

AN INTEGRATED AND CUSTOMISABLE SUPPORT SYSTEM FOR BUILDING HYPERMEDIA TRAINING APPLICATIONS¹

Aphrodite TSALGATIDOU, Zissis PALASKAS*, Constantin HALATSIS, Michael HATZOPOULOS

Dept. of Informatics, University of Athens

Panepistimiopolis, TYPA Buildings, Ilisia157 71 Athens, Greece

email: {afrodite, halatsis, mike} @ di.uoa.gr

ABSTRACT

The multimedia platforms which are currently available are not suitable for the development of hypermedia training applications because they lack features dedicated to support the specific task of training. HTAS is an integrated and customisable support system for building and executing hypermedia training applications. by adopting the Systematic Approach to Training (SAT) methodology [14] and the Hypertext Design Model HDM [9]. This system also provides a set of tools which support the construction of a generic application schema and a schema instance which defines a particular application. An innovative feature of HTAS is the customisation of the individual courses according to the individual trainees' profiles using appropriate knowledge based tools. Distance learning support is also provided by HTAS.

Keywords: Hypertext, Hypermedia, Computer Based Training (CBT), Hypermedia Training Applications, Hypertext/Hypermedia Design Model, Trainee Modelling

1. Introduction

Training as an activity is a methodological process that takes advantage of the pedagogical experience and expertise of human teachers. Computer based training (CBT) systems, exploiting modern multimedia technology, may prove to be much more effective, compared to current self-training techniques such as studying manuals, instruction booklets etc.

Today, there is a proliferation of multimedia platforms which can be used for developing applications for education, recreation, tourism, publishing etc. However, a common characteristic of such platforms is the lack of dedicated features to support specific tasks such as training. Developers of training packages have to use general purpose multimedia platforms and thus, they face a number of problems, such as implementing training specific features on a general purpose multimedia platform, a fact that in turn imposes an additional overhead to the overall cost of developing an application. Other problems may be related with porting the

¹*In Procs. of the 5th Conf. On Applications of Databases and Expert Systems (DEXA-94), Athens, Sept, 1994, Springer-Verlag, pp. 540-549.*

* 01-PLIROFORIKI S.A., 438 Aharnon str., Kato Patisia, Athens, Greece, email: zpalas@isosun.ariadne-t.gr

developed application to other development platforms and/or modifying it to satisfy different training requirements.

This paper presents the design of an integrated and customisable application support system for building Hypermedia Training Applications (the HTAS system). HTAS is based on existing and widely available technologies for authoring and adapts existing methodologies for structuring hypermedia information. The goals of the HTAS project are to:

- * investigate and implement knowledge based techniques for hypermedia information classification
- * define a methodology for developing hypermedia training applications
- * develop an efficient and cost effective platform for building hypermedia training applications.
- * test and evaluate the HTAS system capabilities through the development of application exemplars.

The following section describes the HTAS system architecture. Section 3 discusses the methodology and the design model used for developing training applications. Distance learning support is another facility offered by HTAS and is analysed in section 4. The HTAS platform is validated using an application pilot discussed in section 5. Section 6 refers to work related to HTAS and section 7 concludes this work.

2. The HTAS architecture

Figure 1 depicts the HTAS system architecture. The first level of this architecture shows the *authoring environment*, which is divided in two parts: the *authoring-in-the-large* part, which refers to the design and specification of the global aspects of a hypermedia application, and the *authoring-in-small* part which refers to the development of the contents of the hypermedia nodes. This terminology is used in HDM [9], which is a Hypertext Design model that is adopted by HTAS and is briefly described in section 3. The specification of a hypermedia application consists of three parts: the specification of training application domain, the specification of the training scenarios and the modelling of the trainee, which are stored in a multi-purpose hypermedia *training application repository*. These specifications are constructed by an author using respectively the three specification tools which can be seen in the *authoring-in-the-large* environment, i.e. the *training application specification tool (TAS tool)*, the *training scenarios specification tool (TSS tool)* and the *trainee modelling tool (TM Tool)*.

It is worth to stress here the importance of the modelling of a trainee. Training applications are systems serving diverse groups of users which require different modes of interaction according to their roles and levels of knowledge. Most of the time, user behaviour does not only depend on the sole knowledge of an application domain but also, on integrated knowledge from many related application domains. The *TM tool* of HTAS captures behaviour of classes of users over training sessions, and adjusts the level of training according to a user's expertise on domain

concepts. The relationships between the application depended training concepts and the corresponding user profiles are captured early in the application specification phase and are stored in the *training application repository*. Also, in the same repository are stored the procedures and the structures required for the production of customised courses with optional topics and emphasis levels.

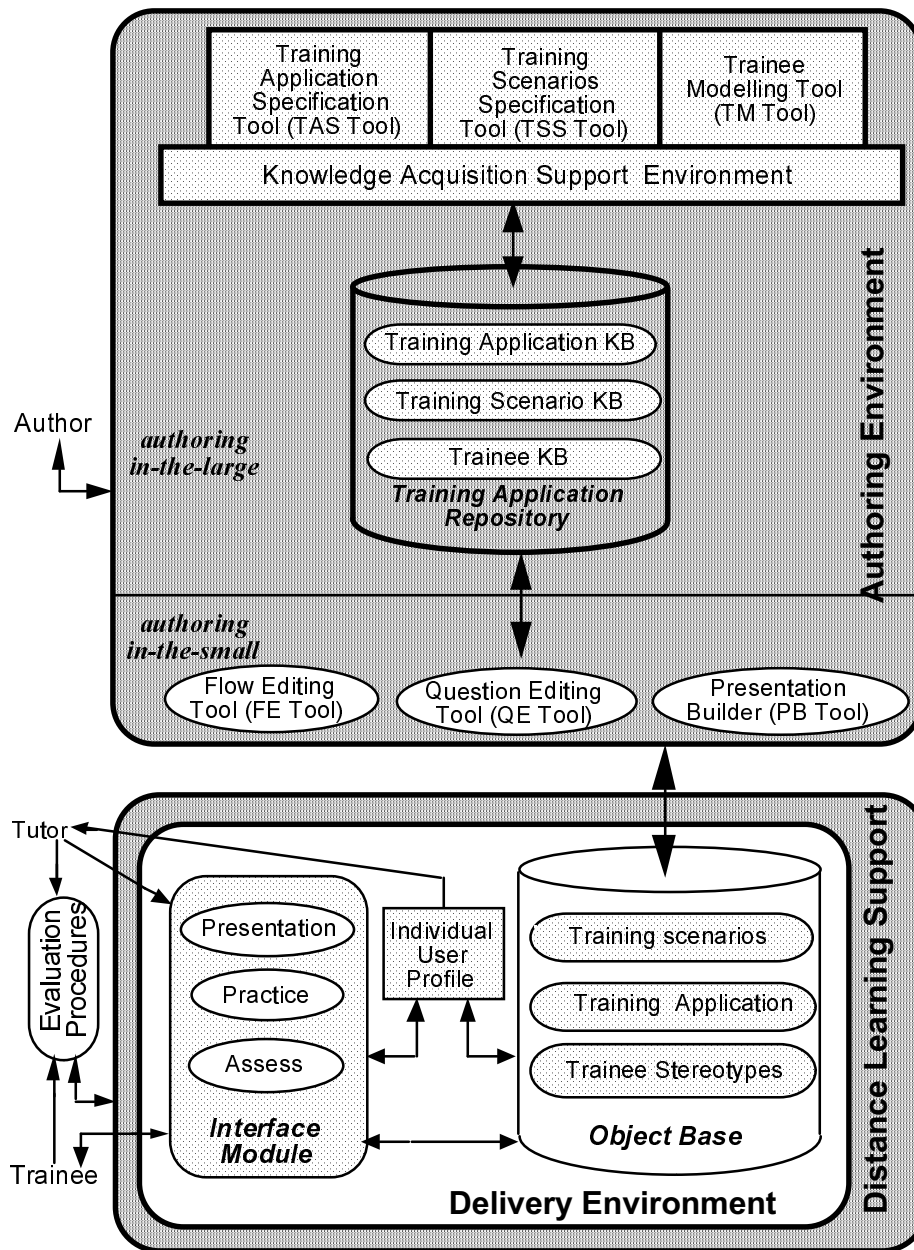


Fig. 1 The HTAS architecture

A hypermedia training application is generated by an author who takes as input the generic specification of an application stored in the *training application repository* and generates the target application using a set of intelligent tools which allow him/her to specify the functionality and the user interface of the application. The tools which support this *authoring-in-the-small*

process also support flow controll editing, question generation and presentation of material. The corresponding tools are the *flow editing tool (FE tool)*, the *question editing tool (QE tool)* and the *presentation builder tool (PB tool)*.

The *FE tool* enables an author to create customised courses and control the flow of the information presented. Thus, this tool supports the definition of the flow of the course in a simple and interactive manner and also manages the presentation of questions and exercises. The *QE tool* supports the generation of questions as individual entities which can be used several times in different contexts. The *PB tool* supports the creation of a sequential time-based presentation. Different items coming from a media folder can be put on a time scale and the synchronisation problems are resolved by the system.

The resulting hypermedia training system consists of an *object base*, which has three parts: a *training scenarios* object base, a *training application* object base and a set of *trainee stereotypes*. It also contains a knowledge base of *individual user profiles* for customising training scenarios to individual user needs. The interaction between the user and the system is carried out through an *interface module*. This module is responsible for the *presentation* of information, the *practise* of students and their *assessment*. The user interface usually consists of a set of screens and dialogue structures. More specifically, the *training application* object base describes the application domain in terms of objects and links between objects thus, describing the domain knowledge necessary to support the training requirements. The *training scenarios* object base describes the training requirements for individual domain objects as well as the possible combinations of them to define a complete training session. The system is populated with a set of *trainee stereotypes* which model the most frequent trainee types of the system. These stereotypes are captured during application specification and they are further extended and adapted based on the knowledge captured for individual users. The latter is the responsibility of a course *tutor* who uses the *individual user profiles* knowledge base that reads in *interaction information* for a particular user and then customises the training scenarios to maximise training quality and minimise training time for each particular user.

Thus, the users in the delivery environment are the *tutors* and the *trainees* who interact with it for different reasons. The *trainee's* goal is to learn and the *tutor's* goal is to adjust the application's interface according to the profile of an individual trainee, to assist trainees and to monitor their progress. Tutors and trainees are also considered as part of the *evaluation procedure* the aim of which is to evaluate the success of the overall course and adjust it according to the evaluation results. Tutors and trainees contribute in the *evaluation procedure* by giving answers to questionnaires, by making remarks on the course, by participating in interviews, etc. In fig. 1, it can be seen that tutors and trainees interact with the delivery environment through the distance learning support provided by HTAS. This latter feature of HTAS is analysed in section 4.

3. Methodology and Design Model for Developing Hypermedia Training Applications

Training is the activity which aims to increase people's performance by teaching them the rules and procedures of a knowledge domain. Computer Based Training (CBT) is a form of a self paced training, but in order to be better accepted by the trainees, the computer courses have to be well designed and the new possibilities offered by technology have to be exploited. The development of team based design for CBT courses must address the phases of the learning process, namely it should:

- * arouse trainee's interest by using graphics, opening questions etc.
- * present an advanced organiser including orientation screens with key concepts of the modules, window based explanations and user manual.
- * identify the necessary pre-knowledge which should be implemented as introduction screens.
- * present the actual learning material using advanced presentation means for holding a trainee's interest.
- * practise and apply the learned material, with multiple choice questions on the theory of the modules and with cases studies.
- * evaluate and test the level of trainee's proficiency with test cases
- * identify and remedy the misconceptions
- * perform final test cases with integration of the concepts between different modules

It is obvious that, in order to tackle the complexity of developing hypermedia CBT and produce well designed training systems, a methodological approach is required to analyse the training requirements and identify the training problems. A Systematic Approach to Training (SAT) [14] proposes a five stage activity mapping of a training process. These stages may be followed by an author of HTAS and are shown in fig. 2 together with the respective HTAS tools supporting each stage. More specifically, these stages are:

- * *Application Domain Analysis*. This activity involves the definition of the application procedures and tasks, the skills and the knowledge needed to complete these tasks and the job objectives; i.e. what the people have to perform in order to complete a job, the context in which this performance will be achieved, the standard of achievement required for the job. This stage is supported by the *TAS tool* of HTAS.
- * *Training Specifications*. This stage takes a pragmatic view of the training required and relates it to what it is achievable by performing a cost-benefit analysis. It specifies the training requirements of the application domain in terms of training tasks, contents of the training, performance conditions and standards. The *TSS* and *TM* tools of HTAS support an author during this stage.
- * *Course Design*. Based on the training objectives, this stage decides the material of the course, the sequence of the material, the series of courses, the sections to be included, the section composition, the selection of section material and the media that this material will be presented upon (e.g. screen designs etc.). This stage is mainly supported by the HTAS tools

that are found in the *authoring-in-the-small* part of the HTAS environment, namely the *FE*, *QE* and *PB tools*.

- * *Course Implementation*. This stage involves the actual implementation of a designed course, as well as the course execution. During the course execution, data which store the trainee achievements and the cost effectiveness of the course are maintained. The HTAS tools used for the previous stage, support the actual course implementation. Other tools, external to the HTAS environment, are also used for capturing raw data which are then incorporated in the *object base* of generated the application. The *interface monitor* of the delivery environment of HTAS support the execution of the implemented course.
- * *Course Assessment*. This phase attempts to identify the strengths and the weaknesses of a course. Thus, it establishes that the course is achieving the initial goals and it satisfies the course objectives. The outcome of this stage is fed back to the training application domain analysis to improve new versions of the course. The *Evaluation Procedure* is the component of HTAS that supports this task.

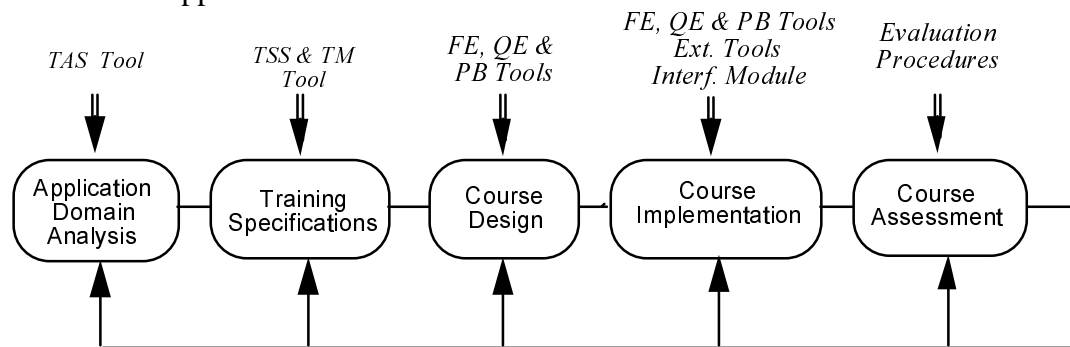


Fig. 2 The SAT stages for the development of a training application and the respective HTAS tools supporting each stage.

The SAT identifies the following training concepts: (a) *objectives* which describe what has to be achieved, how well an objective is to be achieved and under which context, (b) *enabling objectives* which are items of knowledge or skills that enable a trainee to achieve higher level objectives and (c) *teaching points* which are items of information that the learner must acquire.

A number of factors, like the method of presentation, the medium of presentation, the time allocated to the training material, or the sequence of the training material, may influence the training quality of a CBT. The sequence that a material is presented to a trainee must conform to the course's objectives. Hypermedia offers a trainee the possibility to navigate through a web of information. A disciplined navigation can be imposed by an intelligent training system to the benefit of the training process. The sequencing of material can be defined so that: (a) general descriptions precede specific ones, (b) concepts required for the understanding of other concepts are presented first and (c) topics which may repeat as contents in the objectives of a set of lessons may be externalised, and form a set of autonomous prerequisites to the lessons.

It is noted here that, when learners are free to select their own way of learning strategies, they do not always select wisely. Research has shown [12] that when learners are given control over instructional variables, they do not make the best decisions. This problem may be alleviated in systems that allow the explicit definition of the instructional strategies by the authors. Instruction is a goal-oriented teaching process that is based on pre-planning and formative evaluation [15]. Instructional systems must support argumentation and rhetorics.

The knowledge based approach to describing the training process may model user's behaviour as one's ability to navigate in the hypermedia training web. In HTAS, this behaviour may be expressed as a set of rules with pre-conditions and post-conditions in the knowledge base schema. The *enabling objectives* correspond to the pre-conditions of the rules which capture the training *objectives* of a course.

Apart from a design methodology, what is needed for the design of a hypertext application is a design model. There are a number of design models used in hypermedia development which are analysed in section 6. The conceptual model of HTAS supports a semantic-oriented description of the application components. HTAS provides an application oriented model for the capturing of the training requirements. HTAS emphasises in the methodological development of hypermedia systems and adopts the Hypertext Design Model (HDM) [9]. HDM describes the structure of the training concepts in terms of the entities populating the application, their features and their interrelations. HDM assumes an *authoring-in-the large* approach. This method facilitates the design of the structural and semantic aspects of hypertext/hypermedia applications in a system independent manner, and without much concern on the node contents and the node lay-out. HDM provides a set of modelling primitives for authoring-in-the-large, which are adopted by HTAS.

The methodology of HTAS follows the HDM paradigm and allows the description of hypermedia applications in terms of a *schema*, an *instance of a schema* and the *access structures*. A *schema* is a collection of type definitions that describe an application at the global level. Thus, a schema defines the structural, the navigational, and the semantic properties of a class of applications. The *training application repository* in the authoring environment of HTAS corresponds to the *schema* notion of HDM. An *instance of a schema* defines the actual information structures of a specific application according to the schema prescriptions. The *object base* in the delivery environment of HTAS corresponds to the notion of the *instance of a schema* of HDM. The *access structures* define user-oriented entry points for directly accessing information structures in an instance of a schema and navigating the application along predefined paths. This is very useful since the definition of navigational properties of the contents of a course, i.e. how information browsing is facilitated in a hypermedia CBT, is an important issue in the development of hypermedia training applications. In HTAS, the *access structures* are incorporated in the *training application repository* of the authoring environment, as well as in the *object base* of the delivery environment.

It is clear now that, in the HTAS environment, authors develop hypermedia training applications by following the SAT methodology and constructing a HDM model. They are assisted by appropriate tools during the specification and design of a global model of a class of applications which is stored in the *training application repository*. Then, they exploit the information stored in this repository by using it in the development of various applications on a specific field, using appropriate tools. Special purpose browsers let authors to interactively select and retrieve the required information from the repository. Other platform tools enable authors to capture an application definition in terms of instantiated application frames (*schema instances*) from the information repository. These frames are derived by generic application templates (part of the global *schema*) which describe the common features of specific classes of applications. Meta-level descriptions of training sessions give the content of the information nodes and the sequences of the information presented to the trainees. The above relationships can be entered into the training system only when a thorough understanding of the application has been achieved.

According to the *authoring-in-the-large* approach, the primitives selected are totally independent from specific development/delivery environments, used for the applications, thus the same description of an application may be compiled towards different hypertext/hypermedia environments (e.g., ToolBook™, HyperCard™). The tools of HTAS allow hypermedia applications to be designed and developed at a higher level than node-link level, and in a system independent manner.

4. Distance Learning Support

Distance learning technologies enable learners to follow courses from places remotely located from the organisations offering the training. Therefore, the conventional classroom setting is being replaced by a distributed learning environment. Distance learning allows a learner to follow up a class with a CBT program either at a workplace or in the classroom and to receive assistance, when needed by a tutor. Additionally, the communication infrastructure may let individuals communicate with other trainees and work in groups by conferencing or by using electronic mail. The principal aim of the distance learning component of the HTAS architecture is to enable a training organisation to perform centralised monitoring and follow up of the training process with respect to the progress of the trainees.

The functionality of the telecommunications based distance training approach of HTAS may be described as follows:

- * one way satellite communication may be used to deliver certain kinds of educational material (e.g. video) cheaper than conventional means (e.g. post)
- * conventional data transmission may distribute software or text based material. The high volume training material may be available on CD-ROM and the telecommunication network will be available for updating the information and for logging the training transactions of the trainees. Narrow-band ISDN will improve the economics of this operation.

* Telecommunications may support human to human interactivity, electronic mail and simple computer conferencing. ISDN can provide extra opportunities.

The communications architecture (see fig. 3) in HTAS recognises three distinct kinds of stations, namely; the *Author's* station, the *Tutor's* station and the *Trainee's* station. The progress of an individual trainee is described as changes in a central historical database which holds the profiles of the trainees. Changes in the *Trainee* database are recorded as transactions initiated by the interactions of individual trainees, in terms of topics attended, sequences of material browsed, questions visited, answers to questions, performance to given tests, etc. A *Tutor* who has been assigned to a number of trainees, may monitor the performance and the progress of the individual trainee. A knowledge based intelligent subsystem of the training architecture may function both as a tutor's and trainee's assistant. The purpose and the functionality of this intelligent subsystem is to support the optimal execution of courses by minimising human intervention.

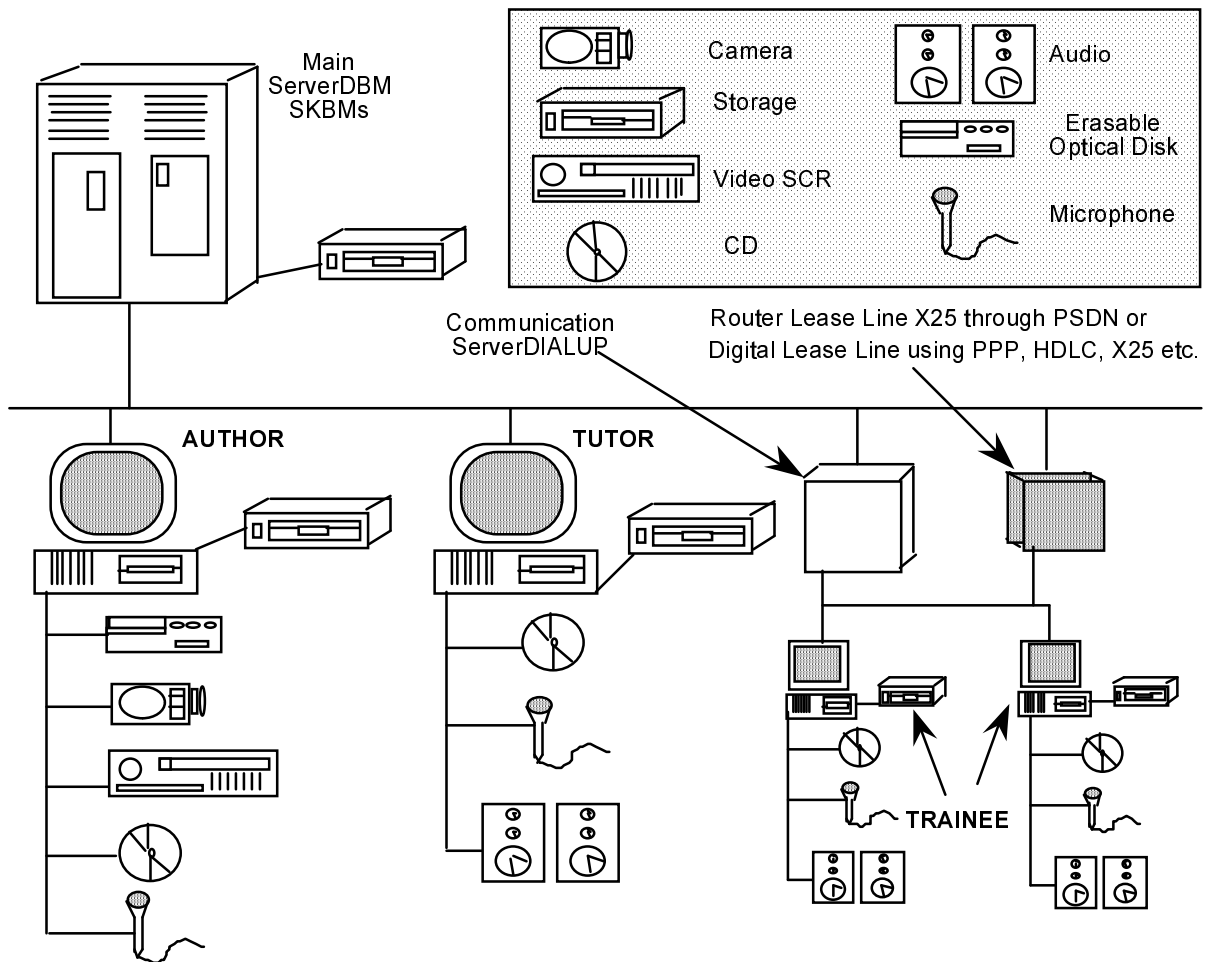


Fig. 3 HTAS Distance Learning Architecture.

The technical support for the communications infrastructure is foreseen to be provided directly by the PTT. However, the implementation of distance learning in HTAS is based on a

pragmatic view of the current economics situation for the communications in Greece and in Europe. Broadband - based distance learning will not be possibly a fully functional and economically feasible option before the turn of the next century. Therefore, the major body of remote training services will be based on interactive communications such as X.25 links and narrow-band ISDN and one-way communications will be based on conventional radio, TV and satellite broadcasts.

5. HTAS Application Exemplar

The HTAS platform will be validated in a sample application which is a modular courseware on Object-Oriented Analysis and Design (OOAD). This courseware is targeted to undergraduate and postgraduate students as well as to professionals of different specialities (e.g. managers of EDP departments, project managers, systems analysts, programmers, etc.) who are interested in this topic for various reasons. Following the SAT methodology and the HTAS tools, we begin with the first stage of the pilot application development, which is the *application domain analysis* and involves the specification of the training levels and the respective educational goals. The pilot provides the user with such a flexibility so as to allow him/her either to study the full course, or to select certain sessions of particular interest. This stage is carried out with the use of the *TAS tool*. The second stage is the development of *training specifications* and is accomplished using the *TSS* and *TM tools*. In this stage it is decided that the training in OOAD will be carried out in two levels: basic training and advanced training. Thus, users who are not interested in the details of OOAD will be just introduced to the object-oriented philosophy (OO) and its application to systems analysis and design. Users with high interest in this topic will take the advance training on specific OOAD methodologies.

The third stage of application development, called *course design*, deals with the design of the training material which is also very important for the successful completion of the courseware. This design is accomplished with the use of the *Flow Editing*, *Question Editing* and *Presentation Builder tools* of HTAS. The basic training course includes a hyperbook with sections related to: OO philosophy, its origins and concepts, the use of OO approach to various fields of computer science (and in particular in software development), the characteristics and advantages of object-oriented systems and their comparison to function-oriented systems, the characteristics of OO programming languages and finally an introduction to OOAD. The advanced training course is a hyperbook with sections related to OOAD according to Booch [3], Coad & Yourdon [6, 7] and Rumbaugh et al [16]. *Access structures* are also defined here. At this stage, the *schema* of the application has already been constructed.

The CBT can be more effective if certain tests are given to trainees. Therefore, the design of evaluation tests for trainees is another important issue in this application pilot. At the end of each training session, there is a list of computerised questions and exercises addressed to the user. The purpose of these questions and exercises is to check, record and analyse the progress of each

trainee as well as to aid him/her to assimilate the concepts used in the session. These tests are also designed during the third stage and are incorporated in the application *schema*, while their actual construction takes place during the next stage which is the *course implementation* stage.

The *course implementation* is carried out mainly with the tools of the previous stage and with other external tools which are used for the capturing of raw data and their incorporation to the HTAS environment. Furthermore, the presentation of material is accomplished by using different kinds of media in order not only to attract and hold the trainee's interest, but also because certain material necessarily requires certain presentation media. The *access structures* defined in the previous stage are refined during course implementation. Thus, here the actual contents of the designed course are constructed and an application is generated, i.e. an *object base* is constructed. This application is *an instance of the schema* and is then executed and used by *trainees* who interact with it via the *interface module* of the HTAS delivery environment and perform the constructed tests.

The results of the various evaluation tests, the remarks of *trainees* and *tutors* and any other information related to the course progress, are used as input to the *evaluation procedures* the execution of which is carried out during the next stage of application development which is the *course assessment*. The results of the course assessment are fed back to the previous stages and thus maximise the chances of having an acceptable and efficient multimedia application, prior to being formatted on optical media.

6. Related Work

The peculiarities of hypertext and hypermedia (e.g. the role of links, navigation, multimedia facility etc.) require the development of new models appropriate for the specific features of hypertext [9]. A number of hypertext models have appeared lately. Some of them describe specific application domains and activities, e.g. gIBIS [8] - a hypertext tool which explicitly models the semantics of the domain of exploratory policy discussion. Other models are more “system” oriented than application-oriented design models. These are attempts to capture the important abstractions found in a wide range of existing (and future) hypertext systems rather than of existing (and future) applications.

A representative example of the “system” oriented models is the Dexter Hypertext Reference Model [10] which aims to provide a principled basis for comparing systems as well as for developing interchange and interoperability standards. This model is divided into three layers: the *storage layer* (which describes the network of nodes and links), the *run-time layer* (which describes mechanisms for supporting the user's interaction with the hypertext) and the *within-component layer* (which covers the content and structures within hypertext nodes). The latter layer is purposefully not elaborated within the Dexter model and is considered to be outside the model per se. Another model is the Trellis model [18] which is mainly a “behavioural” model for hypertext. Hypertext networks are modelled as Petri nets and various browsing semantics are

discussed in terms of Petri nets computations. The Trellis model completely abstracts from the contents and structure of the nodes, which can be arbitrarily complex information structures.

Ogawa et al [13] propose a four-level design model and describe a number of design strategies, based on the model, for scenario-based hypermedia applications, i.e. applications in which the basic structure is based on a written story or a detailed scenario. Each level of the design model specifies respectively the global structure, the detailed structure and the content specification and the presentation style of an application. Ogawa et al claim that since the model and the strategies are described in general terms, they can be a good framework to construct a general scenario writing method for hypermedia applications.

HDM [9] is an application oriented model for *authoring-in-the-large*. HDM is used to describe generic applications and it has been adopted by HTAS as it is more suitable than others for application generation. HDM prescribes the definition of an application *schema*, which describes overall classes of information elements in terms of their common presentation characteristics, their internal organisation structure, and the types of their mutual interconnections. A schema, therefore, captures semantic and structural regularities in the representation structures for a given class of applications. Once a schema has been specified, the model also allows it to define a particular application, by providing primitives to describe a *schema instance*, i.e., actual *instances* of information classes and of connection types. In defining a schema instance, a significant number of connections can be left implicit, since they can be automatically derived from a conceptual-design level description. Additionally, HDM can be used to generate running implementations of hypertext applications. The HCT [5] is a set of tools which translate HDM specifications of hypertext-hypermedia applications into applications implemented by the commercial hypertext system ToolBook™. HDM differs from gIBIS [8] in the fact that it does not freeze, a priori, the application domain, and therefore its representation primitives are more “general”, oriented towards allowing the design of hypertext applications in most domains. HDM differs from the Dexter model [10] as it is aimed at modelling applications rather than systems. It also differs from the Trellis model [18] as it is more concerned with representational issues rather than behavioural aspects of hypertext.

Other approaches, less formal, emphasize preferred topological structures as building blocks to create the structure of hypertext networks. For example, HyperCard [2] uses linear structures to organise the constructed hyperdocuments. Guide [4] also prescribes the extensive use of linear structures while KMS [1] prescribes the extensive use of hierarchical structures for organising information.

From all the above models and approaches, it was decided that the HTAS environment adopts the HDM model since it is application oriented, it has a well defined set of primitives for designing hypermedia applications and it can be used for designing applications in a system independent way. The HTAS environment can be related to the ThyDoc [17] and SEPIA [19] hypermedia authoring environments. ThyDoc is an *authoring-in-the-large* environment, which

treats authoring as a knowledge-acquisition process that captures formally (in terms of a conceptual schema) and informally (through text, graphics, etc.) all the knowledge needed by teams of authors for the analysis, design, development and maintenance of complex hypertext documents. SEPIA is built on top of a multi-user DBMS and provides persistent and shared data storage, hypermedia data model with composites, and authoring functionality and support for cooperative work. HTAS differs from these environments in the sense that it is oriented for building hypermedia *training* applications; thus it addresses the peculiarities of training and places much emphasis on the trainee modelling, the trainee scenarios modelling, the adaptation of the scenarios according to the individual trainee's profile and the support for distance learning.

7. Conclusion

The proposed HTAS platform addresses the complexity of the effort for developing interactive hypermedia training applications and at the same time it offers *distance learning* facilities. HTAS addresses the deficiencies in current hypemedia technology with respect to the production of hypermedia training systems by enabling the systematic integration of teaching strategies in the production of courseware. Furthermore, HTAS addresses the problems observed in existing multimedia and AI authoring tools [11] by providing adequate methodology support (data collection and analysis), computer managed instruction support (course planning, grading), support for customising paradigmatic training examples for new domains, and support for integrating training into workplaces as job aids.

More specifically, the advantages accruing from the innovative architecture of HTAS are:

- * enhanced productivity of authors, as the tools offered to them minimize the effort for producing courses
- * improved courseware production, due to the closer cooperation between hypermedia development experts and subject matter experts
- * fast delivery of new courses and minimized production cost, due to the reuse of existing courses and/or due to the use of generic course templates for production specific courses
- * improved quality assurance due to the evaluation procedures which contribute significantly in the improvement of the quality of a course
- * individualisation of courses by taking into account the trainee stereotypes and using the individual user's profile
- * distance learning support.

The HTAS is currently at the end of the design stage, while a number of prototypes have already been implemented for visualising the various parts of the system. We believe that the HTAS environment will contribute significantly in the quality production of training hypermedia applications and will facilitate the overall hypermedia training applications production process

References

1. R.M.Akscyn, D.L. McCracken & E.A. Yoder, "KMS: A distributed hypermedia system for managing knowledge in organisations", *Communications of the ACM*, Vol 31, No 7, July 1988.
2. W. Atkinson, "HyperCard", in *Software for Macintosh Computers*, Apple Computer Co, Cupertino, 1987.
3. G. Booch, *Object-Oriented Design with Applications*, Benjamin Cummings, 1992.
4. P. J. Brown, "Turning ideas into products: The Guide System", In *Proceedings of the ACM Hypertext '87*, Chapel Hill, N.C. 1987, pp. 33-40.
5. A. Caloini, "Matching Hypertext Models to Hypertext Systems: a Compilative Approach", *Proceedings of ECHT-92*, Milano, Nov. 30 - Dec. 4, 1992, ACM Press, pp. 91-101.
6. P. Coad & E. Yourdon, *Object-Oriented Analysis*, Prentice-Hall, 1991.
7. P. Coad & E. Yourdon, *Object-Oriented Design*, Prentice-Hall, 1991.
8. J. Conklin & M. L. Begeman, "gIBIS: A Hypertext Tool for Exploratory Policy Discussion", *ACM Trans. on Information Systems* 6 (4), 1988, pp. 303-331.
9. F. Garzotto & P. Paolini, "HDM - A Model-Based Approach to Hypertext Application Design", *ACM Transactions on Information Systems*, 11 (1), Jan. 1993, pp. 1-26.
10. F. Halasz & M. Schwartz, "The Dexter Hypertext Reference Model", *Comm. of ACM*, 37 (2), pp. 30-39, Feb, 1994.
11. A. James & J.C. Spohrer, *Simulation-based Learning Systems: Prototypes and experiences*, In proceedings of CHI '92, Monterey Cal, May 3-7, 1992, ACM, NY, 1992, pp. 523-524.
12. D.H. Jonassen & R.S. Grabinger, *Problems and Issues in Designing Hypertext/Hypermedia for Learning*, in *Designing Hypermedia for Learning*, D.H. Jonassen, H. Mandl, Eds, NATO ASI Series, Springer Verlag 1990.
13. R. Ogawa, E. Tanaka, D. Taguchi & K. Harada, "Design Strategies for Scenario-Based Hypermedia: Description of its structure, dynamics and style", *Proceedings of ECHT-92*, Milano, Nov. 30 - Dec. 4, 1992, ACM Press, pp. 71-80.
14. R. Palmer, "Designing and Using CBT Interactive Video", NCC Publications, Manchester 1988.
15. M. Richartz & T.D. Rudebusch, *Collaboration in Hypermedia Environments*, in *Designing Hypermedia for Learning*, D.H.Jonassen, H. Mandl, Eds, NATO ASI Series, Springer Verlag 1990.
16. J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen, *Object-Oriented Modelling and Design*, Prentice-Hall, 1991.
17. R. Sobiesiak & J. Mylopoulos, "A Conceptual Modelling Approach to Authoring-in-the-Large for Hypertext Documents", *Proceedings of ACM Conf. on Organizational Computing Systems*, Atlanta, Georgia, *SIGOIS Bulletin*, 12 (2,3), pp. 225-239.

18. P.D. Stotts & R. Furuta, "Petri-Net-Based Hypertext: Document Structure with Browsing Semantics", *ACM Transactions on Information Systems*, 7(1), Jan. 1989, pp. 3-29.
19. N. Streitz, J. Haake, J. Hannemann, A. Lemke, W. Schuler, H. Schütt, M. Thüring, "SEPIA: a Cooperative Hypermedia Authoring Environment", *Proceedings of ECHT-92*, Milano, Nov. 30 - Dec. 4, 1992, ACM Press, pp. 11-22.