CAPE: A Visual Language for Courseware Authoring

Position Paper

Second Workshop on Domain-Specific Visual Languages

Larry Howard, Institute for Software Integrated Systems, Vanderbilt University
howardlp@isis.vanderbilt.edu

Abstract

Applications of technology to teaching and learning at school, home, and the office are increasing at a rapid pace. There are legitimate concerns that the demand for courseware is already exceeding the supply—a situation that, at least in part, reflects the current practice of custom-crafting education and training applications. This situation has motivated interest in alternative development strategies that involve "repurposing" content. Inherent in such strategies is the ability to address the adaptation and integration of learning materials and activities as a distinct aspect of design. We describe research on an authoring environment called CAPE that addresses this aim, among others, by representing instructional designs using a domain-specific visual formalism. We further describe issues and activities surrounding the creation and use of such representations.

Introduction

Traditionally, instructional designers represented their work as sets of teaching materials combined with descriptions of learning objectives and guidance on how the materials might to be used to achieve and assess intended outcomes. These things were packaged together for use by a teacher, who might tailor the design and materials to a particular use and then enact and assess the learning experience. In short, instructional designs were a human-to-human communication fully intended to be interpreted by a teacher in the context of the targeted learning situation.

Early applications of computing technology to learning were typically "designed-to-fit" artifacts. The advent of powerful multimedia authoring technologies, associated with the world-wide web, have enabled the creation of more compelling learning experiences, but the result has been custom applications that are expensive to produce and afford very little opportunity for *re-purposing*; that is, the reuse of the constituent materials in related learning experiences.

Increasing demand for computer-based learning applications has engendered interest in a new development strategy, increasingly referred to as *reusable learning objects*. In this approach, well-circumscribed elements of instruction are created and then integrated to form larger instructional units. Evidence of this interest can be seen in the production of a standard courseware representation called the Shareable Content Object Reference Model (SCORM) by the Advanced Distributed Learning Initiative[1], a collaboration involving government, industry, and academia whose primary sponsor is the U.S. Department of Defense, the world's leading acquirer of courseware. Representations such as SCORM begin to make it possible to pursue a reuse-based approach to creating computer learning experiences, although clearly many technical and non-technical challenges remain to be addressed before such a strategy is practicable.

Early in a project for the National Science Foundation's Engineering Research Center on Bioengineering Educational Technologies (VaNTH)[2], we recognized the opportunity for an authoring technology that could be used to assemble modular educational materials into bioengineering courseware. We further recognized that it would be advantageous to have this authoring capability produce a standards-based courseware representation, since even among the member institutions of the ERC there would likely be differences in choices of courseware delivery platforms. We called this new technology the Courseware Authoring and Packaging Environment (CAPE).

[1] http://www.adlnet.org

[2] VaNTH is a multi-institutional center involving Vanderbilt University, Northwestern University, the University of Texas at Austin, and the Harvard-MIT Division of Health Sciences and Technology. (http://www.vanth.org)

Since SCORM is the standard representation most relevant to the aim of composable learning materials, we provisionally adopted it as a target representation. But SCORM is a general-purpose XML-based representation, and we were interested in an authoring language that was easier to create and closer to the domain of authoring bioengineering courseware for VaNTH. The author's home institution, ISIS, has pioneered a paradigm known as model-integrated computing [1] in which domain-specific modeling representations are used to automatically generate the input representations for analytical engines and data-driven run-time environments. It was, therefore, natural to consider the possibility of a visual language for authoring VaNTH courseware from which a textual representation such as SCORM could be generated. An investigation of this design strategy was begun.

Today, CAPE has grown into a representation incorporating nearly a hundred distinct concepts and relationships. Its evolution has been largely opportunistic, using needs and interests from pioneering users and reflections on their experiences as motivations for extensions and refinements. We continue to view CAPE as more a vehicle for exploring the possibilities and challenges of using a domain-specific visual language within the learning objects strategy, and less an attempt to arrive at a definitive representation. It is hoped that understanding gleaned from our experiences will help inform the evolution of standards-based courseware representations and their associated delivery platforms. However, our primary aim is establishing a robust authoring capability serving the distinct interests of the VaNTH ERC.

The remainder of this paper presents and discusses our experiences developing and using CAPE in three sections. The first addresses the "meta-design" of the CAPE authoring language; that is, the design of the design representation. The second addresses the environment that supports authoring tasks using this representation. The third and final section addresses the delivery infrastructure that supports the use of the resulting courseware by learners.

1. The Visual Language

Using CAPE, courseware authors primarily address three interrelated sets of concerns:

1) Integrating learning materials into instructional units at various levels of aggregation and determining in what sequences and under what conditions materials are delivered to learners
2) Establishing learning objectives for instructional units and associating them with elements of content (or domain) knowledge
3) Providing metadata that describes the instructional units to instructors, to learners, or to the delivery infrastructure

VaNTH has defined an aggregation model that provides four levels:

- *Granules*

  These are atomic content elements that provide basic resources for authoring. Granules can be used within all higher-level aggregations.

- *Modules*

  These are the basic instructional unit, where learning objectives address teaching sets of interrelated domain concepts.

- *Mosaics*

  These are compositions of modules with some unifying theme or challenge.

- *Courses*

  These are compositions of mosaics or modules intended to provide coverage of some body of knowledge from a bioengineering curriculum.

VaNTH granules correspond to the SCORM Content Aggregation Model (CAM) concept of "assets". VaNTH modules, mosaics, and courses correspond to SCORM's "shareable content objects".

In devising a meta-representation for CAPE, we used concepts from SCORM as a starting point. CAPE's meta-design was captured using a meta-modeling facility[2] of the Generic Modeling

Environment (GME)[3] developed by ISIS. This representation is based on the Unified Modeling Language (UML)[4] and Object Constraint Language (OCL)[5]. Within GME's meta-modeling facility, meta-designs are organized into definitional units called paradigm sheets.

Figure 1 shows the paradigm sheet corresponding to the core courseware representation for CAPE. It defines the VaNTH aggregation model, together with a set of simple delivery sequencing concepts influenced by SCORM's precondition predicates: sets and logicals. It also provides for references to granules and content aggregations that support incorporating these items from folders of reusable resources.

Other definitions established by the *CoursewareCore* paradigm sheet establish three "aspects" for courseware designs[3] that parallel the sets of concerns identified earlier. The first aspect, *Structure*, addresses the integration and sequencing of learning content. This aspect is used to define the order in which materials or activities are presented to learners. The second aspect, *Objectives*, concerns establishing learning objectives and associating these with concepts from taxonomies of domain knowledge from the disciplines constituting bioengineering. The third aspect, *Metadata*, is used to associate metadata tags with a courseware model and provide values for these tags. The paradigm sheet also provides definitions of constraints governing the creation of the modeling representation and definitions of attributes for the constituent modeling concepts and relationships.

As VaNTH authors began to experiment with the representation defined by *CoursewareCore* it quickly became apparent that there were needs that transcended its reach. First, while SCORM is targeted to asynchronously delivered courseware, VaNTH is interested in combining in-class and out-of-class experiences in an approach called "blended instruction". There needed to be concepts addressing coordination between these learning contexts. VaNTH authors were interested in adapting learning experiences to individual learners, and SCORM's concepts for conditional, or adaptive, delivery were widely acknowledged to be underdeveloped. Concepts were needed for authoring interactive content with the ability to use interactions as a basis for adaptation.

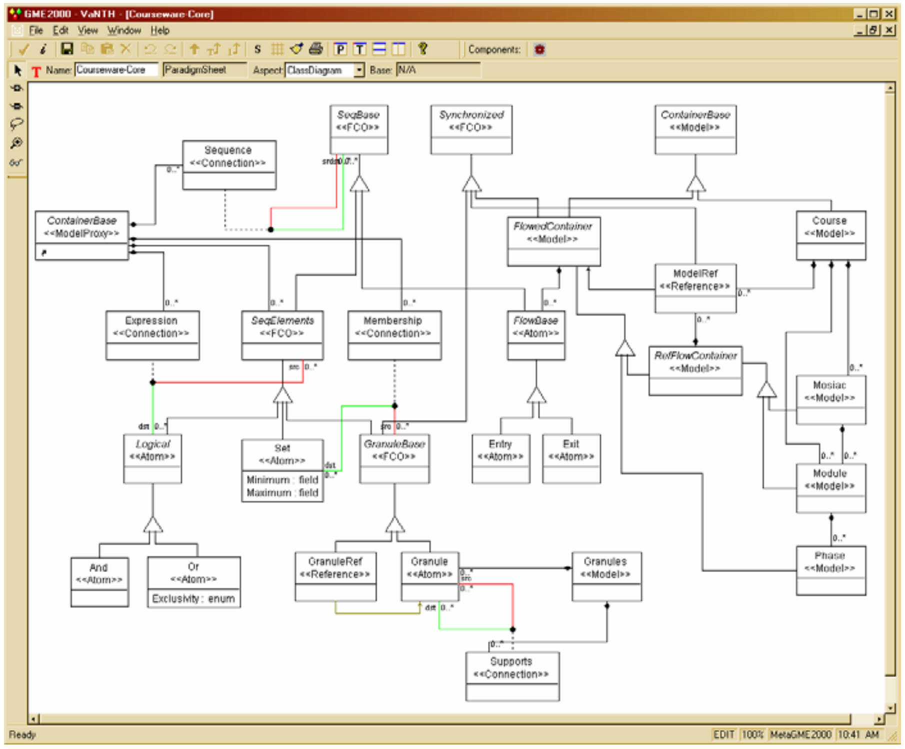


Figure 1: An Element of CAPE Meta-Design

Additional needs arose in pursuing an approach to forming relationships between learning objectives and taxonomies, which VaNTH uses to represent domain knowledge within its bioengineering sub-domains, that was somewhat different to the approach taken by SCORM.

In answering all of these needs, an additional set of concepts and relationships were defined

[3] Aspects support a separation of concerns in a modeling representation and are interrelated through shared concepts and relationships.

through another paradigm sheet called *CoursewareAnnex*. These VaNTH-specific extensions were composed together with the definitions provided by *CoursewareCore* using composition capabilities of GME's meta-modeling facility[2]. This approach preserved the SCORM-influenced portion of the CAPE meta-design so that if authors did not require any of the VaNTH extensions, the resulting courseware could be represented for delivery using a SCORM-compliant delivery platform. But the use of these extensions would require a VaNTH-specific delivery platform. This platform is discussed later in Section 3.

Other definitional units of CAPE's meta-model address creating collections of descriptive resources, the authoring of assessments, and a general-purpose data definition facility that supports the integration of learning technologies, such as simulations, diagram editors, or problem-solving environments, among other uses. VaNTH's domain taxonomies and SCORM metadata tags are examples of collections of descriptive resources. These are represented through sets of tag types, and instances of these types are used to create palettes of tagging resources. Authors drag instances of these tags onto a modeling canvas and associate them with other model elements. Attributes are specified, where required, to complete tag instantiation. Assessment authoring concepts for CAPE were influenced by the IMS Question and Test Interoperability standard.[4] These permit CAPE authors to create quizzes that are rendered as HTML forms by the delivery platform. Learner responses can be used in adaptive delivery strategies, with the authoring representation supporting referencing these responses in delivery predicates associated with the conditional delivery concepts: *Select* and *Condition*. The first of this is a multi-way branching concept with delivery predicates associated with the graph edges. The latter is a binary test-and-branch concept with the test condition associated with the node and branching via *True* and *False* edges. The use of these capabilities in conjunction with assessments can be seen in Figure 2.

A simple synchronization concept is used to coordinate asynchronously delivered learning experiences with in-class activities. The concept introduced into the authoring representation provides a kind of barrier synchronization that enables asynchronous delivery to be suspended until some synchronous event occurs, such as a lecture or lab. These barriers can be time-triggered or explicitly opened by an instructor using services of the delivery platform.

Other examples of VaNTH-specific authoring concepts include context-specific help resources

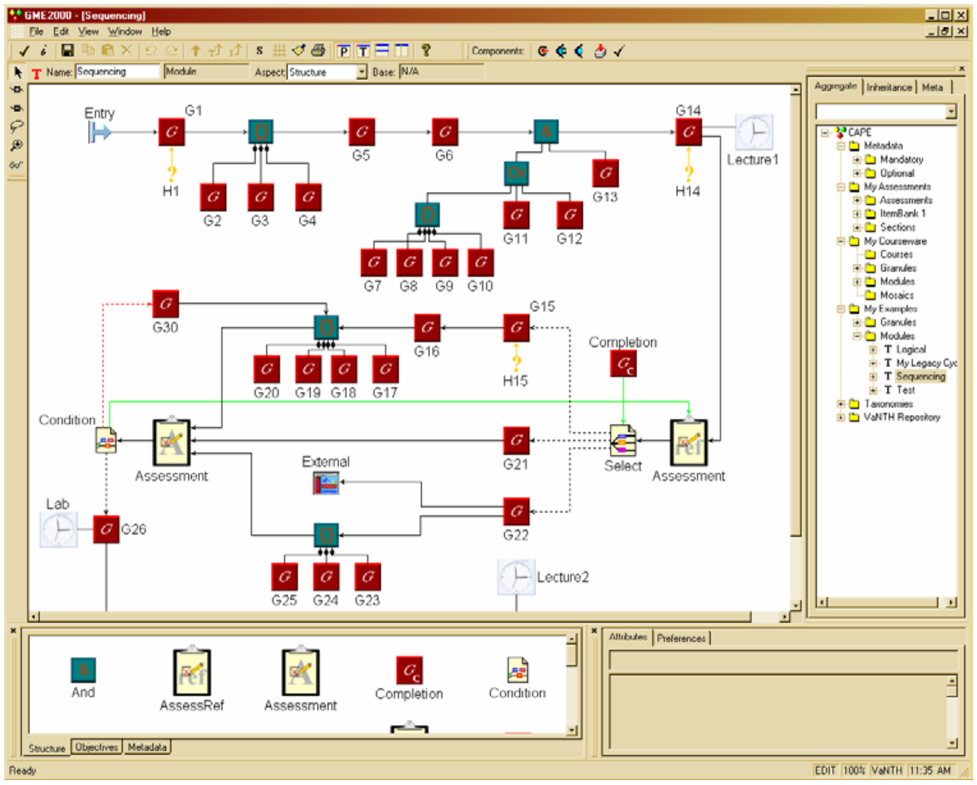


Figure 2: Delivery Sequencing in CAPE's *Structure* Aspect

[4] http://www.imsproject.org/question

associated with granules, incorporating external web resources into a courseware delivery sequence, and execution coordination and data interchange with embedded learning technologies. For the sake of space, we will not discuss these concepts and their use in authoring here.[5]

2. The Authoring Environment

The CAPE authoring environment is automatically generated from the meta-design representation, or paradigm, created in GME's meta-modeling facility. The generated environment provides a powerful abstraction facility and is extended with paradigm-specific tools that support the authoring task and that create the output representation(s). Such extensions are called *model interpreters* and can be implemented in any language that supports Microsoft's COM component integration technology. In this section we will describe uses we have made of the abstraction facilities and the set of model interpreters that have been created to support courseware authoring in CAPE.

VaNTH is interested in the disciplined application of learning science in the creation of its learning experiences, particularly the HPL Framework[6]. In considering the design of the CAPE authoring environment, we were likewise interested in how the insights from learning science might inform the authoring task. We observed that many times these insights could be represented as canonical design forms that captured recurring pedagogical strategies that could serve as the starting point for instructional designs. These could be represented using abstraction facilities providing by the GME. These facilities enable any model to be used as a "type", with the ability to create derivative types (or subtypes) and instances. We could represent *instructional design patterns* as abstract models using these capabilities and authors would refine these to form instances with particular learning content.

Figure 3 shows a simple example of the use of this strategy. The pattern is for a set of tutors addressing vector arithmetic. The pattern provides the form of an activity that provides the learner with three attempts at a solution before providing the solution to them. A correct solution at any attempt takes the learner to an acknowledgement that they have succeeded (the *Correct* granule). Otherwise the learner proceeds to successive attempts (phases) until finally receiving the solution (the *Solution* granule) after the third phase.

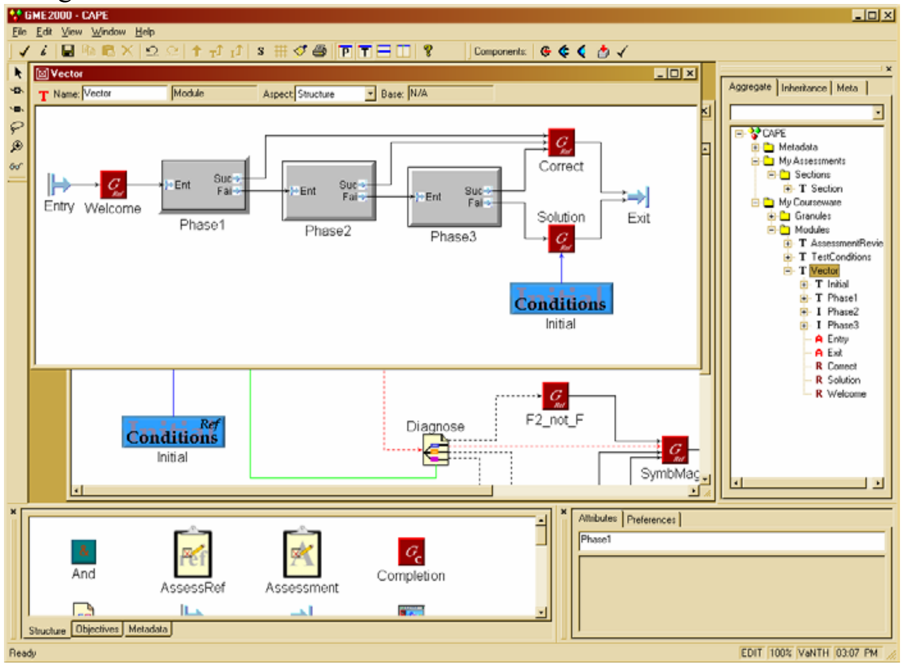


Figure 3: An Instructional Design Pattern for a Math Tutor

A further use of abstraction is employed in modeling the diagnostic procedure for each phase of an instance of the pattern. The diagnostic procedure consists of matching the learner's responses to set of recognized problems. The procedure is consistent among the phases, but the remediation is progressively more detailed in later phases. We used GME's

[5] More information is available at the Courseware Authoring Technology (CAT) Project web site. (http://www.isis.vanderbilt.edu/projects/VaNTH)

abstraction facility to make the diagnostic procedure in the first phase a type and the subsequent phases its sub-types. The only variation among the phases is that the references to granules providing the remediation point to remedial content appropriate to the phase. This approach provides a single point of modification to the diagnostic procedure.

Still other innovations in this tutoring pattern involve the use of a ConditionSet, CAPE's data modeling facility. The *Initial* ConditionSet is used to parameterize the math problem being posed, with the ability to randomize initial conditions. This presents each learner with a slightly different variation of the problem. While this approach is fairly common (see, for example, the CAPA system [7]) we have pioneered the use of random conditions in dynamically generating the remediation, instantiating formulae with the randomized initial conditions in presenting the particular solution to the learner and the use of the initial conditions in dynamically driving a problem visualization.

Finally, instructional design patterns provide the opportunity to assemble together descriptive resources that support the pattern. For example, sets of metadata can be specialized and included in the pattern and inherited by subtypes and instances. This requires only those metadata tags that are specific to the use of the pattern be specified, rather than requiring a complete metadata description of each instance.

Several extensions (model interpreters) have been created to support the authoring task in CAPE.

- The *ContentImporter* can be used to automatically build content metadata for granules. This is especially useful when employing de-aggregated content produced by a content authoring tool such as Powerpoint.
- The *ContentPreviewer* allows the learning content represented by a granule in CAPE to be launched in a browser with the click of a button. This is useful when there is some uncertainty about what content the granule represents. It is also useful when browsing folders of granules created by others.
- The *DeliveryPreviewer* enables authors in enact the courseware being authored within the authoring environment to assure that the authoring task has been performed correctly.
- The *ContentPackager* creates the target representation of the courseware artifact and optionally uploads it to the delivery platform.

Other extensions support coordinating the modeling representation of VaNTH domain taxonomies with a relational database which supports taxonomy authoring tools. CAPE can also be used to author and modify taxonomies. Currently under development are capabilities that will allow CAPE to interoperate with a centralized web-based repository of VaNTH learning materials.

3. The Delivery Platform

To support the delivery of courseware authored with CAPE using the full set of VaNTH extensions, we created the Experimental Learning Management System (eLMS). This delivery platform is "experimental" in two senses. First, the platform is extensively instrumented, enabling its use as a vehicle for experimentation with asynchronously delivered VaNTH learning experiences by students. These capabilities support the research mission of the ERC. Second, since VaNTH is pioneering new concepts and capabilities in its authoring technology, solutions must be found for enacting these capabilities and eLMS provides a vehicle for experimenting with such solutions.

Particularly with the latter needs in mind, we chose to base the eLMS platform on an adaptable web application framework called Zope.[6] Zope[8] is an open source framework implemented in the Python language. In addition to extensibility in this dynamic language, Zope provides novel dynamic content

[6] http://www.zope.org

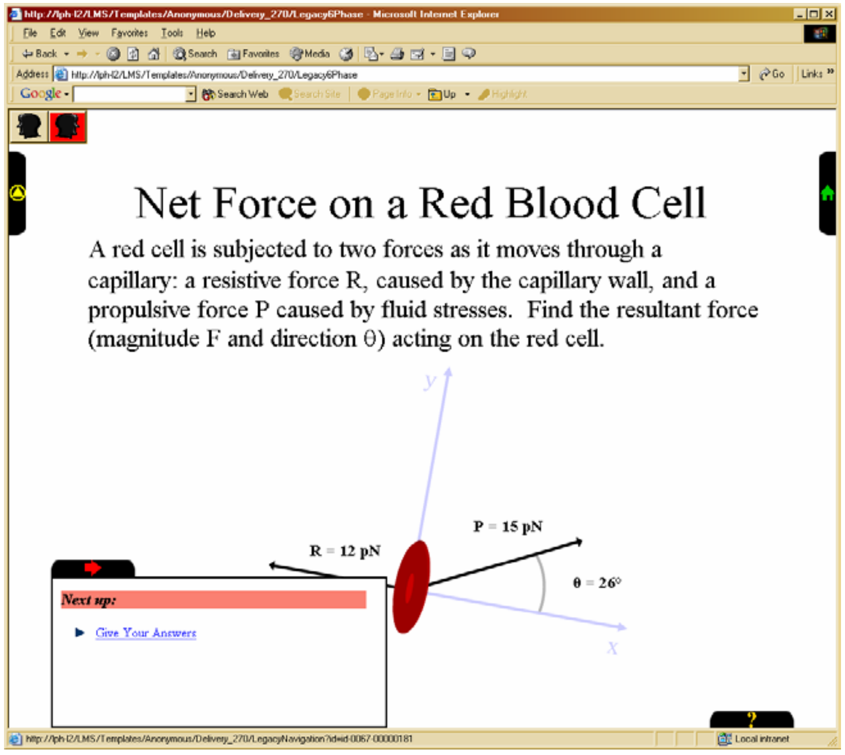
capabilities through its Dynamic Template Markup Language (DTML) and the newer Zope Page Templates (ZPT). Zope is backed by an object-oriented database called ZODB that enables an object-oriented approach to constructing large web applications.

The heart of the eLMS platform is a model-based delivery engine. This engine uses the courseware models created in CAPE as instructions for enacting learning experiences. Further, a run-time representation of these models is persisted in ZODB for each student delivery and is "decorated" with artifacts produced by learners to create the record of the use of the materials. The eLMS delivery engine is extensible and can be specialized with delivery semantics associated with instructional design patterns used for authoring in CAPE. This is accomplished using *delivery templates* that are the enactment counterparts of the design patterns.

Web services for the eLMS platform are implemented in Python and can be accessed using HTTP requests or XML-RPC. A novel Zope capability called *acquisition* allows web services to be contextualized by object identified by the called URL. This capability is used extensively by the eLMS delivery engine in invoking delivery-related web services using the context of a particular student delivery. Web services also support the integration of embedded learning technologies in web-based VaNTH courseware. Services are provided for execution coordination and data interchange, including persistence of outcomes in the student's delivery record.

The courseware delivery interface of eLMS platform is another area where we a pioneering new capabilities. Rather than consuming browser "real estate" with navigation and other kinds of information and features for the learner, pull-in accessories are used to support extending the interface. These can be seen in Figure 4 as tabs along the browser border. The *Navigation* accessory (bottom left) is shown extended. Using the approach of pull-in accessories we can provide a large amount of interface functionality without overly intruding on the instruction. Accessories can be associated with a delivery template to provide interface extensions specific to an instructional design pattern—a specific review accessory, for example.

While eLMS is primarily a vehicle for exploring delivery issues arising in the development of the CAPE authoring technology and experimenting with the use of VaNTH courseware by learners, it must nonetheless provide more conventional capabilities found in a production LMS. These include the abilities for teachers to form classes, assign courseware, and review student delivery



Net Force on a Red Blood Cell

A red cell is subjected to two forces as it moves through a capillary: a resistive force R, caused by the capillary wall, and a propulsive force P caused by fluid stresses. Find the resultant force (magnitude F and direction θ) acting on the red cell.

Figure 4: The eLMS Courseware Delivery Interface

records. For authors, capabilities are provided to upload and update courseware. Discussion forums are supported and learners can create private notes for courseware and classes. These capabilities are currently at various stages of development for eLMS.

Summary

We have described a domain-specific visual language called CAPE used for authoring web-based learning experiences. Our seminal motivation was to support a compositional approach to courseware authoring using adaptable learning materials. Our early experiences suggest that a visual language provides an effective approach to assembling and sequencing materials, and for providing necessary descriptive information. Our experiences further suggest that currently available standards-based courseware representations need additional capabilities to support the kinds of courseware under development by the VaNTH ERC.

We have described the use of design abstraction facilities of the modeling environment technology upon which CAPE is built that can be used to capture reusable designs informed by learning science research as starting points for authors. Early evidence suggests that design patterns are a very potent concept for instructional design, just as they have proved valuable for software design. The extensibility features of GME have been effectively used in providing domain-specific tools that support the authoring task in CAPE.

Finally, courseware delivery is the flip side of the courseware authoring "coin", and we have by necessity engaged in exploring capabilities and features of delivery platforms that support the enactment of the courseware designs authored in CAPE. Here, the concept of delivery templates associated with instructional design patterns has proven quite powerful in building an extensible delivery engine. Interface design innovations in the eLMS platform are pointing the way towards feature-rich interfaces that don't intrude on the instruction provided through the interface.

It is hoped that our experiences will contribute to the evolution of standard-based courseware representations that support the reusable learning objects strategy and future delivery platforms that will support these representations. To that end, VaNTH is become a member of the Academic CoLaboratory of the ADL and is actively tracking other standards efforts.

References

[1] Sztipanovits J., Karsai G.: *Model-Integrated Computing*, IEEE Computer, pp. 110-112, April, 1997.

[2] Ledeczi A., Bakay A., Maroti M., Volgyesi P., Nordstrom G., Sprinkle J., Karsai G.: *Composing Domain-Specific Design Environments*, Computer, pp. 44-51, November, 2001.

[3] Ledeczi A., Maroti M., Bakay A., Karsai G., Garrett J., Thomason IV C., Nordstrom G., Sprinkle J., Volgyesi P.: *The Generic Modeling Environment*, Workshop on Intelligent Signal Processing, Budapest, Hungary, May 17, 2001.

[4] *UML Semantics*, ver. 1.1, Rational Software Corporation, et al., Sept. 1997.

[5] *Object Constraint Language Specification*, ver. 1.1, Rational Software Corporation, et al., Sept. 1997.

[6] Bransford, J. D., Brown, A. L., & Cocking, R. R. (1999). How people learn: Brain, mind, experience, and school. Washington, DC: National Academy Press.

[7] D.J. Morrissey, E. Kashy, and I. Tsai*: Using Computer-Assisted Personalized Assignments in Freshman Chemistry*, J. Chem. Ed. **72**, 141 (1995).

[8] Latteier A., Pelletier M.: The Zope Book. New Riders, July 2001.

Acknowledgements

This work was supported primarily by the Engineering Research Centers Program of the National Science Foundation under Award Number EEC-9876363.