ED 388 213 IR 017 362

AUTHOR	Gecsei, J.; Frasson, C.
TITLE	SAFARI: An Environment for Creating Tutoring Systems in Industrial Training.
PUB DATE	94
NOTE	7p.; In: Educational Multimedia and Hypermedia, 1994. Proceedings of ED-MEDIA 94--World Conference on Educational Multimedia and Hypermedia (Vancouver, British Columbia, Canada, June 25-30, 1994); see IR 017 359.
PUB TYPE	Reports - Descriptive (141) -- Speeches/Conference Papers (150)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	Artificial Intelligence; Computer Assisted Instruction; Computer Simulation; Foreign Countries; Higher Education; *Industrial Education; *Institutional Cooperation; *Intelligent Tutoring Systems; Professional Training; Programmed Instructional Materials; *Programmed Tutoring; Research and Development; Training Methods
IDENTIFIERS	Prototypes; Quebec

ABSTRACT

Safari is a cooperative project involving four Quebec universities, two industrial partners (Virtual Prototypes, Inc., providing the VAPS software package, and Novasys, Inc., a consulting firm specializing in artificial intelligence and training), and government. VAPS (Virtual Applications Prototyping System) is a commercial interface-building and simulation system. The main objective of Safari is to develop a methodology and an environment for the creation of tutoring systems to be used in professional training. The focus is on teaching mostly procedural knowledge concerning the operation of devices such as medical instruments, manufacturing robots, consumer appliances, control instruments, aeronautical instruments, etc. The basic idea is to add a tutoring component on top of device models (microworlds) built in VAPS. This permits the use of models written in VAPS ("virtual instruments"), instead of the real, expensive devices, for training and practice. The distinguishing features of the Safari environment are that: (1) an attempt is made to represent knowledge at two levels: at the physical level corresponding to a simulated device, and the plan level; (2) tutoring is based on four instructional modes: demonstration, exploration, coaching, and critiquing; and (3) within every mode the development of progressively more complex prototypes is foreseen, and tutoring in every mode involves the two levels of knowledge representation. The evolution of prototypes can be roughly divided into three phases according to the complexity of the knowledge structures involved, each of which is outlined. Safari prototyping of the Flo-Gard 6201 Volumetric infusion pump is descried as an example. Three figures illustrate concepts and the infusion pump prototype. (MAS)

SAFARI: an Environment for Creating Tutoring Systems in Industrial Training

J. GECSEI and C. FRASSON
Département IRO
Université de Montréal, CP 6128, Montréal, Québec, H3C 3J7, Canada
email: gecsei@iro.umontreal.ca

Abstract: Safari is a cooperative project involving universities, industrial partners and government. It aims at developing various architectures for industrial training, with different levels of complexity. The distinguishing features of the Safari environment are that (a) an attempt is made to represent knowledge at two levels: at the physical level corresponding to a simulated device, and the plan level; (b) tutoring is based on four instructional modes: demonstration, exploration, coaching and critiquing; (c) within every mode the development of progessively more complex prototypes is forseen, and tutoring in every mode involves the two levels of knowledge representation.

Creating tutoring systems remains difficult for various reasons: the different components of their architecture are not stabilized, the structure and type of knowledge to be used are of different granularity and complex to handle, and they require multiple expertise which is difficult to integrate [Fath *et al.* 1990, Lesgold *et al.* 1992].

The utility of tutoring systems having more or less intelligent capabilities is obvious. Most intelligent tutoring systems have been designed for academic domains such as mathematics, physics and computer programming. Although we do not lack teachers in such domains, there is still a well-known need for improving the quality of teaching in universities and colleges. However, there is an even more urgent need for computerizing industrial training. In todays global industrial competition, the skills required from workers are rapidly changing with the shifting technological environment, and employees must acquire new know-hows in limited time. The problem is compounded by a permanent lack of qualified instructors. For this reason, the research undertaken in the Safari project is focused on developing a tool set for the rapid construction of various tutors for industrial training.

Safari is a project under the auspices of Synergie, a programme sponsored by the Ministry of Technology and Science of the Governement of Québec. The main objectives of Synergie are (1) to enhance cooperative research and development between universities and industry, (2) to accelerate the product development cycle, (3) to facilitate the transfer of knowledge between research establishments and industry, and (4) to educate highly qualified professionals in the domain. Safari involves four Québec universities, two industrial partners and a government agency. The industrial partners are Virtual Prototypes Inc., providing a software package VAPS and Novasys Inc., a consulting firm specialized in the AI and training domains. VAPS (Virtual Applications Prototyping System) is a high-quality commercial interface-building and simulation system, used in many areas (such as industrial design and evaluation of airline cockpits).

The Safari team includes seven professors conducting research on a part-time basis : C. Frasson (project leader), G. Gauthier, J. Gecsei, G. Imbeau, M. Kaltenbach, S. Lajoie and B. Lefebvre; about 20 M.Sc and Ph.D students, two full-time programmers and several engineers provided by the industrial partners. The project time frame spans 4 years (1993-96).

Building reasonable tutoring components

The main objective of Safari is to develop a *methodology* and an *environment* for the *creation of tutoring systems to be used in professional training.* The focus is on teaching mostly procedural knowledge concerning the operation of devices such as medical instruments, manufacturing robots, consumer appliances, control instruments, aeronautical instruments, etc.

The basic idea is *to add a tutoring component on the top of device models (microworlds) built in VAPS.* This permits the use of models written in VAPS ("virtual instruments"), instead of the real devices, for training and practice. Thus the expensive machinery can be kept for its actual purpose and damages from incorrect manipulations during learning can be prevented. Safari can be seen as a value-adding component attached to the VAPS package, making it into an integrated toolkit for the creation of device models together with a corresponding computerized training course on how to operate the device. Since a large base of VAPS device models are already available, these can be conveniently used to validate the Safari approach.

In order to remain within the bounds of practicality, we limited a priori the target applications to devices which have the following characteristics:

- They permit the definition of a set of clearly distinguishable interaction procedures (e.g. through control panels, switches).
- The device functions are decomposable into a number of well-defined *tasks*, each task corresponding to a meaningful operation in terms of the device's main purpose.

These characteristics exclude very complex equipment such as nuclear reactors (although parts of it may be covered) and devices whose operation requires hard to enumerate, "look-and-feel" actions such as handling certain manufacturing machines, or performing surgery.

Tutoring systems created in the Safari environment will be capable of :

- tutoring workers in an industry (factory, hospital) through *real tasks* recreated on device models using the VAPS software,
- constructing *models of individual users*, reflecting the user's reasoning style and level of knowledge,
- dispensing *individualized and adaptive tutoring* based on the user model,
- diagnosing the actions of a trainee when asked to perform a task on the virtual instrument, and
- composing an entire curriculum for teaching the operation of the device, taking into account the learner's expertise and style of learning.

Some distinguishing features of Safari are that

- it involves contractual cooperation between the industrial and university partners who are committed to a binding schedule including the development of commercially viable products,
- the proposed methodology is applicable to a large class of devices and domains, not only to a single device type,
- it is based on an existing software package VAPS, and
- it enhances learning by using multiple representations of knowledge, and by encouraging the learner to pass between them.

We do not aim exclusively at deeply intelligent tutoring; this would be somewhat unrealistic, given the limited success and the problems such systems have even within narrow application domains. Instead, Safari provides a combination of some existing tutoring approaches to build easily and rapidly a tutoring system for a given device.

A realistic training sequence

Throughout the project time frame we use the following functional paradigm. In Safari, tutoring functions (modes) are based on the observation that the natural cycle in which most people acquire a given skill is by first *observing* someone's demonstration of the skill, then freely *experimenting* with the device in question (given the availability of the device, and that such experimenting is not hazardous), then *executing* precise tasks (assignments) in terms of the device functionalities under the guidance of an expert, and finally by *describing* the learned skill in an abstract form (which can be used as a basis for a critical examination by an expert, or for communicating to other people).

Somewhat simplifying, in our approach this translates into four distinct *tutoring modes*: demonstration, free exploration, coaching and critiquing. These modes are presented to the learner through an interface which is similar in all modes, and which displays two major windows: the Device Window showing the simulated device, and the Plan Window showing an abstract representation of a task called *task graph*. In this arrangement the learning process involves two related views of the learning space, encouraging the learner to develop his knowledge simultaneously on the concrete and abstract level. This method has its origin in the "PIF" approach to tutoring, described in [Frasson *et al.* 1992]. PIF supports three interrelated "worlds" (i.e. levels of representation): the Physical, Intentional and Functional. In Safari we retained the P and I worlds, corresponding to the Device and Plan windows.

In *demonstration mode,* the student can observe the execution of various tasks applicable to the device, somewhat like a videotaped demonstration. However, in Safari the demonstration is based on the VAPS model, executing automatically a task graph (TG) which had been edited by a human expert. A *demonstration scenario* is a TG augmented by hypermedia comments and explanations attached to certain elements of the TG. The student can navigate in the scenario by stopping, repeating, jumping to an arbitrary point (i.e. by clicking on an action in the TG), and by asking for hypermedia explanations. However, he *cannot* interact directly with the simulated device, only with the scenario. As the demonstration progresses, actions in the TG are highlighted along with the corresponding simulated manipulations of the device (such as operating the controls).

The purpose of *exploration mode* is to permit direct interaction with the microworld (virtual device). Here the learner is allowed to freely experiment with the device, to observe the consequences of his actions in a realistic environment, but without the inconveniences of experimenting with the real device [Fath *et al.* 1990]. The system may attempt to determine which task the learner is trying to execute; in this case the TG of the conjectured task is gradually displayed in the task window according to the learner's progress in manipulating the device. Exploration can also be done within the context of a given task; then the student is allowed to interact with either window, with automatic highlighting of the corresponding actions in the other window.

In *coaching mode* the student is asked by the system to perform a given task or to solve a given problem scenario (=sequence of tasks) by interacting only with the Device Window. Initially the correct TG is hidden; it is gradually revealed as the learner's solution is being built. The process is supervised by a coaching component, essentially analyzing the difference between the correct solution(s) from the knowledge base and the learner's construction, and generating appropriate coaching interventions.

The fourth tutoring mode is *critiquing,* where the learner has to prove his knowledge by constructing a solution in the form of a TG for a given problem. The learner can "experiment" in the device window as in exploration mode, and build the solution in the Task Window. In this way he is encouraged to pass frequently between alternative representations in the two windows; in other words, the student has to prove not only that he can correctly execute a task, but also that he *understands* (to a certain degree, of course) what he is doing. By requesting the student to construct a TG, he is effectively asked to express his knowledge in an abstract form. When the proposed solution is complete, a critiquing component (similar to the coach) will comment on the solution.
At any time during work in the above tutoring modes, the student can ask for help (non-contextual) or contextual advice.

Evolution

As shown in Figure 1, the four tutoring modes are considered as evolving in time from minimal versions to more sophisticated. In this way we can keep the project consistent over a period of time, and by producing a series of prototypes of increasing complexity we can exploit the experience learned on the way. We firmly believe that a well functioning complex system can be created only by evolving through simpler versions. The evolution of prototypes can be roughly divided in three phases according to the complexity of the knowledge structures involved. In the following, we briefly characterize these phases.

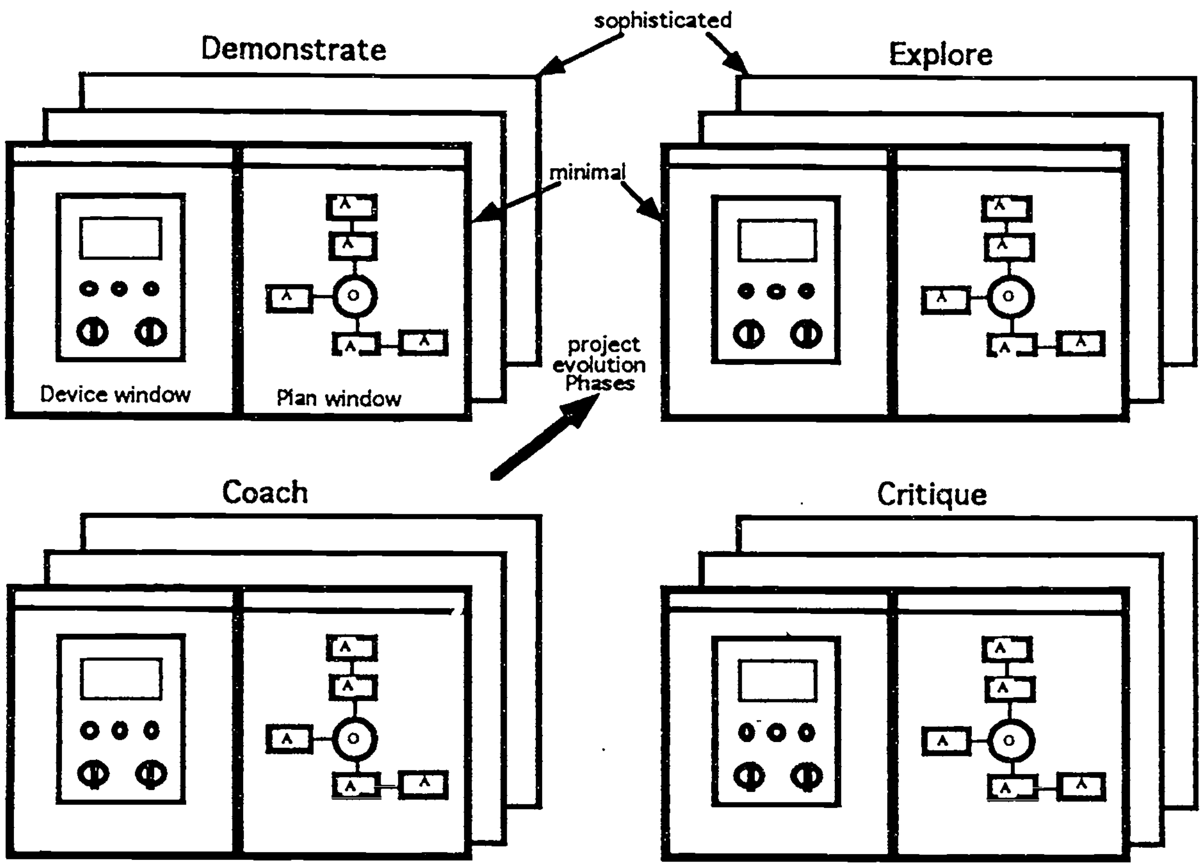

Figure 1. Safari tutoring modes and evolution.

Phase 1: In Phase 1, task graphs are equivalent to flowcharts containing two kinds of nodes: Action (A) and Observation (O) nodes [Mittal *et al.* 1988]. Actions are operations to be executed by the user, Observations serve as decision points where the following action depends on some observable behavior of the device. There is one TG per task, and all tasks can be clearly distinguished. The knowledge base consists of the set of corresponding TGs. A demonstration is a pre-recorded manipulation of the VAPS virtual device; in exploration mode the student can interact freely with the model. Coaching and critiquing are based on overlay-like comparisons of the student's input and the correct TG from the knowledge base. Even so, some nontrivial comments can be generated such as "Your solution is almost correct, except that you inversed actions A and B."

Phase 2: Here, we will take into account some factors that make the model of Phase 1 overly simplistic. First, in many cases there is more than one way to accomplish a task. These ways may be equivalent, or can have different attributes in terms of efficiency, cost, etc. We have to deal with this multiplicity in terms of efficient knowledge base representation and retrieval mechanisms. Second, the VAPS-based virtual device may be insufficient for efficient coaching and critiquing interventions which may need, for example, causal explanations. To achieve this we will introduce a *cognitive model* of the device, complementary to the lower level VAPS model. In Phase 2, the internal representation of knowledge (tasks, cognitive device model) will be based on concepts and relations integrated in an object structure. A simple student model (overlay) will permit to individualize tutoring interactions.

Phase 3: Here, tutoring will involve other types of device knowledge, such as structural, functional and maintenance. In further experimenting with the coaching and critiquing modes, we will emphasize the idea of having the learner gradually *create* and *refine* his knowledge simultaneously on two levels (instead of just showing a task and asking him to prove his knowledge). The student model will permit to infer the student's reasoning from his actions and adapt the coaching interventions accordingly. We will attempt to make certain types of knowledge transferable to new situations [Wielinga *et al.* 1992]. Figure 2 shows the main functional blocks of the Safari system.

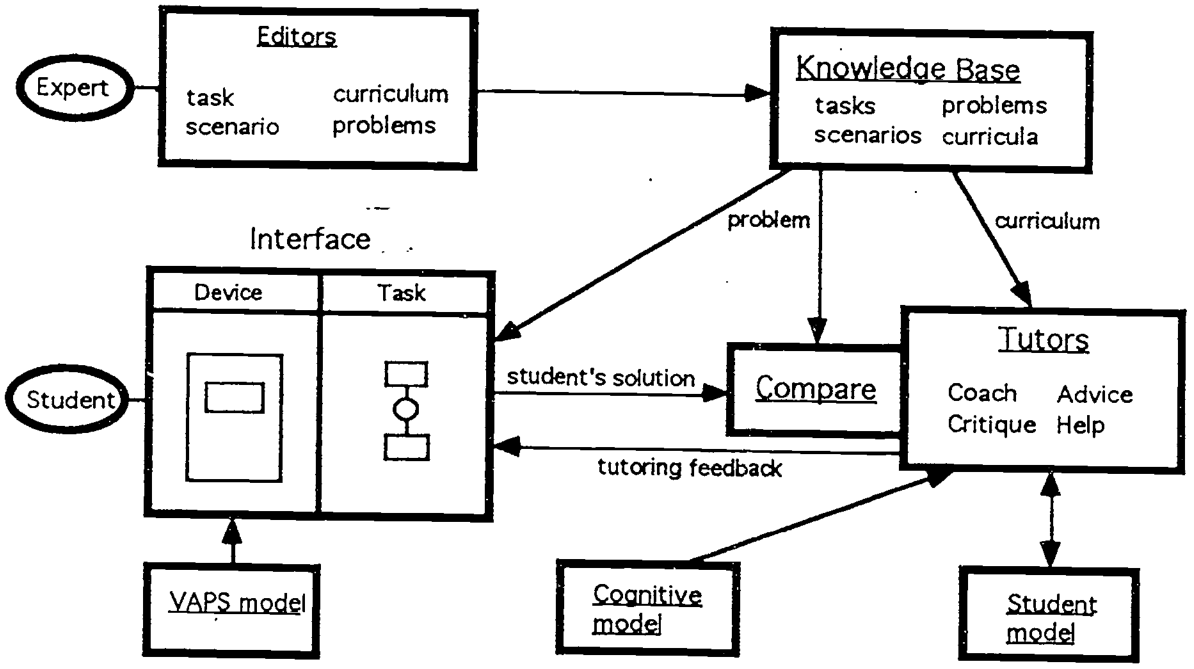

Figure. 2. Safari functional architecture.

Each component will gain in complexity in successive phases. The architecture is organized in a way that makes it possible to implement a layered development of the components. Each component contains a base to which a more complex module can be attached. Prototype implementation is under way on Sun and Silicon Graphics workstations. The two main programming languages used are VAPS (used for the virtual device and the demonstration editor) and Smalltalk (used for the other components).

Implementation: example

One of the first devices used for Safari prototyping is from the medical domain: the Flo-Gard® 6201Volumetric infusion pump fabricated by Baxter Healthcare Corporation. The device, used mostly in emergency rooms, can deliver a variety of fluids over a broad range of infusion rates. The rates can be programmed in different ways, such as over a fixed period of time, or with constant rate until a given volume has been reached, or as a sequence of infusion programs. The machine can control two independent simultaneous infusion processes. During an infusion, a number of special events can occur, such as lack of fluid in the reservoir, or an air bubble in the infusion tube. The trainees (health technicians and nurses) have to learn the basic operating procedures, as well as to handle special events. The Device Window on the left of Fig. 3 shows the front panel of the pump, as simulated in VAPS.

A cognitive analysis of the learning situation resulted in some 20 tasks which are expressed as task graphs. Related tasks are regrouped in curriculum issues; these were in turn classified according to their degree of difficulty (novice, intermediate, advanced). The righthand side of Fig. 3 contains a scenario editor used for creating a demonstration of a given task. This type of demonstration consists of actions (pre-recorded manipulations of the virtual device, represented by trace-footstep icons), and written and spoken comments (pre-recorded files represented by appropriate icons). The editor permits to arrange the icons in a desired presentation sequence. When an action is being replayed, the corresponding manipulations of the device controls (switches, keys) are visualized by the hand icon moving to the appropriate places, and by highlighting the activated controls.

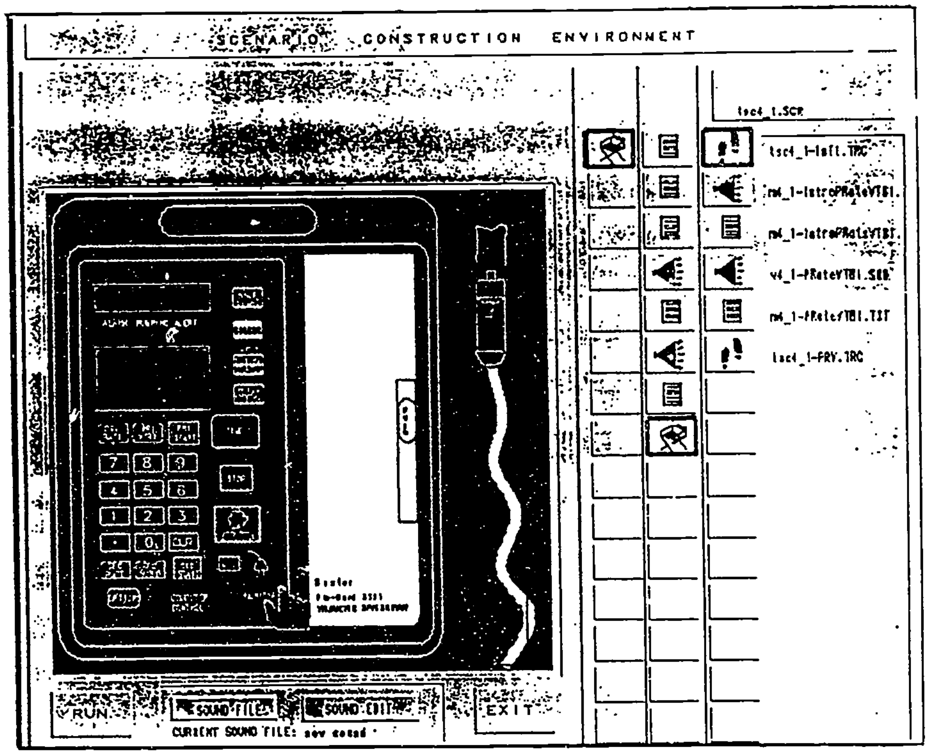

Figure 3. Infusion pump demonstration scenario editor.

Acknowledgements

A number of students contributed to the work reported here. We are particularly indebted to J.-Y. Djamen, A. Kabbaj, A. Mayers, R. Bouchard, A. Kengne, L. Quirion, Ch. Abadie, L. Duperval, R. Nkambou, Tang Ho Le, D. Phalp and N. Sandrasegaran.

References

Fath, J.L., Mitchell, C.M., Govindraj, T. (1990) *An ICAI Architecture for Troubleshooting in Complex, Dynamic Systems*, IEEE T-SMC, 20,3,pp. 537-558.

Frasson, C., Kaltenbach, M., Gecsei, J., Djamen, J-Y. (1992) "*PIF: an iconic intention-driven ITS environment*", in Frasson, C., Gauthier, G., McCalla, G. (eds), "Intelligent Tutoring Systems", Lecture Notes in Computer Science, vol 608, Springer-Verlag.

Lesgold, A., Lajoie,S., Bunzo, M. and Eggan, G. (1992)*SHERLOCK: A Coached Practice Environment for an Electronics Troubleshooting Job*, pp. 201-238 in "Computer-Assisted Instruction and Intelligent Tutoring Systems: Shared Goals and Complementary Approaches", J.H. Larkin and R.W. Chabay (Eds.), L. Erlbaum Associates.

Mittal, S., Bobrow, D.G., de Kleer, J. (1988) *DARN: Toward a Community Memory for Diagnosis and Repair Tasks*, Ch. 4 in: J. A. Hendler, Ed., Expert Systems: The User Interface, Ablex.

Wielinga, B. J. et al. (1992) *KADS: A Modelling Approach to Knowlegde Engineering*, Knowledge Acquisition, 4(1).