HYPERMEDIA NAVIGATION AS A CENTRAL CONCEPT
FOR INSTRUCTIONAL TOOL ENVIRONMENTS

Max Mühlhäuser, University of Kaiserslautern, Informatics Dept., Telematics Group
Erwin-Schrödinger-Str., D-6750 Kaiserslautern, max@informatik.uni-kl.de

Summary

Instructional tool environments need much more than a common database in order to integrate their tools and in order to allow for instructional material to be built in a sophisticated manner; we propose a model for hypermedia navigation which provides threefold support to the author, learner, and tutor: for modeling and controlling the learning process, for guiding through the actual domain knowledge, and for coordinating the use of tools and data. The approaches described were developed in a joint project of Universities and Digital Equipment in Germany, called Nestor.

1 INTRODUCTION AND CONTEXT

1.1 The Nestor Project

Nestor is a joint research project of German Universities (Kaiserslautern, Karlsruhe, Freiburg) and of the Digital Equipment Corporation CEC Karlsruhe research center. Nestor tries to facilitate the creation and use of sophisticated 'courseware' in networks of cooperating multimedia workstations. Our work can be divided into four areas:

- *Principles of instructional tool environments*: the basic architecture for an environment for authoring, learning, and tutoring (i.e. online learner support).
- *Computer aided instructional design (CAID)*: 'computerizing' instructional design and didactic knowledge in the environment in a machine- and human-readable format and in a highly reusable form (!).
- *Hypermedia and multimedia aspects*: enhancements to the hypermedia paradigm for better content representation, navigation, multimedia synchronization, etc.
- *Cooperation aspects*: generic computer supported cooperative work (CSCW) support for group authoring / learning, network transparent multimedia access, etc.

1.2 Hypermedia and Navigation

A *hypertext* shall be defined as a collection of 'nodes' ('pieces of information'), interconnected by 'links' ('references', 'relations') in a coherent graph. Usually, coherent subgraphs can be abstracted into *composite* nodes (Conklin, 1987).

*Hypermedia* is the notion used instead of hypertext if the 'information' in the nodes can consist of multimedia data. *Multimedia* in turn denotes a set of interrelated (synchronized) data of different time-independent and time-dependent (audio, video) media (Aambo and Hovig, 1988, Ambron and Hooper, 1988).

Hypermedia is viewed as an ideal concept for CAI (Jonassen and Mandl, 1989), supporting adequate course content structuring and optimal presentation media.
Hypermedia navigation shall denote the support for organizing one’s personal path through a hypermedia network (selection of starting node and further links to follow / nodes to visit, choices offered among potential links, etc.).

Our approach to navigation is sometimes called computed (conditional) path (Zellewe- ger, 1989). It is unique with respect to its formality, de-coupling from the underlying hypermedia networks, and reusability support. We want to advocate hypermedia navigation as a key concept for introducing a new quality of computer support in CAI, helping to
- ‘computerize’ concepts of didactics, instructional design, learner modeling, and human computer interaction in a reusable (!) way,
- exploit the true computational power available through the use of computers,
- catalyze a synthesis of the - up to now rather incompatible - fields of authoring systems, intelligent tutoring, and hypermedia-based CAI,
- integrate different authoring / learning tools flexibly in a common environment.

2 APPROACHES TO COURSEWARE AUTHORING

2.1 Alternative Trends in Hypermedia, Navigation, and CAI

Some hypermedia protagonists advocate the hypermedia metaphor for computers as equivalent to the book metaphor for print media (Kay and Goldberg, 1988), thus focusing on the storage/retrieval capabilities. We believe, however, that the computational capabilities of computers are too much neglected this way, that these capabilities should definitely be exploited for CAI, and that hypermedia navigation support is one way of doing this.

Others discourage navigation in the light of the experiences in the CAI field: research in intelligent CAI (‘ICAIs’) (Self, 1988) focused on intelligent and flexible learner models (for ‘intelligent tutoring systems’, ITS) which should reflect all possible users, including all possible misconceptions. This attempt turned out to be too ambitious: ICAI systems were helpful, but the learner models were not ‘perfect’ and the development effort was usually very high. This made people place their hope on hypertext; some tend to believe that learner modeling and instructional decisions (and, hence, navigation in hypertext) can not be seriously handled by a computer at all. Thus they propose to leave navigation decisions totally to the user and to support these decisions by optimal structuring and presentation of the learning material, i.e. through the use of hypermedia (cf. Landow, 1989).

We believe that unguided hypermedia exploration is not the ultimate CAI solution either. We rather propose a synthesis of different approaches: hypermedia concepts for information structuring and optimal presentation, plus navigation for a variable amount of ‘guidance’ through the hypermedia network. ‘Variable’ here means the range of
- free browsing without navigation support for special exploratory learning purposes;
- limited choices according to instructional strategies and learner models, where the navigation software makes preliminary choices and suggestions; the navigation software can range from simple algorithmic approaches to ICAI-like sophistication;
- fully program-controlled navigation for particular instructional strategies such as ‘games’ where the software needs full control over the path.

2.2 Known approaches

We see authoring/learning as a synthesis of the following two areas:
- ID knowledge (instructional design knowledge), more specifically:
  - the “ID process” applied, i.e. the selection, sequence and variants of steps which are executed (e.g., goal selection, audience assessment, storyboarding,...). In analogy to the software engineering process, the ID process can be very detailed and strict or more informative and liberal, it can be single-author or team work oriented, etc.
  - ID strategies (and rules); ID strategies denote coherent sets of guidelines (theories, models, rules, constraints, algorithms...) for the creation, selection and presentation of course contents, applied globally (to an entire course) or locally (to a course ‘unit’). Such strategies may be adapted from non computer aided instruction (e.g., strategies for tutorials or drill & practice courses), they may be based on ‘canonical’ ID theory such as ‘component display’ or ‘progressive deepening’ (Reigeluth, 1987), or they may be experimental and geared towards exploiting the computational strengths of computers. ID rules denote individual, non-coherent advises, rules or constraints to be followed during course design, such as general sequencing or layout rules. We will use ‘ID strategies’ to denote both strategies and rules as above.
  - Strategic learner model; this refers to the part of the learner model (learner knowledge, skills, preferences, misconceptions, etc.) which can be evaluated independent of the content domain, but dependent on the ID strategy.

- Domain knowledge, specifically:
  - the contents to be taught,
  - the relations within the contents which may influence the teaching sequence,
  - the domain learner model; e.g., the learner’s knowledge level as concluded from specific evaluations; this is part of the ID knowledge at the same time.

We now want to sketch three major approaches to courseware development, cf. Fig. 1.

![Fig. 1: Known approaches to courseware development](image-url)

In the authoring system approach, almost all of the knowledge (ID and domain) has to exist in the author’s head. Primitive instructional operations (‘instructional transactions’ such as ‘presentation’, ‘evaluation’, ‘decision’) are instantiated with domain knowledge and...
assembled, typically using "flowchart"-like techniques. During this assembly process, the author has the content references, ID strategies and rules, and maybe some learner model in his mind (!). Note that the generic ID and domain knowledge rest 'outside' the computer, only very course-specific, non-reusable software and information are 'computerized'.

In the ICAl approach (see above), mainly the domain dependent learner modeling aspect is considered, and to some extent other ID knowledge. ICAl systems bring substantial ID knowledge into the computer. But again, the generic ID and domain knowledge can hardly be reused in other ICAl systems since, in the ID process, they are first tailored to the specific ITS system and then computerized.

The hypermedia approach can be seen as an attempt to represent the plain domain knowledge (contents and references) in the computer. Pathways for integrating hypermedia techniques with authoring systems or ICAl techniques are hard to find.

In all of the above approaches, control of the ID process is either not computer supported at all (i.e., it is intuitively defined, described verbally, or else), or it finds very limited computer support, e.g., by general-purpose isolated project management tools.

3 THE NESTOR APPROACH

3.1 Principle Idea

In Nestor, learner models and sophisticated ID strategies are represented in generic reusable form as hypermedia navigation software. They are instanciated ('mapped', see below) with a specific hypermedia-based content to form a specific piece of courseware.

![Diagram illustrating the Nestor approach to courseware development]

The advantage is fourfold: a) domain knowledge and ID knowledge can be reused once they are 'computerized' (as hypermedia structures and navigation programs, respectively); b) since creation or change of navigation programs (ID strategies, learner models) is a relatively rare activity and decoupled from courseware implementation, it can be deferred to instructional design experts; c) the effort for courseware implementation is reduced to the 'mapping' process (see below); d) since domain and ID knowledge are 'computerized', computer support for the whole courseware lifecycle and hence, ID process, can be provided (instead of just for the 'implementation phase').

3.2 Underlying Hypermedia System

The underlying Nestor hypermedia system is superior to the general hypermedia model (cf. 1.2) in several conceptual and technical details. The major enhancements are:

- Links may have multiple sources/destinations and may be 'active', i.e., contain code;
- Links and nodes are 'typed' in the object-oriented sense (with classes/superclasses etc.);
- Several advanced technical features are added such as distribution, versioning, and base collaboration (CSCW) support.

3.3 Navigation

The Nestor navigation approach is based on the "PreScript" concept. A PreScript can be thought of as a navigation program for a "family of hypermedia networks", consisting of:

- A construction part, where the "family of hypermedia networks" is defined by a sort of graph grammar (cf. Stotts and Furuta, 1990) and
- A navigation part, where the path of the user through the network is controlled.

At the top level of the construction part, a hypermedia meta-network is defined, consisting of mandatory, optional, and/or repetitive links and nodes. Nodes may be either 'terminal nodes' or 'meta-nodes' (cf. terminal and non-terminal symbols in programming language grammars). The number of repetitions may be restricted to an interval, and the actual instantiated number of repetitions may be assigned a variable name.

Meta-nodes are recursively defined:

- they may themselves be defined as meta-networks in the same PreScript,
- they may point to another PreScript, i.e. to its top level meta-network,
- they may be left unresolved until 'mapping time' (see below).

In the navigation part, Nestor will (as a first attempt) support navigation programming according to a 'source/target/condition/action' transition table. This means that from a source node (node type, node selection, etc.), traversal of a target link (type...) or towards a target node (type...) can be restricted to conditions and associated with the execution of actions.

(The transition table may remind the reader of extended finite automata; note, however, that our approach is generic and supports a dynamic and variable number of nodes and links, hierarchies, typed links, and optional parts.)

Fig. 3 shows a primitive example of a PreScript in its graphical representation. Elliptic nodes represent non-terminals, limits for repetitions are indicated as y:1b...ub with 1b indicating the lower bound and ub the upper bound, and y indicating the variable that will contain the actual number of repetitions after mapping (see below).

The example shows a very simple "family of networks" for tutorial-type courseware:

- A (terminal) node 'In' (introduction) is followed by a sequential chain of 1+x 'Short' meta-nodes (short topics), which give an overview of the content domain ('x' is greater than or equal to zero, i.e. one or more Short meta-nodes may exist).
- The last Short meta-node is linked to a terminal node 'End' (learner evaluation).
- From the introductory node 'In' there are also 'y' links leading to meta-nodes of type
Main (main topic). From there, an optional link (number between zero and one) may exist to any other node (the link type ‘pr’ indicates that the starting node contains prerequisite knowledge for the target node).

Construction:
Tutorial:
methods eval, hist;

\( y: [1..2] \)
Main
\( y: [0..1] \)

Navigation:
Tutorial:
source link/dest cond. action
\( \ast \rightarrow \text{End} \)
\( \ast \rightarrow \text{MT} \)
\( \ast \rightarrow \text{eval(80)} \)

\( \ast \rightarrow \text{hist} \)

Short:
Main:

valid tutorial network 'basic arithmetic'

Fig. 3: Example Navigation PreScript for Tutorial-Type CAI

In a complete example, the construction part for Short and Main would follow. Fig. 3 also indicates that we require the ‘tutorial’ navigation program to contain the methods (object-oriented term for an operation or procedure) ‘eval’ and ‘hist’.

In addition, Fig. 3 shows an example tutorial (out of infinitely many) called ‘basic arithmetic’ that matches the PreScript construction definition; this example demonstrates the fact that additional links may exist in the actual network (in the example, these are the links leading from nodes of type Main to node End).

The example navigation rules in Fig. 3 are to be interpreted as follows:
- from any node (*) the learner is unconditionally allowed to follow any link that leads to node End (in other words, ‘exit’ is always allowed); along with the transition, the ‘eval’ method is to be executed (i.e., before exiting, an evaluation is forced).
- from any meta-node Main, the learner is allowed to follow a link of type ‘pr’ if the evaluation exceeds 80%. The learning history is updated during the transition in ‘hist’.

Note that each meta-node may stand for a composite node, i.e. a set of nodes.

3.4 Mapping

In order to prepare a specific PreScript for a specific hypermedia network, an author has to map the construction part of the PreScript onto the actual hypermedia network. This means that the construction part has to be expanded into an actual hypermedia network by:
- deciding about optional links and nodes;
- determining the specific number of repetitive links and nodes;
- instantiating every meta-node with
  - its corresponding meta-network in the same PreScript by recursive expansion, or
  - another PreScript, again by recursive expansion, or
- a composite node of the actual network (which means that within this composite node, no navigation support is given, i.e. exploratory browsing is assumed);

Three kinds of mapping are distinguished:
- generative mapping: the PreScript is driving a (graph-syntax directed) graphical hypermedia editor which allows to create the actual network (i.e. domain knowledge) according to the PreScript construction part; the mapping is done automatically.
- static mapping: the construction part of a PreScript is “expanded” under author control so as to fit with an existing network; limited automation will be supported in the future.
- dynamic mapping: left over from initial generative or static mapping by the author, some meta-notes may remain unmapped until execution time, being dynamically mapped by the learner or by a runtime operation (e.g., a hypermedia query). Dynamic mapping will be supported in the future, allowing for network modifications as discussed in 3.5.

Note that ‘author’ and ‘learner’ just denote ‘roles’, not persons.

3.5 The Triple Navigation Approach

Contents vs. Concepts: One of the oversimplifications in the above example which was introduced for the sake of shorthand was that every concept or skill (procedural, factual, tactile,...) to be taught (learning objective) corresponded one-to-one to a node or composite node in the content network. This assumption was implicitly introduced when ID knowledge (which copes with concepts) was expressed through nodes and links of the ‘construction part’ (family of networks) and was straightforward one-to-one mapped onto the domain knowledge, i.e. the nodes and links of the content network (e.g., a Short node named '*' and a Main node named 'multiply' existed). Since this is not the normal case, we want to describe the three principle Nestor navigation aspects of courseware implementation for the general, more sophisticated case where concepts and contents do not correspond one-to-one.

Content Network Navigation: Usually, the major content-related implementation step is ‘content assessment’, i.e. creation of the nodes of the content network with different media editors, inserting the content-specific references, finding abstractions (composite nodes) etc. In the Nestor approach, we add another step which defines the navigation through the content network which is necessary in order to obtain a certain concept (skill). A concept-meta-node plus navigation rules are thereby created for every concept, using the PreScript approach. Domain learner modeling plays an important role here, e.g., since the concept-related navigation will in part be based on learner evaluation, which in turn has to be included in the respective skill-meta-node(s).

Concept Map Navigation: In the sophisticated navigation approach, ID strategies and strategic learner modeling are expressed as related to a family of concept networks (or concept ‘maps’, as we call them) instead of being directly related to a family of content networks as in our primitive example. The lowest-level meta-nodes of an ID strategy (PreScript) are then mapped to the concept-meta-nodes of the content network navigation PreScript (see above).

Activity Graph Navigation: Up to now, we concentrated on the implementation (or development) step within an ID process. As indicated earlier, ID processes include earlier steps like ‘goal assessment’, later steps like ‘course evaluation’, and many others.

Since the Nestor hypertext model is enhanced such as to contain active nodes, Nestor
tools can be regarded as nodes. Data and milestones produced and used during the ID process can be regarded as hypermedia nodes as well. And dependencies such as 'has-to-follow' between steps or 'uses-input' between data and tools can be represented as links. With this interpretation, ID processes can be described as paths (threads of activities) on hypermedia nodes (tools, data) and links (dependencies, access rights), and hence as navigation.

The residual problem is the one that during execution of the ID process, new nodes and links may be produced, hence navigation rules have to reflect such dynamic structure modifications. As mentioned, we are currently elaborating the Nestor navigation approach in order to expand the PreScript approach to support 'navigation with modification'.

4 CONCLUSION

We have discussed the Nestor hypermedia / navigation fundamental concept, which is based on an enhanced hypermedia model (with object-oriented behavior, versioning, etc.), on top of which navigation is expressed via PreScripts. PreScripts are reusable, hierarchical navigation programs for a 'family of hypermedia networks', in the first phase based on transition tables. With this approach, formerly incompatible approaches like ICAI, authoring systems, and hypermedia can be combined, and a high degree of reusability can be reached.

Furthermore, ID knowledge finds its way in a generic form into the computer, enabling the use of sophisticated ID strategies by non ID experts. This is extremely important since authors are often not ID experts but rather domain experts.

In the future, we hope to find a concrete concept for using the Nestor hypermedia / navigation concept for controlling ID processes, i.e. 'navigation with modification', too.

5 REFERENCES


Zellweger, P.T., 1989,