Engineering Web-Based Multimedia Training: Status and Perspective

Max Mühlhäuser
Darmstadt University of Technology, Telecooperation Group
max@informatik.tu-darmstadt.de

Abstract

Like Computer-Based Training, Web-Based Training (WBT) has embraced multimedia based content as a must-have. The market offers a variety of WBT authoring (hence, MSE) tools. Within their defined scope, these tools offer a high degree of sophistication. The following article discusses the state of the art for such tools. While little may be blamed if one accepts this defined scope, the article puts WBT in perspective, analyzing the requirements which one might *put on web-based multimedia learning systems at a second glance. Given these new requirements, the tools available on the market leave a big gap. Approaches towards filling this gap are discussed, with a particular focus on XML and related standards.*

1. Introduction –the WBT approach

Web-Based Training (WBT) is offered as a silver bullet for the training and education needs of the information society. WBT authoring is an issue of multimedia software engineering since WBT content is by and large expected to be multimedia in nature. Yet WBT as it is common on the market today is just what used to be called Computer-Based Training, with two exceptions: i) the delivery channel has moved onto the Internet, and ii) the available Web technology is used - more or less, see below -, with little benefit for the user. This article is split in half, with one part analyzing the present status and the other one offering exciting yet rough-road perspectives. Part I describes important features of and approaches to WBT software engineering. Major requirements are deducted and used to evaluate some of the most successful such tools. Only few of these tools have been newly developed for the Web, the majority has been Web-enabled. Plain HTML editors will also be included in the comparative evaluation. Part II challenges the state of the art discussed in part I. Both with respect to the evolution of "teaching theories" and with respect to the state of the art of academic hypermedia based learning systems, WBT tools lag considerably behind. This claim is substantiated. It turns out that many desirable features are hard to realize with present HTML-based Web technology. Therefore, the article finishes by discussing how the upcoming new Web technology (cf. XML, RDF, Xlink, SMIL, ...) can help to realize the considerable advancement of WBT practice in quest.

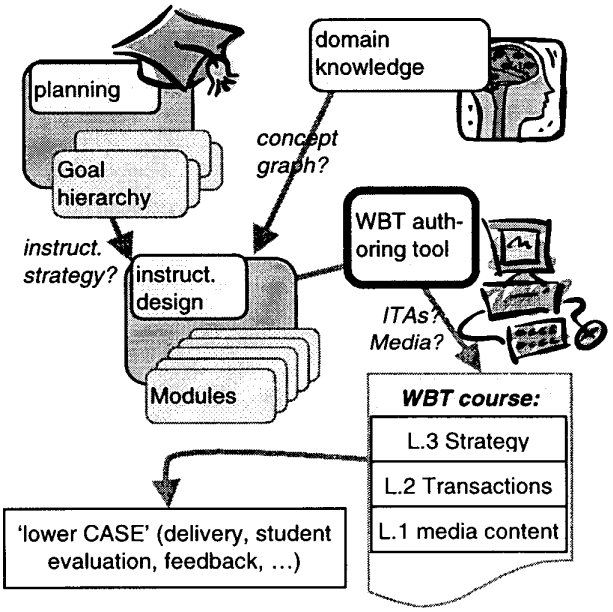

Fig. 1: WBT engineering lifecycle (upper CASE)

For the remainder, we will consider WBT material (a course) to consist of three layers:

1. The bottom layer contains individual (multi)media contents, representing 'atomic' entities of the domain knowledge.
2. In layer two, we find the 'instructional transactions' (ITAs) that make up the heart of WBT tools. ITAs access the media layer to display content.
3. The top layer realizes the instructional strategy (based on course types such as 'tutorial' or 'drill and practice', and background theories such as 'progressive deepening' or 'component display theory'). This 'macro' strategy determines the course-of-action of ITAs.

0-7695-0933-9/00 $10.00 © 2000 IEEE

Fig. 1 illustrates that the use of WBT authoring tools is the last step in the 'upper CASE' part of the lifecycle (the part that leads to a first implementation). According to common practice, the major steps preceding WBT authoring are as follows:

- didactic planning, such as determination of the hierarchy of learning goals (including target audience analysis) and of the instructional strategy to choose
- content elicitation, including semantic description of the contents (e.g., using a concept graph) and syntactic description (i.e., creation or capturing of media)
- detailed course planning, including determination of the course modules and submodules, their interrelation with the instructional strategy, and their realization via ITAs and related content (media).

WBT authoring tools on the market do not, by themselves, offer support for the above-mentioned upper CASE, they are basically high-level programming tools. However, platforms like Attain™ contain tools for both upper and lower case tasks.

2. Available WBT tools

2.1. Basic choices

In the remainder, we will distinguish WBT tools according to i) their general class, ii) the metaphor used, and iii) the degree of ITA support.

General class: Since WBT is intended for delivery on the Web, any general Web authoring tool (without particular dedication to learning or training) may be used, a choice with obvious disadvantages, but also with advantages as we will see. At first sight, one would prefer dedicated WBT authoring tools. Since such tools made their market entry more recently than the other two classes considered, they face strong competition from the feature-rich general Web authoring tools and from the third class, Web-enabled CBT authoring tools.

Metaphor used: apart from the above classification, the most obvious distinction considers the underlying metaphor used to organize entire courses and to assemble and align ITAs. For our purpose, it is sufficient to distinguish three major classes:

- The *book/card metaphor:* this term merges two notions into a single class since both a 'book consisting of pages' and a 'deck of cards' translate into frames (windows) displayed on the computer monitor plus a control flow scheme in the background which determines the 'next frame to be displayed'.
- The *time-axis* metaphor views the WBT course as one large 'movie'-like project. Both *book/card* and *time-axis* assume a pre-dominant sequential organization of the course and consider branching the 'exception'. For *book/card*, continuous media run within a page, and the *advancement* to the next-page is an event to be explicitly programmed. For *time-axis*, continuous media are most naturally attached to the axis; they act as self-advancing ITAs, other ITAs which *wait* for events such as learner input must be explicitly programmed.
- The *icon-flow* metaphor requires the author to arrange the course as a kind of high-level control flow. The icons offered by the system are instantiated as 'boxes' of the control flow and represent the ITAs directly. Usually, background computation (e.g., for maintaining a user model) can be easily linked to the ITAs. This metaphor is obviously most appropriate for implementing non-trivial instructional strategies.

Degree of ITA support. ITAs can be roughly classified into three categories: presentation ITAs, interaction ITAs, and control flow ITAs.

Presentation ITAs represent the most obvious category which matches 1:1 with media types and which is used to convey contents to the learner. WBT tools differ here with respect to both the kinds of media supported and the formats supported. While external tools may compensate for lacking format support, lacking media types may considerably restrict an author's creativity: WBT quality may depend considerably on, e.g., animations and synchronized multiple-media presentations (or other types, depending on the subject). Proper inclusion of these presentation types may not be feasible if they are created using an external tool and presented as 'external media'.

Interaction ITAs can be considered the spice of WBT since extensive use of these keeps the learner involved. Different tools offer again different levels of sophistication here, reaching from multiple-choice to natural language support, drag-and-drop placement, etc.

Control Flow ITAs finally determine branches, case selections etc. They are of course most explicit in *icon-flow* based tools and at best implicitly available in time-axis based ones. Their ease-of use and sophistication may vary considerably from tool to tool.

Lacking support for instructional strategies: note that the instructional strategy as such is not explicit in common WBT tools, but must rather exist 'in the head of the author' and mapped onto the ITA arrangement (which in turn is only truely explicit for icon-flow based authoring tools). We will get back to this lacking instructional strategy support later in this article.

2.2. Evaluation of sample tools per class

We will now mention one example from each general class and discuss advantages and disadvantages. Please note that the statements refer to the status of the respective tool given in the version(s) evaluated and may already have changed by the time this paper is published. The

following evaluation is thus thought as a sample, helping the reader to get familiar with important criteria – it is *not* thought as a buyer's advice.

General Web authoring tools: if one accepts the view that a single HTML document is basically a page without size limits, then all tools of this first class can be considered to use the book/card metaphor. Obviously, one cannot expect a general Web tool to support learning-related issues particularly well, such as media or ITA types which are restricted to didactic purposes (e.g., drag&place visual questionnaires). In the past, interaction ITAs were also rather restricted, but in this respect, this class of tools was improved a lot recently (reflecting the fact that Web applications tend to become increasingly interactive). Persistent drawbacks include i) the lack of integration with other learning-related tools in the lifecylce (cf. fig. 1), and ii) the lack of explicit support for instructional strategies (which is not included in any of todays tools, but might be so for advanced tools of the other two classes in the future).

In order to demonstrate difficulties with using a tool of this class for WBT, let us consider the straightforward issue of creating a multiple choice test with a tool like FrontPage™. The good news: creating the corresponding page is easily achieved with buttons and forms. However, processing the result is a rather clumsy task which involves email to the server, Java programming, or some other heavy-weight mechanism. Another area of possible difficulties is the consistent management of the set of HTML pages which makes up the WBT course. In this respect, some tools (like, e.g., NetObjects Fusion™) provide sophisticated support, others lack behind.

Is there any considerable advantage in using a standard Web authoring tool? Yes, several! 1. These tools are sold in a very competitive market, they are forced to be comprehensive and user-friendly, and in particular to keep pace with the fast-evolving world of web standards much more than the 'niche market' of WBT tools. *If* an author realizes that the nature of a course to be built goes way beyond the WBT state of the art (see next chapter), such that he has to accept a considerable programming-from-scratch effort anyway (based on Java, say), and / or if an author wants to make use of a life-cycle platform which is rather independent from the 'implementation' tool used anyway (such as the Lotus learning space platform), then this class of authoring tools may be the right choice.

Dedicated WBT authoring tools: At a first glance, one would expect this second tool class to be ideally suited. On one hand, however, general authoring tools compete in a more dynamic market (as was mentioned); on the other hand, many CBT tools exist since a pretty long time and have evolved a lot over time and based on the massive feedback from years of CBT development. In other words, both 'competing' tool classes tend to offer much more mature products. Since HTML is the 'mature' Web technology compared to XML and since HTML is going to be around for years, most WBT authoring tools are expected to stick to HTML for some time to come. The remaining chapters of this paper will discuss, however, that HTML is inappropriate for realizing many of the desired features Some of the WBT authoring tools use Java to overcome these restrictions. This means, however, to chose closed-shop solutions 'hidden' in proprietary Java code, not accessible to the world of well-structured Web-based hypertexts. In other words, if a WBT authoring tool manufacturer tries to stay on the 'pure HTML, no Java' track, they are very restricted by HTML technology. If the manufacturer accepts to include Java (or JavaScript or a plug-in), they are suffering the same drawbacks as CBT-related tools (see below). Brief, the choice of pure WBT tools is not as obvious at second thought.

To cite a positive example, Dreamweaver Attain™ is a WBT authoring tool that evolved out of an HTML editor. Augmentations concern, e.g., sophisticated interaction-ITAs (called 'knowledge objects' in Dreamweaver) and integration of the learner evaluation cycle (called 'knowledge track'). Besides, Attain™ is a well integrated suite of tools covering most of the lifecycle. Nevertheless, the disadvantages of HTML based tools as discussed in the next sections applies.

Web-Enabled CBT authoring tools: this class is the arena of tools well established in the CBT market such as Authorware™, Toolbook™, Quest™, or IconAuthor™. These brands stand for a rich choice in functionality, for the whole spectrum of metaphors, and for a rich selection of ITA types and media. Several manufacturers have concentrated on adding complementary tools to provide improved lifecycle support, but the most competitive aspect of these tools is currently related to server support. Web-enabled CBT authoring tools, like the other classes, have replaced the CD-ROM delivery medium by the Web and augmented this deployment channel by more or less intensive support for learner management and feedback. The most advanced tools include streaming, caching, and compression of continuous media on the delivery channel.

While these features are lively discussed in press, they represent technology-related features but not the core criterion in the context of this paper. Rather, the question is – like in the second class of authoring tools – to what extend Web technology is exploited. Again – and more drastically than for the second class – two strategies can be distinguished. Some manufacturers try to avoid home-brewn programs or plug-ins and must compromise functionality (note that these tools can still be used to 'compile' courses for other distribution channels such as CD-ROM, so that this strategy will usually lead to restricted features if compiled for standard Web technology). Some manufacturers use plug-ins to deliver courses. This means, however, that Web browsers represent merely a 'window frame' for the course, while the course as such remains 'untouchable' for

standard Web technology (e.g., HTML links *into* compiled courses cannot be created).

Toolbook – to cite an example – offers even two alternatives: a completely designed course may be compiled for the so-called Neuron-Plugin for Web Browsers (second strategy above) or it may be compiled into a HTML+Java solution. Although this latter alternative might overcome the restrictions of the first strategy (at the cost of 'hiding' the Java parts in proprietary programs), the version evaluated by the author does not offer all possible features of a Toolbook project on this delivery path.

Table 1: Raw comparison of WBT authoring tools

tool class	metaphors	major pro's	major con's
general Web authoring tool	book/card	up-to-date competitive	WBT/platform support low
Web-enabled CBT tool	book/card time-axis icon-flow	mature feature-rich	proprietary solution or restricted features
dedicated WBT tool		improvements likely	

A summary of the comparison discussed in this chapter is given above in table 1. In summary, we can conclude that there is a rich choice of authoring tools for WBT projects, all with some advantages and disadvantages, but all of them living up to most of the requirements imposed on a 'standard' WBT tool. Coined like this, one might think that the "WBT world" is pretty much in good shape. The real problems and deficiencies come up, however, if we dare to question if WBT (as it is understood today) aims at the appropriate goals.

3. Towards Web-Based learning systems

Questioning the state of the art in WBT starts with recalling the fact that CBT represents just *one* of several possible forms of learning systems, and that WBT today is not much more than an effort to port CBT to the Web. CBT/WBT is in fact the only realization of learning systems which is in the large scale commercially successful, and this has to do with its rather modest pedagogic sophistication. Pedagogy and computers did not marry even nearly as easily as many have envisioned, even promised – but this may improve if we manage to exploit advanced Web technology. This issue is what the remainder of this article is about.

3.1. Computer-Based learning system classes

Different categorizations of learning systems (LS) have been proposed in the literature. The following list tries to summarize and harmonize these classes. For the sake of simplicity, we will distinguish three top-level classes: *descriptive, model-based, and tools-based* learning systems.

A. Descriptive Learning Systems: this class denotes courseware which (to a large extent) describes the subject matter domain based on text/graphics or multimedia.

- *Tutorial LS* represent the class which current WBT tools emphasize (by and large). They support the simple cycle of presenting information (facts, examples, methods, ...), giving assignments, and deciding about the next iteration of presentation and assignment. The decision-making is not supposed to be pedagogically sophisticated, and the 'next iteration' is supposed to be a pre-authored sequence with little adaptation to the current learning status.
- *Drill&Practice LS* are more sophisticated with respect to the assignments. These are generated (in a simple form, out of a large set of pre-authored building blocks). The 'drill' part resembles what we called 'presentations' above, but tends to be less sophisticated than with tutorial LS. Generally speaking, this class does not differ much from the tutorial one.

Traditionally, books and lecture notes (with exercises) were the pillars of descriptive teaching. CBT authors and manufacturers of CBT authoring systems often (implicitly) consider CD-ROMs as 'multimedia books', hence the affinity of CBT to descriptive learning systems. CBT production is an enormous effort (100 to 1000 hours of production for a 1 hour course), but is still less costly and yields more dependable results than the production of a learning system of the next classes to be discussed. These facts and experiences and the given legacy lead to the dominance of descriptive learning systems in the WBT world – although Web experts would agree that the Web is much more than a collection of multimedia documents.

B. Model-based learning systems: the following list of prominent members of this class seems to be heterogeneous only at first sight:

- *Intelligent Tutoring Systems (ITS)* incorporate some degree of AI approach. Most important, they try to build a generally valid model of the learners' evolving knowledge structures (including misconceptions), so that during the learning activity, the actual learner's state-of-mind can be inferred from his or her behavior. In addition or as an alternative, the domain knowledge may be modeled as a rule based system (although this aspect is more characteristic for the next LS type discussed). The high expectations put in AI were also put in ITS in the past, and these expectations had to be considerably reduced, too.
- *Simulative Approaches* map an excerpt of the real world (existing, possible, past, or planned) into software. The essence of simulation is abstraction since the part of the world to be simulated is considered

much too large and too complex to be modeled in a simulation program in full detail. Thus, the art of simulation is to build a model which concentrates on the aspects to be investigated (in the 'cut-out' of the world), and which abstracts from the rest as much as possible without invalidating conclusions (knowledge) drawn from simulation experiments. Four derivatives must be mentioned:

i) Management simulations and *'learning role-play'* are based on putting the learner in a certain role and have him explore the world to be learned about.

ii) Fun learning games separate the 'game goals' from the 'learning goals' and try to challenge and motivate the user by having him concentrate on the game goals (e.g., find a hidden treasure). The learning goals and learning steps are rather disguised.

iii) Behavioral and application simulations exploit the classical advantages of simulations: they provide a 'safe playground' were the real environment is costly or unavailable (or dangerous). Behavioral simulations concentrate on particular situations for which the learner is to be trained, preparing him or her for the alternative actions possible, and concentrating on improved performance with respect to 'correct choice', 'fast reaction', etc. Application simulations prepare for the application of a certain technique, machinery, tool, or software.

iv) Microworlds differ from the above in that they concentrate on depth, not breadth of the simulation model; here, abstraction is the art of choosing a minimal sub-set of the world, not that of leaving out unimportant details *within* this sub-set. Microworlds are small enough to be modeled rather exact and complete. Such formalized descriptions are much more viable for domains closely related to (not too complex) mathematics and logic, such as e.g., Newton physics. The depth i.e. relative completeness makes microworlds well-suited for combination with learner-model based approaches, similar to those used in ITS.

- *Programming approaches* leave both the execution *and* the construction of a (more or less simulated) dynamic system up to the learner. Three classes shall be distinguished here:

  i) Programming environments for learning purposes emphasize simple and intuitive programming paradigms (often, visual-programming based) and error checking / correction (e.g., explanation of the nature of errors, guidance). They are often customized for specific classes of programs. The corresponding subject matter may be programming, but also domains closely related to maths, logic, and/or algorithms.

  ii) Problem solving systems offer building blocks which learners have to select and arrange in order to solve problems given as assignments. Selection and combination are much more restricted than in programming environments, offering chances for better guidance but restricting creativity and explorative space.

  iii) Advice-giving / help systems, in their simplest form, consist of (maybe hierarchically organized) lists of 'frequently asked questions' (in the broadest sense) and corresponding answers. More sophisticated systems are based on learner modeling. If attached to software (e.g., desk-top publishing tools), the model can be actualized as the user interacts with the software (not only the help system!); otherwise, the advice-giving or help system may pose a questionnaire in order to tune the model to the user.

Model-centric software. The above list of terms found in the literature shows one major commonality: all classes described represent model-centric software. The term *model-centric* refers to the fact that one or more of three possible areas are modeled: the *learner* i.e. software user, the *subject* matter i.e. learning domain, and/or the *motivational aspect.* All learning system classes cited represent *software* as opposed to the former class descriptive LS which rather represents document-centric learning.

Subject model. Concerning the subject matter or domain model (the term *subject model* will be used in the remainder), the software representing a learning system aims at carrying out a *plan* on the model. They can be further categorized with respect to *who builds and carries out the plan.* In programming environments, the learner clearly builds the plan. In applications simulations, the user carries out the plan ('events' happen only as a result of learner activity), whereas plan execution is shared for role-play (events may be learner- or system-initiated).

Other models. The learner model is the design-center of ITS, but may also be used (with less emphasis) in other classes listed above. Fun games focus on the motivational model as part of the game strategy; game strategy, learner model and subject model are usually treated separately.

C. Tools-Based learning systems: these are meant to help learners organize their tasks (i.e. exercises, assignments) and to represent the intermediate steps and results i.e. domain-related findings as digital artifacts. Quite often, the tools themselves are not specifically designed for learning purposes only. It is the embedding (i.e. tool combination), the given assignments, and the system- or teacher-based guidance / control that make this class a learning-specific one. Major subcategories are as follows:

- *Cognitive tools* for the acquisition, organization, and sharing of knowledge; examples comprise frameworks for human-readable representation of cognitive processes (e.g., rhetoric and argumentation spaces), mind-mapping tools, etc.

- *Groupware* such as group discussion and group decision tools, augmenting the above-mentioned subclass with group support.
- *Work organization tools* such as time/project management software and literature databases.

A further categorization known from groupware (support software for computer-supported cooperative work) applies to this whole group, too: the distinction between synchronous (cf. computer-based conferencing) and asynchronous work (cf. workflow management).

Table 2 below summarizes the learning system classes and their strengths as discussed. In addition, it points at weaknesses further elaborated in the following sections.

Table 2: Classes of Learning systems

LS Class		Subclass	Strengths	Weaknesses
descriptive learning system		tutorial LS	commercial success (WBT)	Lacks: models, cooperation, exploration
		drill&pract.	exercises	
model-based learning system		ITS	learner model /subject model	high effort, low re-use
	simulation	role-play	models, interact	Web technology not used today, no standards, low re-use
		game	motivational	
		B/A simul.	low-risk trial	
		microworld	accuracy, AI	
	programming	erogramming environment	creativity + freedom	as above no cooperation no openness
		problem solving system	creativity + models	
		help system	just-in-time, (learner mod.)	
tools-based learning system		cognitive tool	conceptual understanding	No 'hooks' for guidance, learner model
		groupware	cooperative	
		work organization tool	self organized learning	

3.2. Web-Related aspects

The present article challenges the current emphasis on descriptive learning systems in the WBT world. This is done by considering pedagogic and didactic guidelines in an up-to-date fashion; beforehand four simple fundamental observations about the Web will be conveyed as follows:

1. The Web is a hypertext system: for a number of reasons, the Web today appears to many as a collection of multimedia documents with optional links (to other such documents). But more and more, the Web becomes what it should be: a true hypertext i.e. a collection of semantically atomic 'pieces of information' (nodes) which are interrelated (via links) in various ways. Links supporting sequential reading are just one such variation. In its most general form, a hypertext is a collection of bubbles and of arcs linking these bubbles. Concept graphs representing the core semantics of a subject matter may be constructed as hypertexts just like graphs of object acquaintances in a piece of software. Current HTML-based technology is marked by the out-dated 'collection of documents' view of the Web, the upcoming XML-based technology supports the more general 'bubbles-and-arcs' view much better.

2. The Web is Open and Global. In drastic opposition to CD-ROMs (which Ted Nelson used to call the 'pre-columbian' view of digital media – the world being a disc with a dangerous border), the Web reaches out to the highly active and dynamic Internet world. In contrast to any compiled (e.g., printed, pressed, released) media, which are in our days out-dated shortly after being compiled, the Internet is the most actual representation of 'what we know'.

3. The Web integrates document-centric and software-centric MSE: from HTML to DHTML to Java Applets, hypertext nodes ('bubbles') exhibit an almost seamless spectrum from passive (documents) to active (programs). The same is about to become true for the 'arcs' (cf. computed links) and for aggregations of bubbles and arcs (collections).

4. The Web is the catalyst of convergence: telecommunications (human interaction via technical means), media (both professional and consumer generated multimedia), and information technology (the software world) merge on the Web. Hence, working in the Web will more and more smoothly integrate interaction with documents, with software, and with humans.

The above observations immediately put the close relation of WBT and CBT into perspective: confined and compiled, document-only courses cannot be re-compiled into the ultimate WBT product. Rather, it becomes immediately obvious that **Web-mature curricula must provide an appropriate mix of descriptive, model-based, and tools-based learning modules.**

3.3. Pedagogic/Didactic aspects

For further insight, we will recall some didactic and pedagogic aspects of computer-based learning and revise them for the Web era (cf., e.g., [5, 7, 10, 13]).

Skills Development: Modern instructional design starts by looking not at a desired 'status of knowledge' but at the desired skills which the learners should master at the end. Normally, skills are related to the subject matter and drive the entire instructional design process. In addition, however, any educational setup inherently makes the learner develop skills which are required for coping with this setup. E.g., a learner using CBT material will develop skills in computer use. These skills are obviously helpful in other domains, too, in both work life and private life.

In the Web era, more and more job descriptions explicitly or implicitly call for the following Web-related skills: *i) knowledge workers* i.e. people able to acquire knowledge just-in-time, often through the Web, in an effort to give their organization a competitive edge as part of, e.g., a task force; *ii) team workers* i.e. people acquainted to working in lean organizations where democratic teams must self-organize and collaborate closely; *iii) net workers* i.e. people who organize themselves as mobile employees, working in and with the net to remain an active part of their community (in essence, the company and its partners) while on the move or while teleworking.

A properly designed curriculum may support orthogonal skill development with respect to knowledge, team, and net working in particular if the mix contains sufficient and appropriate tools-based learning modules. WBT however, as it is typical today, provides pre-compiled material, pre-digested by the author to a very high degree, intended to be 'light meal' which is easy to consume.

Pedagogic Philosophies: For years now, pedagogy calls for a move from instructivism to constructivism [5, 13]. The learner is not to be considered an 'empty page' but an individual with prior knowledge who continuously constructs individual new knowledge, among others by interacting with learning systems. As there is no 'empty page', there cannot be a process for writing on this empty page and for testing the degree of success either. Rather, there is a call for offering choices and for task-oriented learning systems. CBT has hardly followed these proposals since guidance through task-oriented assignments is very difficult and costly to implement (especially if combined with a call for a broad spectrum of tasks), and since CBT vendors do not want to compromise eprogram-based learner assessment (which could only be retained in advanced LS if associated with a sophisticated learner model; this however is costly and risky as ITS has shown).

In the Web era, two possible improvements nurture hope: a) the virtually unlimited offerings (of information sources and tools) support the constructivist quest for a broad variety, b) the merged communication/data networking provides a basis for human-assisted guidance (for task-oriented assignments) / learner assessment.

Learning theories: the instructional transaction (ITA) types discussed earlier characterize WBT as behaviorist: these ITAs may be related to author's intentions like stimulus, response, feedback and enforcement and thus mark a learning theory where learning is basically an objective, measurable change in behavior. *Cognitivism has largely replaced behaviorism* in modern pedagogies. It is based on models of mental state (of the learner) and on the appropriate choice of strategy (induction, deduction, drill&practice, etc.) for any given learning situation. Obviously, the importance of learner models closely relates cognitivism to model-based learning systems, in particular to ITS. High cost and limited measurable success have as of yet hindered the large-scale application of cognitivist approaches in CBT/WBT.

In the Web era, we must recall that most successful cognitivist learning systems were built using hypertext systems. Hypertext is the only approach suitable for covering domain contents, subject model, and learner model, and for flexibly coupling these in an encompassing learning system – usually drawing from both rule-based and procedural programming. However, these successful systems were not built based on current Web technology but on hypertext systems with more advanced features. The lacking wide-scale commercial success of these hypertext-based cognitivist approaches is, to a large part, due to the high cost and very limited re-usabiliy of software development. This issue will be resumed later.

Teacher roles: teachers were always and still are considered experts in their field.

In the Web era, there is an important call to have teachers evolve from 'the sage on the stage' to 'the guide on the side'. In other words, the former expert who incorporated, processed, and conveyed the subject matter (in the case of CBT, as an author) turns into someone who 'merely' assists the learners in the process of acquiring and applying domain knowledge themselves. This move is, on one hand, inevitable in our time due to information overload and rapid innovation which make it impossible for a human to be truly a sage. On the other hand, this move is very desirable since it supports the development of the orthogonal skills mentioned above (knowledge worker etc.). The Web is of course an ideal ground for this move since it is the best base for knowledge-working and for guided self-discovery. Guidance during an unlimited Web exploration, however, is already difficult to implement in the form of a 'hot-line'; it is much more difficult to realize as a human-centered monitoring and control activity (which it typically has to be for non-adult learners and which is often desirable even for adults). As a program-based (instead of human) activity, guidance during Web exploration is only feasible today if the scope of exploration is restricted – a contradiction to the goals stated above (e.g., truly up-to-date information can most likely not be found if the search space is defined at courseware compilation time). In any case, the call for the 'guide on the side' is equal to a move away from instructivist and behaviorist approaches towards constructivist and cognitivist approaches, re-enforcing both the chances and the open issues of current Web technology.

Motivational aspects: for our purposes, intrinsic motivation shall denote motivation of the learner evoked on purpose by the learning system, in contrast to extrinsic motivation which is out of the intentional scope of the

learning system (extrinsic motivation may stem from the learner's knowledge gap on the job). Intrinsic motivation may be individual (cf. fun learning game) or interindividual (cf. cooperative or competitive group assignments or tasks). Obviously, group-based fun learning games may be considered the ultimate learning system from this perspective. They also have a disadvantage, however: the degree of explorative and experimental freedom given to the learners with respect to subject domain has to subordinated to the game model (e.g., a learner can not be allowed to explore parts of the subject model which he should not know with respect to the game model).

In the Web era, intrinsic motivation might be increased to a certain extend by the mere fact that assignments are given as self-organized Web-based tasks – many learner will (still) like to 'surf the net' in the learning context. In addition, and more sustainable, the emerging prominence and increasing visual appeal of group games on the Web offers an important key to interindividual motivation (through cooperative fun learning games). High cost, low re-usability, and required expert knowledge associated with the development of appealing Web group games represent again substantial obstacles between the theoretic possibilities and the practical usability of Web technology in this respect [11].

Table 3: Pedagogic / didactic requirements (LS: most pertinent learning system class: tools-based (T), subject / learner / motivational model based (S/L/M), descriptive)

ped./didactic aspect	requirement	LS
skills development	knowledge / net / team worker	T
pedagogic philosophy	instructivism -> constructivism	T
learning theory	behaviorism -> cognitivism	L
teacher role	sage -> guide	T
motivation	extrinsic -> intrinsic (interpers.)	M
training ground	academic -> authentic	S+T

Training ground: Pedagogy distinguishes between academic and authentic training grounds. Academic means that exercises are created on-purpose (cf. fill-in texts), authentic means that methods and procedures may be practiced 'in real' – more or less in a form of apprenticeship. Simulations represent the attempt to bridge these two extremes.

In the Web era, there are two chances for improvement: i) costly simulations (and the like) may be offered on the Web, with different teachers or institutions sharing either cost or contributions; thus, simulations can become more wide-spread among learning systems; ii) more and more business happens on the Web actually. Thus, apprentices for a growing number of subject matters may be linked to the 'real' domain for controlled intervention. For instance, a growing number of journalists cooperate with one another and with agencies and media businesses over the Web. Student journalists may be linked to such a virtual private network in a controlled way.

Table 3 above summarizes the key points elaborated.

3.4. Quest for a mix of Web-Based LS types

The above section may be related to the three classes of learning systems (descriptive, model-based, tools-based) to yield a vision of desirable classes of Web Learning Systems (WLS).

Descriptive WLS: for this class of learning system, chapter one has emphasized the advanced state of WBT today. As of yet, this advanced state has to be paid for: high quality WBT authoring systems still apply proprietary technology on the delivery end (browser plug-ins, proprietary Java modules, etc.) and produce rather monolithic course modules.

Requirements: Future descriptive WLS should be highly modular, based on public Web standards, and be highly re-usable.

Learner-Model based WLS: the (up to now still mostly academic) success of hypertext-based ITS systems has shown that advanced hypertext is an ideal basis for the development of learning systems based on sophisticated learner models. However, HTML cannot compete with the hypertext systems used in the literature.

Requirements: Learner-Model based WLS must be supported via multi-layer hypertexts where different layers represent the user model, the concept graph (or alternative representation of the subject model), the descriptive material for the subject matter itself, and the instructional strategy to be applied.

Subject-Model based WLS: simulations, microworlds, problem solving systems, and other learning systems based on extensive dynamic subject models have hardly been built using hypertext systems in the past. This is due to the fact that such systems are considered sophisticated programs while hypertext was, in the past, not easily associated with procedural programming. As this contrast vanishes (e.g., because of Java), a chance for unifying all kinds of learning systems under one common (XML based) technology arises.

Requirements: subject-model based WLS should also be built using XML based technology, rendering the subject model sharable and re-usable for other WLS types and making modules related to the WLS type re-usable for other subject models.

Motivational-Model based WLS: for this WLS class, the Web-related advantages have already been discussed: the mere use of the Web has some motivational value;

more important, Web based group games provide an excellent starting point for learning-specific approaches.

Requirements: towards the use of true open Web standards and towards a wide-spread use of group fun learning games, there are several major obstacles, all of which have however been mentioned in already. Again, re-usable WLS types (here: re-usable game engines) and re-usable subject models are fundamental requirements. They require the use of public Web standards, multi-layer hypertexts, and highly modular approaches.

Tools-Based WLS: From the quest for constructivism, we derived a quest for learning systems based on Web exploration. Human guidance has been found both feasible and desirable, guidance based on learner models was found desirable but unfeasible with current Web technology (except at the cost of drastic restriction of the openness). Cooperative exploration is to be offered as far as possible in the attempt to support development of orthogonal skills.

Requirements: if (cooperative) Web exploration methods are to be combined with program-driven, model-based training, information on the Web must be organized such that semantics of the information provided becomes machine-readable in a generally agreed way. While this sounds utopian, corresponding efforts are under way as we will see below. Another important requirement is the customizability of groupware, cognitive, and work organization tools to specific domains (here: the subject matter), both for improved efficiency of tools-based WLS and for improved guidance (which must again be based on learner models and subject models).

3.5. Requirements vs. XML-based technology

The most important requirements listed above are now related to some key developments related to XML which are available or under way.

1. XML-based multimedia i.e. hypertext standard compliant modular multimedia support: most present WBT tools use proprietary solutions for multimedia presentations which embrace all media and their scheduling i.e. playout strategy (cf. Macromedia Shockwave™ / Flash™). The individual media are not accessible as 'nodes' of the hypertext via 'arcs' (HREFs in HTML terms). While HTML offers no suitable solution here, the XML community has developed the SMIL [12] standard for the definition of multimedia presentations whose components are XML based nodes.

2. Learning- and subject-related metadata standards i.e. Modularity and re-usability of learning objects and subject-related objects. As to learning-related objects, the LOM, Ariadne, and IMS initiatives [3] propose meta models and ontologies for learning-related data and objects. This is an important step towards building re-usable components of authoring environments. As to subject-related information (concept graphs, etc.), efforts for the definition of ontologies are under way for many domains, virtually all of them based on XML. They also lead the way for machine-readable semantics of subject-related information in an open i.e. unconstraint Web space (as stated as a pre-requisite for true tools-based Web exploration).

3. Re-Usable hypertext/engine types i.e. models for the construction of constraint hypertexts and for the re-use of (modules of) 'navigation engines'. The term 'navigation engine' here relates to any WLS software which operates on top of' a hypertext (such as a hypertext-based ITS system or simulation). A problem often underestimated by the scientific community working on 'learning standards' is the fact that re-usable objects do not standardize hypertext structures and even less make re-usable 'engines' that would operate on such structures. To this end, fig. 2 shows an example of a so-called WebStyle [6] taken from a project headed by the author. For a simple example (bibliographies), a re-usable hypertext construction module is shown which describes a family of hypertext components (interrelated nodes and links). While the actual number of nodes and links remains open (to be determined according to the actual rhetoric (sub-)space built), required and optional node and link types and their interrelation are specified. The actual hypertext is derived from the WebStyle via successive instantiation of nodes and links. Fig. 2 shows a screen-shot of the original WebStyle (left) and a hypertext-under-construction after a number of instantiation steps. The same project also emphasizes re-usable engines whose operation is defined in relation to WebStyles, so that they can be combined with any hypertext constructed in compliance with that WebStyle. XML based technology offers a number of pre-requisites for (or alternatives to) WebStyles not offered in HTML, such as (application specific) typed nodes and links, bi-directional links separation of links and anchors (cf. Xlink and Xpointer [1, 2]), and RDF for the definition of elementary node-link relations, and others [8, 9].
These XML-related features, together with XML-related software engineering approaches (WebDAV [4] etc.) and the above-mentioned meta-data and ontology efforts (cf. items 1 and 2), may be combined with WebStyles or a similar approach to support true re-usability of models, domain descriptions (i.e. multimedia or WBT modules), and engines for model-based WLS. In order to support the combination of such models (subject / learner / motivational) and domains, such an effort must include support for *multi-layer hypertexts* in the sense of different interrelated hypertexts.

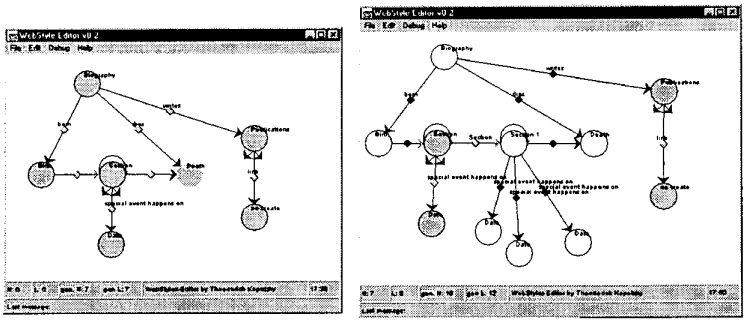

Fig. 2: WebStyle and corresponding hypertext

Note that the vision described in this item is not yet readily available today, but proven to be feasible based on XML-related technology.

4. Adaptive Web exploration tools: In the last section, the quest was made for tools-based WLS which support open Web exploration – either human- or machine-guided, at best cooperative. A first pre-requisite for this vision was discussed in item 2 (domain-specific metadata and ontologies). Another pre-requisite relates to sophisticated guidance: the guiding WLS or human must be able to monitor or control how the learner(s) used the (cognitive, groupware, work-organization) tools for accessing the (now 'understandable') information on the Web. This monitor/control activity as well as the task of the learners can be drastically improved if the tools can be customized to specific subject matter domains. One possible approach is to develop all tools (e.g., tools for structured Web queries, for time management, for rhetoric spaces, etc.) as 'navigation engines' in the sense described in item 3.

5. Summary

Web-Based Training today is by and large restricted to descriptive Web Learning Systems, which in turn suffer from an obvious CBT legacy. Available WBT authoring tools were classified in this paper and it was shown that they support descriptive-WLS development with a high degree of sophistication. Beyond this WLS class, this article called for a mix of (Learner, Subject, and Motivational) Model-Based WLS and Tools-Based WLS to yield highly effective solutions. It was shown that XML-based technology, in contrast to HTML, is a good basis for developing next generation authoring systems which support these additional WLS types.

6. References

[1] R. Daniel, S. DeRose, E. Maler, *XML Pointer Language (XPointer) V1.0.* W3C Recommendation, June 2000. (See http://www.w3.org/TR/xptr.)

[2] S. DeRose et al., *XML Linking Language (XLink) 1.0,* W3C Recommendation July 2000, (See www.w3.org/TR/xlink/)

[3] *Draft Standard for Learning Object Metadata,* IEEE Learning Technology Standardization Committee, March 2000 (See http://ltsc.ieee.org/doc/wg12/LOMv4.1.pdf)

[4] F. Dridi, G. Neumann, "How to implement Web-based Groupware Systems based on WebDAV", Proc. WETICE '99, IEEE 8th Intl. Workshop on Enabling Technologies: Infrastructure for Collab. Enterprises, Stanford, CA, June 16-18, 1999.

[5] I. Harel, S. Papert (Eds.). *Constructionism,* Norwood, NJ: Ablex, 1991.

[6] R. Hauber, T. Kopetzky, M. Mühlhäuser, "Lifecycle Support for Hypermedia Based Learning", in: *Educational Multimedia and Hypermedia Annual,* 1998. AACE Charlotteville, VA, pp. 484-489.

[7] D. Jonassen (Ed.), *Handbook of Research for Educational Communications and Technology.* Macmillan, New York, 1996:

[8] M. Klein et al. "The Relation between Ontologies and Schema-Languages: Translating OIL-Specifications to XML-Schema" In: *Proc. Workshop on Applications of Ontologies and Problem-solving Methods,* 14th European Conference on Artificial Intelligence ECAI-00, Berlin, Germany Aug. 20-25, 2000.

[9] O. Lassila, "RDF Web Metadata: A Matter of Semantics", *IEEE Internet Computing,* Vol. 2, No. 4, July/Aug. 1998

[10] T. Reeves, "A Research Agenda for Interactive Learning in the New Millennium", Proc. Ed-Media '99, Seattle, WA, USA; AACE, Charlotteville, VA, 1999.

[11] L. Rieber, "Seriously considering play: Designing interactive learning environments on the blending of microworlds, simulations, and games", *Educational Technology Research & Development,* Vol. 44, No. 2, 1996, pp. 43-58.

[12] *Synchronized Multimedia Integration Language (SMIL) 1.0 Specification,* W3C Recommendation June 1998 (See http://www.w3.org/TR/REC-smil/)

[13] B. Wilson, M. Lowry, "Constructivist Learning on the Web", in Liz Burge (Ed.), *Learning Technologies: Reflective and Strategic Thinking.* Jossey-Bass, San Francisco, 2001. Web: http://ceo.cudenver.edu/~brent_wilson/WebLearning.html