World Academy of Science, Engineering and Technology
International Journal of Educational and Pedagogical Sciences
Vol:2, No:3, 2008

Component Based Framework for Authoring and Multimedia Training in Mathematics

Ion Smeureanu, Marian Dardala, and Adriana Reveiu

***Abstract*—**The new programming technologies allow for the creation of components which can be automatically or manually assembled to reach a new experience in knowledge understanding and mastering or in getting skills for a specific knowledge area. The project proposes an interactive framework that permits the creation, combination and utilization of components that are specific to mathematical training in high schools.

The **main framework's objectives** are:

- authoring lessons by the teacher or the students; all they need are simple operating skills for Equation Editor (or something similar, or Latex); the rest are just drag & drop operations, inserting data into a grid, or navigating through menus
- allowing sonorous presentations of mathematical texts and solving hints (easier understood by the students)
- offering graphical representations of a mathematical function edited in Equation
- storing of learning objects in a database
- storing of predefined lessons (efficient for expressions and commands, the rest being calculations; allows a high compression)
- viewing and/or modifying predefined lessons, according to the curricula

The whole thing is focused on a mathematical expressions mini-compiler, storing the code that will be later used for different purposes (tables, graphics, and optimisations).

Programming technologies used. A Visual C# .NET implementation is proposed. New and innovative digital learning objects for mathematics will be developed; they are capable to interpret, contextualize and react depending on the architecture where they are assembled.

***Keywords*—**Adaptor, automatic assembly learning component and user control.

I. Introduction

ONE of the most fascinating domains is that of learning and teaching; besides, the progress of the human society is directly related to the manner of producing, transmitting and acquiring knowledge. Retrospectively looking, it is easy to emphasize the **evolution of learning techniques**, from example-based simple learning, to more sophisticated and abstract mechanisms.

With the development of information technologies, it was natural that the learning process will benefit from these achievements; nowadays, *computer based training* software are so spread, that learning without computer became unconceivable. How true this is, which the computer assisted learning paradigm is, are still major questions. The spreading of the Internet only widens the basis of accessible knowledge, strengthening the dependency between good information and the network.

Internet technologies have the potential to modify not only the way people access information and knowledge, but also to transform and reorganize the traditional models of teaching and learning. Their development is also related to other major changes of today society, facilitating permanent learning, anytime, anyplace, and leading to a greater demand for training in the present knowledge based society.

It can be easily established that the Internet determines the shift of the paradigm, the transition from books to electronic books, from classical lessons to computer assisted lessons; we can state without mistake that there are persons that read better from a monitor screen than from a book. But, does this really change the way people learn? On what depends this, and what are the chances of a major and real change?

E-learning groups methods and techniques, traditional or computer aided (like multimedia processing, asynchronous or synchronous communication, web pages, large databases etc.) assisting the subject in the learning process.

In 1997 Maddux, Johnson and Willis provided a simplified approach for educational software classification; they defined two application type levels:

First level includes software applications which are targeted for an easier, more intense and more efficient delivering of the same knowledge as in the classic method. The user involvement is **low**, the **software pre-determines** almost all that is going on the display; **interaction** between user and computer is **pre-configured** by the software author; user contribution complies with a predetermined scenario; applications aim to acquire knowledge by memorizing.

Second level applications mean new and more efficient learning techniques. They allow more active involvement of the user; **the user controls everything that happens**; user-computer interaction is driven by user at runtime; there is an extended range of inputs and actions accepted by the computer; **creative** actions **prevail**.

In the **first phase**, didactic materials delivered, and the way to access them, are just a complement to the classical manner of learning: electronic books, prefabricated lessons, little reconfigurable or parameterized; in this way, they do not point out the teacher abilities of constructing the lesson; except some little variations obtained by parameterizations or

changes of ambient properties, the lessons look alike to all teachers, and the examples will be the same. In addition, the programming effort and time spending to change the lessons are not insignificant. The classical learning components must be brought to digital format (scanning, OCR, etc.).

In order to create the premises of the transition to **a new phase**, that will restore the teachers' role, we must first answer a key question: does creating a lesson by the teacher himself, need advanced knowledge about computers? Depending on the answer to this question, we will see the chances of using computer in the learning process in a totally different manner.

The component technology opens up an era of a new type of internal communication between objects, from executable to executable, mediated by a virtual machine. This technology brings back the problem of developing and integrating complex applications.

II. E-Learning and Innovation: Component Learning Model

In practice, as in specialized literature, the concept of *learning object* is intensely used, but it is not strictly defined. A **learning object** is defined as any entity, digital or non-digital, that may be used for learning, education or training (IEEE, 2002); it can be reused or referenced any time in a computer-based learning process. Examples of such objects are prints, studies, exercises, texts, audio lessons, courses, curricula etc.

The **learning component** is an object implementing interfaces; these interfaces make it able to recognize other related objects with which it can interact **inside a semantic network**. Functionally, the components are small executable pieces which interact with each other **even during the design time of a client application**; they complete one another, they bring themselves new properties and references through which they connect, "coupling" and acting as a whole (this is where the term component comes from).

The success of new learning technologies is related to the paradigm-shift, from traditional content-centered and instructor-led models towards an interactive focus on the teacher/learner. **Component-based learning is the process of assembling existing software components in an application in such a way that they interact to get a predefined functionality.**

The task of **content management** is partially accomplished by learning components, in a way that innovative digital learning objects can be developed; they are capable to **interpret, contextualize and react depending on the architecture where they are assembled**.

The **semantic model** can be partially supplied by the human subject because learning components can be **assembled manually**, so as to provide a large opportunity both for students and professors to exercise their creativity and vision, and to conceive and develop learning resources by themselves. An important feature of e-learning systems based on learning components is that both **teachers and learners can become active producers of educational content**. Tools for high quality content authoring are already available and anyone with enough creativity can compete in innovation. Component-based education requires active engagement of students' effort over an extended period of time, progressive and innovative. The students experiment, learn from failures, reply to the challenges and become deeply involved. Even the homework can be delivered as component-based activities.

Objects used for learning exist and cooperate at different levels of granularity. We no longer talk of individual objects, but of learning frameworks that can work in two interchangeable modes:

- **author**, when professor and student create and test training applications;
- **reader**, when already authored lessons are experienced.

Learning Objects Properties

Although a large number of specific properties of learning objects can be identified, we will point out only those with direct implications over the practical implementation of this learning technology.

The learning objects are **self-contained**, including information about themselves, which allow them to be independently used in another location where they perfectly integrate. For this purpose, learning objects contain metadata; these metadata facilitate resource identification and indexing for fast searches, for example: title, version, resource type, keywords, author, owned formats and available formats for conversions, accepted facilities etc.

Granularity – reflects the complexity of learning objects and the degree of their decomposition into smaller objects. Although there are no specific limits regarding granularity, few criteria are essential in establishing these:

the **semantic criterion**, expressing that the object must have a self-standing meaning; this does not stop us to use more coupled objects, in order to transmit some sophisticated information. The semantic criterion forces us to design the object so that its parts have an independent meaning, and can be assembled in other ways different from the original manner. If this can be achieved, then the granularity can probably be reduced, creating the possibility of detecting new meanings, by finding other ways of recombining the parts.

the **financial criterion**, according to that, the objects dimension and complexity depend on the resources needed to create them;

the **efficiency criterion**, meaning that an elementary level will be chosen, that assures the message or knowledge deliverance, in an appropriate relation with the effort of creating the object.

Directly related to the granularity, two other properties can be revealed: **composability** and **decomposability**, implying the possibility that a complex object can be decomposed into elementary objects and recomposed, eventually following other rules. This is exactly what a professor does in order to

prepare the teaching material, reusing older lessons, decomposing and recomposing them into a new one, according to a new objective.

The composability is based on the possibility of learning objects to couple, so as to form larger objects; choosing the proper object to teach a thing, and the manner of combining elementary objects to increase the training efficiency, these depend on the teacher abilities; regarding the problem of composing, a good teacher will probably always win in competition with an automated agent.

Finding the most suggestive elements, and coupling them in a personalized manner, individualize each teacher; on the other hand, the learning results can be improved based on feedback information and lesson adaptation to the general level of students' knowledge.

The main questions regarding the learning process are as follows:

In how many ways can the objects be coupled?

- if there are no restrictions of coupling, then who is controlling their order, taking into account that a product is build up of components, that are assembled in certain sequences. This phase, based on the "lego" metaphor, named after the well known game meant to stimulate children creativity, does not seem to satisfy the meaning of learning objects;
- if they are designed to permit a restricted composition, in some specific ways, in predefined structures, then we can use the "atom" metaphor, in which the valences allow limited reactions. What this means for the learning objects programming, depends on each case in particular; a manner of approaching this is probably by using some templates. In this case, the objects are just assembly supporting and empty containers, which are to receive later the adequate content. Can the combination possibility be left to the teacher choice? Are there possible combinations that do not lead to any knowledge gain? In time, can the way of combining be modified, enabling to make new connections between knowledge, which were not foreseen from the start?

There are different ways of structures (Gibbons 2000), depending on the criterion basis of the organisation:

- the **media-centred** structure, regarding the medium delivering the information: books, pages, films etc.
- the **message-centred** structure, the central idea being to deliver the information faster, using already acquired knowledge, analogies of previously known things, past experiences etc.
- the **strategy-centred** structure, where important is the gradual organisation, the central point being the essence of the transmitted knowledge.
- the **model-centred** structure, where the construction follows an interactive model and its relations with the extern medium, and the finality consists in evaluating the performance, while acquiring knowledge.

Another persistent problem refers to the way of controlling the design, taking into account that the aesthetics plays a central role in the learning process (for example, the mental associations between images and memorized things).

The framework has to support two kinds of processes: **the decomposition of learning objects** into their components as well as the **automatic or manual assembly** of these components in real-world applications. It is not enough that learning objects satisfy some formal criteria of coupling/decoupling; the aggregation must also be pedagogically effective. The coupling /decoupling of learning objects is a considerable challenge, mixing ideas from pedagogy and software engineering [4]; the challenge is to attain new significance by composing reusable components; some advantages are revealed in the example below, which is about learning objects for mathematics.

Personalizing the learning process means creating a development plan that is perfectly adapted to the knowledge level, needs, expectations, personal pace and learning habit of the student. Intelligent e-learning systems aim to implement customized learning models; a personalized course can be configured by automatically selecting and sequencing the needed learning components.

Components are running since client application design, therefore they can automate a significant part of component-coupling, in the way that the application requires. Part of the properties is browsable and therefore can be visually modified by mouse right-click actions, without writing code or with minimal programming effort.

Components benefit of abstractization using interfaces, which standardize communication and force components to respect minimal communication rules that are generally accepted (like *IComponent* inherited by an user control, *IDataObject* for drag and drop action).

```
public class EvalExpresCtrl : System.Windows.Forms.UserControl, IDataObject
{
  public EvalExpresCtrl() {InitializeComponent(); }

[Browsable(true),EditorBrowsable(EditorBrowsableState.Always),Category("Custom")]
  public string ExpressionLatex
    { get { return latExpression; }  set {latExpression=value;} }
  public string[] GetFormats(bool autoConvert)
    {
      if (autoConvert) { return new string[] { "Math", "Latex","Bitmap", DataFormats.Text}; }
      else { return GetFormats(); }
    }
  /* ....  */
}
```

By running in a virtual machine the components are no longer platform-dependent; the components adapt themselves

to new versions, inconsistencies due to versioning thus vanishing. Components can be easily reused and rapidly integrated in new applications, without the need of implementation details, only by knowing the coupling interface. A schematic structure of an educational component is represented in the Fig. 1.

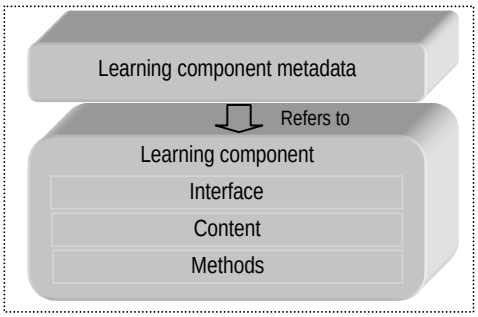

Fig. 1 Learning component structure

Learning objects are *self-contained*, bearing information about them, which allow to be independently dragged in another location where they are perfectly integrated. It is because of this that every learning object encapsulates metadata; these metadata ensure: resource identification (by title, version, and resource type), indexing for fast searches (keywords, author), exposing owned formats or newly available formats using converters, publishing accepted facilities etc.

From the point of view of educational strategies, having a warehouse with such components is not enough; using metadata and interfaces it is very hard to depict all possible semantic relationships because of the diversity of educational processes.

Visual .NET environment offers at the moment, one of the most interesting models for components aggregation and communication. In addition, it facilitates objects storing (ADO.NET), Web forms design (ASP.NET) or distributed services requesting (Web Services). .NET provides a **dynamic publishing and subscribing mechanism**. The .NET components are executables, therefore they know (or they can find about) their interoperability characteristics **at runtime** in a determined context; consequently they can publish their interface. A classic example is a graphing component that can subscribe to a varying number of external mathematical functions or, in his turn, ask for data from a table or dataset object.

III. Practical Approach

A. Motivation

Mathematics uses a specialized and strongly formalized language, which makes difficult understanding the terms or the solution of a problem, presented in books. Using a multimedia product that:

- would allow sonorous presentations of the problems and of the solving ideas
- would offer graphical representations of the mathematical functions, as well as the results
- would present the solution to the problems in an equation manner
- would take over the routine of some calculations

Will allow overtaking the major difficulties of understanding and will satisfy the students need for exercises. We take into account that practical exercises have a greater impact than other forms of learning.

We can easily establish the lack of such an integrated training medium, which will allow the combination of different communication mediums, but also the difficulty to develop it, because it will have to combine the facilities of independent software products, like Mathematica – for calculations and graphics, Latex and Word – for equation editing, Winzip – for compression and efficient storage, indexing and search programs, multimedia tools (audio/video processing software, "try & show" mechanisms etc.).

B. Learning Objects Typology for Training in Mathematics

A telling example can be an application for mathematics training; it raises enough problems so that the definition, the integration and learning object handling will be made clear. A schematic interaction of objects used in such case is represented in Fig. 2.

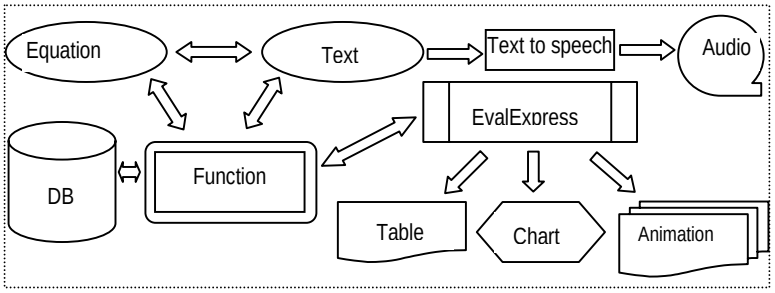

Fig. 2 Component interaction in figuring out the function concept

Learning object diversity is extremely large; we just point out the most representative types of components, in order to easier understand component assembling ways and typical application structures.

A. Content – is the skeleton of an application, offering generic support for an architecture aiming a clear learning goal; usually consists in a hierarchical/graph network for logic navigation among inter-related knowledge sets. It expresses the relationship between learning objects and the syllabus, the course or other higher organizing structure in which they are delivered.

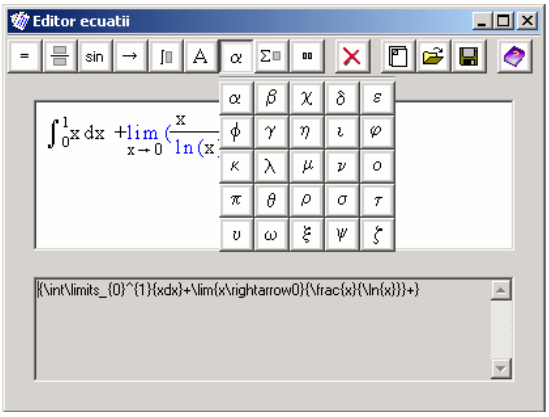

Fig. 3 *Equation* Component (capture)

B. Elementary objects usually placed as leaves in the tree and having specialized editors; Text / Rich Text, Equation, Sound, Graph, Image, Animation, Video are the most used elementary objects.

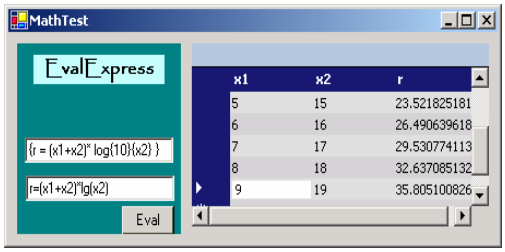

Fig. 4 EvalExpress – Data Grid - Chart interaction (capture)

C. Matching mechanism as an abstract class managing logical associations between objects such as belonging to an object set or one to one/multiple-choice quizzes.

D. EvalExpress – mathematical expression evaluator providing support for **runtime compilation** and work with parameterized functions. The **EvalExpress** acts as a minicompiler, doing syntactic validations, memory allocation and dynamic evaluations, during the whole execution of the client application.

E. Converters and adapters aimed to adapt the outputs to the coupling interfaces, usually calling an overloaded cast operator. The diversity of elementary components requires bringing them to a common format; even in this case, problems remain to be solved, such as: what conversion is preferred when faced with multiple choices, which component initiates conversion at coupling time etc. For instance, the output of **Equation component** working in a visual form, can be a Latex string, easily managed, compressed or re-entered in a visual format for updating; often we need conversions from/to formats accepted by a wide spread editors, like MS Word. Text to speech could be another usual converter.

Another common adapter is a **database adapter** charged with data compression and persistence of the objects. It offers a built-in mechanism for storing the state of an activity or students' work using component serialization. Serialization writes at low-level the binary representation of the .NET component content.

Another usual converter is an **XML converter**; it offers a structured and text-based format for storing and retrieving the state of **a component aggregation** as a support for cross platform portability.

F. Standalone Application – is an entity able to be executed in a standalone play regime on a specific platform. It includes also sub-categories "script" and source application, written in a programming language and becoming platform dependent after compiling and linking.

G. Function – covers a mathematically important data type, a continuous function represented by a method that takes a numeric argument and returns a numeric value. In addition, other attributes of a function are important, such as the domain over which it is defined. The function object offers the possibility to handle mathematical functions by analytical expression or by pointer to a library code; an expression evaluator control allows syntactical validation of the analytical form. Functions can be viewed using multiple components, such as graphs, visualizations or tables**.**

The final goal is to automate the coupling of components, building an adaptor which forces the system to expose **only a set of safe or desired interfaces** for a specific context. By exploiting the metadata and partial specification of the learning goal that must be enforced, it can automatically and progressively build a centralized adaptor. This adapter will mediate contextual interactions among components by both performing the specified behaviour and simultaneously managing possible deadlocks.

REFERENCES

[1] Cleborne D. Maddux, D. LaMont Johnson, Jerry W. Willis, Educational Computing: Learning with Tomorrow's Technologies, Allyn & Bacon; 2001.
[2] Gibbons A. S., Nelson, J., Richards R. *The Nature and Origin of Instructional Objects,* Utah State University, 2000.
[3] Smeureanu, I., Reveiu, A., Dârdală, M., Educational Technologies Based on Software Components, *Informatica Economică*, vol. X, nr. 3, Editura INFOREC, Bucureşti, 2006.
[4] Tom Boyle, Design principles for authoring dynamic, reusable learning objects, in *Australian Journal of Educational Technology*, 2003, 19(1).
[5] Wiley, David, A. Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy, Utah State University, Digital Learning Environments Research Group, 2002.
[6] http://math.hws.edu.
[7] http://portal.acm.org.
[8] http://ocw.mit.edu.
[9] http://www.epsilonlearning.com.