7

A two-phased development shell for learning environments: a design proposal

A. Arruarte
I. Fernández-Castro

University of the Basque Country
Donostia
Spain

ABSTRACT

This paper describes the design and architecture of a shell for building Intelligent Tutoring Environments in a wide range of domains. The proposed tool is based on the generic architecture INTZA, previously developed and tested, and is able to adapt and modify that generic architecture according to the requirements of the tutor. In this paper we mainly focus on the teaching-learning domain and propose a generic modelling environment which incorporates both content and pedagogical aspects. At *Concrete Level* the domain Basic Learning Units and their relationships are defined; *at Pedagogical Level* Requisite Relationships, Instructional Objectives and Support Information are defined.

Main conference themes: artificial intelligence, tutoring
Educational areas:
Study topics:
Secondary keywords: authoring systems, computer assisted instruction, knowledge based, tools

J. D. Tinsley et al. (eds.), *World Conference on Computers in Education VI*
© Springer Science+Business Media Dordrecht 1995

INTRODUCTION

During the last two decades Intelligent Tutoring Systems (ITS), a particular area of the Artificial Intelligence, have been studied in depth and a consensus has been reached on its basic architecture [1] of four components: domain representation, student model, pedagogic or didactic model, and interface.

Although most of the built ITSs are focused on concrete domains, nowadays there is an interest for developing a general ITS architecture valid for a wide range of domains. TUTOR[2] and INTZA [3] were developed by our research group with this aim. TUTOR deals with conceptual domains, while INTZA is able to work with both conceptual and procedural domains. The architecture of INTZA therefore generalizes the architecture of TUTOR extending the type of domains it can manage.

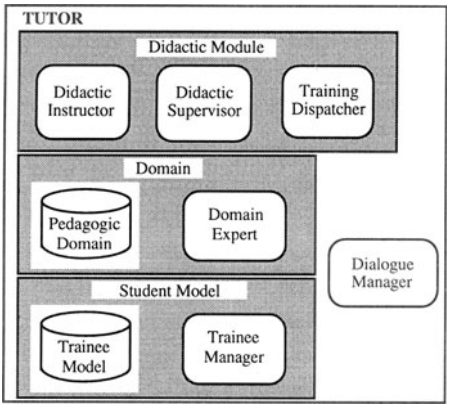

Fig. 1 Generic architecture of INTZA (adapted from [5])

Figure 1 shows the basic components of INTZA corresponding directly to the basic architecture.

Domain

The domain is composed of both Pedagogic Domain and Domain Expert. Pedagogic Domain contains the subject matter to be trained (procedures and malfunctions) organized from a pedagogical point of view. Domain Expert contains the expert's abilities for operating procedures and for detecting and solving malfunctions. It is used to provide the trainee with expert operations as well as to compare trainee and expert performances in order to identify differences showing deep errors.

Student model
The student model groups together Trainee Model and Trainee Manager. Trainee Model records the long term characteristics of the user together with acquired knowledge and skills. Trainee Manager analyzes, treats and evaluates the trainee's interactions updating the Trainee Model in consequence of results.

Pedagogic component
This component generates, replans and carries out the instructional plan for the session. It is structured in several cooperating submodules: a) Didactic Instructor generates the Instructional Plan for the session deciding which instructional objectives should be reached and which instructional strategies should be applied during the training process; b) Training Dispatcher carries out the Instructional Plan; c) Didactic Supervisor decides how to treat the new conditions in the session caused by the trainee's interactions and how to integrate this treatment in the Instructional Plan.

Interface or dialogue manager
Here the communication process is handled between the Tutor and the outside (trainee and human instructor).

With this generic architecture as a starting point our goal is to build a set of tools for developing intelligent teaching-learning systems (Fig. 2). We aim to let teachers, not experts in the computational field, build teaching systems for those domains in which they are experts. The human instructor will establish the characteristics or requirements of the resulting tutor which will be used to adapt the architecture of INTZA and to produce this tutor. Thus the resulting tutor shares the working scheme of INTZA which is generating, executing and replanning the Instructional Plan and which maintains the representation structure and the reasoning scheme with changes only in the content of the components. The tools proposed in this paper have the following functionality:

- data acquisition related to characteristics of the tutor system, to the semantics of the teaching domain and to the final users of the tutor;
- determining the resulting tutor architecture on the basis of the requirements specified;
- generating the final tutor system.

Before introducing our proposal in the third section we will describe some aspects related to the domain representation according to different instructional theories.

THE TEACHING DOMAIN IN INSTRUCTIONAL THEORY

Instructional design theories are primarily concerned with prescribing optimal methods of instruction to bring about desired changes in student knowledge and skills; on the other hand these also specify what must be learned [4] and some way to represent knowledge. These aspects give rise to three basic parts in ITS design: the learning units or teaching contents, the relationships between the elements to be taught, and the instructional objectives or skills to be reached. Next we will illustrate these aspects with some examples which together cover a broad spectrum of tutor characteristics: TUTOR working in a programming language domain, INTZA working in a physical power plant domain and WHY [5] working in a meteorological domain.

Basic learning units

Different instructional design theories and learning techniques based on the 'Component Display Theory, (CDT)' [6] present only facts, concepts, procedures and principles as the Basic Learning Units (BLUs). From this perspective it is possible to determine a complete schema of knowledge representation organized from three different views:

- Conceptually, when a conceptual structure (taxonomy of parts or types) is used to organize the concepts and the facts, e.g. in the TUTOR system.
- Procedurally, when a procedure based structure is used for this organization, e.g. in the INTZA system.
- Theoretically, when a structure based on principles or theories is used for its organization, e.g. in the WHY system.

Basic relationships between elements

In order to establish a pedagogical view useful for selecting and sequencing content Reigeluth [7] identifies four different kinds of relationships between teaching contents of the same type:

- Requisite relationships, e.g. 'The learner must know X (or must be able to do X) before learning Y (or be able to do Y)'. These appear in the TUTOR and INTZA systems.
- Conceptual relationships, e.g. 'X is Y-type', 'X is part of Y'. These appear in the TUTOR and INTZA systems.
- Procedural relationships. These can be order relationships, e.g. 'The learner must do X before doing Y' (INTZA) or decision relationships, e.g. 'Given a condition A, the learner must do X rather than Y or Z'.
- Theory or principle based relationships. These can be cause-effect relationships, e.g. 'Y is the effect of X' (WHY), or prescriptive relationships,

e.g. 'In order to get Z is necessary that X and Y happen in that concrete order'.

In particular a genetic graph has been one of the most widely used techniques for representing the relationships between concepts [8].

Instructional objectives

Instructional Objectives (IOs) refer to the application of particular skills over BLUs. These form a useful part both in planning the teaching process and in creating procedures to assess the learner's knowledge. The objectives can be hierarchically organized in order to establish a logical sequence of didactic activities. The most accepted taxonomic classifications in the psycho-educational field have been the taxonomy of the teaching objectives [9] and the taxonomy of the learning objectives[10].

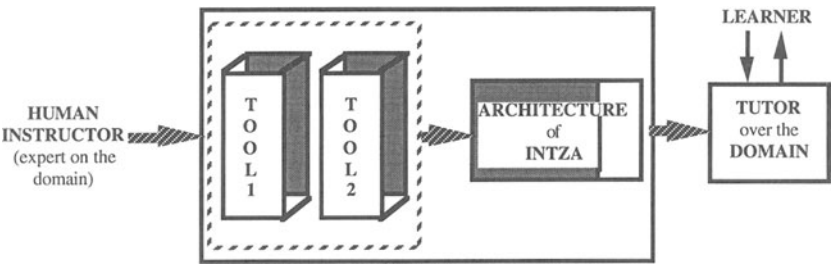

Fig. 2 Information flow in the building of an ITS

The former identifies three different learning categories: cognoscitive, affective and psychomotive. Inside the cognoscitive category six IOs have been defined: knowledge, comprehension, application, analysis, synthesis and evaluation. Different tutors such as INTZA[3] and PEPE [11] organize their objectives following this taxonomy. In particular the INTZA system uses the objectives knowledge, application and analysis. The latter identifies six categories of learning: intellectual skills, cognitive strategies, verbal information, motorial skills and attitudes; only the first three are valid for acquiring static knowledge and problem solving abilities.

Taking these aspects into account we distinguish two different levels in the domain representation (Fig. 3):

Concrete Level: focused on how to represent the basic contents or associated elements of the teaching-learning process;

Pedagogical Level: focused both on the representation of the skills to be mastered by the student and the relationships between these elements in order to get an effective teaching-learning process.

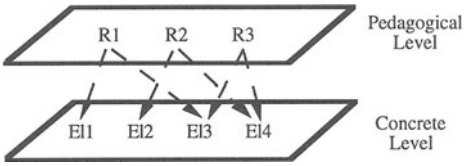

Fig. 3 Different levels in the domain representation

SUPPORTING TOOLS FOR BUILDING ITSs. A DESIGN PROPOSAL.

Our main goal is to propose a development shell to build learning environments for a wide range of domains, adapted both to particular teaching-learning domains and to specific learner characteristics.
Building such a tutor based on the generic architecture INTZA has two phases (Fig. 4):

- In Phase 1 the requirements are used to adapt the generic architecture resulting in a skeletal architecture of the final tutor. The requirements (Fig. 5) are focused on three basic aspects: the characterization of the tutor, the description of both the Concrete Level and the Pedagogical Level of the teaching-learning domain and the learner characterization. These aspects are not independent from each other.
- In Phase 2 the contents of each module established in Phase 1 is defined.

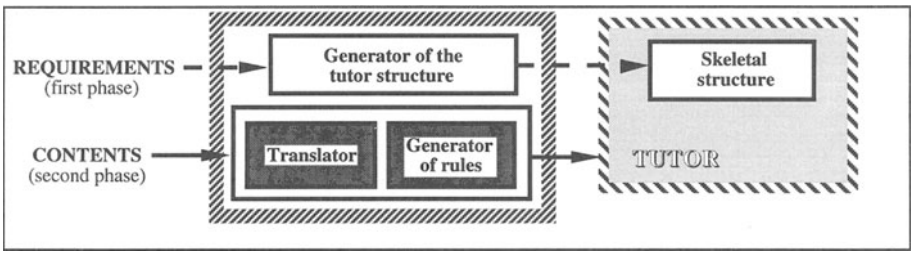

Fig. 4 Phases in building a tutor

Specification of the tutor requirements

To define the basic and domain independent characteristics of the Tutor (Fig. 5) we have to specify the Instructional Method(s) and the Motivation Resources. The former is related to the kinds of interactions between the final tutor and the student, the latter defines the teaching domain independent activities/strategies which the tutor could use in order to maintain the students' attention and interest during the learning process. In our first prototype we have identified impact (audio-visual) resources and messages, although these will be extended in the future.

Specification of student requirements

Depending on the user of the system several groups of strategies may be used. These cover the general learning characteristics (kinds of learners) and the learner goals (Fig. 5).

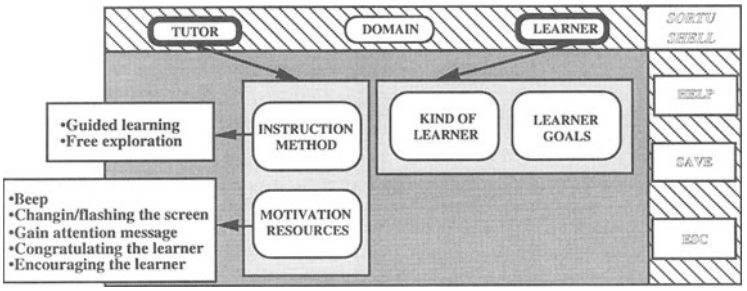

Fig. 5 Specification of the tutor and the learner requirements

The former are interesting for the possible instructional decisions and will be also used to generate most of the Student Model attributes. The latter define the learner's interactions which the resulting tutor will be able to identify and deal with falling into two groups: control of the session or the BLUs defined in the teaching domain.

Domain specification

The specification of the domain (Fig. 6) requires different classes of information:

Meta-information for characterizing the domain in a general way: Two attributes have been included: Domain General Goal and Domain Relevance. The human instructor will select one or both; however it is possible to include new slots with more information (bibliographical references, etc.).

Specification of the domain BLUs: Each BLU must be described at Concrete and Pedagogical levels. Our tool allows the instructor to represent any domain in terms of the already referenced BLUs: concepts, procedures, principles and facts. For example, the photography domain can be represented using mainly concepts and procedures. Fig. 6 shows the specification based on conclusions obtained from different instructional theories (second section), of the concept BLU in the photography domain; the slots selected are those underlined.

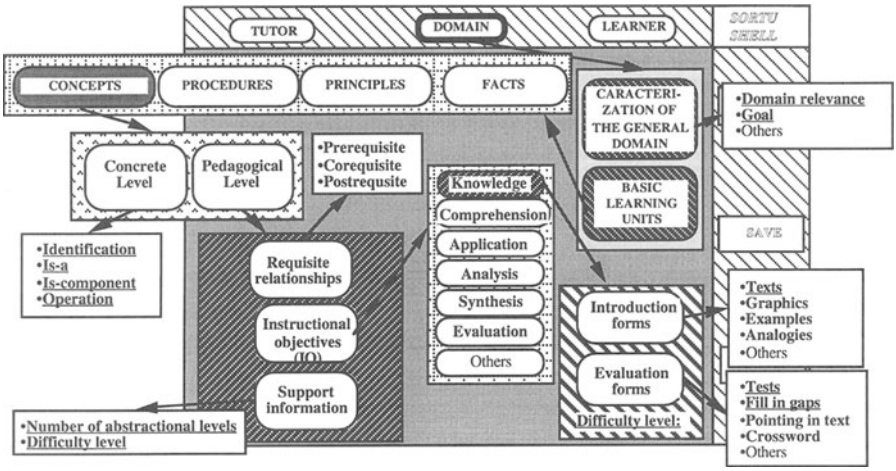

Fig. 6 Concept BLU specification of the teaching-learning domain

At Concrete Level we represent exclusively aspects of content. At Pedagogical Level a pedagogical organization of the domain is specified including descriptors such as Requisite Relationships, IOs and Support Information. The slots selected for each BLU in the first phase make up its corresponding class; in the second phase the BLU instances are defined giving values to the previously established slots. The default relationships defined are prerequisite, corequisite and postrequisite.

The IOs in the prototype shell are those of the widely accepted Bloom's taxonomy [9]. Instructors may define their own IOs. In our photography domain example we have chosen the IO Knowledge. This objective has different slots associated with it:

- Instruction forms or techniques which can be used to introduce the domain concepts to the learner (in the example: text);
- Evaluation forms or techniques for assessing domain concepts (in the example: tests and fill in gaps);

- Difficulty Level which is a refinement of the difficulty level slot associated to the corresponding BLU.

Finally at Pedagogical Level Support Information will be specified on the basis of:

- Abstraction Levels defining the same BLU at different complexity levels;
- Difficulty Level.

Below we show the class of concept BLU resulting from the specification phase, it will be filled in the second phase with domain concepts:

CONCEPT
Identification:
Is-a:
Is-component:
Operation:
Number of abstraction levels:
Difficulty level:
IOs: KNOWLEDGE
Texts:
Tests:
Fill in gaps:

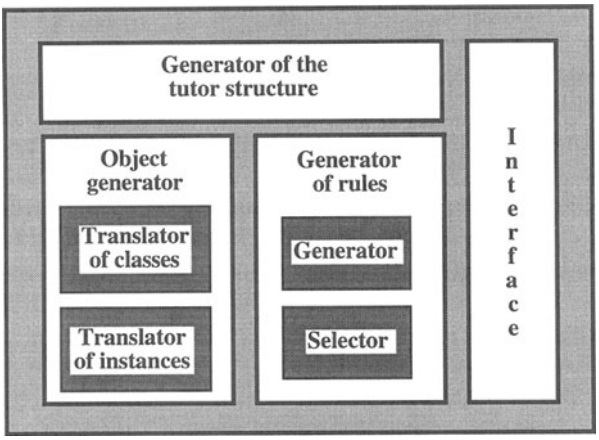

Fig. 7 The basic architecture with four components

A basic architecture

As mentioned before the proposed tools work in two different phases which can run in parallel: in the first phase the tutor structure is generated on the basis of requirements and in the second the defined modules are filled with content.

The basic architecture therefore includes the next four components (Fig. 7):

Generator of the Tutor Structure

- It will adapt the general architecture of INTZA taking into account the particularities of the domain and the teaching style specified by the instructor. Thus it generates the skeletal architecture. Selection of a specific instruction method has consequences for the Didactic Instructor subcomponent. If we want to build a Guided Learning based tutor rules for content selection must be specified, and for ordering and refinement of IOs. If we want to build a Free Exploration based tutor the content selection rules will not be necessary.
- It will check creation preconditions and completeness of the specified contents in each phase. For example no information related to the teaching domain may be introduced before specifying the new tutor's instructional method.

Generator of Rules

After the skeletal architecture of the tutor is produced this component generates and/or selects sets of rules necessary in order to get an operational component. It is composed of a Selector and a Generator. If we select default values in the first phase the Selector submodule will be the activated; if, on the contrary, we introduced new slots the Generator will be activated instead.

For example, if we only would select two IOs (Knowledge and Application) from the six IOs which the tool offers by default, the sets of IO's ordering rules and refinement rules (in the Didactic Instructor subcomponent of the Didactic Module) will be composed only of rules related to those two IOs. If we define new IOs it will be necessary to generate the corresponding ordering and refinement rules. The Generator can do this using some minimal criteria.

Object Generator

This component generates and saves the knowledge associated to each BLU (its classes and instances). It will create the domain objects corresponding either to a new class (in the specification phase) or to new instances in second phase. At the moment objects are generated in the CLOS language.

This module has two subcomponents:

- Translator of classes *generates classes of objects related to the different kinds of BLU considered in the domain. For each BLU it creates an object with its corresponding slots.*
- *Translator of instances uses the classes previously generated and the corresponding slot values to create the instances pertinent to the pedagogic domain.*

Interface or Communication component
This component maintains the interaction process with the human instructor. The communication process is carried out using several screens already shown in the previous subsection. A first prototype of the interface has been already implemented using MOTIF based on C language.

CONCLUSION

To both extend the use and facilitate the building of ITSs [12] we are developing a set of generic tools which can be applied to a wide range of domains. Our goal is a tool based on the generic architecture of INTZA which can be adapted to different requirements specified human instructors.

In this paper we have focused mainly on domain aspects, separating concrete and pedagogical aspects. The Concrete Level includes Basic Learning Units (BLUs) and their relationships. The Pedagogical Level deals with the pedagogical organization of knowledge. It includes different relationships between BLUs, support information and Instructional Objectives (IOs) associated to each BLU.

We have proposed a basic architecture for the development shell focusing mainly on aspects related to the domain representation. We have implemented the Concrete Level for the domain of photography. At present we are developing the Pedagogic Domain and implementing a first prototype of the over-all tool.

ACKNOWLEDGEMENT

This work has been partly financed by the Economy Department of the Foral Diputation of Gipuzkoa.

REFERENCES

1. Wenger, E. (1987) *Artificial Intelligence and Tutoring System.* Morgan Kaufmann.

2. Fernández-Castro, I., Díaz-Ilarraza, A. and Verdejo, F. (1993) *Architectural and Planning Issues in Intelligent Tutoring Systems.* Journal of Artificial Intelligence in Education, **4** (4) pp. 357-395.

3. Gutiérrez, J. (1994) *INTZA: un Sistema Tutor Inteligente para Entrenamiento en Entornos Industriales.* Phd Thesis, University of the Basque Country, Spain.

4. Scandura, J. M. (1983) Instructional Strategies Based on the Structural Learning Theory in Instructional-Design Theories and Models: An overview of their current status. (ed. Reigeluth, C. M.). Lawrence Erlbaum Associates, pp. 213-246.

5. Stevens, A., Collins, A. and Goldin, S. (1982) *Misconception in students' understanding in Intelligent Tutoring Systems* (eds. Sleeman, D. and Brown, J.), Academic Press, pp. 13-24.

6. Merril, M. D. (1983) *Component Display Theory in Instructional-Design Theories and Models: An overview of their current status* (ed. Reigeluth, C. M.). Lawrence Erlbaum Associates, pp. 279-333.

7. Reigeluth, C. M., Merrill, M. D. and Bunderson, V. (1978) *The Structure of Subject Matter Content and Its Instructinal Design Implications.* Instructional Science, **7**, pp. 107-126.

8. Goldstein, I. P. (1982) *The Genetic Graph: A representation for the evolution of procedural knowledge,* in Intelligent Tutoring Systems (eds Sleeman, D. and Brown, J. S.), Academic Press.

9. Bloom, B. S., Engelhart, M. D., Murst, E. J., Hill, W. H. and Drathwohl, D. R. (1956) *Taxonomy of Educational Objectives: The Cognitive Domain.* Longmans.

10. Gagné, R. M., Briggs, L. J. and Wager, W. W. (1974) *Principles of Instructional Design.* Holt, Rinehart and Winston.

11. Bretch, B. (1990) *Determining the Focus of instruction.* Phd Dissertation. University of Saskatchewan, Canada.

12. Woolf, B. (1992) *Towards a Computational Model of Tutoring,* in Adaptative Learning Environments: Foundations and Frontiers. (eds. Jones, M. and Winne, P.). Nato ASI Series.