A New Business Model for 21st Century Educational Software

By Marc Prensky

[3450 words]

"Education is a public service – not a place to make a buck."
—Marc Prensky

Time was, you could make a pretty good living off of educating our kids—just ask the textbook publishers. Heck, with a market of 50 million kids, and some pretty large states doing standard adoptions and the others falling basically in line, your retirement was pretty much guaranteed. Did it matter for the kids? Not much—the curriculum was the curriculum, teaching was pretty much reading, telling, resuming and exercises. The textbooks themselves only had minor differences, if any. If you kept a text in use for 10 years in most subjects, that would be fine—nothing changed.

Then along came software. At first it didn't much matter, as the schools had little (or generally no) infrastructure in place to use it. So the first educational software (often called "Edutainment") got sold to parents in the from of putative entertainment. Since the parents didn't know any better and the kids were getting something that appeared new, edutainment did OK for a while—until kids started playing real computer and video games, and figured out that the edutainment discs were basically the same crap they got in school, but with pretty graphics. After that, the market pretty much went south in a hurry. Mattel bought The Learning Company for 2.5 billion, and finally unloaded it a couple of years later for under 400 million.

Someone got rich off the kids. But Jill Barad rightly got fired, as she should have been. The lesson? Charge for entertainment, *not* education

Here come the Games

In the meantime, a new home software entertainment market for kids was taking off—although at first mostly under the radar—video and computer games. The kids, who were being served the best cutting-edge software being made, ate it up—and they began to learn, tremendous amounts in the process. Learning that was unknown to, or dismissed as irrelevant or worse by their parents and teachers, and undiscussed until recently (see my new book "Don't Bother Me Mom -- I'm Learning" : How Computer and Video Games Are Preparing Your Kids For 21st Century Success -- and How You Can Help!)

The games did however, set the kids' expectations for learning, just as Sesame Street had, decades before.

Today

Now, finally, we are at a stage where our homes and our schools, have a great deal of technology infrastructure. So it finally makes sense to ask:

"What are the best models, financial and otherwise, for taking the learning possibilities that software represents—possibilities we have not even begun to explore and tap—to kids in the 21st century, and particularly right away?"

Stay the Course?

Of course, one could just assume that the models already in place will continue to work. One could just wait for the textbook companies to, gradually, shift their offerings from books to bits. Same model, same sales process, same profits.

Except that these companies have *no idea* how to build the software that works best (none of us do, at this point.) But worse, they have a poor (and I would maintain unworkable) business model for something that is in a rapid state of change and improvement, i.e. software.

Send In The Cavalry?

Enter the Venture Capitalists (tata!) trumpeting the mantra: "The seedbeds of innovation have always been small companies and startups." "Let's just fund some entrepreneurs with some interesting ideas," they say, "and see how they do. And if one or two do well, we even have our exit strategy ready-

made, because the textbook companies will buy them!" Can't miss! Wadda deal! *Charge!!* (Pun intended.)

But there's a huge problem with this strategy as well. Venture-funding small companies, which is an excellent strategy for some types of innovation, only works well in a real, competitive, marketplace—one where customers can easily switch and replace one product with another as innovation occurs. Music players and cell phones are fine for venture-funded innovation. But education is not—it's not the same kind of marketplace.

Why Not?

The economics of education dictate that any purchase made needs to be used (and therefore amortized) over a long period of time—often up to 10 years in the case of textbooks. So if a school (or district or state) were to buy a software package, and a better one came out the next year (which would likely be the case), they would have no way to pay for the switch. *"Sorry, kids,"* they'll have to say.

And that's not even the worst part of having VC-funded vendors vying for our limited education dollars. In a fast-innovating market like educational software, we are likely to get (and already have) lots of different approaches and useful "pieces." In the ideal educational world (i.e. the one we want to create), *everyone* should have access to *all* of these approaches and pieces of software. But in a VC-funded world, what counts is "competitive advantage," which typically comes in the form of "intellectual property," i.e. the innovations companies "own." What that means is that if X tries to put Y's innovation and Z's innovation into their software, they'll be sued.

The consequence of both these things is that, in the VC-funded system, *no educational software on the market is ever fully state-of-the-art*, and the kids lose, bigtime.

It would be great if, recognizing the difference between competitive markets and education, we had a policy of "educational eminent domain," that allowed anyone to appropriate and reuse, for the common good of all our students, any innovation in educational software that any proprietary company should make. But that's a bit further out.

A New Model

For now, let's contrast the VC-funded model with another model for creating and distributing educational software, one that has been developing for a

number of years in the software world under a number of names and rubrics, including *collaborative development, open architecture, open source, and creative commons*.

The first axiom of this new approach is this: In this day and age, many very bright, capable, people are willing to innovate and create useful software without a profit motive (assuming they can earn a decent living, or even not.)

The second axiom is that the education of our kids is a public good, one that should be supported by the public either through taxes or philanthropy, and that no one should make big bucks off of it. (A corollary is that those who try to do so should be barred from the business, starting with Michael Milkin!—see later.)

We certainly don't have enough money in education to be throwing large sums of it after ever-improving software. Especially since we don't have to. Education can, and should, get the best software in the world, for free!

Here's the concept. Thanks to Tim Berners-Lee, who developed the World Wide Web (for science, not money), more and more of us now have the Web available to us both, at home and at school. Very soon we all will have it, one-to-one (see www.projectinkwell.com).

So all we have to do is put all the educational software in the world up there, with free access by anyone who want to use it to learn. In the process, we'll combine all those good, formerly proprietary, features that I talked about into one or a few programs for each subject and level.

But Who Pays The Piper?

Who creates this "free" software, who pays for it, who maintains it, and who upgrades the software and keeps it state of the art? Important questions.

The "who creates it" is easy. The answer is "anyone who knows how and wants to help kids learn." The world is full of people, from myself to Will Wright (creator of *Sim City, The Sims,* and *Spore*), who are interested in education for its own sake and have great ideas they want to get implemented. We will do our part.

But my guess is the most important creators of this stuff, if we give them the chance, will be young people—those who are from the digital generations and understand the power of this technology for learning. They are the ones who can most easily say "This would work a lot better if we did it this way." We

are already starting to see instances of kids making learning games for other kids. The Hidden Agenda contest, sponsored of the last two years by the Liemandt Foundation, has been awarding prizes of $25,000 to whichever competing team of college kids makes the best educational game for middle schoolers, and has had great results. (See www.hiddenagenda.org.) To get many of these people, companies, and contests going, all we need is funding, via grants, foundations etc., and some sort of structure, which we are in the process of creating.

But many developers won't even need funding, if the conditions are right. If the initial funding is used to build not one-off projects but rather templates, than teachers from all over the world can enter additional content easily, and will, as it has been amply demonstrated, do it for free. (Templates are software in which the learning structure is provided, but the content is changeable according to the subject. Templates can be (and have been) built for questions, ideas, tasks, skills, behaviors—any educational process you can think of. Savvy teachers have been downloading, populating and using templates for years.)

Open Architecture and Open Source

This way of creating software—where someone builds and maintains a structure, but others enter content (and change parameters when appropriate) is known as "open" architecture. We developers set it up so that others—especially teachers—can contribute easily, without any programming knowledge. An open architecture product is not always the easiest to create, versus building a product that is complete and self-contained, but it is by far the most efficient over time, as content can be changed and updated continually. (Content, in this context, can be everything from questions, to videos, to problems, tasks, to cases.) Games2train's *Monkey Wrench Conspiracy*, for example, allowed anyone in the world to make up challenging examples for teaching 3D design, and enter them into the learning structure. Scores of such open architecture templates already exist.

"Open source" takes this one step farther, with people around the world able to change the entire software structures (under certain well-defined conditions), with any improvements made accruing to the original. The entire Linux operating system evolved in this way, with "control" being left loose in general, and tightened only when necessary to keep the software robust.

So "creation" of educational software is no problem, and requires no VC funding or profit motive. Lot's of people will do it for pleasure and minimum

compensation. And what's more, these are the "right" people. They are the people who care passionately about education rather than about making money off of their ideas.

Distribution

Now let me talk about distribution, maintenance and improvement over time.

It's important to remember that the distribution we are looking for in educational software is to *students*. To the extent teachers are a way to get to students, all well and good, but today kids do a lot of learning outside (i.e. after) school, and there is no reason to think that good educational software needs to be sold either to schools or teachers. In fact, rather than designing software that neatly fits lesson plans, class periods, etc., we would be a lot better off designing software that works and covers the curriculum, in whatever way is best. That way kids can use the software to learn on their own (assuming it is motivating enough.)

If it is, and if it really works, kids will find it quickly via word of mouth, the way they find everything. If kids really can, as the motto of *The Algebots* says, "beat the game and pass the course," *and* the game is fun and works, they will flock to it in droves.

Their teachers will no doubt also begin hearing about it as well, and may find ways to either integrate it or assign it as homework. If so, all well and good. But much better it should come to the teachers via demand from the kids than vice versa.

And, from the software creators' point of view, what this does is eliminate the entire vetting and buying process that makes it so difficult to sell anything to schools (along with – *boo hoo* – the profits that justify salesmen spending careers doing this.)

From the schools' point of view it's even better—the stuff is free! Will *the schools* find it? Just today I received the following email, representative of dozens I have gotten in the past year: "Hi Marc. How can I get to try out Algebots or buy it? Harry Foxall, Dundee, Scotland" This from only a trailer buried deep in my website. Imagine if I went to the trouble of making it search-friendly!

Maintenance

But, argue the for-profit companies, if the software is free, who will maintain it? After all, someone needs to be sure it works every day, that it's bug free, that it's updated when necessary, and that it's kept free of viruses and other problems. And since it's going to be on the Internet, who, by the way, will host it?

I have the perfect candidate for this—our education schools. Everybody admits they're broken and in desperate need of new directions. How about if one of those directions is to take charge of our curricular software in a particular area and level? The students in these schools are, increasingly, the digital natives, the very ones who are most likely be using the software in their teaching. How much better if they know it inside out having worked with it for the time they are studying? Maintaining, using and improving the software could be made part of required courseware for prospective teachers.

And if we really want to make *sure* there is always at least one individual focusing on a particular piece of software, and it doesn't get lost in the crush of other things, why not endow fellowships at the education schools, awarded in exchange for students agreeing to take care of particular software programs or categories? The recipients of those fellowships each semester would have the ultimate maintenance responsibility for the software, and the continuation of the fellowships through their endowments would mean that this would continue every semester in perpetuity.

Continuous Improvement

It would also be the host school's responsibility to keep improving the software, which they could do in any number of ways. Grants (from the government of foundations) would allow schools to commission whole new parts and templates, in partnerships with local, or worldwide, developers. Connections with other departments in the school, such as IT, engineering and game development would allow them to get much of the work done inexpensively. Contests among the schools participating could keep innovation at its peak. Required sharing of new developments and code between schools would assure that innovations in one school get spread to all. Code and content contributions from around the would can be solicited, vetted and, if worthy, integrated. Systems of quality control can be developed both internally and using external vendors.

Engagement

All this hinges, of course, on the software that is developed being engaging and motivating to learners, whether through game-based or other

approaches. But who better to ensure this than people still in school, who are closest in age and digital experience to the students? The Hidden Agenda contest I described previously has shown that learning software development by college students is feasible. It is far more likely that engaging software will be developed by young, interested people than by any designers hired by for-profit companies.

And these student developers needn't lack for professional advice, either. Successful, rich, game developers around the world—now with kids—are eager to jump in and help—they just need a structure in which to do so. We already have a great model in *Ben's Game*, a game developed by a game software developer working pro-bono with a kid with cancer.

Instructional Design?

Yet another objection I often hear is that if kids and gamers design the curricular learning software, where will the "instructional design" come from? My sense is that that this will be the greatest boon of all to learning – the kids getting freed from formal, stilted "instructional design.". Despite what you may think, we don't need formal "instructional designers" to design effective learning, although we do need good learning designs, which can be created by any bright person. Sadly, traditional instructional design has led us to a dead end in educational software. It has taught us almost nothing that is not intuitively obvious from thousands of years of learning, while at the same time, in the words of one game designer, "sucking the fun out" of everything it has touched. It is time for a new approach.

We all know learning involves activities like comparing, deciding, estimating, imitating, listening, observing, practicing, predicting, questioning, reflecting, trying, and verifying. It doesn't take an old-school "instructional designer" to figure out how to include these, or even measure them. This is why the games industry has been able to develop the very best teaching tools of all time purely by intuition and iteration. "Game designers, says MIT professor Seymour Papert, have a better take on the nature of learning than curriculum designers." Adds James Paul Gee, Professor of Education at the university of Wisconsin, Madison, "Better theories of learning are embedded in the video games many children in elementary school and particularly in high school play than in the schools they attend."

Bottom Line

So the bottom line, interestingly enough, is that there need be *no* bottom line for educational software. Let's leave the VC funding and profits out of it—let

them make their fortune elsewhere, in true competitive markets, and plow as much of it as possible back into educational software in the ways I am suggesting.

In my view, any group interested in funding the educational software of the future should put creative developers together with kids, the curriculum, and forward thinking education schools, and inject some funds to get the ball rolling. Once it is, we will see kids, teachers, and the whole world jump on board, and the ball will acquire momentum of its own.

Examples

I am often asked if there are existing examples of what I am talking about. While the model I describe for educational software has yet to be fully put into operation, there are many precedents and pieces that are already in place and functioning quite well. Useful examples to consider include:

- The development of the World Wide Web as a free "innovation commons" (see the work of Lawrence Lessig, Stanford Law Professor)

- The development and maintenance of Linux by the open source community (with the occasional assistance of IBM an others) to the point where it has become a standard rivaling Microsoft, now used by whole corporations and countries.

- The development of the free, open Wikipedia as a world wide open source project to build a communal encyclopedia.

- The development of the free, open source Moodle as an alternative to expensive, proprietary Learning Management Systems such as Blackboard. Their sustainability model is that consultants they certify to customize the system for clients pay them a 10 percent rebate that goes to future development. All enhancements made anywhere are available to the benefit of all users

- The development of hundreds of free sites for sharing, from Myspace to Limewire, and their huge usage by kids.

- The large number of free educational mini-games already on the Web; mostly the work of interested educators around the world who want to contribute.

- The failure of the for-profit educational model to engage kids. Edutainment, and now Leapfrog are probably the best examples of this. Leapfrog makes products ostensibly for kids, but they are actually for the parents, to alleviate their anxiety about their kids' falling behind. As a savvy marketer put it at a White House conference, "You can't sell a product to both kids *and* to parents—you have to choose one." I'll take the kids.

- The success of the games world in engaging kids, along with the refusal of that world to get into anything educational as part of their business model. They know the stuff won't sell. But they also know it will work. Again: *Charge for entertainment, but not for education.*

But Do People Value Free?

I have heard it argued repeatedly that "people don't value things if they're free." That idea may apply to the past, but it surely doesn't apply to the digital world. Who (at least what young person) doesn't value Google search, Google maps, the Wikipedia, free mp3 music, free downloaded movies? Today it's *charging* for stuff that causes suspicion.

A Way To Test

Curiously, and fortunately, we currently have a perfect way to test whether either of these funding models is better than the other, or whether they can coexist. In addition to Games2train's developing *The Algebots,* an Algebra I curricular game, under what I am calling the "new" model, a VC-funded company, Tabula Digita, is developing an Algebra I curricular game, *DimenXion*, to be sold to schools and/or after school programs. With our development plans roughly on the same schedule, this makes a perfect test case.

So let the contest begin!

Marc Prensky is an internationally acclaimed thought leader, speaker, writer, consultant, and game designer in the critical areas of education and learning. He is the author of Digital Game-Based Learning (McGraw Hill, 2001) and the upcoming Don't Bother Me, Mom, I'm Learning (Paragon, 2006). Marc is the founder and CEO of Games2train, a game-based learning company, whose clients include IBM, Bank of America, Pfizer , the U.S. Department of Defense and the LA and Florida Virtual Schools. He is also the creator of the sites www.SocialImpactGames.com, and www.GamesParentsTeachers.com . Marc holds an MBA from Harvard and a Masters in Teaching from Yale. More of his writings can be found at www.marcprensky.com/writing/default.asp . Marc can be contacted at marc@games2train.com .