
Educational Software Components of Tomorrow (ESCOT)

Jeremy Roschelle, Chris Digiano, Roy Pea
SRI International

Jim Kaput
University of Massachusetts, Dartmouth

Contact Jeremy Roschelle: jeremy @ unix.sri.com or (650) 859-3049

Abstract

Prior research and development demonstrates that dynamic notations and multiply-linked representations can enable ordinary students to achieve extraordinary learning of scientific and mathematical concepts (Kaput, 1992). However, prior techniques for building such software have resulted in expensive, incompatible, and inflexible products (Roschelle & Kaput, 1996). The vast majority of educational software projects, many of which are funded by the public sector, show promising results in small tests but never reach a larger market (OTA, 1988; PCAST, 1997).

How can sustainable and scalable production of high quality math and science software be achieved?

The Educational Software Components of Tomorrow (ESCOT) project is exploring one promising direction of innovation: assembling math education software from *components* rather than hand-crafting new programs or applications for each curricular need. Typical components include graphs, tables, simulations as well as tools for manipulating geometry and algebra. Using capabilities of the Java language and associated frameworks, it is possible to construct individual educational components so that non-technical authors can flexibly combine them, composing new activities and lessons.

One way to understand the potential of this approach is by analogy. Only a very few people can build their own stereo system by wiring together transistors and other components at the circuit level, but many people can assemble a stereo system to meet their needs by attaching cables to their preferred amplifier, receiver, disc player, turntable and speakers. The move to higher level components has created a much larger marketplace of compatible products that yield flexible arrangements for particular needs. Similarly, ESCOT is raising the level of components for educational software from the programming language level to the pedagogic/curricular level.

In this paper, we report some of the early results from our testbed for developing interoperable components for middle school mathematics, as well as some of the difficult challenges that lie ahead.

Introduction

Systemic reform of K-12 mathematics education is now firmly under way, with reform-based curricular and pedagogical objectives that embody high national standards being implemented at increasing numbers of sites. The National Science Foundation (NSF) catalyzed these reforms by funding standards-based materials development. As intended, these ambitious materials development projects, most of which are reaching publication and implementation stage during the 1997-98 academic year, are now setting the pattern for reformed practice in all grade levels.NSA's investment in materials development not only instantiates the best thinking and research on student learning of mathematics, it also serves as the basis for widespread teacher development work and as a focus for systemic programmatic reform across the nation: statewide, urban, local, and rural.

This remarkably productive national materials development effort did not involve substantial technology, for three reasons: (1) neither practitioners nor facilities embodied sufficient capacity to support everyday technology use by mainstream teachers; (2) consequently, materials requiring substantial technology use could not be commercially viable; and (3) the innovative software products resulting from NSF research lacked interoperability, and thus could not accumulate, integrate, or scale to meet the needs of reformed practice. The net result is that the powerful curricular *materials* foundation for reform lacks a *technological* parallel, despite the wealth of research showing that technology can be a powerful catalyst for reform (see review in SPA, 1997).

The Educational Software Components of Tomorrow Project (ESCOT) is a response to this technological shortfall, beginning with a middle school mathematics testbed and creating a basis for a more extended response spanning all school mathematics and science.Our approach builds on the concept of an "Educational Object Economy" (EOE) (Spohrer, 1997), which combines a *digital library* of reusable, interoperable software resources with a

knowledge network that facilitates distributed, collaborative effort to improve the quality and quantity of resources. By "reusable," we mean that the software can be adapted without the help of the original developers to meet unforeseen needs (Basset, 1996). By "interoperable," we mean that software modules can be mixed and matched (Orfali, Harkey, & Edwards, 1996) to provide more flexible support for learning than can be provided by one module alone. By a "knowledge network," we mean a diverse Internet-based community of experts-- teachers, researchers, developers, and others-- that self-organizes to publish, share, find, critique, and improve software resources and associated materials such as activities and assessment rubrics.

The ESCOT hypothesis is that a knowledge network & digital library of components for K-12 math and science learning can lead to a much more sustainable and scalable approach to developing educational software (Roschelle, Kaput, Stroup & Kahn, 1998). Our grounding for this argument builds upon economic analysis of the potential of reusable software & widespread networking to introduce new business models. In particular, reusable software can lead to a supply chain is longer, more modularized, and more differentiated, which in turn, presages much greater productivity for educational authors (Baetjer, 1998). In addition, the Internet enables the creation of knowledge networks by supporting the formation of communities that share expertise and exchange "objects" (software modules and contextualizing information) at extremely low transaction cost, resulting in "increasing returns" (Hagel & Armstrong, 1997). Under these conditions, economists argue, a "network economy" can emerge and provide exponential growth with only linear initial costs (McKnight & Bailey, 1997).

We are exploring these hypothesis through a testbed of shared software components for middle school mathematics and a community of developers, curriculum experts, authors, and web

facilitators. In this article, we report on early progress ESCOT is making in three areas:

1. **Interoperability:** enabling modules for graphing, geometry, and simulation to be incorporated into a single activity document, and used as one composite tool.

2. **Mapping Curriculum & Technology:** constructing a database of the technology needs of 5 modern curricula for middle school mathematics, and a related database of existing technologies that meet those needs.

3. **Integration Teams:** developing an approach to composing classroom lessons that relies on short-term, focussed teams comprised of a master teacher, software developer, and web facilitator.

Interoperability

Our ESCOT testbed initially includes three anchor technologies, as shown in Table 1 below.

Table 1: ESCOT Anchor Technologies

SimCalc	dynamic graphing and animation tools for understanding calculus ideas
Java Sketchpad	dynamic geometry, based on the top-selling "Geometer's Sketchpad"
AgentSheetsTM	student and teacher constructable simulations

Each of these tools was previously a stand-alone application in a low-level programming language such as C++ or Lisp, with very limited capabilities to coordinate with other tools in the context of

a classroom lesson. Prior to ESCOT, each independent author had developed a Java version of their software, opening the possibility of interoperability. Within ESCOT, we have begun to realize this possibility through three steps:

1. Adapting each component to the JavaBeans and InfoBus specifications. JavaSoft, the organization controlling Java technology, supports these specifications as generic mechanisms for components to shared data and coordinate behaviors.
2. Defining interfaces between components specific to middle school mathematics needs. In particular, we have defined interfaces for some of the mathematical data types that appear in middle school mathematics curriculum, such as a measurement, a time-series of measurements, and a continuous mathematical function. In addition, we have defined some interfaces for pedagogical control of the display, for example, selectively hiding information from the student, and gradually revealing it. These interfaces allow separate components act as linked representations of the same mathematical object, and support dynamic updating as the mathematical object changes.
3. Creating an prototype authoring tool, the MathKit Designer, that allows a non-technical author to choose a combination of components from a set of templates, customize those to the needs of a particular lesson, and publish the combination as a new "applet" on a web page.

The results have been successful; we can now combine these components into a single applet, where they act as if they were a single educational tool tuned to the needs of a particular lesson. For example, Figure 1 shows a tool for exploring the relationship of the area of a circle and a square. This tool combines two separate components, the sketchpad component (from Key Curriculum Press) and the table component (from SRI International). The data in these components is interlinked: as a student drags a red hotspot in the sketch, the geometric figure

changes shape and the rightmost column of the data table updates its values. The table reports the area of the circle and the square, and their ratio, which not surprisingly, is always *pi*.

Figure 1: An activity comprised of the sketchpad and a table component

Efforts at expanding interoperability are on-going. We have plans to add additional anchor technologies to our suite. For example, NIH Image, a powerful tool for measuring and analyzing digital images is available in open source version. We will attempt to adapt this version to ESCOT interfaces. In addition, we plan to expand the testbed through new volunteer members from other research projects which are developing complementary components such as the E-Slate project in Greece. We will also encourage contributors to the Educational Object Economy to use ESCOT components and interfaces. Interested developers and authors of middle school math software are invited to contact ESCOT for information on how to join the testbed.

Interoperability is a challenging problem with many dimensions, of which ESCOT is just addressing a few. Related and complementary efforts include the IMS Project, which is creating specifications for courseware authoring tools and repositories of components, and the IEEE Learning Technology Standards group, which is working on issues such as recording assessment information, linking to intelligent tutors, and defining an overall architecture for learning tools.

Mapping

Another crucial aspect of the ESCOT approach is coordinating the supply of components with the demand (needs) created by new curricula. In particular, we are addressing the needs of five NSF-supported curriculum projects in coordination with their associated implementation group, the Show-Me Center at the University of Missouri directed by Barbara Reys. The curriculum projects are:

1. *Connected Mathematics* published by Dale Seymour/Addison Wesley, development centered at Michigan State University.
2. *Mathematics in Context* published by Encyclopaedia Britannica (under renegotiation), development centered at the University of Wisconsin-Madison.
3. *Middle School Math through Applications Project (MMAP)* unpublished, development centered at the Institute for Research on Learning.
4. *Seeing and Thinking Mathematically* published as MathScape by Creative Publications, development centered at the Education Development Center.
5. *Six Through Eight Mathematics (STEM)* published by Houghton-Mifflin, development centered at Montana State University.

ESCOT is developing a database that describes each of the units in these curricula, and their technology needs. The database is searchable by grade level, conceptual strand, technology type, technology use, and technology functionality. For example, it is possible for a developer to identify all the units that involve graphing. From examining these units, the developer could identify the requirements of the actual curriculum, and use these as guidance in designing an effective graphing tool. Eventually the database will map from the curricular needs to existing software components. This will allow teachers to locate components useful for a particular lesson, and will allow developers to identify unmet needs by finding units with no technological support. (We expect a

draft version of the database to be publically accessible by the time of the M/SET meeting. Contact the authors after March, 1999 for a URL.)

Our early explorations with this mapping database reveal an additional kind of information which needs to be mapped out: activity structures. In order to design effective components, developers need to know not just *what* representations will be used, but also *how.* For example, in a common activity structure, students have to match a target, given incomplete information. This activity structure requires that the components have the capability to provide hide information about the target, and perhaps later reveal it, so the student can confirm their answer. We are working on ways to represent activity structures in future versions of the database.

Integration Teams

If the component interoperability and curriculum mapping objectives of ESCOT are successful, we will still need a process for pulling it all together: composing new activities and lessons that utilize components to meet the needs of the curricula. Our approach emphasizes "integration teams" rather than authoring tools. In the past, software publishers sold authoring tools (e.g., Authorware, Director, HyperCard) with the expectation that teachers would use these to produce a wealth of lessons; indeed, some teachers did so, but far too few. The vast majority of teachers have too little time, more pressing responsibilities, and not enough technical expertise to become independent multimedia authors or programmers. Our experience suggests that it will be more practical for teachers to join short-term teams that include developers who have the necessary authoring and programming experience and Internet facilitators who can support the development of on-line conversations, Web pages, and student

projects. Teachers could contribute their pedagogical experience, share the work of their students, and provide field testing opportunities (Kollock & Smith, 1996) while developers contribute programming experience and debugging capabilities. Such teams will be formed by shared interest, with limited short-term objectives, for example: (a) produce a set of Web pages to introduce the Pythagorean theorem using dynamic Java sketches, then (b) present student work afforded by these tools, and (c) offer discussion areas to assist teachers who use these materials.

ESCOT's integration team approach is being developed by members of the MathForum team (the MathForum is the leading web site of teaching resources for school mathematics). Teams will include master teachers who are involved with systemic reform initiatives, as well as ESCOT component developers. We plan to investigate the integration team approach by building on a successful precedent, the "Ask Dr. Math" and "Problem of the Week" features of the Math Forum. In "Ask Dr. Math," the behind-the-scenes team responds to student math questions with suggested strategies, resources, and ideas for further investigations. The "Problem of the Week" (POW) posts challenging materials from a variety of authors for use in teachers' classrooms. Students receive replies to both correct and incorrect responses to encourage them to rethink their work. At present, these areas of Math Forum contain only pictures and text. ESCOT integration teams will connect these to the middle school math curricula (from Objective 2) and extend these by adding Java component software from our testbed such as simulations, animations, dynamic graphs, interactive notations, and other cognitively and pedagogically appropriate supports for units from the curriculum projects.

Conclusion

According to recent large-scale research studies (Education Week, 1998; Wenglinsky, 1998), technology only has a large scale effect on student learning if it is used in appropriate ways. Simulation, visualization, and exploration top the list of appropriate ways to use software, and dynamic notations and linked representations are primary attributes of these uses. However, it is much more costly to author educational software with these features than it is to produce ineffective drill and practice and page turning applications.The component approach under development by ESCOT and other projects offer a technique for lowering the cost, increasing the reuse, reducing incompatibilities, and enhancing flexibility of educational software. In the future, we look forward to extending ESCOT to a broader range of grade levels, and to include science as well as math.

ESCOT, even if it is completely successful, will only provide solutions at one level of the overall problem of enhancing the sustainability and scaleability of educational software development. Related efforts such as the Educational Object Economy are addressing related issues of community growth, intellectual property licensing and auditing, and integration of commercial and freeware marketplaces. At a lower level, companies such as Sun/JavaSoft are producing basic platform facilities that support componentware. And at a higher level, efforts such as the IMS Project and the IEEE Learning Standards Group are working towards general-purpose specifications for sharing educational objects. All these efforts will need to converge for a new educational component marketplace to emerge and thrive. Should this convergence occur, we can anticipate better quality math and science education software at much lower costs of development.

Acknowledgements

Many ESCOT partners contributed to the work reported in this article. We thank Dave Barnes, Alex Repenning, Nick Jackiw, Steve Wiemar, and Mark Chung in particular for their efforts.

The work reported in this article was partially supported by the National Science Foundation's Knowledge and Distributed Intelligence (KDI) Program, Award REC-9804930. The opinions expressed are those of the authors, and may not reflect those of the funding agency.

References

* Baetjer, Howard Jr. (1998). *Software as capital: An economic perspective on software engineering.* Los Alamitos, CA: IEEE Computer Society.
* Basset, P. (1996). *Framing software reuse: Lessons from the real world.* New York: Prentice Hall.
* Education Week. (1998). *Technology Counts.* Bethesda, MD: Education Week.
* Hagel, J., & Armstrong, A. (1997). *Net Gain.* Cambridge, MA: Harvard Business School Press.
* Kaput, J. (1992). Technology and mathematics education. In D. Grouws (Ed.), *Handbook of research on mathematics teaching and learning.* New York: Macmillan, 515-556.
* Kollock, P., & Smith, M. (1996). Managing the virtual commons: Cooperation and conflict in computer communities. In S. Herring (Ed.), *Computer-mediated communication.* Amsterdam: John Benjamins.
* McKnight, L., & Bailey, J. (Eds.). (1997). *Internet economics*. Cambridge, MA: MIT Press.
* Orfali, R., Harkey, D., & Edwards, J. (1996). *The essential distributed objects survival guide.* New York: John Wiley and Sons.

* OTA (Office of Technology Assessment). (1988). *Power on! New tools for teaching and learning*. Washington DC: U. S. Government Printing Office.
* PCAST (President's Committee of Advisors on Science and Technology). (1997, March). *Report to the President on the use of technology to strengthen K-12 education in the United States*. Washington, DC: White House Office of Science and Technology Policy.
* Roschelle, J. & Kaput, J. (1996). Educational software architecture and systemic impact: The promise of component software. Journal of Educational Computing Research, 14(3), 217-228.
* Roschelle, J. Kaput, J., Stroup, W., & Kahn, T. (1998). Scalable integration of educational software: Exploring the promise of component architectures. Journal of Interactive Media in Education. http://www-jime.open.ac.uk/98/6/
* SPA (Software Publishers Association). (1997). *Report on the effectiveness of technology in schools, 90-97.* Washington, DC: Author.
* Spohrer, J. (1997). *Something for everyone*. Educational Object Economy Web site (http://www.eoe.org).
* Wenglinksy, H. (1998). *Does it compute? The relationship between educational technology and student achievement in mathematics.* Princeton, NJ: Educational Testing Service.