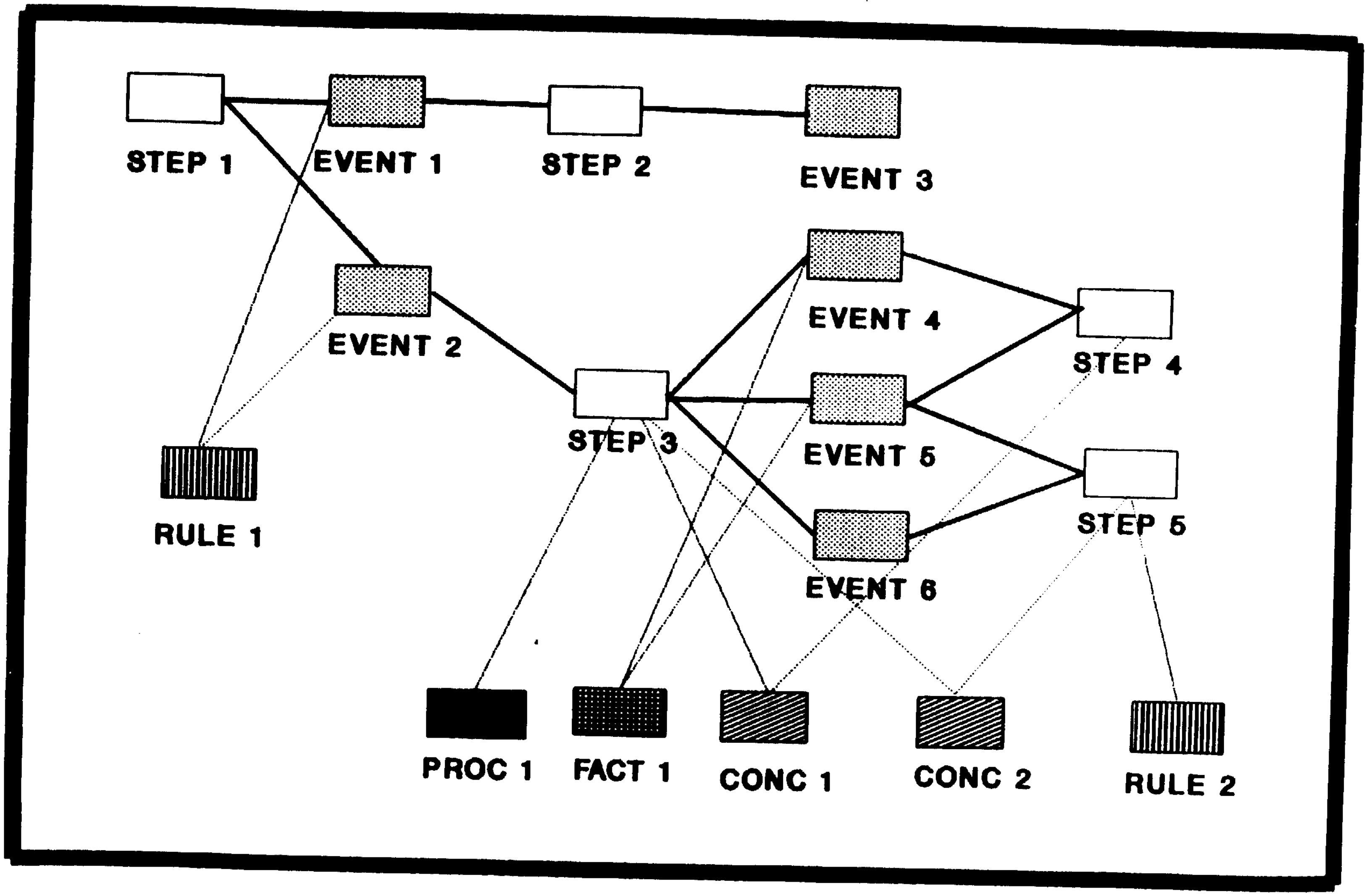
CONCEPT: LOAD PLANS

CONCEPT: EXECUTE PLANS

FACT: CHANGES ON LOAD AND EXECUTE PLANS
ARE INDEPENDENT OF EACH OTHER

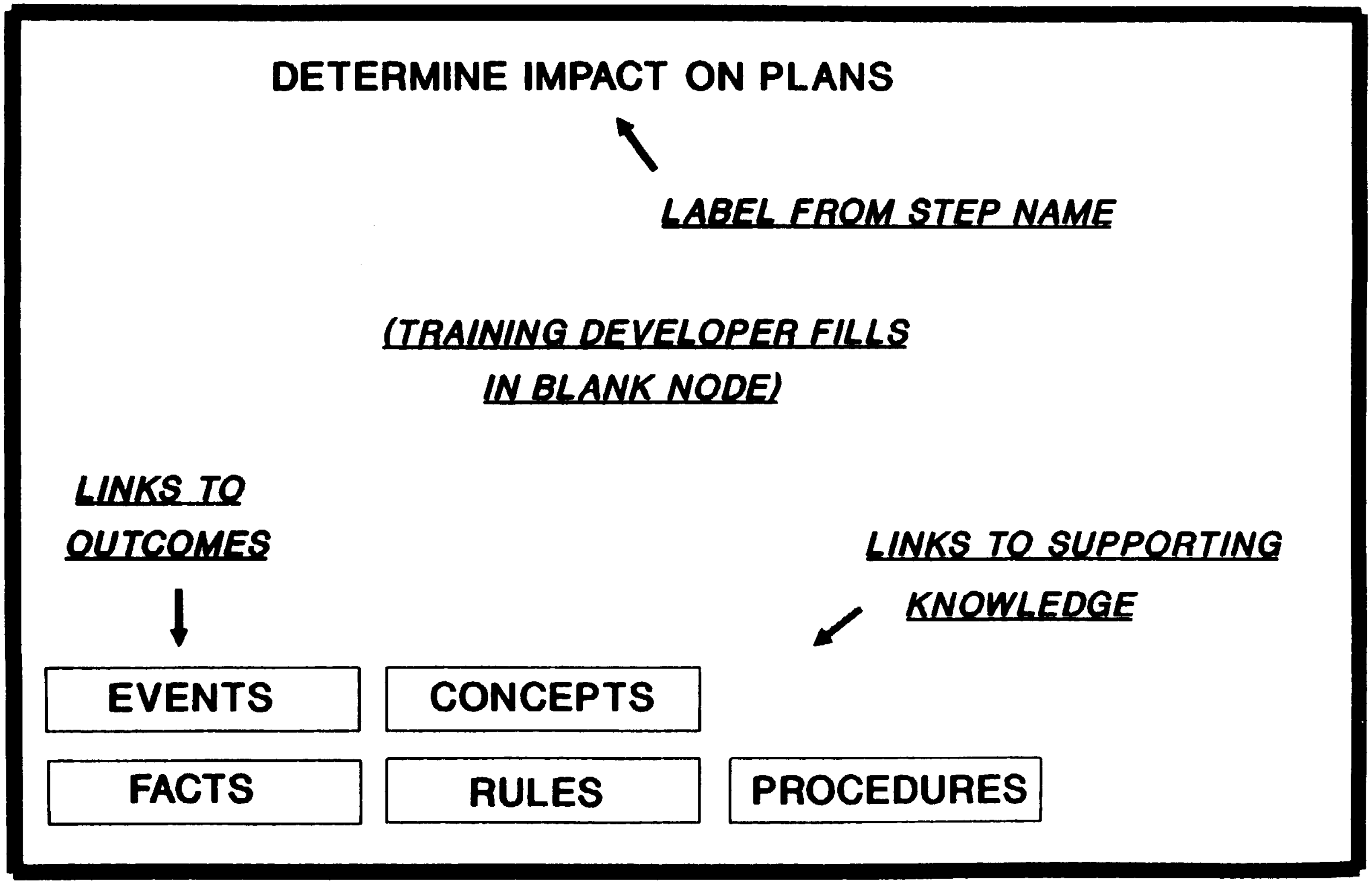
PROCEDURE: COMPARE PLANNED AND ACTUAL DATA BY ...

HYPERMEDIA KNOWLEDGE STRUCTURE



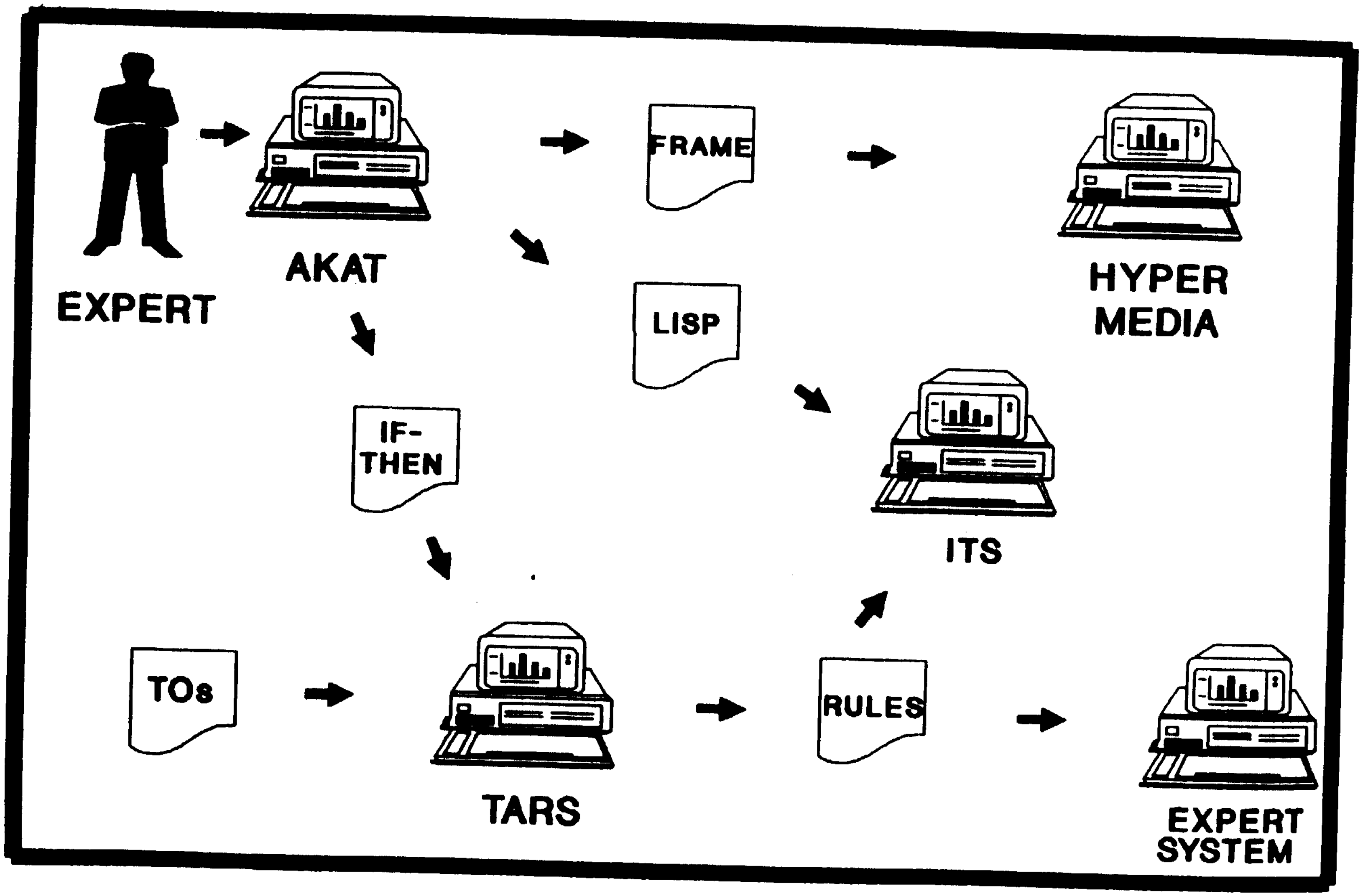
SLIDE 10

HYPERMEDIA NODE


DETERMINE IMPACT ON PLANS
LABEL FROM STEP NAME
(TRAINING DEVELOPER FILLS IN BLANK NODE)
LINKS TO OUTCOMES
LINKS TO SUPPORTING KNOWLEDGE
EVENTS
CONCEPTS
FACTS
RULES
PROCEDURES

Slide 11

EXPANDED APPROACH



SLIDE 12

TEXTUAL AUTOMATED REDUCTION SYSTEM (TARS)

- INPUT FROM TECHNICAL MANUALS

  "ROTATE THE AZIMUTH ADJ DIAL AS LONG AS
  THE TARGET INDICATOR IS ABOVE 3 VOLTS."

- INPUT FROM AKAT

  "IF LOAD OA SUCCESSFUL, THEN WAIT."

  "IF LOAD OA UNSUCCESSFUL, THEN DETERMINE IMPACT."

TEXTUAL AUTOMATED REDUCTION SYSTEM (TARS)

- OUTPUT FROM TECHNICAL MANUAL

```
(defrule rule0
(not 9<target indicator is above 3 volts> )) =>
 (printout t "<ROTATE <THE AZIMUTH ADJ DIAL>> " crlf)
(printout t "IS <TARGET INDICATOR IS ABOVE 3 VOLTS>? (Y/N)"
(bind ?ans (read))
(if (eq ?ans yes) then (assert (<target indicator is
 above 3 volts> ))
```

- OUTPUT FROM AKAT

```
(defrule7
(<LOAD OA UNSUCCESSFUL> ) =>
(assert (<DETERMINE IMPACT>)))
```

SLIDE 14

SUMMARY

- KNOWLEDGE
  - EXISTS IN MANY FORMS
  - MULTIPLE USES
  - NEEDED IN MANY FORMATS
- TOOLS TO HELP TRANSLATION
  - AKAT
  - HYPERMEDIA
  - TARS

SLIDE 15