Université de Montréal

An Intelligent Help System to Support Teachers to Author Learning Sessions in Decision-Making in Network Design

par

Arnoldo Rodríguez

Département d'Informatique et Recherche Opérationnelle

Faculté des Arts et Sciences

Thèse présentée à la Faculté des études supérieures
en vue de l'obtention du grade de Philosophiae Doctor (Ph.D.)
en Informatique

January, 2006

Université de Montréal
Faculté des études supérieures

Cette thèse intitulée :

An Intelligent Help System to Support Teachers
to Author Learning Sessions in Decision-Making

présentée par :

Arnoldo Rodríguez

a été évaluée par un jury composé des personnes suivantes :

Claude Frasson, Ph.D., président-rapporteur
Esma Aïmeur, Ph.D., directeur de recherche
Felisa J. Vázquez-Abad, Ph.D., co-directeur
Yann-Gaël Guéhéneuc, Ph.D., membre du jury
Ramón Brena, Ph.D., examinateur externe
Jacques Viens, Ph.D., représentant du doyen de la FES

Résumé

Cette recherche adresse deux problèmes auxquels les professeurs d'université font face lorsqu'ils sont amenés à créer du matériel pédagogique pour leurs cours de « prise de dècision », et propose un assistant Web du nom de ARIALE (pour *Authoring Resources for Implementing Adaptive Learning Environments*) afin d'aider les enseignants durant ce processus de création.

Les deux problèmes mis en avant dans cette recherche sont :

1. Les enseignants ne disposent pas d'assez de temps pour créer du matériel pédagogique. En particulier, des chercheurs ont pu observé que les enseignants doivent en général fournir beaucoup de temps et d'efforts afin de préparer le matériel d'enseignement utilisé pour appuyer leurs cours.

2. Les enseignants n'ont souvent pas le temps nécessaire pour apprendre à utiliser les outils de développement qui pourraient leur permettre de gagner du temps lors de la création de matériel pédagogique.

La question fondamentale à laquelle cette recherche tente de répondre est comment peut-on rendre la phase de création de matériel pédagogique plus facile pour les enseignants lorsque le domaine d'enseignement considéré est celui de la prise de décision pour la conception des réseaux des ordinateurs?

La solution que nous avons développée pour répondre à ce problème se nomme ARIALE. Il s'agit d'un assistant Web qui se compose d'un site Internet proposant un outil de développement pouvant être utilisé pour créer des sessions d'apprentissage et d'un système d'aide intelligent facilitant l'utilisation de cet outil de développement. Ce dernier dispose de deux fonctionnalités principales : d'une part générer, tester et recommander des exemples et des sessions d'apprentissage aux enseignants et d'autre part offrir une aide adaptative et sensible au contexte afin de maîtriser l'utilisation de l'outil.

La plus importante contribution de notre recherche a été la conception et la mise au point d'une aide en ligne intelligente pour les outils de développement des Hypermédias Adaptives Éducatives (HAE). Le fruit de notre recherche est une outil qui aide les professeurs à développer et à éditer leur matériel pédagogique. Plus spécifiquement, notre outil introduit un *modèle probabiliste* pouvant être utilisé pour la *génération automatique* de topologies de réseaux des ordinateurs et pour la *recommandation* de topologies sélectionnées à partir d'une base de cas. Cette génération automatique et cette recommandation ont pour but d'alléger l'effort fourni par le professeur lors de la création de matériaux pédagogiques. Une autre innovation est l'utilisation d'une aide en ligne adaptative qui est sensible au contexte et qui a pour but de réduire la *charge cognitive* comparativement à des techniques d'aide plus traditionnelles. Dans cette thèse, nous avons aussi développé une méthode permettant de construire des systèmes d'aide pour des outils de développement. La recommandation et l'adaptation de l'aide se font sur la base de données stockées dans le profil de l'enseignant, permettant ainsi à notre système de prendre ses décisions. Au fur et à mesure que les enseignants enrichissent le système en créant des sessions d'apprentissage, en ajoutant des exemples de topologies, en recevant des recommandations et en accédant à l'aide en ligne, ARIALE apprend à partir de ces nouvelles données et améliore au fur et à mesure des enseignants la qualité de l'aide qu'il peut fournir. Notre recherche a aussi eu pour but de faciliter l'accès aux enseignants aux nouvelles technologies de l'information et la communication appliquées à l'éducation.

Mots clés : génération automatique de cas, système de recommandation, apprentissage machine, raisonnement à base de cas, aide intelligente, aide adaptative, aide en ligne, hypermédias adaptatifs, interaction Personne-Machine.

Abstract

This research focuses on the application of principles and techniques of artificial intelligence for the *generation, testing* and *recommendation* of teaching materials, and for providing *Web-based context-sensitive help* to teachers engaged in authoring Educational Adaptive Hypermedia (EAH). We study how to apply Machine Learning (ML) techniques to support teachers in authoring learning sessions. This investigation also studies the use of data related to teachers to support the recommendation of teaching materials and the adaptation of Web-based help.

Our research also pays attention to the teacher's problems in authoring learning sessions for teaching decision-making in network design. Specifically, this research addresses two problems that university teachers face when they are authoring teaching materials for their courses: lack of time to create teaching materials and lack of time to learn how to use the authoring tools that reduce the time required for creating materials. Regarding these problems, this investigation answers the following questions:

1. What can be the general structure and functionality of an assistant to support teachers authoring learning session in decision-making?
2. Which specific functionalities and characteristics of an authoring tool can allow implementer teachers to adapt teaching material according to their pedagogical goals?
3. How can the assistant generate and recommend examples to support teaching decision-making?
4. How does the assistant make decisions about which kind of help content to show, and which media to use for displaying the content?
5. How does the assistant learn to help teachers in a personalized or customized way?

These questions are intended to address and solve the problem of time needed to create adapted case studies for teaching decision-making in network design. Another goal is to reduce the time required to learn the use of an authoring tool to create teaching materials. Consequently, our main idea is to help teachers use a tool rather than teach how to use it. The solution that we created is a Web-based assistant, ARIALE (Authoring Resources for Implementing Adaptive Learning Environments), that supports teachers during the authoring process. ARIALE, is made up of an authoring tool for the creation of learning

sessions to teach network design, and an intelligent help system to support the use of the authoring tool. This intelligent help system has two main functions:

- Generating, testing and recommending examples and learning session to teachers.
- Offering adaptive context-sensitive help about how to use the authoring tool.

We consider that the recommendation of learning sessions and examples, the automatic generation of examples, and the learning of the teacher's decisions related to teaching style and help, are the most important aspects that make our research different, innovative and advantageous in comparison with other studies developed previously. In addition, ARIALE provides context adaptive Web-based help according to the teacher's experience.

Our system uses a probabilistic recommender supported by techniques of artificial intelligence, such as classification learning (Bayesian classifier) and Case-Based Reasoning (CBR), to reduce the complexity of finding an appropriate learning session or example for a particular teacher. The recommendation in ARIALE is also based on the automatic generation of examples according to the teacher's preferences, on the addition and reuse of existing examples in a case base, and on the learning of the teacher's decisions related to which examples to use.

ARIALE keeps, classifies and uses data related to each teacher's attributes, learning and experience using examples. Decisions about how to provide recommendations and adapt help are based on teacher's data stored in a Teacher Model, and ARIALE learns from these decisions to improve future support to teachers. Data related to used examples models each teacher's teaching style. Each style evolves according examples changes. This view of the Teacher Model is new and helpful, because each teacher must follow a static Pedagogical Model in a classical EAH; instead, our Teacher Model is a knowledge base that allows our system to make more flexible and vary the Pedagogical Model.

Our idea of providing more personalized problem-solving support is a new one and is a step forward to support more intelligent collaboration between teachers, beyond the simple sharing of examples between them.

Keywords : Automatic case generation, Recommender System, Machine Learning, Case Base Reasoning, Intelligent Help, Adaptive Help, Web-based Help, Adaptive Hypermedia, Human-Computer Interaction

Table of Contents

List of tables

List of figures

A mis mejores amigos

Acknowledgments

I want to thank the members of the evaluation committee Claude Frasson, Ph.D., Esma Aimeur, Ph.D., Felisa J. Vazquez-Abad, Ph.D., Yann-Gaël Guéhéneuc, Ph.D., Jacques Viens, Ph.D., and Ramón Brena, Ph.D., external examiner, for their observations and suggestions.

This research was funded by the Universidad de Costa Rica, the National Council for Scientific and Technological Research of Costa Rica, and the Ministry of Sciences and Technology of Costa Rica. This work was also partially supported by National Sciences and Engineering Research Council of Canada (NSERC), the Fonds Québécois de la Recherche sur la Nature et les Technologies (FCAR), and Valorisation Recherche Québec (VRQ) of the governments of Canada and Quebec.

Chapter 1. Introduction

1.1. Research Topic and Scope

This research looks at problems related to the application of Artificial Intelligence (AI) in Human-Computer Interaction (HCI) for Educational Adaptive Hypermedia (EAH). We study how to apply Machine Learning (ML) techniques to support teachers in authoring learning sessions. A learning session is a sequence of tasks that the students will perform to acquire expertise in a field.

In this investigation, principles and techniques of artificial intelligence are applied to the *generation, testing* and *recommendation* of teaching materials, as a way to providing *Web-based context-sensitive help*. We also study the use of data related to teachers to support the recommendation of teaching materials and the adaptation of Web-based help.

1.1.1 Subject of application

The subject of application of our study is teaching decision-making skills in network design.

According to Turban and Aronson (1998), decision-making is a process in which the decision-maker defines the problem to be solved by means of a decision (intelligence phase); he also defines the options to chose and the criteria to evaluate these options (design phase). As well, he evaluates the options and selects one or more that solve the initial problem (choice phase). In this research we focus on a particular type of decision-making that is Multi-Criteria Decision-Making (MCDM). "MCDM can be defined as the study of methods and procedures by which concerns about multiple conflicting criteria can be formally incorporated into the management planning process" (MCDM, 2003).

According to Addie (2003), "Network Design is the discipline of choosing the components out of which a network should be built, both the types of components and how many of each is needed, and how they should be put together". For practical purposes, in our research Network Design is defined as choosing one or more concentrators, defining links between the nodes of the network, and selecting the type of links for each link. According to this vision, Network Design is seen as a decision-making process for which the student has to develop the expertise required to master.

1.1.2 Teacher´s interest

Our research focuses on the teacher's interests in authoring learning sessions that integrate the theory and practice involved in the decision-making process in higher education. We do not study aspects related to students and their interaction with EAH. More specifically, we see the subject of teaching network design as a decision-making process that takes into account multiple criteria such as network reliability, cost, and capacity to define an appropriate topology for a given network.

1.2. The Problem

This research addresses two problems that university teachers face when they are authoring teaching materials for their courses:

1. Teachers do not have enough time to create teaching materials.
2. Teachers do not have enough time to learn how to use the tools that could reduce the time required for the creation of materials.

1.2.1. Time

Usually, teachers do not have enough time to create materials for their courses. Particularly, researchers are concerned because teachers must spend a lot of time and effort preparing *Computer-Based Teaching* (CBT) or using *Web-based technology* (WBT) to enrich and support their courses (Nkambou and Laporte, 2001; Johnson, 2001).

Intelligent Tutoring Systems (ITS) and Educational Adaptive Hypermedia (EAH) are options to support teaching, but their creation is time consuming. An ITS is a system that uses artificial intelligence to provide customized learning and supports students in their learning (Shiri-Ahmadabadi, 1999). EAH systems are based on the hypertext structure, and they use information of each student to adapt learning to his needs (Brusilovsky, 2001). Authoring an ITS or an EAH still involves manipulating potentially large amounts of data interconnected in highly complex ways (Williams, 2001). Developing an ITS is a time-intensive task. For example, preparing a two hour class requires three or four working days for a non-programmer novice teacher, or ten hours for an advanced teacher (Murray, 1999).

1.2.2. Complexity of systems

In general, teachers do not have enough time to learn how to use authoring tools for the creation of their courses. "Authoring tools are tools that an author, in a wide meaning, would use for the production of texts, presentations, lectures, and intelligent tutoring systems" (Johansson, 2001). They can not deal with difficult interfaces or spend a lot of time in training. In addition, many of the current applications of the *new information and communication technology* (NICT) in education, such as multimedia and the Internet, are difficult for students and teachers to use (Edwards and Clear, 2001), and require training for effective use (Gay, 1999). While some tools, applications and information systems are easier for programmers or advanced users, most tools are difficult for most people. More capable software is not necessarily easier to use (Lieberman and Selker, 2000). For example, most authoring tools, which are required to create an ITS, remain difficult to learn and use (Frasson, 1998). Although educational Web sites are spread widely and there are many environments in which to create Web-based courseware, some of these authoring tools, for example, Tangow (Carro, 2001) and WebCT (WebCT, 2002), lack easy-to-use help facilities.

Studies about university teachers report that the amount of time required in learning to use new technologies is the second problem they face after general equipment failures (Butler and Sellbom, 2002). Some authoring tools include different types of help such as wizards, tutorials, and context-sensitive online help, but teachers ask for more personalized support on technical and pedagogical aspects (Weiss-Lambrou and Raymond, 2002). A survey showed that 57% of teachers, who do not use this type of new technology, claim technical support is not adapted to the reality of university teaching (Karsenti, 2004).

Personalized help

However, teachers may face problems personalizing help according to their needs. For example, if a teacher had to adapt the help that she requires manually, she would have to perform many tasks. Figure 1.1. shows an example of a menu she might face to adapt help to use an authoring tool. This menu includes seven different options related to the context, the complexity, language, media, bandwidth, help technique and display mechanism.

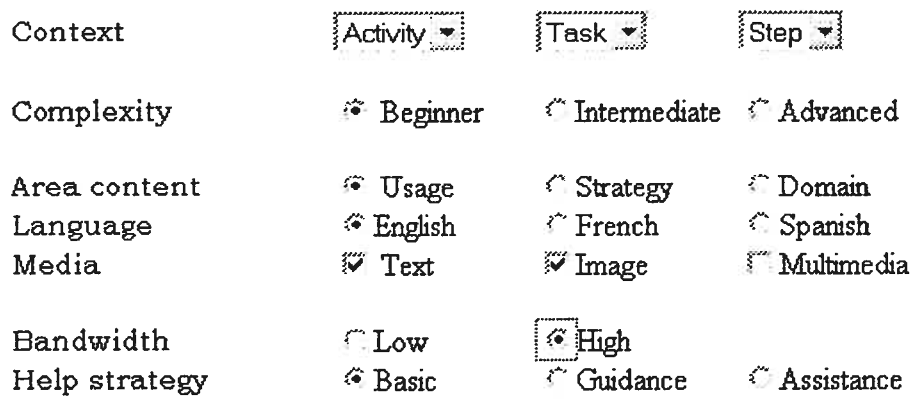


Figure 1.1. Possible menu to set up the help for a teacher.

This menu with multiple options could become more complex if we added the option to navigate between the current and previous help messages displayed. For example, we could add buttons or links to go backward or forward to check these messages. A teacher might also want to check help from the general public level, and this other type of navigation between the help levels increases the complexity of the decisions. Of course, an option to turn off the help can further increase the number of options.

Finding specific help to perform a task is another problem that teachers may face while authoring a learning session. For example, a teacher can be looking for help related to the task she is performing currently, but the appropriate help can be embedded in a huge document and, then, getting the correct help becomes a time consuming task.

Manipulation more than learning

Research into human-computer interaction finds that general users do not really read the help provided; as they are too eager to start the real work. They only want to know how to do a task even tough they do not know how to access the tools or when to use them, and they are not aware of how to combine, adapt, and modify available tools and functions according to their specific needs (Capobianco, 2002; Wang, 2001; Encarnaçao, 1997). In addition, just because users can access equipment, telecommunications, and software, it does not mean they know how to use software. If they do not know how to use software, they will be excluded (technological exclusion) from this infrastructure (Syme *et al.*, 2003; Selbach *et al.*, 2003). This situation provokes a kind of *digital divide*.

Learning to use the authoring tool

In many cases, the teacher's goal is not the same as the explicit task that the authoring tool was designed for. A descriptive example of this type of problem is analysed by (Guzdial, 1999). It is clear that the teacher's main goal is to create materials for teaching; rather than learn how to use the authoring tool. A teacher is an expert whose goal is not learning to use a tool, but rather to achieve an objective with the help of the tool.

As Draper points out, a teacher "has to accomplish some goals, and learning is only a side-effect" (Draper, 1999). Many teachers are attempting this kind of training while they are impatient and under pressure because they have goals different to learning to use software.

Gap between designers and implementer teachers

Another problem is the existing gap between designers of CBT, WBT, EAH or ITS and the teachers who use these tools for teaching courses. While designers create computer-based educational environments, it is the teacher who adapts these environments for teaching courses according to their particular interests. Kinshuk establishes that there is a difference between an implementer teacher and a designer (Kinshuk, 2003). Usually, a teacher who wants to adapt a system to his needs (*implementer teacher*) can not change any parameter of the system.

Scenario of authoring a learning session

The scenario of a teacher authoring a learning session could be very complex because he must pay attention to teaching, which is the main task, but, at the same time, he must know how to manipulate the authoring tool, and if she needs help, she would also have to interact with the help. Her mind would be involved in three cognitive processes: teaching, manipulating the authoring tool, and dealing with the help features.

But the picture is far worse. A help system with an index for searching help according to a particular vocabulary or a list of help options can add more headaches to a teacher, because he must know the keywords to find the required help. Thus, implementer teachers authoring learning sessions may well experience a cognitive overload when learning how to use an authoring tool. In the context of learning, *cognitive overload* is a situation in which the processing demands required by "the learning task may exceed the processing capacity of the cognitive system" of a teacher (Mayer and Moreno, 2003).

1.3. General Purpose and Objectives

1.3.1. General Objective

The purpose of this research is to identify how a Web-based authoring tool for the creation of learning sessions in an Educational Adaptive Hypermedia (EAH) can better interact with teachers who do not have skills in the New Information and Communication Technology (NICT).

The main question to answer is how to apply appropriate techniques of artificial intelligence for the *generation, testing* and *recommendation* of teaching materials, as a way to providing *Web-based context-sensitive help* to teachers engaged in authoring EAH. We also study the use of data related to teachers to support the recommendation of teaching materials and the adaptation of Web-based help. These previous aspects are intended to address and solve the problem of time needed to create adapted case studies.

In addition, we are interested in an environment in which a teacher can propose to her students to integrate theory and practice in the analysis of different points of view about a decision. Our idea is to translate a particular approach for teaching multi-criteria decision-making into an authoring tool in which different types of support are applied.

The approach behind this authoring tool comes from the Web site BestNet (Vázquez-Abad *et al.*, 2001), which was conceived to improve students' skills on network design, and from the Fritze's pedagogical approach (Fritze, 2003) that bridge theory and practice. The BestNet model breaks down the study of decision-making processes into a series of activities that incorporate game elements to help the student integrate theory and practice in a "learning by doing" process of challenges and competition. We assume that our solutions are oriented to graduate students, who start in a phase of *competence* because they already know basic aspects and merely require an enhancement of their knowledge.

1.3.2. Goals

This research answers the following questions:

- What can be the general structure and functionality of an assistant to support teachers authoring learning session in decision-making?

- Which specific functionalities and characteristics of an authoring tool can allow implementer teachers to adapt teaching material according to their pedagogical goals?
- How can the assistant generate and recommend examples to support teaching decision-making?
- How does the assistant make decisions about which kind of help content to show, and which media to use for displaying the content?
- How does the assistant learn to help teachers in a personalized or customized way?

1.4. Solutions

The most important decision we made to guide our research was to focus on the teachers` problems to provide solutions to these problems. In addition, we determined that a teacher using an authoring tool to create her course materials may or may not learn how to use the tool. Consequently, our main idea was to help teacher use a tool rather than teach how to use it, and the solution that we created is a Web-based assistant, ARIALE (Authoring Resources for Implementing Adaptive Learning Environments), that is made up of:

A Web site with:

- An authoring tool for the creation of learning sessions to teach network design and an intelligent help system.

The intelligent help system has two main functions:

- Generating, testing and recommending examples and learning session to teachers.
- Offering adaptive context-sensitive and Web-based help about how to use the authoring tool.

An example of application

In this research, we use an example of teaching the decision-making process of designing a network to implement and test our ideas. According to our approach (Vázquez-Abad *et al.*, 2001), a teacher uses four main concepts (activities) to teach the Network Design process. The example that we use along this research includes the following activities:

Reliability. In this activity, the goal is to teach the concepts of reliability and availability, as well as the manner in which the choices of links affect overall performance.

Concentrator Location. This activity consists of defining the optimal number and location of the "concentrators" (central processors storing a database). The goal is to teach how to place the concentrators at minimum cost to fulfill the minimum safety requirements for information backup.

Capacity Allocation. This activity is intended to teach how to find the optimal choice of capacity for each of the links of the given topology. The learning goal is to allocate capacity per link and define the optimal routing that minimises cost while satisfying the bandwidth requirements.

Design of a local area network. In this activity, the teacher creates an example of a network and each student must propose the optimal design of a network that incorporates all the choices described above.

The first three activities correspond to three sub-problems in network design, all of which can be unequivocally stated as mathematical optimisation problems. The fourth activity is targeted at the integration of the methods and the use of heuristics for network design. The interactivity of the site is done entirely through the concept of guided *tasks*. The first three activities also integrate knowledge from different sources, and the last activity allows the integration of the three previous ones, as described in section 4.1.1.

1.5. Contributions

We consider that the recommendation of learning sessions and examples, the automatic generation of examples, and the learning of the teacher's decisions related to teaching style and help, are the most important aspects that make our research different and innovative in comparison with other previous studies. The integration of these aspects into an assistant that uses methods and techniques of artificial intelligence to solve problems related to the support of teaching is an important step in the progress of educational adaptive hypermedia. The assistant is a more complex and difficult solution to support teachers to author their materials. This view is innovative and better because it not only lets teachers share examples of teaching materials but also search, generate, and recommend examples according to the teacher's characteristics. Particularly, our solutions use artificial intelligence techniques as Bayesian classification and Case Base Reasoning (CBR) to learn about teachers preferences. Using classification learning and CBR is an important improvement over previous EAH that did not include teachers′ differences and the evolution of their teaching styles. Chapter five focuses on automatic generation and recommendation of examples, which is the main contribution of this research.

The context-sensitive help provided is adapted to the teacher's learning and accumulated experience. The teacher's attributes, learning and experience conform a Teacher Model that models his teaching styles. This view of the Teacher Model is new and helpful, because classical approaches in EAH suggests that each teacher must follow a static Pedagogical Model; instead, our Teacher Model is a knowledge base that allows our system to flexibilize and variate the Pedagogical Model. This knowledge increases because Bayesian classification and Case-Based Reasoning (CBR) change information in the Teacher Model as each teacher uses new examples or teaching materials. Teacher′s decisions about teaching improve our system performance to recommend and to help.

Regarding support for authoring tools of EAH in decision-making and network design, we did not find authoring tools that integrate intelligent Web-based and context-sensitive help for implementer teachers. This subject has not been studied frequently, and only a few researchers are working in topics indirectly related to Web-based intelligent help and supporting teachers. In the past years, research on EAH and ITS focused on the Students Model and static Pedagogical Models but pioneer efforts related to the Teacher Model are arising (Kinshuk, 2003; Virvou and Mondriduo, 2002).

Using the classical ITS structure, which is composed of a Domain Model, an Expert Model, a Communication Model, a Pedagogical Model, and, a User Model, we have improved some aspects of the last three models to get a better application for Web-based adaptive and intelligent help. The next section offers a summary of the main contributions of this research for those models.

Communication Model

Recommending learning sessions and examples, as part of the problem-solving support delivered by our system ARIALE, is the most important contribution of this research related to the Communication Model.

We are using a recommender that reduces the complexity of finding an appropriate learning session or example for a particular teacher. Our system uses a probabilistic recommender supported by a Bayesian classifier. The recommendation in ARIALE is based on the automatic generation of examples, on the reuse of existing examples in a case base, and on the learning of the teacher's decisions related to which examples to use. ARIALE keeps, classifies and use data related to the examples used by each teacher. The teacher's data is stored in the Teacher Model, discussed later. The use of this type of classification is an advantage for the generalization of our approach and the learning in ARIALE. Bayesian classifiers are not commonly used as part of the generation and testing of questions, examples, and other components of Intelligent Tutoring Systems and Educational Adaptive Hypermedia. These aspects make our research different because nobody used classification learning to reduce the time and the complexity of finding and delivering personalized support to teachers. Simple sharing of examples is an old type of collaboration between teachers, but providing more personalized examples (problem-solving support) makes collaboration between teachers more intelligent than simple sharing of teaching materials.

Furthermore, to generate examples and test them according to teacher's preferences by means of classification learning (Bayesian classification) is a innovative manner to support teachers. The system ARIALE can generate new examples according to the teacher's preferences and add each new example to the case-base. This *automatic generation of complex examples,* such as network topologies, is not commonly used by ITS and EAH. This generation is also based on a probabilistic method that provides a great variety of examples.

Another change in our Communication Model is providing context adaptive help according to the teacher's progress and experience using the help included in our system. This help is not only adapted to the context content but also to the teacher's progress authoring learning sessions. ARIALE also stores data related to the help that each teacher receives and this data is the basis to personalize Web-based help according to each teacher's degree of progress accessing the help. The objective of this personalization is to decrease the teacher s' cognitive overload accessing, finding and adapting help according to his needs.

An additional contribution is the creation of a tool for teaching decision-making in Network Design. In this research, teaching decision-making is seeing as a progressive process that allows students to assimilate a set of basic decision-making patterns to master more complex skills in decision-making. This process was translated into a Web-based flexible on-line authoring tool that allows teachers to customize the specific tasks by creating, sharing, generating, and editing learning sessions. The authoring tool divides the curriculum of any course into the main concepts that the students will learn in terms of actions and performances. Each concept maps to an activity and each activity will allow three specific tasks to be done. These three tasks implement the pedagogical strategies used for teaching in our Pedagogical Model (detailed later): analysis by comparison, sensitivity analysis and optimization. The different tasks are accomplished by following a sequence of steps and actions (Rodríguez *et al.*, 2003).

This type of support and the recommendation of examples are parts of an intelligent help system that helps the implementer teacher while he is authoring learning sessions. More specifically, when teachers are using the authoring tool to create an interactive learning process in decision-making, an adaptive hypermedia intelligent help system can help them. The system interacts with the implementer teacher to help and exchange information. Information about the interaction between the teacher and the system allows ARIALE to learn how a teacher uses examples to teach and how he uses help, thus enabling the system to customize the support to be provided. In the case of Web-based help, context-sensitive help adapted to the teacher's characteristics is a useful principle for designing the support required by teachers.

In addition, we found that there are not enough studies on traditional help for desk applications and about the help required for Web-based authoring tools. In this research, we present an initial and partial analysis of traditional help and Web-based help as an attempt to address the problems related to the latter.

Pedagogical Model

A Pedagogical Model models the teaching process, for example, what aspects should be presented to the learner, when to review, when to present a new topic, and which topic to present. A classic approach in ITS and EAH consists in using a unique Pedagogical Model oriented to students that each teacher must follow. Our main contribution in this area is a more flexible Pedagogical Model oriented to students and teachers. This Pedagogical Model is hybrid because it includes how to support each teacher authoring learning sessions (Rodríguez *et al.*, 2003; Rodríguez *et al.*, 2004a; Rodríguez *et al.*, 2004b), and this model also contains how to teach a particular subject. Our solution also provides an authoring tool and resources (Vázquez-Abad *et al.*, 2003) to create learning sessions. Chapter four explains our Pedagogical Model with more detail.

The most important difference is that this Pedagogical Model allows the system to support teachers, according to each teacher's experience authoring learning sessions and using support. Information related to each teacher's experience and received help is stored in the Teacher Model, which is discussed later.

Our Pedagogical Model applies an on-the-job-training approach to support teachers while they are using the authoring tool for the creation of learning sessions. The on-the-job-training approach is intended to supply support while the teacher is authoring instead of offering training before authoring. Thus, we have considered "on-the-job training" to support the development of on-line help, especially that of Web-based adaptive and intelligent help systems to support authoring tools. We consider that "on-the-job training" is more appropriate for Web-based help than classical methods used in ITS, which include previous training, pre-test, post-test, and other techniques used to teach and evaluate the learning of a tool (Aïmeur and Bassil, 1999). The reason is that teachers are not intended to master specific software or authoring tools to achieve their pedagogical objectives. Then, previous training and tests should be reduced to a minimum. Seeing the teacher as an author who needs on-the-job training to use an authoring tool is a new view of the teacher's role.

Another aspect is that our system ARIALE is not designed for teaching decision-making with emphasis in numerical methods that could be very specific for particular subjects. This could leave out some teachers whose teaching styles are based on the use of numerical methods, but our approach is to support teaching based on qualitative approaches, expert knowledge, and heuristics. This Pedagogical Model is also limited to this research because pedagogy is neither the focus of our project nor the central issue of this thesis. There are many pedagogical models in the literature, but in our approach we only focused on one of them in order to implement and test our system.

Adapting the Pedagogical Model according to each teacher's teaching style involves processing data stored in the Teacher Model, as explained in the next section.

Teacher Model

Usually, an EAH has many teachers who implement specific learning sessions following the designer's Pedagogical Model. The focus of this research is the teacher who implements a course and our main contribution related to the Teacher Model is the use of an evolving *teaching style* that allows the system to adapt instruction for students according to each teacher's pedagogical goals. A teacher can follow the Pedagogical Model but adapting it by using particular examples and goals. The examples and pedagogical goals that a teacher uses to teach a subject are his teaching style. This teaching style is not static because it evolves as the teacher's examples changes. The system stores data related to these changes and doing this the systems learns and improves its ability to deliver future support. Chapter four details the Teacher Model.

Our Teacher Model allows the system to adjust to teachers' pedagogical objectives in a specific domain: decision-making. Using the authoring tool, a teacher can adapt contents according to his requirements, limitations, and pedagogical goals, and such changes are included in his Teacher Model. The system keeps data about the teacher's experience, stores statistics about his performance in authoring learning activities and about his experience with help. Consequently, our research includes a Teacher Model that keeps data about the teaching and learning styles of the teacher. For example, by means of the Teacher Model, we can classify teachers in three basic categories according to their ability and experience in teaching and using the system: *beginner*, *intermediate*, and *advanced* users.

In this research, the teacher is an *implementer teacher* (Kinshuk, 2003), who is an expert in teaching decision-making skills. He has to solve a problem: create materials for his courses. To create these materials, he has a plan and a specific task to perform in a short term. In addition, he has experience in teaching and using authoring tools, but his goal is not to become an expert about a specific authoring tool.

Research on intelligent tutoring systems attempts to create a perfect teacher rather than a teacher's tool (Kinshuk *et al.*, 2001). But, since an EAH cannot replace all the functions of a human teacher, we focus on an *adaptive help* as a tool for enhancing the role of the human teacher in the authoring process of an EAH. In our system, a teacher is an implementer of a course who needs to be helped not only in the different pedagogical strategies involved in teaching decision-making skills, but also in the usage of an authoring tool for creating EAH.

An interesting method to help teachers is the "user as a student" strategy (Brusilovsky and Schwartz, 1997), which allows us to see teachers as students who must be evaluated about their progress using a tool. However, teachers´ main goal is to create materials for teaching. This is their most important objective, rather than learning to use the authoring tool to be evaluated (Guzdial, 1999). Consequently, we have adapted the view of the teacher as learner of new technologies to that of a trainee who requires adaptive help and support to solve problems such as creating case studies. In this scenario, the teacher's learning becomes a by-product of the support that he receives. This is a fundamental characteristic in our research because we do not see the teacher as a typical student. Rather, we see the teacher as a worker who can learn while he is on the job, applying the on-the-job-training methods, as explained in the previous section devoted to the Pedagogical Model.

1.6. Organization of the Dissertation

This Introduction describes the scope of the research, the problems studied, the theoretical questions that we formulated, our approach, solutions and the contributions of our investigation. The second chapter includes the state of the art about *pedagogical aspects*, *teacher modeling,* and the role of artificial intelligence in *human-computer interaction*. Chapter three describes related studies that are close to our research, and compares it to relevant theories, concepts and antecedents that we have applied in this research. In chapter four, we describe the theoretical basis of our research: the Pedagogical Model and the Teacher Model. We model the characteristics of a human teacher with emphasis on his teaching style and analyse the application of the Teacher Model in our system ARIALE (Authoring Resources for Implementing Adaptive Learning Environments). Chapter five describes an innovative probabilistic method for the automatic generation of examples and for the recommendation of examples and sessions. This chapter also discusses other characteristics of the help provided. We pay attention to aspects intended to reduce cognitive overload problems, such as the excess of help not related to the task on hands, which are often caused by traditional help techniques. We describe the methods that our system ARIALE uses to adapt help to the teacher's attributes and plan. Chapter six describes the implementation of the system, its architecture, general layout, general functionality, and main processes. This chapter also details the method used for building the intelligent help system and shows a general scenario of interaction between a teacher and our system. Finally, chapter seven discusses the results of the evaluation of ARIALE, and includes the conclusion of this research and future ways to continue investigation into this subject.

Chapter 2. Background literature and previous work

Summary: This chapter summarizes theories and concepts that are fundamental in this research. Section 2.1. includes information about *pedagogical aspects* and *teacher modeling*. We explain the approach that we apply to guide teaching and structure our authoring tool, taking into account the students as the intended public. In addition, we detail concepts that we use to support teachers while they are authoring teaching materials. Complementary, some concepts used to model the human teacher are also explained. In section 2.2., we describe some aspects of artificial intelligence related to *human-computer interaction*, which are applied in our system ARIALE (Authoring Resources for Implementing Adaptive Learning Environments). Section 2.3. compares and discusses relevant theories and concepts that we applied in this research.

2.1. Pedagogical aspects

In this section we describe concepts related to pedagogical aspects and teacher modeling. Those concepts are the basis for the organization of our authoring tool and the support to be provided to teachers. Many concepts are related to teaching a subject, helping a teacher and modeling the characteristics of a human teacher.

2.1.1. Teaching approach

In this section we explain the pedagogical approach for teaching decision-making that is on the basis of our work. This part is related to important concepts about how to teach the subject to graduate students and how to support teachers while they are creating teaching materials.

Our research is inspired by the BestNet project, a Web site that attemptes to complement traditional teaching with a series of web-based activities. These activities provide virtual practice and the integration of concepts for a course on Network Design (Vázquez-Abad *et al.*, 2001).

The objective of BestNet is to integrate traditional teaching, computer-assisted methodologies, online resources and qualitative approaches based on expert knowledge and intuition, to help students visualise effectively how all the design choices integrate. This approach is also intended to allow the teachers to integrate theory and practice by "applying understanding to real-world problems, 'authentic' and ill-defined tasks" by means of simulated environments, role games or problem-based learning (Fritze, 2003).

As Vázquez-Abad *et al.* (2001) expressed it: "BestNet follows an original concept: it is designed as a series of virtual competitions and as a complement to traditional course material". The BestNet model requires the students to assume the roles of "players" engaged in a competition to optimize the design of a bank network. Two other strategies that complement competition are analysis by comparison and analysis of sensitivity of cases. As part of a sequence of three tasks, *comparison* and *modification* of cases are the two initial tasks that converge into the third task: *optimization*. These strategies are repeated in different units throughout a course to provide a virtual environment where students can gradually improve their expertise in making decisions about how to design a network.

2.1.2. Decision-making

In this research we focus on a particular type of decision-making that is Multi-Criteria Decision-Making (MCDM), which concerns multiple conflicting criteria that are the basis for a decision. A criterion is the set of elements (objective, goal and attribute) that the decision-maker takes into account to evaluate each option. For example, an attribute can be the *cost of a network*, an objective can be to *reduce the cost* of the network, and a goal is to *reduce specifically 15%* of the cost. The most important aspect is that two or more attributes can be conflicting; for example, an objective can be to reduce the network cost without decreasing the network reliability[1], but the reduction of the cost can also impoverish the network reliability; as a result, the solution requires finding a network design with a tradeoff between the conflicting attributes.

However, "any complex decisional problem requires explicitly considering several points of view. The classical approaches (utility theory, game theory, benefit-cost analysis, mathematical programming, etc.) consider only one objective function to be maximized or minimized; hence, they do not ensure a faithful modelisation of real world decision problems" (Gerad, 2003).

---

[1] In this research we define reliability as the probability that all nodes of the network are connected by operational links.

2.1.3. Expertise Development

The BestNet approach to develop the student's expertise in decision-making is also supported by Cognitive Load Theory (CLT) (Sweller, 1988), which is proposed as a promising tool for developing expertise (Rickers *et al.*, 2004). In order for students to develop expertise in decision-making, CLT proposes providing problems, with an appropriate difficulty level, in a repetitive process that includes feedback, the opportunity to correct errors and, gradually, problems with an increasing level of difficulty.

The starting point in CLT is *cognitive overload*, a situation that a user may experience when the processing demands required by a task exceed his processing capacity (Mayer and Moreno, 2003). For example, while a worker is trying to calculate the average price of some products, at the same time he has to learn how to use a spreadsheet application (software) to calculate the average. Doing the two tasks simultaneously, she must switch between the spreadsheet and the documentation that explains how to use the software to compute the average. She must carry information from one environment to the other quickly, because she does not have more than 18 seconds to keep information in mind (Peterson and Peterson, 1959). In addition, her main goal is to compute the average, not learning to use the software.

According to CLT, learning is the product of a process in which the learner uses his limited working memory efficiently. *Working memory* or *short-term memory* is the part of the human cognitive process "in which all conscious cognitive processing occurs". This immediate memory limits the amount of information that a person can receive, process and remember around seven (seven plus minus two) chunks of information at a time (Miller, 1956). Moreover, a human being can deal with a small number of *interactive* elements, "possibly no more than two or three" (Pass *et al.*, 2003). This means that a teaching process should free the student's working memory, allowing him to learn chunks of knowledge in an iterative process (Moreno, 2004). After the learner assimilates a part of the knowledge, he frees up his working memory to acquire more complex aspects and so on. As a result, step-by-step and chunk-by-chunk, the learner acquires a big chunk of knowledge, which is located in the *long-term memory*. Long-term memory stores vast quantities of *schemas* or patterns that integrate multiple elements of information related to a specific function. A schema organizes information according to the manner in which the person is going to reuse it (Gerjets and Scheiter, 2003).

CLT in decision-making follows the same strategy used by long-term memory: a learner acquires a simple pattern used for making decisions and, iteratively, he learns more complex patterns until achieving a high level of expertise. Finally, he can retrieve a complex high-level schema from the long-term memory by putting lower level schemas in the working memory to process situations that require expert skills. For example, a person can learn how to make decisions about designing a network with a minimum degree of network reliability; after that he can learn more about the location of concentrators to support an acceptable degree of network reliability; in a third activity, this person can learn how the assignation of capacity to different links increase or decrease the network cost. Once he has accumulated knowledge about network reliability, concentrator location and capacity allocation, he will apply these three aspects together in the design of a network, a more complex activity that integrates the previous three.

Working memory has a limited capacity to deal with novel information, thus Cognitive Load Theory also recommends the use of worked-out examples to reduce the cognitive load that learners may experience while they are learning new concepts (Pass *et al.*, 2004). "*Learning by doing*" with worked-out examples, the learner can develop the skills to recognize patterns in the provided examples. In this way, the student develops the expertise to retrieve schemas from his long-term memory to compare them with the examples and identify differences and similarities. This view is close to Case-Based Reasoning (CBR), an artificial intelligence technique explained in section 4.5, and to the *case studies* method widely used in teaching business administration (DeLacey and Leonard, 2002). A case study is an example or a set of examples that a teacher proposes to the students to analyze or change. The objective is that the students grasp the rules behind the case study. For example, analyzing two examples of network designs, the students can detect the differences between their measures of performance and the relationships between those measures and the configurations of topologies.

2.1.4. Educational strategies

The Cognitive Load Theory provides an appropriate way for a learner to improve his knowledge and skills to pass from novice to expert. This theory lacks implementation details that are, however, supplied in part by the Payr's model of teaching and learning (Payr, 2003).

According to Payr (2003), the teaching/learning process supports the learner to pass through the five phases of skill development, from the "novice" phase to the "expertise" phase. She defines five phases that a learner can pass through as he progresses: *novice*, *beginner*, *competence*, *proficiency* and *expertise*. In each phase, the learner performs some activities intended to achieve some skills. These types of activities are: *receive* and *remember; imitate* and *apply; select* and *decide; understand* and *design; recognize* and *master*. As the learner improves his skills, he also acquires a more personal perspective, greater responsibility, experience and expertise. Table 2.1. shows the five phases and their corresponding activities. In the case of graduate students, we assume they start in the phase of *competence* because they already know basic aspects and merely require an enhancement of their knowledge.

Table 2.1. Phases of the learning process, according to Payr (2003).

	1	2	3	4	5
Phase	**NOVICE**	**BEGINNER**	**COMPETENCE**	**PRO-EFICIENCY**	**EXPERTISE**
Activity	**Receive**	**Imitate**	**Select, decide**	**Understand, design**	**Recognize, master**

In addition, this model defines the corresponding skills that a learner can develop as he passes through the different phases from novice to expertise: skills to deal with free-context situations, skills to act in context-sensitive conditions, skills for problem solving, skills to recognize patterns, and skills for managing complex situations. Thus, this approach is fundamental not only for teaching decision-making skills to students gradually, but also for adapting support to teachers as they progress and accumulate experience in using an authoring tool, for example.

2.1.5. Education and Artificial Intelligence

One of the most promising trends in the research in education and artificial intelligence has been the development of Intelligent Tutoring Systems (ITS) and Educational Adaptive Hypermedia (EAH). An ITS is an expert information system that teaches learners or supports them in their learning. ITSs apply artificial intelligence to interact with learners and to make inferences about the learners' knowledge. Additionally, an ITS is a system that applies artificial intelligence and Computer Based Technologies (CBT) to provide customized learning (Shiri-Ahmadabadi, 1999). EAH systems are based on the hypertext structure, and they create a model of the knowledge and characteristics of each individual user and apply this model to adapt the system to the needs of that user (Brusilovsky, 2001). The main objective of using these types of systems is designing strategies to support an interactive process which evaluates students' reactions and generates the required feedback for each reaction until the proposed pedagogical goal is achieved. A classical architecture of an ITS can have five main models (Beck, 1996): Student Model, Pedagogical Model, Domain Knowledge, Communication Model, and Expert Model.

Student Model

The student model stores information about each individual learner, as well as the material being taught and the student's knowledge. For example, the name, the student's age, his knowledge about a subject and his level of progress studying a topic are data stored in the student model.

Domain Knowledge

This model contains information that is used to teach. In other words, the domain model stores information that the tutor is teaching, for example a definition of the concept bandwidth if the subject is Network Design. This definition can be stored as plain text, a sound file or a video file.

Communication Model

This interface or communication model lets the systems interact with users by presenting the student with the material in the most effective way. For example, a way to present teaching material is through multimedia.

Expert Model

The expert model or teacher model includes information about how to teach the expertise needed to solve problems in the knowledge domain. It is a model of how someone skilled in a particular domain stores such knowledge in his mind. For example, one teacher may prefer to use simple examples and another may prefer case studies that are very complex. In a system, a simple example can be represented as a set of variable-value pairs and a more complex example can include rules to structure the variable-value pairs. The expert model is detailed later in this chapter.

Pedagogical Model

The Pedagogical Model provides a model of the teaching process, which is adapted to the different needs of each student. Based on the student model, the pedagogical model can make pedagogical decisions about what aspects of the domain knowledge should be presented to the learner, when to review, when to present a new topic, and which topic to present.

Pedagogical Strategies

Pedagogical or teaching strategies refer to the methods of instruction, how the material is presented and which pedagogical acts (formulate, develop, demonstrate a concept, do an exercise, discuss) are included (Crozat, 2000). The term "*pedagogical strategy*" also refers to the instructional strategy, media, content and decisions about learner support and assessments, which allow the system and student to communicate or interact in a more effective way. In the case of an ITS, "tutorial strategies are the set of teaching events (actions and decisions) that motivate and interest the learner while improving his performance" (Abou-Jaoude and Frasson, 1998).

Examples of pedagogical strategies are *learning by disturbing* and *double test learning*. Using learning by disturbing (Aïmeur and Frasson, 1996) is a strategy that uses a troublemaker that occasionally misleads the student to test his grade of confidence. Double test learning is a strategy in which a simulated classmate receives the same training as the learner, then an intelligent tutor system tests the classmate first and the learner can check the classmate's mistakes. After that the tutor tests the learner who benefits from the classmate's mistakes (Aïmeur and Bassil, 1999). Another pedagogical strategy is *discovery learning*, a strategy where the learner can construct his own knowledge by interacting with a domain to infer rules from the results of the experiments (Van Joolingen, 1999). *Learning by doing* is a very similar pedagogical strategy that allows the learners to be involved in hands-on experience manipulating simulated devices or following procedures. A pedagogical strategy that is useful for teaching decision-making skills is learning *by problem solving* within simulated environments with increased feedback effort. This strategy shows a number of examples and counter-examples of an idea. Learners will notice similarities and differences among these examples and then will grasp the "rule" (Copernicus, 1996).

2.1.6. A model for the teacher

In the previous sections we have described some concepts related to the Pedagogical Model behind this research. We focus on the concepts related to the Teacher Model. The original idea of BestNet (Vázquez-Abad *et al.*, 2001) was to give the teacher the option of tailoring the case studies. A teacher should be able to define his pedagogical goals for the activities that he proposes to the students. For example, specifying some criteria to be applied in a particular type of analysis that his students must develop. This view is also supported by the concepts *human teacher model* (Kinshuk *et al.*, 2001) and local *implementer teacher* (Kinshuk, 2003). These concepts refer to the teacher as a person with particular characteristics and preferences, who wants to adjust or adapt an existing ITS or EAH according to his interests and expertise. The expertise is personal and it varies from one teacher to another one (Grandbastien, 1999); as a consequence, an ITS or EAH needs to keep teachers' profiles to allow adaptation to needs, limits, and pedagogical goals of different teachers. This personalization requires using a Teacher Model that stores data such as the language used by the teacher, his skills performing a particular task and personal information (name, age, identification).

A human teacher model

As defined in a section above, a Teacher Model is the information about each human teacher and his personal manner to teach a subject. In many cases, this type of information is included in the Pedagogical Model, but we are considering the Teacher Model as a specialization of the user model. The user model is a set of information describing a particular teacher who is using a system, and this user model is used in determining "how to present data, what type of help to give, and how the user interacts with the interface" of the system (Keolle, 2002; Kobsa, 2001). A human teacher model includes personal characteristics of each teacher, information about how he teaches and, in our system, information about how he acquires more knowledge and experience using an authoring tool. For example, a human Teacher Model keeps the teacher's name, his age, the course he teaches, the type of examples that he prefers to use, the learning sessions that he has authored and the help received. Another common specialization of the user model is the Student Model. The student model stores personal characteristics of each individual learner and tracks of his progress and knowledge learned. This research does not pay attention to the Student Model because our focus is the teacher.

An implementer teacher

The concept of local *implementer teacher* (Kinshuk, 2003) involves the teaching styles and personality attributes of a human teacher. A Pedagogical Model is a unique and static way that a particular teacher applies to teach a subject, and a teacher using an ITS or a EAH must follow that way. However, many teachers using that style can need to adapt the Pedagogical Model to their particular interests and styles to implement a course.

But this vision is not enough because some times a teacher can be seen as a user who is a student who must learn how to use software (Brusilovsky and Schwarz, 1997). For example, a teacher using a specific EAH for teaching decision-making in network design can need to learn to use EAH, its attributes and functions. Other visions are more flexible to see the current situation of a teacher as a challenge in which we see the *teacher as a trainee* (Vázquez-Abad *et al.*, 2003). Viewing a teacher as a trainee or a *teacher as learner* also entails that "teachers must see themselves as learners as they grapple with new technologies, changing roles and understanding the requirements of their students" (Fritze, 2003). As a consequence, modelling a teacher also implies to store information about the teacher's preferences, his activities and interests (frequent sessions, exercises), his level of expertise (Virvou and Mondriduo, 2002), his goals and plans, his experience and skills (Encarnação, 1997), his teaching style, his prior knowledge about the platform, his discipline subject to teach, his learning ability, and the capabilities of the computer hardware and software he is using (Patel *et al.*, 1998). For example, an implementer teacher can have some learning sessions in use and some additional ones to be finished; his skills to author tasks to compare examples are high but he does not have any ability in defining a competition between students to find the best solution for a problem. This information is different in comparison with another teacher who uses completely finished learning sessions only and who has very high skills performing only tasks related to the analysis of the sensitivity of an environment.

A system acquires some data mentioned in the previous paragraph when a teacher provides data directly or explicitly, for example when he fills in a form with details about his interest and preferences. Other type of data is acquired implicitly from the interaction between the teacher and the system. Examples of usage data are actions, temporal behaviour, frequency of actions, navigation patterns or environmental data such as software platform and hardware (Fink and Kobsa, 2000).

Populating a Teacher Model

The data for the Teacher Model of a particular teacher can be also acquired using two methods: *stereotype* and *overlay model.* Often, some teacher's information is not available at the beginning, and, to overcome this problem, the available teacher's data can be compared with other teachers' data to reuse or copy their data (Encarnação, 1997). This is the first method, stereotype. For example, if a new teacher has the same characteristics of teachers in the "beginners" category, missing data can be taken from teachers belonging to this category. Stereotype is a powerful method because it can be based on few observations to assign the teacher to a group of teachers (Tsiriga and Virvou, 2002). The other method is the overlay model, which represents the knowledge of an individual teacher as a record of the knowledge that a user has of the domain concepts. For example, for each concept of the domain there is a counter of the number of times that the teacher consulted this concept (Encarnação, 1997). The difference between stereotype and overlay model is that a stereotype compares a teacher´s characteristics to other teacher´s characteristics, and an overlay model stores data related to a particular teacher´s knowledge.

Many methods can be used to model the teachers' characteristics, especially for the initialization of a Teacher Model, such as rules of production, Case-Based Reasoning (CBR) or classification. Classification maps teacher's data into one of several predetermined classes. "In the Web domain classes usually represent different user profiles and classification is performed using selected features that describe each user's category. The most common classification algorithms are decision trees, naïve Bayesian classifier, neural networks, k-nearest neighbour and so on" (Eirinaki and Vazirgianis, 2003). For example, using a naïve Bayesian classifier or a k-nearest neighbour algorithm, a particular teacher's data can be classified into a category of teachers whose data is very similar to his.

Any stereotype requires initial teacher's information, such as his name, age, number of times he has taught a course and used Web-based technologies. These data are compared to other teachers' data, and then he is classified in a category of teachers who share characteristics that are similar to his. For example, a teacher can fill some forms with his personal information and data that say that he has taught a subject six times and he has been using Web-based technologies for three years. If he did not give more information, lacking data will be supplied with data from other teachers who have taught the same subject a similar number of times and have a similar number of years using Web-based technologies.

In the case of the overlay model, a system acquires data from its interaction with the user, in implicit and explicit ways. For example, using a dialogue box, a teacher provides explicit information about his preference using or rejecting a recommendation. If a teacher accesses a help topic, this event is implicit information included as a part of his overlay model.

Information acquired by means of a stereotype or an overlay model becomes part of the Teacher Model. Once a Teacher Model is built, its information can be used to recognizing the teacher's plans and provide help according these plans, for example. This application of the Teacher Model is detailed in chapter five. Data stored in the Teacher Model also allows the system to identify the teacher's interest and preferences, store and update new information or provide information that the system needs to make decisions (Johnson and Taatgen, 2003). These decisions can be related to different system functions, but in the next sections we pay attention to concepts referred to the application of artificial intelligence to support the interactions between an EAH and teachers who create teaching materials.

2.2. Artificial Intelligence and Human-Computer Interaction

A challenge of help systems is providing appropriate support to help users in the right way (Fischer, 2001). To achieve these goals, research in *Human-Computer Interaction* (HCI) and *Artificial Intelligence* (AI) integrates efforts in the creation of knowledge-based user interfaces and help systems. In this section, we describe concepts that are applied in our research and that come from these two areas.

2.2.1. User interfaces

The field of Human Computer Interaction (HCI) involves knowledge from multiple areas particularly psychology, computer science, and social sciences. Working together, experts in these areas are trying to make computing systems that are both useful and usable. As states by Olson and Olson (2003):

> Human-computer interaction (HCI) is the study of how people interact with computer technology. One major area of work in the field focuses on the design of computer systems. The goal is to produce software and hardware that are useful, usable and aesthetically pleasing.

People interact with information systems by employing a user interface that lets them exchange information. In this context, the concept of "*interface*" can not be seen with restrictions, and it could mean any way that the user uses to exchange information with a system, including the documentation (Redmond-Pyle and Moore, 1995; Boy, 1998; Raskin, 2000). In the words of van Vliet: "*the interface is the system*" (Vliet, 2000). In addition, dealing with interface design is very complex because there are many fields involved (Preece, 1993), such as cognitive psychology, linguistics, design, and computer science.

Interfaces for authors

Regarding the problem of supporting the use of Computer-based Teachnology, Web-based Technology, the developing of Intelligent Tutoring Systems (ITS) or the creation of teaching material, our initial point of view is that ITSs, Educational Adaptive Hypermedia (EAH), and other authoring tools should contain *facilities* for teaching (Warren, 1996), that are not only *useful but also usable*. Many EAHs should have guidance to support teachers in selecting pedagogical strategies or "translating" educational contents to Web pages. Intelligent help would also be very useful in planning a course or in designing online interactive learning processes because then teachers could better guide students to progress gradually (Dufresne, 1996). Guidance, help and facilities for teaching require flexible help systems that can adapt to the teacher's requirements.

2.2.2. Adaptive Learning Environments

Adaptive systems are Adaptive Learning Environments (ALE), such as ITS and EAH, that can change their parameters and behaviour to meet a goal. The user would not only be able to adapt the environment to his needs, but the system should also adapt itself to different changes generated by users. Thus, the system should be adaptable and adaptive. According to Stephanidis, adaptive systems would "be accessible by anyone, at anytime and from anywhere" (Stephanidis, 2001). Two general categories of adaptation are:

1- *Adaptability*: users have the possibility to select different types of presentation and interaction among the ones included in the system.

2- *Adaptivity*: the system possesses the capability to monitor user interaction and to generate automatic adaptation that suits different users or contexts of use (Stephanidis, 2001). Adaptivity also means to monitor and record the user's behaviour; to deduce and generate the appropriate response, to change the interface presentation, the help, and the dialogue sequences to guide the users to reach their goals (Wang, 2001). For example, a system monitors the user's decisions when he is looking for help, and according to these decisions, the system changes the help contents from minimum to more complex, and the medium from animation to t ext.

The most common adaptation of a system is the adaptivity to the user's characteristics, such as his goals and tasks, his knowledge background and previous experience. Other characteristics to take into account are the user's individual traits, which are features that define a user as an individual, including his preferences and interests. Adaptation can occur when a system makes changes based on the knowledge stored in a user model. A user model is a set of data referring to a particular user.

There are many ways a system can adapt to different user's actions (Brusilovsky, 2001), but there are also three *types of adaptation* to something other than user characteristics, namely adaptation to *user data*, to *usage data*, and to *environment data* (Kobsa *et al.*, 2001). For Brusilovsky, hypermedia ITS must provide adaptive presentation (*content level* adaptation) and adaptive navigation (*link level* adaptation) (Brusilovsky, 2000). Adaptive presentation is subdivided into text adaptation and multimedia adaptation technologies; adaptive navigation support is subdivided into link hiding, sorting, annotation, direct guidance, and hypertext map adaptation.

Context-based adaptation

In Web-based systems, adaptation to user's environments is necessary because contextual information about user location and user platform can improve the adaptivity. A goal of the adaptation of the system is to minimize input explicitly provided by the user, and adaptation to the user's work context is a way to do it. *Context-aware* and *context-sensitive* systems can adapt easily and without more complex interfaces (Lieberman and Selker, 2000).

A specific sort of adaptive function in hypermedia systems is the personalized presentation of multimedia information according to specific user interests and preferences (Maybury, 2000; Light and Maybury, 2002). Personalization and, more specifically, *Web personalization* takes into account users' data, usage data, and context data, all of which customize a Web site content and structure according to the needs of specific users (Eirinaki and Vazirgiannis, 2003). For example, users' data can be acquired when teachers fill a registration form with personal information; an overlay model can store usage data related to the help concepts that a teacher has consulted; and, in the case of contextual data, a variable can keep information about the page where the teacher is currently.

2.2.3. Intelligent user interfaces

According to Maybury, intelligent user interfaces are human-machine interfaces aimed to increase the efficiency, effectiveness and naturalness of human-machine interaction (Maybury, 2003). IUI uses knowledge representation and reasoning based on models of users, domains, tasks and media to personalize and improve the interaction between a system and the user.

Intelligent user interfaces are an option to overcome some of the problems of human-computer interaction, such as direct manipulation (Höök, 2000). Direct manipulation is the use of techniques such as pointers, mouse, drag and drop to interact with computer programs and receive immediate feedback (Shneiderman and Maes, 1997). An intelligent interface applies reasoning, knowledge, learning, planning, perception and communication to make decisions on behalf of the user. This type of interface adapts to different users' characteristics to support computer-based tasks (Maes, 1994).

Intelligent user interfaces "integrate an adaptive user interface and an intelligent help system to provide *context-sensitive* and active help" (Encarnação, 1997). An intelligent user interface has different components such as *plan recognition*, *dynamic presentation* that shows data in a clear way, user modelling and *interface adaptivity*. These components allow the IUI to support the user when he is interacting with the system (Encarnação, 1997;

Ehlert, 2003). Usually, an intelligent user interface stores a user model with the history of the user-system interaction; this information allows the intelligent user interface to improve its ability to interact with the user (Langley, 1999).

In many cases, intelligent user interfaces are implemented by means of agents that perform different actions to adapt the system and improve the human-computer interaction.

Agents

One way to improve the usability of pedagogical Web sites can be by using intelligent agents, which help users to better interact with applications (André and Rist, 2002). An agent can be defined as a computer system situated in some environment and capable of autonomous actions to meet its objectives. The agent interacts with an environment, which can be accessible or inaccessible, deterministic or non-deterministic, static or dynamic, discrete or continuous. The agent and its environment interact continuously (Wooldridge, 2002).

An agent can also be seen as a "an assistant or helper, rather than as a tool in the manner of a conventional *direct-manipulation* interface" (Lieberman, 1995). In this shift, we can add that

> An intelligent agent is a computerized entity that can perform a particular task that has been assigned to it. Usually intelligent agents are autonomous so they can perform this task without any explicit instruction from the user that controls them. Another aspect of an intelligent agent is that they are (very often but not always) capable of learning. This *learning capability* allows them to adapt to new situations they encounter (Ehlert, 2003).

According to Russell and Norvig, intelligent and autonomous agents are programs that perceive their environment and act upon this environment without direct human intervention (Russell and Norvig, 1995). Depending on their characteristics, agents can have a high or low degree of autonomy and intelligence. They can be reactive or proactive and they can be distributed in societies of co-operating agents. A particular type of agent is the interface agent, which is a way to implement intelligent user interfaces.

Interface agents

Ehlert (2003) enumerates some characteristics of interface agents:

> An intelligent interface agent, also called software agent or softbot, cooperates with the user to accomplish his task and functions as the user's personal assistant. The agent can take the initiative, rather than passively wait for instructions. It can provide the user with information, or detect and correct the user's misunderstandings.

The best-known examples of interface agents are intelligent tutoring systems and *context-sensitive help systems* (Lieberman, 1995). Context-sensitive help provides information that relates to the specific field of the Web page in which the user is currently working (Self, 1998). By using artificial intelligence techniques, for example rules, a system can adapt help content according to a particular user's characteristics. Interface agents could thus help computer novice users to manipulate information systems, because intelligent interface agents can allow systems to monitor the user's actions, develop models of user abilities, and automatically help out when problems appear. Intelligent interface agents try to solve some of the problems that the current direct-manipulation interfaces cannot, such as "creating personalized systems, dealing with information overflow or filtering problems, providing help on using new and complex programs, taking over tasks from the user, and other forms of interaction (e.g. speech or gestures) or helping people with a disability" (Ehlert, 2003).
One of the most important characteristics of interface agents is the ability to adapt to different types of users. Therefore, these "adaptive interfaces" can adapt the presentation of information and the interaction techniques according to user's profiles (Dufresne and Paquete, 2000; Dufresne, 2001). According to its different functions, an agent can process the user's input to generate a new output in the interface. For example, interface agents can present information by using multimedia presentations or particular sets of media adapted to the user's requirements or to his profile (Maybury, 2000; Light and Maybury, 2002; Wahlster, 2000; André and Rist, 2002).

2.2.4. Intelligent help systems

There is some confusion related to the definition of the two concepts Intelligent User Interfaces (IUI) and Intelligent Systems (IS), and Ehlert warns about the difference between IUI and IS should be taken into account before explaining other concepts that are derived from them. An often-made mistake is to confuse an IUI with an intelligent system. It is important to note that a system exhibiting some form of intelligence is not necessarily an intelligent interface. There are many intelligent systems with very simple non-intelligent interfaces and the fact that a system has an intelligent interface does not say anything about the intelligence of the underlying system (Ehlert, 2003).

Intelligent systems

An *intelligent system* can perform some tasks by means of artificial intelligence techniques. In general terms, an intelligent system chooses actions based on knowledge to achieve a main objective. This knowledge base can be composed by its experience, data provided by an expert, data acquired from distributed databases, data inferred by itself or data encoded as part of the system structure. This kind of system is not entirely passive; it can perceive information from the context and reacts to changing that context. Some characteristics of an intelligent system are the representation of information, the reasoning using the information representation, the ability to learn from its experience and the generation of new information. An intelligent system can help users to deal with information overflow and with complex decision-making processes needed to select appropriate help (Höök, 2000), but it does not imply an improvement of the process the user follows to reach his goals.

Intelligent Help

A more specific means to support users is *intelligent help*. Intelligent help is a sort of adaptive support that the user might need at a particular time, or in a specific situation or context. We use the concept *Intelligent Help Systems* (IHS) as equivalent to *Adaptive Help Systems* (AHS) and according to the definition of Brusilovsky and Schwartz (Brusilovsky and Schwartz, 1997).

Intelligent Help Systems provide personalized, dynamic, and contextual support by using artificial intelligence techniques for adaptation (Lorés *et al.*, 2002). "*Personalized*" help means adapting to the user's individual characteristics, taking into account information about the type of user (such as the user's knowledge). "*Contextualized*" means supporting the user, taking into account the current plan, and the user's context, and the situation in which the user-computer interaction occurs (such as the application's current state) (Wang, 2001; Aberg, 2002). By monitoring the human-computer interaction, a help system can assist users with the functionality of a computer program, or with some other independent task (repairing a plane, for example).

In general terms, an intelligent help system should not disturb the user's interaction; the user should be free to choose his personal interaction style with the system. Operating in real time, intelligent help can watch what the user is doing, and while the user analyzes his next action, the system can perform some other actions (Ehlert, 2003).

Intelligent help can be either *passive* or *active*. Passive help occurs when the user explicitly calls up "help" on a system, for example, in the case of *context-sensitive* help. Active help occurs when the system makes a decision autonomously to support the user or suggest some things that the user can do. An example is when the system detects the user is starting a task and offers to finish the task on behalf of the user (Encarnação, 1997). Of course, in general terms, active help can fail to detect the task in which the user to be supported, can fail to deliver help just-in-time or can fail to provide appropriate help. In addition, this type of help can surprise users, who could feel a loss of control. On the other hand, passive help can be "ineffective, inefficient, and leads to behavioural problems" because users could spend a lot of time looking for information instead of focusing on the task to be performed, or users could avoid accessing any help because they do not want to spend time learning (Leung, 2001).

The interaction between the intelligent help and the user could be richer than described previously. Sometimes, the user can ask for help, but the system proposes or decides on the type of help to provide and executes it. In other cases, the user asks for help, and the system proposes options, but the decision is shared between the user and the system; finally, the system executes. Another option in active mode, the system offers help (based on observations of the user's activities), the user provides feedback, and the system proposes, decides and executes (Encarnação, 1997). This is the case when the intelligent help offers an example to follow, and if the user accepts, the system selects and shows an example adapted to the user's characteristics.

2.2.5. Methods to support users

There are two additional approaches in supporting users: *learning support* and *problem solving support*. The learning support helps the user to extend his knowledge about a subject. Problem-solving support helps users to find the solution to a problem (Aberg, 2002). These types of intelligent help decrease the user's cognitive load when the user has to deal with complex or unknown environments. While some systems only provide one type of support, others offer both.

Problem-solving support

Problem-solving support helps the user in solving a problem. Instead of helping the user to achieve a goal step-by-step, problem-solving support provides the solution to the problem. This situation occurs when a user is not interested in spending time learning how to use a system or how to solve a problem, and prefers to use the application directly to finish his tasks. Aberg explains that:

> It is clear that problem-solving support is more directly relevant for the user's current task, while learning support may be of more importance for the user's long-term performance (Aberg, 2002).

Recommender Systems

Problem-solving support can be implemented by means of a *recommender system*. A recommender system is intended to "learn about a person's needs and then proactively identify and recommend information that meets those needs" (Callan *et al.*, 2003). For example, recommender systems can provide a sorted list of possible solutions to a problem according to the characteristics of the user.

Basically, recommenders use two methods: *content-based filtering* and *collaborative filtering*. Content-based filtering takes into account the individual user's preferences. For example, if a teacher prefers examples of networks with high reliability and medium cost, the system recommends such examples. Collaborative filtering infers that users with similar behaviour can have analogous interests (Eirinaki and Vazirgiannis, 2003). For example, two teachers (A and B) with the same characteristics (age, preferences, background, and skills) can share the same type of examples for teaching. If the two teachers belong to the same group, then the system infers that Teacher A's examples can be reused by Teacher B because both teachers share other common characteristics.

Learning support

The learning support helps the teacher to perform some tasks. According to Encarnação (Encarnação, 1997), there are four types of user support: 1. on-line support tools (help, guidance, on-line manuals); 2. off-line documentation; 3. training; and 4. advice. His research focused on the on-line support tools such as help and guidance. Intelligent help should be able to support the teacher whenever he needs help on any related topic.

To help users, it is possible to apply the *minimalist* approach and *layering* technique used by Selbach, Sieckenius and Barbosa (Selbach *et al.*, 2003), and the techniques of *incremental learning* and *adaptive help* proposed by Brusilovsky (Brusilovsky and Schwartz, 1997).

The minimalist approach means that the users receive small chunks of information to support them while performing a task or learning how to do it. Although the minimalism principle was not created for use in on-line help systems, its "less is more" approach is usable in Web-based applications in which the screen space and the bandwidth do not make it possible to display a lot of information. The system can provide the minimal information and users would then have to interact with the content to improve their abilities.

Incremental learning and layering technique mean that the system adapts to the user as he learns and progresses from beginner to advanced. According to Brusilovsky, it is necessary:

1. To track the user's actions to know the user knowledge
2. To use task models to deduce the goal of the user.

These two activities allow the system to know when a user is a beginner, thereby showing him some subset of a complex interface. As the user progresses, more advanced interface features will be available incrementally. Tracking the user's actions, the system can obtain information in three ways:

- A dialog can be developed between the user and the system
- The user can provide an example of what she wants to the system
- The system can infer new information based on information previously received.

On-line help and Web-based help

Intelligent help is a general concept that covers many options to support users in different activities, but we have focused on the help that users require to perform some tasks to achieve a goal in a job. This is the case of a teacher who needs to perform some tasks to create courseware, for example. A teacher performing a task might need on-line help to complete his job.

On-line help provides brief information to solve a specific problem while the user is working (Lorés *et al.*, 2001) on an application that is not Web-based. This type of help allows the user to search for and receive information about topics related to the application he is using at the very moment he is interacting with the application. On-line help can include on-line manuals, tutorials and wizards (Tsuji and Yamamoto, 2001), and usually the user can search for help related to a specific word or application command (Gallagher and Daigle, 2002).

Electronic Performance Support Systems (EPSS) is the definition of a type of on-line help "which provide integrated, on-demand access to information, advice, learning experiences, and tools to enable a high level of job performance with a minimum of support from other people" (Leung, 2001). This kind of help system is a "*task support system*" that facilitates the user's task completion, for example, to make decisions or diagnoses (Susarla *et al.* 2003).

There are three criteria or principles to provide this kind of on-line support.

- **The first is *just-in-time* support**. With just-in-time support, the system supports the user when he is on the job, at the moment when he needs to develop the competence to perform a task. An advantage of this approach is that the user is not taken away from his work to be trained.
- **The second type is *just-enough* support**. This type means the user is going to receive just enough information to complete the current task. The idea is to avoid cognitive overload for the user provoked by information that is not going to be used. Otherwise, the user can become lost and taken out of his job context.
- Finally, Leung explains that "most users handle their tasks on hand without necessarily optimizing the solution" Leung, 2001). Thus, the ***third idea*** behind task support systems is that integrating *just-in-time support* and *just-enough support* will allow a reduction in the cost of learning a new task. This third idea is called continuous *performance development*. It is assumed that if a user receives just-in-time just-enough support, his skills will improve continuously.

Web-based help

Web-based Help Systems are help systems intended to provide help to Web-based applications. For example, Web-based help could be the help that a Web site provides to a user who is looking for a product or trying to follow a procedure. Web-based help can be seen as a specialization of on-line help for Web-sites or Web-based application connected to Internet or an Intranet. Some Web sites use help links to provide information about procedures, navigation tips, and frequently asked questions (FAQ). Other sites include site maps that show the user's location in the Web site hyperspace, or search facilities to find help content (Welinske, 1998). Self recommends that a Web-based Help System be context-sensitive. This means that the help for a concept in a Web page would be related to this specific Web page context, not to the set of all the Web pages in the site (Self, 1998). Adaptation and context-sensitive help are not common because Web sites usually have a single page with the whole help content to support all the pages in the Web site. As a consequence, a user looking for help related to a particular concept, must check a lot of information, included aspects that are not related to the searched concept. Moreover, the application of artificial intelligence for Web-based help is not a well studied area (Delisle and Moulin, 2002).

2.2.6. Methods and techniques for building on-line help

As shown previously, there are many concepts and principles relating to intelligent help systems, adaptation, and Web-based help; however, the number of methods for the implementation of adaptive interfaces and on-line help is not so extensive. As Rothrock *et al.* (2002) have established, it is necessary to find an interface design methodology for real-time adaptability. Two attempts to provide this kind of methodologies are the *Guide to Adaptive Interface Design* proposed by the previously cited authors (Rothrock *et al.* 2002), and the *Semiotic Method for Building On-line Help Systems* formulated by Selbach *et al.* (2003).

Rothrock *et al.* have proposed three guidelines for adaptive interface design:

1. *Identify variables that call for adaptation.* This step involves the identification and selection of information that the system will take into account to make a decision about the adaptation of the interface. This information includes variables such as the user's performance, plans and tasks to be performed, previous knowledge, preferences, context, and environmental data.
2. *Determine modifications to the interface.* Depending on the variables selected in the previous step, the designer determines how and when the interface should adapt to the calling variables. In addition, the designer decides which changes will be performed by the user and which by the system. The user can participate in the

adaptation depending on the degree of autonomy that the designer assigns to the system.

3. *Select the inference mechanism*. With the variables selected in step one and with the possible transformations determined, it is necessary to select a means to identify instances that call for adaptation and to decide on the appropriate modifications for the interface.

However, adaptation is not synonymous with help system. Adaptation is used to adapt help to the characteristics of the user, the context, and the environment. To develop an intelligent help system, it is necessary to take into account additional aspects such as the possible reactions of the system to the user's inputs, the reasoning method that the system uses, the system learning method, the way to generate new information and the communication process between the user and the system. To take into account some of these aspects, it is also necessary to use complementary methods to design and implement help systems. However, those methods are not common. Particularly, we found only the method proposed by Selbach *et al.* (2003), which focuses on content and offers a sequence of eight steps:

1. Designing help taking into account the possible users' questions.
2. Organizing the help content.
3. Refining the content and indexation.
4. Building a general help module.
5. Linking the help to the interface components.
6. Testing.
7. Evaluating possible difficulties for matching questions and answers during the user-system interaction.
8. Redesigning help content.

The guide and the method cited above are important tools that we apply to design our system, but its implementation requires more specific techniques and resources. Some of them are the principles and techniques used in the classical help of many commercial applications.

2.2.7. Help principles and techniques

Purchase and Worril have studied principles and techniques used in conventional help for many applications (Purchase and Worril, 2002). Although their study focused on off-line help for desktop applications, it offers an important framework for further research on techniques for on-line intelligent help.

According to the Purchase and Worril analysis, the most useful *principles* for designing help are:

- Help should be easy to understand.
- Help should be procedural: step-by-step instructions.
- Help should be unobtrusive. The system should know when the user does not need help.
- Help should be accurate, complete, and consistent, not confusing.
- Help should speak the user's language because incomprehensible help is useless.

These principles are usually mapped to help techniques that are supposed to deliver help content to users in an easy way. The help techniques are the functions or features which any user can access in a word processor application. The most commonly used *help techniques* are (Purchase and Worril, 2002):

- Index. Alphabetical index of words linked to text help information.
- Find. A search mechanism to find information related to a specific word or concept.
- Contents. Hierarchical organization of help information in topics and subtopics.
- Balloon help. Pop up hints that the user can receive when he accesses a concept.
- Hyperlinks. Hypertext and hypermedia links to information.

Purchase and Worrill also studied another type of more common help, namely the FAQ (Frequently Asked Questions). FAQ is a list of questions and answers related to a subject. FAQ is not very common in desktop commercial applications but is frequently used on the Internet to support users of Web sites and Web-based applications. As in the case of Index, Find and Contents help features, FAQ requires the user to know or have a good idea about the key words included in the questions he should consult. The advantage is that FAQ are usually task-based because questions are related to tasks to be performed instead of focusing on isolated functions. Other researchers have also done research on the application of Frequently Asked Questions (FAQ) as a way to provide on-line help (Selbach *et al.*, 2003; Susarla *et al.*, 2003). They conclude that the design of the questions is very important to make them intuitive and clearly related to the referred help topic. That is the reason why Selbach *et al.* focused on the difficulties of matching questions and answers during the user-system interaction.

Gelernter (Gelernter, 1998) refers to other techniques to deliver help such as *Definition*, *Tutorial*, *Quick Reference* and Hint. A Definition is a description of a specific concept. This definition is part of the help content. The Tutorial is a sequence of contents to teach a process. A Quick Reference shows a small set of different help contents related to a concept. A Hint is an advice that is displayed as a dialogue, and the user can accept or reject the hint. Explanation is a technique defined by Brusilovsky and Cooper (Brusilovsky and Cooper, 1999) to deliver explanations related to a concept. By Demonstration Liu (Liu, 2001) understands showing the development of a process.

2.2.8. Learning

Learning includes modification of the representation of facts that this system has (Cortés *et al.*, 1994). An intelligent help system learns when stored knowledge changes to reflect new data (Ehlert, 2003), or when the organization of the knowledge changes. More specifically, a system learns when it changes its knowledge, the organization or representation of its knowledge, and its skills to perform a task better in the future. For example, a system includes new data in its knowledge base. These data are related to examples of network design type A used for teaching a topic under a certain condition C. This change increases the probability of examples type A to occur under a certain condition C in comparison with the probability of other types of examples under this condition C (knowledge representation). As a consequence, the system uses this new probability to prefer examples of type A under condition C because they have a higher probability to occur. Thus, the system improves its ability to prefer a type of example under condition C. This learning is incremental because the system updates the probability of A and compares it to the probability of other types of examples instead of starting from scratch the computation of the probabilities of the entire set of examples (Russell and Norvig, 1995).

Next sections detail some machine learning concepts and artificial intelligence techniques that we apply as the part of problem-solving support developed in this research.

Machine Learning

Machine learning is an area in artificial intelligence that analyses how a system can extract knowledge from examples or experience to change its knowledge base (Taatgent, 1999). Machine learning algorithms increase knowledge or skills of the system to perform a task or to solve a problem in different ways depending on the context and without human intervention. The learning process involves information selection and the adaptation of new information. Given some data and prior knowledge, machine learning allows a system to derive a knowledge representation of a concept (Taatgent, 1999).

Learning can be classified in three different general types: *supervised*, which uses sets of training data; *unsupervised*, which discovers information by analysing data; and, *reinforcement learning*, which depends on the feedback the system receives to add or reject a new knowledge component (Russell and Norvig, 1995).

More specific machine learning methods are *induction*, which generalizes a concept from a set of examples; and, *analysis*, which assumes that a concept can be derived from a given domain theory, such as a set of characteristics of some items in a classification process (Taatgent, 1999). We are mainly interested in two main procedures of machine learning related to induction and analysis:

- Classification learning: learning to put instances into pre-defined classes
- Association learning: learning relationships between the attributes (Keller, 2002a).

These procedures require that a system has a knowledge representation, a set of operations to manipulate these representations, as well as a heuristic search to choose the appropriate knowledge among its representations (Luger, 2002). In the next two sections we discuss more specific methods and mechanisms used to implement machine learning solutions, such as Case-based Reasoning (CBR), an artificial intelligence method used in this research, and two specific techniques of classification learning that use statistical learning and similarity measuring.

Case-Based Reasoning

Case-based Reasoning (CBR) is an attempt to copy the human ability of applying our experience to problem solving. For example, an expert detects a new problem and compares it with previous problems that he solved in the same domain. If he finds a similar problem in his memory, he tries to reuse or adapt the same methods to solve the new case (Cortés *et al.*, 1994). The idea is to allow a system to learn from its own experience. In CBR, learning is a by-product of problem solving.

In CBR, a case is the representation knowledge of an experience by which an expert achieves a goal. The case represents knowledge associated with specific problems. A CBR system has a base of cases or library of old cases. Each case has a structure that includes its characteristics of the problem to be solved and the associated solutions. The characteristics can be divided into *indices* and *complementary information*. An *index* is an attribute that is used as a means of identification of a given case. The cases grouped together in a base have the same type of indices, which serves to speed up the search process. For example, in a base of network examples, the cases can be indexed by their cost. If the system needs to solve a design problem of a new network, it would be possible to find an example in the base with the same cost as the new one, and then the old configuration can be proposed for the new case. Complementary information is not fundamental to finding a case, but it could be important for understanding and using the cases.

CBR includes four phases: *identify* a similar case, *adapt* the solution of this case to solve the new one, *evaluate* if the new solution works, and, if it works, *add* the new case and its solution to a case-base (Luger, 2002). Adding the new example or problem and its solution to the case-base is a type of learning. Moreover, classifying an example in a case-base is a typical exemplar-based approach where the solution to a problem is its right classification (Aamodt and Plaza, 1994).

Matching the new case indices to one or more examples in the case-base requires applying algorithms to compare the indices. First, it is necessary to filter the case-base to retrieve only the most similar cases. Second, the filtered cases must be ranked. One option is to measure the similarity (similarity metric) or distance between the new case and each example in the case-base. In problem-solving support, if it is necessary to recommend a network design example for teaching a topic related to this subject, and the system has a partial example that a teacher is providing, then the system compares the provided example to the cases in the base and retrieves and recommends the most similar case with a network design example.

Measuring similarity

A method to measure similarity is *Instance-Based Learning* (IBL). IBL is a method which classifies an example by taking into account examples previously classified. Each time a new instance has to be classified, the classifier method compares it to the stored examples to assign a value to the new example (Palu, 2004).

The Nearest Neighbour is a well known IBL algorithm that calculates the distance between cases. This algorithm can measure the distance between two cases or it can be used to find a specific number of examples that are close to the new one. When the algorithm looks for the *k* most similar examples, we call this algorithm k- Nearest Neighbour. Palu supplies a short definition of the algorithm:

> The nearest neighbours of an instance are defined in terms of standard Euclidean geometry (distances between points in n-dimensional space). More precisely, an arbitrary instance x can be described by the feature attribute lists: < $a_1(x)$, $a_2(x)$, $a_3(x)$, ..., $a_n(x)$>, where $a_r(x)$ denotes the value of the r^{th} attribute of instance x (Palu, 2004).

Given two instances x_i and x_j, the distance between them can be calculated by the following equation:

$$dist\left(x_i, x_j\right) \equiv \sqrt{\sum_{r=1}^{r=n}\left[a_r\left(x_i\right) - a_r\left(x_j\right)\right]^2} \quad (1)$$

When a system finds a group of the *k* nearest instance, it is possible to classify them according to each value of distance. For example, using the nearest neighbour algorithm we can get ten examples that are very close to our case x_q. After identifying an instance or a set of instances, the system applies the solutions associated with them to the problem in the new case. This phase entails applying a solution directly or adapting it to the new problem. In the latter, the system requires a procedure to adapt and evaluate the solution, or an expert can evaluate the new solution and include it and the new case in the case-base. Otherwise, the solution would be wrong and rejected. In the future, the system will have a new experience or knowledge in its base of cases, given a new case and its solution or a mistake report to avoid that type of example in the future. In any event, the system will have learnt a new example or a counterexample.

Learning examples or counter examples by means of this classification algorithm is fine but using k-Nearest Neighbour algorithm in CBR to retrieve and rank cases has some disadvantages because sorting examples requires more computational resources. Another problem is that this algorithm compares all the characteristics of all the examples in the case-base, and this process is time consuming. But, if only a few of the attributes are taken into account, there is the risk that perhaps the most similar cases may well be left out (Palu, 2004).

Because search performance looking for similar cases is a central problem in artificial intelligence, reducing the search time and complexity is an important goal in this area. Sometimes, examples in the case-base are also classified in different classes or categories because they share a similar solution or attributes (Aamodt and Plaza, 1994). In addition, the search can be more complex if the examples lack complete information about their characteristics, and if the system must deal with some degree of uncertainty.

Probability and uncertainty

This section refers to the use of probability and uncertainty in learning, and the application of probabilities to improve the classification and retrieval of cases in the context of CBR for problem-solving support.

People and machines usually have to deal with limited knowledge and incomplete information for making decisions or predicting events. Probability theory allows us to determine the chances of an event occurring. Statistical correlations are useful to predict events, and especially, to classify components of a set. A system that classifies new examples with incomplete data in a case-base is improving its knowledge (learning) base under uncertain conditions.

A *supervised* type of learning that applies uncertainty, probabilities and statistics is the *Bayesian learning*. This method is based on Bayesian theory, which uses previously known results to calculate more probabilities. According to this theory, an event has unconditioned probability or *prior probability* assigned in the absence of knowledge supporting its occurrence or absence. A conditional probability or posterior probability is the probability of an event given some evidence. Bayesian theory assumes that the different attributes in a class or category are independent (Domingos and Pazzani, 1997). Then, assuming that A and B are independent events, we can say that the probability of an observation A given another observation B is

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{2}$$

and, assuming that $P(A \cap B) = P(B \cap A)$, then we have that

$$P(B|A) = \frac{P(A \cap B)}{P(A)} = \frac{P(B)P(A|B)}{P(A)} \tag{3}$$

Equation (2) is the Bayes Theorem that can also be expressed as:

$$P(A|B) = \frac{P(A|B)P(A)}{P(B)} \tag{4}$$

This equation states:

> the probability that hypothesis A is true given that evidence B is observed, is equal to the product of the probability of B being observed given that A is true and the probability of A being true, divided by the sum of the products of the probabilities that B is observed given that alternative hypotheses are true and the probabilities of these alternative hypotheses (Johnson and Taatgen, 2003).

Assuming that A is a hypothesis to prove and B is given evidence, the calculation of the probability of A is usually more difficult than getting the probability of B. Then, based on (3) we calculate the probability of B given A using the following equation:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} \tag{5}$$

because $P(A)$ is constant, it can be dropped, and then we get

$$P(B|A) = P(A|B)P(B) \tag{6}$$

A naïve Bayes classifier is a method to classify new evidence according to given prior evidence. This classifier assumes that non conditional relationships are among attributes of the evidence, and that is the reason why this type of classifier is called "naive". This type of classifier can classify sets of inputs and learn new inputs to anticipate behaviour or a result (Rothrock *et al.*, 2002). A Bayesian classifier requires the definition of classes and a dataset of training examples previously grouped in these classes. A set of classes can be represented as $V = \{v_1, v_2, \dots . v_j\}$.

Given a new example E with some attributes and a set of old examples in a base of cases divided into classes, we can represent the new example as a vector of attributes $E = \langle a_1, a_2, \dots a_n \rangle$. These attributes have to be compared to the attributes of each example in the case-base. Unlike k-Nearest Neighbour, a naïve Bayesian classifier does not compare each attribute directly. Instead, this classifier takes into account the *prior probability* assigned to the set of examples in the base, and combines it with the observed data of a new instance still to be classified. The result of this combination is a posterior probability assigned to each class.

To arrive at this posterior probability, we start calculating in which class the given example E can be classified, by adapting equation (4):

$$P(v_j|E) = \frac{P(E|v_j)P(v_j)}{P(E)} \tag{7}$$

where:

- v_j is the class about which we want to know if the example can belong.
- E is the example to be classified.
- $P(v_j)$ is the prior probability that the class v_j is the class where the new example E can be classified
- $P(E)$ is the likelihood of the example

Because $E = \langle a_1, a_2, \dots a_n \rangle$, we can express:

$$P(v_j|E) = P(a_1, a_2, \dots a_n|v_j) = \prod_i P(a_i|v_j) \tag{8}$$

Then, by adapting equation (6), it is possible to calculate the probability of each attribute a_i to belong to v_j:

$$P(v_j|E) = P(v_j)\prod_i P(a_i|v_j) \tag{9}$$

to find the most suitable class $v(E)$ with the maximum prediction, using the following equation:

$$v(E) = \arg\max_{v_j \in V} P(v_j)\prod_i P(a_i|v_j) \tag{10}$$

Finally, the class with the maximum likelihood is the class where the new example can be classified (Keller, 2002b).

However, the Bayesian classifier performs poorly when the number of instances with a particular attribute a_i in the training set is very small, or in the worst case when it is zero. In this last case, zero in any attribute multiplied by any attribute will produce zero as the final probability. To solve this problem, we can apply a formula that uses *m*-estimate of probabilities (Meisner, 2003):

$$P(a_i|v_j) = \frac{n_c + mp}{n + m} \qquad (11)$$

where:

- a_i is the value of each attribute a_i tested for each class v_j.
- n is the total number of instances in each class v_j.
- n_c is the number of instances with attribute a_i and class v_j.
- p is a priori estimate for $P(a_i|v_j)$. If there is not information available, it is possible to assume that p = $1/k$, where k is the number of values that a_i can take (Keller, 2002b). We can assume that the probabilities of all attributes are equiprobable (equally likely to be true). In other words, each attribute can have k possible values.
- m is a constant used to avoid the possible consequences that could arise if $n_c = 0$ (in this case the calculation would be zero).

Acoording to Russell and Norvig, Bayesian learning methods use observations and probabilistic inference to get prior probabilities. The authors summarize that "Bayesian learning simply calculates probabilities of examples, and makes predictions on that basis" (Russell and Norvig, 2002).

A Bayesian classifier is a supervised learning method that requires a training phase before starting to work. This training implies the creation of classes in which the examples will be classified. In addition, the classes require typical examples that fit in each one. To classify a new example, it is compared with the old examples in each class. As a consequence, the designer of a classifier must create the different classes and provide the examples to train the classifier.

Classifying an Example

Let us study an example of a classification process, given a training set or base of cases with two classes of networks as shown in Table 2.2:

Table 2.2. A set of classes to classify examples (training set).

Example	Attribute 1	Attribute 2	Class
	Reliability	*Cost*	
1	High	Costly	A
2	Very high	Costly	A
3	Very high	Costly	A
4	High	Very costly	B
5	High	Very costly	B

We need to classify a network *E'* with a cost attribute which is "Very costly" and a reliability attribute which is "Very High". There is not any "Very high" reliability in class B and there is not any "Very costly" cost in class A. In which class must the new example be classified?

To answer this question, we first have to calculate the probabilities:

$P(\text{Very costly}|A)$, $P(\text{VeryHigh}|A)$

$P(\text{Very costly }|B)$, $P(\text{VeryHigh}|B)$

We apply equation (11) to calculate these probabilities:

$$P(a_i \mid v_j) = \frac{n_c + mp}{n + m} \quad (11)$$

where:

- a_i is the value of each attribute a_i tested for each class v_j.
- n is the total number of instances in each class v_j. Class A has three instances and class B has two.
- n_c is the number of instances with attribute a_i and class v_j. Class A has two instances of "Very High" and Class B has none.

- p is a priori estimate for $P(a_i|v_j)$. According to Keller (2002b) p means the probability of a_i having one of its possible values; in this case P is ½ because we assume that the probabilities of all attributes are equiprobable (equally likely to be true). In other words, each attribute can have two possible values.
- m is a constant used to avoid the possible consequences if $n_c = 0$ (in this case the calculation would be zero). We use m = 2 as a constant because we have two attributes (Reliability and Cost) in our training set.

We calculate the probability of the example to be classified in A and the probability to be classified in B. Then we apply the equation (11) m times for each class. Table 2.3. displays the values for n, n_c, p and m.

Table 2.3. Values to calculate where the example can be classified.

Class A	Class B
Very High $n = 3$ $nc = 2$ $p = .5$ $m = 2$	Very High $n = 2$ $nc = 0$ $p = .5$ $m = 2$
Very Costly $n = 3$ $nc = 0$ $p = .5$ $m = 2$	Very Costly $n = 2$ $nc = 2$ $p = .5$ $m = 2$

Table 2.4. shows the values for the probabilities of the example in each class.

Table 2.4. Probabilities for each attribute of each class.

Class A	Class B
$P(VeryHigh \mid A) = \frac{2+2*.5}{3+2} = 0.6$	$P(VeryHigh \mid B) = \frac{0+2*.5}{2+2} = 0.25$
$P(VeryCostly \mid A) = \frac{0+2*.5}{3+2} = 0.2$	$P(VeryCostly \mid B) = \frac{2+2*.5}{2+2} = 0.75$

Using the partial results in Table 2.3., it is possible to get the final results using equation (10):

$$v(E') = \underset{v_j \in V}{\arg\max}\, P(v_j) \prod_i P(a_i | v_j) \qquad (10)$$

where:

- V is the set of classes v_j.
- $P(v_j)$ is the overall probability of a class. This means the frequency of each class in the training set. The probability for Class A is 3/5 and that for Class B is 2/5.

In our example we adjust the expression to:

$$v(E'') = \underset{v_j \{A, B\}}{\arg\max}\, P(v_j) \prod_i P(a_i | v_j) \qquad (12)$$

where:

- {A,B} is the set of classes available.

Then, multiplying the probabilities for each attribute, we get the probabilities for each class:

P(A)= 0.6 * 0.2 = 0.12

P(B)= 0.25 * 0.75 = 0.1875

Now, each result has to be multiplied by the corresponding $P(v_j)$:

P(A)= 0.12 * 0.6 = 0.072

P(B)= 0.1875 * 0.4 = 0.075

This calculation can also be performed as follows:

P(A)*P(VeryCostly/A)*P(VeryHigh/A)= 0.6 * 0.6 * 0.2 = 0.072

P(B)*P(VeryCostly/B)*P(VeryHigh/B)= 0.4 * 0.25 * 0.75 = 0.075

Finally, a network with a cost "Very costly" and reliability "High" must be classified in Class B, because 0.075 >0.072.

A naïve Bayesian classifier is simple and consumes low time and memory requirements (Domingos and Pazzani, 1997). It performs well under a broader range of conditions, and the prior and posterior probabilities can be updated each time a new example is classified. Learning in naïve Bayesian classifier derives from updating those probabilities based on the frequencies in the training data. In addition, there is nothing to search in a tree or a database; thus, the classification of a new instance involves only computing the maximum class posterior probability (Keller, 2002b). Furthermore, a Bayesian classifier does not compare an example that is yet to be classified with those already classified. Instead of this, a Bayesian classifier can keep a summarized image of the frequency with which different characteristics of an example appear in the database. This informed search reduces the time needed to find the most suitable category for an example. Consequently, naïve Bayesian classifiers have been used and recommended for adaptive hypermedia and Web-based applications (Webb *et al.*, 2001; Miyahara and Pazzani, 2002) because they reduce the latency between requests and answers via Internet.

2.3. Conclusion

As a result of the review of literature related to our research subjects, we defined a series of concepts that supports our approach and that we applied in our system. Those concepts are grouped in three areas related to pedagogical aspects, teacher modeling and artificial intelligence in human-computer interaction.

In the area of pedagogical aspects, we take the concepts of Pedagogical Model, Teacher Model and Communication Model, which are used in classical ITS, to organize our work in areas related to each of those models. In our Pedagogical Model, we apply the approach to teach decision-making in Network Design that comes from the Best Net project. This approach is complemented by the Cognitive Load Theory (CLT) and Payr´s model for teaching and learning. With this group of concepts we structure a framework that supports teaching and helping in ARIALE. This framework allows us to map the theoretical view of the BestNet project for teaching Network Design to graduate students into an authoring tool to create teaching materials. A fundamental idea is using "worked-out examples" to reduce the cognitive load that teachers may experience while they are learning new concepts. Using worked-out examples requires that the Teacher Model keeps data related to which examples a teacher can reuse. In addition, worked-out examples are stored as cases that ARIALE manipulates applying artificial intelligence techniques.

Regarding the teacher's aspects, the concepts of human teacher model and implementer teacher are basic to model teachers` data in ARIALE. In addition, we explain the stereotype and overlay model, as methods to populate a Teacher Model with data. This model supports the decisions that ARIALE makes to help teachers.

At the end of this chapter we focus on concepts related to the application of artificial intelligence in human-computer interaction, particularly adaptation, intelligent help systems, knowledge based help, Web-based help, automatic generation of case studies and recommender systems. We also pay attention to machine learning and artificial intelligence techniques such CBR and Bayesian Learning because we apply them to deliver help by recommending worked-out examples.

Table 2.5. summarizes the most important concepts studied in this research that were applied in the development of our system ARIALE.

Table 2.5. Most important concepts used in this research.

Area	**Concept**	**Definition**
Pedagogical Model	BestNet Pedagogical Model	The BestNet Pedagogical Model integrates the optimization (competition) into other two strategies: analysis by comparison of cases and sensitivity analysis of cases. As part of a sequence of three tasks, comparison and modification of cases converge into the third task: optimization (Vázquez-Abad *et al.*, 2001).
	Network Design	In this research Network Design is defined as choosing a concentrator, defining links between the nodes of the network, and selecting the type of links for each link. According to this vision, Network Design is seen as a decision-making process and the student has to develop the expertise required to master this process.
	Decision-Making	According to Turban and Aronson (1998), decision-making is a process in which the decision-maker has to define the problem to be solved by means of a decision (intelligence phase); he also has to define the options and the criteria to evaluate these options (design phase). As well, he has to evaluate the options and select one or more that solve the initial problem (choice phase).
	Multi Criteria Decision-Making	Multi-Criteria Decision-Making (MCDM). "MCDM can be defined as the study of methods and procedures by which concerns about multiple conflicting criteria can be formally incorporated into the management planning process" (MCDM, 2003).
	Cognitive Load Theory (CLT)	The starting point in CLT is *cognitive overload*, a situation that a user may experience when the processing demands required by a task exceed his processing capacity (Mayer and Moreno, 2003). According to CLT, learning is the

		product of a process in which the learner uses his limited capacity working memory efficiently.
	Payr's model of teaching and learning	According to Payr (2003), the teaching/learning process supports the learner to pass five phases as he progresses: *novice*, *beginner*, *competence*, *proficiency* and *expertise*. In each phase, the learner performs the following activities intended to achieve some skills: *receive* and *remember; imitate* and *apply; select* and *decide; understand* and *design; recognize* and *master*.
Teacher Model	Local implementer teacher (Kinshuk, 2003) and human teacher model (Kinshuk *et al.*, 2001)	These concepts refer to the teacher as a person with particular characteristics and preferences, who wants to adjust or adapt an existing ITS or EAH according to his interests and expertise.
	Teacher as a trainee	Viewing a teacher as a trainee (Vázquez-Abad *et al.*, 2003). or a *teacher as learner* means that "teachers must see themselves as learners as they grapple with new technologies, changing roles and understanding the requirements of their students" (Fritze, 2003).
Human-computer Interaction and Artificial Intelligence	Adaptation, adaptivity	Adaptive systems can change their parameters and behaviour in order to meet a goal. Two general categories of adaptation are: 1- *adaptability*: users have the possibility to select different types of presentation and interaction from among the ones included in the system prior to the initiation of interaction. 2- *adaptivity*: the system possesses the capability to monitor user interaction to generate automatic adaptation that suits different users or contexts of use (Stephanidis, 2001).
	Intelligent interface agent	"An intelligent interface agent cooperates with the user to accomplish his task and functions as the user's personal assistant. The agent can take the initiative, rather than passively wait for instructions. It can provide the user with information, or detect and correct the user's misunderstandings" Ehlert (2003).
	Intelligent Help Systems	Intelligent Help Systems provide personalized, dynamic and contextual support by using artificial intelligence techniques for adaptation (Lorés *et al.*, 2002).
	Learning support	The learning support helps the user to extend his knowledge about a subject (Aberg, 2002).
	Problem Solving support	Problem-solving support helps users to find the solution to a problem (Aberg, 2002).
	Recommender system	A recommender system is intended to "learn about a person's needs and then proactively identify and recommend information that meets those needs" (Callan *et al.*, 2003).
	Automatic generation of case studies	Case studies, examples to be analyzed or, more specifically, network topologies can be generated automatically by a system instead of provided by a teacher or expert. The system requires functions or programs that follow rules and procedures to generate a case according to a expert criterion.
	Incremental learning, layering technique and adaptive help	These concepts mean the system adapts to the teacher as he learns and progresses from beginner to advanced (Brusilovsky and Schwartz, 1997; Selbach *et al.*, 2003).
	Minimalism	The minimalist approach means that the users receive small chunks of information to support them while performing a task or learning how to do it. (Selbach et al., 2003).

	Electronic Performance Support Systems (EPSS)	EPSS a type of on-line help "which provide integrated, on-demand access to information, advice, learning experiences, and tools to enable a high level of job performance with a minimum of support from other people" (Leung, 2001).
	Just-in-time help	Just-in-time help means that the system supports the user when he is in on the job, in the mere instant when he needs to develop the competence to perform a task (Leung, 2001).
	Just-enough help	This type of help means the user is going to receive just enough information to complete the current task. The idea is to avoid cognitive overload for the user provoked by information that is not going to be used (Leung, 2001).
	Context-sensitive help	Context-sensitive help provides information that relates to the specific field of the Web page in which the user is working currently (Self, 1998).
	Guide to Adaptive Interface Design	The guide to adaptive interface design proposed by (Rothrock *et al.*, 2002), has three rules for designing an adaptive interface: • identify variables that call for adaptation; • determine modifications to the interface, • select the inference mechanism.
	Case Base Reasoning	CBR is an artificial intelligence method that a system can use to learn. The idea is to allow a system to learn from its own experience. For example, an expert detects a new problem and compares it with the previous problems in the same domain that he solved before. If he finds a similar problem in his memory, in order to solve the new case he tries to reuse or adapt the same methods he used for solving the old problem (Cortés *et al.*, 1994). In CBR, learning is a by-product of problem solving.
	Classification learning	By means of classification learning, a system can learn to put instances into pre-defined classes (Keller, 2002a).
	Similarity measuring	This method calculates or measures the distance between two cases. This is a method to classify an example, taking into account examples classified previously. Each time a new instance has to be classified, the classifier method compares it to the stored examples to assign a distance value to the new example (Palu, 2004).
	Bayesian classification (Bayesian learning)	Bayesian classification is a method to classify new evidence according to given prior evidence. This type of classifier can classify sets of inputs and learn novel inputs in order to anticipate behaviour or a result (Rothrock *et al.*, 2002). This method is based on Bayesian theory, which uses previously known results to calculate more probabilities. According to this theory, an event has unconditioned probability or prior probability assigned in the absence of knowledge that supports its occurrence or absence. A conditional probability or posterior probability is the probability of an event given some evidence. Bayesian theory assumes the different attributes in a class or category are independent (Domingos and Pazzani, 1997).

Chapter 3. Previous studies

Summary: This chapter focuses on the description of previous studies that are close to our research in the areas of pedagogical aspects, teacher modeling, and the role of artificial intelligence in human-computer interaction.

3.1. Experience in supporting people

The next sections describe examples of previous development related to teaching and help, help system examples, support for authoring tools, problem-solving support, and Web-based help.

3.1.1. Teaching and help in Decision-making

Comparing the method that our system ARIALE (Authoring Resources for Implementing Adaptive Learning Environments) and other systems follow to support teaching decision-making is important to highlight improvements. Our approach for teaching guides the structure of our authoring tool, the kinds of problem-solving support and learning support to provide. This approach refers to teach students and it is the way that teachers must follow in this investigation. A recurring question in decision-making is how to deal with complex scenarios in which people must make decisions. An attempt to answer this question is Towle's ITS (Towle, 2001) to teach decision-making in complex environments. He proposes simulations of *crisis management*, using educational multimedia software with a task-based approach and a goal-based scenario. Another antecedent for teaching decision-making skills is "ADELE", a Web-based agent to support learning that has been applied to teaching medicine (Johnson *et al.* 2003; Shaw *et al.*, 1999). Another previous research study in this area is CACTUS (Hartley, 2003), an interactive simulation environment that provides facilities for supporting decision making and whose interface evolves to accommodate and support the varying roles of trainers and trainees.

Competition for optimization is one of the most widely used methods for "learning by doing" decision-making in disciplines such as business, finance and management (Erkol, 1998; Netstrat, 2003), but it is almost unused as a pedagogical strategy in Intelligent Tutoring Systems (ITS) and is very rare in Educational Adaptive Hypermedia (EAH) except for one research study (Aerts *et al.*, 1999).

Supporting Network Design

In Network Design there are not many projects that support teaching high-level design. One previous research study on teaching network management was developed to help novices improve their skills managing networks on the basis of the analysis of low-level information and the use of network faults to challenge students. This project permitted a gradual and controlled introduction of additional complexity and error conditions that the students face (Pattinson and Dacre, 1998). Other research in this area points to the development of intelligent agents to assist people with network configuration (Rezazad, 2003). For an amore detailed study of antecedents, see Pierre and Gharbi (2001), who also developed DESNET, a tool for designing, storing and reusing networks.

Researcher in Teacher Modeling

Using a Teacher Model to support adaptive hypermedia applications is not a very widespread practice, but based on a teacher's model, the WEAR system (Virvou and Moundridou, 2002) helps authors with teaching (strategy), and allows teachers to share exercises with one another. In other cases, researchers have taken into account the pedagogical knowledge that teachers have about how to teach a specific skill (Heffernan, 2001; du Boulay and Luckin, 2000). Other authors have captured users' decisions to model and emulate their decision-making processes (Webb *et al.*, 2001). An application of machine learning for user modelling can be seen in adaptive Web site agents and intelligent systems, which learn from visitors' access to a Web site. Research on intelligent profiling and interactive learning facilitates the acquisition of user information without unnecessary effort on the user's part (Shearin and Lieberman, 2001). Another view is that of a system which can watch the expert and learn from him, the expert supervises the learner, and, then the learner works alone (Kimiavi, 1998).

In general, researchers are interested in modelling the expertise of one human teacher to insert his expert knowledge into an ITS (Hefferman, 2001; du Boulay and Luckin, 2000). An exception is ARTHUR, a system with a different approach that includes versions of different teachers for the same course and recommends to each student the course that fits his learning style (Gilbert and Han, 1999). However, ARTHUR courses are static and adapting contents to other teachers' visions is not possible.

In contrast, research about user modelling and, more specifically, student modelling is very common. There are many investigations and projects related to the application of Bayes Theory and user modelling in general (Henze and Nejdl, 1999; Stern *et al.* 1999; Wittig, 1999; Carreira *et al.*, 2004). Billsus and Pazzani (1999) applied a hybrid user model for News Story recommendation. They used short-term and long-term models for interests. The short-term model is for known news and recently rated events to discover news related to them. The long-term model is for general preferences and trends. Millán (2000) uses Bayesian Networks for student modeling. Another important case is the application of a Bayesian model in ANDES (Conati *et al.*, 2002). In ANDES, the system predicts if a general concept has been learned based on the ability that the student demonstrates learning a particular concept.

3.1.2. Help system examples

In the area of assistance for users, some research aims at guiding workers while they are working, such as ADAPTS, an adaptative hypermedia ITS to guide in maintaining complex equipment (Brusilovsky and Cooper, 1999). Another important antecedent is Telephone Triage Assistance (TTA), a help system that supports nurses when they are answering by telephone (Leung, 2001). Moreover, Leung analyses and implements strategies and techniques for on-line help that are very useful to design on-line help and that can be applied to Web-based help systems. For example, TTA implements the "just-enough" and "just-on-time" approaches that a "task support system" can use to provide contextual help.

Other researchers have focused on supporting procedures like using a word-processor or a telephone-based system (Edwards and Hendy, 2000; Gorrel *et al.*, 2002), managing files (Virvou and Kabassi, 2002), or supporting medicine applications (Encarnação, 1997). Some efforts have been focused on intelligent help for supporting task completion in a complex command-base environment (Unix operating system) (Jerrams-Smith, 2000; Matthews *et al.*, 2000). Weber developed a guide to help students use a simulator in electrical and computer engineering classes (Weber, 1998).

Another trend is the analysis of how human experts support users when they are performing a computer-based task (Capobianco, 2002), and how a human expert can complement a help system (Aberg, 2002). A previous study related to the relation between experts and help systems is Kumar's investigation, which is intended to assist helpers while they are helping students on-line (Kumar, 2002).

An important concern is the automatic generation of user support, more precisely interface agents to help users perform tasks. COLLAGEN is a middleware that automatically creates collaborative interface agents which help users to understand an application (Einsenstein and Rich, 2002). The result is a *personal assistant* that learns "how to assist the user by (i) observing the user's actions and imitating them, (ii) receiving user's feedback when it takes wrong actions and (iii) being trained by the user on the basis of hypothetical examples".

3.1.3. Support for authoring tools.

The idea of using intelligent help to support users is not a new one. Carroll and Aaronson refer to SMARTHELP as an apparatus that allows a computer to monitor the activity in another machine in order to provide support (Carrol and Aaronson, 1988). Asked about how people can learn with a tool that is itself hard to learn, Carrol and Aaronson advised using artificial intelligence to support that difficult task.

However, past research has not focused on on-line intelligent and adaptive hypermedia that helps teachers to use ITS, which are an important aspects of this research. In fact, few authors have mentioned the lack of research on the subject of teacher modelling or the advantages of using information about teachers to improve Intelligent Tutoring Systems (ITS) (Kinshuk, 2001). For Virvou and Moundridou (2002), ITS require intelligent help to assist teaching decision-making skills while users are authoring, because regular help is not enough. Most authoring tools do not provide interactive support to exploit their functionalities.

Only very few systems specifically address the issue of how to support teachers to create an intelligent tutoring system. CACTUS is one of these systems that help teachers to create ITS. CACTUS includes an authoring tool to create ITS, as well as an agent that monitors and guides users in using the tool (García, 2000). Using a Teacher Model, WEAR helps authors with teaching (strategy) and allows teachers to share exercises among themselves (Virvou and Mondridou, 2002; Virvou and Moundridou, 2001). Another antecedent is Logic-ITA (Yacef, 2004), an Intelligent Teaching Assistant (ITA) that provides the teacher with information about students' performances. This information allows teachers to adapt their teaching to the students. Finally, the approach for authoring support for Web courseware proposed by Aroyo *et al.* (2002) in their system AIMS is a very elaborate option that recommends helping authors by providing: 1. Semi-automatic performance of some authoring activities; 2. Hints and recommendations; and, 3. Building and reuse activities support.

3.1.4. Problem-solving support

Another aspect related to supporting authoring is problem-solving support by means of automatic creation of courseware material. This is a really rare topic. An interesting antecedent is the automatic generation of quiz questions proposed in MULTIBOOK (Fischer and Steinmetz, 2000). In this system, a static number of components is combined to generate a limited set of questions. Based on an ontology of key words that represents concepts to evaluate, this system generates standard questions such as "Which are the parts of <Key word of the concept>?". Another earlier study in this area is HYPER-ITS, a system that includes a random problem generator which "selects concepts that will be treated independently given and creates instances of these concepts by randomly generating values within specified boundaries" (Kinshuk *et al.*, 1999). Pierre and Elgibaoui (1997), Pierre (1998), and Pierre and Leagault (1998) have proposed methods that use machine learning techniques for the generation of network topologies with low cost and a minimal acceptable delay. Obraczka *et al.* (1997) have used a random method for the generation of 50-node networks with determined reliability.

Recommendation

Recommending examples or case studies for teachers is another central aspect of our research. However, recommendation of examples is rare in teaching decision-making or teaching Network Design but also in teaching other subjects. AIMS (Aroyo *et al.*, 2002) includes recommendations in its functions of intelligent help for authors, but these recommendations are limited to leaning support and they do not include problem-solving support, as in our case. In contrast, Naïve Bayes classifiers are applied in recommender systems (Schwab *et al.*, 2000; Miyahara and Pazzani, 2002), but not in recommendations to support some functions of authoring tools.

3.1.5. Web-based help antecedents

Other research on on-line and Web-based help can help to refine the vision about authoring courseware and the generation of the respective help. For example, the ORIMUHS is a system independent of hosting software that supports a variety of users and tasks (Encarnaçao and Stoev, 1999). This help system is based on:

- Detection of user's actions.
- Identification of the tasks context.
- Creation of correspondences between concepts and user support.

Using these three features, the help system adapts to the user's knowledge, the context, goals, and actions to provide a multimedia help presentation. ORIMUHS has been applied in two areas, namely medicine and computer assisted design. This system is one of the very uncommon Web-based help systems that use artificial intelligence to adapt support to the context and to the user's characteristics.

Other examples of Web-based adaptive help are found in Interbook, a system intended to support students learning computer science. Trying to overcome the interface complexity, Interbook uses an incremental interface (Brusilovsky and Schwartz, 1997). According to the authors, with an incremental interface "a novice user starts with a subset of a complex interface, and then more advanced interface features are enabled incrementally as soon as the user needs them and is ready to use them". This idea of an incremental or gradual increase of the complexity of an interface and the help content has also been implemented in other systems that use buttons allowing users the access to different levels of help (Johnson *et al.*, 2003; Davis *et al.*, 2003).

3.2. Conclusion

The survey of literature allowed us to know the state of the art in the areas related to our research subjects and to define the most important differences and contributions that make our research innovative with practical solutions to real problems.

Regarding pedagogical aspects, we did not find another pedagogical approach as the one that we use to structure our work. In addition, we found that researchers are not interested in modelling the implementer teacher expertise in teaching because they are concentrating in creating the perfect teacher rather than a teacher's tool (Kinshuk *et al.*, 2001).

Researchers in this area have been working for a long time in student modelling and systems that can interact with students but not necessarily with human teachers. In general terms, research that focus on the teacher's problems to adapt ITSs and EAHs to his pedagogical goals and preferences is uncommon in the community of researchers in ITS and EAH. In contrast, our research pays attention to the teacher model, an aspect that has been neglected by the majority of researchers in ITS and EAH, and we have used the teacher model to support the decision of the system in recommendation, adaptation and help.

Some EAH and ITS share materials between teachers (Virvou and Moundridou, 2002; Aroyo *et al.*, 2002), leaving the selection of examples in the teacher's hands. These kinds of system usually do not recommend examples in a personalized manner, as ARIALE does to reduce the cognitive overload provoked by leaving the example selection to teachers. It is clear that if there are many examples available, the selection will be more difficult for teachers. Another important aspect related to examples is that automatic generation of courseware material, particularly examples, is not a common function of ITS and EAH. In contrast with MULTIBOOK and HYPER-ITS, the contribution of our system is not only the generation of more complex examples in a probabilistic manner, but also the use of these generated examples to increase the knowledge base of the system. By classifying examples, our system also learns the class of examples that the teacher prefers.

Although CACTUS, WEAR and the AIMS approach address the problem of supporting teachers authoring ITSs, these investigations do not focus on strategies and techniques to offer context adaptive Web-based help. Other researchers have developed tools for the design and implementation of hypermedia ITS (Carro, 2001; Wu, 2002), but these tools do not include features that allow implementer teachers to adapt the contents of their previously designed curricula. With these tools, if a teacher wants to adapt something, he has to author a new curriculum that can do this function. From some teachers' implementations, Arthur (Gilbert and Han, 1999) selects the style that best matches the student, but contents in this system are static and Arthur does not allow any adaptation; thus, the course must be used as it is provided. Capobianco's research is a good basis to develop help from strategies used by human experts, but it requires the implementation of strategies in Web-based help techniques. In contrast, our system has focused on adaptive help techniques.

The interaction of users with INTERBOOK and ORIMUHS interfaces is complex, and the interaction of the user with the help functionality also becomes difficult. Interbook uses conventional help features such as Index, and finding specific help with an Index help feature is time consuming (Leung, 2001; Capobianco, 2002) that we avoid in our research. As the creators of INTERBOOK commented, "INTERBOOK interface is too complex for many users. Some interface features, such as the separate table of contents, were misunderstood . . . Such helpful features of INTERBOOK as the search interface or prerequisite help have never been used" (Brusilovsky and Schwartz, 1997).

The main problem with the help in ORIMUHS occurs when the user wants to customize the support as this functionality is offered in an additional window with too many options to enable or disable. Therefore, the use of many windows is another source of cognitive overload in Web-based help since the user has to leave the workspace and switch between windows. The use of multiple windows is avoided in our work.

In conclusion, we found that the purpose and goals of our research are justified because they are pertinent, important, and attack relevant problems that other researchers have not studied with an approach similar to ours.

Table 3.1. compares ARIALE to other systems on the basis of the most important concepts applied in our research. For each of the aspects in the left column, a letter "x" in the corresponding row means that the system in the respective column includes this characteristic. For example, in the case of the aspect "Network Design", the systems ARIALE, BESTNET and DESNET have "X" because all of them are related to Network Design.

Table 3.1. Comparison of the characteristics of ARIALE with other systems.

SYSTEM	ARIALE	BESTNET	WEAR	CACTUS	LOGIC-ITA	AIMS	ORIMUHS	INTERBOOK	ADAPTS	TTA	CACTUS	NEGOPLAN	DESNET	MULTIBOOK	HYPER-ITS	ARTHUR
ASPECT																
Pedagogical Model (three tasks)	x	x										1				
Decision-Making	x	x									x	x				
Network Design	x	x											x			
CLT / Payr model for teaching and learning	x															
Local implementer teacher	x														x	
Different teaching styles from implementer teachers	x															2
Support for teachers	x		x	x	x	x					x					
Authoring tools for implementer teachers	x															
Intelligent help	x					x	x	x	x	x						
Context-sensitive Web-based help	x															
Automatic generation of some materials	x												x	x	x	
Automatic completion of tasks						x										
Learning (machine learning)	x									x						
Recommendation	x															

1 Only competition

2 The number of styles and the contents of courses are static.

Chapter 4. Teaching and teacher's characteristics

Summary: In this chapter we describe the theoretical basis of our research: the Pedagogical Model and the Teacher Model. In the section related to the Pedagogical Model, we answer the following question: What should the Pedagogical Model, methods and the teaching strategies to teach decision-making expertise and support teachers involve?
In the section devoted to the Teacher Model, we model the characteristics of a human teacher and his teaching style, and we analyse the application of the Teacher Model to support teachers in authoring learning sessions by means of our system ARIALE (Authoring Resources for Implementing Adaptive Learning Environments).

4.1. The Pedagogical Model

The Pedagogical Model is the set of teaching strategies and techniques that an Intelligent Tutoring System or an Educational Adaptive Hypermedia can apply to teach students. This model includes the knowledge and strategies about how to teach. The theoretical definition of the Pedagogical Model was discussed in pages. 16 and 17, section 2.1. of Chapter two. In this research, we extend the definition of the pedagogical model to include strategies and techniques that support teachers while they are authoring learning sessions (see section 4.1.2.). This is a Pedagogical Model for teachers, who must follow to teach decision-making in Network Design and this model also supports helping teacher while they are authoring learning sessions. A learning session is a sequence of tasks that the students will perform to acquire expertise in decision-making.

4.1.1. Teaching an expertise

According to Woolf (2000), ITS lack flexibility once deployed. The general rule is to author an ITS or an EAH for teaching a particular subject of predetermined content. Other authors have proposed that the objective of ITS should be to provide a way of teaching that can adapt to learners (Frasson and Aïmeur, 1997), thereby reproducing the behaviour of an intelligent human tutor (Aïmeur *et al.*, 2000). While an ITS or EAH is flexible and adaptive for the students, it is not necessarily flexible for a teacher if he wants to adapt it to his particular needs or preferences. Some authoring tools allow a teacher or some teachers working in collaboration to create an ITS for a course, but the tutor still reflects only the vision of its designer or designers (Carro, 2001). In ARIALE the Pedagogical Model and the structure of the curriculum allow teachers to adjust contents according to their particular goals. The next section explains the Pedagogical Model that our system uses to teach students and section 4.1.2. describes the Pedagogical Model applied to support teachers.

The pedagogical approach

Teachers who teach courses related to complex decision-making are usually university professors or highly qualified practitioners who often do not have the time to create case studies. In addition, they do not have time to learn how to use software for authoring cases, and as a consequence, it is difficult to train them to use a particular tool or a Web site. This is not only a problem for these particular teachers, but one that touches all teachers (Cuneo *et al.*, 2002). Motivated by these difficulties, we develop an authoring tool and an intelligent adaptive help system that enable any qualified teacher to create his own course-work, especially case studies, with minimal programming effort.

Our objective is for the teacher to guide students in learning how to apply operators or actions to transform the initial state of a problem to a new state. The pedagogical model behind this research was conceived to improve the students' skills in Network Design. This model breaks down the study of the decision-making processes into a series of activities that incorporate game elements to integrate theory and practice in a virtual environment of challenges and competition, similar to the training of engineers in the subject of Network Design.

In ARIALE, we extend the initial pedagogical model of BestNet (Vázquez-Abad *et al.*, 2001) to support the teaching of complex decision-making. The extension that we propose incorporates a pedagogical model for the course material to encompass courses in diverse areas such as medicine, business, engineering, and commerce. In accordance with this model, we can consider a decision-making process as the transition (T) of a state (S_n) to another (S_{n+1}) provoked by an action (Act):

$$S_{n+1} = T(S_n, Act). \tag{13}$$

In the context of multiple criteria decision-making, a problem is a decision to be made, for example, to design a network topology for a bank by taking into account many criteria, such as reliability, the network capacity and number of concentrator. An initial state is a specific configuration of the network topology and a new state is another configuration that results from applying an action, such as adding links to the network to improve its reliability. The new state can be the final design or just an additional step that can be closer to the final decision. A state is the state of a decision-making process.

The syllabus of courses related to complex decision-making is usually composed of a series of subjects, each one dedicated to the description of how different sub-systems behave as a function of the actions. For instance, textbooks on Network Design contain a chapter on network reliability, which are related only to the network connectivity and links characteristics. Another chapter may deal with capacity allocation, which is a function of the demand. The decision-making problem related to each subsystem (such as network reliability) is stated as the problem of finding the best action for each state to optimize the corresponding performance criteria. Different subsystems focus on different performance criteria and may emphasize different transition functions. Mathematically, each performance:

$$F_k (S_n, A_n) \qquad (14)$$

is a function (F_k) of the transformation of the state (S_n) that focuses on a subset of possible actions (A_n). In the example of Network Design, the reliability (probability that the network is functional) is a performance criterion associated with a mathematical function of the chosen links that connect the nodes, independent of the traffic demand patterns. When allocating capacity, the actions represent the capacity of each of the given links, and the performance is expressed in terms of overall utilization. Therefore, each possible action has N components:

$$A_n = (a_n(1), a_n(2), \dots , a_n(N)) \qquad (15)$$

and each of the performance criteria is expressed as a function of the states given a subset of all the possible decisions:

$$F_k (S_n , a_n (j)) \qquad j \in [1,N] \qquad (16)$$

This model considers a state as a complex system of elements and relationships; thus, when an element or connection changes as a result of an action or an operation, all the other components and relations are modified, and the existing state is turned into a new state.

Mapping the approach to courses

According to this pedagogical model, we developed a curriculum for teaching decision-making in Network Design. A Curriculum is the sequence of *activities* and *tasks* of a *course* related to a *subject*. Our system supports the creation of sessions for teaching multi-criteria decision-making. We have developed a Web site to support the task of adapting curricula, and we have included an example of a curriculum for teaching decision-making in Network Design. Many designers can design courses with a particular number of activities (units or chapters). The Curriculum is very flexible because each course Curriculum is just a structure that the system fills dynamically. The number of activities for courses is flexible--for example, a professor, let's say Dr. Warfield, can design a course with eight activities for teaching decision-making in Network Design, while another professor, in this case Dr. Vázquez, can design a Network Design course with four activities. The content can be different, but the parallel structure of each unit or activity is similar.

A course can be implemented by many teachers. The implementation allows each teacher to adjust the course to his pedagogical goals by creating specific learning sessions. Then, a teacher can implement the course in a variety of ways by authoring different sessions. Each session is an instance of the course. The assignation of sessions to groups involves the assignation to students. A session is like a hypertext that can be represented by an all-connected graph. The nodes are the activities, and a teacher can go from one to another by following a sequence, or he can jump from any node to another, according to his interest. We have not expanded on the aspects related to designer and students because our focus is the teacher. Figure 4.1. shows the organization of possible curricula.

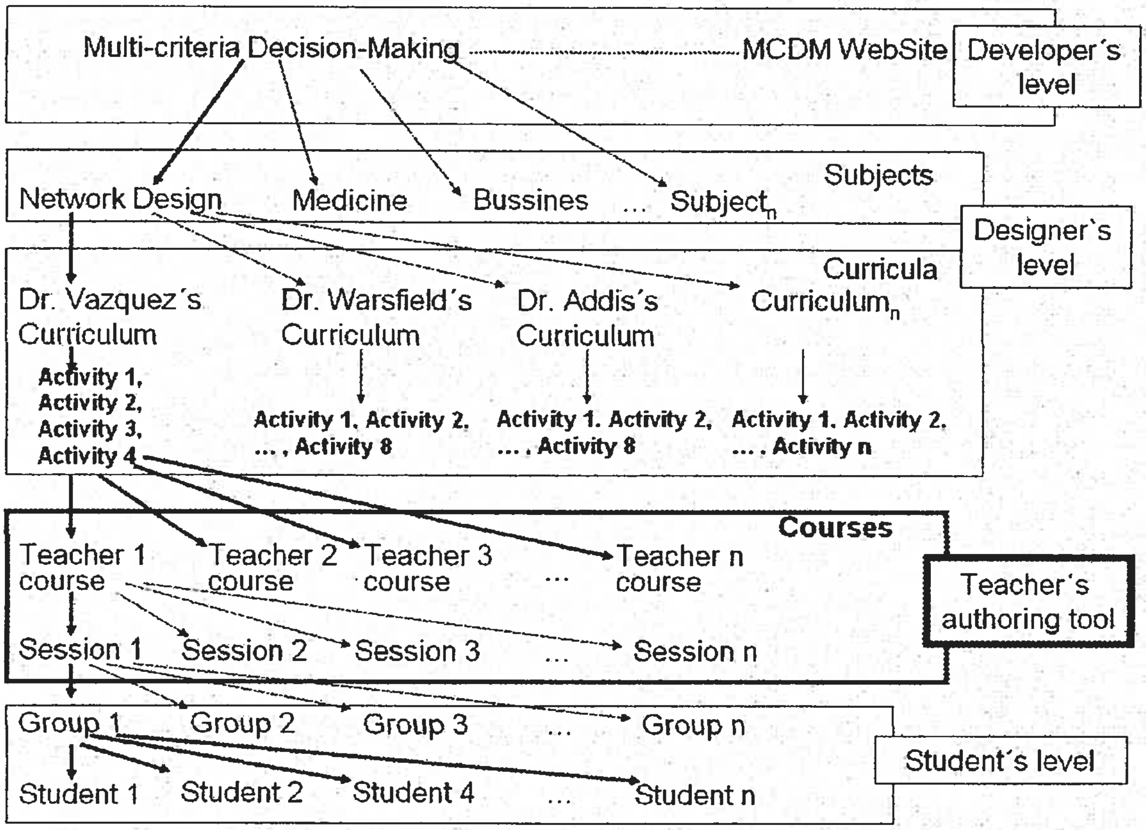


Figure 4.1. Structure of different Curricula

From a general schema to specific patterns

A curriculum is also a sequence of *activities* that allows students to learn how to apply concepts previously covered in class. It is also a plan which includes the main concepts used in the decision-making process that the students will master. Each concept maps to an activity. For every subject, at the end of the sequence of concepts another activity is added to integrate all the concepts studied in previous activities. It is conceived as a virtual competition where an underlying simulation of IF-THEN cases is presented to the students to prompt them to integrate all the concepts according to the teacher's pedagogical objectives. For example, in our research, teaching the subject Network Design involves three main concepts or activities: Realiability, Concentrator Location and Capacity Allocation. A last additional activity is LAN Design, which integrates the other three activities. Each activity is broken down in three specific tasks for the integration of the knowledge of a subject. These three tasks implement the pedagogical strategies used for teaching in this Pedagogical Model: analysis by comparison, sensitivity analysis and optimization. As part of a sequence of three tasks, *comparison* and *sensitivity analysis* of cases are the two initial tasks that converge into the third task: *optimization*. These tasks are repeated in different activities throughout a course to provide a virtual environment where students can gradually improve their expertise in making decisions about how to design a network. The different tasks will be accomplished by following a sequence of steps (Web-pages) and actions (Rodríguez *et al.*, 2003). Each step is also broken down into a sequence of actions. An action is a specific decision that the teacher can make while authoring a learning session in a particular field of a step, for example, clicking to select an option of a radio button or writing an instruction in a text box. The organization of a curriculum is depicted in Figure 4.2., and, later, in Table 4.1. is an example of the Network Design curriculum.

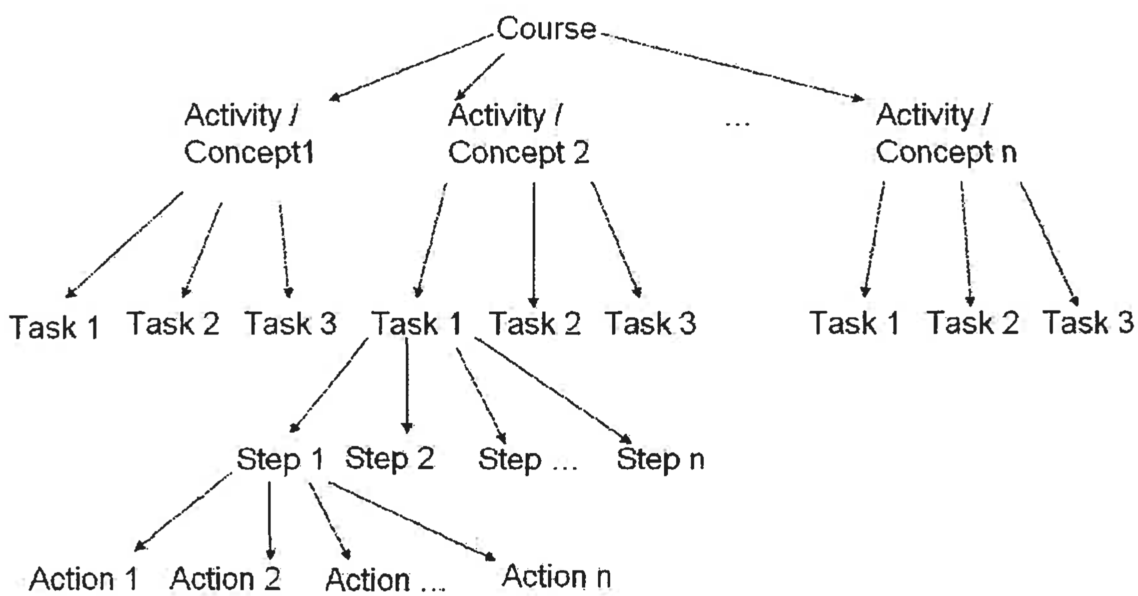


Figure 4.2. General organization of a curriculum.

Each activity (except the last, which always consists in the integration of the previous ones) entails the following three pedagogical strategies:

- *Analysis* of cases: In the case example, the activity prompts the student to compare various networks on the basis of reliability, cost, traffic balance, failure scenarios, etc. The students gradually acquire the expertise to apply particular decision-making patterns by comparing and analyzing different aspects related to Network Design cases. Comparison allows students to notice structural features which are different or similar among problems of different categories (Gerjets and Scheiter, 2003).

- *Modification* of cases: Following the same type of example, it is possible to select actions (delete or add a link) that cause a given change in the reliability of a case. The modification of a topology is an application of sensitivity analysis, which is a technique for observing the relative change in a model response (output) if we change one input at a time (Max and Lynn, 2003).
- *Competition*: An example of competition is when a teacher asks students to search for a network topology that meets a specified reliability level. The "winner" is the person who designs a topology with the specified reliability level at minimum cost. The competition is an optimization process in which each student applies knowledge learned in the previous tasks by changing a given network and comparing it with the goal until achieving a network that fits or is close enough to the goal. As a result, "from an example structure, learners have to infer concepts that go beyond the information provided in the examples" (Gerjets and Scheiter, 2003).

As a student progresses, he moves on from the level of competence, passing to the proficiency level until he reaches the expertise level, according to Payr's phases (Payr, 2003) already explained in pages 19 and 20, section 2.1.4. The analysis of cases allows the student to select and decide between two options through a comparison of the two cases. This is the phase in which the student acquires competence. After this, the student explores an environment, modifies it, and arrives at the proficiency phase. During the competition, the student has to master the process to create solutions. By doing this, he acquires expertise. This process is repeated with each activity, and each time the student acquires new schemas or patterns for making decisions, he accumulates more and more expertise.

A detailed example

In page eight of Chapter 1, we describe an example of teaching Network Design by means of a curriculum with four concepts that map to the correspondent activities. Now, we follow this example to show what a curriculum is. In our example we have the following activities to be applied in the decision-making process of designing a network:

Reliability. In this activity, the goal is to understand the concepts of reliability and availability, as well as the manner in which the link choices affect overall performance.

Concentrator Location. This activity consists of defining the optimal number and location of the "concentrators" (central processors storing the bank database). The goal is to place the concentrators at minimal cost to fulfill the minimum safety requirements for information backup.

Capacity Allocation. This activity is intended to find the optimal choice of capacity for each of the links of the given topology. The goal is to allocate capacity per link and define the optimal routing that minimises cost while satisfying the bandwidth requirements.

Design of a local area network. In this activity, each student has to propose the optimal design of a network that incorporates all the choices described above (Vázquez-Abad *et al.*, 2001).

The first three activities correspond to three sub-problems in network design, all of which can be unequivocally stated as mathematical optimisation problems. These three activities integrate knowledge from different sources, and the last activity allows the integration of the three previous activities and methods and heuristics for network design. For example, it is common to teach mathematical formulations of the reliability independent of the demand of the network, and the problem of capacity allocation is formulated assuming no link failures. In real life, however, traffic patterns may affect the probability of link failure, and link failures may affect the utilization of the network. In addition, a person can learn how to make decisions about designing a network with a minimum degree of network reliability; after that he can learn more about the location of concentrators to support an acceptable degree of network reliability; in a third activity, this person can learn how the assignation of capacity to different links increase or decrease the network cost. Once he has accumulated knowledge about network reliability, concentrator location and capacity allocation, he will apply these three aspects together in the design of a network, a more complex activity that integrates the previous three. Table 4.1. illustrates the structure of our Network Design example.

Table 4.1. Example of the Network Design curriculum.

Course	*Activity*	*Task*
Network Design	Reliability	1. Comparison of two network configurations
		2. Modification of a network configuration
		3. Competition to get the best reliability for the network configuration.
	Concentrator Location	1. Comparison of two allocations of concentrators
		2. Modification of the location and number of the concentrators
		3. Competition to get the best location for the concentrators of a network at minimal cost at minimal risk.
	Capacity Allocation	1. Comparison of two networks with different link capacity.
		2. Modification of the capacity of the links.
		3. Competition to get the best capacity allocation for the network at minimal cost, to satisfy the demand.
	LAN Design	A single several-stage simulation integrating all aspects of the previous activities.

As stated by Vázquez-Abad *et al.* (2001), "using the metaphor of a bank network, each student is assigned a character role as an Engineer interviewing for a position for chief network manager. Students are supposed to design a telecommunications network to connect a series of bank branches. Their mission is to complete a design that will "beat" all other competitors, by deciding how the various objectives and constraints will be integrated into a model as well as taking into account other considerations that may not enter mathematical models directly".

4.1.2. Helping to teach

Traditional AEH and ITS use a Pedagogical Model that only take into account the strategies to teach students, but in our case we also use strategies and techniques to support the tasks that teachers must perform while authoring learning sessions with ARIALE. Our system includes three help strategies that are *Basic Help*, *Guidance*, and *Assistance* (Rodríguez et al., 2004b). Depending on the teacher's skills or experience in performing a task, the system can use one of three types of help strategies.

Basic help is a strategy oriented to provide just-enough contextual help or problem-solving support. *Just-enough contextual help* means the system displays only short messages with the most relevant information related to a specific aspect of the current Web page. *Problem-solving support* provides a complete solution for a problem; in our case problem-solving support is the recommendation of a worked-out example to be reused. *Guidance* is intended to supply short information about how to perform a specific task. In a condensed way, *Assistance* reminds the teacher about information previously displayed. These strategies can explain a concept either in brief or in detail, depending on the teacher's skills (beginner, intermediate or advanced). We use these strategies to reduce the cognitive load that teachers may experience while they are authoring an adaptive hypermedia ITS. The strategies are mapped to help techniques that allow the system to provide adaptive context-sensitive help. Table 4.2. shows the help strategies and their corresponding help techniques.

Table 4.2. Help strategies and their techniques

Help Strategies	*Basic help*	*Guidance*	*Assistance*
Help Techniques	Definition Introduction Hint Instance	Demonstration Guide Tutor	Quick reference Explanation • Example • Description • Analogy

The techniques of the *basic help* strategy are the simplest ones in the system, and it provides the easiest contents to novice authors. If the system applies the basic help strategy in a Web page, one or all of these techniques can be included in the Web page:

- *Hint*. This is advice that is displayed as a dialogue, and the teacher can accept or reject the advice. For example, when a Web page to select a session is loaded, a beginner teacher can receive a hint that recommends a session to reuse or edit. An example of a text message in a hint is "You have some incomplete sessions. Do you want to finish the last one?". If the advice is rejected, ARIALE stores this teacher's answer and use this data for future decisions. The system learns from the teacher's actions and, for example, a different advice can be displayed next time.

- *Definition*. This is a short definition of a concept associated with a field. For example, when the teacher moves the mouse over a "Session" link in the menu, the system displays a definition of what that link allows the teacher to do: "Edit a learning session".
- *Introduction*. This is an introductory explanation of a concept associated with a field. In the case of text-based help, this introduction is divided into three complementary sentences. These sentences are a short introduction; a second sentence adds more detailed information, and a third one is a long and more complex sentence with complementary information.

  Each sentence is added on demand in a gradual manner. For example, when the teacher clicks a "MORE" link in a Web page for a first time, the system displays a sentence with a short introduction about the concept associated with this link. After a second click, a second sentence will be displayed and so on. An example of a text for an Introduction related to the criteria to be used in analyzing examples is: "You might select reliability, availability and cost as the criteria to analyze the example".
- *Instance*. Instance is a technique used in problem-solving support to recommend examples. A particular instance is a worked-out example (of a session) to be reused or edited. For example, if the teacher accepts a recommended session, the system loads and displays the recommended session.

With the exception of Instance, the contents used by help techniques are specific resources (texts, graphics) that the system has to retrieve and integrate in a Web page. The Instance is a set of instructions and examples that is part of a learning session (a learning session is a case and the structure of a case is explained in section 2.2.7.).

The techniques used in the Guidance strategy are:

- *Demonstration*. This is an animation or video that shows the development of a process or explains a concept. When there are bandwidth restrictions, the system uses a slideshow that includes static images and text.

- *Guide*. This is a brief description of a sequence of steps needed to perform a task or to explain a concept.
- *Tutor*. This technique is a more detailed description of a process or a concept, including examples.

In the case of the strategy Assistance, the corresponding techniques are classified in two groups: Quick Reference and Explanation. Quick Reference is a short list of topics referring to a concept or process. The Explanation can be displayed as:

Example. A specific example of a concept or an instruction about a particular aspect.

Analogy. A comparison of two instances to explain their commonalities.

Our system uses these techniques because the most used techniques of classical help (Index, Find, Content) (Purchase and Worril, 2002) generate many problems that increase the cognitive overload of users (Capobianco, 2002; Leung, 2001). We decided to include Instruction, Guide, and Instance and to redefine other techniques to adjust them to the context of our research. We did not test all the strategies proposed in this section because we focused on the technique Instance of the strategy Basic help. The design of the techniques Instance is explained in the next chapter; its implementation, as a part of ARIALE, is described in chapter six; and the results of testing the generation and recommendation of an examples or instances are discussed in chapter seven.

4.2. The Teacher Model

In this research, we identified three types of users: designers, implementer teachers, and students (See Figure 4.1.). The designer organizes the curriculum of a graduate course. He defines the different activities or units to be taught and their sequence. By implementer teachers, we mean university professors who are using a Web-based authoring tool to create courseware for a subject or course prepared by a designer. These teachers may be experts in the subject, but not necessarily skilled at using Web-tools. The student receives the final product, which is a virtual environment implemented by the teacher and based on the course structure provided by the designer. However, the focus of this research is neither on the designer nor on the student. Our interest is the implementer teacher.

The teacher is not only a learner

We have adapted the view of the teacher as learner of new technologies (Brusilovsky and Schwartz, 1997) to that of a trainee who requires only adaptive contextual help and support to solve problems, rather than learning to use the authoring tool in order to be evaluated (Guzdial, 1999). In this scenario, the teacher's learning becomes a by-product of the support that he receives. This is a fundamental characteristic in our research because we do not see the teacher as a typical student. Rather, we see the teacher as a worker who can learn while he is on the job.

4.2.1. Our model

In this research, *the Teacher Model* profiles the characteristics of the teacher according to his identification, knowledge level, learning style, teaching style, preferences, and history of previous activities. It is also possible to classify the characteristics according to: *cognitive factors* related to the knowledge; *interests and skills* of the teachers; as well as *affective aspects* that imply preferred media for receiving messages, willingness to share and communicate knowledge, or layout details such as the size of the letter to show running text on the screen. Table 4.3. shows some examples of each type of aspects.

Table 4.3. Examples of cognitive and affective characteristics of a Teacher Model.

Type of characteristic	
Cognitive	**Affective**
Years of experience teaching this course.	Preference of text, image or video for receiving help.
Class of the sessions used generally by this user (teaching style).	Preference for sharing knowledge.
Experience using computers.	Acceptance or not of animated agents.
Concepts most frequently required from help.	Layout of a page.
Language.	Preference of media to teach.
Background area of the teacher.	Preference for editing existing sessions.

Modelling the teacher

The Teacher Model is the main source of information that the system uses for making decisions. Section 2.1.6. discusses general concepts related to the Teacher Model. In this research, the Teacher Model includes information that comes from the registration process, the interaction between the teacher and the system, and the examples that teachers use for teaching decision-making on Network Design. Our Teacher Model organizes teachers' information in six groups: Background Knowledge, Teaching Styles, Learning Styles, Preferences, Demographic Information and Performance Statistics. Table 4.4. shows the groups and type of information included in each one.

Table 4.4. Classification of teachers' characteristics.

Group	**Description**
Previous Knowledge.	Teacher's general background.
Learning style.	Preferences for learning.
Teaching style.	Preferences for teaching.
Demographic aspects.	Data related to age, gender, etc.
Preferences.	Attitude toward sharing examples, communication, type of help, etc.
Performance using the system.	Statistics about the interaction of the teacher with the system.

The Teacher Model is also a repository of two types of the teacher data: the initial information (background, preferences, etc.) that forms the Teacher's Profile, and the history of his performance and pedagogical decisions, namely the Overlay Model. The information in the Teacher's Profile, does not change frequently, such as the teacher identification, username, background knowledge and some preferences. This profile is used for the initialization of the Teacher Model (the initialization is described in section 4.2.2.). The Overlay Model changes continuously because it keeps data on the sessions, the times a specific fragment of text has been accessed, and the teacher's preferred class of examples. For example, during the initialization of the Teacher Model for a particular teacher, our system ARIALE compares the teacher's data such as the subject to be taught, the teacher's background and the expertise in teaching the subject with other teachers′ data. As a result of that comparison, the teacher is classified in a group of teachers with similar expertise, background and subject to teach, and the system infers that examples used by the teachers in the selected group can also be used by the new teacher (See collaborative filtering in Chapter two, page 34). When the teacher has chosen examples by himself, data related to those examples that the teacher prefers is stored in the Overlay Model to support future recommendations of examples. That data allow ARIALE to generate, test and recommend new examples that matches the teacher's preferences about examples. Figure 4.3. shows the components of the Teacher Model.

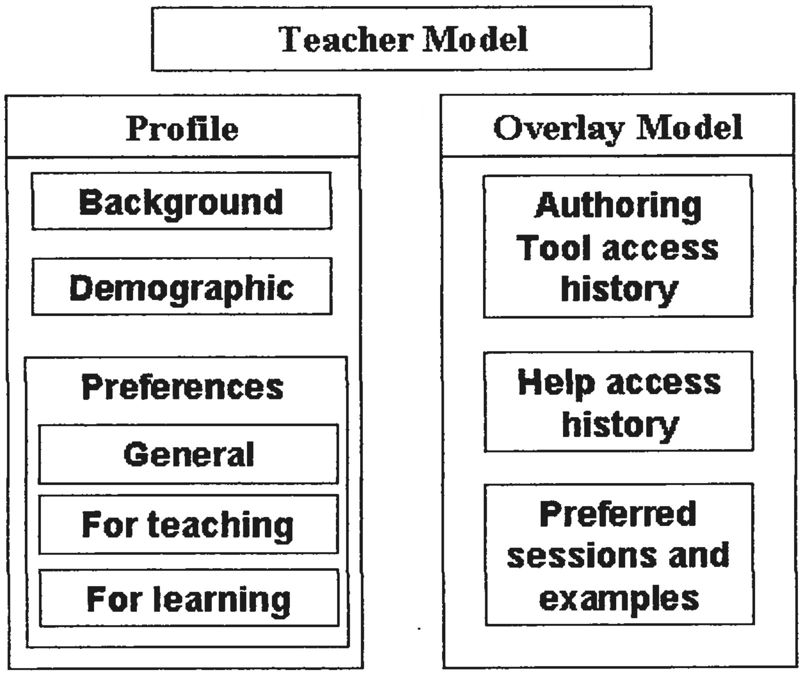


Figure 4.3. The components of the Teacher Model.

Creating a Teacher Model

The creation of the Teacher Model starts in the registration phase, continues with the creation of a teacher's profile, and finishes by classifying the teacher as a *Beginner, Intermediate* or *Advanced* teacher. These three categories are based on teachers′ characteristics related to teaching a subject. Our decision of using three categories is based on the Virvou`s esperience (Virvou and Mondridou, 2001; Tsiriga and Virvou, 2002) who used the categories "novice", "having little experience", and "experienced" to classify teachers` teaching expertise. In fact, we can find two types of teachers: with or without experience in teaching a subject and in using an authoring tool. But, there is an intermediate group of teachers who are neither highly skilled nor novice. That is the reason why we create three categories of teachers. These categories are use to initialize the Teacher Model as we explain in the next section.

In the registration process, the teacher provides data while filling out specific forms. Some data identify the teacher and his preferences to guide the system in recommending examples and help while the teacher has not selected or created a particular learning session with defined examples. Other data is related to general preferences such as the interest for sharing examples with other teachers. Using other teachers' information entails aspects related to privacy and security that are not the focus of this research. We decided to ask each teacher if he wants or not to share his information and examples with other teachers. If a teacher accepts to share his information, he is going to know that no more teachers will know his particular information but some of his data will be used for statistical purposes. The data collected during the registration refer to:

- Course subject to be taught.
- Background area of the teacher.
- Expertise in teaching a course.
- Number of years teaching a course.
- Level of the course to teach (undergraduate, postgraduate).
- Experience using computers, Web-based technology and our system.
- Interest in communicating and sharing examples with other teachers.
- Preferred media in which to receive help.
- Time available to learn how to use the Web site.
- Total number of students the teacher will teach.
- Language.
- Age.
- Gender.

In general, women and seniors are seen as the "digital have-nots" and that is the reason because we consider age and gender as two important characteristics of university teachers, too. In the case of women, Cuneo (Cuneo, 2002) explains that "Gender differences are very small in North America and some of the Scandinavian countries, but are quite pronounced in other countries, such as Saudi Arabia…".

Once a teacher finishes the registration, he receives a username and a password for future login. Simultaneously, the system creates an initial Teacher Model, one with no statistical information because he has not authored anything yet. The creation of the initial Teacher Model involves classifying the teacher in one of the three categories described above, as explained in the next section.

4.2.2. Initializing the Teacher Model

The purpose of ARIALE is to recommend sessions and examples that a teacher likes. The biggest problem that the system faces after a new teacher registers and encounters the system the first time is how to start recommending examples if the system does not have enough information about which examples are suitable to him. This is the called "startup problem" or "cold-start" problem (Schein *et al.*, 2002) and it is a common problem found when there are not enough data to support a recommendation. In our system, we solved this "cold-start" problem by recommending examples that some teachers can like (like-minded teachers). Each new teacher is placed in a group or category of teachers (Beginner, Intermediate, Advanced) whose characteristics are similar to his. Each group of teachers conforms a stereotype and a stereotype is a representation of teachers' characteristics pertaining to this category. Even if every teacher does not start up with the same knowledge about the domain being taught, some of them can share similar partial knowledge and other characteristics. ARIALE assumes that a new teacher can reuse the type of examples most often used by other teachers in his category. For example, teachers of similar number of times teaching a Network Design course, who have the same background and experience teaching the subject, can have similar preferences to use a particular type of examples of network topologies. Of course, there is a failure risk because they can use different examples, but we took the risk of failing to start recommending examples, instead of making a simple random choice or a fix option for every beginner. This assumption is more likely than a random method because we are taking into account similarities between teachers rather that assuming they are absolutely different. We assume this position because there were not many professors teaching this curriculum and the respective statistics to generate some correlations. In addition, we assume that teachers with similar characteristics share a teaching style, because in this research there are no statistics between a teacher's category and a teacher's teaching style to establish correlations. This means that the teacher is assumed to have the teaching style that most teachers in a particular category have (the generation and recommendation of examples are explained in section 5.4.). If the teacher does not have that style, he can modify the example provided to teach the system the kind of examples that he really prefer.

In our case, ARIALE chooses a group of teachers as a predictor group of a class of examples and, examples of that class are recommended. This is an application of collaborative filtering that uses user-to-user associations based on similar interests, patterns or characteristics of teachers to group similar teachers to categories. Even if the system fails matching the teacher's style, the teacher will receive a worked example to start authoring. Stereotypes and collaborative filtering are useful when there is not enough information about a new teacher but the accuracy of the initial Teacher Model is low. After the new teacher authors his first learning session, the information from the teacher's category is not taken into account anymore for recommendations. Over time, as each teacher acquires more experience using the Web site, teaching the subject and authoring his own learning sessions and examples, the system switches from the stereotypes and collaborative filtering to take into account each teacher's own examples (See content-based filtering explained in Chapter two, page 34). With examples created or selected by each teacher, the system can continues recommending examples with a higher probability of success, in comparison to the method applied at the beginning or cold-star phase. Using an overlay model and content-based filtering, the system gets a more accurate personalization of each teacher automatically (Callan *et al*, 2003), by monitoring the examples that each teacher uses.

Classifying a new teacher

After the new teacher registers, the system takes his data and classifies the teacher in one of the existing categories (Beginner, Intermediate, Advanced) by means of a Bayesian classifier, a common technique in user modeling. Our system uses a naïve Bayesian classifier to classify teacher's characteristics and assigns the new teacher to the category in which the teacher's data are most suitable. Once the teacher belongs to a category, the most frequently used class of examples in this category is used to recommend sessions and examples to the new teacher, as explained in section 5.4.

Naïve Bayes classifiers have been used and recommended in adaptive hypermedia (Webb *et al.*, 2001; Miyahara and Pazzani, 2002) because they reduce the latency between requests and answers via Internet. Furthermore, a Bayesian classifier does not compare the teacher's data with cases in the database as the K-NN classifier would. Instead of this, a Bayesian classifier keeps a summarized image about the frequency with which different teacher's characteristics appear in the database. This kind of informed search reduces the time needed to find the most suitable category or class for a teacher.

As explained in section 2.2.7., the Bayesian classifier is a *supervised learning* method that requires a training phase. Synthesized data was used for the training phase because there are not enough teachers teaching our curriculum. Using synthesized data (see Table 4.6.) is not an uncommon practice when constructing complete systems (Herlocker *et al.*, 2004) that apply collaborative filtering. Each of our categories (beginner, intermediate and advanced) stores two types of teacher data, *Demographic* and *Background knowledge*. Demographic data is used because there is evidence of the relation between age and gender and the preferences of university teacher to use information technology (Cuneo *et al.*, 2002). The data in both types is detailed as follows (we associate each characteristic with a letter for future identification):

<u>Demographic</u>

A. Age.

B. Gender.

<u>Background</u>

C. Teacher's self-evaluation of his skill or experience teaching the selected subject.

D. Number of times the teacher has taught this type of course.

E. Teacher's background discipline or area of expertise.

F. Teacher's skills using computers.

G. Teacher's experience using Web-based technology for teaching.

H. Teacher's experience using this Web site.

An example of a new teacher to be classified can be:

Age = 45

Gender = Male (M)

Self-evaluation = Beginner (B)

Times teaching this course = 1

Teacher's background = Network design (ND)

Teacher's skills = Medium (M) average

Teacher's experience using Web-based technology = Never (N)

Teacher's experience using this Web site = Never (N)

Generally, people until 30 years have the highest use of Internet, after that the use of information technology becomes lower with the pass of the time; finally, the period over 55 years age is of very low Internet use, for example (Cuneo, 2002). We have defined these rules that organize data about age in three ranges:

If teacher's age $\leq$ 30, then his age range is 1.

If teacher's age > 30 AND $\leq$ 55, then his age range is 2.

If teacher's age > 55, then his age range is 3.

This new teacher's characteristics are summarized as follows in Table 4.5.:

Table 4.5. A new teacher's attributes and values.

	Teachers' Characteristics								
Teachers	A	B	C	D	E	F	G	H	Class
A new teacher *E''*	2	M	B	1	ND	M	N	N	?

This example has to be classified in one of the three categories. A set of training examples is shown in Table 4.6., and Table 4.7. includes the values that the system uses for the different attributes.

Table 4.6. A training dataset.

A	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*Class*
3	F	I	1	ND	A	N	S	Beginner
3	NA	I	2	DM	H	S	S	Beginner
3	M	B	2	ND	L	N	S	Beginner
3	M	I	1	DM	A	S	N	Beginner
3	F	B	2	DM	A	S	O	Beginner
3	NA	I	3	DM	L	O	N	Beginner
3	M	B	1	DM	A	O	N	Beginner
3	F	B	3	O	L	O	O	Beginner
3	F	I	1	ND	H	O	S	Beginner
1	M	I	3	ND	H	S	S	Intermediate
1	NA	I	1	O	H	N	O	Intermediate
1	NA	B	2	O	A	O	N	Intermediate
1	M	B	3	ND	A	O	O	Intermediate
1	NA	B	3	O	H	O	S	Intermediate
1	N	A	1	O	A	S	O	Intermediate
2	F	I	3	O	H	S	O	Advanced
2	NA	I	2	ND	L	N	S	Advanced
2	F	A	3	DM	L	O	N	Advanced
2	F	A	1	O	H	S	S	Advanced
2	M	I	2	ND	H	S	S	Advanced

Table 4.7. Possible values for the different attributes.

A	*B*	*C*	*D*	*E*	*F*	*G*	*H*	*Class*
1	F	B	1	ND	L	N	N	Beginner
2	M	I	2	DM	A	S	S	Intermediate
3	N	A	3	O	H	O	O	Advanced

The meaning of the values in Table 4.7. are

For age

1= teacher's age ≤ 30

2= teacher's age > 30 AND ≤ 55

3= teacher's age > 55

For gender

F= female

M= male

NA= not available

For teacher's self-evaluation

B= beginner

I= intermediate

A= advanced

For times teaching a course

1= never

2= 1 to 5 times

3= more than 5 times

For teacher's background discipline

DM= decision-making

ND= network design

O= other

For teacher's skills using computers

L= low

A= average

H= high

For teacher's experience using Web-based technology

N= never

S= sometimes

O= often

For teacher's experience using the Web site

N= never

S= sometimes

O= often

With these training dataset, it is possible to apply equation (11)

$$P(a_i \mid v_j) = \frac{n_c + mp}{n + m} \qquad (11)$$

where:

- a_j is the value of each attribute a_i tested for each class v_j. For example, an instance of a_i can be 2.
- n is the total number of instances in each class v_j, In this case of class Beginner, $n = 9$.
- n_c is the number of instances with attribute a_i and class v_j. This is the number of times that the value of the attribute a_i appears in each class v_j. For example, the value "M" appears one time in the Advanced category, in this case $n_c = 1$.
- p is a priori estimate for $P(a_i|v_j)$. We assume that the probabilities of all attributes are equiprobable and each attribute can have three possible values and $p = 1/3$.
- m is a constant used to avoid the possible consequences that could arise if $n_c = 0$ (in this case the calculation would be zero). We use m = 8 as a constant because we have eight attributes in the examples of our training dataset.

After the calculation of the probabilities of each attribute, we apply equation (10) adapted to this example:

$$v(E'') = \frac{\arg\max}{v_j \in \{B, I, A\}} P(v_j) \prod_i P(a_i|v_j) \tag{10}$$

the result is as follows:

Class	**Probability**
Beginner	0,0002572241
Intermediate	0,0000351170
Advanced	0,0000061686

Consequently, the new teacher is classified in the category "Beginner" because the classifier predicts that this class has the highest probability for this teacher. Once the teacher is classified, the system has new information with which to support the teacher with help and recommend sessions and examples. Chapter four explains the contribution of our research related to recommendation and help.

The algorithm to initialize the teacher's skills and teaching style is a key process that allows the system to start working, as follows:

```
Procedure Initialize Teacher Model () {
        Retrieve Demographic and Background indicators from the teacher's profile
        Classify indicators in the most suitable category of teachers (Bayes classifier)
        Assign the skills level of the category to the new teacher
        Retrieve the most frequently used class of examples in the category
        Assign this class to the new teacher

        IF there are some data missing in the teacher's profile THEN
                Replace missing data from the assigned category

        Detect teacher's preferences
        Initialize the teacher's overlay model (history statistics)
        Initialize first plan to perform
}
```

The teaching style

Teachers work in different manners and they have particular tendencies in teaching, for example formal authoritarian, demonstrative, suggestive, facilitative, collaborative, delegative (Grasha, 1996). According to Cristea and Garzotto (2004), a teaching style can be problem solving, case-based, progressive, sequential and others. While there is not a particular definition, a teaching style can be described by the teaching methods that a teacher uses, for example, his personal characteristics, the way in which a group of teachers deals with some type of subjects, or the general patterns of classroom behaviour, such as audiovisual or text based pedagogical strategies, examples, simulations, demonstrations and group work used to teach. Then, this data can be included in a Teacher Model that stores the teaching style in terms of the demonstrative, case-based, and audiovisual methods and the characteristics of the examples that a teacher uses. The data stored in the Teacher Model can be used as a basis to chose and recommend examples which characteristics are similar to the characteristics of the examples that a teacher uses.

Our definition of teaching style takes into account aspects such as the class of examples that the teacher uses for teaching. For example, if a teacher uses topologies of high reliability and low cost, these networks can be classified in a particular class, and so the teaching style of this teacher will be associated with this class. This is our focus in model the teacher expertise in teaching a subject. In addition, we establish that a teacher classified in a category has a similar teaching style to that of other teachers in the same category. This is a representation of the teaching style of a category. ARIALE takes into account the teaching style information in the Teacher Model to recommend examples that can be useful for a particular teacher (recommendation is explained in chapter 5).

Updating the teacher model

This initial Teacher Model allows the teacher to start using the Web site, but when the teacher interacts with the system, it will update the Teacher Model according to the new data related to this interaction. In this way, the profiled information is complemented by the statistical information about the teacher's performance. The system collects data related to his performance while monitoring the teacher-system interaction. These data refer only to pertinent variables that the system uses to make decisions in adapting recommendations and help. Variables that the system monitors are the first access to the Web site, the last page accessed and the number of tasks the teacher has done, the sessions he has authored or is authoring, the contents (examples, instructions) that the teacher includes in each session, and the number of times that he has accessed each help concept while authoring a task. This model also keeps information about the class of examples most frequently used by the teacher. With the information about interaction, the system creates an overlay model which keeps track of this interaction. Information from the user interaction is monitored by an interface agent, and the corresponding data is inserted in the overlay model when the system updates the Teacher Model, for example, each time a Web page is unloaded. When a Web page is loaded, the counter of this Web page in the overlay model increases to reflect that the teacher's experience level has increased. When the teacher is authoring a task in a Web page, an interface agent in the client-side monitors the activity (accesses to specific fields), and increases the counter of each help concept accessed to reflect a more detailed change in the teacher's experience level.

By using the overlay model that stores the teacher's levels of experience for the different task, the system adapts to the teacher as he progresses from beginner to advanced user. The user can work with the authoring tool, access help and information, and the system will record that interaction. The teacher's level of experience using the tool is measured by the number of times he has done a task, and the number of times he has accessed each topic of help related to this task. According to the teacher's history of interaction, the system will provide him with different support as his experience increases (Chapter five gives details related to problem-solving support and learning support). The following algorithm describes the process used to update the Teacher Model in the server side when a Web page is unloaded:

```
Procedure Update Teacher Model (){
        Send information about interaction to the server side
        Update teacher's overlay model (history statistics)
        Generate a new plan to follow
        IF a session is finished THEN
                Update the teacher's teaching style
                Update the teacher's category
}
```

4.2.3. Applications of a Teacher Model

According to Encarnação (1997), a Teacher Model, in general, can be used to determine current context, for example, the page where the teacher is authoring a task. A particular application of the Teacher Model is goal recognition and action prediction, which allow the system to define the sequence of tasks and steps that a teacher must follow (Encarnação and Stoev, 1999). Virvou and Mondridou (2002) apply the Teacher Model to personalize help, adapt the system to users, and notify them about new examples in the system knowledge-base.

In our research, the system uses information from the Teacher Model to determine the teacher's expertise, make decisions about updating the teacher's style, selecting the class of examples to recommend, choosing the content and media to provide help according to the working context and teacher expertise, or detecting the plan (sequence of steps in a task) that the teacher might follow. The main function of our Teacher Model is to allow the system to adapt to different teaching styles in order to recommend appropriate examples. Information in the Teacher Model is also applied to provide appropriate help according to teachers' attributes. For example, information in the Teacher Model allows ARIALE to recommend some network topologies and select the right information and format for help messages. The next sections describe two applications of the Teacher Model: the use of the Teacher Model to manage the teachers` teaching styles, and the application of the Teacher

Model in plan recognition. Section 5.2. explains the use of data stored in the Teacher Model in the core of our work: how our system recommends examples and provides help.

Application in Teacher's Teaching Style

In ARIALE, a teaching style is the class of examples that a teacher prefers to use. When the Teacher Model of a teacher is initialized, the style of his category is assigned to him. After a while, however, this original teacher's teaching style can change. This idea of changing is presented in other research related to recommendation (Koychev, 2002; Billsus and Pazzani, 1999). As a consequence, a question to answer is how the system can adapt to this change and learn the new teacher's teaching style. These changes in the teacher's teaching style are updated in the corresponding Teacher Model and, doing this, the system also learns a particular teaching style. This style is deduced from the examples that the teacher includes most frequently in his learning sessions.

The examples used by the teacher illustrate the way he thinks and the way he teaches. This information delineates his knowledge (expertise) about teaching the subject. If the teacher changes this preference, the system has to learn this change in order to adapt and provide appropriate support. A teacher's style changes when the most frequent class of example used by that teacher changes. The focus is on his current style, not on his global style, because the objective is to reflect the teacher's latest tendency. For instance, if a teacher has three sessions classified as "class 2", and then starts creating a session with examples from "class 4" and accumulates four sessions with this class, the style will automatically change from "class 2" to "class 4". After recognizing a change in the teacher's style, the system will recommend sessions that contain examples matching the new style.

Each time a teacher finishes a session, the system updates the information about the most frequently used class of examples in the corresponding category. For example, in the category "Beginner", most teachers prefer examples from the class of example 1; thus, the teaching style of this category is class 1. The following algorithm shows the process that updates the teaching style:

```
Procedure Updating Teaching Style (){
        IF a session is finished THEN
                Classify the index example of the session
                Increase the teacher's counter for the selected class of example

                IF the current teacher's preferred class == the class of the last classified session
                THEN
                        Do not update the teacher's preferred class
                ELSIF teacher's preferred class counter < the teacher's counter for the last selected
                class of example THEN
                        teacher's preferred class = the last selected class of example

                Increase the counter of the same selected class of example in his category

                IF the current teacher's preferred class == the class of his category THEN
                        Do not update the category class
                ELSIF the current category class counter < the counter of the same selected class
                of example in his category THEN
                        current category class counter = the same class of example that the
                        teacher had selected.
}
```

Every time this algorithm is executed, the system might acquire new knowledge about the teaching style of the teacher and his category. This method is important because it recognizes that a teaching style is not determined by an initial diagnosis, but rather that this style is dynamic and evolves over time.

Plans

A fundamental aspect for the recommentation of examples and the adaptive help is the recognition of the plan that the teacher must perform and the corresponding plan representation. We apply a Keyhole plan recognition (Wærn and Stenborg, 1995) that assumes teachers are indifferent to the plan-recognition process and requires less complex recognition mechanisms. Our representation of a teacher's plan in the Teacher Model allows the system to find appropriate help associated with each field in a Web page. In our plan representation, each teacher's plan and each activity, task and step are represented in the Teacher Model by a simple code made up of an eight-character string. We structured this code as follows: two characters represent the Curriculum, two characters are for the activity, the next character is for the task, and the following for the step, and the last two represent a field. Examples of the representation of the Curriculum of Network Design are shown in Table. 4.8. These examples detail the representation of a Network Design course as 01000000; the Reliability activity of this course as 01010000; a step of a task of this activity as 01011100; and, the first field of this step is represented by 01011101.

Table 4.8. The representation of a plan.

Course	*Activity*	*Step*	*Field*	*Meaning*
01	00	00	00	Network (Course 01 is not a specific Web page)
01	00	10	01	Session number 1 (Step 10 is used for Session numbers)
01	00	01	00	Session Step (Step 01 is for Session selection page)
01	00	01	01	Field 1 in Session Web page
01	01	00	00	Reliability activity (it is an activity, not a Web page)
01	01	11	00	Reliability activity, Task1, Step1 (just a Web page)
01	01	11	01	Reliability activity, Task1, Step1, Field1 (a field)

Each *eight-character code* identifies a plan or part of a plan and allows the system to store data related to each activity, task, step, and field (Rodríguez *et al.*, 2004b). The code identifies a record in the table of a database, which includes other data associated with this identification. For example, the code allows the system to identify each field of each page and to store statistics on the accesses to the help in each field. The date of creation or modification of a step can also be stored. When the teacher is following a sequence of steps to create a learning session, this plan representation allows the system to know the last step that the teacher authored to define the next step for the authoring tool. Information related to the next step supports the selection of help resources. If the teacher quits the sequence and jumps to other step, the system will use the plan representation of this new step to make decisions on recommending examples and selecting help. To recommend examples according to the current task (step), the generator and the recommender also take into account the information of the plan. ARIALE uses plan representation and other information in the Teacher Model to support its decisions.

4.3. Conclusion

In this chapter, we described the Pedagogical Model and the Teacher Model that ARIALE uses. Different to classic ITS and AEH that only have a pedagogical approach to teach a subject, our Pedagogical Model includes a part for students and other for teachers. In the section related to the Pedagogical Model, we gave details about a particular method and its strategies designed to teach decision-making expertise to students. According to the method cited previously, learning decision-making skills can start from simple tasks of increasing complexity that iterates until the learner acquires a certain degree of expertise. During this process the learner acquires partial patterns or schemas about decision-making, and gradually masters a very complex schema or expertise about a subject. We also described the strategies and techniques used to support teachers while they are authoring learning sessions. We propose help techniques that are different from traditional help because classic help techniques increase the cognitive load of people working with an application.

In the section devoted to the Teacher Model, we modelled the characteristics of a human teacher and his teaching style (type of examples that he uses most frequently). This model also includes information about his background, preferences, and progress in interacting with the system. A probabilistic method used to infer the teaching style of a new teacher, whether the system does not have any information about him and he needs to start using the system. Instead of using a static Teacher Model, ARIALE includes an evolving model that is applied to track the changes of the teacher's teaching styles. We also designed other applications of the Teacher Model to support teachers in authoring learning sessions by means of our system. The recommendation of cases and other types of help depends on the information contained in the Teacher Model. The next chapter deals with generation, test and recommendation of examples, and other types of support for teachers as well.

Chapter 5. Support for teachers

Summary: this chapter describes the methods that our system ARIALE (Authoring Resources for Implementing Adaptive Learning Environments) uses to adapt support to the teacher's teaching style, attributes and plans. A first section explains the methods for problem-solving support that include the automatic generation and test of topologies, the recommendation of topologies and sessions, and how ARIALE learns the type of examples that each teacher prefers. The generation and test of topologies are based on a probabilistic model, and the recommendation is based on Bayesian learning and on case-based reasoning. Bayesian learning is a kind of classification learning technique of artificial intelligence that uses probabilities. This chapter also defines the design of the learning support that complements the problem-solving support to reduce cognitive overload problems caused by traditional help techniques.

5.1. Complementary methods

While the application of intelligence or adaptation in help, namely desktop on-line help, Web-based help, on-line documentation or any help system, is not commonly used, the use of artificial intelligence techniques for adaptation in help systems is not frequent (Delisle and Moulin, 2002). Thus, we designed an intelligent help system that uses artificial intelligence techniques to provide Web-based help for teachers using our authoring tool. This system applies knowledge base adaptation to profit the evolving data collected from the interaction between teachers and ARIALE. Our assistant, ARIALE, helps teachers with problem-solving support and learning support. Both types of support addresses the critical teachers′ problems studied in this research. This chapter presents the general design of our intelligent help system that involves the recommendation of learning sessions and examples, explained in section 5.2, as the most important of the help techniques studied in this research. This technique is an implementation of problem-solving support and recommends topologies that the teacher can include in his teaching, when he does not have enough time to create an example from scratch. The learning support that we designed gives a frame of adaptive and context-sensitive help for the interaction between ARIALE and the teachers. Learning support is a way to help teachers who do not have enough time to learn the use of a particular tool. Section 5.3 refers to the role of this type of support in this research.

5.2. Providing examples

In this section we answer one of the main questions formulated in the introduction of this dissertation (chapter one): How can the system generate and recommend examples to support teaching decision-making?

While a teacher is authoring a session, he has to include examples that students will analyze. The teacher can create examples from scratch, or he can reuse some of his own old examples or he can receive examples recommended by the system to save time. In this section we explain how the system generates, tests and recommend examples that teacher can reuse to save time when he does not have enough time to create examples from scratch. Recommending examples is a help technique that is part of an intelligent help system that supports teachers while they are authoring learning sessions.

According to the approach of our system for teaching Network Design, an example is a network topology with different number and types of links, no more than seven nodes, no more than 20 links and three basic measures of performance (reliability, capacity and cost). Figure 5.1. shows an instance of a network topology with 17 links. The structure and representation of a topology and the calculation of the measures of performance are explained in the next section. A subsequent section will explain how the system generates and recommends topologies.

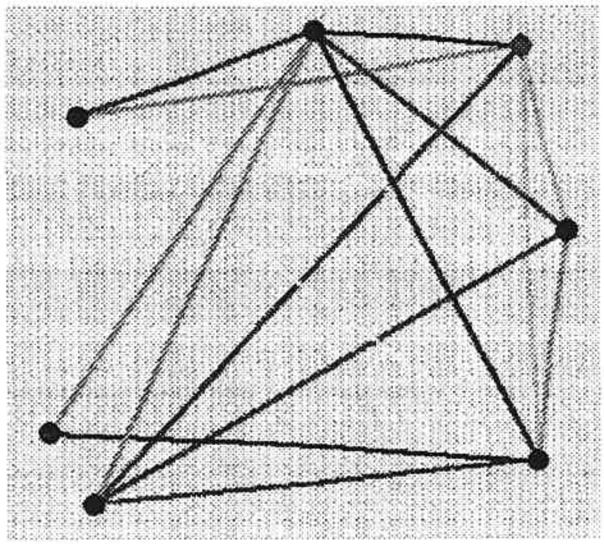


Figure 5.1. An example of a topology.

5.2.1. A topology representation

In ARIALE, we define a topology as a non-directed graph

$$G = (N, L) \tag{14}$$

Where

N is a set of vertices (nodes) $N = \{n_1,n_2,\ldots n_k\}$

L is the set of edges (links) joining pairs of vertices $L = \{l_1,l_2,\ldots l_m\}$

The system represents graphs by an adjacency list (Shaffer, 1998). The adjacency list is structured as an array of some lists of vertices where the list i includes the vertex j, if there is an edge linking the vertex i to the vertex j. Each adjacent vertex j keeps a vector of attributes (t,e) that contains data about the linked node (e) and the values assigned to the corresponding link (t) between this adjacent vertex j and the corresponding vertex i. Figure 5.2. shows the representation of a network topology. In Figure 5.2., we can see a list of seven nodes that is represented vertically and for each node there is another horizontal list with the nodes to which it is linked. Each node in the horizontal list has a vector of a data with its type of node (e) and the type of link between this node and the node in the vertical list.

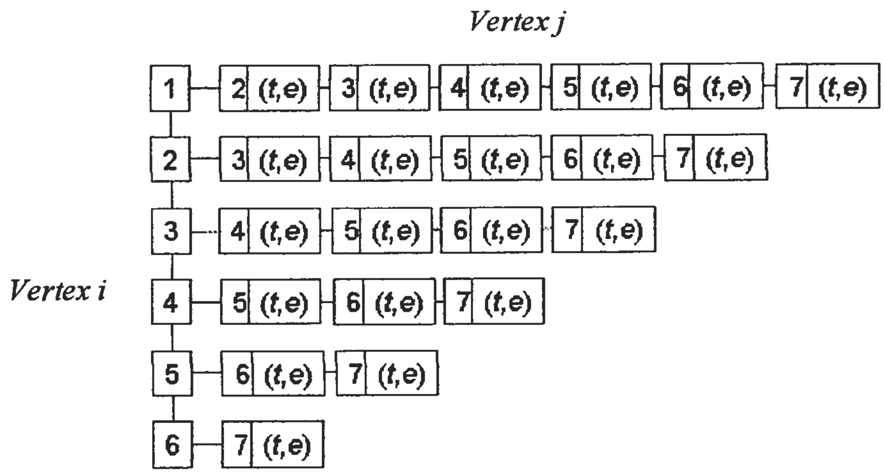


Figure 5.2. An adjacency list representing a topology.

Representation of nodes

Then a node can be represented as

$$n_k = (e) \tag{15}$$

where

n_k is a particular node

e is a type of node

A type of node is defined as the set of attributes

$$e = (s,h,x,y) \tag{16}$$

where

s is an identification number

h is the role of the node (branch or concentrator)

x is the vertical coordinate of the node location on the screen

y is the horizontal coordinate of the node location on the screen

For example, a node can have the following attribute values

s = 2

h = concentrator

x = 1,8

y = 0,2

and they mean that the node number 2 is a bank concentrator located in the coordinates x,y.

Basic representation of links

Each link has some attributes that depend on the type of link assigned. For example, two nodes can be connected by a coaxial link. In this case "coaxial" is the type of link, and this type has default attributes such as cost, distance, capacity and reliability that are defined by the designer of the curriculum. We define a link as

$$L_{ij} = (t) \tag{17}$$

where

i is a node

j is a linked neighbour node

t is the type of link

For example, between nodes $i = 2$ and $j = 5$ there could be a coaxial link (t = coaxial)

A link type is defined as the set of attributes

$$t = (d,k,w) \tag{18}$$

where

d is the distance that the type of link can cover

k is the cost per unit of measure of each medium of transmission

w is the capacity of each link to transmit data. Usually, this information is given in megabites per second (Mbps).

For example, the type coaxial can have a 28-unit distance and each unit can have a cost of 150. The capacity can be 100 Mbps.

Extended representation of links

A teacher can add more attributes to a type of link and, in this case, the representation of a specific link can be

$$t = (d,k,w, g_1,g_2,\dots g_n) \tag{19}$$

where

$g_1, g_2, \ldots g_n$ are personalized attributes added by the teacher, and
the other attributes are similar to equation (18).

An example of an additional attribute can be the maintenance cost.

Because the designer of a curriculum defines default types of links and the corresponding attributes for each type of link, it is not mandatory for an implementer teacher to define each default type of link, but he can change the values of each attribute. The designer also defines the values for the different attributes. In this case, the designer of the curriculum for teaching decision-making skills defined six types of links that each topology can use. Each type of link has some default attributes.

Cost. This is the cost per unit of distance of each medium of transmission. For example, the price of a meter of twisted pair is cheaper that the price of a meter of fiber optics.

Distance. This is the distance to be covered by a medium of transmission. For example, a satellite type of link is supposed to cover more distance, in kilometers, than an infrared connection, which could cover a few meters.

Capacity. The bandwidth capacity for a type of link. For example, a modem-based connection can work with a rate of 56 Kbps and a twisted pair link can have a bandwidth of 100 Mbps.

Defining a topology

Because each link L_{ij} between two nodes has different attributes, it is possible to model a topology a as the list of all the vectors of attributes related to each link. In this definition, we take into account only the default attributes provided by the designer of the Network Design curriculum, as shown in the next equation:

$$a = \{L_{ij} = (\mathrm{t})\} \tag{20}$$

where

a is the topology

L_{ij} is each link connecting node i to node j

t is the assigned type of link.

As a consequence, any topology in our system will be represented according to this definition, and the measures of performance of each topology will be based on the values of the different link attributes.

5.2.2. Assignation of values for network components

This system uses nominal values to represent a range of possible discrete values for each link attribute. Table 5.1. shows the default values for the six types of links used in the Network Design curriculum.

Table 5.1. Default values for types of links.

Types of link	*Nominal values for the default attributes*		
	Cost	***Distance***	***Capacity***
Twisted	Low	medium	low
Coaxial	Low	medium	medium
Fiber	High	long	high
Infrared	medium	short	low
Microwave	High	short	low
Satellite	High	long	high

Based on this table, it is possible to represent each type of link as a vector of pairs attribute/value, as follows:

L_{ij} = <*Cost*/high, *Capacity*/high, *Distance*/long>

An actual link can be represented as a vector of pairs attribute/value, for example:

Fiber = <*Cost*/high, *Capacity*/high, *Distance*/long>

Depending on the type of link, each nominal value maps to a numeric value. These numeric values make possible the easy calculation of some measures of performance (global values) of a network, such as cost. Table 5.2. maps nominal values to numeric values and Table 5.3. shows the static distance that ARIALE uses to calculate the cost of each link. To build this table, we took into account the values for costs and capacity that we found in some suppliers of transmission media in different Web-sites in Internet and we created a fictitious range conserving the ratio between different values for each type of link.

Table 5.2. Default numeric values for types of links.

Types of link	*Numeric values for the default attributes*	
	Cost for unit of measure	***Capacity of each type of link***
Twisted	150	10
Coaxial	150	100
Fiber	1800	5000
Infrared	200*	4
Microwave	600*	10
Satellite	3500*	2000

The costs marked with * do not dependent on the distance of the link because they are defined for a unit of time, let us say, a year-long contract for rental.

Table 5.3. Distance of each one of the 21 possible links of a topology.

Link		**Distance in units of measure**
Start node	*End node*	
1	2	9
1	3	21
1	4	32
1	5	31
1	6	22
1	7	10
2	3	11
2	4	27
2	5	28
2	6	25
2	7	16
3	4	22
3	5	25
3	6	29
3	7	24
4	5	4
4	6	19
4	7	27
5	6	16
5	7	24
6	7	12

A teacher can link two nodes with any of the six types of links available, and default attributes and values of the selected type will be assigned to the actual link between the two nodes. In section 5.2.6.3 we detail the method used to assign types of links when the teacher asks the system to generate a topology.

5.2.3. Measures of performance of a network

As a pedagogical rule, we are using seven-node topologies because analysing larger network models could be a very time-consuming task. Vázquez-Abad *et al.* (2001) maintain that "a seven-node network is large enough that the main concepts and ideas can be illustrated and small enough to keep a reasonable time pace". In our topologies, the seven nodes always have the same position, and they are all connected by links of different types of media transmission, namely coaxial, satellite, or infrared, for example. There could be up to 21 links connecting the seven nodes, but we are not using all-connected or complete graphs because they are unrealistic, never used in real life and do not impose challenges or problems to be solved by students. Each node represents one branch of a bank network as explained in the Pedagogical Model (chapter four). A red node represents a concentrator. A topology also has a background, usually a map of the city where the network is placed.

Because each link between nodes has attributes and values, the editor calculates global network values or *measures of performance* of the network, based on the value assigned to each link. ARIALE calculates some measures of performance that are only indicators used to distinguish some topologies from others. For example, the capacity of a network is measured as the minimum bandwidth of a link, but our system takes into account the bandwidth of all the links in order to get a more general idea of the characteristics of each network. We are using some of the measures of performance for the characterization of computer networks proposed by Pierre and Gharbi (2001). These measures are cost, capacity, number of links and reliability. For example, each link has a cost and a distance value and the system uses these values to calculate the global cost of the network. The global cost is the result of the summation of the multiplication of the corresponding cost by each link distance. Then, we can affirm that measures of performance (global values) are functions of the values of the attributes of the links in a topology. For example, given a network topology a, there are some functions of performance $F_i(a)$ to calculate its measures of performance:

$F_i, \forall i \in \{1,...n\}$

For example, $F_1(\text{cost}, a)$ is the function of performance "network cost".

If $F_1(\text{cost}, a)$ is the global cost of the network a, then an instance of global cost can be $F_1(\text{cost}, a) = \text{High}$. A generalization of this example is

$$F_i(m,a) \in \{\text{Low, Medium, High}\} \qquad (21)$$

where

F_i is a particular function that calculates a measure of performance

a is any topology

m is a particular measure and

Low, Medium and High are the values that the function F_i can return.

Cost

As said before, the system also calculates numeric global values in order to get the measures of performance of the topology. The default values of the attributes of each type of link are defined by the designer of the course. For instance, the function that calculates the global cost of a network is:

$$F_1(\cos t, a) = \sum_{\ell=1}^{m} \left(c_\ell d\right)_\ell \qquad (22)$$

where

d is the distance covered by the link *l* between two nodes

c_l is the cost per unit of measure of each medium of transmission of each link

m is the number of links in the network

Capacity

Capacity is the summation of the bandwidth of each link

$$F_2(capacity, a) = \sum_{l=1}^{m} b_l \qquad (23)$$

where

T is the capacity of the network

B_l is the default bandwidth for a type of link

m is the number of links in the topology

We are aware that the capacity of a network is defined by the minimum bandwidth of its links, but in this case we are using the summation not as a precise measure but rather as an indicator of the general dimension of the network.

Number of links

The total number of links is another global indicator that identifies a network. In our system, there are networks that have from seven to 20 links. A seven-links network is a typical ring and a 20-link network is a high reliability network (we do not use all the 21 possible links and we do not use more than one link between each pair of nodes). Formally, the computation of the number of links is done by the following equation (24)

$$F_3(links, a) = \sum_{l=1}^{20} l_l \qquad (24)$$

where

l_l is each existing link of the 20 possible that each network can have.

Reliability

In this research, we define reliability as the probability that the network is functional because all the nodes of the network are connected by operational links. This means a network is reliable if it can continue functioning although some node or link fails, and this definition is linked to the concept of connectivity (Pierre and Gharbi, 2001; Pierre and Elgibaoui, 1997). We are using connectivity degree as an indicator of network reliability in terms of "the minimum number K of disjoint routes over all the node pairs" (Pierre, 1998). A network is 2-connected (k = 2) if between each pair of nodes there are at least two disjoint paths. A network is 5-connected (k = 5) if between each node pair there are at least five disjoint paths. In Network Design, the minimal acceptable *k*-connectivity is $k = 2$ and a high degree of connectivity (reliability) is $k = 5$. In order to calculate the *k*-connectivity, we have applied the Ford-Fulkerson algorithm as proposed by Obraczka *et al.* (1997):

```
Procedure Check_feasibility (topology)
    {
    for nodes Ai, 1 <= i <= k
        {
        for every other node B
            {
            if ( disjoint_paths(Ai, B) < k )
            then return(FALSE).
            }
        }
    return(TRUE).
    }
```

```
Procedure disjoint_paths(G, s, t)
{
        initialize residual network R to graph G.
        while there exists a flow f from s to t in R
        {
        extract f from R.
        number_of_paths++.
        }
return paths.
}
```

This algorithm calculates the number of disjoint paths between each pair of nodes in a graph, and then it is only necessary to identify the pair with the minimum number of disjoint paths to detect the *k*-connectivity of the network. Finally, we can say that

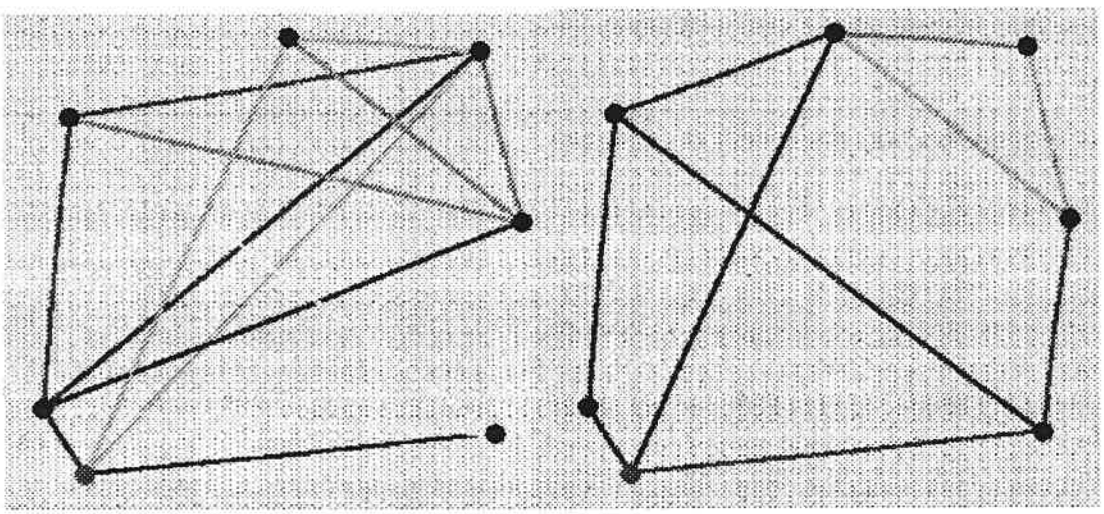
$$F_4(realiability, a) = k\text{-connectivity} \tag{25}$$

5.2.4. Examples and counter-examples

An *example* is a topology to be studied *alone* in tasks 2 or 3, according to our pedagogical model (section 4.1.1. describes the different tasks). Given an example A, a *counter-example* is another topology B that must be *compared* to A, in task 1. Both examples and counter-examples have the same number of nodes. Examples are topologies that have equivalent measures of performance; the number of link can or not be the same. Equivalent examples are classified in the same class. Given an example, a counter-example is another topology classified in the same class, which has measures of performance that are also equivalent to the given example, but the number of links **has** to be different. Note that an example can play the role of "counter-example". The topology A in Figure 5.3. is a network that can be used alone as an example in tasks 2, and the topology B (see also Figure 5.3.) can be used as another different example in task 3. In tasks 2 and 3 each topology is displayed as a single example of a network. However, if a teacher wants to use the topology A in task 1, then the topology B can play the role of counter-example because they are in the same class but have different number of links. As a consequence, ARIALE might recommend the topology B as an example for tasks 2 or 3, and as a counter-example in task 1. Table 5.4.

Table 5.4. Measures of performance of network topologies A and B

Measures	*Topologies*	
	A	B
Reliability	3	3
Capacity	17258 (Medium)	16134 (Medium)
Cost	128400 (Medium)	122450 (Medium)
Number of links	14	11



A B

Figure 5.3. Two topologies that can be an example (A) and its counter-example B.

The creation of examples and counter-examples is a time-consuming task for teachers; as a consequence, our system supports teachers in two ways by:

- Providing tools and resources to create examples
- Recommending examples to be edited or reused.

This last type of help offers learning sessions that the teacher can reuse instead of starting from scratch. For example, when a beginner teacher, who has never authored a learning session, starts using the authoring tool, the system offers an existing learning session to reuse. If the teacher does not like the session that the system recommends, the system provides additional tools and resources for authoring examples that will be part of the session.

In the next sections, we explain how a learning session is composed of examples, as well as how the generation and the recommendation of sessions and examples is performed.

5.2.5. Session and examples

As we explain in chapter four, a session is a sequence of steps authored by an implementer teacher. While a teacher is authoring a session, he includes the examples he wants the students to analyze.

Depending on each task, the definition of an example is different. In the case of task 1, an example is the set of an example and its counter-example. In task 1, the teacher has to author the comparison of a given topology with another one. Both topologies must have equivalent measures of performance, but a different number of links. In task 1, the teacher has to create the example and the counter-example. The teacher has to configure each topology (example and counter-example) by defining links between nodes and the location of the network concentrator. The role of the system is to suggest examples and counter-examples in order to save teacher's time.

For task 2, an example is a topology that uses only one type of link. In the first step of this task, the teacher has to create the example of the topology by assigning the same type of link to each connection between nodes. He can create a new topology or reuse the topology of one of the examples used in task 1. After that, the teacher writes an instruction asking the student to add or delete a link, change the position of the concentrator or change the capacity of a connection between two nodes in order to visualize the effects of the changes. In a second step, the teacher reuses the example of step 1 and asks students to execute an action that increases a network measure of performance. For example, the teacher can write an instruction asking students to add any link that improves the reliability of the network.

In task 3, the example is just a topology that the teacher has to create. Of course, he can reuse a topology from previous tasks. In addition, the teacher has to specify the goal that students have to achieve. For example, the teacher can propose a certain global cost be achieved and, in order to get this goal, the students have to change links, move the concentrator to different locations or vary the capacity of some links.

In conclusion, a session is also the set of all the examples and counter-examples used in the different tasks of the different activities of a particular curriculum.

A teacher can use two examples in Task 1, and he can reuse one of these examples in Task 2 and 3. Of course, a teacher could use different examples in each task, so a learning session will need anywhere from two to five examples. Each session is a set of examples and counter-examples that reflect the teaching style of its author.

5.2.6. A recommender

This section explains how the system recommends sessions and examples. Our system has a recommendation module that solves the following problems:

- How does the system guess which sessions and examples might be useful for the teacher if he has not done a session or example?
- How can the potential cognitive overload be reduced for a teacher if examples are filtered for recommendation?

5.2.7. Classes

According to this research, recommendation is based on the classification of examples that the teacher uses in teaching decision-making skills. As a consequence, the designer of the course has to define the classes that the Bayesian classifier requires to place each new example in the respective class. In our case of Network Design, our classes are based on the characterization of computer networks proposed by Pierre and Gharbi (2001). According to this characterization, response time, reliability, availability and cost are some of the most usual indices for measuring the performance of a network. We decided to use reliability, capacity, cost and number of lines as the four indices to represent and classify networks in our system. The calculation of these indices was explained in section 5.2.3 and the process to classify examples is explained in the next sections. Here, we define the classes and the criteria used to create them.

Each class includes examples with indicators that are the measures of performance of each topology and the number of links connecting the nodes of a topology. According to the values assigned to each type of link (see section 5.2.2.) and by applying the equations also explained in that section, we obtain the values corresponding to the measures of performance capacity, cost, reliability and the number of links. Some examples of measures of performance are shown in Table 5.5. Because the measures of performance related to capacity and cost are too varied, we decided to group each one in three categories (High, Medium, Low). These categories allow the classifier to deal with fewer possibilities (high dimension). Table 5.6. shows a conversion table from numeric values to nominal values in the cases of cost and capacity measures of performance.

Table 5.5. Examples of measures of performance of some topologies.

Reliability (R)	*Number of links (L)*	*Capacity (Ca)*	*Cost (Co)*
2	7	11014	67100
2	10	19492	174100
5	20	7590	56100
4	16	16154	91200
3	11	16134	122450
3	15	33128	227850

Table 5.6. Details of the categories of measures of performance capacity and cost and the ranges of their values.

Nominal values	*From*	*To*
High capacity	Higher than 30001 kbsp	
Medium capacity	10001 kbsp	30000 kbsp
Low capacity	0 kbsp	10000 kbsp
High cost	200001 units	400000 units
Medium cost	100001 units	200000 units
Low cost	0 units	100000 units

Using the nominal values detailed in Table 5.6., ARIALE uses six classes to classify examples of network topologies. We defined the six classes on the basis of an analysis of the empirical distribution of 200 topologies that our system generated automatically. This analysis was based on the previously cited characterization of computer networks (Pierre and Gharbi 2001) and on the advice of the expert in network design, Dr. Samuel Pierre, professor of the École Polytechnique of the Université de Montréal. For the pedagogical use in this research, we give the priority to the reliability and, after that, to the capacity and the number of links of each topology. Three classes of networks in our set of 200 topologies have high reliability (k = 4 and k = 5 connectivity) but different cost, capacity and number of links. Two other classes have low reliability (k = 2 connectivity) but the other indicators are dissimilar. A last class has k = 3 connectivity and the other measures are very close. Table 5.7. shows some examples of the main characteristics of the classes.

Table 5.7. Examples of general characteristics of the classes used in our system.

K-connectivity	Capacity	Cost	Number of links	Class
4 or 5	Low	Low	More than 13	1
4 or 5	Medium	Low	More than 13	1
4 or 5	Medium	High	More than 13	2
4 or 5	High	High	More than 13	2
2	High	High	More than 6	2
4 or 5	Medium	Medium	More than 13	3
4 or 5	High	Medium	More than 13	3
3	Low	Low	More than 11	4
3	Medium	Low	More than 11	4
3	Medium	Medium	More than 11	4
3	Medium	High	More than 11	4
3	High	High	More than 11	4
2	Low	Low	More than 6	5
2	Medium	High	More than 6	6
2	Medium	Low	More than 6	6
2	Medium	Medium	More than 6	6

These six classes are the basis for the functionality of the recommendation module in ARIALE. Some actual examples belonging to the six classes that ARIALE uses are shown in Table 5.8.

Table 5.8. Examples of topologies classified according to their measures of performance.

k-connectivity	Capacity	Cost	Number of links	Class
5	9514	45450	20	1
4	11366	83450	17	1
5	31268	234850	19	2
4	31158	234100	17	2
5	23460	200750	18	2
4	17568	176550	18	3
4	14536	115350	16	3
3	26268	163150	18	4
3	10428	78850	11	4
2	7308	68550	8	5
2	2262	14750	12	5
2	20124	218900	9	6
2	16124	92450	10	6
2	12114	102700	7	6

5.2.8. Functionality

Based on the classes described in the previous section, the recommendation module in our system performs the following functions:

- Recommending sessions
- Recommending examples

Our system uses a hybrid method to recommend sessions: one is based on the teacher's characteristics and the other takes into account the examples that the teacher uses to teach.

First, to solve the *cold-start* problem (Schein *et al.*, 2002) when the teacher has not authored any session and example, the system uses a *collaborative filtering* method that takes into account the similarities between teachers according to their teaching styles. This method allows the system to recommend learning sessions created by teachers with similar backgrounds and preferences. The system believes that teachers in the same category can also share a teaching style or pattern. The sessions are classified according to the examples used in them. In order to recommend sessions, the system detects the type of sessions that teachers in a particular category use. After that, the system recommends sessions of this type to the new teacher. The classification of teachers is explained in section 4.2.2, the Bayesian method for classification is detailed in section 2.2.7 and the application in recommendation is detailed in the next sections.

Second, when the teacher has created at least one session, the system uses a content-based filtering method that looks for session with examples that are similar to the example used by the current teacher in his session or sessions. Filtering examples by taking into account the preferred examples of a teacher allows the system to recommend sessions and examples that are close to his current teaching style.

Adaptive generation according to the teaching style

This kind of recommendation is dynamic because it adapts to the changes of the teacher's teaching style. Instead of taking into account just the information that the teacher provided explicitly during the registration, the system relies on the examples used by the teacher. If the teacher changes his preferences for another type of examples, the recommender adapts its recommendations to match the new class of examples and forgets the old ones. This forgetting means that ARIALE always switches to the teacher's new trend.

5.2.9. Recommendation of sessions and examples

There are four main scenarios for recommendation where the functions enumerated above are applied:

- When a teacher does not have any session or example done and there is not enough information to support decision-making about what session to recommend.
- When a teacher prefers to start from scratch and there is not enough information to support decision-making about what example to recommend.
- When the system knows the teacher's preferred class of examples.
- When the teacher does not want to accept any recommendation and ask the system to generate new examples, based on enough or partial information available.

Let us see these scenarios in detail.

First scenario

If a teacher is supposed to author a learning session and he does not have any session or example done (cold-start), the system tries to guess and recommend some sessions that could be useful for him.

The method used to define and recommend a session is as follows: from the Teacher Model, the system retrieves the class or category into which the teacher is classified. With this information, the system also retrieves the class of example that the teachers in the same category use most frequently.

After detecting this most frequently used class of examples, the system selects sessions that contain examples of the same class. Because in this case the teacher is a beginner user, the recommender offers one of the sessions retrieved. If he rejects the offer, a list of recommended sessions remains available in case the teacher changes his mind and accepts to reuse a session. This algorithm synthesizes the decision-making process of the recommender in this case:

```
Procedure Recommend a session () {
        IF the teacher does not have any session done THEN
                Retrieve the teacher's category
                Retrieve the most frequently used class of examples in the category
                Retrieve sessions that contain examples of this class
                Recommend a session to start editing

                IF recommended session is accepted THEN
                        Load the recommended session
                ELSE  display a list of recommended sessions

        ELSE  Define the class of example most used for this teacher
                Recommend sessions that have examples of this class

}
```

The recommendation of sessions occurs when a teacher accesses the session selector Web page. If the teacher does not want to receive recommendations because he prefers to have access to all the sessions and examples available, he has to adapt the system to this requirement, by editing his preferences in his Profile Manager Web page.

Second scenario

A teacher prefers to start from scratch, but there is not enough information to support the decision-making about which examples to recommend. Then, the solution is that the system follows the same process as in the first scenario, but instead of recommending complete sessions, it recommends examples as part of the Load function of the example editor. The example editor is an application (a Java applet embedded in a Web-page) that allows the teacher to configure a network topology by adding or removing link between the nodes of a network. This editor also has functions to change the type of link and other characteristics of links and nodes. The implementation of this editor is explained in chapter six. In this case, instead of proposing complete sessions, the recommender just includes in the example editor a list of seven examples that are in the same class of examples preferred by teachers in the same category of the current teacher.

```
Procedure Recommend an example () {
        Retrieve the teacher's category
        Retrieve the most frequently used class of topology in the category
        Retrieve topologies from this class
        Display a list of recommended topologies
}
```

If the teacher does not want to use some of the recommended examples, on demand, the generator generates an example that belongs to the teacher's class. The algorithm for generation of topologies is described in the part that explains the automatic generation of topologies, later in this section. We decided to use a list of seven recommended topologies because it is easier for a beginner teacher to choose a topology from a small number of options instead of from a set of 30 or 50 examples accumulated by many teachers (Mayer and Moreno, 2003). The seven topologies selected are the newest ones in the database, because they represent the other teacher's trend. A teacher could increase the number of displayed examples by tailoring manually the respective configuration in his profile.

The hypothesis behind the recommendation methods used in these scenarios is that the sessions and examples the teacher could use may depend on the ones used by teachers classified in the same category previously. If the system knows which examples have been used by other teachers of the same category, then the system can recommend using such sessions or examples. This is a combined application of the collaborative filtering and content-based filtering.

Third scenario

When the system knows the teacher's preferred class of examples, the task for the recommender is easier because it only has to retrieve the class of example that the teacher uses more frequently and, with this information, the recommender retrieves a list of sessions or topologies belonging to this class. When the teacher is choosing a session in the session selector Web page, the recommender provides complete sessions; when the teacher is using the example editor, the recommender supplies examples. The recommender uses the same sequence of steps: retrieving the teacher's example class, retrieving a list of sessions or examples, and displaying the list. The hypothesis behind the recommendation method used in this scenario is that the examples the teacher will use in the future depend on the ones used previously. Because the system knows the class of examples the teacher has used in the past, it can recommend using sessions or examples from the same class. In this case, the system only applies content-based filtering for recommendations without taking into account information from other teachers.

Fourth scenario

If a teacher has not authored any sessions or does not want old sessions created by other teachers, the question is how to generate an example from scratch in order to offer it to the teacher. Our system has a particular module, the Generator, which creates examples. This module has two functions:

- *Generate examples* according to the criteria defined by the designer.
- *Verify that any* generated topology fits in the class of examples used by the teacher (the teacher's teaching style). If the generator creates an example that is out of the scope that the teacher prefers, the example is aborted and a new one is generated.

Generation process

In order to generate examples, the system uses a probabilistic method. For each one of the seven nodes of the topology, this method establishes links with the other nodes; for example, node 1 can have links with nodes 2, 3, 4, 5, 6 and 7. The type of link between each pair of nodes is taken from the set of available links provided by the designer of the curriculum. For each topology a, each link L_{ij} has a type of link t, so we can reuse equation (12) to represent a topology as a set of nodes, each one with a type of link assigned.

$$a = \{L_{ij} = (t)\} \qquad (12)$$

At random, the generator selects a type of link for each pair of nodes or it does not assign any link to represent that no connection exists between those two nodes. In this way, it is possible to generate a network with a "ring" topology of seven nodes and seven links or any other configuration from seven up to 20 links. According to the designer, each type of link has a set of attributes. The designer also establishes the values of each link attribute, as explained in section 5.2.2.

The measures of performance of each topology or example change according to the distribution of types of links. For example, if a topology has a high percentage of high cost links, consequently, the function of the performance "cost" of this example will be high. Table 5.1. illustrates default values used for different types of links in this system. According to this table, if a topology uses a great quantity of fibre links, the network may cover long distances with high cost and high capacity.

The generator creates examples using the following algorithm:

```
Procedure Generate examples () {
        Given 7 nodes and a set of types of link
        M=<twisted, coaxial, fiber, infrared, microwave, satellite>
        WHILE the example does not belong to the teacher's class of examples nor matches his
        preferred measures of performance DO
                FOR each link Lij connecting node i to node j DO {
                        At random, select a type of link from M
                        or do not assign any link
                        Assign the selected type of link to Lij
                }
                Locate a concentrator at random
                Calculate the measures of performance
                Verify the measures of performance match the measures of
                performance that the teacher prefers
                Verify the example belonging to the teacher's class of examples
        END of WHILE
        Display the topology in the editor
}
```

Test

The verification of each example is a process that requires classifying a generated example in the teacher's preferred class of examples. The system keeps records of the examples that each teacher uses or has used most frequently, as explained previously in this section. The classification of an example implies calculating the probability of the example a belonging to the class of examples that the teacher prefers. Given

$a \in A$

where

a is an example

A is the set of all the acceptable examples

ARIALE uses a naïve Bayesian classifier (see also section 2.2.7.) to calculate this probability. The classifier is intended to estimate $P(a_i \mid class)$, the probability of a specific example a_i, to belong to a particular class of topologies. As this calculation is difficult, thus we use the probability of a specific class given the evidence a_i.

$$P(class \mid a_i) = P(class)P(a_i \mid class) \tag{16}$$

where

class is the class of examples for which the generator is testing to see if the current example belongs to it.

a_i is the evidence

A network topology has four indices, and any example a_i can also be represented as a vector of four indices:

$a_i = (r, m, b, c)$

where

r is the network reliability

m is the number of links

b is the total network capacity

c is the network cost

For example, a given topology a_1 can be represented as a vector of its measures of performance and number of links. One example could be a network with the following values for each measure of performance:

Reliability = 2

Number of links = 11

Capacity = 21214

Cost = 144350

Then, the representation of this example is the vector

$a_1 = (2,11,21214,144350)$

and we have to check in which of the classes described in section 5.2 this example can be classified. According to Table 5.5., we are representing the capacity and cost values with nominal values such as "low", "medium" and "high". Because the system basically requires checking if a topology belongs to a particular class or not, we are using here a two-class classification to explain how the Bayesian classification classifies topologies. The two classes are "A" and "B" and the example to be classified is the one detailed previously. For this explanation we use the following training dataset shown in Table 5.9.:

Table 5.9. A training dataset.

Id	*Reliability (R)*	*Number of links (L)*	*Capacity (Ca)*	*Costo (Co)*	*Class*
1	2	7	High	High	A
2	2	16	High	High	A
3	2	9	Medium	High	A
4	2	14	Medium	High	A
5	2	11	Medium	Low	A
6	2	12	Medium	Low	A
7	2	10	Medium	Medium	A
8	2	13	Medium	Medium	A
9	3	15	High	High	A
10	3	17	Low	Low	A
11	3	18	Low	Low	A
12	3	15	Medium	High	A
13	3	16	Medium	High	A
14	3	11	Medium	Low	A
15	3	13	Medium	Low	A
16	3	17	Medium	Medium	A
17	5	19	High	Medium	B
18	4	18	High	High	B
19	5	19	High	High	B
20	5	18	Medium	High	B
21	4	18	Medium	High	B
22	4	18	Medium	High	B
23	4	19	Medium	High	B
24	4	19	High	Medium	B
25	5	18	Medium	High	B
26	4	19	Medium	High	B
27	4	19	High	High	B
28	4	19	High	High	B
29	4	18	High	High	B
30	5	19	High	Medium	B
31	4	19	High	High	B
32	4	19	Medium	High	B
33	4	17	High	High	B
34	5	20	High	High	B
35	4	18	Medium	High	B

We must calculate the probabilities for each measure of performance. In order to make the comprehension of the calculations easier, we use the following letters between parentheses to identify each class and measure of performance:

Class A (A)

Class B (B)

Reliability (Re)

Number of links (Li)

Capacity (Ca)

Cost (Co)

Given the example to be classified and the training dataset, for the measure of performance, for example in the case of *cost*, it is necessary to calculate its different probabilities of appearing in each class. Formally,

$$P(Co \mid class) \qquad (17)$$

where

Co is the value of the measure of performance Cost

class is the class of examples for which the generator is testing if the measure of performance is included

P(Co|*class*) is the probability that *class* presents *Co* as evidence.

For each value in each position of the vector, the naïve Bayesian classifier uses equation (11) to calculate its probability of belonging to each class of examples. Equation (18) shows equation (11) adapted to calculate the probabilities of types of link

$$P(f_i \mid v_j) = \frac{n_c + mp}{n + m} \qquad (18)$$

where

f_j is each value of each measure of performance (Re, Li, Ca, Co) in the vector for each class,

v_j is each class

n is the total number of instances in each class v_j

n_c is the number of instances with attribute f_i and class v_j

p is a priori estimate of $P(f_i|v_j)$ for each f_j. We assume that the probabilities of all attributes are equiprobable (equally likely to be true). For example, for the attribute "reliability" there are four possible values and $p = 1/4$.

m is a constant used to avoid the possible consequences that could arise if $n_c = 0$ (in this case the calculation would be zero). We use m = 4 as a constant because we have four attributes in the examples of our training dataset.

This equation is applied to calculate the probability that each class include the given topology. Table 5.10. shows the corresponding values for the calculation of probabilities using equation (18).

Table 5.10. Values to calculate where the topology can be classified.

Class A	**Class B**
Re (2) $n = 16$ $nc = 8$ $p = .25$ $m = 4$	Re (2) $n = 19$ $nc = 0$ $p = .25$ $m = 4$
Li (11) $n = 16$ $nc = 11$ $p = .08$ $m = 2$	Li (11) $n = 19$ $nc = 8$ $p = .08$ $m = 2$
Ca (Medium) $n = 16$ $nc = 3$ $p = .33$ $m = 4$	Ca (Medium) $n = 19$ $nc = 3$ $p = .33$ $m = 4$
Co (Medium) $n = 16$ $nc = 2$ $p = .33$ $m = 4$	Co (Medium) $n = 19$ $nc = 0$ $p = .33$ $m = 4$

Table 5.11. shows the corresponding calculation of the probabilities of each measure of performance of the given example for each class.

Table 5.11. Probabilities for each attribute for each class.

Class A	**Class B**
$P(\text{Re}\vert A)=\frac{8+4*.25}{16+4}=0.2$	$P(\text{Re}B)=\frac{0+4*.25}{19+4}=0.04$
$P(Li\vert A)=\frac{2+4*.07}{16+4}=0.12$	$P(Li\vert B)=\frac{0+4*.07}{19+4}=0.01$
$P(Ca\vert A)=\frac{11+4*.33}{16+4}=0.62$	$P(Ca\vert B)=\frac{8+4*.33}{19+4}=0.04$
$P(Co\vert A)=\frac{3+4*.33}{16+4}=0.22$	$P(Co\vert B)=\frac{3+4*.33}{19+4}=0.19$

After these calculations, it is possible to multiply the probabilities of each measure of performance to appear in a class in order to obtain the global probability of the vector to belong to a particular class. The maximal probability obtained determines to which class the topology can belong, according to equation (10) in section 2.2.7., which we adjust as follows for the current example

$$v(E) = \underset{v_j \in \{A,B\}}{\arg\max}\, P(v_j)\prod_i P(a_i|v_j) \qquad (19)$$

where

{A,B } is the set of the different classes to classify examples, and

E is the example to be classified.

This calculation can also be performed as follows:

P(A)*P(RE/A)*P(Li/A)* P(Ca/A)* P(Co/A) = 0,472 * 0.2 * 0.12 * 0.62 * 0.22 = 0,001456019

P(B)*P(Re/B)*P(Li/B) *P(Ca/B) *P(Co/B) = 0,527 * 0.04 * 0.01 * 0.41 * 0.19 = 0,000023470

Finally, the classifier has a prediction for each class. The maximal number identifies the class to classify our vector ***a***. In this example, the class of the given example is "A" because 0,001456019 > 0,000023470.

In this case, the generated example is recommended because we had assumed that the teacher prefers the class "A".

If the example is displayed to the teacher, he can accept or reject it. If the teacher does not like an example generated by the system, the system can prompt the teacher to edit the offered example until he is satisfied with it. If the teacher does not want to edit it, the generator continues to produce examples until it gets one that the teacher accepts. This validation method is intended to recommend examples that fit the specific teacher's teaching style. Once an example is saved as part of a learning session, the example is classified and included in the knowledge base of the system.

The generator not only generates examples but also counter-examples for Step 3 of Task 1. In this case, the system generates a topology that does not belong to the teacher's class of examples, but has the same measures of performance.

Bayesian learning is a method that allows the system to learn new input in order to anticipate a teacher's request for support. Each time a new example is included in the database, the system learns from it. ARIALE applies a type of knowledge acquisition that models each teacher's expertise in teaching a subject. This method allows the system to acquire the teaching styles (teaching pattern) of teachers. There are more reasons to prefer Bayesian learning in this case. The Naïve Bayes algorithm has *no tuning parameters*. In contrast, tunning a k-NN classifier (see section 2.2.7) according to other topic characteristics and parameters requires a lot of work and tests. For example, a k-NN classifier requires adjusting *the weights* of each attribute properly in order to generate good results. Instead, Bayesian classifiers are more *flexible and easy to adapt* to different domains and number of attributes. For example, predicting the class for a topology or any other instance requires counting only the frequency of attribute values within the training examples, *only*.

5.2.10. Session base and example base

According to Case-Base Reasoning (CBR), a case is conceived as an initial state S_1, a final state S_n, and the solution or plan that facilitates passing from one state to the other (Cortés *et al.*, 1994). Our cases, however, are not modeled in this manner because each session is a case made up of many examples, instructions and additional parameters that the students will take into account. Each case is a possible solution to the problem of authoring a session. By storing cases in the case-base, the system can find a set of examples with good probabilities to be recommended as worked-out examples. The teacher is in an initial state with no session on hand, and the system helps him by providing a session in order to arrive at the final state: to have a session according to the teacher's preferences.

The structure of a learning session includes:

- The index. Each session is identified by a number, its creator or owner, and the class of an example. We use the example of Task 1. The counter-example and the other examples are not needed to identify the session.
- The additional components. This part includes the counter-example of Task 1 and the examples of the other tasks.
- The instructions. This part includes the instructions provided to the students and the values of the goals they will have to achieve during the competition (Task 3 for optimization).

Each example has the following structure:

- The index, which identifies each example by a number, its creator or owner, and the corresponding class.
- The representation of the internal structure, which includes the number of nodes, number of links, the type of link assigned to each link (this is another complex structure including several attributes and the corresponding values), and a set of other attributes and values (measures of performance) whose number can vary.

When a teacher creates a session in which the index is similar to that of another session, the teacher will be able to keep the session for his own use. A session is also fundamental because every time a teacher finishes a session, the system checks whether the teacher's teaching style has changed.

The reasoning behind the recommendation mechanism of our system is based on CBR, and the hypothesis is that the examples the teacher will use in the future depend on the ones used previously. If the system knows which examples the teacher has used in the past or which have been used by other teachers of the same category, it can reasonably choose examples that the teacher will probably be able to use. With this view, we are also taking advantage of the CBR filtering approach (Funk and Conlan, 2003) in order to filter and recommend examples according to the teacher's characteristics.

5.3. Learning support for teachers

This section describes the context of the recommendation of examples and the design of the learning support that complement problem-solving support. We also answer these questions formulated in the introduction (chapter one):

- What can be the general structure and functionality of an assistant to support teachers authoring learning session in decision-making?
- How does the system make decisions about which kind of help content to show, and which media to use for displaying the content?

The general answer to those questions is that an assistant must have the structure and functionality of an Intelligent Help System (IHS) that provides learning support and problem-solving support. The aspects related to problem-solving support were discussed in the previous section 5.2 devoted to generation, test and recommendation of examples. In the next sections, we discus the following series of more specific problems related to learning support and the questions above:

- The cognitive overload and the production paradox that could be experienced by teachers.
- The access to the help.
- The adaptation of the help.
- The interaction between the teacher the help system.

Closing this chapter, we explain how and why a knowledge-base is used to support the learning of the system.

5.3.1. Cognitive overload and production paradox

In general, users of software and, particularly, authors of educational material such as teachers face two difficulties:

1. The teacher is not familiar with the functionalities of the tool he is going to use, he has only a limited potential use of the tool and thus he can not properly use many of the options that the tool provides. As a result, his performance using the tool is low. He can not improve his skills using the tool because he does not have enough time to

learn how to use the tool, but, at the same time, he can not learn because he needs the time to perform the tasks that he has to do with the tool. In addition, most of the authoring tools to create ITS do not provide interactive support to exploit their functionalities. This paradox is the *production paradox* (Leung, 2001; Papanikolaou *et al.*, 2003), and it is a typical contradiction that many users of software face every day.

2. A potential cognitive overload could affect teachers using an authoring tool if they split their attention between authoring, finding, and selecting help according to his characteristics to improve their use of the tool. For example, if a teacher is involved in (1) planning a learning session, (2) manipulating the authoring tool to create the session, and (3) selecting help according to his needs and preferences, his mind would be involved in three processes at the same time: teaching (which is the main task), manipulating the authoring tool, and dealing with the help features.

In section 2.2, we discussed the concepts and definitions related to Human-Computer Interaction (HCI) and Artificial Intelligence (AI), which are commonly applied in intelligent help systems. Some of those concepts are traditional principles and techniques of on-line help, which are not useful in our case because they provoke cognitive overload and increases the production paradox. As seen in chapter three (Table 3.1.), intelligent help, automatic generation of teaching materials, automatic completion of tasks and direct support for teachers, are different types of help used by other systems to improve the performance of teachers using ITS and EAH.

The option we found to reduce the cognitive load and the production paradox in ARIALE is a help system that works parallel to the authoring tool, and finds, selects and adapts *context-sensitive* help according to a teacher's skills, preferences and current task. This help system focuses on the manipulation of the help to free the teacher's mental capacity and allow him to pay better attention to the translation of his plans in the authoring tool. The help features are one way to support the manipulation of the authoring tool and decrease the cognitive load generated by the interaction with this tool. We preferred this option in comparison to traditional help techniques without the application of artificial intelligence. For example, automatic generation of teaching materials is not usually adapted to each teacher's characteristics and style (Virvou and Mondridou, 2002; Kinshuk *et al.*, 1999). Another problem is that those types of help mentioned above are not frequently integrated to be complementary as in ARIALE. To discriminate our option from other possibilities, such as simple sharing of examples or finding help topics in a traditional manner, we analyzed the advantages of integrating adaptation and different help features, and the novelty of doing it in a Web-based help system. An important aspect to distinguish our option is the integration of the generation, testing and recommendation of examples as part of the intelligent help system in ARIALE. This integration tries to solve the problems of cognitive load, production paradox, and the teachers′ time availability. Figure 5.4. shows two different scenarios: a teacher without assistance and a teacher supported by the help system.

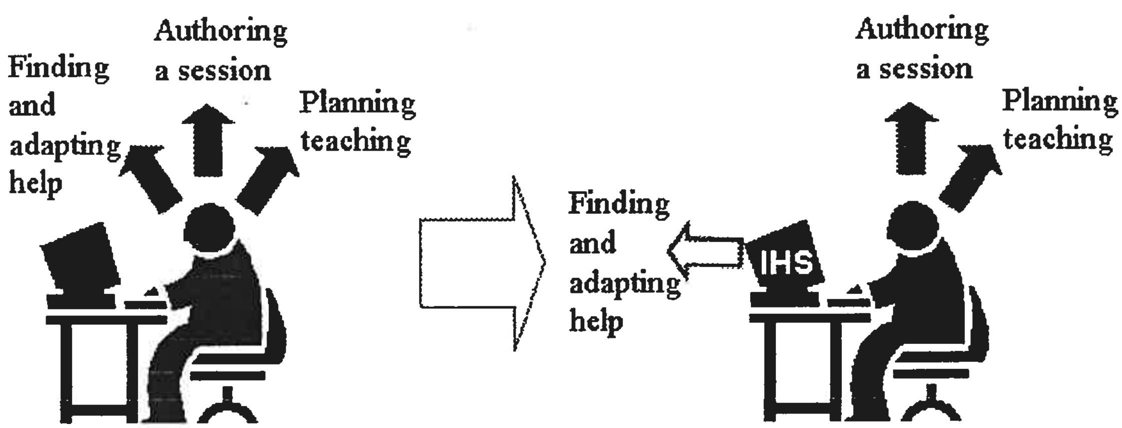


Figure 5.4. The Intelligent Help System (IHS) manipulates the help in order to free the teacher's mental capacity.

5.3.2. Accessing help

To help teachers to find and access specific and personalized information, we designed a process based on the knowledge existing in the Teacher Model. The process to find and personalize specific information starts when the teacher gives explicit information to the system during the registration. With this information, the system initializes the particular Teacher Model for this teacher, as explained in chapter four. After the *creation of the Teacher Model*, every time the teacher authors a learning session, a cycle of interaction (see Figure 5.5.) and changes is repeated to provide adaptive help. In this cycle, the system takes information from the Teacher Model to define the teacher's *skills*, *plans* and *preferences* to create Web pages with resources adapted to the teacher's characteristics. For example, *setting the teacher's skills* means that the system stores information about the current teacher's skills in the working-memory of the system, to support its decisions. *Defining a plan* is coding the short-term activity, tasks and steps that the teacher has to perform in the format for representing a plan, which was explained in section 4.2.3. The teacher's preferences, such as the teacher's language, are also included in the working-memory of the system to support its decisions about contents and specific resources to be retrieved.

In ARIALE, a teacher authors a learning session by changing values in a given a step. These changes are the teacher's feedback that the system receives. Once the teacher has finished the edition of the step, a new Web page is loaded, the Teacher Model is updated with new information acquired during the authoring process and the interaction in a new Web page starts again, taking into account the last changes in the Teacher Model. In this interactive process, the teacher interacts with the authoring tool and with the help system and both parts generate feedback to support the decision of each other (Rodríguez *et al.*, 2004a). Figure 5.5. shows the process to select the appropriate help for a teacher.

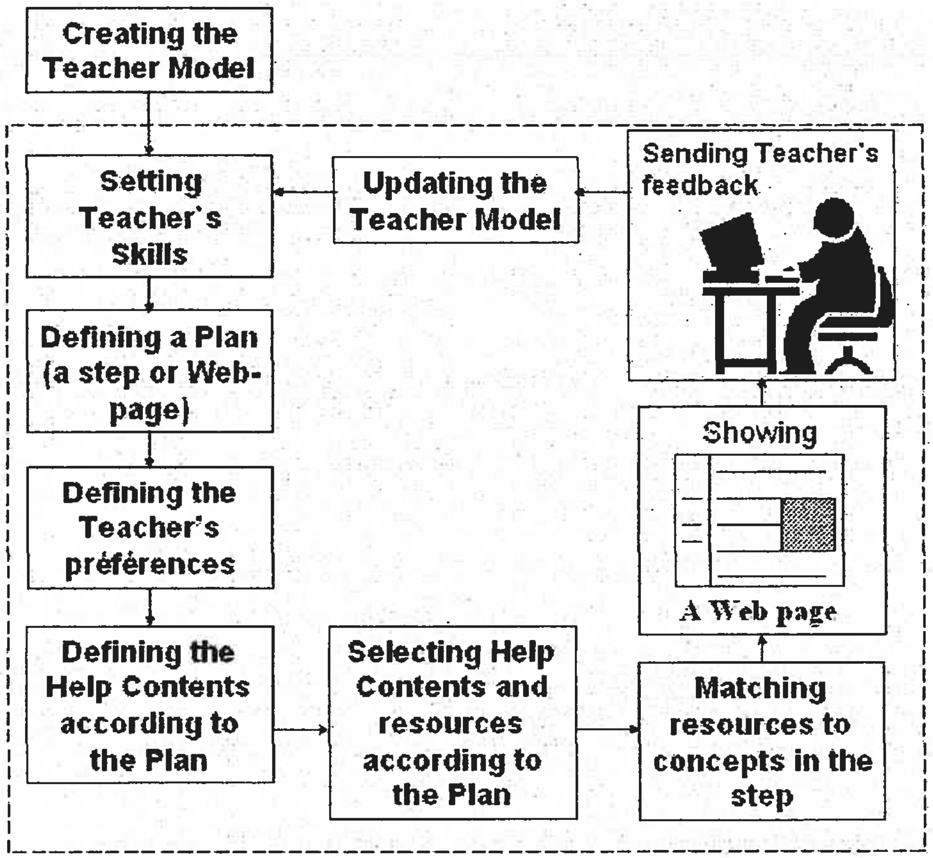


Figure 5.5. Stages for the selection of contextual help for a Web page.

5.3.3. Adaptation

We assume that when an author is creating a learning session with the authoring tool, he faces two main barriers: he requires *specific information* related to his current task, and he finds it *difficult* to locate this information (Capobianco, 2002). To provide specific information in an easy way, we applied the "just-enough" and "just-in-time" techniques (Leung, 2001), as we explain now.

Sometimes help can be available, but it is not really accessible (Leung, 2001; Brusilovsky and Schwartz, 1997) because users do not know how to access it or access is difficult, or the help is available and accessible, but not adapted to the users' needs. More specifically, two of the problems that teachers may face while authoring a learning session can be 1. *Personalizing help according to each teacher's attributes*, and 2. *Finding specific help to perform a task.*

Höök (2000) remarks that few users spend time configuring their systems according to their preferences and needs; thus, it is unrealistic to expect that one would spend time doing this. If a teacher wants to adapt help according to his current plan, skills and preferences, this operation takes into account multiple criteria such as the context, media and bandwidth available. For example, a teacher would configure the system to receive only static images because the bandwidth is too small. In our system, if a teacher had to adapt manually the help that he requires, he would have to do the following tasks:

- Finding the appropriate help according to:
    - The context, which can be the current field, step, task or activity. For example, writing instructions in a text box can involve specific or general information about this action.
    - The complexity; for example, a basic or intermediate level according to the skills.
- Choosing the help found according to his preferences concerning
    - The area of interest; for example, information about what an instruction is, how to write an instruction, or why to write an instruction.
    - The language; for example, the teacher would have to set up the help display in his preferred language if the machine he is using has an operative system or browser in a different language.
    - The media to display; for example, text or video.
    - The bandwidth; for example, video could be incompatible with a slow modem connection.
    - The help strategies and techniques; for example, the selection of a demonstration instead of an introduction
    - The display mechanism; for example, showing texts and images in a text box instead of using an applet. Applets are more commonly used in problem-solving support.

Our system is an assistant that simplifies the situation for the teacher. A goal of the intelligent help system is to adapt the interface contents and their presentation to the user according to his plan, skills and preferences. According to the guide to adaptive interface design proposed by Rothrock *et al.* (2002) (see more details about this guide in section 2.2.5.), there are three rules for designing an adaptive interface:

- Identify variables that call for adaptation; for example, a click event on a link.
- Determine modifications to the interface; for example, changes in the text-based help to be displayed.
- Select the inference mechanism; for example, filter the resources to be displayed or recommend a network topology according to the teacher characteristics.

This guide is part of a framework to study adaptive interfaces or intelligent user interfaces, and it considers the most common elements in an adaptive interface to be:

- *Inputs*: keystrokes and mouse events performed by the user.
- *User variables*: information in the Teacher Model that triggers adaptation.
- *Identification inference mechanism*: a method to identify instances that call for adaptation and to decide on the appropriate modifications for the interface.
- *Interaction model*: a mode of interaction between the user and the interface.
- *Decision inference mechanism*: a method to decide which changes to make to the interface.
- *System or environment variables:* Characteristics of the system, which do not depend on the user's actions and trigger the adaptation of the interface.
- *Selection mechanism:* to control different aspects of the adaptation, such as the complexity of the help to be provided or who decides about the adaptation (teacher or system). The teacher must be able to alter the status of the parameters of adaptation.

- *Evaluation mechanism:* to assess the adaptation accuracy according to the user's preferences, and to check if the adaptation is overfitting or underfitting. A comparison of the adaptation proposed by the system and the teacher's choices allows the system to adjust the adaptation when needed.

In this research, the inputs, the user variables, the decision inference mechanism and the interaction model provide most of the information and functions that ARIALE requires to make decisions. The other elements play complementary roles. Rothrock *et al.* (2002)

define nine user variables that call for adaptation (User Performance, User Goals, User Workload, User Situation Awareness, User Knowledge, Groups of Users, User Personality and Cognitive Style, Situation Variables, Task Variables), and we pay more attention to the following five that provide the most important information needed for adaptation:

User Performance (mouse events, for example).

User Goals (plan to be performed).

User Knowledge (previous experience teaching a subject, for example).

Groups of Users (recommending to the user something that is common to the user's group, for example).

User Personality and Cognitive Style (such as preferences and examples used for teaching).

ARIALE adapts the content to be displayed, the mechanism of teacher-system interaction, and the degree of adaptation using these variables mainly. This adaptation process enables the system to select the appropriate quality and quantity of information, and to decide on a particular method to display help.

5.3.4. Changing the interface

After defining the variables that call for adaptation, the next step is to determine the modifications to be made in the interface. The modifications can be both in the presentation and in the navigation.

In the next sections, we analyze the changes of the interface presentation and, after that, we discuss the aspects related to the interaction and navigation among the nodes of help information. Sometimes, changes in the presentation are the result of navigation.

The adaptation of the interface presentation is achieved by changing:

- The help content.
- The medium to display help messages.
- The language.
- The help techniques.
- The display mechanism.

The changes can be applied to all these criteria or only to some of them. For example, the whole presentation of the content of a Web page can change just by changing the language. Additional examples of the implementation of the interface are included in chapter six. As explained in section 5.2, the generation, testing and recommendation of topologies are based on the teacher's information stored in the Teacher Model. The Teacher Model also supports a decision inference mechanism that uses a three-dimensional representation of the complexity of the support required by teachers. This representation and its application are described as follows.

The Dimensions to Select Help

Changing the interface of the authoring tool and the embedded context-sensitive help requires to choosing different resources. ARIALE uses a three-dimensional structure to support resource selection according to the teacher's characteristics and preferences. This structure organizes data related to (1) the teacher's *skills* to perform a certain task; (2) the *plan* or tasks to be performed (*analysis, transformation of cases, competition*); and (3) the type of *help* required to support the teachers. These criteria are stored in a Teacher Model and used for planning what support is provided to the implementer teacher. For example, for each activity and task, we classify teachers in three basic categories according to their ability and experience in teaching and using the system: *beginner*, *intermediate*, and *advanced* users. The difficulty level of a task is initially static, but its difficulty level changes as the teacher repeats a particular step and his ability varies. The system checks the Teacher Model to verify whether the teacher has accessed a particular kind of template. Records of the number of times the user accessed the page and the associated help are stored in the database. In addition, if the user has accessed demonstrations in a particular step, during a later access to this step the system will provide guidance because the degree of complexity of the step will be relatively lower. Furthermore, the interface uses a lower number of multimedia file demonstrations depending on the table of preferences for that user. According to the stored information, a teacher with experience creating learning sessions in the analysis phase receives less basic guidance, but he receives more intensive help to create a competition for the first time. ARIALE takes into account data about the teacher's skills, checks the plan to follow, and decides the type of learning support or problem-solving support that the teacher requires (strategy and techniques). This decision depends on the events triggered in each Web page. For example, when the teacher is a beginner with no sessions in progress and a Web page is loaded, our system offers problem-solving support by providing a worked-out example to reuse. In all cases, learning support is displayed when the teacher moves the mouse over a link. Figure 5.6. shows the three-dimensional organization of the key factors used to support teachers.

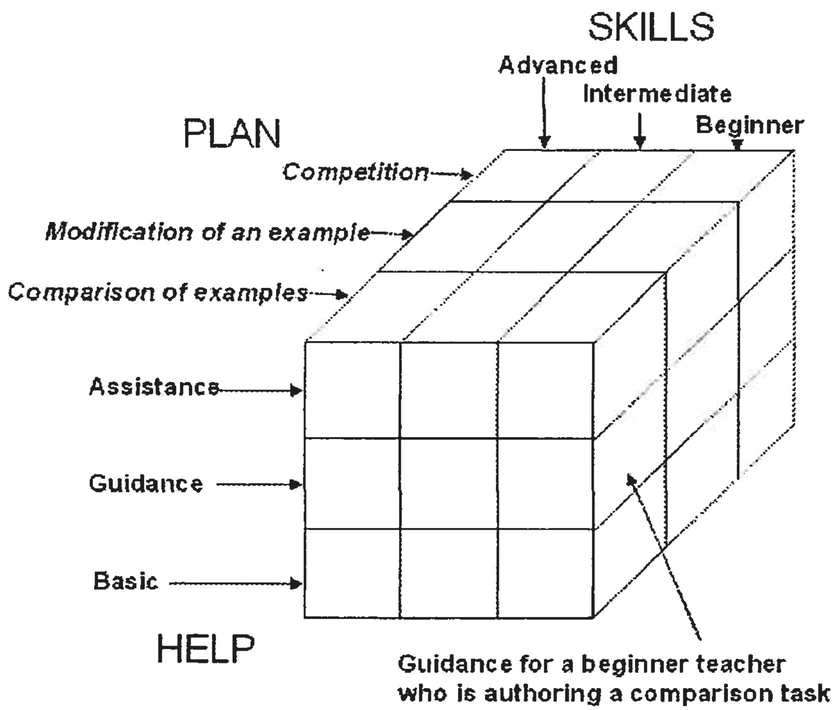


Figure 5.6. A 3-D model of tasks and guidance (Rodríguez *et al.*, 2003).

This 3-D model (see Figure 5.6) is useful to reduce the search space of support. This model is the start point to find the appropriate support for a teacher who has particular skills and is following a specific plan. ARIALE uses the teacher's skills and his plan to find the right help (a help strategy and the respective help techniques) and adapt the help according to the teacher's skills, plan and other characteristics. Problem-solving and the automatic generation, testing and recommendation of examples were discussed in section 5.2. Now, first, we focus on finding the right support and on adapting contents and resources to display the help. Second, we pay attention to the rules for navigation through the help contents.

Finding help

Finding the right help and adapting it can be complicated and time-consuming jobs for a teacher. That is the reason because ARIALE has the function of providing the information the teacher needs. Unfortunately, sometimes a teacher is not sure about what information he needs, and then the system has to infer and predict what information the teacher might need. If the system knows what information is required, the system can supply just-enough pieces of information just-in-time. Consequently, the system needs information about the teacher's context (user data, usage data and environment data) to support its decisions about what information to find for the teacher. Based on this data, the system can always offer information that the teacher might need, except in the case of a teacher who has accessed all the support available. As Leung (2001) proposes for learning support, the system can offer help and the teacher can use it or just ignoring it. In the case of problem-solving support, for instance, if the teacher rejects an example recommended by ARIALE, he can create a different example and the system takes into account this decision to modify the teacher's teaching style. ARIALE is efficient to provide learning support because help contents are indexed to each field of each Web-page, following the same idea as in ADAPTS (Brusilovsky and Cooper, 1999). But, different to ADAPTS, our system adapts help contents not only to the plan but also to the teacher's skills and personal attributes. Then resources to be displayed are not always the same for a certain field. The method to choose appropriate resources is explained now.

Retrieving and assigning resources

Once the teacher enters a Web page, the teacher's skills and plan representation (code) are in memory. The first five characters of this code show the context (course, activity and task) of the current step. The teacher's skill determines the type of help strategy and technique to use. Once the plan, the teacher's skills, and the help technique are defined, the next phase is filtering help resources to retrieve the appropriate resource for each help technique (Rodríguez *et al.*, 2004b). These resources can be files with texts, graphics, videos, applications or simulations. Each resource file is indexed by a set of pairs *attribute/value*; these attributes are *complexity*, *media*, *language*, *load*, *help techniques*, *context area*, and *display mechanism*, and some of their possible values are detailed in Table 5.12.

Table 5.12. Attributes used to index resources

Attributes	Values		
Help Stategies	Basic help	Guidance	Assistance
Language	English	Spanish	French
Load	Low bandwidth	High bandwidth	
Area content	Usage	Domain	
Media	Text	Image	Multimedia
Complexity	Beginner	Intermediate	Advanced
Display mechanism	Form	Layer	Applet

Based on the three-dimensional model of the teacher's characteristics, the system defines the resources that are more suitable according to the teacher's skills, plan, help strategy, and preferences. This definition consists of retrieving the values of each teacher's characteristics that are stored in the Teacher Model, for example, his skills, plan, language, and preferred media to see messages. After retrieving a set of values from the Teacher Model, the system chooses the resources whose attributes match these criteria. By matching these values to the attributes of the resources, the system defines the appropriate resources for the step that the teacher is going to perform. For example, a resource such as an image can be selected because its attributes match the values of the teacher's characteristics (Rodríguez *et al.*, 2003). Using this same method (filter), the system also predicts the resources that might be needed later in a Web page. With the resources identified, the system can retrieve the corresponding resources and thus build a Web page linking resources to fields in the template. Some predicted resources are not used immediately and the Web page loads them for future use with help techniques.

To link a resource to a specific field in a Web page, the resources (texts, images, multimedia files) are indexed by hand in a many-to-many way to fields in different Web pages. This process is tedious and time consuming but highly precise. The reasons to use this method are that other methods make more mistakes choosing resources (Leung, 2001). In addition, resources, especially audiovisual ones such images, sounds, video must be indexed to identify them according to the pairs attribute/value. A table in a database stores these relationships to enable the system to assign resources to fields in templates while a Web page is being created. The filtering process of resources is depicted in Figure 5.7.

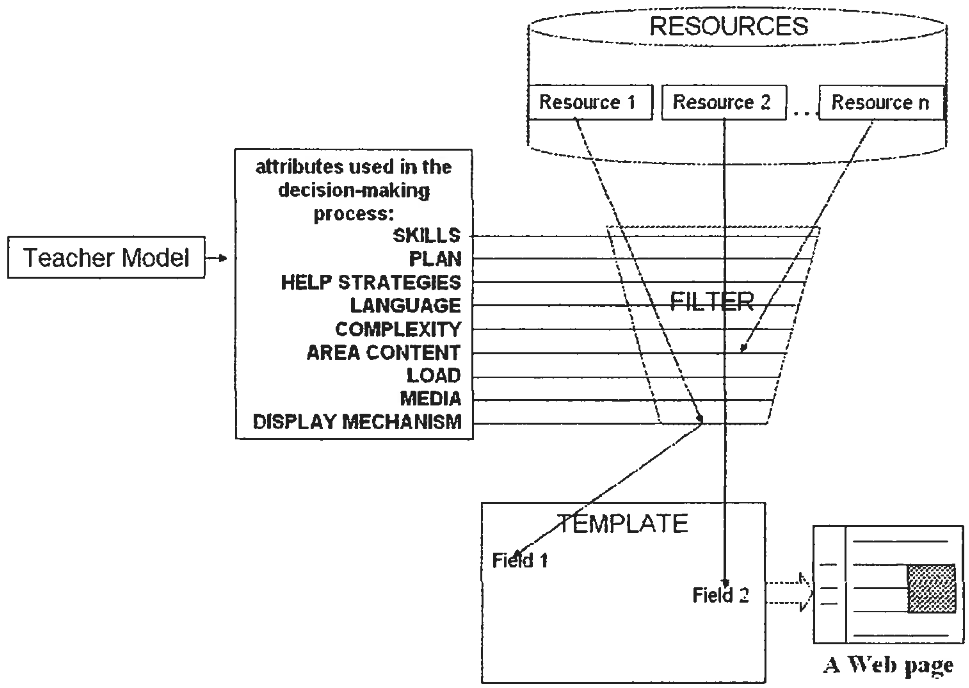


Figure 5.7. The system uses criteria to make decisions about which resources are located in a Web page (Rodríguez *et al.*, 2004b).

The advantage of this method is the flexibility to reuse resources and to build Web-pages dynamically according to multiple attributes that acts as a filter. Most systems do not take into account additional information about personal characteristics to adapt help content and select the corresponding resources. Usually, Web-based help systems are not knowledge-based and they only provide context-sensitive help regarding the task that a user is performing at that moment. Choosing resources based on information stored in the Teacher Model and predicting the help contents and resources that a teacher might need later are a new vision of adaptation in help systems, particularly Web-based help systems.

5.3.5. Embedded context-sensitive help

In ARIALE, problem-solving support and learning support are context-sensitive help (see section 2.3.3). This type of help is not only related to the current plan that the teacher is following but also to each topic displayed in the respective template. Context-sensitive help provides information that relates to the specific field of the step in which the teacher is currently involved. Furthermore, the teacher receives context-sensitive help about the Web page he is working on (Self, 1998), and he sees only the information about that part of the page. Because context-sensitive help provides just-enough help according to the current teacher's task, we consider this to be a useful mechanism to decrease the teachers' cognitive overload.

Another important aspect of this system is that the help is embedded. There is an area in the browser window that displays help. Thus, the help appears within the same window, rather than in a separate help window. Furthermore, when the teacher needs help, he does not have to leave the browser window where he is working. In the case of problem-solving support, when an example is recommended and accepted, it is loaded as part of the Web-page. An innovation of ARIALE is the integration of embedded context-sensitive help, adaptation and prediction of support in a help system. As a positive consequence, the teacher does not switch between two different Web-pages to check or edit the example or adapt help contents and resources according to his needs and preferences. The navigation between help nodes does not need long trips through different windows. Instead, help contents are displayed according to the teacher's interests in the same Web-page where the teacher is working. Details related to the implementation of context-sensitive help in our authoring tool are described in chapter six.

Rules for navigation

Traveling though the help contents is like following a sequence of states. In the case of adaptive context-sensitive help, typically assigned to the "MORE" link, each time the teacher clicks this link the help displayed is updated. As the teacher accesses more help messages, he passes from Basic to Guidance. When the teacher accesses the Assistance level he continues using Quick Reference. The system keeps track of the teacher's accesses to each one of the three "MORE" links in each step. As a result, the teacher can receive help of different complexity for each link. There are some rules that the system applies to promote a teacher from one strategy to another. Sometimes the teacher has not received help of the same complexity for the three links, but, in global terms, he has received enough information. Then, he can skip to the next strategy. The rules to skip from one level to another are described below.

Contents of different help strategies and techniques conforms a hypermedia and each field in the authoring tool matches a node in the help hypermedia. The organization of the help hypermedia and the correspondence of each node to the authoring tool are described in chapter six. Now, we defined three rules that the teacher has to follow to navigate the help hyperspace. The rules are heuristics that ARIALE includes to decide when the teacher passes from one strategy or level to another. These rules take into account the help received by the teacher and his experience authoring steps. The number of times that the teacher must access help messages and the number of times that he must perform a step before passing to the next help level, are defined arbitrary. For example, we assume that a beginner must get some experience and know all the messages of a strategy to pass to the next help level. The set of rules that ARIALE uses are:

Rule 1.

IF the teacher has done the current step fewer than three times.

THEN he requires accessing all the help messages of the current help strategy in order to pass to the next strategy.

Rule 2.

IF the teacher has done the current step between three and five times

AND he has accessed at least five help messages of the current help strategy for each field in the current step

THEN he can pass to the next strategy.

Rule 3.

IF the teacher has done the current step more than five times

THEN he receives help by means of the Quick Reference technique.

For example, a teacher can consult all the help available about a field using the Basic help strategy and he can ask for more information. Then the system decides to retrieve more help for all the topics in the Web page because the experience accumulated authoring and the help accessed by the teacher are both enough. This rule is based on the everyday decisions that people make. For example, if somebody repeats a question about a topic, the person answering the questions will infer that more information about the same topic could be required, and he will be prepared to answer potential future questions. As ARIALE offers help and leaves the teacher free to access or not it, each teacher's decision to ignore or access help is a teacher's feedback that our system takes into account.

5.4. Learning from interaction

This section summarizes how the system learns to help teachers in a personalized or customized way.

ARIALE has a knowledge base with data that supports its decisions about providing learning support and problem-solving support. The knowledge in this base must improve to offer better support. The main areas of knowledge that our system improves with new data are:

- The similarity of a particular teacher with other teachers, as explained in section 4.2.
- The changes in the teacher's teaching style (see section 4.2.3.).
- The access of the teacher to learning support in each step of each task.
- The changes of each teacher's preferences about learning support.
- The teacher's experience in authoring learning sessions.

ARIALE uses some methods to improve its knowledge, such as classification learning and an overlay model in the Teacher Model. Related to problem-solving support, the main method that ARIALE uses to learn is *Bayesian classification* (Keller, 2002a) (see also sections 2.2.7 and 5.2). By classifying the examples that each teacher uses, the system learns which type of example the teacher prefers. Our system learns when its knowledge base changes as a result of Bayesian classification. For example, a new topology is classified and new data is stored, and then the system updates not only information in the respective Teacher Model but also some aspects related to the teachers´ categories. ARIALE also creates a history of the teacher's access to help based on the interaction between the teacher and the system. The system learns by monitoring the teacher's accesses to learning support, the changes in his preferences, and the tasks he has performed. After the classification of each new example and after every time a Web page is loaded, ARIALE updates the data which is necessary for making decisions about learning support and problem-solving support. For example, every time a teacher receives a Hint, uses an Instance, or accesses an Introduction, respective counters are increased on the client-side. These counters allow the system to update the records of the teacher in his Teacher Model.

More specifically, since each field in a Web page is identified by an eight-character code, the system can monitor the teacher's access to help and increases the respective counters. These counters allow ARIALE to decide by itself if, for example, the complexity of the help must change, the Web page has to be reloaded to update new help content, or if it is necessary to update a database.

Relying on the information collected from the counters on the client-side, the system will not only be able to adapt the authoring tool and the help contents in the future, but also learn from the teacher's actions. For example, if the teacher has accessed more help about only one particular task, he is considered an advanced user of this task, but a beginner in other tasks. This information about accessing help is in the Teacher Model and the system makes decisions based on it (Rodríguez *et al.*, 2004a). These decisions are not always the same, and they change as the teacher progresses, receives more help, and improves his experience working with the authoring tool. The system knowledge about the help that the teacher has received is detailed because the system keeps tracks of the teacher's status in each page and task. For example, if a teacher starts a new session, the system remembers the level of skill that he has achieved, thus making it possible to provide appropriate help.

Many functions performed by ARIALE would not be possible if our system did not remember the teacher's knowledge or learn changes that have occurred in its knowledge base. For example, if a teacher has progressed and accumulated more experience authoring learning sessions, our system has to keep track of this to avoid repeating support that has already been provided. In addition, if a teacher frequently uses a particular type of example in his teaching, the system should remember this tendency (teaching style) to offer examples of the same type. Learning also allows our system to deal with unforeseen situations, such as when a teacher changes his teaching style.

5.5. Conclusion

This chapter explained the methods that our system ARIALE uses to support teachers. ARIALE is an assistant with the structure and functionality of an Intelligent Help System (IHS) that provides learning support and problem-solving support. The first section of this chapter explained the most important contribution of our research, which is a problem-solving support that includes automatic generation, test and recommendation of topologies.

The second section described the design of the learning support that complements the problem-solving. Finally, we focused on the methods that ARIALE applies to learn how to support teachers.

To recommend topologies, our system accesses a case base that stores topologies and selects a series of examples matching the teacher's teaching style (teaching style is the class of topologies that a teacher prefers). If the teacher does not accept to reuse any example that our system recommends, ARIALE generates additional examples by using a probabilistic method, which assigns at random the links between the nodes of a topology. We use Bayesian learning, an artificial intelligence technique (see also section 2.2.7.), to test if a generated example matches the teacher's teaching style; if the example matches his style, ARIALE recommends the topology. Bayesian learning is a kind of classification that uses probabilities. Using a Bayesian classifier makes the generalization, scalability, and reusability in ARIALE easier. An important remark is that ARIALE creates and not only reuses topologies that match teachers' characteristics. Automatically generated teaching materials are not usually adapted to each teacher's characteristics in other AEH and ITS. Once a teacher accepts a new topology, this example is included in the ARIALE's knowledge database and the system has learned new knowledge to support its future performance in an improved manner.

The learning support in ARIALE gives a frame of adaptive and context-sensitive help for the interaction between ARIALE and the teachers. Learning support works parallel to the authoring tool, and finds, selects and adapts *context-sensitive* support for Web-based help, according to a teacher's skills and plans. Then, our system selects the help strategy and techniques to deliver support, the kind of help content to show, and the media to display the content adapted to the teacher's preferences. This help system focuses on the manipulation of the help to free the teacher's mental capacity and allow him to pay attention to the translation of his plans in the authoring tool. Our system applies knowledge base adaptation to profit the evolving data collected from the interaction between teachers and ARIALE. As a particular teacher interacts with the system authoring, accessing help and getting experience, ARIALE stores data related to this interaction. This data allow the system to adapt help according to the teacher's evolution.

Chapter 6. Architecture and implementation of ARIALE

Summary: This chapter describes the implementation of the problem-solving support and learning support explained in chapter five. Here, we include the architecture of ARIALE, its general layout, general functionality and main processes. This chapter also details the method used for building the intelligent help system and we detail the organization of the authoring tool.

6.1. Architecture

ARIALE is a prototype of a Web-based adaptive hypermedia assistant made up of an authoring tool and an Intelligent Help System (IHS). The authoring tool and the intelligent help system depend on three components:

- Information that supports the decisions of the system (knowledge).
- Resources used to build Web pages.
- Processes that use information and resources to build Web pages.

The data that the system needs for decision-making is related to the definition of the system and includes the curriculum, order of concepts to be presented, teacher's data, environment data, pedagogical strategies, structure of sessions and examples, and rules that the system uses to make decisions. This set of components forms the knowledge of the system. The resources are the texts, images, and multimedia files used to generate menus, forms for the authoring tool, and help messages in the Web pages. The system is a "decision-maker" that uses knowledge to guide the construction of Web pages by selecting low-level resources such as a string of text or an image file. Figure 6.1. shows a scheme of our system and Figure 6.3. details its architecture.

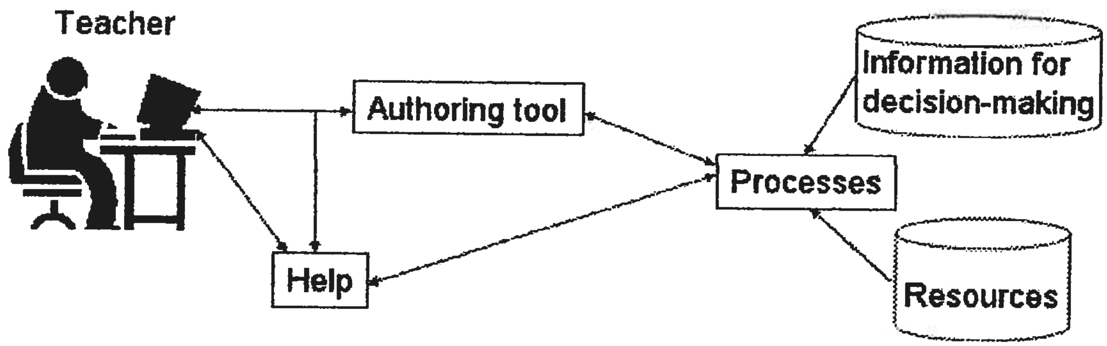


Figure 6.1. Our system depends on processes that use information about the teacher, the environment, and resources.

ARIALE has a client-server architecture (Figure 6.2.) with a three-tier structure that includes most processes working on the server-side, the database on the server-side, and a client-side.

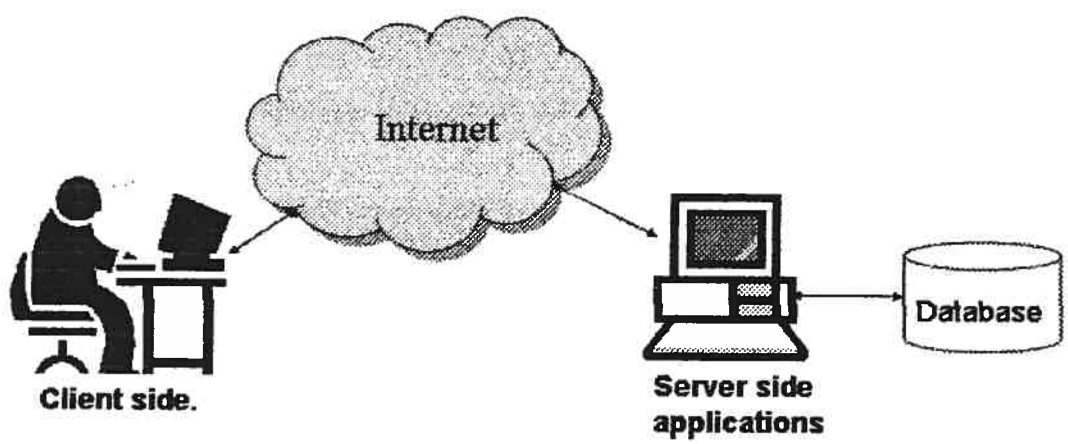


Figure 6.2. A general solution includes a three-tier client-server configuration.

On the server side, the system includes components such as Curriculum, Teacher Model, Pedagogical Model, Sessions Base, Resources, a Session Manager, a Planner and a Helper. On the client side, an interface agent interacts with the user, thereby monitoring his environment and sending information to the server (Vázquez-Abad *et al.*, 2003). Figure 6.3. shows the general architecture of the systems, whose components are explained in the next section.

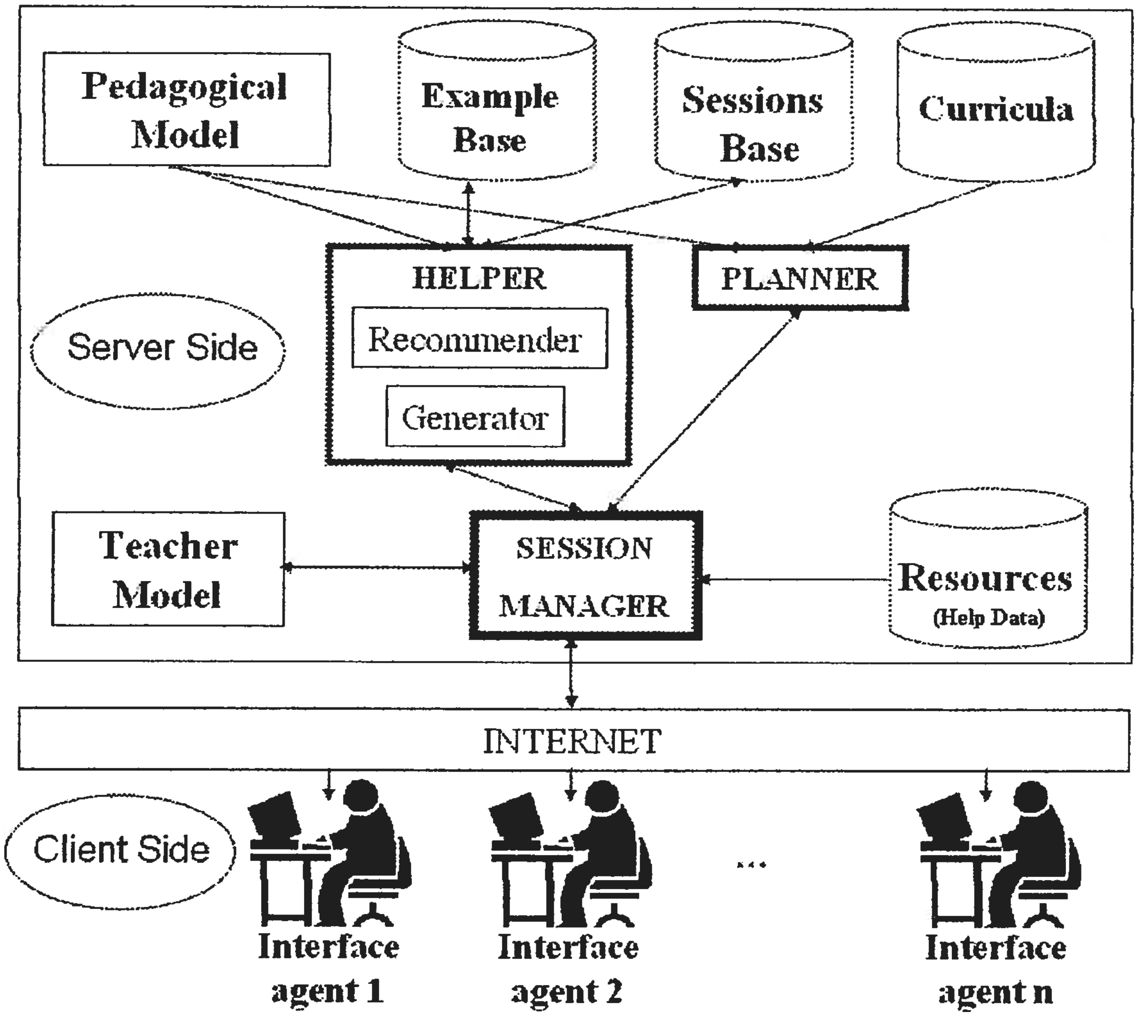


Figure 6.3. General architecture of the system.

6.1.1. Models and Resources

Information in the Pedagogical Model, the Teacher Model, the Curriculum, the Session-base, and the Example-base, make up a *knowledge base* (Fischer, 2001) of our system. This knowledge base supports the different decisions that ARIALE makes. The Teacher Model and the Pedagogical Model were defined in chapter four, and here we explain the implementation of the Curriculum and the Session-base as parts of the knowledge base of the system. In addition, we describe the resources used in the Web page and define some of the processes used to provide help, create, and modify sessions.

Curriculum

As explained in chapter four, the Curriculum is a sequence of *activities* and *tasks* related to a *subject*. According to this Pedagogical Model, each activity of a curriculum (except the last one) includes the following pedagogical strategies: *Analysis* of cases, *Modification* of cases, and a *Competition* among students to provide the best solution to a problem. Each pedagogical strategy is mapped to a particular task and each activity has three tasks, one for each pedagogical strategy. The last activity is a single task that consists of a *virtual competition* where an underlying simulation of IF-THEN cases is presented to the students to prompt them to integrate all the concepts they have studied in the previous activities and tasks. Each task also has some steps and each step is a template of a particular Web page. (See Figure 4.2. and Table 4.1. in chapter four, which illustrate this structure).

Each Curriculum is created by a designer, and different designers can create Curricula with a different number of activities (units or chapters). The designer of a curriculum has to create the structure (units or activities) and contents to be taught. To perform this task, it is possible to use a high-level authoring tool similar to Tangow (Carro, 2001) or Aha! (De Bra *et al.*, 2003). However, the system can also be programmed directly, as we do. In our system, a Curriculum has two faces: the part directly related to teaching, and the part related to the help. The creation of a tool allowing an implementer teacher to adapt or personalize an ITS requires a lot of effort from both the developer and the designer. A curriculum includes the teaching content and its organization. The structure of the curriculum includes mapping contents, concepts, and the corresponding activities and tasks to specific steps.

Any Curriculum can be implemented by many *implementer teachers* in different ways by using an intermediate authoring tool to author different *courses*. Each course is an instance of a Curriculum, and each course has at least one *learning session*. A session includes the structure of activities, tasks and steps of the corresponding Curriculum. In such a way, a session becomes a plan that is composed of all the partial plans (steps or Web pages) of the current session. We use "step" as a synonym for a partial plan or Web page, and "plan" as a synonym for session.

The help contents are another Curriculum that is parallel to the Curriculum to be taught, and the designer must provide resources for the two *parallel hypermedia*. One or more hypermedia working simultaneously are *parallel hypermedia*. For example, the designer must supply the texts for each instruction displayed in the authoring tool area of the screen, as well as the texts for each help message related to this instruction. Two hypermedia are parallel when there is a correspondence between the nodes of one hypermedia and the nodes of another hypermedia. For example, in ARIALE the contents to be taught and their relationships conform a hypermedia that is parallel to the hypermedia composed by the help contents and their links. The Curriculum and the Help are connected parallel hypermedia, and when a designer creates a curriculum, this curriculum is the basis for different hypermedia. For example, each learning session adapted to an implementer teacher is composed of two parallel hypermedia: the authoring environment and the corresponding help. The help nodes that adapt to the progress of the teacher (for example, Introduction when clicking the link "MORE") are Adaptive Context-Sensitive Help (AC-SH). The help that does not adapt continuously as the teacher progresses (help for menus, for example) is Non Adaptive Context-Sensitive Help (C-SH). Figure 6.4. shows the structure of a curriculum hypermedia and the help hypermedia.

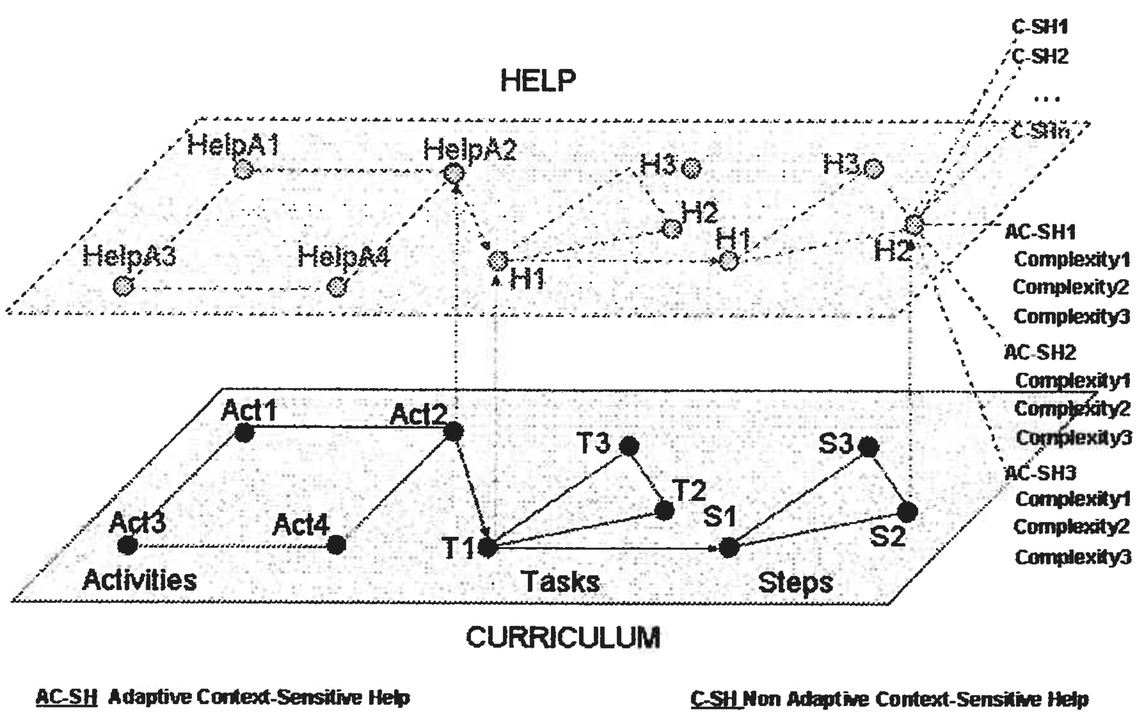


Figure 6.4. Curriculum and Help are parallel hypermedia that interact.

Accessing help is a type of navigation through the help hypermedia. When a teacher clicks links to view help or when the system provides active help, this interaction allows the teacher to traverse the nodes of the help hypermedia. The authoring tool and the help system act as parallel layers. As a result, the system supports the teacher's navigation in a hypermedia that corresponds to the current step that he is working on. This interaction with the help system avoids the feeling of "lost in the search space" of help that some people could experience when they are looking for help on a long Web page or through several help windows. A teacher can navigate between the nodes of the main curriculum, which includes the level of the general public and the level of the authoring tool. In the case of the level of general public, the nodes are the different Web pages that display general information (Registration, Login, Demonstration, Example, etc.). The level of the authoring tool is a set of nodes (steps) that are the different Web pages belonging to the authoring tool. Parallel to each level, there is a help hypermedia, where each node is linked to one or more nodes of the level of the general public area or the level of the authoring tool. For example, for each activity in a curriculum there are one or more parallel help nodes; for each task in an activity there is a parallel help node for the corresponding task; in addition, for each step in a task there are one or more corresponding parallel help nodes.

Sessions

Each session is an instance of the specific teacher's course and teaching style. The different learning sessions are stored in a *Session-base*. Basically, a learning session is a sequence of different steps belonging to a set of tasks and activities. The system has some standard templates for these steps. When a teacher finishes the actions to be done on a step, a new record with information about the step and its resources is stored in the database, as part of a particular learning session. Each template arranges fields with resources (texts, text boxes, graphics, labels, links, and buttons). If the material is shared, a teacher can reuse the activities of another teacher. In this case the relation between both courses and activities is many-to-many as shown in the entity-relation diagram in Figure 6.5., which includes the relationships between Curriculum, sessions and implementer teachers (Rodríguez *et al.*, 2004b). Most of the entities in Figure 6.5. are explained in section 4.2. (see Figures 4.1., 4.2. and Table 4.1.), the concepts of designer, implementer teacher and student are also defined in section 4.2 and the concept of example is explained in section 5.2.4. The next section details what a resource is.

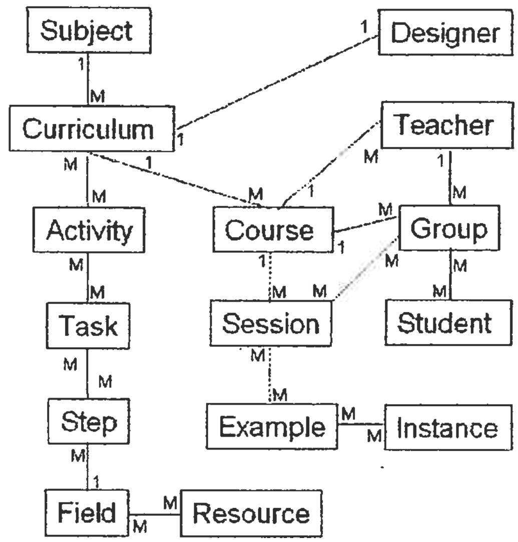


Figure 6.5. Entity-relation diagram for subjects, curricula, sessions and implementer teachers

Resources.

Since each Web page is generated dynamically according to the teacher's skills, plans, and preferences, it is not possible to keep static pages on the server side. We refer to teacher's skills as the ability and experience in teaching and using the system; plan is the activity, tasks and step that the teacher has to perform; and teacher's preferences are particular attributes, such as the teacher's language. An alternative is to have a database of reusable resources: templates of Web pages, graphics, text files, sounds, video clips, specific applications, and simulators. The Resources are stored in a repository and the information required to find, store, and retrieve them is kept in a database. Each resource has a series of attributes or characteristics: name, author, location, description, format, type (text, image, sound, video, application, and simulation), language, if it is shared or not, degree of complexity. These attributes enable the systems to find the right resources for each Web page, as explained in chapter five, section 5.3.4. Figure 6.6. shows how resources can be shared by different templates and sessions.

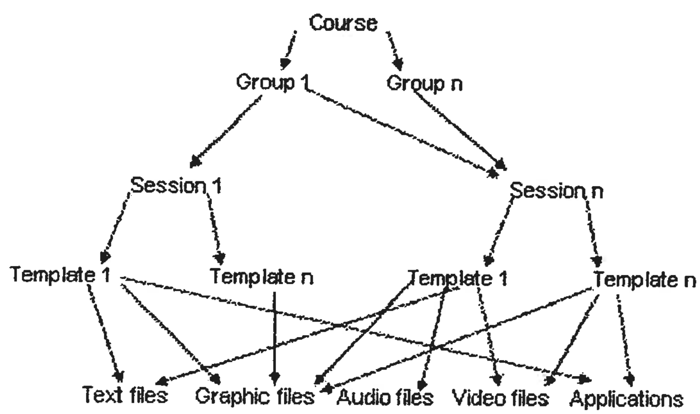


Figure 6.6. Resources contained in files are shared by different Web pages (templates).

6.1.2. Main processes that ARIALE performs

Our system can be accessed by four types of users: *staff*, or people in charge of the administration of the system; *designers*, who are experts who create Curricula and provide resources; *students*; and *implementer teachers*. Because most previous research on ITSs and EAH has paid attention to the functions that support the designers and students' tasks, our research now focuses on the functionality that supports implementer teachers. Figure 6.7. shows a general diagram of our system and its users.

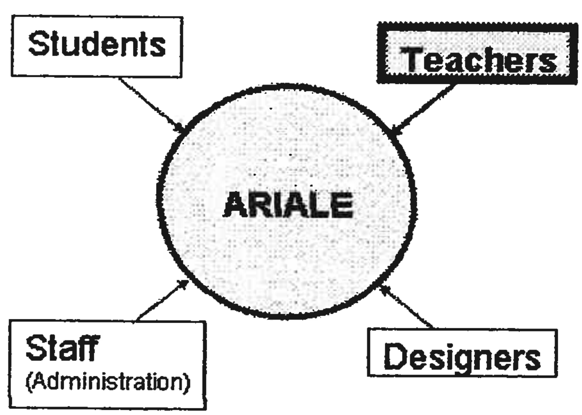


Figure 6.7. The system and its users.

There are two main processes in this system: the processes directly related to teaching and learning activities, and the processes associated with the management of the Web site, its users, resources and databases. Both processes communicate by means of the data stored in the databases (See Figure 6.8). The databases store data of the Teacher Model, the Pedagogical Model, Curricula, Sessions and Examples, and Resources. The contents in the database can be classified in two types: data of resources used in Web pages, and data that the system uses for making decisions about (1) the adaptation, (2) learning and (3) other functions. The former does not change frequently, while the latter is updated continuously. For example, data of resources are the identification of images and text used for building a Web page. An example of data that supports decision-making is the Overlay Model of the teacher, and more specifically the number of sessions and examples that the teacher has done.

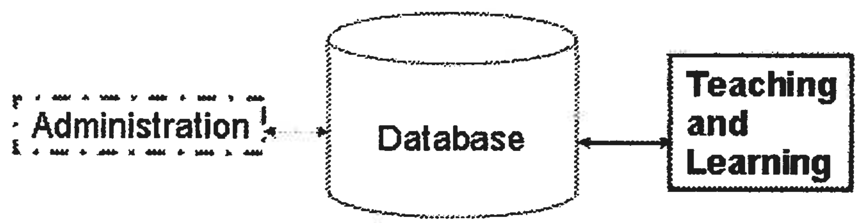


Figure 6.8. Administration processes communicate with Teaching and Learning Processes by sharing databases.

In the Administration area, there are two main types of processes that are related to *Teacher's Profile Management* and *Resources Management*. We focus on processes related to the former, specifically the process linked to Registration and Updating the Teacher Model. Registration includes the initialization of the Teacher Model and Updating Teacher Model is in charge of updating the data related to the teacher's performance.

In the Teaching and Learning area there are three types of processes: a group related to the course design, a second group related to the functionality for students, and the group of functions for authoring learning sessions (See Figure. 6.9). Our research focuses on the functionality that helps implementer teachers to author sessions.

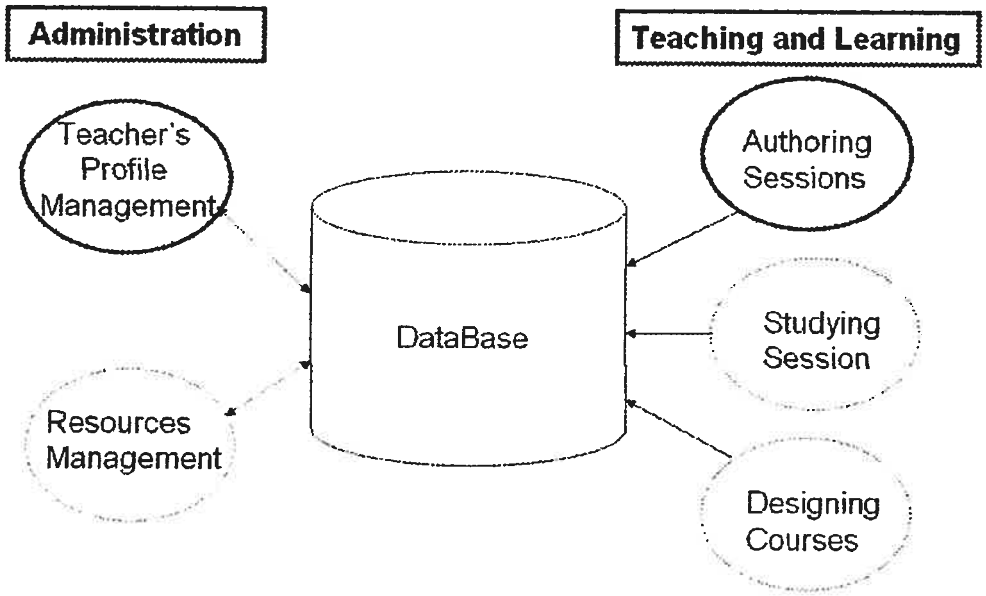


Figure 6.9. This research focuses on the User's Profile Management and processes of Authoring Sessions.

Most processes that support implementer teachers while they are authoring sessions are distributed in the modules Session Manager, Helper, and Planner. Other functions are performed by the Interface Agent on the client side. The functions of these modules and their processes are described as follows.

Session Manager

The most important role of the Session Manager is the coordination and control of the communication between the server side and the client side, and among the other modules on the server side. The Session Manager receives and sends information to and from the Helper, the Planner, and the Interface Agent. Tables 6.1. and 6.2. show some input and output that the Session Manager exchanges with the other components of the system. The Session Manager also inserts and retrieves data from the Teacher Model and the Resources Base.

Table 6.1. Main inputs that the Session Manager receives.

Main Inputs	***Example***
Explicit teacher's data from Registration forms	The subject of the course that the teacher wants to teach
Data from the client side	An instruction that the teacher entered.
Plans to be performed	The teacher's last task that is incomplete
Teacher's skills	The teacher's level: Beginner, Intermediate or Advanced. This information is in the Teacher Model.
Teacher's preferences	The language to be used in the authoring tool.
Resources definition	The identifications of resources that the Helper passes as parameters.
Interface Agent messages	A request for additional text-based help.
IP request	A request that our system receives from any computer connected to Internet.
Helper requests	A request to get the current teacher's skills that the Helper requires for making a decision.
Planner requests	A request to get the current teacher's plan.

Table 6.2. Main outputs that the Session Manager sends to other components of ARIALE.

Main Outputs	***Example***
Teacher Model	The Session Manager builds and maintains the teacher's Profile and Overlay Model.
A Web page	A step of the task 2 for the activity 2.
Request for a plan	Request for a new plan to be followed.
Request for help	Request for text help to be sent to the client side.
Request for a recommendation	Request for a list of sessions that a teacher could use.

Session Manager Processes

The Session manager deals with two types of processes: general aspects of the Web site, and manipulation of the Teacher Model. The most important processes of these two areas are enumerated below:

<u>Access to the Web site</u>	<u>Manipulation of the Teacher Model</u>
Accessing Web site	Initialize Teacher Model
Login	Retrieve Teacher Model
Logout	Update Teacher Model
Retrieve resources for a template	
Build a Web page	
Send a Web page to the client side	
Process interface agent messages	
Coordinate communication between client-side and server side	
Save and retrieve learning sessions	

For example, the Session Manager uses the following algorithm in the process "Accessing Web site":

```
Procedure  Accessing Web site {
        Call the Procedure Setup Screen
        Process an IP request from a teacher
        IF IP address is known THEN
                Display the Login Web-page
                IF the Login data is correct THE
                        Call the  Procedure Build a Web page
                        to generate a personalized teacher's Web-page
                ELSE  Display the Login Web-page
        ELSE   Display  the main  Home Page of the Web-site
}

Procedure Setup Screen {
        Detect client-side characteristics
        Set up client-side characteristics in the Teacher Model
}
Procedure Build a Web page{
        Call the Procedure Update Teacher Model
        Call the Procedure Retrieve Teacher Model
        Call the Procedure Include the Menu according to the teacher's Level and  Language
        Call the Procedure Include a Template for the respective step
        Call the Procedure  Include Help according to the teacher's Plan
        Retrieve resources  according to the teacher's Plan, Level and Language
        Call the Procedure Send a page
}
```

This algorithm for "Accessing Web site" calls other procedures, including some that are part of the Planner and the Helper. For example, the procedure "Build a Web page" calls "Include a Template", a procedure that belongs to the Planner, and "Include Help", a procedure that belongs to the Helper module. The next algorithm describes the functions of the Session Manager as a page builder.

Procedure Build a Web page{
Call the Procedure *Update Teacher Model*
Call the Procedure *Retrieve Teacher Model*
Call the Procedure *Include the Menu* according to the teacher's Level and Language
Call the Procedure *Include a Template* for the respective step
Call the Procedure *Include Help* according to the teacher's Plan
Retrieve resources according to the teacher's Plan, Level and Language
Call the Procedure *Send a page*
}

Procedure Retrieve Teacher Model {
Retrieve the teacher's profile
Retrieve the teacher's overlay model
Call the Procedure *Retrieve a Plan* // this procedure belongs to the Planner

}
Procedure Send a page {
Send a Web page to Client side
Establish communication client-server
Start interface agent applet
Start monitoring the interaction teacher-system
}

To process the messages that come from the interface agent, the Session Manager applies the following algorithm.

Procedure Process Interface Agent PostRequest{
Receive a request from the client side
Get parameters from the client side
Pass parameters to the respective programs or modules
Call the Procedure *Update Teacher Model*
Retrieve resources to send to the client side
Return results to the client side
}

To initialize and update the Teacher Model, our system applies the corresponding algorithms already described in section 4.2.2.

Planner

The Planner has two main responsibilities: the generation of the plan and the selection of resources for the authoring tool.

Main Inputs

Request for a plan; for example, the Session Manager can ask for the next plan to be followed.

Main Output

A next Step Web page (a plan). For example, a particular template and the text that this template requires.

Planner Processes

Generation of plan	Selection of resources
Define a Plan	Define the resources
Verify Sequence	

In order to build the authoring tool section of each Web page, the Planner defines the appropriate resources. For example, if the teacher's language is English, the Planner establishes that the texts have to be in this language. The following algorithm details how a plan is generated.

```
Procedure Defining a Plan {

    IF there is a current session THEN
        Retrieve the identification code of the next step to be performed
        Call the Procedure Verify Sequence
    ELSE{
        IF there is no session available for the current teacher THEN
            Ask HELPER to recommend a session
            Retrieve the identification code of the first step of a recommended session

        IF there is an incomplete session THEN
            Retrieve the identification code of the next step to be performed
            Call the Procedure Verify Sequence

        ELSE ask HELPER to recommend a group of sessions
        IF teacher asks for a new session THEN
```

```
                    Retrieve the identification code of the first step of a new session

                Define the template of the step to be performed
                Call the Procedure Define the resources
                Return template and resource definitions to the Session Manager
            }
}

Procedure Define Resources {
        FOR each field in the template
                Retrieve the field identification
                Retrieve resource identification of the resource assigned to the field
                Return resource identification and field identification
}

Procedure Verify Sequence {
        IF the next step does not follow the current step THEN
                Warn the teacher to verify if he wants to continue skipping or follow the
                sequence
}
```

Helper

The Helper is a "device generator" (Virvou and Du Boulay, 1999) or inference mechanism for adaptation (Rothrock *et al.*, 2002) that identifies instances that call for adaptation and decides on the appropriate modifications for the interface, namely to select the resources to be displayed or recommend a network topology according to the teacher characteristics. The Helper is in charge of providing the adapted learning support and the problem-solving support and, therefore, it includes the processes to generate examples and the recommender of sessions and examples. Tables 6.3. and 6.4. show some input and output that the Helper exchanges with the other components of ARIALE.

Table 6.3. Main inputs that the Helper receives.

Main inputs	*Example*
A plan to be performed by the teacher	A step of the task 1 for the activity 3.
Teacher's attributes	The teacher's skill level (Basic, Intermediate, Advanced)
The teacher's class	The class of example that the teacher prefers, let us say 2.

Table 6.4. Main outputs that the Helper sends to the other components of the system.

Main outputs	*Example*
Help Resource selection	The identification of an image file.
A set of recommended sessions	A set of learning sessions that match the teacher's preferred class.
A set of recommended examples	A set of examples that match the teacher's preferred class.

Helper Processes

To provide learning support, this Helper organizes the sequence, frequency and quantity of information to be sent to the client side, especially the resources needed to provide help such as text files, multimedia files and complementary applications. The most important processes performed by the Helper are:

<u>Help definition</u>	<u>Recommendation</u>
Define help	Generate examples
	Recommend sessions
	Recommend examples

The next algorithm shows a summarized example of how the Helper defines the help to be provided:

```
Procedure Define Help {
        Match relationships between teacher's plan and preferences and resources attributes
        Define the resources to display the  help contents
}
```

To provide problem-solving support, the Helper also defines and retrieves the sessions with the examples that are more suitable for a particular teacher, according to the data in the knowledge base of the system. This process is one of the most important because it supports the functionality of the *Session* and *Example Recommender.* The algorithms *"Procedure Recommend a session"*, *"Procedure Recommend an example"* and *"Generate examples"* in section 5.2.6.3 illustrate the methods that the Helper uses to recommend sessions and examples.

Interface agent

- On the client side there is an interface agent that interacts with the teacher, monitors the context and the teacher's events in order to understand his requests, anticipate his needs, and decide when to act. The interface agent perceives data related to the teacher's mouse events and keystrokes and makes decisions based on this data. The interface agent does not wait until the teacher loads a new Web page to make decisions on behalf of the teacher or after the teacher's confirmation. This agent exchanges information with other components of the system, sends requests for additional help if the teacher passes from one help strategy to the next one, and retrieves additional help from the server side. The Interface Agent *updates* the preferred type of presentation (text, image or multimedia) or other preferences in the server side.

Main inputs

This refers to implicit teacher's data from interaction with the system, for example, the clicks that the teacher does over a link to get more information. The interaction between the teacher and the interface agent occurs when:

- The teacher performs an event, the interface agent initiates a dialogue and the teacher confirms or rejects the offer included in the dialogue. For example, although the teacher had rejected an example previously, the interface agent insists to confirm the teacher's preference.
- The interface agent detects the teacher's events and generates changes according to the teacher's preferences.
- The teacher describes an example and the interface agent coordinates the recommendation of similar instances. This function is explained in the section 5.2. devoted to the problem-solving support.

Main outputs	**Example**
Problem-solving support	Recommended examples.
Request for additional help	Requests for additional images with help content.
Display definitions	Messages displayed when the teacher moves the mouse over a link.

Interface Agent Processes

The interface agent communicates with the components hosted in the server by means of the Session Manager. The interface agent is also in charge of monitoring the teacher's events and retrieving adapted help. Some processes that the interface agent accomplishes are:

- Monitor the teacher's events in order to evaluate if the teacher can pass from one help strategy to the next one. *Monitor* the teacher's events to understand his requests, anticipate his needs, and decide when to act.
- Request for additional help (help messages, examples) if the teacher can pass from one help strategy to the next one. Send a request for additional help if the teacher passes from one help strategy to the next one.
- Retrieve additional help (help messages, examples)
- Send data to the server side
- Display dialogue windows for the user
- Update the Teacher Model in memory. *Update* the preferred type of presentation (text, image or multimedia) or other preferences on the server side, if the teacher changes them or if they have to be changed without reloading the current step.

An example of how the interface agent works is as follows: the interface agent on the client-side retrieves new information on demand. To do this, the click event triggers a JavaScript function that calls an applet function, this applet function calls a servlet in the server side, and the servlet returns the new required information to the applet. The applet integrates the new resources in the Web page, which did not have to be reloaded. The servlet is part of the Session Manager, which asks the Helper to define the resource to retrieve. In addition, the Helper also predicts the resources that the teacher could ask later for the same concept later.

6.2. Methodology used to develop our system

The developing of ARIALE was based on two main aspects:

- Data and implementation are independent (Costagliola *et al.*, 2002); thus, content adaptation must be easier. This means that no template has a static component such as text, or image and each template is filled dynamically.
- The system has a typical three-layer model (Cristea and de Mooij, 2003; De Bra *et al.*, 2003; Koch, 1999; Costagliola *et al.*, 2002):
  - a presentation level, to display information;
  - a storage level, with the physical representation of data (files and database entries),
  - a logical level, with the correspondences between data stored in databases and the structure of the templates; these correspondences are according to the teacher's skills, plans and preferences.

Taking into account those two main aspects, we had to create the following method to build our hypermedia system because we did not find a complete method to develop intelligent help systems for authoring tools. Our method integrates fragments from other methods and guides for building ITSs (Kimiavi, 1998), EAH (Wu *et al.*, 1998), help systems (Selbach *et al.*, 2003), and adaptive interfaces (Rothrock *et al.*, 2002). Our method has two parts: the first is intended to create the general functionality of a Web site and authoring tool; the second part includes the intelligent help.

6.2.1. Part 1. Implementing the Web site and the authoring tool.

Requirement analysis.

In this phase, the needs of the course designer[1] were partially defined in order to be mapped to initial Web pages. The expert content required was also defined, the general tasks that the implementer teacher (author) had to perform were identified, and the most important areas of problem-solving support and learning support were also detected. For example, we had to identify the type of course that the designer wanted, the output he wanted the system to generate and the method that the implementer teacher and the system must follow to produce a learning session for a student. In addition, it was necessary to define the content

---

1 The designer of the example of a Network Design course that we use in our system is Dr. Felisa J. Vázquez-Abad.

related to the domain or area of expertise (Decision-Making and Network Design) to be used in the authoring tool. At this level, it was clear that the author needed to receive examples and some support to perform tasks.

Task analysis

The general tasks for authoring a learning session were broken down in sequences of more detailed tasks and these were refined until they identified specific actions (click a button, write an instruction) that an author had to perform in order to create a learning session. Once these actions were identified, the different types of support were also defined. For example, we pointed to the events that call for learning support and problem-solving support.

Definition of authoring tool contents.

In this phase, the contents to be displayed as part of the authoring tool were defined and included in the database.

Prototype of the authoring tool.

A first prototype of the authoring tool is produced with no help nor adaptation added. This phase involves many activities:

- Implementing the Web site. We applied the *Relationship Management Methodology* (RMM) for Hypermedia Design (Isakowitz *et al.*, 1995). This method helps the designer to identify the different nodes of a Web site and their links and types of navigation. The hypermedia structure is seen as a set of relationships among information objects (nodes). RMM is a seven step methodology to manage relationships between information objects of a high structure and high information volatility. Using an entity-relationship diagram, relevant entities are selected to group their attributes. The access to the groups of attributes (slices) is designed by using uni-directional and bi-directional links between slices of an entity. Navigation between entities are implemented by means of menus, guided tours or indices. An advantage of RMM is its focus on the design of the navigation through a hypermedia application.

  The navigation design starts in step one (E-R design), to define the entities and their relationships; this step is useful to detect data to be reused. In step two (slice design), the presentation of each entity and the user access are defined. Each entity is split into slices as in a hypertext network (interconnected slices), and one slice is chosen as the head of the entity. This step generates an entity-relationship enriched with information about slices (E-R+ diagram). Third step involves the design of the paths to navigate. This means that information components, their groups and the ways to access them are defined. Specific entity attributes are grouped and menus or guided tours are designed to allow the user's navigation. The result of this step is a Relationship Management Data Model (RMDM) including all the access structures to be used in the hypermedia application. The next step is intended to map the RMDM into objects, such as a HTML form, a hyperlink or a pop-down menu. In step five, different objects (buttons, images, text fragments) are arranged in a particular layout screen. The sixth step is critical because the designer must define how the navigation will work. For example, navigational mechanisms are implemented to bring the user forward or back between different entities, namely Web-pages. The last step consists in the construction of the hypermedia application. Figure 6.10. shows the seven steps sequence.

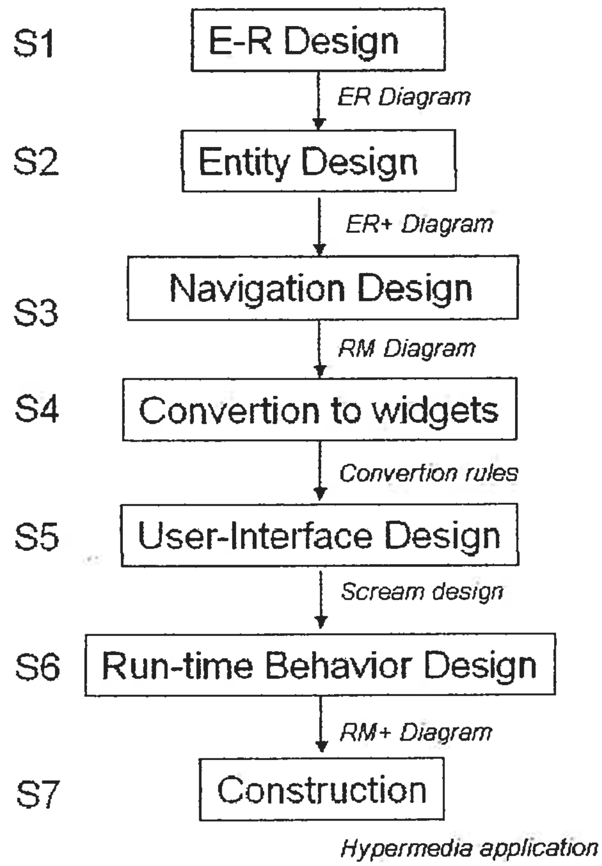


Figure 6.10. Sequence of the seven step *Relationship Management Methodology* (RMM) for Hypermedia Design, adapted from (Isakowitz *et al.*, 1995).

- Creating the Teacher Model, the Curriculum, the Planner, an initial Session Manager, the resources for the authoring tool and all the components required in order to have a functional Web site.
- Building the interface
- Creating Web pages.
- Coding the different fields in each template in order to make the integration of help in Part 2 easier.
- Implementing the navigation between Web pages.

A page-builder engine was developed to generate Web pages dynamically.

6.2.2. Part 2. Implementing and integrating the help system.

Identify variables that call for adaptation

In this step, it was necessary to define the information that the system had to take into account for the adaptation. In ARIALE, the main sources of adaptation are

User Performance (mouse events, for example)

User Goals (plan to be performed)

User Knowledge (previous experience teaching a subject, for example)

User Groups (recommending to the user something that is common to the user's group), and

User Personality and Cognitive Style (preferences and examples used for teaching)

Related to these sources of adaptation, we use three more specific types of variables that call for adaptation: variables activated by a system event, such as when a Web page is loaded or unloaded; variables that are activated when the teacher performs events explicitly, such as clicking a link; and, variables that are activated without any explicit intention of the user, but result from his interaction with the system. For example, when the teacher's teaching style changes, the system recommends other teachers' sessions, which are different from the sessions suggested previously.

Determine the type of adaptation to be performed

In this case, the help content, the medium to display help messages, and the help techniques are the most important aspects that are adapted in ARIALE. In the case of help techniques, the recommendation of sessions and examples was also one of the types of adaptation determined.

Define the inference mechanism.

This is the mechanism that reacts to changes in the teacher's information or triggers functions when an event occurs and decides what modifications have to be made in the interface. For example, generation, testing and recommending a topology that matches the teacher's teaching style is a process that the system performs when the teacher clicks a button to generate a new topology in the network editor.

Definition of help contents

The actual texts, images, animations and video for the help were defined in this step. The generator of examples was also created.

Definition of the content structure

The organization of the help content included not only defining attributes to classify help resources but also defining the classes of examples to be classified and developing an example classifier.

Definition of the relations between fields in templates and contents.

In this step, we linked the help to the interface components (indexation). For example, the recommendation of sessions was linked to the Session Selector Web page. The different texts for Introductions were indexed to each "MORE" link by means of a table in the database.

Because the approach for the development of ARIALE is prototyping, the method explained previously is iterative and many aspects can be improved during later repetitions.

6.3. Implementation

ARIALE is a prototype of a Web-based assistant made up of:

- A Web site including
    - an authoring tool to create learning sessions in teaching network design and
    - a help system to support teachers while they are using the tool.

The intelligent help system has two main functions:

Generates, tests and recommends examples and learning session to teachers

Offers adaptive context-sensitive and Web-based help about how to use the authoring tool.

These general solutions involve more specific problems and their corresponding solutions. In the next section, we describe the organization of the Web site, the general layout of the system interface and the structure of the authoring tool.

6.3.1. The Web site

While there are available simulations, tools, and resources, there is not a Web site that supports the gradual development of expertise according to the Pedagogical Model explained in chapter four. Our solution to this problem is a Web site that allows teachers to access the site from wherever they are. The justification for using a Web site instead of a stand-alone desktop application is the possibility of offering easy access with minimal or no requirements of installation of software on the client-side. The benefit of using a Web site is that it affords the access not only to university professors but also to any teacher interested in supporting teaching in a distance-learning environment.

This Web site is developed using HTML (Hypertext Mark-up Language), JavaScript, Java Server Page (JSP), Java Servlets and Applets. The server that supports the site is Tomcat 3.1. The content of the Web page is stored in a database supported by MySql 3.23.58.

The Web site can be seen as a hypertext structure with a main node (Home page) and two branches: an area for the general public and an area for registered teachers. In the area for public access there is information about the services available, the option of registration for new users, and the link to the Web page for registered teachers. In the area for registered teachers, there is the authoring tool composed of a set of templates called "Steps". After logging in, a teacher can access the steps through the "Session Selector" Web page. The navigation between steps is sequential, but the teacher is free to jump from one step of a task to another task. Figure 6.11. illustrates this structure.

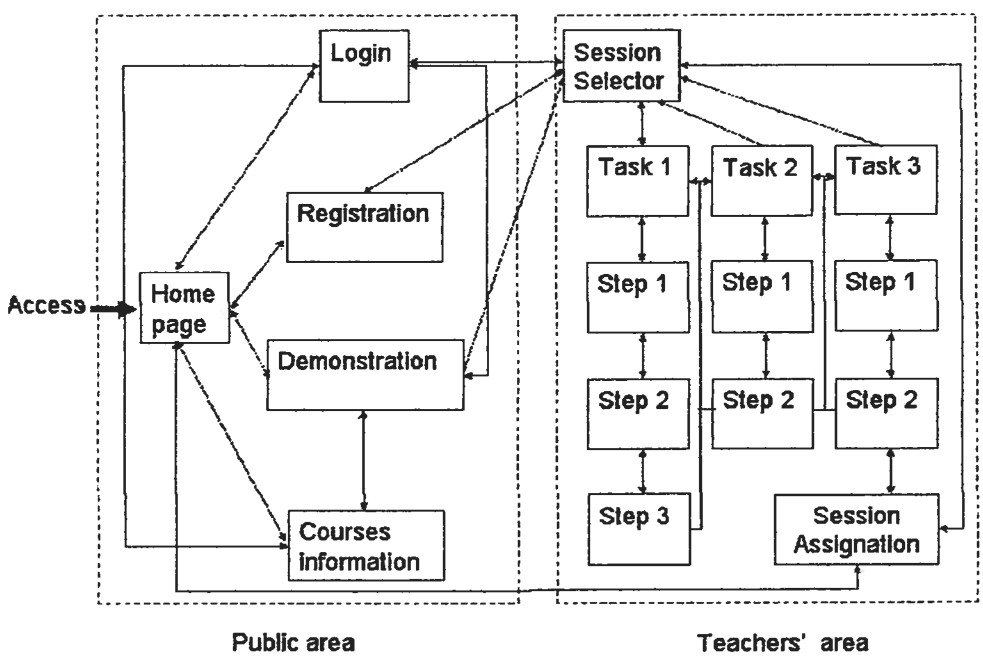


Figure 6.11. Structure of the Web site.

6.3.2. General user interface layout

In many applications, a teacher must switch from one window to another window or frame to check the help information. This type of switching from one window to another requires the teacher to remember (to keep in his short-term memory) help information from a help window in order to apply it in the authoring tool. Sometimes the help window can be confused with other windows and the switching process becomes more disruptive. This is a cognitive overload that needs to be eliminated or at least reduced. In order to offer less disruptive help, our system has a standard layout for all the Web pages in the site. We decided that

- our system does not use additional windows that oblige the teacher to interrupt his workflow and take him out of his current environment. The use of additional windows will be permitted only in the case of problem-solving support.
- the left area of the screen is used to display the menu for navigation
- the authoring area uses the middle area of the screen and
- the right area of the screen is used to display help information.

These decisions are based on the following four criteria:

- *Parallelism.* Parallelism (Yu *et al.*, 2003; Yu, 2002), means that information required by the user can be displayed simultaneously in a parallel manner, for example, a column of information on the left side accompanied by a column with complementary information on the right.
- *Embedded help.* Embedded help (Weber, 2000) is displayed in the same screen and window that the user is working on.
- *Shortened line of text.* The line length of text used in this layout is between 35 and 70 characters to avoid excessive movements of the eye and neck from the end of one line to the beginning of the next line (Lynch and Horton, 2001). These movements slow down reading causing retention rates to fall.
- *Short paragraphs* enable the teacher to scan instead of read. Shorter paragraphs and shorter lines of text allow the user to quickly scan and find the information he is looking for (Spool, 1999; Murray, 2003). Spool found that people do not actually read Web pages because they prefer to scan text in order to find keywords.

Figure 6.12. shows the standard layout of the user interface for the Web site. This layout is intended to support novice teachers, and it remains almost the same for advanced teachers, except some changes are made to benefit the latter's use of the Web site.

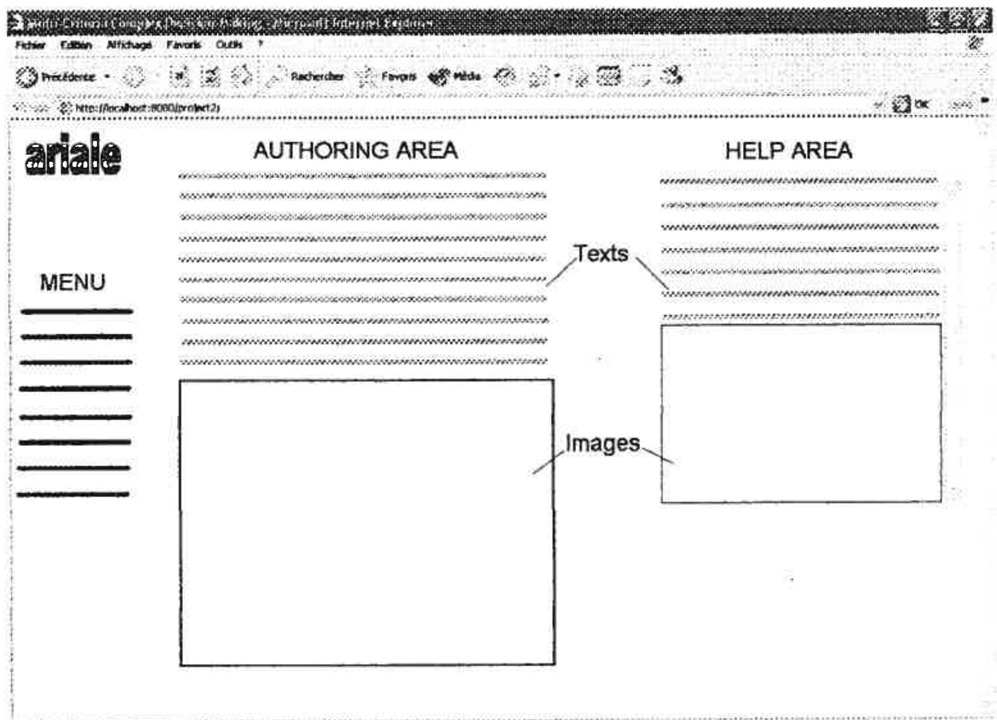


Figure 6.12. The standard layout of the user interface.

The standard layout is important in order to help the teacher create an image of the tool. It is difficult for any user of Web sites to create a mental model of each application they can access, but an advantage of using a standard layout is that the teacher will be able to recognize the functions of the authoring tool easily. Another advantage of this interface structure is the possibility of generalizing and then applying it to other activities where we can have a main working area as the authoring area and a complementary area as the help area. For example, in an e-commerce Web site it is possible to find a main area for electronic payment and a complementary area with adaptive context-sensitive help in the same Web page. Using this three-area layout, the system can change the contents in each area easily.

6.3.3. Menus

The menus support three levels of navigation with a different menu for each level of navigation:

- For the general public (visitors, non registered teachers).
- For the selection of learning sessions.
- For the authoring environment.

A general rule in this research is to maintain a small number of options that the teacher has to analyse when making decisions. Because Cognitive Load Theory (CLT) recommends not using a greater number of options for selection, each menu has no more than seven links. The use of this type of context-adaptive menu is helpful in this project and in other projects where very complex menus can generate cognitive overload. This is an application of a technique of navigation that recommends hiding or disabling links in order to support the user's navigation (Brusilovsky, 1996).

Level for the general public

This level allows any person to navigate the site without using the authoring tool. The links in this menu are:

Courses. This link shows the Home page of the Web site.

Network Design. This links shows general information about the Network Design course.

Health Services. This links shows general information about a potential course related to Medicine.

Financial Risk. This link shows general information about a potential course related to Finances.

E-Auctions. This link shows general information about a potential course related to Electronic Auctions.

Login. This link allows any registered teacher to enter his username and password in order to use the authoring tool.

Register. This link allows any person to enter his data and become a registered teacher.

Demonstration. By clicking this link, it is possible to see a demonstration about what it is possible to do on the Web site.

Example. This link allows any person to test a limited version of the authoring tool without saving their changes.

French and *Spanish*. These links allow the visitors of the Web site to choose the language used to display information. Figure 6.13. shows the menu for the general public.

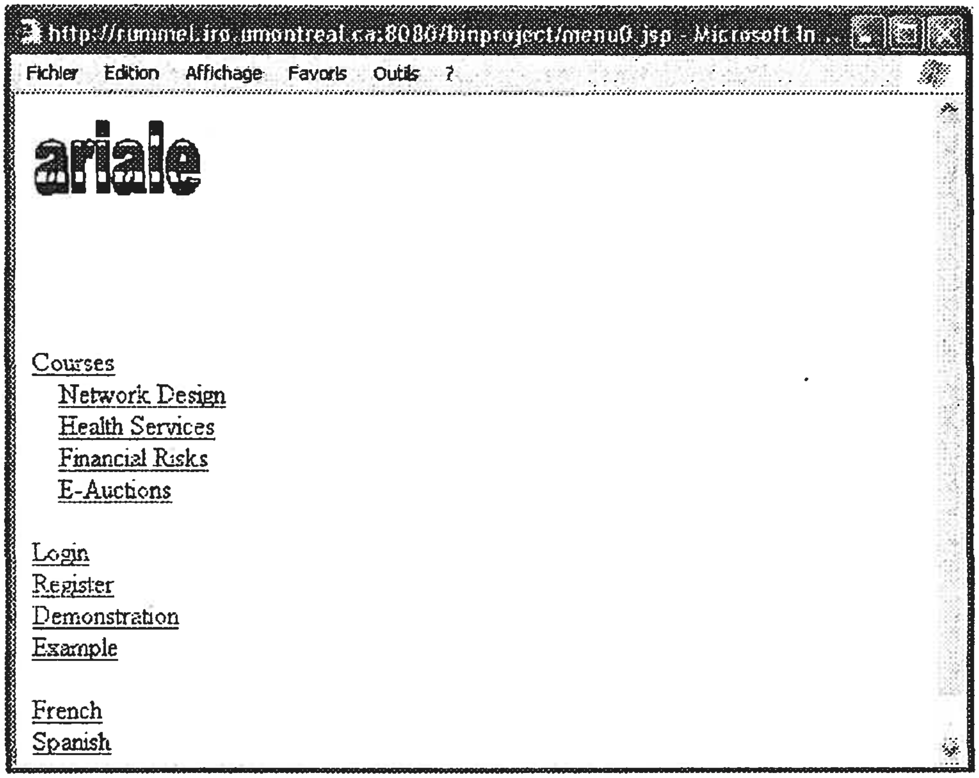


Figure 6.13. The menu for the general public.

Level for the selection of learning sessions

When a teacher logs in the Web site after providing his username and password, he enters the level for the selection of learning sessions and sees a menu (See Figure 6.14.) associated with this level. This menu allows a teacher to navigate the authoring tool at the level of selection of learning sessions. This menu reduces the options to the minimum because the teacher has few tasks to perform. The links in this menu are:

Courses. This link shows the Home page of the Web site.

Sessions. This link shows the page to select a session in order to edit it.

Profile. This link shows the tool to edit the teacher's profile with his personal information, background, preferences and level of experience.

Logout. This link allows the teacher to close the current session and start another one by going to the Home page of the Web site. Figure 6.14. shows the menu for session selection.

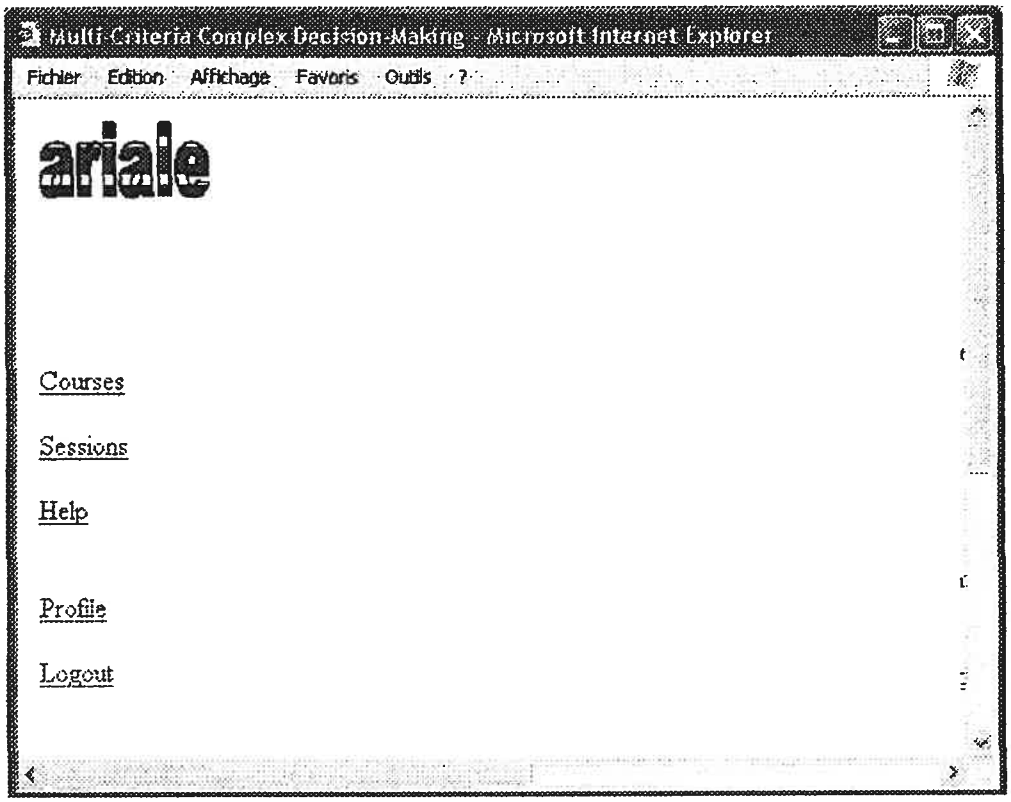


Figure 6.14. The menu for the session selection

Level for the authoring environment.

This menu (Figure 6.15.) allows a teacher to navigate the authoring tool to change activities or go to different tasks. This menu reduces the options to the minimum because the teacher has few tasks to perform. The links in this menu are the same as at the previous level plus the following links:

Activities. This link enables the teacher to select the activity he wants to author.

Task 1. This link lets the teacher select Task 1 in order to author it.

Task 2. This link lets the teacher select Task 2 in order to author it.

Task 3. This link lets the teacher select Task 3 in order to author it.

Figure 6.15. The menu for the authoring tool.

6.3.4. The Web pages for the general public

In the area of the screen that is used for the authoring tool, different information is displayed according to each level of navigation. For the general public there are contents that give basic information about the services available. These contents are related to the objective of the Web site, information about offered courses, a demonstration of the functions of the authoring tool, an access to a limited version of the authoring tool with a session, the registration forms, and the login form. The characteristics of the contents of the authoring tool area for the other two levels are described in the next section.

Courses (Home page)

The Home page shows general information about the objective of the Web site (See Figure 6.16.). This page is displayed when any visitor enters the Web site or when he clicks the "Courses" link in the menu for the general public.

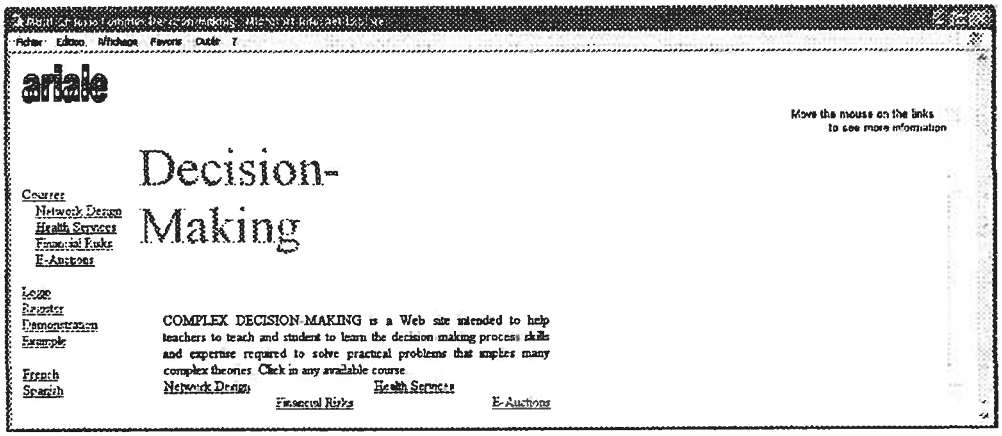


Figure 6.16. The Home page of the Web site.

Course information pages

Each course information Web page shows general information about courses available currently and others available in the future. In Figure 6.17. it is possible to see information about the Telecommunications course. These pages are accessed by clicking the corresponding links in the menu for the general public.

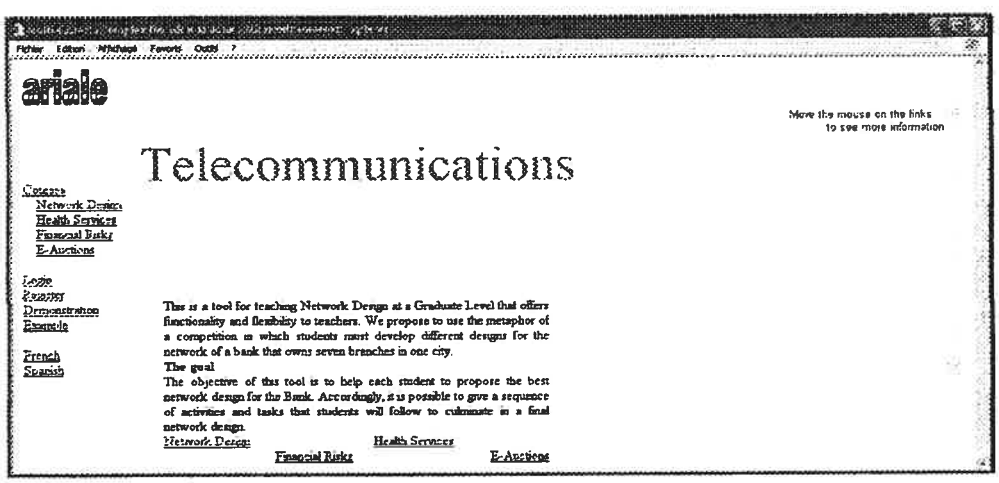


Figure 6.17. Information about the Telecommunications course.

Demonstration

The Demonstration Web page is a slideshow that displays images and text with information about the functions of the authoring tool. This option allows any visitor to get a general idea about what the Web site provides. Figure 6.18. shows a slide of the demonstration.

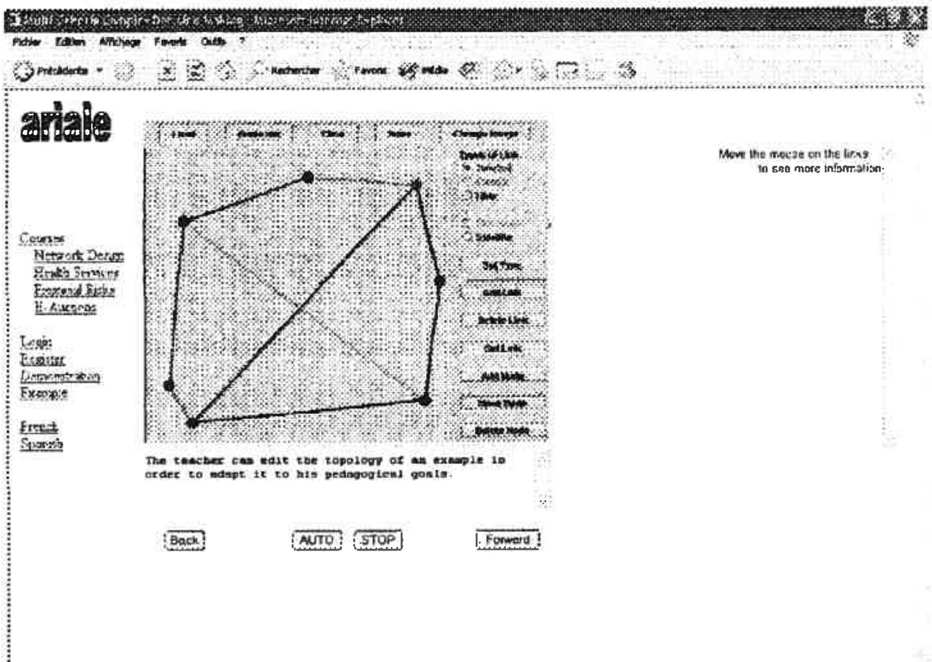


Figure 6.18. The demonstration shows the services of the Web site.

Access to an example

The area for the general public includes a worked out example of Task 1 of the Reliability activity of the Network Design course (See Figure 6.19.). Any visitor can, as a guest, access this example, edit it, test the help available, and get a rough idea about what he will be able to do after registering. This option allows any visitor to get a general idea about how the Web site works.

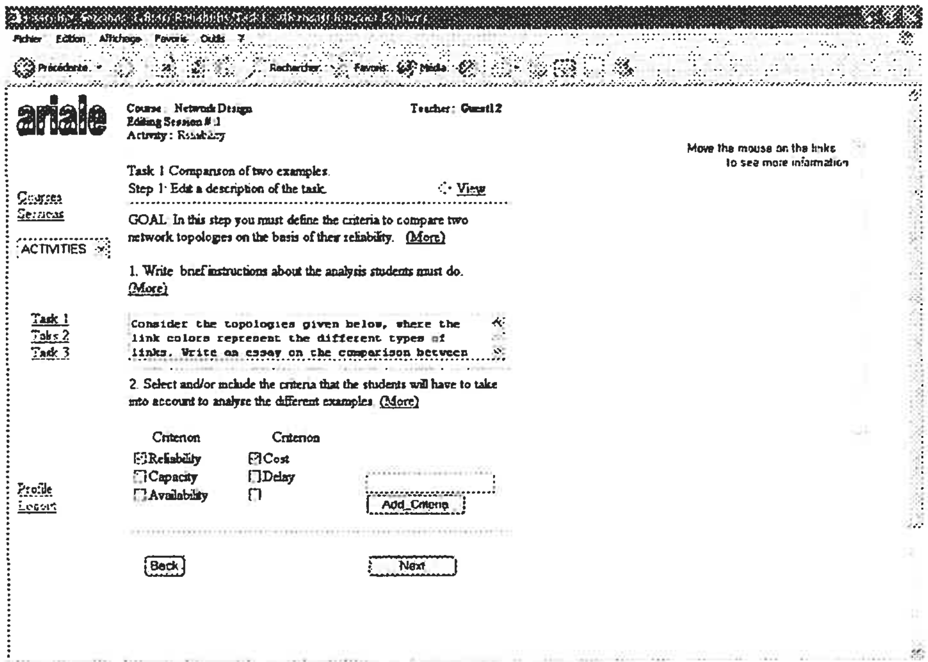


Figure 6.19. An example Web page of the authoring tool is available for general public.

Register

The registration process is the method that the system uses to acquire explicit teacher data. The registration is a set of forms that the teacher has to complete with his personal information, preferences and academic background. Data are stored in the database and the system uses this information to initialize the Teacher Model (see chapter four). When the teacher finishes the registration phase, he receives a username and password (See Figure 6.20.) and the system offers him the option to start using the authoring tool (See Figure 6.21.).

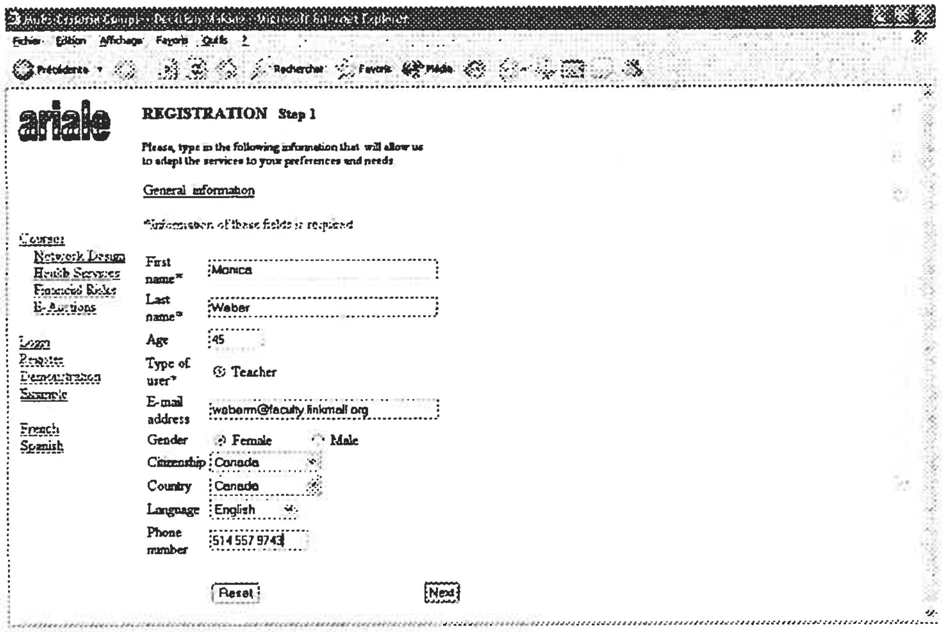


Figure 6.20. The system uses the Registration process to acquire explicit information from the teacher in order to initialize his Teacher Model.

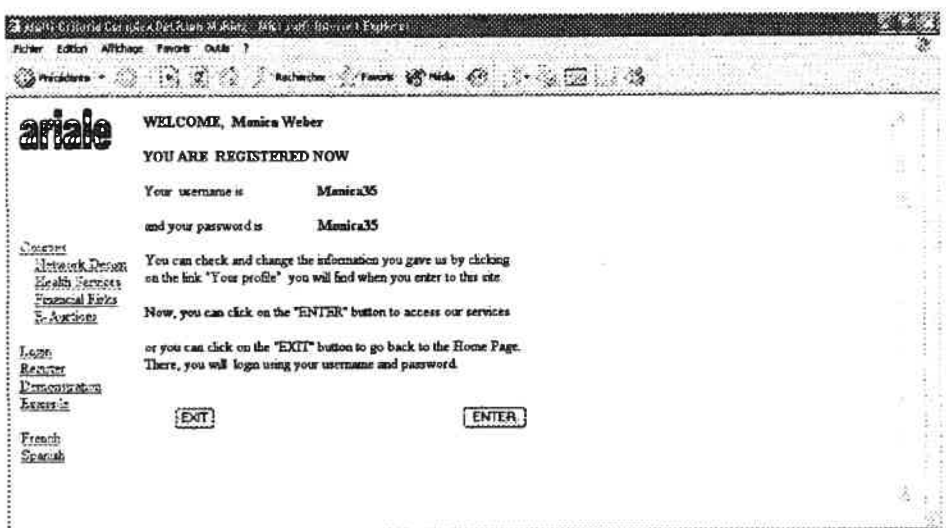


Figure 6.21. At the end of Registration, the Teacher Model has been initialized and the teacher receives his username and password.

Login

After registering, the teacher has his login and password needed to log into the authoring tool. Once the teacher's username and password are validated, the teacher can access the authoring tool and the system simultaneously loads the corresponding Teacher Model. Loaded in memory, the information in this model supports the decision that the system will make. A snapshot of this Web page is shown in Figure 6.22.

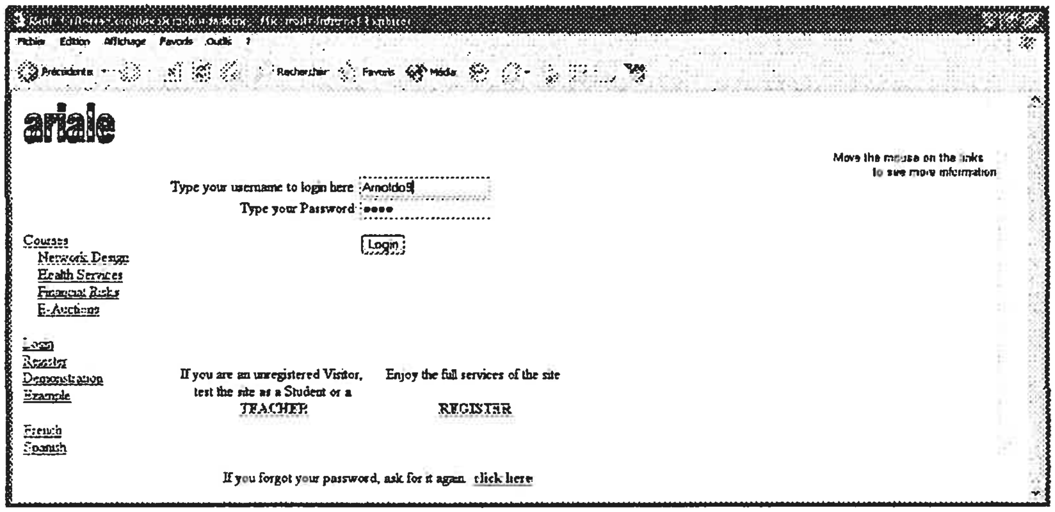


Figure 6.22. The login Web page validates the teacher's username and password.

6.3.5. The Web pages for the implementer teacher

We built an authoring tool for teaching expertise in decision-making, a process that is complex and has to be developed gradually by repeating similar exercises until achieving an expected degree of expertise. This tool is a Teacher's Virtual Authoring Space (T-VAS), which allows each teacher to adapt contents to his particular pedagogical goals. The concept of T-VAS derives from the Teacher Virtual Office (TVO) used by BestNet (Vázquez-Abad *et al.*, 2001). TVO is an environment where the teacher can manage the information about courses (marking, assignation of exercises, etc.).

In general terms, the objective of this authoring tool is to *decrease* the skills and time needed by an implementer teacher to create or adapt an existing curriculum to his particular goals. In order to achieve this objective, the authoring tool supports the teacher to modify and readapt some contents to create specific learning sessions. For example, contents can be modified by entering new contents in forms designed with this goal. The teacher can also reuse the resources provided by the designer and other teachers because the authoring tool allows each implementer teacher to select learning session from a list of session created by other implementer teachers. As a result, a teacher can build a learning session, either by reusing examples or by changing previous sessions. After that, any learning session and resource added or changed by a teacher is saved in a base of sessions to share with other implementer teachers and designers.

Our authoring tool is neither a tool for designing a complete instance of a curriculum nor a tool to change the structure and functionality of the Web site; instead, this authoring tool enables implementer teachers to adapt some contents of a previously-designed curriculum in order to reflect their own teaching styles. The tool is simple in order to encourage its use by teachers who find authoring tools difficult to use. Consequently, our authoring tool is the result of tradeoffs among scope, depth, learnability and productivity (Murray, 2002).

Functions and layout

This section presents the design of the Web-based authoring tool in order to answer the following question: *Which specific functionalities and characteristics of an adaptive hypermedia authoring tool will enable teachers to adapt learning sessions to their requirements, limitations and pedagogical goals?*

A learning session is a set of Web pages with a sequence of tasks and steps that students have to perform. The authoring tool allows the teacher to insert part of the information that the students will see in each Web page. This tool also includes an example editor that is explained later. The information that the teacher can edit refers to his pedagogical goals. A learning session is also a set of Web pages with a sequence of tasks and steps that *teachers have to edit* in order to author a learning session for the students. By editing some fields in each Web page of the authoring tool, the teacher adapts or customizes the learning session according to his teaching style. The system stores the information entered by the teacher in a database and the system can reuse this information and build the Web pages that the students will view. The main functions of the authoring tool are to:

- provide a Teacher's Virtual Authoring Space (T-VAS) that allows each teacher to adapt contents to his particular pedagogical goals.
- acquire and save the instructions, criteria and pedagogical goals that the teacher wants the students to take into account in order to perform each task.
- acquire and save the examples the teacher wants the students to study in each task.

- allow the free navigation among different activities, tasks and steps.

To support these functions, the authoring tool has some Web pages that map the order of activities and tasks described in chapter four.

The selection of sessions

In order to load learning sessions, the authoring tool has a Session Selector Web page that allows a teacher

- to create a new session from scratch
- to reuse his old own sessions or sessions created by other teachers.

For a novice teacher, the session selector Web page offers a short presentation of the objective, procedures, activities and tasks to be performed. There is help associated with the different parts of the presentation. In the case of an advanced teacher, this presentation is omitted. The Web page also has two buttons:

- The "Create a session" button, which allows the teacher to pass to the first step of the first task of the first activity to be performed.
- The "Open a session" button, which offers a list of available sessions that the teacher can edit.

The session selector Web page is the space that allows *teachers with similar characteristics* to share their sessions, reusing them partially or completely. Beside the "Open a session" button, the system offers the list of the current teacher's finished sessions followed by the other teachers' sessions. The method to find the teachers sharing similar characteristics is explained in chapter four. In chapter five we explain how the system recommends other teachers' sessions. Figure 6.23. shows a Session Selector Web page for a beginner teacher.

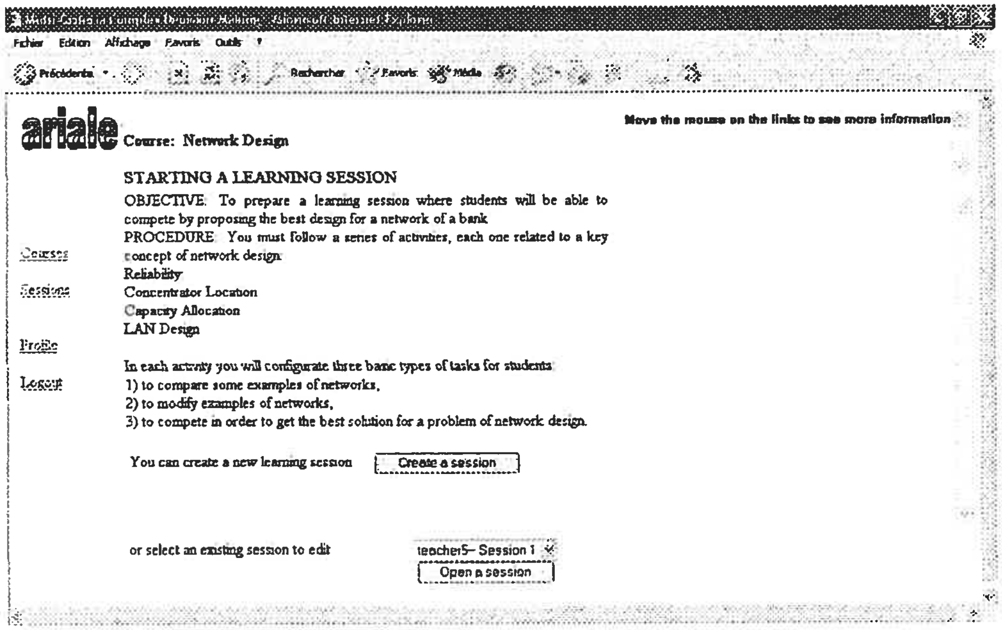


Figure 6.23. The Session Selector Web page is the gate to start authoring a learning session.

The Web pages for authoring sessions step-by-step

In general, the Web pages of the authoring tool provide information and mechanisms to acquire and save the instructions, criteria and examples the teacher wants the students to study in each task. For each task there are particular templates, one for each step to be performed. Each template has a particular structure composed of different sections and fields. There are four sections in each template (see Figure 6.24.):

Head. This section provides information about the current course, teacher, session, activity, task and step.

Goal. This section instructs the teacher about the task he has to perform in the current step.

Edition. This section provides instructions and mechanisms for entering or changing information and for creating examples.

Navigation. In addition to the menu, there are buttons that allow the teacher to go ahead, back or jump between the steps. Figure 6.24. shows an example of sections in a step.

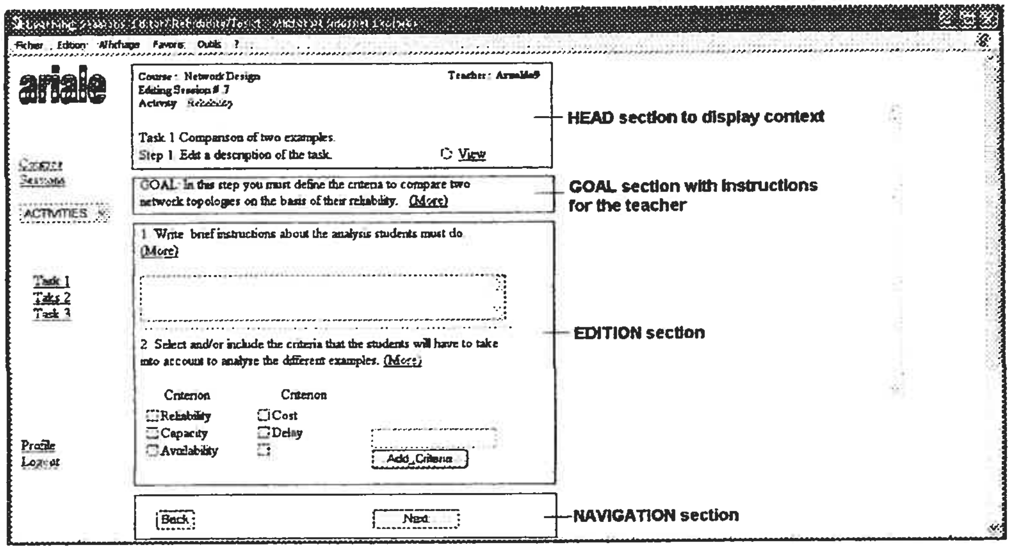


Figure 6.24. Each step has four sections: head, goal, edition and navigation.

Each section has fields that change or not depending on their functions:

- *Variable fields.* There are two types: fields that change with each step and fields that change when the session or the activity changes. The fields that change with each step are the descriptions of the *task*, the *step*, the *step goal* and the *instructions* that the teacher must follow. These fields change every time a new step is loaded. For example, the step description in a Web page is "Edit a description of the task" and in the following step the step description changes to "Edit a description of Example 1". The fields that change when the session or the activity change are the descriptions of the session and the activity. The description of the course and the username are fields that remain without change during the entire time the teacher is authoring a learning session. For example, the activity description changes once an activity is finished or when the teacher jumps to another activity.
- *Fixed fields* do not change. For example, navigational buttons (next, back), the links "View" (to view the result of the edition of a step) and "More" (to see more help) are constant.
- *Session fields.* These are the fields that the teacher edits in order to tailor the session according to his pedagogical interest. For example, the teacher can write some lines of text with instructions for students and click check-buttons in order to define some

values that students will have to take into account. Figure 6.25. shows different fields.

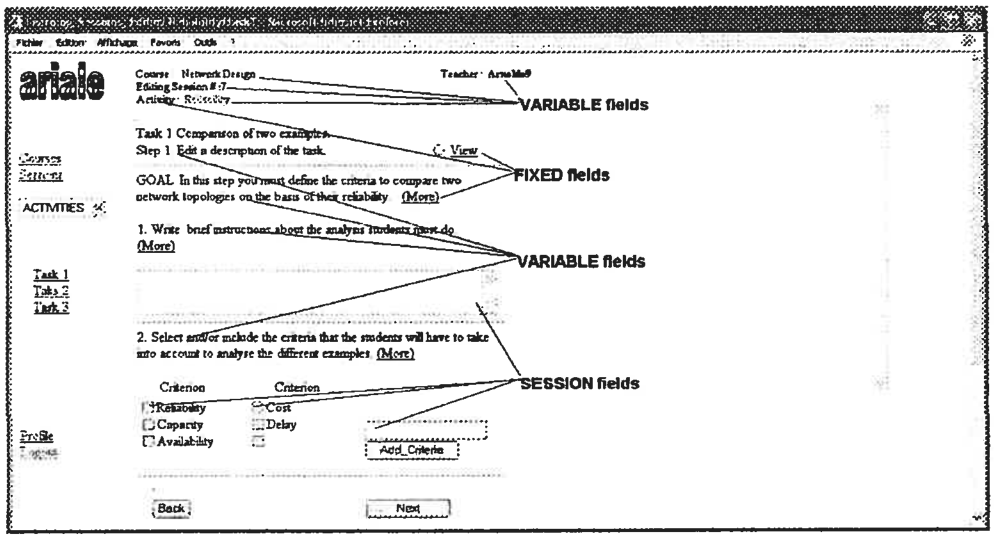


Figure 6.25. The fields display information that changes or not with each step.

The process of authoring a learning session consists of many Web pages and each one is a step in this process. Each activity of a learning session always includes three tasks and each task has some steps. Our system also includes a viewer that allows the teacher to check the result of his work. The steps are detailed in the next section. ARIALE includes an editor of examples specialized in the edition of network topologies, which is described in section 6.3.7.

6.3.6. Editor interfaces for different steps

Depending on each step, task and activity of the curriculum, the editor shows different interfaces with additional functions or disables some of them. The different interfaces used for the three tasks are explained as follows:

Task 1.

The *Task* 1 (Comparison of two examples) has three steps:

Step 1. The step 1 is intended to capture instructions and the criteria that the students will use to compare the two topologies of the Task 1 example. This step includes forms in which the teacher can write instructions for the students. Figure 6.26. illustrates Step 1 for

Task 1 and the embedded context-sensitive help (text-based) linked to the hyperlink "MORE". In Figure 6.27. there is another example of Step 1 with visual help.

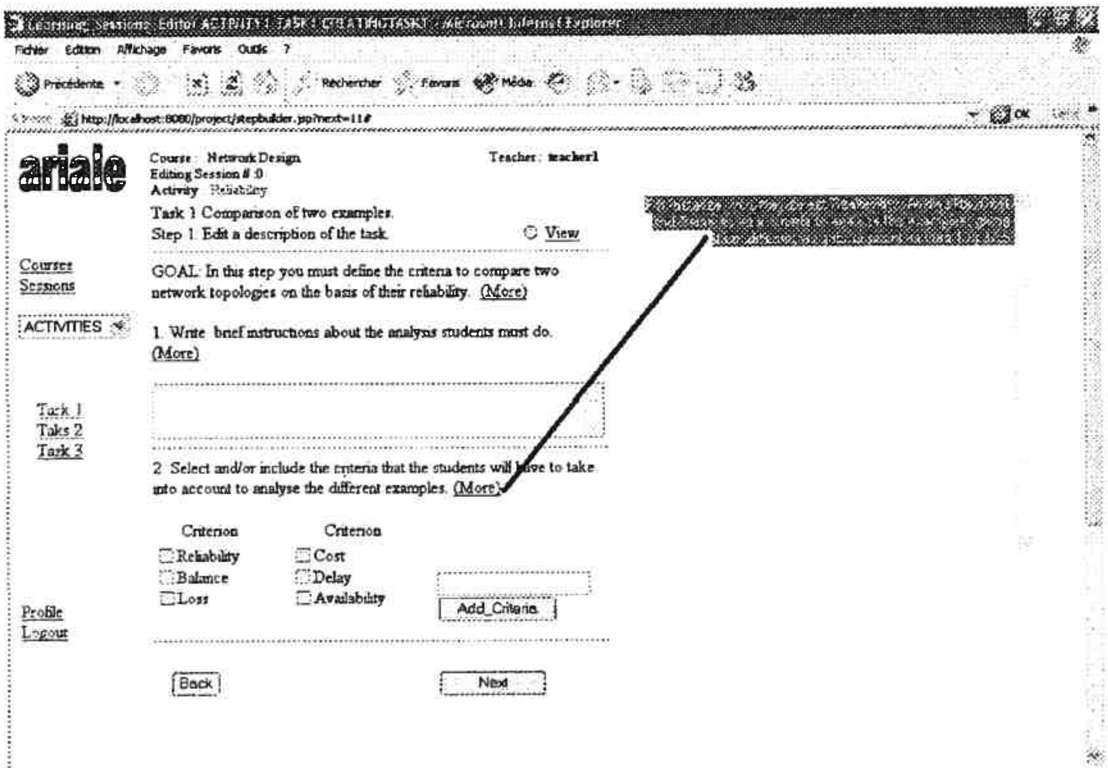


Figure 6.26. A variable activated by a click (Rodríguez et al., 2004a) in Step 1.

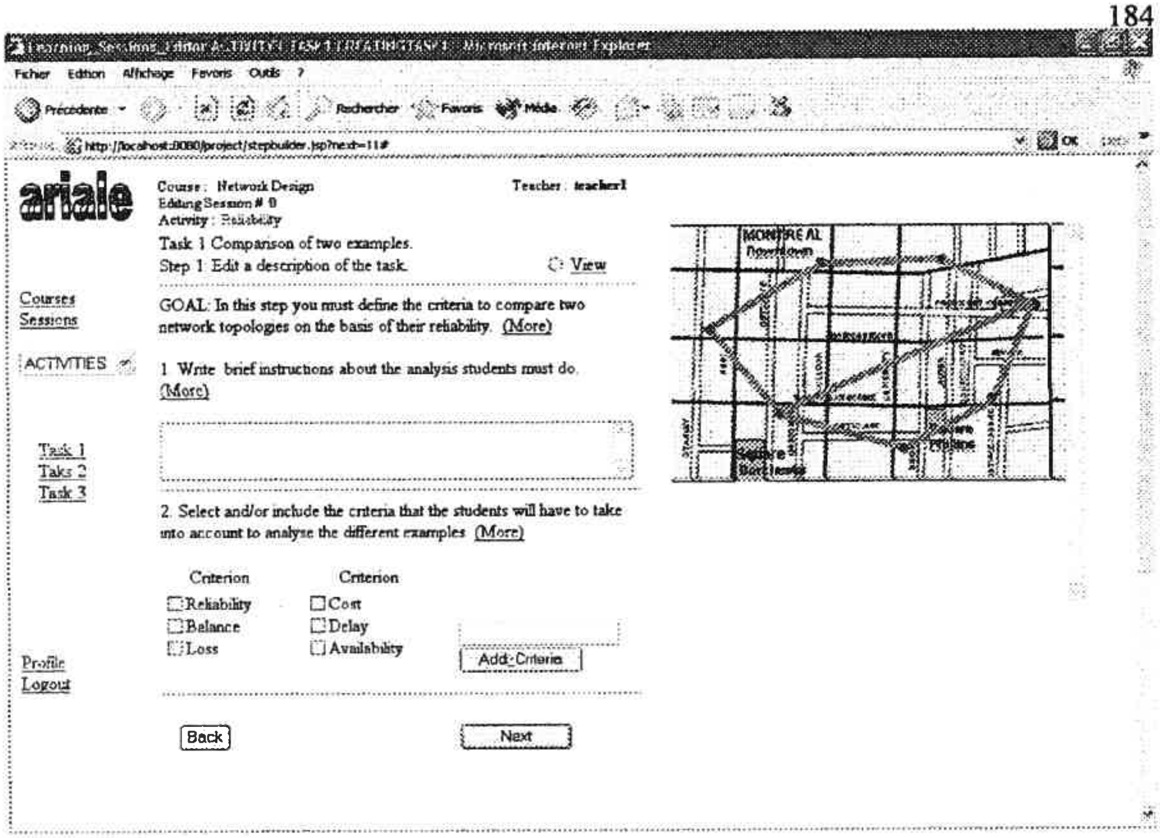


Figure 6.27. Step 1 interface with help using an animation.

- *Step* 2. This step allows the teacher to provide the first example he wants to use in order to compare it with the example of step 3. This step also includes a form for optional instructions and an editor to define the example that the students will compare with a counter-example.
- *Step* 3. In the step 3, based on the previous example, the editor supplies a counter-example. This example has the equivalent measures of performance, but the configuration of the topology is different.

An example viewer. Each step has a "View" function that displays the results of the authoring process. This viewer shows two examples in a student page. The viewer shows the network graphs, the link attributes and values, the types of links and the measures of performance if the teacher decides to show them. Figure 6.28. is an example of the interface of the example editor in step 2 of Task1.

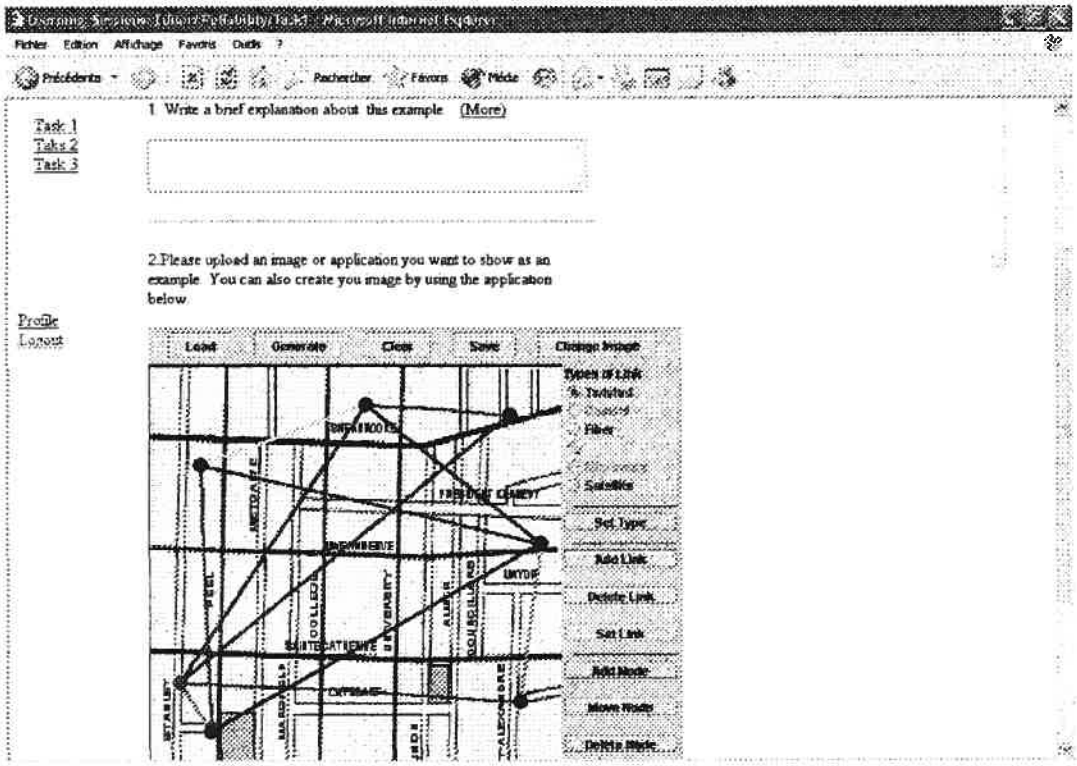


Figure 6.28. The interface of the example editor in Step 2 of the Task 1 of the Reliability activity.

Figure 6.29. shows the corresponding viewer interface for this step 2 of Task 1 and Figure 6.30. shows an example of a final result of authoring this task.

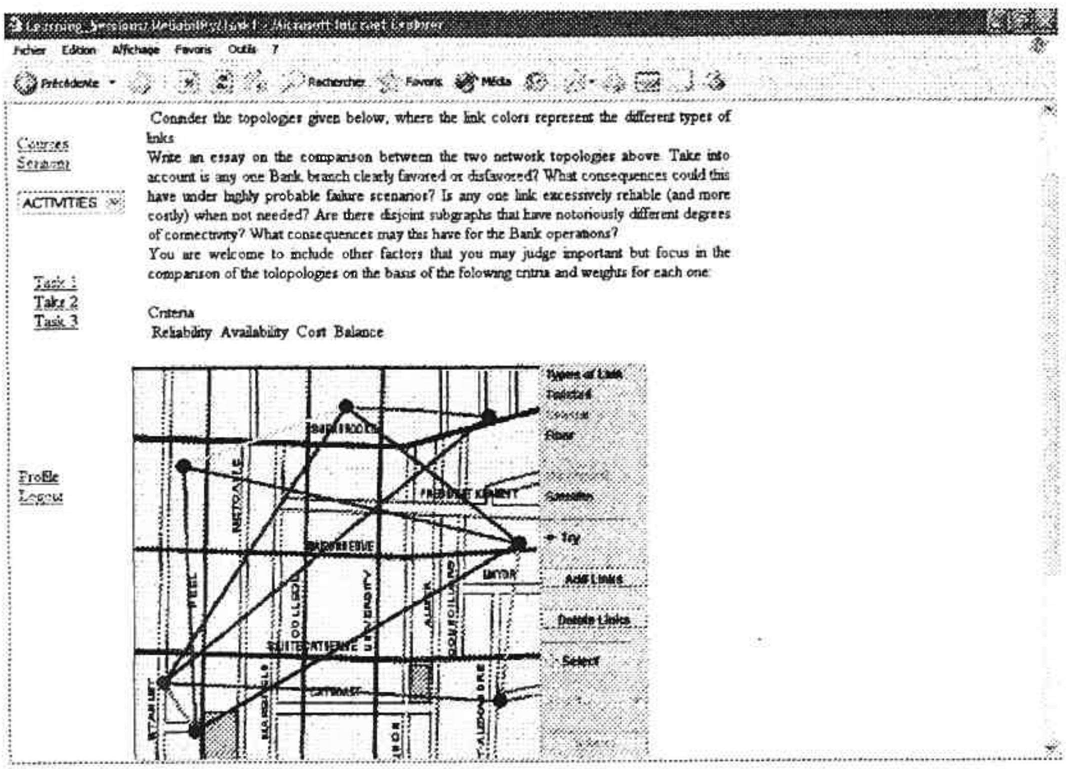


Figure 6.29. The interface of the Viewer in Step 2 of the Task 1 of the Reliability activity.

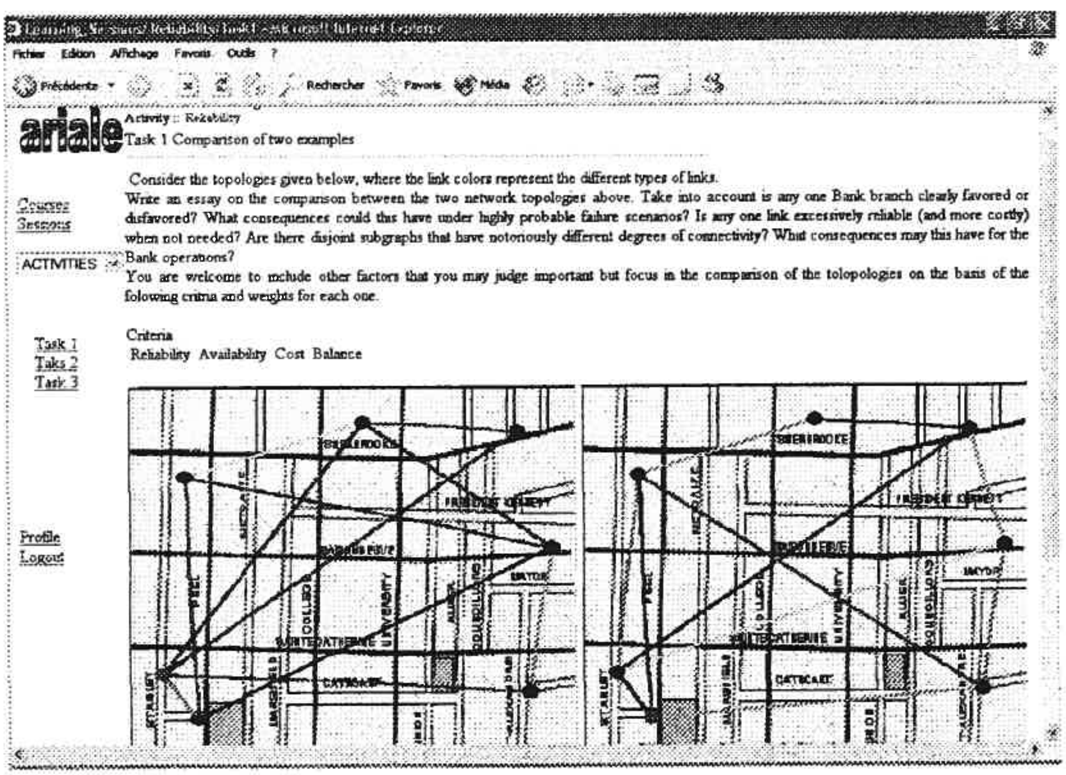


Figure 6.30. The authoring tool includes a function to view the result of authoring.

Task 2.

The Task 2 (analysis of sensitivity) has two steps:

- Step 1. The step 1 allows the teacher to define a topology with only one type of link where students will have to add or delete links in order to change the measures of performance of the topology, as in the case of the Reliability activity. In the case of the activity "Concentrator Location", the students have to play with changing the number and location of concentrators in the network, and, in the Capacity allocation activity they have to modify the capacity of a link in order to analyze the sensitivity of the network to these changes.

Figure 6.31. is an example of the interface of the example editor in step 1 of Task 2.

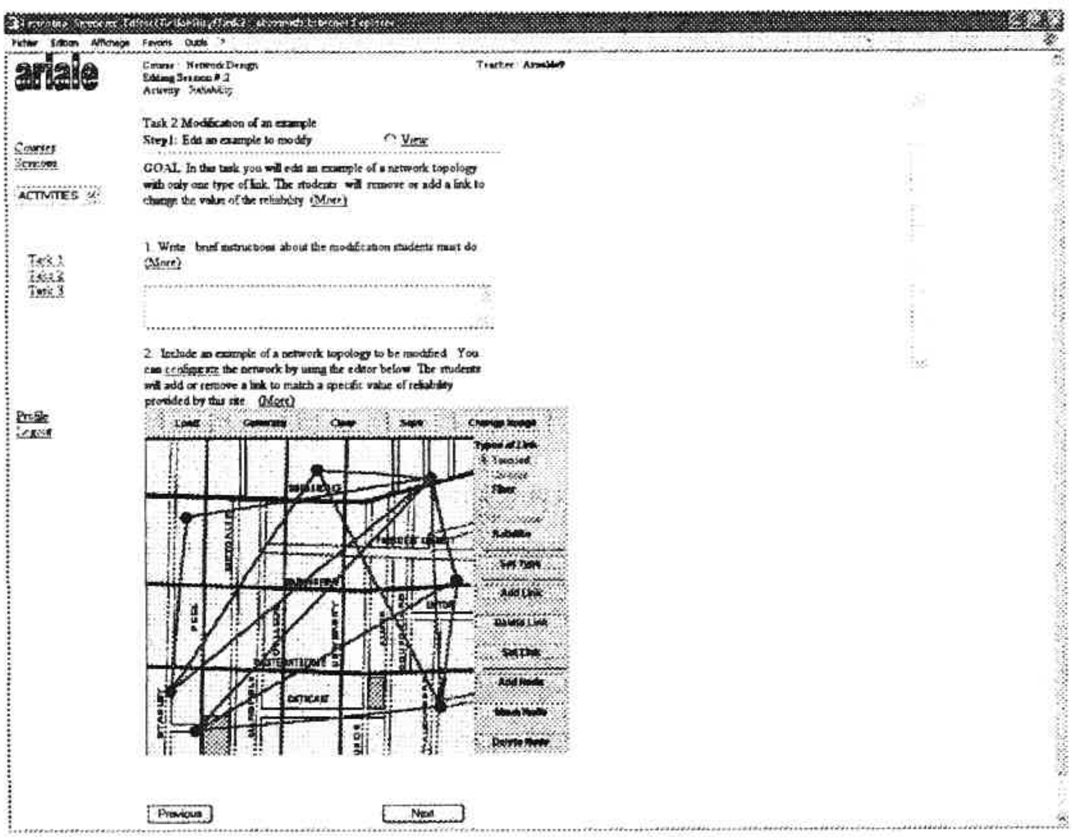


Figure 6.31. The interface of the example editor in Step 1 of the Task 2 of the Reliability activity.

Figure 6.32. shows the corresponding viewer interface of the example editor in step 1 of Task 2.

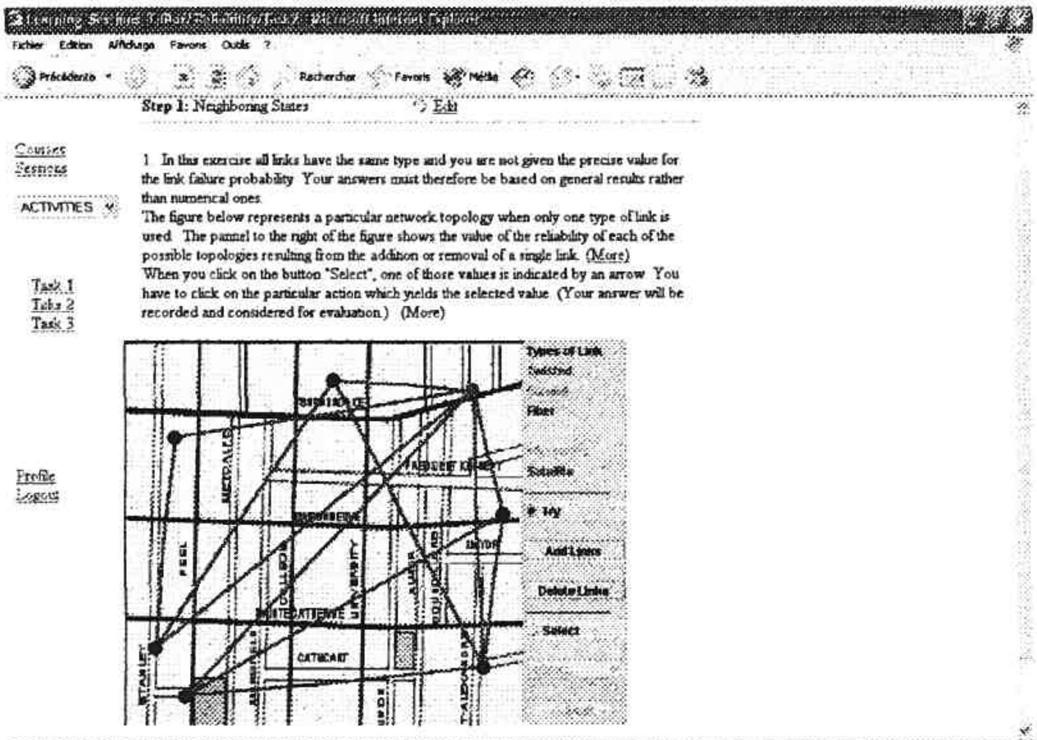


Figure 6.32. The interface of the Viewer in Step 1 of the Task 2 of the Reliability activity.

- Step 2. The step 2 of Task 2 is basically similar to the step 1, and the system offers this step because the teacher might want to specify another topology instead of reusing the previous example used in step 1. Figure 6.33. shows the corresponding viewer interface.

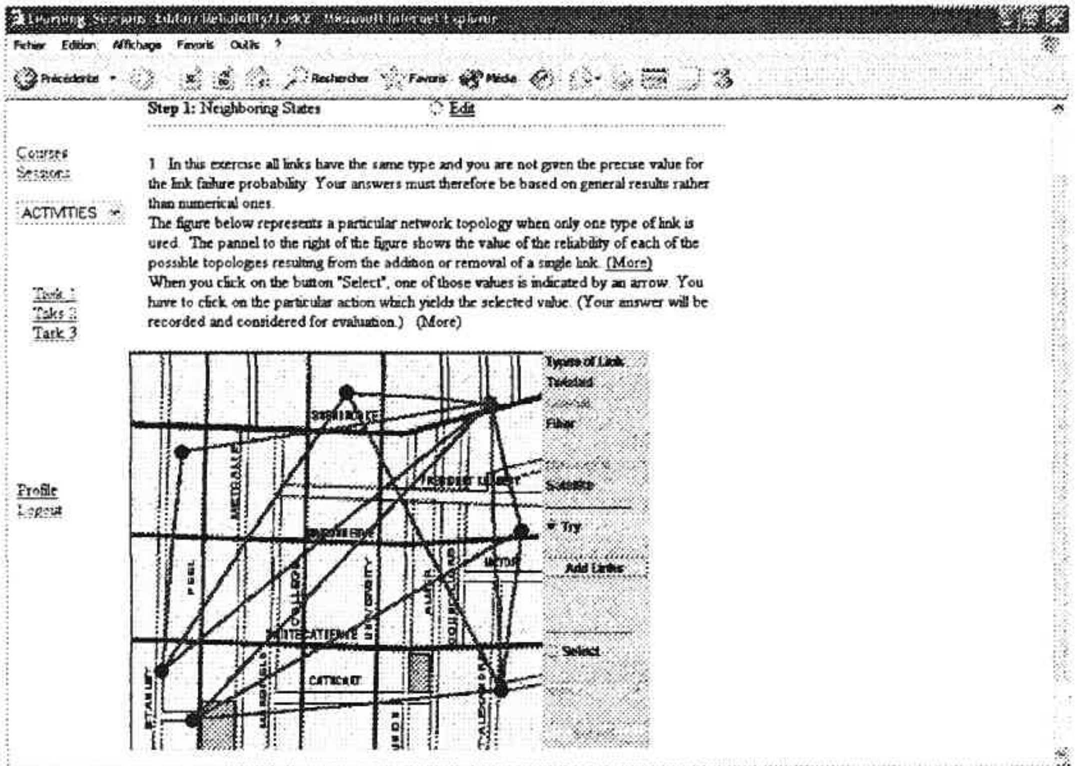


Figure 6.33. The interface of the Viewer in Step 2 of the Task 2 of the Reliability activity.

Task 3.

The Task 3 (competition to optimize a network design) has two steps:

Step 1. The step 1 allows the teacher to explain the rules that the students have to follow during the competition.

Step 2. The step 2 allows the teacher to define the goal that the students have to achieve. For example, the teacher can define a reliability level or a particular cost that the students have to achieve by modifying the given topology. In fact, the editor reuses the example provided in the step 1 of Task 1, but the teacher can create a new example or reuse one of the examples shared by other teachers. Figure 6.34. illustrates the interface of the example editor in Step 2 of the Task 3 of the Reliability activity.

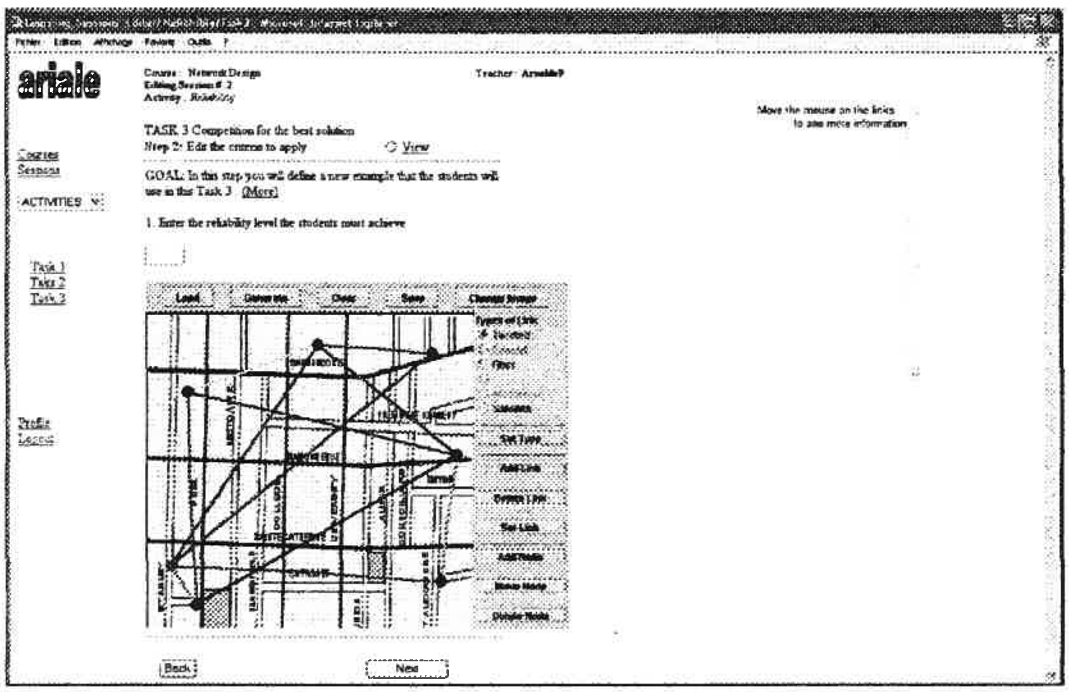


Figure 6.34. The interface of the example editor in Step 2 of the Task 3 of the Reliability activity.

6.3.7. Editing examples

The tool used to create network topologies is the topology editor (see Figure 6.35.). This editor allows teachers to edit examples of the type of network they want their students to analyse. It also allows teachers to try changing links and nodes until they get the network they want the students to receive. The example editor is based on the NetEditor developed as part of the BestNet project (Vázquez-Abad *et al.*, 2001). NetEditor is a tool with a graphical user interface used by the BestNet Web site to allow students to improve their skills in network design. The editor was designed to provide an easy tool to create examples of network topologies because the configuration of so many links is more difficult in a text-based tool. Our editor allows the teacher to create a static model of a network.

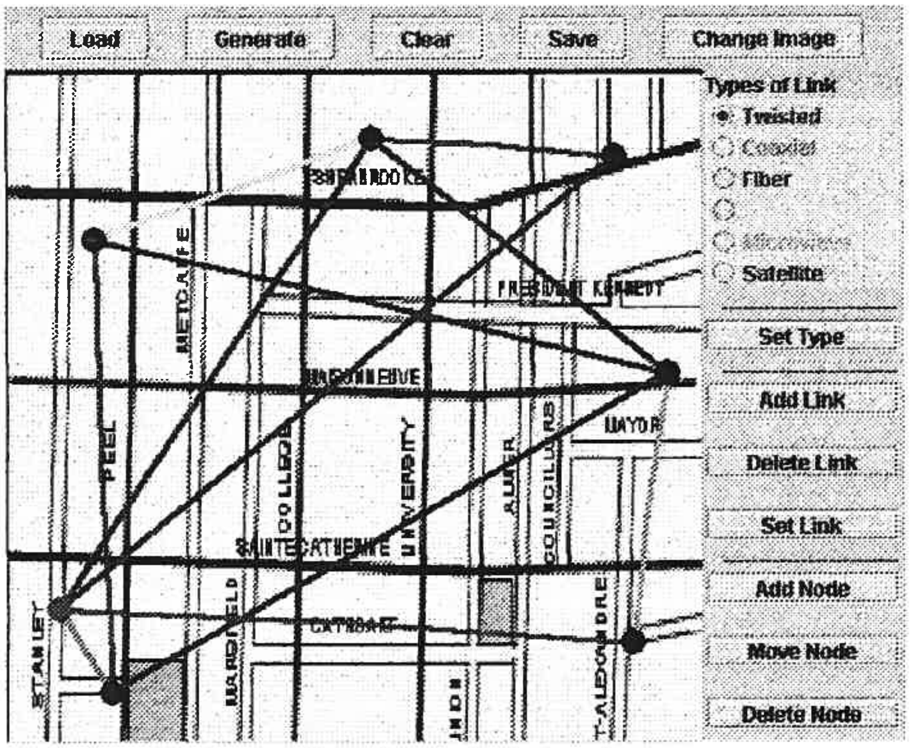


Figure 6.35. Interface of the network editor.

While the teacher is authoring a session, he can use this editor to define the configuration of a network topology. Using the graphical editor, the teacher can manipulate nodes and links as drag-and-drop objects with different properties. These properties can also be edited. Most importantly, the graphical editor allows the teacher to visualize the network configuration in a WYSIWYG[1] style. Figure 6.36. shows a general view of the graphical user interface of the example editor embedded in a Web page.

---

[1] What You See Is What You Get

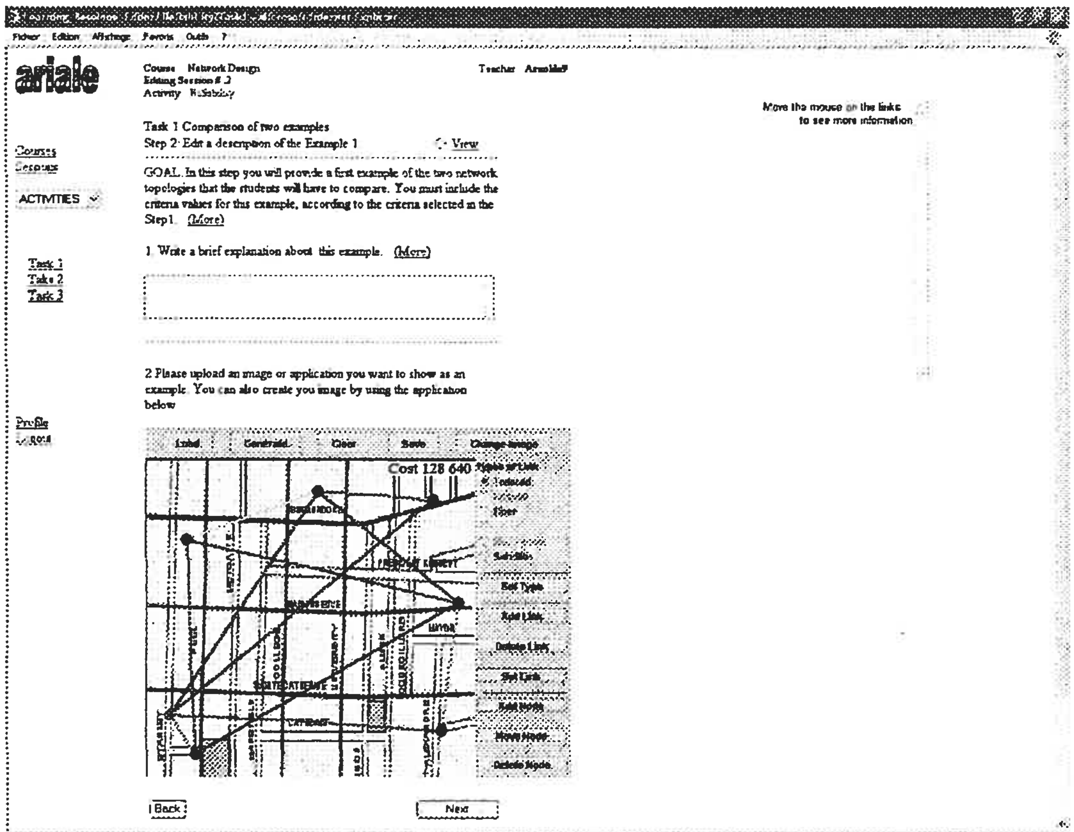


Figure 6.36. The editor of network topologies is a specific resource used in the Network Design course.

The editor is a Java applet embedded in the template of the step of the task that the teacher is authoring. This applet is embedded to avoid problems when interacting with more than one window.

Functionality

In order to support the editing of examples, the example editor has the following general functions:

- acquiring teaching styles (each teacher's expert knowledge)
- displaying examples
- editing examples
- performing the required calculations of network measures of performance.
- displaying different interfaces according to the current step (context).

- recommending shared examples among teachers.
- generating examples

The general editor interface includes buttons that allow the teacher to access different functions. The use of buttons facilitates quick access to the functions instead of navigating a hierarchy of menus. At the top of the editor there are buttons with general functions such as load a topology, generate a new topology, and change the background. On the right there is a panel with the types of links available, buttons that allow the teacher to access functions related to the manipulation of links and nodes. When the teacher starts editing an example, the editor provides the following functions:

a. Load. Loading an example that can be selected from two lists:

- The teacher's old examples list, and
- The other teachers' examples list (examples that are shared by other teachers).

Figure 6.37. illustrates the Load function of the editor.

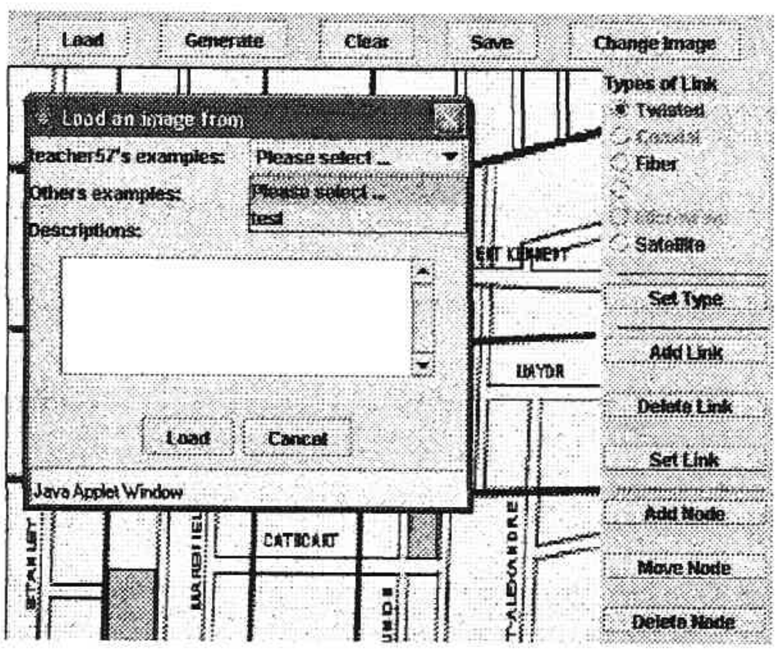


Figure 6.37. The Load function allows teachers to retrieve examples.

The other teachers' examples are a list of recommended examples. If a teacher wants to retrieve examples of other teachers from the case base, he can select one from the menu of other examples of the same class. The system recommends these examples because they are similar to the examples that the current teachers prefers to use (see section 5.2.). Figure 6.38. shows a dialog with some examples recommended as part of the Load function in the example editor.

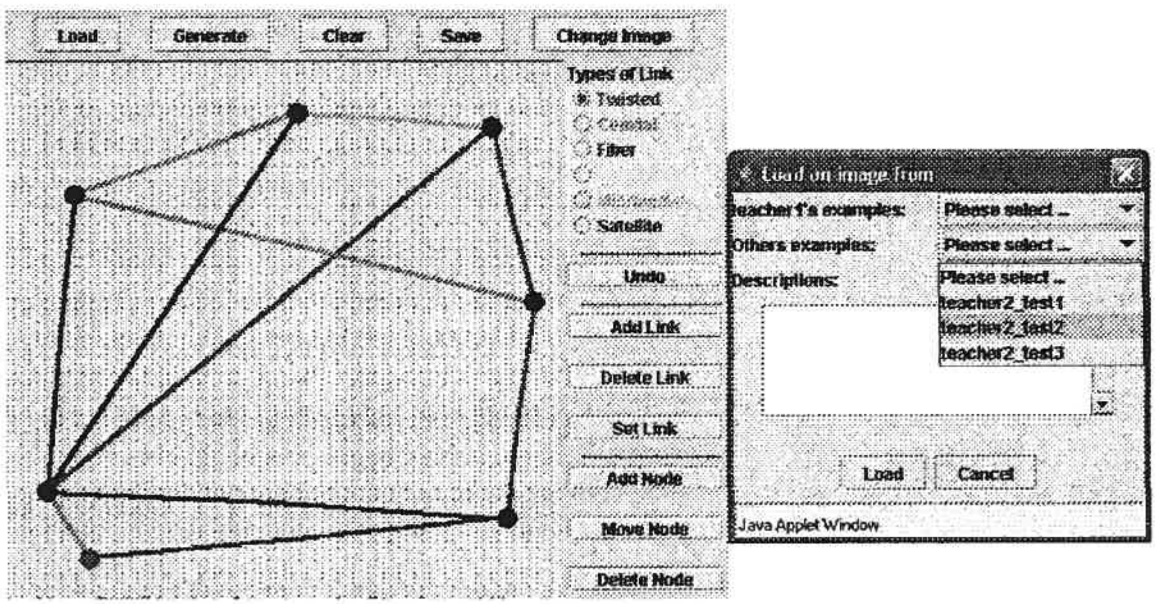


Figure 6.38. Recommended examples in the example editor.

b. Generate. Sometimes a teacher does not want to reuse a session or a particular example, but rather he prefers to receive more recommendations. The editor can supply a new example when the teacher clicks the Generate button at the top of the editor. The *example generator* is detailed in section 5.2. Figure 6.39. shows an example generated automatically.

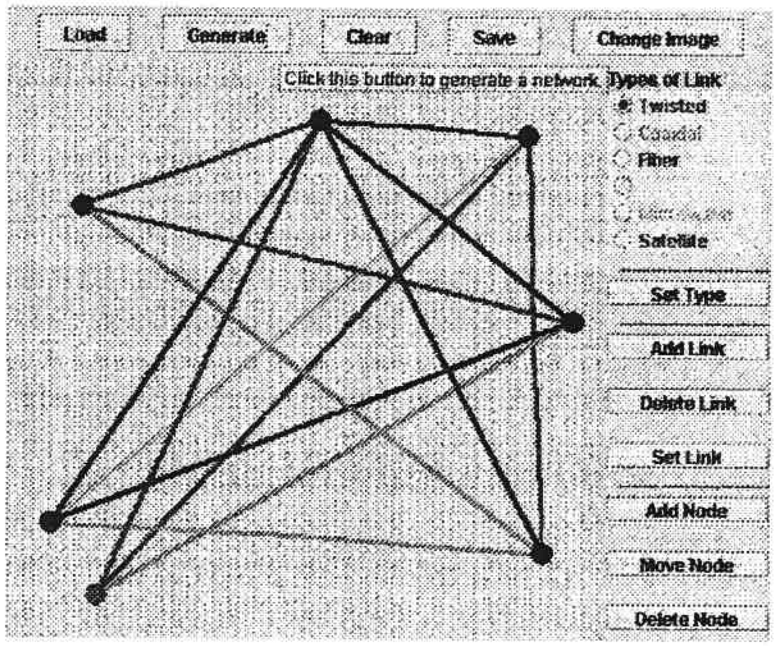


Figure 6.39. The editor generates new topologies on demand.

c. Clear. This function just removes the nodes and links of the current network in order to prepare the editor for the creation of a new topology.

d. Save. This function saves the example in the database of resources of the system. Saving a topology is a complex process that includes different phases.

1. *Storing the configuration of the network as part of the step where the editor is currently displayed.* This process also includes linking the topology to the task, activity and session which the step belongs to. The configuration includes information about the number of nodes, the position of nodes, the node that is the concentrator, the type of link between each pair of nodes, the background of the topology (usually a map of a city), the measures of performance of the network (reliability, cost), a text-based description of the topology, and the class in which the topology is classified.
2. *Classifying the current topology in one of the classes of network topologies.* This classification process is the basis for the case-based reasoning and learning technique used by this system. The Bayesian classification learning used in this system is explained in section 5.2.

The save function displays a dialogue where the teacher specifies the name of the topology to be saved. The teacher can also include a description of the topology and specify if he wants the editor to display the measures of performance of the network and the attributes of the links. Another option allows the teacher to define whether or not if he wants to share this example with other teachers. Figure 6.40. shows the Save dialogue box.

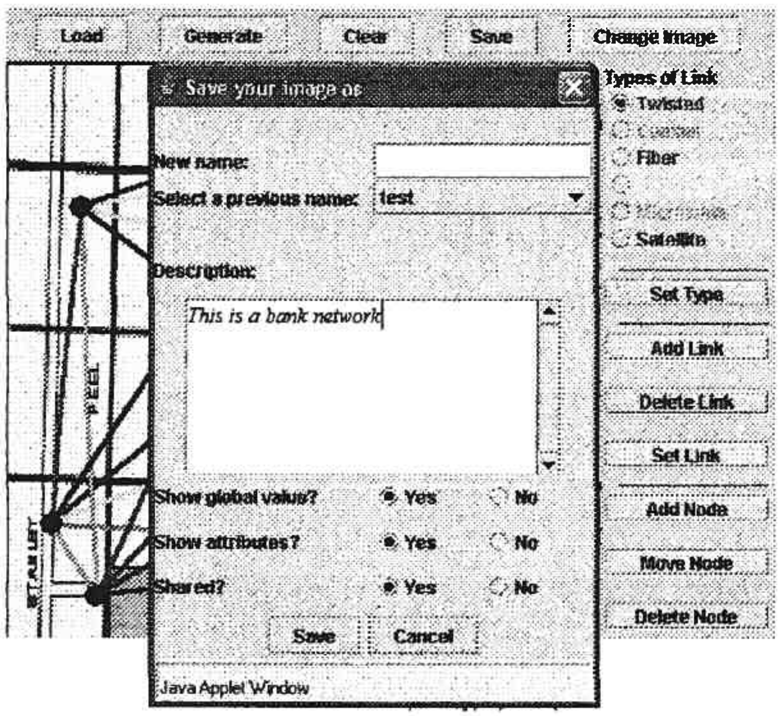


Figure 6.40. The system saves each topology as part of a session.

d. Change image. This function changes the background of the topology. A background can be a map of the city where the network topology is supposed to work, for example.

e. Link types. There are six link types that the teacher can use to connect nodes. Each type is a medium of transmission (twisted, coaxial, fiber, infrared, microwave, satellite) identified by a different color.

f. Set type. This function allows the teacher to redefine the characteristics of a specific type of link according to his goals. By default, the designer of the curriculum defines a series of global values for the different attributes of a type of link. Using the set type function, the teacher can change the default value of the cost of a twisted connection, for example. The teacher can also add or remove attributes to a link type, and he can decide whether or not to show the link type attributes. The process to set a link type includes two sub-functions:

- To set up the nominal values of the attributes for a certain link type.
- To add an attribute to a link type.

The first sub-function is triggered when the teacher clicks the "Add Link" button, and it displays a window to select the new values for the default attributes of this example. In the case of the second sub-function, when the teacher clicks the button "Add/Remove", another window replaces the previous one and shows a form to specify a new attribute to add and its corresponding nominal values. To remove an existing customized attribute, the teacher has to write its name to delete it. If the user does not want to add or remove more attributes, he clicks on the Back button and the application closes the current window, and then opens the Set Link window including the new attributes. In this way, the user can see the attributes that were added. In addition, this dialogue window allows the teacher to specify whether or not he wants to display the values of each type of link in order to show them to or hide them from the students. Figure 6.41. illustrates the dialogue used to reset link type values, and Figure 6.42. shows the dialogue box to specify new attributes.

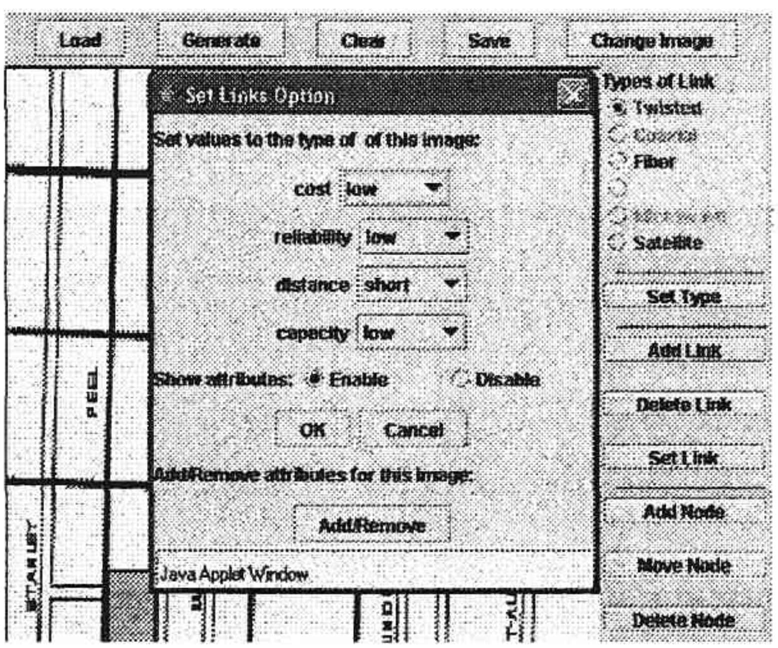


Figure 6.41. Set link options dialogue is an editor feature to change the values of the attribute of the link types.

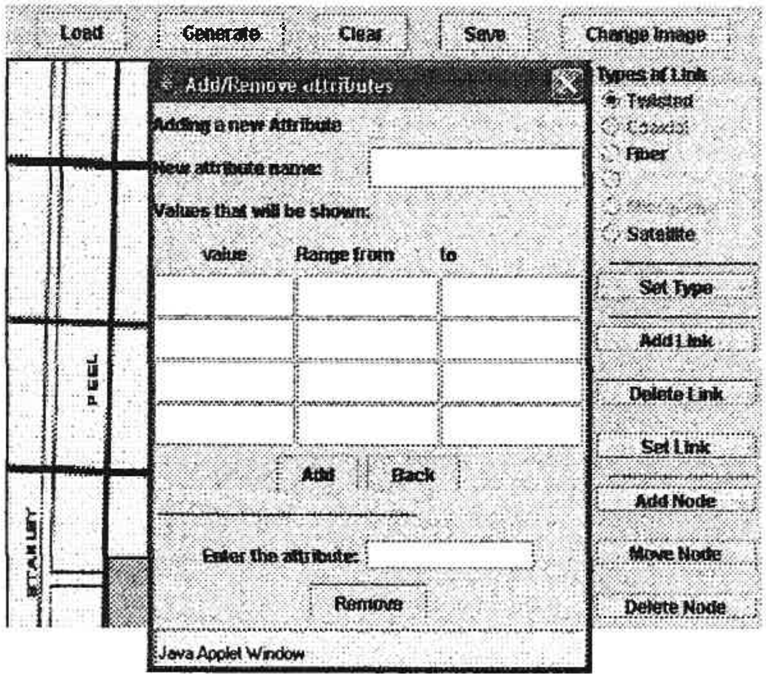


Figure 6.42. The teacher can add or remove attributes of link types.

g. Add link. This function allows the teacher to add a link of a particular type between two nodes.

h. Delete link. This function allows the teacher to delete a link between two nodes.

i. Set link. This function is used to vary the attributes of a specific connection between two nodes. Figure 6.43. shows the dialogue box to change the attributes of a certain link.

j. Add node. This function allows the teacher to add one or more nodes to the topology.

k. Move node. This function allows the teacher to relocate a node. The links that connect the node to other nodes will follow the selected node by stretching themselves.

l. Delete node. This function allows the teacher to delete one or more nodes of the topology. By deleting a node, its links to other nodes are also deleted.

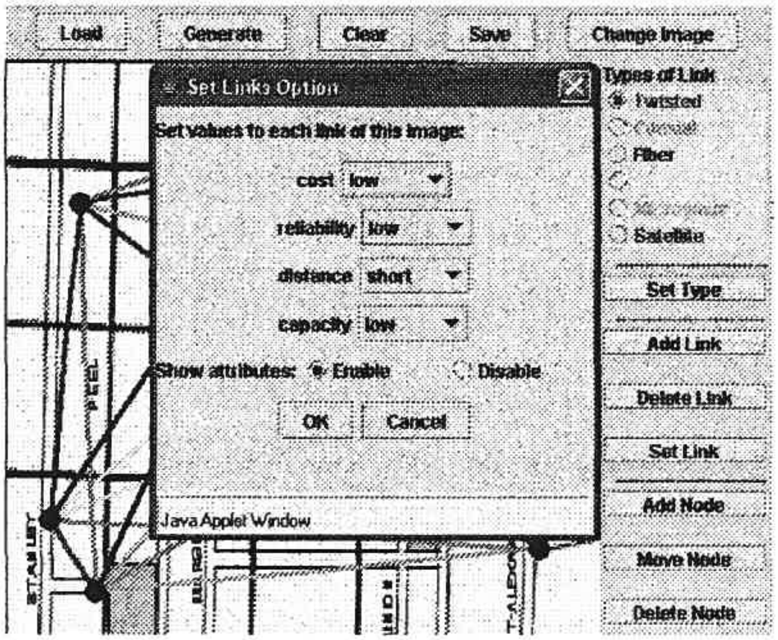


Figure 6.43. The attributes of a particular link can be reset.

6.3.8. Delivering support

As the previous components (menus and authoring tool), the system organizes in three levels the support to be delivered. These levels to access to information in the Web site also are: the general public, the level to select learning sessions and the level for authoring sessions. These levels of access to support are different contexts that determine not only the information to be displayed in the menus and authoring tool, but also in the help area of the interface (the right area of the screen). For example, all the templates of the different steps belong to the level of authoring session, and when the system is creating a Web page it uses a page-builder that integrates a menu, a step, and a help page with information at the same level. For each level, different information will be available according to the teacher's model and the subject to be taught. For example, when the teacher is accepting a learning session that ARIALE recommends to reuse or edit, he does not need specific information about the different steps to be performed later, but would benefit more from general information about activities and tasks.

For our system we have designed different contexts distributed in the three levels of navigation:

- *Level of general public*

- General information about the Web site, the authoring tool and the different subjects available for teaching; this level also includes information about the registration process and the login process.

- *Level of the selection of learning sessions*
  - More specific information. For example, context-sensitive help (learning support) about the subject, activities and tasks, and problem-solving support by means of recommendation of learning sessions.
- *Level of the authoring tool*
  - At this level, the teacher has access to specific information about each field in the step. For each step, the system retrieves all the help content related to a certain level of complexity, but the information is not provided all at once. Instead, the system provides learning support one chunk at a time. Hence, the teacher receives just-enough information and he does not have to scan long lists of links to find possible help. In the case of problem-solving support, the system recommends topologies in the example editor.

6.4. Scenario of interaction

When a registered teacher enters, the system asks him for a login and a password. An invalid password implies the teacher should update some requirement or register again. If the password is valid, the Session Manager retrieves information from the Teacher Model. Based on the Teacher Model, the Planner arranges the sequence of steps to be followed by the user according to the teacher's characteristics and the current step or template to work on. The Helper, which assigns the support for the activities organized by the Planner, generates the help associated with the plan that the teacher will perform and includes this help in the template to be sent to the client side. The Helper selects the strategies, techniques, content and media to guide the user according to information available, and asks for the resources needed to train the *teacher* such as text files, multimedia files, and complementary applications. By selecting and retrieving the resources needed, the Session Manager creates the Web pages where the user will author learning sessions. Once the Planner and the Helper have finished their job (providing a plan, defining resources and recommending sessions or examples), the Session Manager sends the corresponding Web page to the client side. The Interface Agent is embedded in a dynamic Web page that the teacher will see. On the client side, each dynamically created Web page looks like a real Web page that integrates material in a template; however, no page is permanently stored because each one must be generated according to a particular teacher's profile and adapted to a particular interaction context. Nevertheless, the Session Manager stores the structure of each Web page, with details of the location of each field and the resources and events associated with each one. Doing that, the system would be able to re-build the Web page in the future and to generate statistics about the history of the interaction with the user.

The Interface Agent will process each new piece of information entered by the teacher. Depending on the explicit and contextual information, the Interface Agent could ask the Session Manager to coordinate the generation of a new Web page of the authoring tool with the guidance that the teacher needs. However, the Interface Agent not only monitors the user's context and his actions to send the information required to adapt to the teacher's learning and teaching styles, but also asks whether to change some aspects of the user interface if necessary. This agent can provide unsolicited help or guidance according to the events performed by the teacher, the level of complexity of the task and also the preferences of the teacher. After each request of the Interface Agent that monitors the teacher's events, the system generates virtual Web pages on demand. Each page is generated on the fly by combining various media objects from the base of resources according to a template for a Web page. The following algorithm summarizes this general scenario of Teacher-System interaction (Rodríguez *et al.*, 2004a).

```
Procedure General Scenario of Teacher-System Interaction {

        Call procedure Accessing Web site

        IF the teacher is new and accepts to load a recommended session THEN
                Display a recommended session

        IF there is an incomplete last step THEN
                Resume the last step that the teacher was performing
                Call the procedure Build a Web page

        IF the teacher does not have incomplete sessions THEN
                Recommend sessions to be reused
        ELSE let the teacher create a new session.

        Call the procedure Build a Web page

        WHILE the teacher is authoring
                Monitor user's events

                IF a predicted event occurs THEN
                        Identify where the cursor is focusing

                IF the event calls for learning support THEN
                        Offer learning support
                ELSE Offer problem solving support
                Update counters of the interaction

                IF event implies internal changes THEN
                        Update fields with internal data

                IF event implies external resources THEN
                        Update fields with external data
                Call the Procedure Update Teacher Model

                IF the step is finished THEN
                        IF there is a next step THEN
                                Call the Procedure Update Teacher Model
                                Select a new step template
                        ELSE    plan is finished

                Save the learning session
                Call the Procedure Update Teacher Model
                Update the system knowledge
                Update the Web page
        END of WHILE
    END
}

    Procedure  Accessing Web site {

        Call the Procedure Setup Screen
        Process an IP request from a teacher
        IF IP address is known THEN
                Display the Login Web-page
                IF the Login data is correct THE
                        Call the  Procedure Build a Web page
                        to generate a personalized teacher's Web-page
                ELSE  Display the Login Web-page
        ELSE   Display  the main  Home Page of the Web-site
    }
```

```
Procedure Setup Screen {
        Detect client-side characteristics
        Set up client-side characteristics in the Teacher Model
}
Procedure Build a Web page{
        Call the Procedure Update Teacher Model
        Call the Procedure Retrieve Teacher Model
        Call the Procedure Include the Menu according to the teacher's Level and  Language
        Call the Procedure Include a Template for the respective step
        Call the Procedure  Include Help according to the teacher's Plan
        Retrieve resources  according to the teacher's Plan, Level and Language
        Call the Procedure Send a page
}
Procedure Retrieve Teacher Model {
        Retrieve the teacher's profile
        Retrieve the teacher's overlay model
        Call the Procedure Retrieve a Plan  // this procedure belongs to the Planner


}
        Procedure Send a page {
        Send a Web page to Client side
        Establish  communication client-server
        Start  interface agent applet
        Start monitoring  the interaction teacher-system
}
```

A more specific scenario related to a particular example is as follows. First, we imagine the case of a novice teacher who has to start using the site to create a learning session on Network Design. Specifically, that teacher must begin with the analysis task of the activity "Reliability" to offer a series of network configurations the students must compare. The Planner uses the information from the Teacher Model, the Pedagogical Model, and the Curriculum to define the menu and the authoring tool contents. The Planner checks the Teacher Model, the Curriculum to teach and the Session-Base to know to which step of which learning session the teacher goes. After that, the agent defines which resources are required and passes the information to the Session Manager. The Helper deals with the information from the Teacher Model, the Pedagogical Model and the Session-Base to define the support that must be provided to the teacher. This Planner and the Helper deliver their definitions of contents to the Session Manager; then, the Session Manager integrates both parts and an instance of the Interface Agent in a Web page to send to the client-side. This Web page includes a navigation menu on the left side, a form for authoring the task, and, the learning support will be displayed on the right side.

In the client side, the teacher receives a Web page that is a template of a set of actions to generate an assignment on which students are required to write an essay comparing different network configurations. Our beginner teacher has entered the environment, filled in the registration forms, but he does not know what to do first. The system has stored the information that he has never used the Web site, he prefers to receive text based support, and he learns slowly. When he had to choose between creating a new session or reusing one from the session-base, the Interface Agent advised him to use a session from the session-base because following an example would be better for him. The author can edit more aspects, change a picture of the network topology and so on and the system will record his job.

6.5. Low level interaction

The method designed for the communication between the teacher and ARIALE allows the teacher to interact by performing events over different Web page components (widgets). Each field in a Web page has particular components. For example, in one field it is possible to find a text link, in another field there could be a text box or a radio button. Our system shows almost the same type of components in each Web page although the contents are different. Independently of the course, session, activity or task, the teacher is going to see Web pages with a relatively standard layout and internal structure. Some sections and widgets are frequently in the same location and, when particular events occur to a certain widget, the teacher sees messages with similar help techniques. This design criterion makes the interface more predictable, and, at the same time, content and help presentation is adapted to the context (plan, user's skills, etc.), as described in Figure 5.7.

The widgets or components can react to determined events (mouse movements and keystrokes) to display particular messages, and these messages use resources that are associated with help techniques. The techniques and events are also associated with fields in a Web page. Furthermore, each widget is an access point to help potentially. The main idea behind this help system is to support or guide the teacher instead of teaching him, for this reason the use of widgets and events allows the teacher to retrieve information about the interface components he is using to perform a task (Encarnação, 1997). Then, a particular help technique is called up when an event occurs when some rules included in the Web page are triggered depending on the event performed. This set of rules says which events trigger determined help techniques depending on the teacher's skills, plan, and preferences. For example, if the teacher's skill is classified as a beginner, he is authoring a task of analysis of cases, and the help strategy is Basic, then a "MORE" reacts to the mouse click displaying an introduction.

Widgets, events and techniques

In our system, we designed active and passive help. The passive help is displayed when the teacher triggers an event with the intention of getting help. For example, the teacher clicks a "MORE" link to see information related to specific fields of the Goal and Edition sections, or the teacher clicks the button "GENERATE" to ask for a recommended network topology. Active help is provided when the system or the teacher trigger events as a result of the interaction teacher-system. For example, the system recommends a worked example to edit when a novice teacher starts using the Web site (see Figure 6.44).

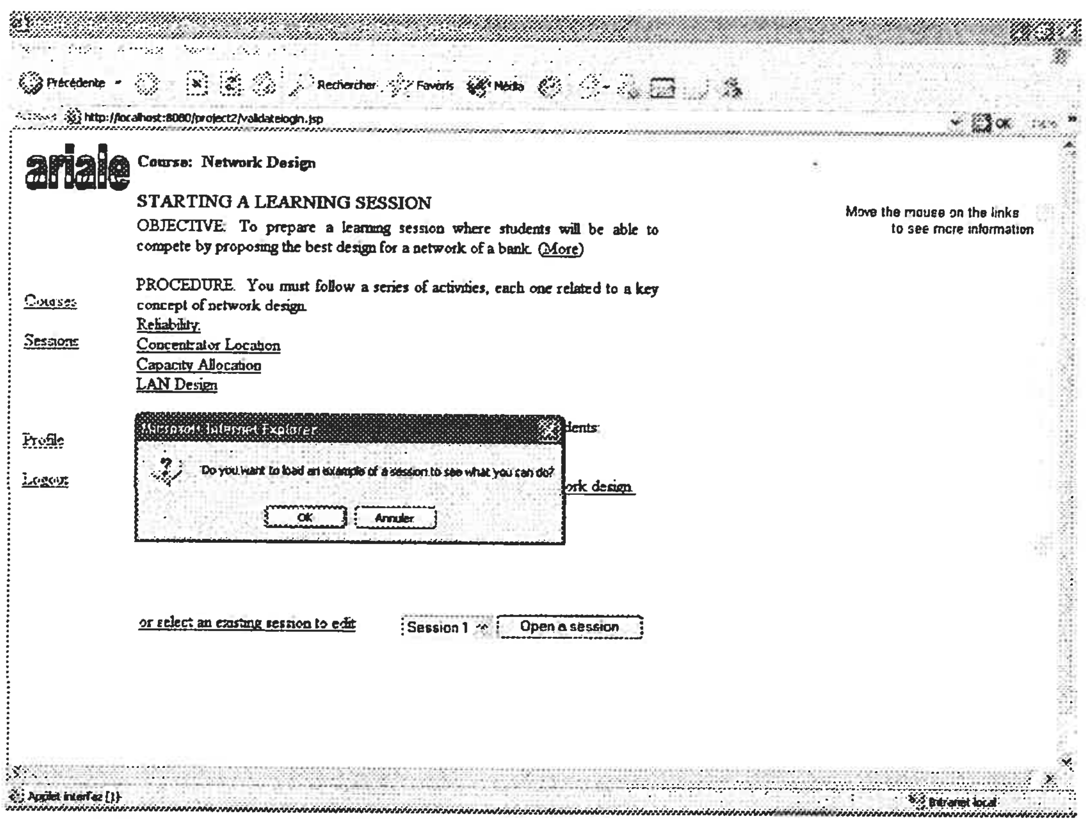


Figure 6.44. A message to recommend a session.

Because many users know that on MouseMove or OnMouseOver events can generate help messages, this system profits from this existing rule of interaction to provide active and passive help. For example, once the Web page is sent to the client, if a teacher moves the mouse over a radio button, he can see a text-based definition related to the concept associated with this radio button. The system offers this definition, and the teacher does not have to find this help among a list of options that includes all the information available for the complete Web site. The text that the teacher can read is also a resource adapted to the teacher situation, after passing the filtering process explained previously. In this way, the system keeps the teacher in control of the flow of help provided, and at the same time, it manages the complexity of selecting appropriate help.

In another case, if a teacher starts writing (onFocus event) in a text area, then he receives a hint. If the teacher rejects this type of hints three times, the system adapts and disables this rule. The number of rejections to disable this rule has chosen arbitrary. To enable the rule back, the teacher must edit his preferences manually, by accessing the profile manager in the "Profile" hyperlink. This and the previous examples illustrate the use of active help. The passive help is used every time the teacher clicks the link "MORE" to get additional help.

Table 6.5. depicts the relationships existing among components, events and help techniques when the help strategy is "Basic help" (see section 4.12.). This table shows the respective help technique in the cell where the component row and the event column intersect. Each pair of event and component is associated with a help technique to generate a meaning for the teacher. This meaning facilitates the further identification and use of each type of relationship to get help (Rodríguez *et al.*, 2004b).

Table 6.5. Relationships between help techniques, events and components for Basic Help.

COMPONENTS	EVENTS				
	OnLoad	**OnBeforeUnload**	**OnClick**	**OnMouseMove**	**OnFocus**
Web page	HINT INSTANCE	HINT			
Text link			INTRO- DUCTION		
Button				DEFINITION	
Radio button				DEFINITION	
Check button				DEFINITION	
Text area					HINT

In the cases of intermediate and advanced teachers, the structure of components and events remains the same, but some help techniques change. The system will not have to prompt recommending worked-out examples to intermediate and advanced teachers when they enter the session selector Web page. Instead, they will see their own examples to reuse, and the system will recommend passively other teachers' sessions. Figure 6.45. gives a picture of the session selector offering other teachers' sessions after the sessions authored by the teacher. Intermediate and advanced teachers will know that other teachers' sessions are available. This is the reason why the system does not use the Instance help technique in these two levels of a teacher's skills or experience.

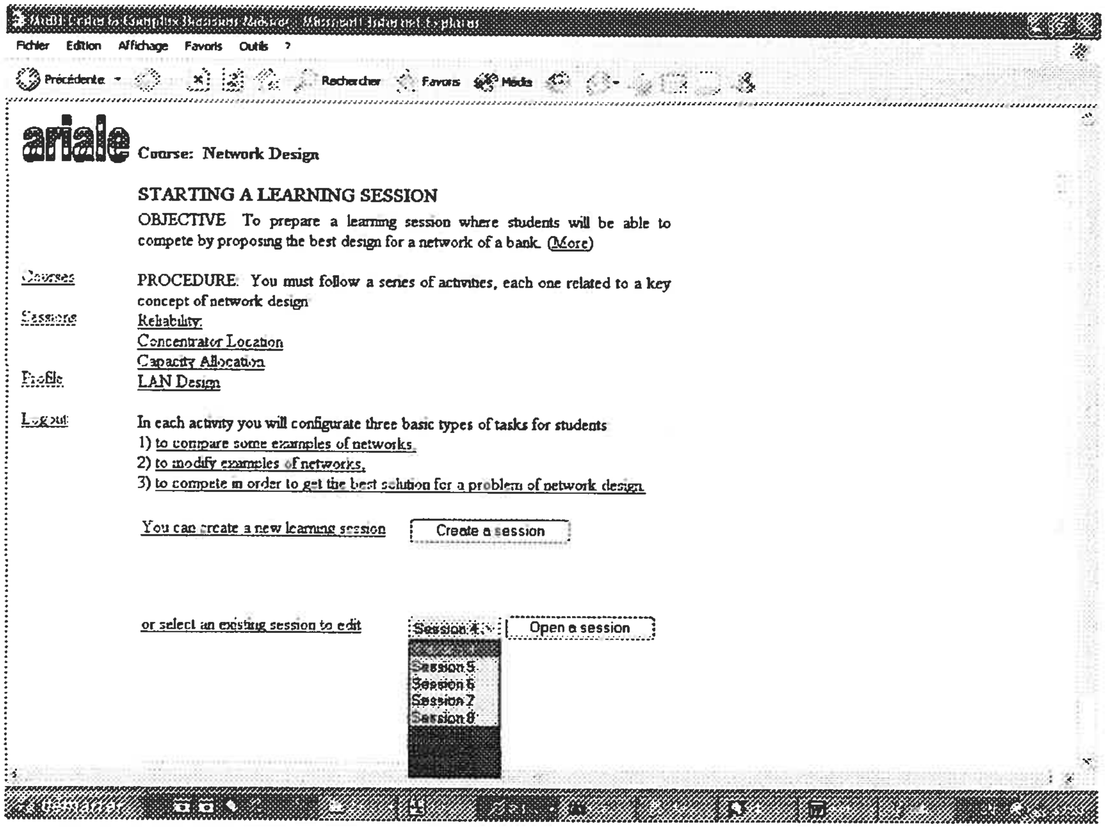


Figure 6.45. The session selector sorts the other teachers' sessions after the sessions authored by the teacher.

The other important change in the case of intermediate and advanced teachers occurs when they click on the "MORE" text link, because different techniques are displayed. In the case of intermediate teachers, the techniques for these events and components are Demonstration, Guide and Tutorial (Guidance strategy), but Demonstration can not be displayed when the teacher prefers only text because Demonstrations are based on animated images. For this type of teachers, the system provides text-based guides and short tutorials when the teacher clicks on the "MORE" link. The same occurs in the case of the advanced level because if the teacher prefers text he will receive text-based examples, analogies and quick references.

Table 6.6.. illustrates the structure for the Guidance help strategy.

Table 6.6. Relationships between help techniques, events and components for Guidance.

Components	Events			
	OnBeforeUnload	OnClick	OnMouseMove	OnFocus
Web page	HINT			
Text link		DEMONSTRATION GUIDE TUTORIAL		
Button			DEFINITION	
Radio button			DEFINITION	
Check button			DEFINITION	
Textarea				HINT

Some examples of help provided depending on the events are:

-A *hint* that offers an instance prompts when:

- The Session Selector Web page is loaded.
- The teacher focuses on a text area to write an instruction after rejecting the previous hint.

-A *definition* is displayed when:

- The teacher moves the mouse on a button.

-An *introduction* is displayed when:

- The teacher clicks a "MORE" link while the Basic help strategy is activated.

-A *demonstration* is displayed when:

- The teacher clicks a "MORE" link while the Guidance strategy is activated, and the medium preferred by the teacher is image and text or multimedia.

-A *guide* is displayed when:

- The teacher clicks a "MORE" link while the Guidance strategy is activated.

-A *tutorial* is displayed when:

- The teacher clicks a "MORE" link while the Guidance strategy is activated, and he has seen the demonstrations or guides available.

-A *quick reference* is displayed when:

- The teacher clicks a "MORE" link while the Assistance strategy is activated.

-An *example* is displayed when:

- The teacher clicks a "MORE" link while the Assistance strategy is activated and he has seen the quick reference available.

-An *analogy* is displayed when:

- The teacher clicks a "MORE" link while the Assistance strategy is activated and he has seen the examples available.

In general, the onLoad event is associated with hints that offer instances (sessions), the onMousemove event displays definitions and clicking the "MORE" links offers more adaptive and context-sensitive help. For each link in the menus only definitions are displayed directly on the menu area. Figure 6.46. illustrates one case of help for menus.

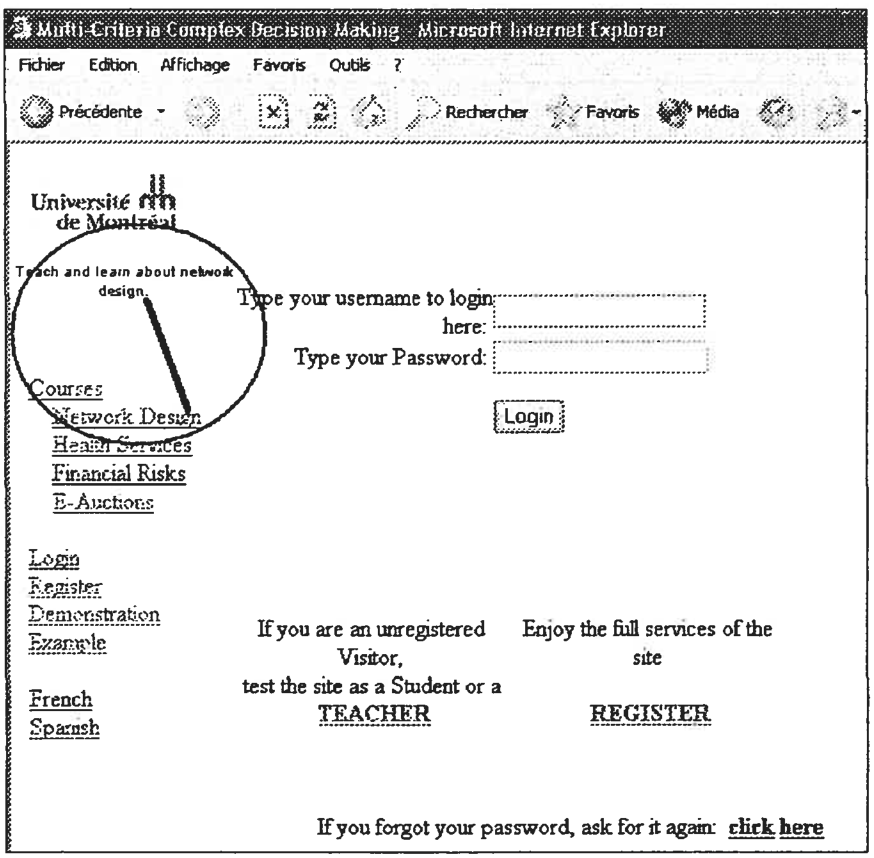


Figure 6.46. Context-sensitive help related to menu links is shown in the left area of the screen.

A set of techniques, events, and widgets

ARIALE uses the help techniques, events, and widgets described in Table 6.6. because they are simple and many of them are well-known by people. For example, definitions displayed as context-sensitive help, pop-up messages with hints, demonstrations and guides, are familiar to many users of computers. The events that we selected are very common for Internet users, for example, and the teacher only needs basic knowledge or skills using computers to use the Web site. In the case of events, the system does not have to ask the teacher to click or double click the right button of the mouse and a context menu many times. When he performs a simple event he can receive help. An advantage of linking help to widgets is the teacher receives information directly related to a particular component in the step. Each widget becomes an access to help. Some widgets have acquired a particular meaning that is shared by many users. For example, a text link is supposed to send the user to another Web page or trigger an additional action such as prompting a dialogue box; buttons are usually related to actions and procedures.

Figure 6.46. shows the structure of a template with the different fields (widgets) and the help techniques that depend on the mouse events to be displayed. The techniques are mapped to specific resources that have the complexity and other attributes required.

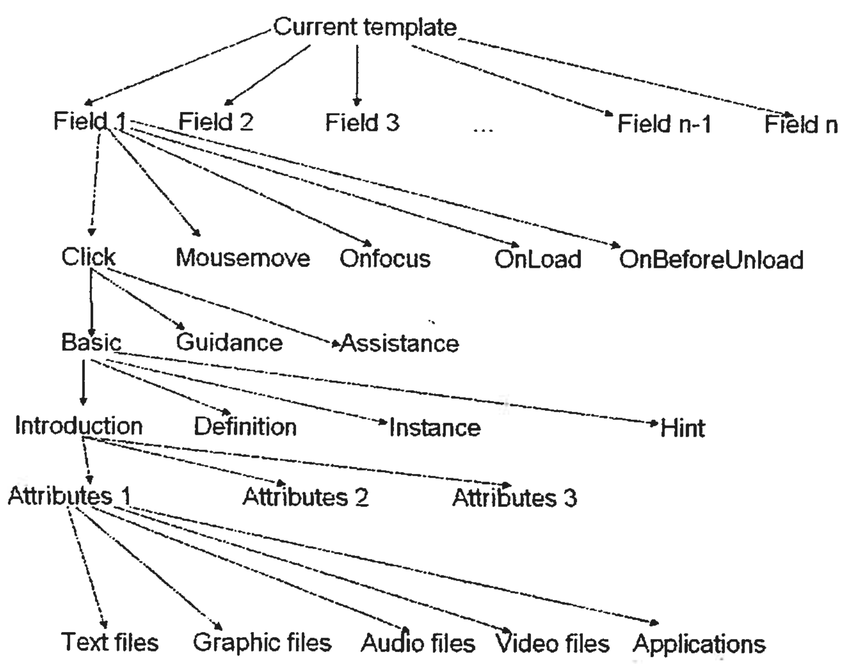


Figure 6.47. Process for displaying resources according to help techniques, fields and events.

Another aspect related to the interaction of teachers with the help system is the display mechanisms used to supply help. One of the most important criticisms directed at *active help* is that this type of help disrupts, generates cognitive overload, and increases problems using an application or a Web site. To reduce this difficulty, we decide to avoid or reduce to a minimum the use of balloons, tool-tips (layers) and pop-up windows. Additionally, a user needs particular help, not a Web page full of all the help available for the site. Loading a page with more help topics than necessary is not efficient if only a part is required. While a specific line of help text on a long Web page could be highlighted to remark its relationship with a particular topic, the rest of the help is still not necessary. This is a typical problem of the FAQ-based help. Why does the teacher have to download 100 topics to see only the help that is relevant for his current task? To avoid this, we have applied a solution that takes advantage of the knowledge-based human-computer interaction (Fischer, 2001) used by our system. The solution is to provide only the help required by a teacher when he is loading a new Web page (Self, 1998), and also predict only the help that he may need if the help previously loaded becomes insufficient. Another important aspect of this system is that the help contents are displayed embedded in the Web-page window.

Embedded help

As described in section 6.3.2., there is an area in the browser window that displays help. Our system displays the help content on the right side of the screen and leaves it there until new information takes its place. Thus, the help appears within the same window, rather than in a separate help window. Furthermore, when the teacher needs help, he does not have to leave the browser window where he is working. This decision was made because of the switching problem that a person faces when he is going to the help and coming back to the working area many times until he solves his problems and finishes his current task (see also sections 2.1.3. and 5.3.5.). Thus, it is possible to deduce that he will want to have the help available many times to consult it until the problem is solved. In addition, the Teacher Model keeps a record of the last help content displayed for each field in each accessed Web-page. This history is a key data to resume the task with the same help contents if the teacher logs out and come back later.

Interaction history

To support its decisions about the help to provide, the system keeps a history of the interaction of the teacher with the authoring tool and the help system. As explained in chapter four, the Teacher Model includes this history with the following data for each step and field:

- Teacher's experience. Our system keeps records of the number of times that the teacher has done or completed each step. For example, if a teacher has authored two complete learning sessions and he is currently performing step 2 of task1, ARIALE knows that the teacher has authored the step 1 three times, because step 1 is a requisite of step 2.
- Teacher's performance. ARIALE also stores the number of times the teacher has accessed each type of help message in each step. For example, if a teacher clicks to see help related to a particular field in step 2 of task 1, then the counter of this help increases and its value is saved to the Overlay Model.

Every time a teacher finishes a step, the records of his experience and performance are updated in the database. This process is part of the method to update the Teacher Model, already explained in sections 4.2.2. and 4.2.3. The system uses the information included in the updated Teacher Model to infer and predict which help the teacher needs when he starts authoring a new step.

6.6. Conclusion

This chapter detailed the implementation of our system ARIALE (Authoring Resources for Implementing Adaptive Learning Environments). We discussed the general structure of the Web site, the three-tier architecture of the system, its knowledge-base and the modules (Session Manager, Planner and Helper) that process that knowledge to make decisions such as building a Web page, recommending examples or providing other types of help adapted to the implementer teacher. These sections in chapter six explain the layout of different Web pages, the general functionality and the main processes that the system performs.

This chapter also detailed *the method used for building our assistant*, principally the aspects related to the design of the Web-site, the authoring tool, with emphasis on the editor of network topologies, and the intelligent help. Finally, a walk-through shows a general scenario of interaction between a teacher and our system and a specific scenario of interaction that shows an example of the back-end behaviour of the system. We included an explanation of the low level interaction of the teacher with widgets, focusing on the events that trigger problem-solving support and learning support.

Chapter six is a portrait of the complexity of building an educational adaptive hypermedia. The importance of this portrait is that it shows the integration of many methods and procedures that are usually studied unconnectedly. This integration results in a methodology to build an intelligent help system and its context in two steps: 1-the construction of the Web-site with the main application or functionality (the authoring tools), and 2- the intelligent help. This integration is needed because the authoring tool and the help must be coordinated.

In addition, chapter six shows the relationships between the different parts of the interface layout (menu, authoring tool and help) and the levels of access of different users (general public, selection of sessions and edition of network topologies). The description of the implementation details related to the interaction between the teacher and the system, in particular the referred to low level, help us to realize the existence of a general but complex structure behind hypermedia applications and their users. The concept parallel hypermedia shows that the navigation through different virtual spaces can be simultaneous. For example, a knowledge worker in a help desk or call center can navigate multiple hypermedia t find the solution for the problem of a user asking for support. Just like writing, painting or making a movie, the creation of multimedia applications or Web sites involves tools, media, and basic rules to structure messages to facilitate the understanding, acceptance and use by the people who consume these messages. The organization of events, widgets, and help contents compounds a particular syntax to provide the meaning of a concept or components of a Web page according to the current teacher's context. This method of interaction explained in this section 6.4. becomes a grammar that can be reused by help systems designers.

In the next chapter there is an evaluation of the most important aspects of this investigation, followed by our conclusions and suggestions for future research related to this subject.

Chapter 7. Validation, discussion and general conclusion

Summary: In this chapter we show the results of the evaluation of some functions of ARIALE, we discuss limitations and strengths of our research, and, in the conclusion, we emphasize the contribution and innovations of our research and proposes areas for future research.

The evaluation of our work focuses on measuring the *accuracy* of ARIALE in recommending examples of network topologies (problem-solving support). We also validate the acceptance of the ARIALE help approach by teachers.

7.1. Evaluation

The objectives of our evaluation are to measure

- The accuracy of ARIALE in recommending examples that match the teacher's interest.
- The acceptance of the ARIALE help approach by teachers.

Measuring the accuracy of our system is fundamental because the problem-solving support provided by ARIALE is its main function, and it relies on the ability of the system to propose the most appropriate examples rather than overwhelming the teacher with an overflow of options.

Additionally, recommendation of examples of network topologies is part of the help approach that ARIALE provides. The trend in training and general e-learning is that trainers or learners must do a pre-tests and post-tests to evaluate learning. For learning support in our case, we prefer to use the *Electronic Performance Support Systems* (EPSS) approach (Leung, 2001) with knowledge-based personalized help, but without pre-test or post-tests. An evaluation of the subjective perception and the satisfaction that teachers express about ARIALE approach allow us to get a measure of the acceptance of this approach by teachers engaged in authoring teaching materials.

7.2. Evaluation of the accuracy of the recommendations

The evaluation of the accuracy of the recommendations involves a comparison of our Bayesian classifier based on *measures of performance* of topologies, and a k-NN (k=1 Nearest Neighbour) method. The degree of difficulty of each method and its performance are analyzed; the results of the evaluation justify the use of our Bayesian classifier based on measures of performance of topologies.

To evaluate our system, we applied the accuracy criterion to check if the system recommends good examples and counter-examples. As explained in sections 5.2.1. and 5.2.4., an *example* is a network topology with different number and types of links, no more than seven nodes, no more than 20 links and three basic measures of performance (reliability, capacity and cost). A *counter-example* is another topology with measures of performance equivalent to a given example, but with a different number of links than that of the given example. According to Bengio and Grandvalet (2003), a standard measure of accuracy is the *prediction of error*. Given a number of instances in a data set, "the accuracy of a classifier is measured by performing a number of classifications and dividing the number of correctly classified instances by the total number of instances" (Lavesso, 2003). In our validation, we not only tested the accuracy of our Bayesian classifier (see sections 2.2.7 and 5.2.6.) but also compared it to another Bayesian classifier based on the types of links used in topologies, and to a k-NN (k=1 Nearest Neighbour) method. The accuracy of our recommendation method was validated by the following three ways:

- 10-fold *cross-validation* of each of the three classifiers.
- Comparison of the performance of the Bayesian classifier based on measures of performance with the other classifier.
- Analysis of recommendations.

7.2.1 Cross-validation

Cross-validation is a procedure that consists of dividing the training dataset data into a number *k* of blocks or partitions. Testing is performed by extracting one partition from the dataset; each example of the extracted block is classified to measure the error rate of the classifier. After finishing with a partition, it is reinserted into the dataset and another block of examples is tested. These operations are repeated for each partition, and finally, an average of the number of errors is calculated (Stone, 1977). The inverse of this average shows the degree of accuracy of the classifier.

We ran 10-fold cross-validation to validate our classifiers. This means that we took a training dataset with 200 topologies, which were classified manually into the six classes explained in section 5.2.6.1. The set was divided in ten partitions of 20 topologies; each partition and each topology of each block was tested, using the remaining 180 topologies as the training dataset. We tested the Bayesian classifier and the *k*-NN classifier, and the results are as follow.

Results of the Bayesian classifier based on the measures of performance of topologies

According to a 10-fold cross-validation, the accuracy of our Bayesian classifier based on the measures of performance is 94%.

After 10 tests, the number of errors was 12. This number was divided by the number of test (10) to achieve the average. The *average error* was 1.2 for the 10-fold cross-validation. This means that this classifier has 6% error according to the test developed with the training dataset of 200 topologies (20 topologies for testing and 180 for training for each test). This classifier works with six classes of topologies (see section 5.2.6.1.) based on the following measures of performance: reliability, capacity, and cost. The number of links in each topology is also taken into account. While the general functionally of our classifier is described in section 5.2.6., some examples of classified topologies are discussed in this section.

For example, topologies A and B (see Figure 7.1.) are part of the set of 200 topologies that we used for testing. They have equivalent measures of performance so that the classifier classifies them into class one. They are high reliability networks with medium capacity and low cost; B has more capacity and is more expensive but both topologies are in the same range of values for capacity and cost. According to Table 5.6., a capacity value is "medium" if it is greater than 10001 Kbps and smaller than 30000 Kbps; a cost is low if it is smaller than 100001 units (section 5.2 details the structure of a topology and the calculation of the measures of performance). Topologies A and B are examples of good classification because the classifier found they belong to the same class. Their measures of performance are typical of networks belonging to class one, according to the definition of classes in section 5.2.6.1. As a consequence of this efficient classification, it is possible to affirm that given A, the topology B is an equivalent example. If a teacher prefers to use examples similar to A, ARIALE recommends learning sessions that contain topologies belonging to class one. The system can also recommend a list of examples of topologies that belong to class one. In the case of an automatically generated topology, the classification allows the system to test and filter the topology in order to reject or recommend it.

Table 7.1. summarizes the measures of performance of each topology.

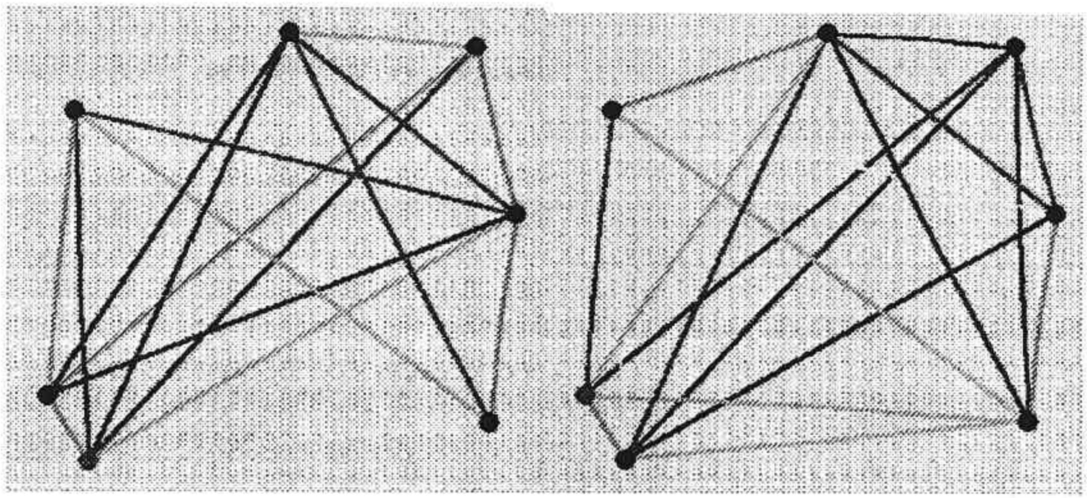
Table 7.1. Measures of performance of topologies A and B

Measures	*Topologies*	
	A	B
Reliability	5	5
Capacity	10472 (Medium)	18378 (Medium)
Cost	44950 (Low)	90300 (Low)
Number of links	19	19

Each type of link can be identified by a specific color:

Types of link	***Color***
Twisted	red
Coaxial	green
Fiber	blue
Infrared	yellow
Microwave	cyan
Satellite	brown

The attributes of each type of link are detailed in section 5.2.2.



A B

Figure 7.1. Configurations of topologies A and B.

Another example of the accuracy of this classifier is the pair of networks B and C, which also belong to the set of 200 topologies in the training dataset. While B and C are classified in class one with similar capacity and cost, they have a different number of links (19 and 15). In this case, B is an example and C is a counter-example, and both of them are correctly classified in class one. The number of links is a criterion that ARIALE applies to discriminate examples from counter-examples in step 2 of Task 1 in any activity, according to the Pedagogical Model explained in chapter four. Table 7.2. summarizes the measures of performance of each topology.

Table 7.2. Measures of performance of topologies B and C

Measures	*Topologies*	
	B	C
Reliability	5	4
Capacity	18378 (Medium)	16348 (Medium)
Cost	90300 (Low)	92200 (Low)
Number of links	19	15

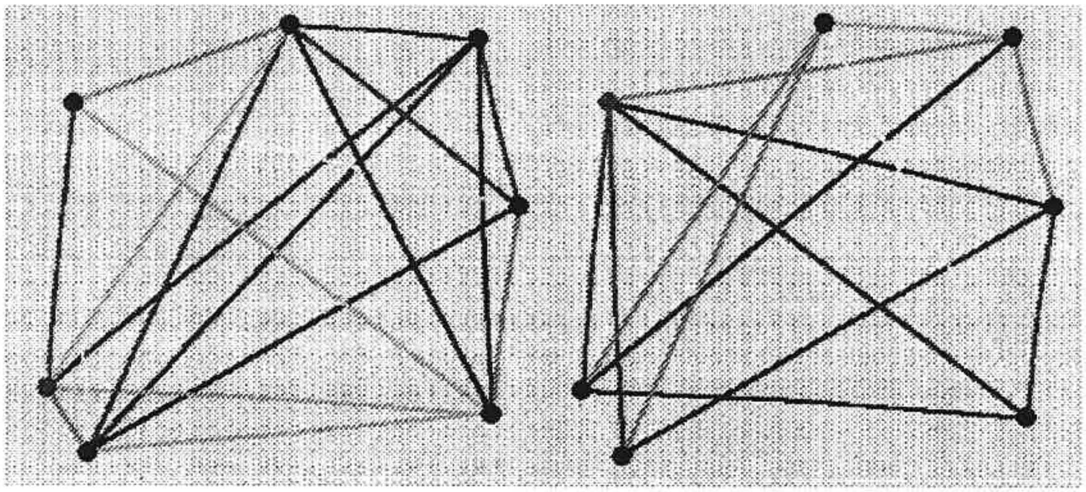


Figure 7.2. Configurations of topologies B and C (an example and its counter-example).

Results for the k-NN classifier

According to a 10-fold cross-validation, the *accuracy* of a k-NN (k=1 Nearest Neighbour) classifier based on the measures of performance is 96% for the classification of network topologies in ARIALE.

After 10 tests, the number of errors accumulated was 8, and we then divided them between the number of tests (10) to achieve the average. The *average error* was 0.8 for the 10-fold cross-validation. This means that this classifier runs with 4% error according to the test developed with the training dataset of 200 examples (20 topologies for testing and 180 for training for each test). This classifier also works with our six classes of topologies based on the same measures of performance: reliability, capacity, and cost. The number of links in each topology is also taken into account. This k-NN classifier measures the dissimilarity between the measures of performance of a given topology and the measures of performance of each network in a set of topologies. ARIALE calculates the dissimilarity between each pair of topologies by using the following equation:

$$Dissimilarity(a_1,a_2)=C_1\sqrt{(G_1-G_2)^2}+C_2\sqrt{(R_1-R_2)^2}+C_3\sqrt{(T_1-T_2)^2}+C_4\sqrt{(L_1-L_2)^2} \quad (20)$$

where:

- a_1 and a_2 are the topologies whose dissimilarity is to be calculated.

- G is global cost.
- R is average reliability.
- T is the total capacity.
- L is the number of links.
- C_1, C_2, C_3, and C_4 are weights to tune the importance of each of the four criteria in the dissimilarity calculation.

As the dissimilarity measure increases, the two topologies become more different. A dissimilarity that is close to zero means that the two topologies are very similar and that they can be classified in the same class. Figure 7.3. shows two topologies B and D that the k-NN classifier found very similar. The dissimilarity of these topologies is the smallest dissimilarity that the classifier found after the comparison between topology B and each of the other 180 topologies in the training dataset, during one of the 10 tests. According to this classification, B belongs to class one, which is also the class of D. Table 7.3. shows the values for the measures of performance for topologies B and D.

Table 7.3. Measures of performance of topologies B and D

Measures	*Topologies*	
	B	D
Reliability	5	5
Capacity	18378 (Medium)	18562 (Medium)
Cost	90300 (Low)	99350 (Low)
Number of links	19	20
Class	1	1

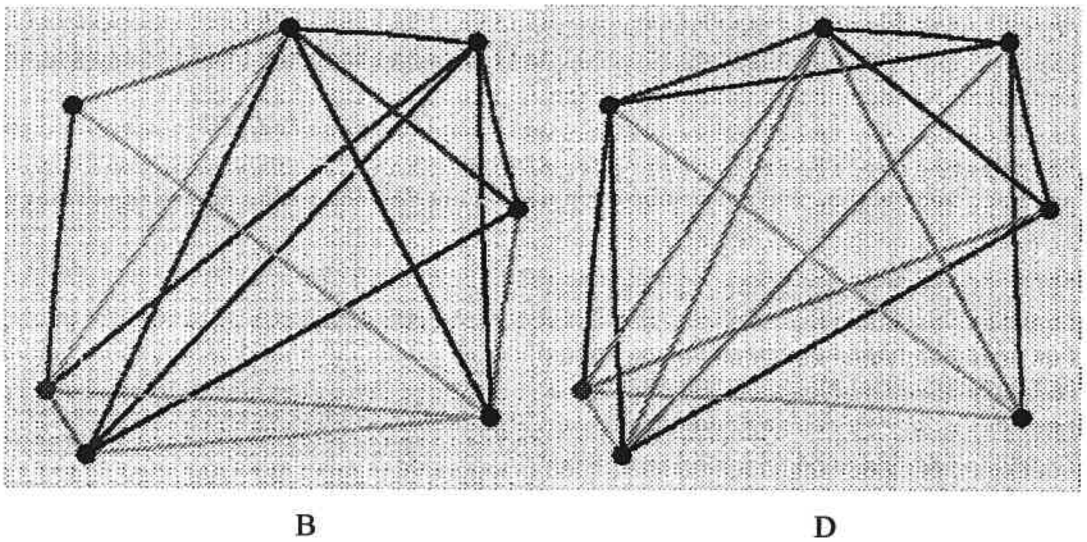


Figure 7. 3. Topologies B and D are very similar according to the k-NN classifier.

The k-NN classifier could also be useful to recommend a counter-example because after it detects the class of a given topology, it can find topologies in the same class and verify that the number of links is different. For example, the k-NN classifier classifies topologies E and F in class five; they can be example and the corresponding counter-example because the number of links of E is different from the number of links of F. Table 7.4. shows the values of the measures of performance of topologies E and F.

Table 7.4. Measures of performance of topologies E and F

Measures	*Topologies*	
	E	F
Reliability	2	2
Capacity	2238 (Low)	2092 (Low)
Cost	22200 (Low)	22700 (Low)
Number of links	9	13
Class	5	5

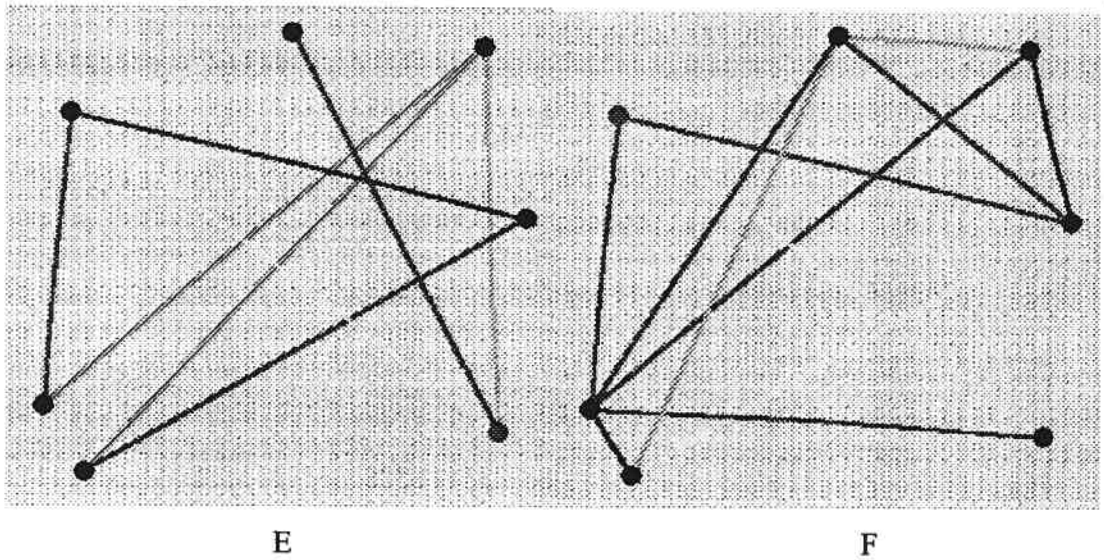


Figure 7.4. Using the classification of the k-NN classifier, topology E is an example and topology F is a counter-example.

Selection of the classification method

Although the result of testing the accuracy of k-NN classifier (96%) is better than the result obtained by the Naïve Bayes classifier (94%), there are some reasons to prefer the latter. The Naïve Bayes algorithm is easy to implement because it has *no tuning parameters*. In contrast, tunning a k-NN classifier according to other topic characteristics requires a lot of work and tests. For example, a k-NN classifier requires adjusting *the weights* of each attribute properly to generate good results. Instead, a Bayesian classifier is more *flexible and easy to adapt* to different domains and number of attributes. In addition, a Bayesian classifier does not compare an example that is yet to be classified with those already classified, as the k-NN classifier, which classification time is long. Instead, a Bayesian classifier can keep a summarized image of the frequency with which different characteristics of an example appear in the database. The probabilities of the attributes are updated each time a new example is classified; thus, the classification of a new instance involves only computing the maximum class posterior probability (Keller, 2002b). In this way, search time needed to find the most suitable class for an example consumes low time and memory requirements. Another argument in favour of naïve Bayesian classifiers is that they reduce the latency between requests and answers via Internet (Webb *et al.*, 2001; Miyahara and Pazzani, 2002). A recommender that uses a Bayesian classifier as ours is scalable because the number of examples can increase without affecting the time to respond highly. Table 7.5. shows a comparison of the two classifiers that we tested. This comparison allowed us to justify the selection of the Bayesian classifier based on measures of performance because the Bayesian classifier is easier to implement than the k-NN classifier. Figure 7.5. illustrates the performance of the two classifiers.

Table 7.5. Comparison of the two classifiers.

	Classifiers	
Aspects	Bayesian classifier	*k*-NN classifier
Parameters required	No	Yes
Tunning parameters	No	Yes
Accuracy	High	High
Scalability	Easy	Difficult

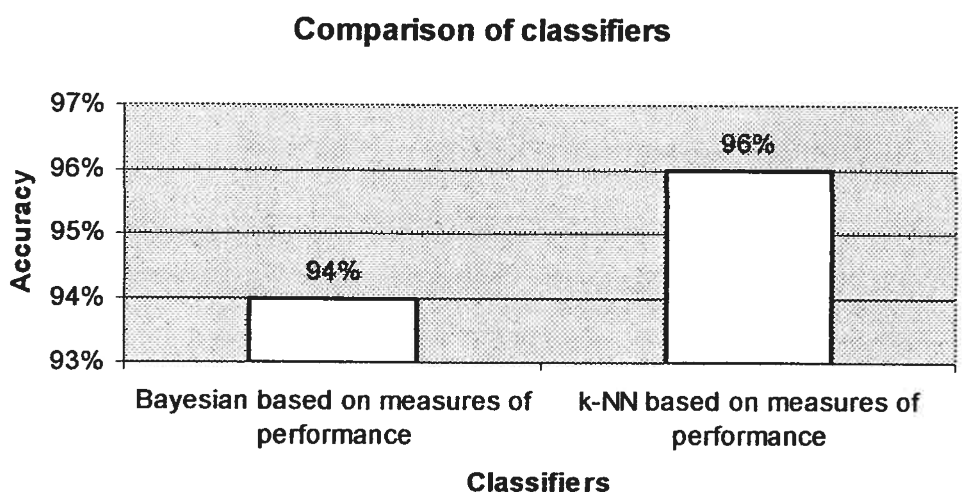


Figure 7.5. Comparison of the accuracy of the different classifiers.

7.3 Evaluation of the acceptance of the help approach

The evaluation of the acceptance of the help approach used in ARIALE includes the analysis of teachers' point of view about

- automatic generation and recommendation of examples (problem-solving support) to save time in creating teaching materials
- our learning support approach to reduce the time that teachers need to learn how to use software required to create teaching materials.

For this study, we selected 27 teachers that work in the Universidad de Costa Rica (a public university in San Jose, Costa Rica) and have been using a Web site to teach a course at least in 2005. We selected those 27 teachers because they have used an authoring tool to create their Web sites and they were receptive to answer the survey. Although this survey is an important study, we think that its results are restricted to this research and can not be generalized to other universities. The teachers are lecturers (51%; 14 teachers) and staff (49%; 13 teachers), most of them hires a master degree (51%; 14 teachers), teach courses related to basic sciences and engineering (74%; 20 teachers) and, after more that five years teaching in their area of expertise, they consider themselves as intermediate (37%; 10 teachers) or advanced (55%; 15 of them) teachers in teaching their subjects. In this group, we find 13 women and 14 men, and all people ages are between 30 and 55 years old, except two teachers who are younger than 30 years old. Teachers think that they are skilled users of computers, the 85% (23) of all interviewed teachers classified themselves as intermediate or advanced users of Internet for teaching, but 62% (17 teachers) consider they are beginners using a Website to support teaching.

Each of the 27 teachers was interviewed personally and they answered 25 questions, 12 related to problem-solving support and 13 about learning support (See Annex B). Some questions refer to the teachers´ experience dealing with authoring tools, other questions focus on general concepts and principles applied in our research.

7.3.1. Acceptance of problem-solving support

Automatic generation of examples, exercises or cases is not a common practice in the group of interviewed teachers. More that the half of teachers does not use teaching material that can be generated automatically, but 45% (12) of teachers think, at least partially, that the automatic generation and recommendation of examples reduces the time required to produce teaching materials. More precisely, 30% (8) of teachers agree that automatic generation and recommendation of examples according to a teacher's profile saves time needed to create teaching materials. 4% (1 teacher) of teachers agree very much this idea and nobody disagree it. 11% (3) of teachers partially agree that automatic generation and recommendation of examples, exercises or cases reduces the time that teachers need to create teaching materials. Figure 7.6. depicts the distribution of these opinions.

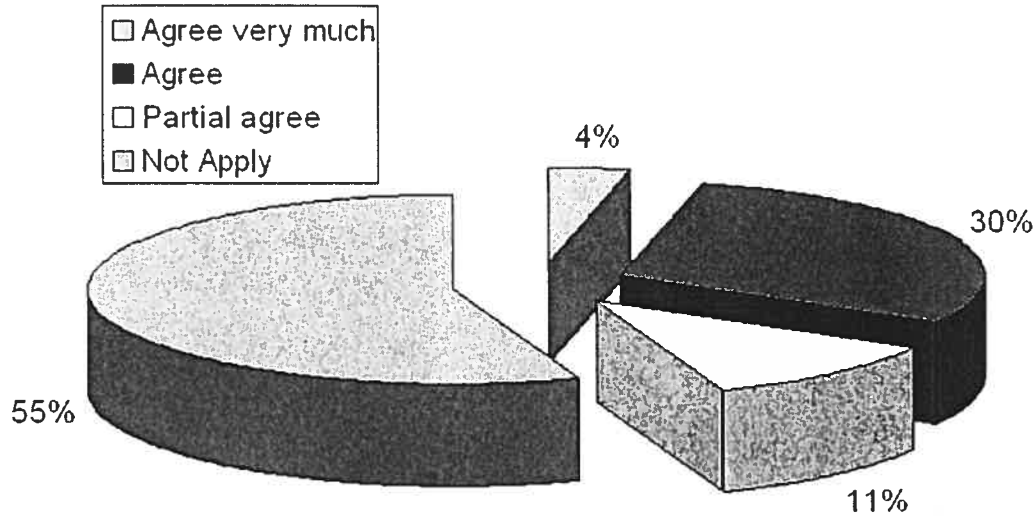


Figure 7.6. Distribution of teachers′ opinions about the affirmation that generation and recommendation of examples saves time in creating teaching materials.

An important aspect behind the automatic generation and recommendation of examples is that university teachers do not have enough time to create their teaching materials, and, according our survey, 55% (15) of the interviewed teachers consider that the time they have to create digital teaching materials is not enough. An additional 15% (4) think the time available barely suffices and 26% of teachers have enough time, or more than enough, to create their materials. In conclusion, 74% (20) of teachers feel that the time available to author teaching materials is short and more time is needed. Though teachers lack time to work on their teaching materials and regarding information collected in this research, it is possible to affirm that the interviewed teachers feel comfortable with the automatic generation and recommendation of examples or cases as a way to get more time available. Figure 7.7. shows the distribution of teachers′ opinions about time available to create digital teaching materials.

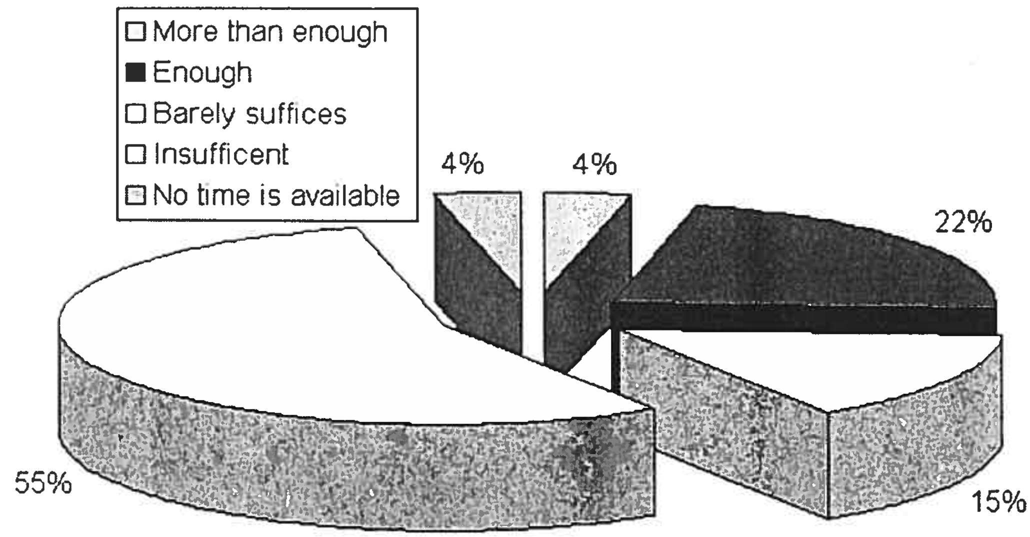


Figure 7.7. Distribution of the quantity of the time that teachers have to create teaching materials.

Another aspect is that 30% (8) of teachers, who generates examples, exercises or cases automatically, considers that between the 10% and 30% of their teaching materials can be generated automatically by means of particular applications or spreadsheets. These percentages of materials are important and, based on them, many teachers expressed that the automatic generation of examples saves time. For example, an economist who teaches finances told that building models to generate materials automatically is time consuming, but once the model is working to create examples or cases is easy and saves time.

Regarding the acceptance of recommendations by teachers, we found that 92% (25) of the teachers accept that a computer or software recommends teaching materials for teaching, and this percentage of teachers also accept that software uses teachers' data related to their preferences to recommend examples, exercises and cases similar to the ones that they apply in their courses. Although 56% (15) of teachers are not using examples, exercises or cases that can be generated automatically, 44% (12) of teachers accept software that generates and recommend examples, cases or exercises according to their preferences. Figure 7.8. shows the distribution of the teachers who accept automatic generation of teaching materials according to their preferences.

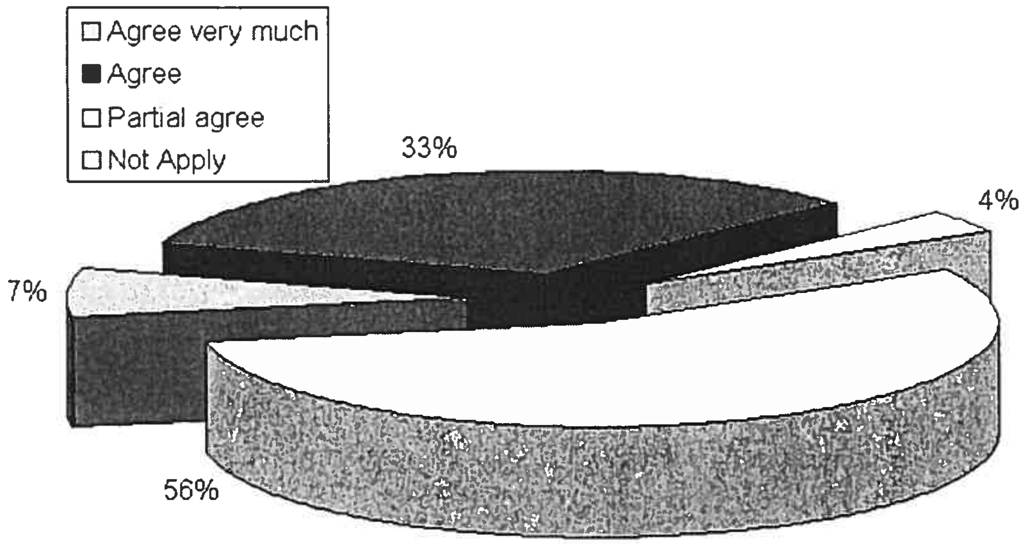


Figure 7.8. Distribution of the opinions of teachers who accept software that generates and recommend teaching materials according to their preferences.

Other aspect of our research is that sharing teaching materials is personalized. In this case, 85% (23) of teachers want to share their own materials with teachers who teach the same course. Moreover, 17 teachers (63%) also agree to reuse teaching materials that are created by other authors, especially if those materials are similar to the materials that most teachers use often. This means that teachers not only want to share materials, but 44% (12) of teachers also like to receive materials that are close to their preferences as ARIALE does.

These statistics mean that, although most teachers are not using teaching materials that can be generated automatically, almost half of the interviewed teachers like automatic generation of examples or cases that are closer to their preferences. In many cases, teachers do not figurate out how to model their typical examples just because they are not familiar with Case Base Reasoning (CBR; see chapter two), for example. They are used to create documents and presentations that includes images and videos, which can not be generated automatically, particularly in the area of social sciences. A final remark of this section that an important percentage of the interviewed teachers accept the automatic generation of examples, exercises or cases, and the recommendation of them in a personalized mode, which is better than just sharing materials between teachers with no adaptation to each teacher's characteristics.

7.3.2. Teachers and the learning support approach

According to the survey that involved 27 teachers, 70% of them (19 teachers) consider that a personalized context-sensitive help that appears in the same window where a teacher is working, as used in ARIALE, reduces the time they need to learn how to use software or applications to create teaching materials. In addition, 11% (3) of teachers agree very much this help approach used in ARIALE, and 15% (4 teachers) partially agree the affirmation that our learning support approach reduces the time required to learn the use of tools to create digital teaching materials. Nobody expressed discrepancy with our approach but one teacher did not answer this question related to learning support. Figure 7.9. shows the distribution of the teachers' opinions.

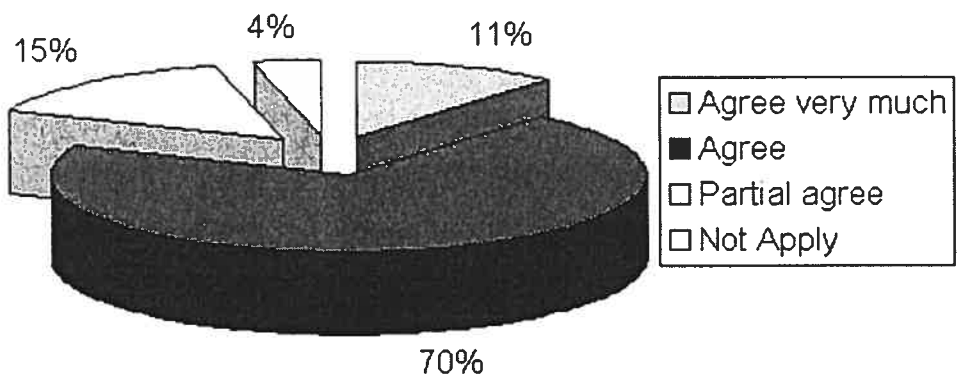


Figure 7.9. Distribution of teachers' data about the advantages of our learning support approach to save the time the teachers need to learn tools.

The interviewed teachers not only think that our approach saves time in learning to use an application or authoring tool, but they also prefer our method to display learning support or help contents embedded in the same window where the teacher is working. As shown in Figure 7.10., 48% (13) of teachers prefer to receive help embedded in the same window they are working, as implemented in ARIALE. Other 30% (8) of teachers prefer help displayed in a window different from the window where they are working, and 22% (6 teachers) likes to see help contents in a tooltip that pops up near to the concept about they need support.

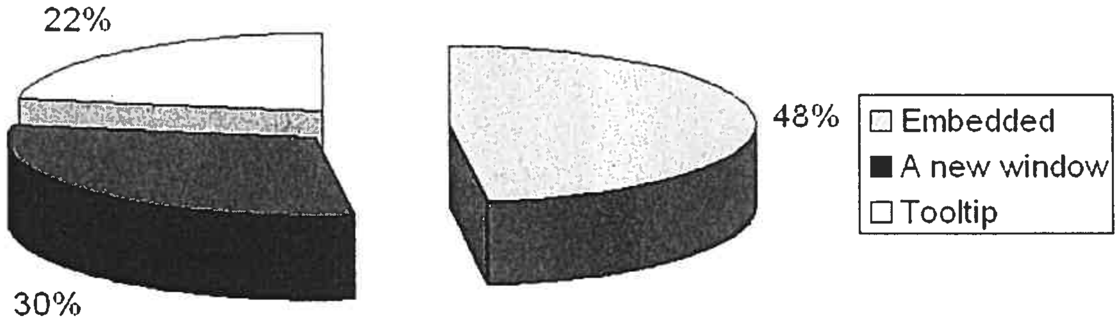


Figure 7.10. Distribution of teachers´ preference about displaying help contents.

In this group of 27 teachers, only three teachers (11%) have never used help usually included in applications or tools used to create digital teaching materials, and most of teachers access help always (7%; 2 teachers), often (30%; 8 teachers), sometimes (30%; 8 teachers or seldom (18%; 5 teachers). One teacher (4%) did not answer the question related to this topic. Then, teachers access typical help included in most of software used to create teaching materials, such as word processors, spreadsheets and authoring tools to create Web-based courses. Using help included in those applications, the enquired teachers have faced problems such as they do not know the keywords to find correct help (18%; 5 teachers), help contents are too long (11%; 3 teachers), help is not precise neither context sensitive (30%; 8 teachers) or there are not examples explaining how to manipulate software in use (11%; 3 teachers). Of course, teachers face more problems while using software but the previously mentioned are the major, according data collected in our survey. To overcome these previous difficulties, 37% (10) of teachers prefer that the application in use displays available help contents to allow teachers to make their own decisions about accessing or leaving help. Other 26% (7) of teachers think that displaying help step by step, starting by short information to gradually show more detailed help adapted to the teacher's progress is another good option. However, 11% (3) of teachers prefer a wizard or an assistant to solve the problem which solution requires help, and 15% (4 teachers) is used to find help information by themselves. Moreover, when teachers are accessing help, they (30%; 8 teachers) prefer to receive information about all functions related to the current task that they are performing; or only about the specific function that they are using (22%; 6 teachers). For 18% (5) of teachers, the best way to get support is receiving help only about the functions that are useful to finish the task that they are performing. In other cases (15%; 4), teachers prefer help contents of all functions related to the one they are using in a specific moment. Only one teacher expressed that he prefers to see information about all functions available in the application or tool that he is using. We did not get answers related to this aspect from 11 % (3) of teachers. According to collected data, teachers prefer help focused on isolate specific functions and on functions related to the tasks they have on hands. The answers of most teachers let us verify that they prefer context-sensitive help, gradually displayed in a passive manner that allows teachers to make their own decisions about rejecting of accessing learning support.

Not only context-sensitive but also personalized help are preferred by teachers. As described previously, the interviewed teachers are familiar with traditional help that very frequently is not personalized. Another aspect is that they prefer to access help adapted to their knowledge about the applications that they are using. More detailed, 74% (20) of teachers agree receiving help adapted to their particular knowledge about the use of an application or tool. In addition, 7% (2) of teachers agree very much to access personalized learning support, and 11% (3 teachers) partially agree personalized help. One teacher did not express any opinion, and another one disagree personalized help. Figure 7.11. shows how teachers preferences about personalization are distributed. Another aspect related to personalization is that 77% (21) of consulted teachers agree to receive help contents by means of their preferred medium. For example, people whose learning style is closer to audiovisual media would prefer multimedia help. Personalization is a characteristic of support in ARIALE, not only in problem-solving support but also in learning support, which teachers would appreciate to find in many other applications and tools to author teaching materials.

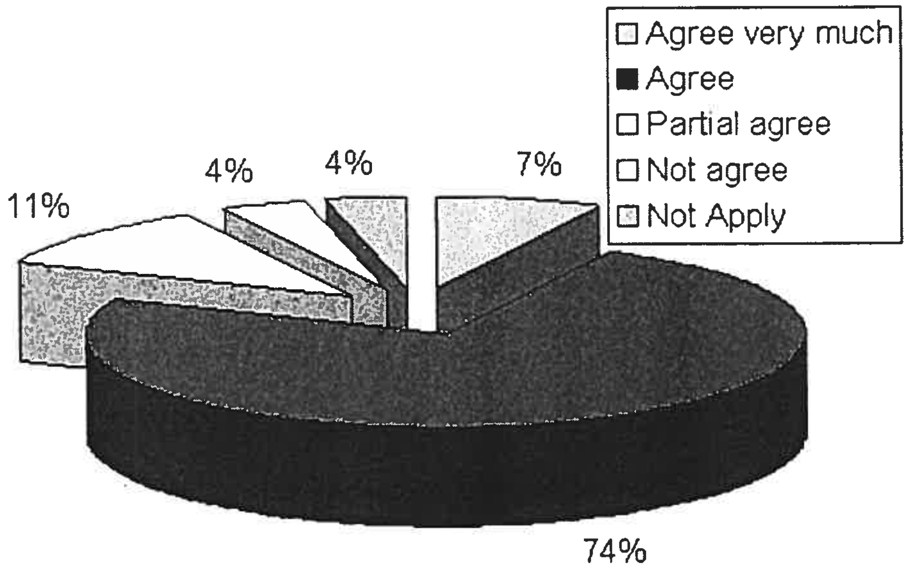


Figure 7.11. Distribution of teachers' preference about help personalization.

A working hypothesis that we used in this research is that teachers disagree to be evaluated about their knowledge and learning to use an application or authoring tool. However, 74% (20 teachers) of the interviewed teachers agree that software evaluates their skills about using applications or tools to create teaching materials. Now, we can say that evaluating teachers' progress using a tool is not a barrier for the on-the-job training approach that ARIALE applies. However, the scarcity of time for learning can be a hindrance for the evaluation of teachers' learning. If teachers do not have enough time for learning, they will not have enough time for evaluations, independent of their purposes. Collected data show that 30% (8) of the teachers think that the time that they have for learning applications or tools is not enough, 4% (one teacher) says there is no time available for learning, and 44% (12 teachers) consider that the time available to learn the use of applications barely suffices. 22% (6) of teachers think that the time the have to learn the applications is enough. Figure 7.12. depicts the scenario of the teachers' opinions about available time to author teaching materials. Furthermore, 55% (15) of the teachers who answered our survey say that a major problem that they have faced to use applications is the scarcity of time to learn the use of those applications.

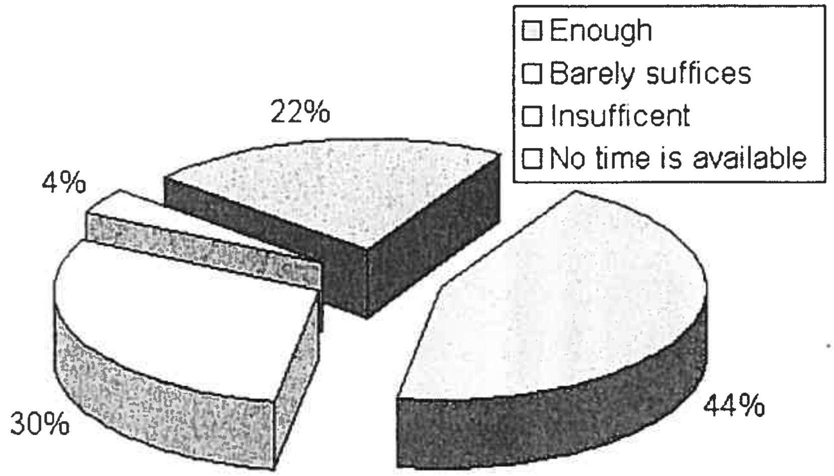


Figure 7.12. Distribution of teachers′ opinions about available time to author teaching materials.

About learning to use software, we found a contradiction in the teachers′ data, because 37% (10 teachers) of them want to receive formal training previous to getting started using an application, but, at the same time 30% (8 teachers) agree that learning by doing is the best way that university teachers can take to progress learning to use an application. The problem is this 37% (10) of teachers would prefer to receive a formal course to learn to use software, but for this group time for learning barely suffices. They partially agree or totally disagree learning by doing. In addition, the survey shows that 40% (11) of teachers partially agree to affirm that learning by doing is the best option that university teachers have for learning. A subset that represents 15% (4) of teachers is the people who prefer formal training. 30% (8) of teachers agree learning by doing, and 15% (4 teachers) agree very much learning by doing. 15% (4) of teachers totally disagree learning by doing to support learning to use an application. The learning style of the teachers who prefers learning by doing, are self learning oriented and involves reading books (25%; 7 teachers), accessing tutorials (15%; 4 teachers) and asking help to other professors (11%; 3 teachers). Figure 7.12. shows the distribution of the teachers preferences about learning by doing.

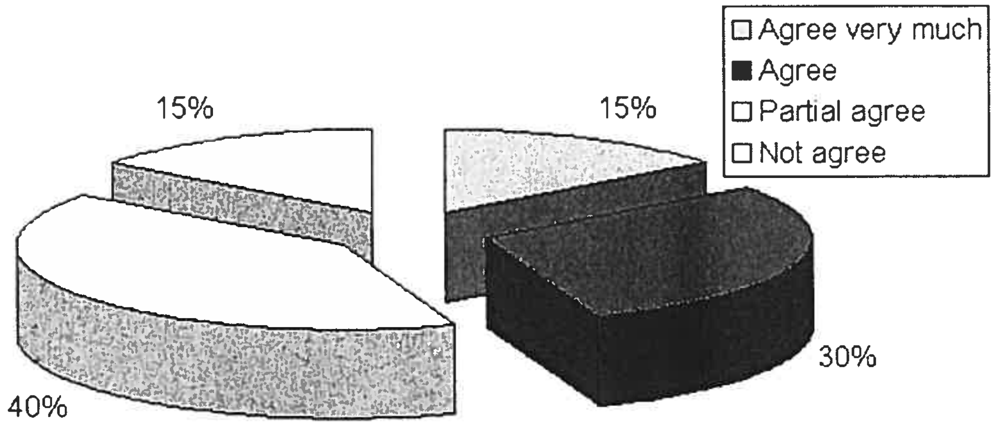


Figure 7.13. Distribution of teachers' preferences about learning by doing as the best way that teachers have to learn the use of software.

Cuneo *et al.* (2000) have analyzed the teachers' preferences about formal training and other methods for learning, and their conclusion is that, in general, universities should spent more resources in supporting teachers with online help, manuals and computing staff support than in formal training activities. We consider that our learning support approach that involves on-the-job training and learning by doing as part of an electronic performance support system is a good alternative to formal training when there is no enough time available. Teachers' data collected by means of our survey support the affirmation that personalized and embedded help makes the learning support for teachers more comfortable than traditional help.

7.4. Discussion

Although this research has generated knowledge touching on different areas, our conclusions are strengthened by offering scientific critics of our own work. In this section, we examine some aspects of ARIALE.

An initial critics to our approach was that an assistant as ARIALE is not needed because if a teacher need some support or the creation of teaching materials, he only calls to human support or live help. For example, if a teacher needs some examples of network topologies, he only contacts a human teaching assistant, namely "Richard", and the teacher receives the examples some hours or days later. But Richard usually can not be a 24 hours helpdesk and, if Richard changes for Charlie, the new assistant must learn the preferences and characteristics of each teacher. If the teachers needs help to use an authoring tool, the argument is that he only needs to call support to solve the problem or check a Frequently Asked Questions file or Web-page. But a teacher usually prefers to communicate other teachers or solve the problem by himself (Cuneo *et al.*, 2002) without checking complex help files. Finally, we decided that an assistant as ARIALE is a good option because teaching assistants are not always available, they can require a while to adapt to each teacher and, many times, support is not personalized.

Other critic focuses on that any process to train a person requires a test to diagnostic his initial knowledge related to the subject to be taught; and, another test after the training shows the improvement that the trainee experienced. Then, ARIALE should also follow this approach. However, most people are eager to use any application with no previous test and training and they prefer the "on-the-job training" style focused on real tasks to be performed. This is the reason because we adopted the *Electronic Performance Support Systems* (EPSS) approach, which provides access to support, tools and resources to perform tasks in a real work environment. This decision is supported by a test that included the 27 teachers mentioned in section 7.3. This test used a paper prototype of ARIALE and a walkthrough method.

A third critics is that a Web-based implementation of ARIALE does not have justification. This comment is partially true because the EPSS approach, a personalized embedded help or the generation and recommendation of examples are not exclusive of the Internet. But it is not forbidden. This project is based on the BestNet Web-site (Vázquez *et al.*, 2001), which objective was to share tools and examples between teachers in Australia and Canada. Then, since the beginning, our project was intended to be accessed from anywhere by any teacher or student, and the structure of an educational adaptive hypermedia working on the Web is a good option to accomplish those requirements. On the Internet, a teacher can access his profile, tools and resources with no dependency in a particular computer that keeps cookies or other type of local files with his data or applications installed in the client side. The Web-based implementation made this project more complex and challenging.

An additional critics to ARIALE is that the automatic generation of teaching materials is not a common research field. This subject and modelling the human teacher characteristics are two neglected aspects in the area of ITS and EAH in which our research has contributed. However, just because a subject is new and neglected does not mean that studying the topic is not justified. Automatic generation is important because the creation of examples is necessary when there are only few teaching materials to share or teachers do not want to share their materials. The point here is that automatic generation, test and recommendation of examples goes forwards the frequent simple sharing of teaching materials that other projects use to solve the lack of time to author teaching materials. Automatic generation studied in this research can be generalized to other fields related to decision making. In addition, there are no ITS nor EAH that follow the same approach and pedagogical model adopted in this research, although Chapter three refers to projects that are related to ours. Hence, there are no similar research projects with which to compare our approach, and this is coherent with the objective of this project that is, precisely, to innovate with new contributions.

An important obstacle we encountered is that *real implementer-teacher's experience is not available* in this research. The reason is that we had only one expert, the designer of the Network Design course used as an example for implementation. This designer was also the only implementer-teacher and end-user of the system. However, we conducted a survey intended to measure the acceptance of our approach by other teacher who also lack time to create teaching materials and to learn how to use author them. Resuls of this survey are discussed previously in this section. Although our approach has been evaluated, more specific experiments with more implementer-teachers and courses in other fields related to decision-making should be conducted in the future.

Experts in Intelligent Tutoring Systems, Educational Adaptive Hypermedia, and User Modeling asked why *the Pedagogical Model is not the Teacher Model in this research.* Usually, any ITS or EAH has only one designer's approach for teaching a subject and no implementer-teacher is taken into account. As a consequence, the Teacher Model is static and embedded in the Pedagogical or Expert Model. For this reason, other researchers do not justify modelling the teacher as we do. But the idea behind this research is to allow the inclusion of different teachers' visions and characteristics. Another criticism is that there is no justification for our Teacher Model if the teacher can be treated as a learner. However, the teacher's role and goals are different from those of students, and it is convenient to take into account each teacher's characteristics, not only for learning but also for teaching. Chapter four details not only our view of the Teacher Model but also the Pedagogical Model behind it.

Limits

We did not study the aspects related to the students in this research. This abstraction was necessary to focus on the implementer-teacher's context, but did not leave us time for the integration of traditional teachers' and students' fields of study. As a consequence, we did not analyze how to help implementer-teachers with the adaptation of lessons to fit each student's characteristics.

Although we have proposed an authoring space for implementer-teachers, we were not interested in developing *tools for course designers* because they were not our focus. To design curricula, the access to our system is limited to designers supported by high-level computer science professionals.

Another aspect is that ARIALE is not designed for teaching decision-making with emphasis on numerical methods that could be very specific for particular subjects. This could leave out some teachers whose teaching styles are based on the use of numerical methods, but our approach is to support teaching based on qualitative approaches, expert knowledge, and heuristics. The interest of this research is to provide an approach that is different and complementary of numerical ones and this is the reason because we decided to focus and deep in our proposal, leaving out other options.

The next section focuses on the innovations and contributions of our investigation. These contributions make ARIALE new, different, valuable and useful for the solution of practical problems in authoring educational adaptive hypermedia and in the development of intelligent help systems.

7.5. Conclusion

We started this research with five questions included in the first chapter. The first question asks the structure and functionality of an assistant to support teachers authoring learning sessions in decision-making. To answer this question, we designed and developed *an assistant* which structure and functions include a probabilistic generator and *a recommender* of network topologies that provides examples according to the teacher's characteristics. Automatic generation and personalized recommendation of examples are functions which distinguish our system from others. In addition, this assistant learns from its interaction with teachers. An assistant that uses methods and techniques of artificial intelligence to solve problems related to the support of authoring teaching materials is an important step forward in the progress of educational adaptive hypermedia.

A second question that guided our study refers to the specific functions and characteristics of an authoring tool to adapt teaching material according to each teacher's pedagogical goals. Our authoring tool divides the curriculum of any course into the main concepts. Each concept maps to an activity that allow three specific tasks to be done. These three tasks implement the pedagogical strategies used for teaching in our Pedagogical Model (section 4.1.): analysis by comparison, sensitivity analysis and optimization. The different tasks are accomplished by following a sequence of steps and actions (Rodríguez *et al.*, 2003). Using the authoring tool, the teacher can reuse learning sessions and examples and edit them. The authoring tool includes an editor of network topologies and both of them are discussed in section 6.3.

The third question focuses on how the system generates and recommends examples to support teaching decision-making. The generation of examples uses a probabilistic method that provides a great variety of examples. The system can generate new examples according to the teacher's preferences and add each new example to the base of cases. The *generation of examples,* that can be modeled and represented by means of numerical structures, decisional tables or graphs, such as network topologies, is not commonly used by ITS and EAH. To test and recommend or not each example generated automatically, ARIALE applies principles and techniques of artificial intelligence, such as a classification learning (Bayesian classifier) and the Case-Based Reasoning (CBR) approach. Using classification learning and CBR is an important improvement over previous EAH that did not take into account teachers′ differences and the evolution of their teaching styles. Automatic generation and the recommendation of examples are explained in section 5.2.

This type of support and the recommendation of examples are parts of an intelligent help system that helps the implementer teacher while he is authoring learning sessions. More specifically, when teachers are using the authoring tool to create an interactive learning process in decision-making, an adaptive hypermedia intelligent help system can help them. To help teachers, ARIALE chooses the appropriate type of help (problem-solving support, learning support or both of them), and the respective content and media to display the content. The procedure to define help is related to the fourth question: How does the system make decisions about which kind of help content to show, and which media to use for displaying the content? Chapter five explains the procedures that ARIALE uses to select the examples of learning sessions, network topologies, help contents and media to display the selected contents. In summary, our system uses a Bayesian classifier to test and select examples of network topologies; a learning session is recommended if the class of its index is the same of teacher's preferred class (teaching style); to choose contents for learning support, ARIALE filters help contents which attributes match the teacher's skills, plan to be performed the help techniques associated to them and the teacher's preferences, such as language or media (see section 5.3. for details). Hence, support provided is adapted to the teacher's characteristics, learning and accumulated experience accessing the support. The teacher's attributes, learning and experience conform a Teacher Model that models his teaching styles and is the basis to personalize support. In addition, we found that there are not enough studies on traditional help for desk applications and about the help required for Web-based authoring tools. In this research, we are presenting an initial and non exhaustive partial analysis of traditional help and Web-based help as an attempt to address the problems related to supporting teachers that author teaching materials.

The final question in our research refers to the procedure that the system applies to learn how to help teachers in a personalized or customized way. Our answer is that information about the interaction between the teacher and the system allows ARIALE to learn how a teacher uses examples to teach and how he uses help, thus enabling the system to customize the support to be provided. More specifically, our system complements artificial intelligence techniques as Bayesian classification and Case Base Reasoning (CBR) to learn about teachers preferences. In addition, ARIALE keeps, classifies and use data related to each teacher's teaching style, attributes, learning and experience using examples. ARIALE learns from these data to improve future support to teachers.

This research not only let us to answer the five questions previously discussed, but our investigation also allowed us to make some contributions and innovations in the area of Educational Adaptive Hypermedia.

7.5.1. Main innovations and contributions of our research

The recommendation of learning sessions and examples, the automatic generation of examples, and the learning of the teacher's decisions related to teaching style and help, are the most important contributions of this research. In addition, we contribute with other innovations intended to improve the role of the implementer teacher in authoring EAH. In this section, we summarize our main contributions and other contributions are described in the next section.

Recommender of session and examples

Our idea of providing more personalized problem-solving support is a new one and is a step forward to support more *intelligent collaboration* between teachers, beyond the simple sharing of examples. To surpass the simple sharing of examples, ARIALE uses a probabilistic recommender supported by a *Bayesian classifier* that *reduces the complexity of finding* an appropriate learning session or example for a particular teacher. ARIALE supports teachers to author interactive learning sessions by adapting support according to their characteristics, plans, and skills. Using a *teacher's profile to adapt problem-solving support* is an innovative aspect that distinguishes our solution from other types of authoring tools. Most authoring tools in ITS and EAH areas do not have traditional help nor knowledge-based problem-solving support. The results of the evaluation of our classifier in comparison with other classification methods (see section 7.2.) let us to affirm that our recommender is appropriate for the recommendation of learning sessions and examples.

Example generator

The *automatic generation* and *testing* of examples or cases to recommend the examples that match a specific teacher's interests, is an innovation in the fields of Intelligent Tutoring Systems and Educational Adaptive Hypermedia. The *generation of examples* is based on a probabilistic method and this type of automatic generation is intended to reduce the effort required by the teacher to create support materials. If a teacher reduces the time that he needs to author this type of teaching materials, then he will be free to spent this time in dealing with edition of video, audio or other multimedia resources that are not only time consuming but also difficult to model.

An assistant that learns from the interaction with the teachers

Another important aspect of this research is that ARIALE learns from the *interaction between our system and the teacher*. This interaction allows ARIALE to learn each teacher's *teaching style* and its evolution. For example, ARIALE learns when it identifies the class of examples of topology that a teacher prefers. The system also learns when the teacher switches to use topologies from another class because information is updated in the Teacher Model (section 5.4. details how the system learns). Because the teacher chooses and edits examples, this process allows the system to learn the teacher's teaching style, to improve future support. Our solution proposes not only to acquire and share expert knowledge from the interaction between the teacher and system, but also to learn how to recommend sessions and examples according to the teacher's evolving knowledge. ARIALE also learns from the progress in the teacher's *level of experience* using the authoring tool and the help associated.

With these main contributions, this research has achieved its intended objective: finding an option to support teachers who do not have enough time for authoring teaching materials or learning how to use an authoring tool to create materials. But this research generated other contributions that are described as follows.

7.5.2. Other contributions

As a complement to the main contribution summarized above, other contributions of our research in the field of EAH are a flexible Pedagogical Model oriented not only to teach students but also to support teachers; a Teacher Model that models each implementer teacher and that evolves according to the decisions that each teacher makes while he is authoring teaching materials; a Virtual Authoring Space (VAS) for implementer-teachers; and, a method for the design and development of an Intelligent Help System for an authoring tool.

A more flexible Pedagogical Model

The classic approach in EAH uses a unique Pedagogical Model oriented to students that each teacher must follow. In ARIALE, the Pedagogical Model is more *flexible* because is oriented to students and teachers. This Pedagogical Model is *hybrid* because not only contains how to teach a particular subject to students, but it also includes how to support each teacher while he is authoring learning sessions. In addition, the authoring tool allows the implementer teacher to tailor the Pedagogical Model by changing *contents* and *examples*. In this shift, the Pedagogical Model becomes an instrument to support teachers, more than to conform a perfect automatic teacher. Finally, traditional tutors and EAH are designed for formal teaching or training oriented to evaluation, but in this case we adopted the Electronic Performance Support System approach, intending it to supply *personalized* support while the teacher is authoring a learning session. In our Pedagogical Model, learning to use an authoring tool is not a requirement for teachers. Learning is a *by product* of accesing support. Section 4. 1. explains the Pedagogical Model.

Another view of the Teacher Model

Our main innovation related to the Teacher Model is that we use an evolving model of the implementer teacher. The examples and pedagogical goals that a teacher uses to teach a subject are decisions that can change with the pass of the time. ARIALE monitors the decisions and stores data related to them. Doing this the system not only updates each teacher's information but also learns and improves its ability to deliver future support. In addition, we do not see the teacher as a typical student. Rather, we see the teacher as a worker who can learn while he is on his job. Section 4.2. details the Teacher Model.

Another important aspect is that our research deals with teacher modelling, which is a neglected aspect of Educational Adaptive Hypermedia (EAH) and Intelligent Tutoring Systems (ITS). The Teacher Model that we have implemented includes information about

- teaching and learning aspects
- teachers' skill levels for each task

In ARIALE, this information about the implementer-teacher is stored and applied to the recommendation of learning sessions, examples, and the adaptation of help. We not only worked on the theoretical definition of this model but also on its application. In addition, this model is part of the bridge that we built between course designers and implementer-teachers.

A space for the implementer teacher

Another new aspect is the development of a Virtual Authoring Space (VAS) for implementer-teachers (section 6.3.). This VAS is the authoring tool that allows implementer-teachers to interact with and adapt an educational adaptive hypermedia to their interests. This tool supports an intermediate level that is absent in classic ITS, EAH, and authoring tools, which usually includes only a space for students and other for designers of complete courses or curricula. As one of the features included in the authoring tool, there is a Network editor that allows teachers to configure topologies according to their preferences. This editor is the current interface between the recommender of examples and the teacher.

About the implementation

As a result of our experience developing ARIALE, we have integrated parts of some guides and methods for the design and development of hypermedia applications and Intelligent Help Systems (section 6.2.). This is a contribution to the development of authoring tools for Adaptive Learning Environments (ALE). This method includes a phase for implementing the Web site and the authoring tool that follow the typical methods used in software engineering. Another phase is for the creation of the help system. The following is a summary of the method that we used:

Part 1. Implementing the Web site and the authoring tool.

- Requirement analysis
- Task analysis
- Definition of authoring tool contents
- Prototype of the authoring tool

Part 2. Implementing and integrating the help system.

- Identification of events that call for adaptation (see section 6.2.2.).
- Determination of the type of adaptation to be performed
- Definition of the inference mechanism
- Definition of help contents
- Definition of the content structure
- Definition of the relations between fields and contents

This method is complemented by the architecture of the system, which includes components such as the Teacher Model, a recommender, an example generator, and an engine to provide help. The implementation of ARIALE has been complex because it covers two environments: *authoring* and *helping to author*. This involves working with both the *learning* and *teaching styles* of the teacher.

7.5.3. Integration of other important concepts

Parallel hypermedia is a concept that we introduce to represent the integration of different hypermedia (the authoring tool and the help in ARIALE) that the teacher can *navigate simultaneously*. We identify the navigation between the nodes of the help hypermedia, which is different from the navigation between the nodes of the authoring tool but closely linked to it.

In this research, we have also used some ideas that are not original, but its application and integration is innovative. We think that the integration of the Cognitive Learning Theory (CLT) and the concept of "cognitive load" in adaptive learning environments such as EAH, is a helpful option to support teaching decision-making skills. In this research, we integrated CLT and the Payr's model for teaching and learning as a theoretical layer on the basis of teaching decision-making skills.

Another integration of concepts involves adaptation, Web-based help, embedded help and context-sensitive help. Usually, these concepts are not presented simultaneously, but in ARIALE we attempted to deal with their integration.

7.5.4. A final remark

According to studies of teachers' preferences and attitudes toward new information and communication technology applied to education (Cuneo *et al.*, 2002; Weiss-Lambrou and Raymond, 2002), we can affirm that our research tackles a current and real problem: university instructors "rely on their own resources, on peer networks or on personalized computing staff assistance rather than attending workshops to obtain academic technology help" (Cuneo *et al.*, 2002). This situation occurs especially with authoring tools such as WebCT, Blackboard and Virtual-U. We agree with the recommendation of Cuneo *et al.* (2002) that universities should invest in "manuals, online help documentation and computing staff support, rather than workshops and conferences, to assist their faculty with computing", but we add that this support should also include intelligent help in cases where adapted and personalized help provided by the computing staff is expensive. One important option can be to "amplify user's intelligence instead of doing intelligently his work ..." or "... to support user's task rather than automate it" (Delisle and Moulin, 2002).

Our experience studying teachers' reactions to computer-based technology in education (Rodríguez, 1997) has taught us that regardless of the number of hours of training they receive, teachers can be excluded from using new information technology because they do not have time to practice the theoretical contents learned in the training phase; the edition of resources is time consuming; and there is not a logical connection between training and real tasks to be performed. Sometimes training is not linked to their real tasks or teachers do not get appropriate support related to the context of the tasks to be performed.

7.5.5. Future work

We consider that the following aspects of our research should receive further study.

The intelligent support for collaboration between implementer-teachers can be improved to integrate designers and students, taking into account the characteristics of the sessions and examples they can share and study. For example, a designer can share parts of different curricula to integrate activities into new courses. The approach of recommendation and learning based on classification can be extended to cover designers and students.

At the beginning of this research, we thought of using a planner for generating examples of network topologies. Ultimately, we switched to a probabilistic generator. However, an alternative that could improve the generation of examples would be a probabilistic planner that selects the operators at random. A planner is a method to transform an initial state into a final state by performing operations that generate a series of temporal states. A complementary option is that a probabilistic planner can transform cases from a case-base.

To support designers, a course editor and the corresponding help should be developed. The editor should include an easy-to-use tool to customize the recommender and the classifier according to the characteristics of each designer and course.

Another improvement to this system would be the addition of a pre-evaluation of students' assignments. For example, if the system classifies the student's answers to task 2 and task 3, it could then check which students are out of the range of possible answers, allowing the teacher an opportunity to analyze these answers and discuss the situation with the students in question. Of course, an important research subject is the adaptation of learning sessions and examples according to the students' characteristics. This is a huge challenge for an adaptive learning environment, especially an EAH, because the adaptation for the student should also match the teacher's teaching style.

Although some studies (Koychev, 2000; Leung, 2001) have analyzed forgetfulness in help and recommender systems, more investigation in this field is required. For example, teachers can forget how to use a function or how to author a task. This problem is linked to the need to develop the functionality of adapting the classes and the classification of examples according to the evolution of teachers' activity. For example, substitution of classes no longer in use for new ones can result in splitting frequently used classes. In this case, the systems would forget an unused class of examples.

We must recognize that a limitation of our research is the *current implementation* of the teachers' teaching style. Our definition of teaching style is helpful for the recommendation of examples and it is a step forward in the study of modelling the teaching style of implementer-teachers. However, more comprehensive study of a definition of teaching style in decision-making is necessary. A major aspect in decision-making in Network Design is *the criteria that each teacher applies* in the analysis of case studies. For example, in the case of network design, one teacher may prefer to analyze network configurations on the basis of their reliability and cost, while another teacher might prefer to use capacity and delay as his criteria. Including these criteria in the Teacher Model can be a valuable complement to improve the teaching style and the recommendations.

Bibliography

(Aamodt and Plaza, 1994) Aamodt, A. , Plaza, E. "Case-based reasoning: foundational issues, methodological variations, and system approaches", *Aicom – Artificial Intelligence Communications*, IOS Press, vol. 7: 1, 1994, pp. 39-59.

(Aberg, 2002) Aberg, J., "Live Help Systems: An Approach to Intelligent Help for Web Information Systems", *Ph.D. Thesis, Department of Computer and Information Science, Linköpings universitet,* Sweden, 2002.

(Abou-Jaoude and Frasson, 1998) Abou-Jaoude, S., Frasson, C., "An Agent for Selecting Learning Strategy", *Conference internationale sur les nouvelles technologies de la communication et de la formation*, NTICF'98. Rouen, 1998 : http://www.iro.umontreal.ca/~frasson/FrassonPub/NTICF-98-Learning-strategy.doc. Accessed November, 2000.

(Addie, 2003) Addie, R. G. Algorithms and Models for Network Analysis and Design. Australia. Electronic book. 2003: http://www.sci.usq.edu.au/courses/CSC3413/sguide/netanal.pdf. Accessed July, 2003.

(Aerts *et al.*, 1999) Aerts, A., Bierhoff, P., De Bra, P.,"Web-CS: infrastructure for web-based competitions", *Proceedings of the WebNet'99 Conference*, USA, 1999, pp. 69-74.

(Aïmeur and Bassil, 1999) Aïmeur, E., Bassil, S., "A Mixed Initiative for Teaching and Learning HTML in Intelligent Tutoring Systems", *Proceedings of the Workshop on Agents Learning About, From and With other Agents, IJCAI-99* Sweden, 1999, pp. 7-15.

(Aïmeur and Frasson, 1996) Aïmeur, E., Frasson, C., "Analyzing a new learning strategy according to different knowledge levels", *Computer Education,* Vol. 27, No. 2, Great Britain, 1996, pp.115-127.

(Aïmeur *et al.*, 2000) Aïmeur, E., Frasson, C., Dufort, H., "Co-operative Learning Strategies for Intelligent Tutoring Systems", *Applied Artificial Intelligence, International Journal,* Vol. 14, Canada, 2000, pp. 465-490.

(André and Rist, 2002) André, E., Rist, T., "From adaptive hypertext to personalized web companions", *Communications of the ACM,* Vol. 45, No. 5, USA, 2002, pp.43-46.

(Aroyo *et al.*, 2002) Aroyo, L., Cristea, A., Dicheva, D., "A Layered Approach towards Domain Authoring Support", *Proceedings of the ICAI 2002,* USA, 2002, pp. 615-621.

(Beck, 1996) Beck, J., "Applications of AI in education", *ACM crossroads student magazine,* USA, 1996: http://www.acm.org/crossroads/xrds3-1/aied.html. Accessed July, 2005.

(Bengio and Granvalet, 2003) Bengio, Y., Granvalet, Y., "No Unbiased Estimator of the Variance of K-Fold Cross-Validation". Technical Report 1234, Département d'informatique et recherche opérationnelle, Université de Montréal, 2003: http://www.iro.umontreal.ca/~lisa/pointeurs/TR1234.pdf. Accessed December, 2004.

(Billsus and Pazzani, 1999) Billsus, D., Pazzani, M., "A Hybrid User Model for News Story Classification", *Proceedings of the 7th International Conference on User Modeling (UM'99),* Canada, 1999, pp. 99-108.

(Boy, 1998) Boy, G., "Cognitive function analysis Stamford", *Ablex Pub. Corp.,* USA, 1998.

(Brusilovsky and Cooper, 1999) Brusilovsky, P., Cooper, D., "ADAPTS: Adaptive hypermedia for a Web-based performance support system", *Proceedings of 2nd Workshop on Adaptive Systems and User Modeling on WWW at 8th International Word Wide Web Conference and 7th International Conference on User Modeling,* Canada, 1999: http://wwwis.win.tue.nl/asum99/brusilovsky/brusilovsky.html. Accessed October, 2002.

(Brusilovsky and Schwartz, 1997) Brusilovsky, P., Schwartz, E., "User as Student: Towards an Adaptive Interface for Advanced Web-Based Applications", *Proceedings of the 6th International Conference on User Modeling,* Italy, June 1997, pp. 177-188.

(Brusilovsky, 1996) Brusilovsky, P., "Methods and techniques of adaptive hypermedia", *User Modeling and User-Adapted Interaction,* Vol. 6, No. 2-3, 1996, pp. 87-129.

(Brusilovsky, 2000) Brusilovsky, P., "Adaptive hypermedia: From intelligent tutoring systems to Web-based education", In: G. Gauthier, C. Frasson and K. VanLehn (eds.) *Intelligent Tutoring Systems*. Lecture Notes in Computer Science, Vol. 1839, Springer Verlag, (*Proceedings of 5th International Conference on Intelligent Tutoring Systems*, ITS 2000), Canada, 2000, pp. 1-7.

(Brusilovsky, 2001) Brusilovsky, P., "Adaptive Educational Hypermedia", *Proceedings of the 10th International Peg Conference (Peg'01),* Finland, 2001, pp. 8-12.

(Brusilovsky and Cooper, 2002) Brusilovsky, P., Cooper, D. W. "Domain, Task, and User Models for an Adaptive Hypermedia Performance Support System". In: Gil, Y. and Leake, D. B. (eds.) *Proc. of 2002 International Conference on Intelligent User Interfaces,* ACM Press, USA, 2002, pp. 23-30.

(Butler and Sellbom) Butler, D., Sellbom, M. "Barrier to adopting technology for teaching and learning", EDUCAUSE Quarterly 2, 2002: http://www.educause.edu/ir/library/pdf/EQM0223.pdf. Accessed March, 2003.

(Callan *et al*, 2003) Callan, J., Smeaton, A., Beaulieu,M., Borlund,P., Brusilovsky, P., Chalmers, M., Lynch, C., Riedl, R., Smyth, B., Straccia, U., Toms, E., "Personalisation and Recommender Systems in Digital Libraries. Joint NSF-EU DELOS Working Group Report", 2003: http://www.dli2.nsf.gov/internationalprojects/working_group_reports/perso-nalisation.htm. Accessed April, 2004.

(Capobianco, 2002) Capobianco, A., "Stratégies d'aide en ligne contextuelles : Acquisition d'expertises, modélisation et évaluation expérimentale", *Thèse de doctorat, Université Henri Poincaré,* France, 2002.

(Carreira *et al.*, 2004) Carreira, R., Crato , J., Gonçalves, D., Jorge, J., "Evaluating Adaptive User Profiles for News Classification", *Proceedings of the 9th* *international conference on Intelligent User Interface* IUI, Portugal, 2004, pp. 206 - 212

(Carro, 2001) Carro, R., "Mecanismo basado en tareas y reglas para la creación de sistemas hipermedia adaptativos: Aplicación a la educación a través de Internet", tesis de Doctorado, *Departamento de Ingeniería Informática, E.T.S. Informática, Universidad Autónoma de Madrid,* 2001.

(Carroll and Aaronson, 1988) Carroll, J., Aaronson, A., "Learning by Doing with Simulated Intelligent Help", *Communications of the ACM*, *31*(9), 1988, pp. 1064-1079.

(Conati *et al*, 2002) Conati,C., Gertner, A., Vanlehn,, K., "Using Bayesian Networks to Manage Uncertainty in Student Modeling", *User Modeling and User-Adapted Interaction*, Volume 12, Issue 4, November 2002, pp. 371 – 417.

(Copernicus, 1996) "Flexible and Distance Learning Through Telematics Networks", *Communication and Information Technologies (CIT) course, Copernicus'94,* Joint Research Project #1445, 1996: http://www.pit.ktu.lt/HP/coper/kiev.new/cit/ap_ch3/les32c.htm. Accessed July, 2001.

(Cortés *et al.*, 1994) Cortés, U., Moreno, A., Armengol, E., Béjar, J., Belanche, L., Gavaldá, R., Gimeno, J., López, B., Martín, M., Sánchez, M., "Aprendizaje Automático", *Ediciones UPC (Universidad Politécnica de Cataluña),* Spain, 1994.

(Costagliola *et al.*, 2002) Costagliola, G., Ferrucci, F., Francese, R., "Web engineering: Model and methodologies for the design of hypermedia applications", In Chang, S. K., editor, *Handbook of Software Engineering & Knowledge Engineering,* volume 2, Emerging Technologies, World Scientific, 2002, pp. 181- 199.

(Cristea and Garzotto, *2004)* Cristea, A., Garzotto, F., "Designing Patterns for Adaptive of Adaptable Educational Hypermedia: a Taxonomy", *ED-MEDIA'04, AACE,* Switzerland, 2004, pp. 808-813.

(Cristea and Mooij, 2003) Cristea, A., Mooij, A., "LAOS: Layered WWW AHS Authoring Model and their corresponding Algebraic Operators", *Proceedings of the 12th International World Wide Web Conference (WWW'03),* Hungary, 2003: http://wwwis.win.tue.nl/~acristea/HTML/Minerva/papers/WWW03-cristea-mooij.doc. Accessed March, 2004.

(Crozat, 2000) Crozat, S., "Designing pedagogical hypermedia: An information centered approach", *Proceedings of the ITS 2000,* Canada, 2000, pp. 65-69.

(Cuneo, 2002) Cuneo, C. "Globalized and Localized Digital Divides Along the Information Highway: A Fragile Synthesis Across Bridges, Ramps, Cloverleaves, and Ladders". The 33rd Annual Sorokin Lecture University of Saskatchewan January 31st, 2002: http://www.humanities.mcmaster.ca/~global/wps/Cuneo022.PDF. Accessed May, 2004.

(Cuneo *et al.*, 2002) Cuneo, C., Campbell, B., Harnish, D., "The Integration and Effectiveness of ICTs in Canadian Postsecondary Education", *2002 Pan-Canadian Education Research Agenda Symposium, Information Technology and Learning,* Canada, May 2002: http://www.cmec.ca/stats/pcera/RSEvents02/CCuneo_OEN.pdf. Accessed April, 2004.

(Davis *et al.*, 2003) Davis, J., Leelawong, K., Belynne, K., Bodenheimer, B., Biswas, G., Vye, N., Bransford, J., "Intelligent user interface design for teachable agent systems", *Proceedings of the 8th International Conference on Intelligent user interfaces,* USA, 2003, pp. 26-33.

(De Bra *et al.*, 2003) De Bra, P., Aerts, A., Berden, B., De Lange, B., Rousseau, B., Santic, T., Smits, D., Stash, N., "AHA! The Adaptive Hypermedia Architecture", *Proceedings of the ACM Hypertext Conference,* UK, 2003, pp. 81-84.

(DeLacey and Leonard, 2002) DeLacey, B., Leonard, D., "Case Study on Technology and Distance in Education at the Harvard Business School", *Educational Technology & Society,* Vol. 5, No. 2, 2002, pp. 13-28.

(Delisle and Moulin, 2002) Delisle, S., Moulin, B. "User Interfaces and Help Systems: From Helplessness to Intelligent Assistance", In *Artificial Intelligence Review*, Volume 18, Issue 2, Germany, 2002, pp. 117-157.

(Domingos and Pazzani, 1997) Domingos, P., Pazzani, M., "On the Optimality of the Simple Bayesian Classifier under Zero-One Loss", *Machine Learning Journal*, Netherlands, 29, 1997, pp.103-130.

(Draper, 1999) Draper, S., "Supporting use, learning, and education. Commentary on Guzdial's 'supporting learners as users'", *Journal of Computer Documentation* Vol.23 No.2, 1999, pp.19-24.

(Du Boulay and Luckin, 2000) Du Boulay, B., Luckin, R., "Modelling Human Teaching Tactics And Strategies For Tutoring Systems", *International Journal of Artificial Intelligence in Education,* Vol. 12, Netherlands, 2000, pp. 235-256.

(Dufresne and Paquette, 2000) Dufresne, A., Paquette, G., "ExploraGraph: A flexible and adaptive interface to support distance learning", *LICEF Research Centre,* Canada, 2000: http://www.telelearn.ca/g_access/research_projects/th6_human_machine/index.html. Accessed June, 2000.

(Dufresne, 1996) Dufresne, A., "Un environnement générique de formation au diagnostic utilisant une base de connaissances, des représentations graphiques et des hypermédias", *Hypermedia et Apprentissage (AFCET),* France, 1996, pp. 71-78.

(Dufresne, 2001) Dufresne, A., "Exploragraph: Improving interfaces to improve adaptive support", *Proceedings of the 10th conference of Artificial Intelligence in Education,* Netherlands, 2001, pp. 306-313.

(Edwards and Clear, 2001) Edwards, M., Clear, F. "Supporting the Collaborative Learning of Practical Skills with Computer-Mediated Communications Technology", *Educational Technology & Society* 4(1) 2001: http://ifets.ieee.org/periodical/vol_1_2001/edwards.pdf. Accessed November, 2002.

(Edwards and Hendy, 2000) Edwards, J., Hendy, K., 2000 "A Testbed for Intelligent Aiding in Adaptive Interfaces", Adaptive User Interfaces, Papers from 2000 AAAI Spring Symposium,Technical Report SS-00-01, 2000: http://www.isle.org/~aui/papers/JEdwards00.pdf. Accessed June, 2004.

(Ehlert, 2003) Ehlert, P., "Intelligent user interfaces: Introduction and survey", *Research Report DKS03-01 / ICE 01, Data and Knowledge Systems group, Faculty of Information Technology and Systems, Delft University of Technology,* Netherlands, 2003: ftp://ftp.kbs.twi.tudelft.nl/pub/ice/Ehlert.P.A.M-report_IUI.pdf

(Einsenstein and Rich, 2002) Einsenstein, J. Rich, C. "Agents and GUIs from task models", Interfaces *Proceedings of the 7th International Conference on Intelligent User Interfaces,* USA, 2002, pp. 47-54 .

(Eirinaki and Vazirgiannis, 2003) Eirinaki, M., Vazirgianis, M., "Web Mining for Web Personalization", *ACM Transactions on Internet Technology (TOIT)*, Vol. 3, No. 1, 2003, pp. 1 – 27.

(Encarnação and Stoev, 1999) Encarnação, L., Stoev, S., "An application-independent intelligent user support system exploiting action-sequence based user modelling", *Proceedings of 7th Conference User Modelling*, Canada, 1999, pp. 245-254.

(Encarnação, 1997) Encarnação, M., "Multi-level User Support through Adaptive Hypermedia: A Highly Application-Independent Help Component", Proceedings of the *International Conference on Intelligent User Interfaces* (IUI'97), USA, 1997, pp.187-194.

(Erkol, 1998) Erkol, E., "A multi-agent extension of Negoplan and its application to a business strategy game", Master Degree Thesis, University of Ottawa, Canada. 1998.

(Fink and Kobsa, 2000) Fink, J., Kobsa, A., "A review and analysis of commercial user modeling servers for personalization on the World Wide Web", *User Modeling and User-Adapted Interaction,* Vol. 10, Netherlands, 2000, pp. 209-249.

(Fischer, 2001) Fischer, G., "User Modeling in Human Computer Interaction", *User Modeling and User-Adapted Interaction,* Vol. 11, No. 1-2, Netherlands, 2001, pp. 65-86.

(Fischer and Steinmetz) Fischer, S., Steinmetz, R., "Automatic creation of exercises in adaptive hypermedia learning systems", *Proceedings of the eleventh ACM Conference on Hypertext and hypermedia,* USA, 2000, pp. 49 – 55.

(Frasson, 1998) Frasson, C., "Using cognitive Agents for Building Pedagogical Strategies in a Multistrategic Intelligent Tutoring System". *Deuxième journée Acteurs, Agents et Apprentissage*, France, 1998: http://www.iro.umontreal.ca/~frasson/FrassonPub/Bayonne-Agents cognitifs-sept98.doc. Accessed November, 2002.

(Frasson and Aïmeur, 1997) Frasson, C., Aïmeur, E., "Lessons learned from a university-industry cooperative project in tutoring systems", *Failure & lessons learned in information technology management,* Vol. 1, USA, 1997, pp. 149-157.

(Fritze, 2003) Fritze, P., "Innovation in University Computer-Facilitated Learning Systems: Product, Workplace Experience and the Organisation", *Dissertation submitted in total fulfilment of the requirements of the degree of Doctor of Philosophy, Faculty of Education, Language and Community Services, RMIT,* Australia, May 2003.

(Funk and Conlan, 2003) Funk, P., Conlan, O., "Using Case-Based Reasoning to Support Authors of Adaptive Hypermedia Systems", *Proceedings* of *Workshop On Adaptive Hypermedia And Adaptive Web-Based Systems*, AH2003, Hungary, 2003, pp. 113-120.

(Gallagher and Daigle, 2002) Gallagher, L., Daigle, J., "Creating Online Help", 2002: http://www.techcomplus.com/CreatingOnlineHelpHandoutWithTables.pdf. Accessed June, 2004.

(García, 2000) García, F. "CACTUS: Automated tutorial course generation for software applications", *Proceedings of the 5th International Conference on Intelligent User Interfaces*, USA, 2000, pp. 113–120.

(Gay, 1999) Gay, G. Document-centered Peer Collaborations: An Exploration of the Educational Uses of Networked Communication

Technologies. *In Journal of computer mediated communication,* JCMC 4 (3) March 1999. United States. 1999: http://www.ascusc.org/jcmc/vol4/issue3/gay.html. Accessed November, 2002.

(Gelernter, 1998) Gelernter, B., "Help design challenges in network computing", *Proceedings of the 16th annual international conference on Computer documentation,* Canada, 1998, pp. 184-193.

(Gerad, 2003) *Gerad, International Summer School on Multiple Criteria Decision (Aid 2003),* Canada, 2003: http://www.gerad.ca/mcda/en/mcda.php. Accessed August, 2003.

(Gerjets and Scheiter, 2003) Gerjets, P., Scheiter, K., "Goal configurations and processing strategies as moderators between instructional design and cognitive load: Evidence from hypertext-based instruction", *Educational Psychologist,* Vol. 38, Great Britain, 2003, pp. 33-41.

(Gilbert and Han, 1999) Gilbert, J. and Han, C. "Adapting instruction in search of a significant difference". In *Journal of Network and Computer Applications*, Volume 22, Issue 3, Austria, 1999, pp. 149-160.

(Gorell *et al.*, 2002) Gorrell,G., Lewin, I., Rayner, M., "Adding Intelligent Help to Mixed-initiative Spoken Dialogue Systems", *Proceedings of the 7th International Conference on Spoken Language Processing*, USA, 2002: http://www.fluencyvoice.com/docs/icslp_2002_intelligent_help.pdf.Acccesed June, 2004.

(Grandbastien, 1999) Grandbastien, M., "Teaching expertise is at the Core of ITS Research", *International Journal of Artificial Intelligence in Education,* Vol. 10, No. 3-4, UK, 1999, pp. 335-349.

(Grasha, 1996) Grasha, A., "Teaching with Style", *Alliance Publishers,* USA, 1996.

(Guzdial, 1999) Guzdial, M., "Supporting Learners as users", *Journal of Computer Documentation,* Vol. 23, No. 2, USA, 1999, pp. 3-13.

(Hartley, 2003) Hartley, R. "An interactive computer-based simulation environment for supporting and developing complex decision-making skills". In *Int. J. Cont.*

Engineering Education and Lifelong Learning, Vol 13, Nos. ¾, 2003, pp. 212-231.

(Heffernan, 2001) Heffernan, N. "Intelligent Tutoring Systems have Forgotten the Tutor: Adding a Cognitive Model of Human Tutors". *Dissertation. School Of Computer Science, Carnegie Mellon University*. United States. 2001

(Henze and Nejdl, 1999) Henze, N., Nejdl, W., "Student modeling in an active learning environment using Bayesian networks", *Proceedings of 7th Conference User Modelling*, Poster, Canada, 1999: http://www.cs.usask.ca/UM99/Proc/posters/Henze.htm. Accessed March, 2003.

(Herlocker *et al.*, 2004) "Evaluating collaborative filtering recommender systems", Herlocker, J., Konstan, J., Terveen, L. and Riedl, J. ACM *Transactions on Information Systems* (TOIS), vol. 22, Issue 1, 2004, pp. 5 - 53

(Höök, 2000) Höök, K., "Steps to take before IUIs become real", *Journal of Interacting with Computers,* Vol. 12, No. 4, Switzerland, 2000, pp. 409-426:

(Isakowitz *et al.*, 1995) Isakowitz, T., Stohr, E., Balasubramanian, P., "RMM: A Methodology for Structured Hypermedia Design", *Communications of the ACM,* V. 38", No. 8, USA, May 1995, pp. 34-44.

(Jerrams-Smith, 2000) Jerrams-Smith, J., "An Intelligent Human-Computer Interface for Provision of On-Line Help", *Artificial Intelligence Review* 14:, Kluwer Academic Publishers. Netherlands, 2000, pp. 5–22

(Johansson, 2001) Johansson, A. "Authoring Tools for Developing Intelligent Tutoring Systems". Dept of Information Science, Uppsala University. Norway. 2001 http://www.csd.uu.se/~alj/ATDITS.html. Accessed October, 2002.

(Johnson, 2001) Johnson, W., "Research in animated pedagogical agents: progress and prospects for training". May 2001. United States. 2001.

(Johnson and Taatgen, 2003) Johnson, A., Taatgen, N., "User Modelling", *The Handbook of Human Factors in Web Design,* 2003: http://www.ai.rug.nl/~niels/publications/usermodels.pdf. Accessed May, 2004.

(Johnson *et al.*, 2003) Johnson, W., Shaw, E., Marshall, A., LaBore, C., "Evolution of user interaction: the case of agent Adele", *Proceedings of the 8th International Conference on Intelligent user interfaces,* USA, 2003, pp. 93-100.

(Karsenti, 2004) Karsenti, T. "Prof branches. Un sondage révèle que 71% des formateurs emploient les TIC", In *Forum* V. 38, N. 31, 31 May, 2004. Canada, 2004, pp. 1-2.

(Keller, 2002a) Keller, F., "Introduction to Machine Learning", *Connectionist and Statistical Language Processing, Course at Universität des Saarlandes,* Germany, 2002: http://homepages.inf.ed.ac.uk/keller/teaching/connectionism/lecture8_4up.pdf. Accessed January 2004.

(Keller, 2002b) Keller, F., "Naive Bayes Classifiers", *Connectionist and Statistical Language Processing, Course at Universit¨at des Saarlandes,* Germany, 2002: http://homepages.inf.ed.ac.uk/keller/teaching/connectionism/lecture10_4up.pdf. Accessed January 2004.

(Keolle, 2002) Koelle, D., "Intelligent User Interfaces: An Independent Study Project", *Computer Science Department at WPI,* 2002: http://www.davekoelle.com/intint.jsp. Accessed June 2004.

(Kimiavi, 1998) Kimiavi, S., "ITS author: A framework for building hypermedia-based intelligent tutoring systems for the World Wide Web", *Dissertation from George Washington University,* USA, 1998.

(Kinshuk *et al.*, 2001) Kinshuk, Hong, H., Patel, A.,"Extended ITS Framework with Human Teacher Model", In C. H. Lee, S. Lajoie, R. Mizoguchi, Y D. Yoo & B. du Boulay (Eds.) *Enhancement of Quality Learning Through Information & Communication Technology (ICT), Proceedings of ICCE/SchoolNet 2001 Conference*), Korea, 2001, pp. 1259-1262.

(Kinshuk *et al.,* 1999) Kinshuk, Patel A., Russell D. "HyperITS: A Web-based Architecture for Evolving a Configurable Learning Environment". *Staff and Educational Development International Journal*, 3 (3), 1999, pp. 265-280.

(Kinshuk, 2003) Kinshuk, H., "State-of-the-art in Adaptive Learning: Keynote Address Pedagogies & Communication", *Proceedings of the International Conference on Open & Online Learning (ICOOL 2003), University of Mauritius,* 2003: http://icool.uom.ac.mu/2003/papers/file/keynote/kinshuk.zip Accessed March 2004.

(Kobsa *et al.*, 2001) Kobsa, A., Koenemann, J., Pohl, W., "Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships", *The Knowledge Engineering Review,* Vol. 16, No. 2, UK, 2001, pp. 111-155.

(Kobsa, 2001) Kobsa, A., "Generic User Modeling Systems", *User Modeling and User-Adapted Interaction,* Vol. 11, Netherlands, 2001, pp. 49-63.

(Koch, 1999) Koch, N., "A comparative study of methods for hypermedia development", Technical Report 9905, LudwigMaximilians –Universitt
Munchen, 1999:
http://www.dsic.upv.es/~west2001/iwwost01/files/contributions/NoraKoch/hypdev.pdf Accessed November, 2002.

(Koychev, 2000) Koychev, I. "Gradual Forgetting for Adaptation to Concept Drift". In *Proceedings of ECAI 2000 Workshop Current Issues in Spatio-Temporal Reasoning*, Germany, 2000, pp. 101-106.

(Koychev, 2002) Koychev, I., "Tracking Changing User Interests through Prior-Learning of Context", *Lecture Notes in Computer Science,* Vol. 2347, USA, 2002, pp. 223-232.

(Kumar, 2002) Kumar V., "Embedding human reasoning in soft computing", *International Conference on Hybrid Intelligent Systems* (HIS'02), Chile, 2002. http://www.sfu.ca/~vivek/personal/papers/HIS2002.pdf. Accessed June, 2004.

(Langley, 1999) Langley, P., "User Modeling in Adaptive Interfaces", *Proceedings of the 7^{th} International Conference on User Modeling UM99 Invited Speakers, Institute for the Study of Learning and Expertise,* USA, 1999: http://www.cs.usask.ca/UM99/Proc/invited/Langley.pdf. Accessed August, 2003.

(Lavesso, 2003) Lavesso, N., "Evaluation of classifier performance and the impact of learning algorithm parameters", *Master Thesis Software Engineering, Department of Software Engineering and Computer Science, Blekinge Institute of Technology,* Thesis no: MSE-2003:18, Sweden, 2003.

(Leung, 2001) Leung, M., "Towards a generic approach to providing proactive task support", *Ph.D. Dissertation, University of Waterloo,* Canada, 2001.

(Lieberman and Selker, 2000) Lieberman, H., Selker, T., "Out of Context: Computer Systems that Learn About, and Adapt to, Context", *IBM Systems Journal,* Vol. 39, No. 3-4, Germany, 2000, pp. 617-631.

(Lieberman, 1995) Lieberman, H., "Interaction is the key to machine learning applications", *Workshop on Machine Learning for Interactive Applications, International Conference on Machine Learning,* USA, July 1995: http://web.media.mit.edu/~lieber/Lieberary/AI/Interaction-Is/Interaction-Is.html. Accessed November, 2002.

(Light and Maybury, 2002) Light, M., Maybury, M., "Personalized multimedia information access", *Communications of the ACM,* Vol. 45, No. 5, USA, 2002, pp. 54 – 59.

(Liu, 2001) Liu, Y., "A Web-based instructional simulation tool for teaching procedural skills", *Dissertation from Utah State University,* USA, 2001.

(Lorés *et al.*, 2002) Lorés, J., Abascal, J., Cañas, J., Gea, M., Gil, A., Martínez, A., Ortega, M., Valero, P., Vélez, M., "Introducción a la Interacción Persona-Ordenador", *Electronic book,* Spain, 2002: http://griho.udl.es/ipo/descarga.html. Accessed August, 2003.

(Luger, 2002) Luger, G., "Artificial intelligence", Addison Wesley Logman, Inc. Fourth edition, USA, 2002.

(Lynch and Horton, 2001) Lynch, P., Horton, S., "Web Style Guide: Basic Design Principles for Creating Website", Electronic book, 2001: http://www.webstyleguide.com/type/lines.html. Accessed June, 2004.

(Maes, 1994) Maes, P., "Agents that Reduce Work and Information Overload", *MIT Media Laboratory,* USA, 1994:

http://pattie.www.media.mit.edu/people/pattie/CACM-94/CACM-94.p1.html. Accessed February, 2002.

(Matthews *et al.*, 2000) Matthews, M., Pharr, W., Biswas, G., Neelakandan, H., "USCSH: An Active Intelligent Assistance System", *Artificial Intelligence Review,* Vol. 14, Netherlands, 2000, pp. 121-141.

(Max and Lynn, 2003) Max, M., Lynn, J., "Symptom research: Methods and opportunities", 2003: http://symptomresearch.nih.gov/chapter_14/Part_2/sec6/chspt2s6pg1.htm. Accessed June, 2004.

(Maybury, 2000) Maybury, M., "Adaptive multimedia information access", *Proceedings of the 1st International Conference on Adaptive Hypertext (Ah00),* Invited talk, Italy, 2000: http://ah2000.itc.it/Maybury-AH2000.pdf. Accessed November, 2002.

(Maybury, 2003) Maybury, M., "Intelligent User Interfaces: An Introduction", *The MITRE Corporation,* January 2003: http://www.iuiconf.org/tutorial1.html. Accessed November 2002.

(Mayer and Moreno, 2003) Mayer, R., Moreno, R., "Nine ways to reduce cognitive load in multimedia learning", *Educational Psychologist,* Vol. 38, Great Britain, 2003, pp. 43-52.

(MCDM, 2003) *International Society on Multiple Criteria Decision Making,* 2003: http://www.mit.jyu.fi/MCDM/intro.html. Accessed October, 2003.

(Meisner, 2003) Meisner, E., Naive Bayes Classifier example, 2003: http://www.cs.rpi.edu/courses/fall03/ai/misc/naive-example.pdf. Accessed October, 2003.

(Millán, 2000) Millán E., "Sistema bayesiano para modelado del alumno", Tesis doctoral, Departamento de Lenguajes y Ciencias de la Computación, Universidad de Málaga, España, 2000.

(Miller, 1956) Miller, G. A. "The magical number seven, plus or minus two". *Psychological Review, 63,* USA, 1956, pp. 81-97.

(Miyahara and Pazzani, 2002) Miyahara, K., Pazzani, M., "Improvement of Collaborative Filtering with the Simple Bayesian Classifier", *IPSJ Journal,* Vol. 43, No. 11, Japan, November 2002: http://www.ics.uci.edu/~pazzani/Publications/IPSJ.pdf. Accessed May, 2004.

(Moreno, 2004) Moreno, R., "Decreasing Cognitive Load for Novice Students: Effects of Explanatory versus Corrective Feedback in Discovery-Based Multimedia", *Instructional Science,* Vol. 32, Netherlands, 2004, pp. 99-113.

(Murray, 99) Murray, T. (1999). "Authoring Intelligent Tutoring Systems: An analysis of the state of the art". In *International Journal of Artificial Intelligence in Education,* #10. Computer Science Dept., University of Massachusetts, Amherst & School of Cognitive Science, Hampshire College, Amherst. USA, pp 98-129.

(Murray, 2002) Murray, T., "Content Design Issues in Adaptive Hyperbooks", *International Journal of Computer Applications Technology on Designing Technology Supported Learning,* 2002: http://helios.hampshire.edu/~tjmCCS/papers/IJCAT2002ML/IJCAT2002.htm. Accessed March, 2003.

(Murray, 2003) Murray, T., "Applying Text Comprehension and Active Reading Principles to Adaptive Hyperbooks", *Proceedings of the 25th Annual Meeting of the Cognitive Science Society*, USA, 2003: http://www.ccm.ua.edu/pdfs/165.pdf. Accessed June, 2004.

(Netstrat, 2003) "Qu'est-ce que Netstrat? ", *Web site of the HEC Montréal,* 2003: http://cetai.hec.ca/netstrat/. Accessed October, 2004.

(Nkambou and Laporte, 2001) Nkambou, R, Laporte, Y. "Simulating emotional response for an intelligent tutoring system". Moore, J. D., *Proceedings of the 10th conference of Artificial Intelligence in Education. Artificial Intelligence in Education*. IOS Press.. Netherlands, 2001, pp. 568-570

(Olson and Olson, 2003) *Olson, G., Olson,J.,* "Human-computer interaction: Psychological aspects of the human use of computing", *Annual Review of Psychology*, Vol. 54, USA, 2003, pp. 491-516.

(Obraczka *et al.*, 1997) Obraczka K., Danzig, P., Wangpattanamongkol, K., "Finding Low-Diameter, Low Edge-Cost Networks", *Computer Science Department - University of Southern California - Tecnical Report,* May 1997. http://inrg.cse.ucsc.edu/katia-pubs/topology.pdf. Accessed August, 2004.

(Palu, 2004) Palu, S., "Instance-Based Learning: A Java Implementation", 2004: http://www.developer.com/java/article.php/1491641. Accessed March, 2004.

(Papanikolaou *et al.*, 2003) Papanikolaou, K., Grigoriadou, M., Kornilakis, H., Magoulas, G., "Personalizing the Interaction in a Web-based Educational Hypermedia System: The case of INSPIRE", *User Modeling and User-Adapted Interaction,* Vol.13, No. 3, Germany, 2003, pp. 213-267.

(Pass *et al.*, 2003) Pass, F., Renkl, A., Sweller, J., "Cognitive Load Theory and Instructional Design: Recent Developments", *Educational Psychologist,* Vol. 38, No. 1, Great Britain, 2003, pp. 1-4.

(Pass *et al.*, 2004) Paas, F., Renkl, A., Sweller, J., "Cognitive Load Theory: Instructional Implications of the Interaction between Information Structures and Cognitive Architecture", *Instructional Science,* Vol. 32, Netherlands, 2004, pp. 1-8.

(Patel *et al.*, 1998) Patel, A., Russell, D., Oppermann, K., Rashev, R., "An initial framework of contexts for designing usable intelligent tutoring systems", *Information Services and Use,* Vol. 18, No. 1-2, Netherlands, 1998, pp. 65-76.

(Pattinson and Dacre, 1998) Colin Pattinson Tony Dacre. "A network model for network management teaching", *Proceedings of the 3rd Australasian conference on Computer science education table of contents*, Australia, 1998, pp. 62 – 66.

(Payr, 2003) Payr, S., "The Virtual University's faculty: An overview of educational agents", *Applied artificial intelligence,* Vol. 17, Germany, 2003, pp. 1-19:

(Peterson and Peterson, 1959) Peterson,L.R., Peterson, M.J., "Short-term retention of individual verbal items". *Journal of Experimental Psychology*, 58, USA, 1959, pp. 193-198.

(Pierre and Elgibaoui, 1997) Pierre, S., Elgibaoui, A., "A Tabu Search Approach for Designing Computer Network Topologies with Unreliable Components", *IEEE Transactions on Reliability*, Vol. 46, No. 3, 1997, pp. 350-359.

(Pierre and Gharbi, 2001) Pierre, S., Gharbi, I., "A generic object-oriented model for representing computer network topologies", *Advances in Engineering Software*, Vol. 32, No. 2, January 2001, pp. 95-110.

(Pierre and Legault, 1998) Pierre, S., Legault, G.,"A Genetic Algorithm for Designing Distributed Computer Network Topologies", *IEEE Transactions on Man, Systems, and Cybernetics*, Vol. 28, No. 2, 1998, pp. 249-258.

(Pierre, 1998) Pierre, S., "Inferring New Design Rules by Machine Learning: A Case Study of Topological Optimization", *IEEE Transactions on Man, Systems, and Cybernetics*, Vol. 28A, No. 5, September 1998, pp. 575-585.

(Preece, 1993) Preece, J., "A guide to usability: Human factors in computing", *Don Mills, Addison Wesley*, England, 1993.

(Purchase and Worril, 2002) Purchase, H., Worril, J., "An empirical study of on-line help design: Features and principles", *Int. J. Human-Computer Studies*, Vol. 56, No. 5, 2002, pp. 539-567.

(Raskin, 2000) Raskin, J., "The humane interface: New directions for designing interactive systems", *Don Mills, Addison Wesley*, USA, 2000.

(Redmond-Pyle and Moore, 1995) Redmond-Pyle, D., Moore, A., "Graphical user interface design and evaluation (GUIDE)", *Prentice Hall*, USA, 1995.

(Rezazad, 2003) Rezazad, H., "An approach to the development of intelligent agents to assist with network configuration design problems", *PhD Dissertation, George Mason University*, 2003.

(Rickers *et al.*, 2004) Rickers, R., Gerven, P., Schmid, H., "Cognitive Load Theory as a tool for Expertise Development", *Instructional Science*, Vol. 32, Netherlands, 2004, pp. 173-182.

(Rodríguez *et al.*, 2003) Rodríguez, A., Aïmeur, E., Vázquez-Abad, F., "Training Teachers in Teaching Decision-Making Skills", *Proceedings of the 2nd*

International Conference on Multimedia and Information & Communication Technologies in Education, Spain, 2003, pp. 1964-1968.

(Rodríguez *et al.*, 2004a) Rodríguez, A., Aïmeur, E., Vázquez-Abad, F., "E-Learning for complex Decision-Making with the support of a Web-based Adaptive ITS", *Proceedings of the International Conference on Knowledge Engineering and Decision Support (ICKEDS'2004),* Portugal, 2004, pp. 47-54.

(Rodríguez *et al.*, 2004b) Rodríguez, A., Aïmeur, E., Vázquez-Abad, F., "Adaptive Help Techniques to Reduce the Teachers' Cognitive Overload", *Proceedings of the International Conference on Computers in Education (ICCE2004),* Australia, 2004, pp. 1741-1750.

(Rodríguez, 1997) Rodríguez, A. *Leguaje iconico orientado a objetos para especificar condiciones en Iyulú.* Tesis de maestría. Instituto tecnológico de Costa Rica. Costa Rica. 1997.

(Romero, 1996) Romero, C., "Análisis de decisiones multicriterio", Spain, 1996.

(Rothrock *et al.*, 2002) Rothrock, L., Koubek, R., Fuchs, F., Haas, M., Salvendy, G., "Review and reappraisal of adaptive interfaces: Toward biologically-inspired paradigms", *Theoretical Issues in Ergonomic Science,* 3(1), 2002, pp. 47-84.

(Russell and Norvig, 1995) Russell, S., Norvig, P., "Artificial Intelligence: A Modern Approach. Englewood Cliffs", *Prentice Hall,* USA, 1995.

(Russell and Norvig, 2002) Russell, S., Norvig, P. "Artificial Intelligence: a Modern Approach". Second Edition. Englewood Cliffs, NJ: Prentice Hall. United States. 2002.

(Schein *et al.*, 2002) Schein, A., Popescul, A., Ungar, L., Pennock, D., "Methods and metrics for cold-start recommendations", *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval,* Finland, 2002, pp. 253-260.

(Schwab *et al.*, 2000) Schwab, I. Pohl, W., Koychev, I. "Learning to Recommend from Positive Evidence", *Proceedings of International Conference on Intelligent User Interfaces,* ACM Press, USA, 2000, pp. 241 – 247.

(Selbach *et al.*, 2003) Selbach, M., Sieckenius, C., Barbosa, S., "A method of Semiotic engineering for the online help systems construction", *Proceedings of the Latin American conference on Human-computer interaction,* Brasil, 2003, pp. 167-177.

(Self, 1998) Self, T., "Implementing a Web-based Help System: Help for the Web Symposium", USA, 1998: http//:www.help4web.org/publication/selfl.htm. Accessed May, 2004.

(Shaffer, 1998) Shaffer, C., "A practical introduction to data structures and algorithm analysis", *Prentice Hall,* USA, 1998.

(Shaw *et al.*, 1999) Shaw, E., Ganeshan, R., Johnson, W.L., Millar, D. "Building a Case for Agent-Assisted Learning as a Catalyst for Curriculum Reform in Medical Education", In *Proceedings of the Int'l Conf. on Artificial Intelligence in Education*, USA, 1999: http://www.isi.edu/isd/ADE/ade.html. Accessed November, 2002.

(Shearin and Lieberman, 2001) Shearin, S., Lieberman, H., "Intelligent Profiling by Example", Proceedings *of the International Conference on Intelligent User Interfaces* (IUI 2001), USA, 2001, pp. 145-152.

(Shiri-Ahmadabadi, 1999) Shiri-Ahmadabadi, M., "Étude et modélisation des connaissances et raisonnement de l'apprenant dans un STI", *Thèse présentée à la Faculté des études supérieures en vue de l'obtention du grade de Philosophiae Doctor en informatique, Université de Montréal,* Canada, 1999.

(Shneiderman and Maes, 1997) Shneiderman, B., Maes, P., "Direct manipulation vs. interface agents", *Interactions,* Vol. 4, No. 6, USA, 1997, pp. 42 – 61.

(Spool, 1999) Spool, J., "Web site usability: A designer's guide", USA, 1999.

(Stephanidis, 2001) Stephanidis, C., "Adaptive Techniques for Universal Access", *User Modeling and User-Adapted Interaction,* Vol. 11, Netherlands, 2001, pp. 159-179.

(Stern *et al.* 1999) Stern, M., Beck, J., Woolf, B., "Naïve Bayes Classifiers for User Modeling", *Proceedings of the* 7^{th} *International Conference on User Modeling (UM'99),* Poster, Canada, 1999:

http://www.cs.usask.ca/UM99/Proc/posters/stern.htm. Accessed April, 2004.

(Stone, 1977) Stone, M., "Asymptotics for and against cross-validation". *Biometrika*, 64, 29-35.

(Susarla *et al.*, 2003) Susarla, S., Adcock, A., Van Eck, R., Moreno, K., Graesser, A., "Development and Evaluation of a Lesson Authoring Tool for Auto-Tutor", *Proceedings of the 11th International Conference on Artificial Intelligence in Education,* Australia, July 2003: http://www.cs.usyd.edu.au/~aied/vol6/vol6_VanEck.pdf. Accessed, December, 2003.

(Sweller, 1988) Sweller, J., "Cognitive load during problem solving: Effects on learning", *Cognitive Science,* Vol. 12, 1988, pp. 257-285.

(Syme *et al.*, 2003) Syme, A., Dickinson, A., Eisma, R., Gregor, P., "Looking for help? Supporting Older Adults' Use of Computer Systems", *Human -Computer Interaction,)* (ed. M. Rauterberg, M. Menozzi and J. Wesson), Switzerland, 2003, pp.924-931.

(Taatgen, 1999) Taatgen, N. "Learning without limits: from problem solving towards a unified theory of learning", Ph.D. Dissertation in Computer Sciences, University Groningen, Germany,1999.

(Towle, 2001) Towle, B. "Authoring tools for building how-to simulations". Ph.D. dissertation, Northwestern University, USA. 2001.

(Tsiriga and Virvou, 2002) Tsiriga, V., Virvou, M., "Dynamically Initializing the Student Model in a Web-based Language Tutor", *Proceedings of the 1st International IEEE Symposium Intelligent Systems (IS 2002),* Vol. 1, 2002, pp. 138-143.

(Tsuji and Yamamoto, 2001) Tsuji, A., Yamamoto, Y., "A Framework to Provide Integrated Online Documentation", *Proceedings of the 19th Annual International Conference on Computer Documentation*, USA, 2001, pp. 185–192.

(Turban and Aronson, 1998) Turban, E., Aronson, J., "Decision support systems and intelligent systems", Firth edition Prentice-Hall, USA, 1998.

(Van Joolingen, 1999) Van Joolingen, W., "Cognitive Tools for discovering learning", *International Journal of artificial intelligence in Education,* 1999, pp.385-397.

(Vázquez-Abad *et al.*, 2001) Vázquez-Abad, F., Rodríguez, A., Ng, A., Zukerman, M., Warfield, R., "Bridging theory and practice in network design using web-based simulation", *Proceedings of the 12th Annual Conference of the Australasian Association for Engineering Education (AaeE 2001),* Australia, 2001, pp. 40-45.

(Vázquez-Abad *et al.*, 2003) Vázquez-Abad, F., Rodríguez, A., Aïmeur, E., "Training the Teacher: A New Approach for Authoring ITSs for Teaching Decision-Making", *Proceedings of the International Conference on Open & Online Learning (ICOOL 2003), University of Mauritius,* 2003: http://icool.uom.ac.mu/2003/papers/file/Rodriguez.pdf. Acessed May, 2004.

(Virvou and Du Boulay, 1999) Virvou, M., Du Boulay, B., "Human Plausible Reasoning for Intelligent Help", *User Modeling and User-Adapted Interaction,* Vol. 9, No. 4, 1999, pp. 321-375.

(Virvou and Kabassi, 2002) Virvou, M., Kabassi, K., "Intelligent Help in a Graphical User Interface", *Proceedings of the 2002 IEEE International Conference on Systems, Man and Cybernetics*. 2002, pp. 170-175.

(Virvou and Mondriduo, 2002) Virvou, M., Moundridou, M., "Adding an instructor modelling component to the architecture of ITS authoring tools", *International Journal of Artificial Intelligence in Education,* Vol. 12, 2002, pp. 185–211.

(Virvou and Moundridou, 2001) Virvou, M., Moundridou, M., "Student and Instructor Models: Two Kinds of User Model and their Interaction in an ITS Authoring Tool", *Proceedings of the 8th International Conference,* UK, 2001, pp. 158-167.

(Vliet, 2000) Vliet, V., "Software Engineering: Principles and Practice", *John Wiley & Sons Ltd.,* England, 2000.

(Wærn and Stenborg, 1995) Wærn, A., Stenborg, O., "Simplistic Approach to Keyhole Plan Recognition", *Technical Report SICS-T-95/01-SE, Swedish Institute of Computer Science,* Sweden, January 1995: http://www.sics.se/~annika/IJCAI.ps.Z. Accessed June, 2004. Accessed October, 2004.

(Wahlster, 2000) Wahlster, W., "Intelligent Multimedia Interface Agents", *Proceedings of the International Conference on Artificial Intelligence 2000,* Mexico, 2000: http://www.dfki.de/~wahlster/micai_tutorial/. Accessed February, 2002.

(Wang, 2001) Wang, J., "Toward the usability of hypermedia adaptive intelligent interfaces", *PhD. Dissertation, The George Washington University,* 2001.

(Warren, 1996) Warren, L., "Intelligent Tutoring Systems", *Mitre Organization,* USA, 1996: http://www.mitre.org/resources/centers/it/g068/its.html. Accessed September, 2002.

(WebCT, 2002) http://www.webct.com/entrypage. Accessed March, 2002.

(Webb *et al.*, 2001) Webb, G., Pazzani, M., Billsus, D., "Machine Learning for User Modeling", *User Modeling and User-Adapted Interaction,* Vol. 11, Netherlands, 2001, pp. 19-20.

(Weber, 1998) Weber, M. "*Architecture for a Web-based intelligent tutoring system for SPICE",* Dissertation, USA. 1998.

(Weber, 2000) Weber, J., "Editing Online Help", *WeberWoman'sWrevenge,* 2000, pp. 155.

(Welinske, 1998) Welinske, J. "A study of how user assistance is employed in Web sites", *Help for the Web Symposium*, Position Paper. USA. 1998: http://www.help4web.org/publication/j_welin.htm. Accessed May, 2004.

(Williams, 2001) Williams, B. "The Role of External Representations in Intelligent Tutoring System authoring: Supporting localised decision-making in a complex and evolving global context". *AIED'2001 Workshop External representations in AIED: Multiple forms and multiple roles.* USA. 2001: http://www.psychology.nottingham.ac.uk/research/credit/AIED-ER/williams.pdf. Accessed December, 2002.

(Wittig, 1999) Wittig, F., "Learning Bayesian Networks with Hidden Variables for User Modeling", *Proceedings of the 7th International Conference on User Modeling (UM'99),* Canada, 1999, pp. 343-344.

(Wooldridge, 2002) Wooldridge, M., "An introduction to Multi-Agent Systems", *John Wiley & Sons Ltd.,* England, 2002.

(Wu *et al.*, 1998) Wu, H., Houben, G., De Bra, P., "AHAM: A Reference Model to Support Adaptive Hypermedia Authoring", *Proceedings of the Zesde Interdisciplinaire Conferentie Informatiewetenschap,* Netherlands, 1998, pp. 77-88.

(Wu, 2002) Wu, H., "A reference architecture for adaptive hypermedia applications", PhD. Dissertation, *Universiteit Eindhoven,* Netherlands, 2002.

(Weiss-Lambrou and Raymond, 2002) Weiss-Lambrou, R., Raymond, D., »*Rapport du sondage sur: L'utilisation de WebCT à l'Université de Montréal et l'appréciation du soutien offert par le programme SUITE* », Centre d'études et de formation en enseignement supérieur, Université de Montréal, 2002: http://www.cefes.umontreal.ca/Documents/CEFES_SondageA01_Resultats.pdf. Accessed October, 2003.

(Woolf, 2000) Woolf, B. *Growth and maturity of intelligent tutoring systems. A status report.* In Forbus, K. and Feltovich, Paul. *Smart machines in education: the coming revolution in educational technology.* MIT Press. United States. 2000.

(Yacef, 2004) Yacef, K. "Making large class teaching more adaptive with the logic-ITA", *Proceedings of the sixth conference on Australian computing education*, Volume 30, New Zealand, 2004, pp. 343 – 347.

(Yu *et al.*, 2003) Yu, T., Min, R., Spenkelink, G., "E-Learning Environments on the world wide web, based on the Concept of Parallelism", *Proceedings of the EARLI Conference,* Italy, August 2003: http://wwwis.win.tue.nl/~acristea/HTML/Minerva/papers/report4jOfCompAssLearning-YuMinSpenkelink.doc. Accessed March, 2004.

(Yu, 2002) Yu, T., "Empirical study to cognitive load and the PI theory with well-designed products for procedure skills and parallel instructions", Final Project of the Master Program: Educational and Training System Design (ETSD), August 2002: http://projects.edte.utwente.nl/pi/Papers/indexYu.html. Accessed July, 2004.

Annex A. List of publications

1. Vázquez-Abad, F., Rodríguez, A. Ng, A., Zukerman, M., Warfield, R., "Bridging theory and practice in network design using web-based simulation", *Proceedings of the 12th Annual Conference of the Australasian Association for Engineering Education (AaeE 2001)*, Australia, 2001, Pp. 40-45.
2. Vázquez-Abad, F., Rodríguez A., Aïmeur E., "Training the Teacher: A New Approach for Authoring ITSs for Teaching Decision-Making", *Proceedings of International Conference on Open & Online Learning (ICOOL 2003)*, Mauritius, 2003.
3. Rodríguez, A. Aïmeur, E. Vázquez-Abad, F., "Training Teachers in Teaching Decision-Making Skills", *Advances in technology-based education: towards knowledge based society. Proceedings of the II International Conference on Multimedia and Information & Communication Technologies in Education (MICTE 2003)*, Spain, 2003, Pp. 1964-1968
4. Rodríguez, A. Aïmeur, E., Vázquez-Abad, F., "E-Learning for complex Decision-Making with the support of a Web-based Adaptive ITS", *Proceedings of the International Conference on Knowledge Engineering and Decision Support (ICKEDS 2004)*, Portugal, 2004, Pp. 47-54.
5. Rodríguez, A., Vázquez-Abad, F., Aïmeur, E., "Adaptive Help Techniques to Reduce the Teachers' Cognitive Overload", *International Conference on Computers in Education* (ICCE 2004), Australia, 2004, (accepted paper).

Annex B. Questionnaire

This is the questionnaire used to evaluate the acceptance of the help approach that we studied in our research. We used 25 questions related to problem-solving support and leaning support applied in our system ARIALE (Authoring Resources for Implementing Adaptive Learning Environments). These questions were asked to 27 teachers who teach different disciplines of the Universidad de Costa Rica in San Jose, Costa Rica. The teachers were selected from a list provided by the Universidad de Costa Rica. Each teacher has built a Web-site to support at least one course in 2005. The teachers were interviewed personally in meetings that lasted between 20 up to 120 minutes. The questionnaire also included 13 additional questions related to general information of each teacher, such as name, age and experience teaching a particular subject.

General Information

Name:____

Grade:

_Bach. _Lic[1]. _Master _PhD

Condición laboral:

_Lecturer. _Staff.

Age

_ ≤ 30 _> 30 y ≤ 55 _> 55

Gender.

_ Female _ Male

Career or expertise area.

_ Decision-making. _ Network Design. Another__________

[1]Lic. is a "licenciatura", a grade that is a speciality in a particular area.

To which program do the courses you are teaching belong?

_ Decision-making. _ Network Design. Another__________

How long have you been teaching in this area?

_ Less than a year. _ 1 to 5 years. _ More than 5 years.

What is your self evaluation of your experience to teach your courses?

_ Beginner. _ Intermediate. _ Advanced.

Times you have taught this course.

_ Never. _ 1 to 5 times. _ More than 5 times.

What is your self evaluation of your skills using computers?

_ Beginner. _ Intermediate. _ Advanced.

What is your self evaluation of your experience using WBT for teaching?

_ Beginner. _ Intermediate. _ Advanced.

What is your self evaluation of your experience using your Web-site for teaching?

_ Beginner. _ Intermediate. _ Advanced.

Questions about our help approach

1-How long do you have to create the teaching material for your course?

_More than enough. _Enough. _Barely suffices.

_Insufficient. _There is no time.

2- You reuse examples, exercises or cases created by other teachers.

_Agree very much. _Agree.

_Partially agree. _Totally disagree (skip to 4).

3-You accept to reuse teaching materials that are:

_Different from yours.

_Exactly similar to the ones that you use.

_Relatively similar to the ones that you use.

_Similar to the materials that other teachers teaching your course use.

4-You share your teaching materials with other teachers.

_Agree very much. _Agree.

_Partially agree. _Totally disagree (skip to 6).

5-You prefer to share teaching materials only with professors who:

_teach the same course.

_teach similar contents.

_apply the same method to teach.

_use examples, exercises or cases that are similar to yours.

6-Do you accept that a computer recommends teaching materials?

_Yes (continue) _No (skip to 8)

7-An application can use your personal data about preferentes relaing to teaching materials to recommend examples, exercises or cases that are to yours.

_Agree very much. _Agree.

_Partially agree. _Totally disagree.

8-Do you use exercises, examples or cases that can be generated automatically?

_Yes (continue) _No (skip to 13)

9-What is the percentage of your labour time that you spend creating examples, exercises or cases that can be generated automatically?

_10% _20% _30% _40% _50%

10- You accept that software generates examples, exercises or cases automatically, according to your preferences to let you to use or reject those examples.

_Agree very much. _Agree.

_Partially agree. _Totally disagree.

11-You accept that software takes your examples, exercises or cases to generate similar ones.

_Agree very much. _Agree.

_Partially agree. _Totally disagree.

12- Automatic generation of examples, exercises or cases, and the recommendation of them according to your preferences, reduces the time required to author teaching materials.

_Agree very much. _Agree.

_Partially agree. _Totally disagree.

13- 1-How long do you have to learn the use of software required to author teaching materials?

_More than enough. _Enough. _Barely suffices.

_Insufficient. _There is no time.

14- What problems have you faced when you have been using software to create digital teaching materials?

_Software is complex.

_Lack of time to learn to use software.

_Lack of previous training to use software.

_Lack of technical suppoprt.

_Lack of online help.

_Other (explain)____

15- What do you prefer to learn the use of software required to autor digital teaching materials?

_Ask help to other teachers.

_Formal training or a course.

_To access online help that the application includes.

_To access a wizard or tutor that shows how to use the application.

_Help from technical support.

_To access online help adapted to your preferences and needs.

_Check a printed manual.

_Try and error until mastering the application, but with no human nor other support.

_Check books or articles on Internet.

16- Learning by doing is the best way for a university teacher to learn the use of software required to crate teaching materials.

_Agree very much. _Agree.

_Partially agree. _Totally disagree.

17- You accept that software evaluates your progress learning the use of software to create digital teaching materials.

_Agree very much. _Agree.

_Partially agree. _Totally disagree.

18- With what frequency have you used help included in software to create digital teaching materials?

_Never (skip to 22) _Seldom _Sometimes

_Frequently _Always

19-What problems do you face when you are using help included in software to create digital teaching?

_Help contents are too long.

_You do not know the appropriate key words to find help.

_You are missing in the structure of the help contents.

_Help contents are not related to the task on hands.

_Help contents are not precise.

_There are not examples about how to use the program.

_Help is not context-sensitive.

_Help is just text, with no images or animations that accelerate understanding the application.

20- When you are accessing help, you prefer:

_Find information by yourself.

_An agent or wizard to solve the problem.

_Software displays available help to allow you to make you own decisions.

_Software displays information step by step, starting with short and precise information, and showing more details if ask for them.

21- When you are accessing help, you prefer to receive information about:

_All functions available in the application.

_Only the function you are using currently.

_All functions related to the function you are using currently.

_All functions associated with the task you must perform.

_Only the functions that are useful to finish the task that you must perform.

22- You prefer to receive help adapted to your knowledge of the application that you are using, instead of standard help.

_Agree very much. _Agree.

_Partially agree. _Totally disagree.

23- You prefer to receive help by the media that you like, namely only text, sound or video, in a personalized manner.

_Agree very much. _Agree.

_Partially agree. _Totally disagree.

24- How do you prefer to receive help o when you are using software?

_On a specific area of the same window where you are working, to avoid stopping the task you are performing.

_In another independent window.

_In a tooltip that pops up near to the concept you are looking help about.

25- A context-sensitive and personalized help that displays on the same window you are working reduces the time needed to learn authoring teaching materials.

_Agree very much. _Agree.

_Partially agree. _Totally disagree.