

A Taxonomy of Simulation Software

A work in progress

Kurt Schmucker
Apple Computer, Inc.

“Science-related software that lets kids make and explore conjectures can dramatically expand children’s math and science thinking.”

Judah Schwartz, Co-Director
Harvard Educational Technology Center

“Computers . . . will be able to simulate the world as well as explain it. Creating or using a computer model can be a great educational tool. . . . Simulation games will get better, but even now the best of them are fascinating and highly educational.”

Bill Gates

“Simulation technologies offer exciting opportunities for students to explore information, pursue their interests, experiment, and demonstrate what they have learned.”

George Lucas

“Genuine computer literacy is not about learning to use tools like a word processor or spreadsheet, but about learning a new language of events, processes, and dynamic relationships that will help make the world and its ideas more understandable, more communicable, and more civilized.”

Alan Kay, Disney Fellow
Walt Disney Imagineering

Simulations provide a unique environment for exploring new concepts, for gaining an understanding of the interplay between related complex phenomena, and for the construction of simplified working models of topics under study. Simulation is also one area in which computing technology is uniquely suited to be the delivery mechanism for an educational experience. The advocates of simulation use herald it as a great advance for education.

The use of simulation in education, particularly in K–12 education, is not a foolproof educational technique, however, nor is it without its critics:

“Experiences with simulations do not open up questions, but close them down.” (Sherry Turkle, MIT sociology professor)

Problems can occur when simulation software is used in education. Sometimes, however inadvertently, an improper type of simulation can be used in an attempt to achieve a given educational objective. Other times, the inevitable differences and simplifications between reality and the simulation of that reality are not properly understood by the student. The end result can be frightening, as in one tenth grader’s statement of what was learned by using SimCity in a civics class: “Raising taxes always leads to riots.”

In the hopes of avoiding some of these problems, this article presents a taxonomy of current simulation software, categorizing some of the educational strengths and weaknesses of each type of simulation in a K–12 environment. Many representative examples of Mac OS and Windows simulation packages that typify the various simulation categories are also surveyed.

The focus of this work is simulations that could reasonably be used in a K–12 classroom or on a home computer. Multimillion-dollar immersive flight simulators or complex simulation codes that run only on supercomputers, while arguably extremely useful and cost effective in certain environments, are generally beyond the scope of this work. Likewise, at the other end of the spectrum, board games such as Monopoly could be considered low-fidelity simulations of small economic systems, but for the purposes of this article, we will consider only computer-based simulations whose complexity is reasonably beyond what could be done by hand.

Learning Technology Review

A Taxonomy of Simulation Software

This is also a work in progress, and reflects the author's observations to date. Comments, criticisms, extensions, or simplifications of the taxonomy are most welcome.

What's a simulation and what's not

The logical first step in building a taxonomy is the definition of what things will be categorized in the taxonomy and what things fall outside of the scope the taxonomy is attempting to cover. Unfortunately, none of the definitions of the term "simulation" that I found seemed to be sufficiently broad or descriptive. These two definitions were representative:

"Games that re-create or simulate in detail a real-life, first-person activity."
(Mindscape web site)

"Simulation software: The adage that 'experience is the best teacher' is true. But it is not easy to provide students with many of the experiences they need to truly understand the material they are studying. Simulation software offers a way to allow students to work on tasks or projects that would otherwise be impractical, dangerous, or prohibitively expensive." (Glossary from *Learn & Live*, George Lucas Educational Foundation)

The first of these is too limiting, since it appears to cover only certain types of simulation games (such as flight simulators) but ignores many other types of simulations. The second is broad enough, but doesn't give sufficient detail to help decide whether a given program is a simulation or not. Is Doom a simulation? What about Barbie Fashion Designer, SimCity, Myst, or Logo?

Let's examine three different software packages that are clearly simulations and attempt to design a definition that better suits the needs of this taxonomy work.

GenScope (Figure 1) is an excellent example of an educational simulation designed for middle and high school students in the area of genetics. [Horwitz] GenScope provides the student with an interactive environment in which the relationships between chromosomes, genes, and observable traits can be both explored and tinkered with. The student is offered several different views of the same information (from pop-up menu chromosomes on idealized "popsicle stick" genes—a great view for the beginning student—to the view of genetic information that is really used in science: family trees of populations with observable traits for each individual labeled). By gently moving from the idealized view to the real view, the student can gradually grow from a simple understanding of basic concepts to the application of those concepts in the messy and complex real world. GenScope is well grounded in the principles of genetics and inheritance, and so as to not

Learning Technology Review

A Taxonomy of Simulation Software

violate this grounding, certain actions are not allowed. While the GenScope user can manipulate the genes of an isolated individual, changing it from a wingless dragon to a winged one, for example, the user cannot change the genes of a dragon in a family tree, since this could make the family tree an invalid example, violating the principles of inheritance.

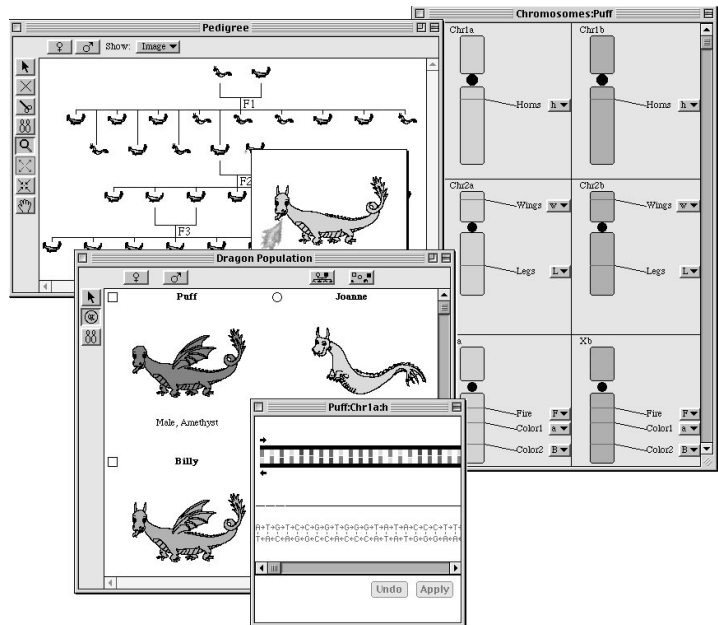


Figure 1. GenScope. GenScope presents several different interfaces to genetic concepts. The Chromosome view (shown above with the yellow and purple "popsicle sticks") gives the user an idealized view of the relationship between genotype (the genetic configuration) and phenotype (the observable characteristics in the individual). The DNA view gives a biochemical view, and the several Population views show the individuals in a population and their observable characteristics. The student can be gently led from the idealized visualization of genetic concepts (popsicle sticks) to a view that closely mimics the way real genetic scientists work (family trees of populations and records of the observable traits of the individuals in those populations).

Although Figure 1 shows GenScope in use in the study of dragon genetics, this, too, is just a bridge from a simplistic early example (in which few students have real-world experience that might interfere with grasping new concepts) to more detailed, more realistic examples. Other GenScope simulations exist (admittedly still simplified) for the exploration of canine and human genetics. Like real life, the inherited characteristics of an individual offspring in a GenScope simulation are not predictable. Will the new baby dragon have wings, have horns, be fire-breathing? Will the new human baby have blue eyes, hemophilia, male pattern baldness? Also like real life, some of the inherited characteristics do not follow a simple dominant-recessive gene model.

Sometimes a characteristic is gender-linked. Sometimes a seemingly simple physical characteristic has a complex genetic model involving multiple genes and various other dependencies. And sometimes a certain combination of genes is fatal to the new offspring. With a well-designed curriculum and by turning on or off various GenScope features, these complications of real genetics can be gradually introduced to the student—something that cannot be done so easily in real life.

The user of GenScope (either the student or the teacher) cannot extend any of these GenScope models to include new traits, such as adding eye color to dragons, for example. Nor can the GenScope user implement a GenScope model for a new species (a frog, for example). These types of tasks must be left to the GenScope implementers.

ActivChemistry (Figure 2) is another example of an educational simulation that is a construction kit in the area of chemistry. It provides the student with a fixed set of parts (Bunsen burners, chemicals of just about any composition, and a wide variety of meters and gauges, among many others). The student can combine these pieces in various ways to perform experiments; gather and graph data; learn about new concepts in interactive and dynamic lessons; or take interactive, dynamic exams that test not just the retention of facts, but also the understanding of processes.

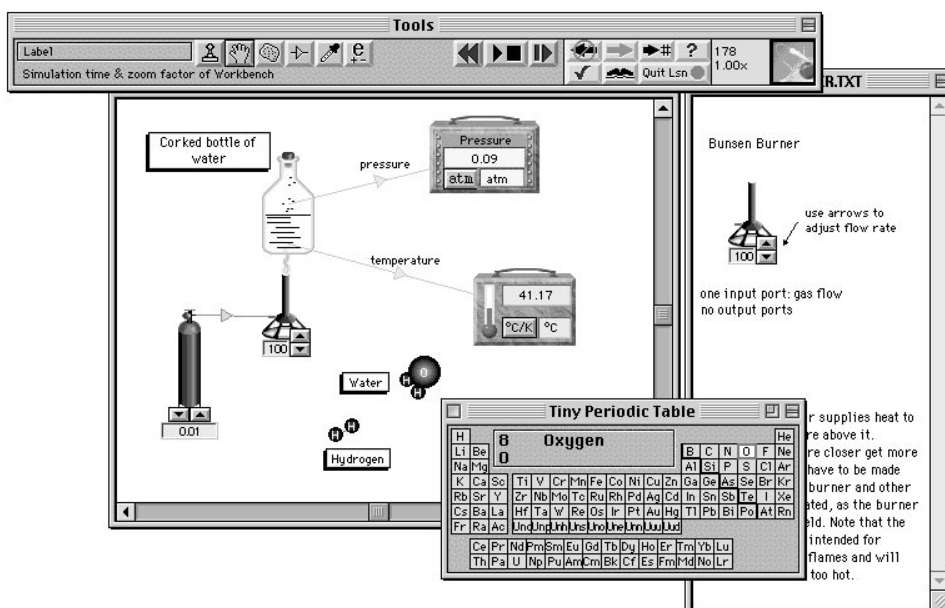


Figure 2. ActivChemistry. A virtual chemistry set construction kit that is well grounded in chemistry theory and provides a cost-effective way to increase chemistry lab time for both high school and college students.

Learning Technology Review

A Taxonomy of Simulation Software

The set of parts in ActivChemistry is sufficient for most of the topics in a high school or freshman college chemistry class. Because of its construction kit nature, the student using ActivChemistry has a great deal of freedom in the types of chemical reactions that are explored. Experiments that would be too dangerous or too expensive for most high school chemistry labs can be done in ActivChemistry. Also, the student using ActivChemistry is not under the time pressures often found in the standard chemistry lab period—pressures that often manifest themselves in having to replicate small, cookbook-style experiments rather than pursue work in a more open-ended fashion.

While ActivChemistry students cannot add new parts, they can combine those parts in many different ways. Students can interact with the simulation as they dispense different chemicals, measure different reactions, “wire up” different meters, and cause unique outcomes every time they conduct an experiment. The unpredictability (to the student) of ActivChemistry includes these measurements (pH, temperature, reactant concentration, pressure and volume of a gas, and so on) as well as the reaction products.

Not all simulations present detailed scientific concepts in as direct a manner as GenScope and ActivChemistry. Some simulations present a simulated environment or phenomena for the user to explore and experience, rather than a set of tasks designed to teach the user a new set of concepts. Virtual reality visualization applications such as Virtus WalkThrough (Figure 3) are the simplest examples of simulated phenomena applications. In Virtus WalkThrough, the user can “walk” and “fly” around a three-dimensional representation of a structure, thereby obtaining a better appreciation of how the structure looks than can be gained by merely viewing two-dimensional blueprints or elevation diagrams. However, the environment is unchanging and the user choices are limited. First-person adventure games, such as Doom, Quake, or Marathon, extend this to a fully simulated phenomena by adding other characters for the user to “interact with,” as well as a rich set of user choices. Flight simulators continue further by adding a detailed computational model of flying and using this model in a tight feedback loop driven by user input.

Learning Technology Review

A Taxonomy of Simulation Software



Figure 3. Virtus WalkThrough. Virtual reality visualization applications such as Virtus WalkThrough present an environment for the user to experience. In Virtus Walkthrough, the user is able to move around a three-dimensional structure, such as Lizzy Borden's house, shown above. For most people, these virtual reality visualization applications present a much easier-to-understand view than looking at two-dimensional blueprints or elevation diagrams.

Inspired by these examples, we can synthesize a better definition of a simulation: A simulation is a software package (sometimes bundled with special hardware input devices) that re-creates or simulates, albeit in a simplified manner, a complex phenomena, environment, or experience, providing the user with the opportunity for some new level of understanding. It is interactive and usually grounded in some objective reality. A simulation is based on some underlying computational model of the phenomena, environment, or experience that it is simulating. In fact, some authors use "model" and "modeling" as synonyms for "simulation."

Also, a simulation usually has unintended consequences outside the intended user's level of expertise in its domain. PacMan is an excellent game, but it is not a simulation. You know the possible outcomes and consequences ahead of time. The game is in seeing if your reflexes and real-time planning are up to the challenge. A simulation, on the other hand, has unpredictability.

Unpredictability, of course, is relative. What is unpredictable to a child might be completely predictable to an adult. Similarly, SimCity might be an engaging, thought-provoking environment to some, but to an experienced urban planner, it might be only a simplistic game. In deciding what is a simulation

Learning Technology Review

A Taxonomy of Simulation Software

and what is not, we have to take into account the intended user. “Simulation” is sometimes confused with “visualization” and “animation,” even by authors of simulations.

So by way of contrast, a data visualization application is a software package that portrays a fixed data set in graphically useful ways. A simulation is based on a computational model whose parameters can be modified (by the user, or by a random process that is built into the computational model) to generate many data sets. In the case of a data visualization application, the goal is to gain an understanding of the underlying data set; in a simulation, the goal is to gain an understanding of the model. Virtus WalkThrough (Figure 3) is an example of a visualization.

Also by way of contrast, an animation or a multimedia presentation, like a movie, is software that presents a graphical depiction that is the same every time it is viewed. A simulation generates a different depiction every time, since the parameters of the underlying model are (usually) different each time the simulation is run.

Simulation can also be viewed as one component of computational science.

“Computational science is an exciting combination of modeling and graphics with science and mathematics. In this discipline, computers are used to ‘do’ science through simulation and visualization. Computational science has therefore become a third way of doing science, along with experimentation and theoretical analysis. . . . Computational science is an effective strategy for middle and high school math and science instruction because it:

- requires real-world strategies and problem-solving contexts
- encourages interdisciplinary and integrated instruction
- uses technology for student exploration and inquiry
- moves students to higher conceptual levels and deeper understandings of concepts addressed
- can be done with desktop computers and modeling/visualization software (accessible to schools).” [Envision It!]

Learning Technology Review

A Taxonomy of Simulation Software

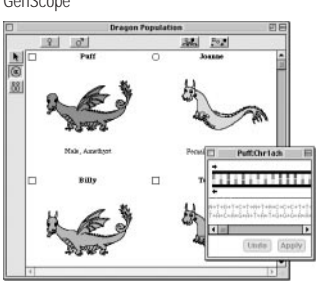
So, for our purposes here, the defining characteristics of a simulation are:


| Characteristic | Remarks |
|---|---|
| Creates (or re-creates) a phenomena, environment, or experience | Can be based either in fantasy or reality. While many fantasy simulations are games, some educational simulations are purposely set in a fantasy environment so that the student won't confuse the simulation with reality. |
| Provides an opportunity for understanding | The user should be able to learn something new. |
| Interactivity | Interactive "steering" of the simulation; in other words, the user's inputs must have some effect on the course of the simulation. |
| Grounding | A consistent model of a theory. |
| Unpredictability | Randomness, or an extreme sensitivity to user inputs. |


Learning Technology Review

A Taxonomy of Simulation Software

The charts on the following pages show how the characteristics of some well-known software packages qualify them as examples or non-examples of simulations, according to the definitions on page 47.


| | | |
|---|--|--|
|  <p>A simulation</p> | Phenomena, Environment, or Experience | Genetics of individuals and populations |
| | Opportunity for Understanding | Mendelian principles, inheritance, genes, chromosomes, genotypes, phenotypes, and so on |
| | Interactivity | User can alter genes, choose mating pairs, determine the level observations (DNA, cells, individuals, whole populations) |
| | Grounding | Firmly grounded in genetics |
| | Unpredictability | Inheritance in individuals, meiosis |

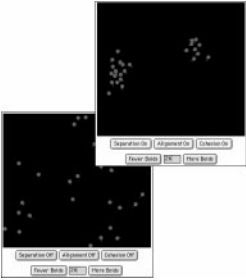
| | | |
|--|--|--|
|  <p>A simulation</p> | Phenomena, Environment, or Experience | Dynamics of a city (traffic, zoning, development, creation of a stable dynamic system [harder than you might think]) |
| | Opportunity for Understanding | City planning, dynamic systems |
| | Interactivity | Design is interactive; running the sim is not too interactive |
| | Grounding | Well grounded in urban planning, but biased |
| | Unpredictability | Behavior of the citizens of SimCity is not easy to predict; natural disasters (earthquakes, floods, and so on) |

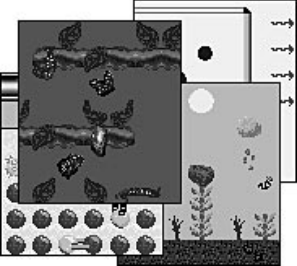
| | | |
|---|--|---|
|  <p>Not a simulation</p> | Phenomena, Environment, or Experience | None really; a series of interconnected puzzles with a wonderfully rendered interface |
| | Opportunity for Understanding | Little |
| | Interactivity | High |
| | Grounding | None |
| | Unpredictability | None |

Learning Technology Review

A Taxonomy of Simulation Software

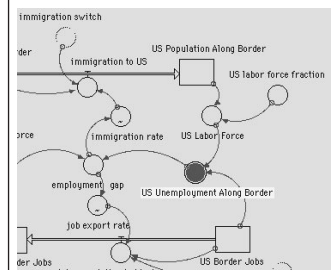
| | | |
|--|--|--|
| <p>Swamp World</p>  <p>Not a simulation</p> | Phenomena, Environment, or Experience | Walking around a swamp |
| | Opportunity for Understanding | Little |
| | Interactivity | Unique interaction via a novel input device: a push toy shaped like the pictured chicken |
| | Grounding | Grounded in walking |
| | Unpredictability | None |


| | | |
|---|--|---|
| <p>Boids</p>  <p>A simulation</p> | Phenomena, Environment, or Experience | Emergent behavior (complex behavior emerging from simple rules) |
| | Opportunity for Understanding | Animal behavior, emergent behavior |
| | Interactivity | Parameter setting only |
| | Grounding | A simple model that explains a complex behavior |
| | Unpredictability | Sensitive to several parameters |

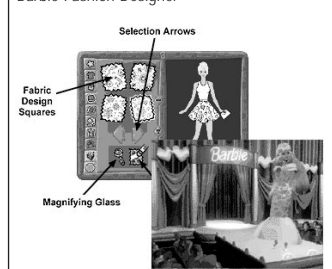
| | | |
|--|--|--|
| <p>Cocoa</p>  <p>A simulation construction tool</p> | Phenomena, Environment, or Experience | Domain-independent; has been used for chemistry, genetics, and ecology simulations and lots of games |
| | Opportunity for Understanding | Building a working simulation of some phenomena or environment |
| | Interactivity | Highly interactive both in design and simulation execution |
| | Grounding | Up to the sim builder; some are well grounded in science, and some are completely fanciful |
| | Unpredictability | Randomness is a basic Cocoa simulation construction technique |

Learning Technology Review

A Taxonomy of Simulation Software

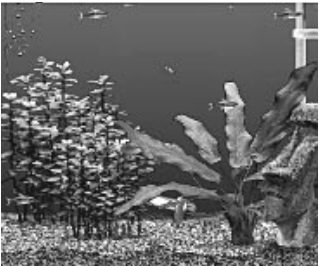
| | | |
|---|---|--|
| <p>Stella</p>  <p>A simulation construction tool</p> | <p>Phenomena, Environment, or Experience</p> | <p>Domain independent; has been used to build simulations in math, chemistry, economics, history, physics, and so on</p> |
| | <p>Opportunity for Understanding</p> | <p>Building a working simulation of some phenomena or environment</p> |
| | <p>Interactivity</p> | <p>Highly interactive in design, and somewhat interactive in simulation execution</p> |
| | <p>Grounding</p> | <p>Up to the sim builder; equations are used to model the changes to the parts of the model</p> |
| | <p>Unpredictability</p> | <p>Interaction between the modeling equations; aggregate measurements on the simulated quantities</p> |


| | | |
|--|---|--|
| <p>Creatures2</p>  <p>A simulation</p> | <p>Phenomena, Environment, or Experience</p> | <p>Breeding, raising, teaching, and protecting a population of intelligent creatures (Norns)</p> |
| | <p>Opportunity for Understanding</p> | <p>Animal behavior, genetics, ecology</p> |
| | <p>Interactivity</p> | <p>Highly interactive</p> |
| | <p>Grounding</p> | <p>Firmly grounded in genetics and artificial life</p> |
| | <p>Unpredictability</p> | <p>Inheritance, random ecological events</p> |

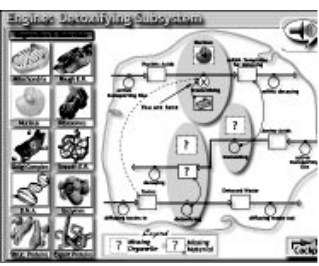
| | | |
|--|---|---|
| <p>Barbie Fashion Designer</p>  <p>A simulation</p> | <p>Phenomena, Environment, or Experience</p> | <p>Design and construction (sewing) of clothes</p> |
| | <p>Opportunity for Understanding</p> | <p>Visual aesthetics</p> |
| | <p>Interactivity</p> | <p>Interactive in design phase, not interactive in 3D runway simulation</p> |
| | <p>Grounding</p> | <p>Fashion design, sewing</p> |
| | <p>Unpredictability</p> | <p>None</p> |

Learning Technology Review

A Taxonomy of Simulation Software


| | | |
|--|--|--|
| <p>Aqua Zone</p>  <p>A simulation</p> | Phenomena, Environment, or Experience | Aquarium design and maintenance |
| | Opportunity for Understanding | Raising fish, maintenance of aquarium ecology |
| | Interactivity | Interactive, but only through dialogs for adding fish food, adjusting tank pH, and so on |
| | Grounding | Grounded in aquarium ecology |
| | Unpredictability | Fish move, eat, breed, and so on randomly |

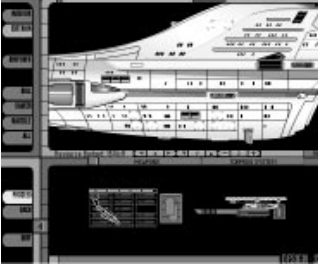
| | | |
|---|--|--------------------------------------|
| <p>Silent Hunter</p>  <p>A simulation</p> | Phenomena, Environment, or Experience | Operating a submarine during wartime |
| | Opportunity for Understanding | Wartime naval strategy |
| | Interactivity | Highly interactive |
| | Grounding | Grounded in submarine operations |
| | Unpredictability | Behavior of enemy; weather |

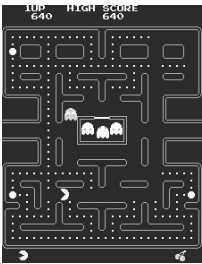
| | | |
|---|--|--|
| <p>Fly a Cell</p>  <p>A simulation</p> | Phenomena, Environment, or Experience | Cell metabolism; internal cell metabolic systems |
| | Opportunity for Understanding | Cell biology and internal systems |
| | Interactivity | Interactive "steering" of the simulation; interactive tutorials on cell metabolism |
| | Grounding | Well grounded in cell biology and biochemistry |
| | Unpredictability | Interaction between the modeling equations; aggregate measurements on the simulated quantities |

Learning Technology Review

A Taxonomy of Simulation Software

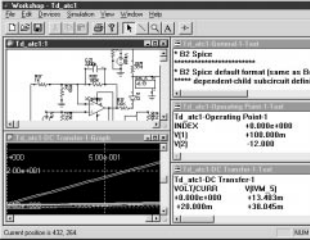
| | | |
|---|--|--|
| <p>Oregon Trail</p>  <p style="text-align: center;">A simulation</p> | Phenomena, Environment, or Experience | Pioneers on the Oregon Trail |
| | Opportunity for Understanding | Deeper and more thorough knowledge of American history |
| | Interactivity | Very interactive |
| | Grounding | Grounded in history and the technology of the day |
| | Unpredictability | Accidents along the trail, illness, fording of rivers, as well as choices of routes, companions, rest stops, and trading of supplies |


| | | |
|---|--|--|
| <p>Starship Creator</p>  <p style="text-align: center;">A simulation construction kit</p> | Phenomena, Environment, or Experience | Design, staffing, and operation of a Star Trek starship |
| | Opportunity for Understanding | Cost/benefit trade-offs |
| | Interactivity | Highly interactive |
| | Grounding | Firmly grounded in the history, science, and personalities of the Star Trek universe |
| | Unpredictability | Many variables in starship design; randomness in mission execution |


| | | |
|---|--|--------------------|
| <p>PacMan</p>  <p style="text-align: center;">Not a simulation</p> | Phenomena, Environment, or Experience | None |
| | Opportunity for Understanding | None |
| | Interactivity | Highly interactive |
| | Grounding | None |
| | Unpredictability | Little |

Learning Technology Review

A Taxonomy of Simulation Software


| | | |
|--|---|--|
| <p>B² Spice</p>  <p>A simulation construction tool</p> | <p>Phenomena, Environment, or Experience</p> | <p>Electronic circuit design</p> |
| | <p>Opportunity for Understanding</p> | <p>Building a working simulation of a circuit</p> |
| | <p>Interactivity</p> | <p>Highly interactive in design and simulation execution</p> |
| | <p>Grounding</p> | <p>Well grounded in physics and electronics</p> |
| | <p>Unpredictability</p> | <p>Student-designed circuits don't always function as intended</p> |

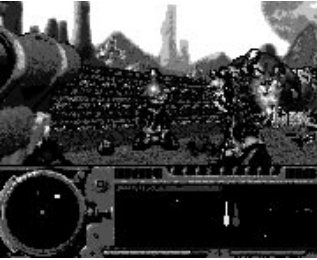
| | | |
|--|---|--|
| <p>Yoot Tower</p>  <p>A simulation</p> | <p>Phenomena, Environment, or Experience</p> | <p>Dynamics of a high-rise hotel, apartment building, or mall</p> |
| | <p>Opportunity for Understanding</p> | <p>Large system behavior, social systems, politics, urban planning</p> |
| | <p>Interactivity</p> | <p>Highly interactive</p> |
| | <p>Grounding</p> | <p>Grounded in urban planning</p> |
| | <p>Unpredictability</p> | <p>Behaviors of residents, natural disasters</p> |


| | | |
|---|---|--|
| <p>Flight Unlimited</p>  <p>A simulation</p> | <p>Phenomena, Environment, or Experience</p> | <p>Flying an aircraft</p> |
| | <p>Opportunity for Understanding</p> | <p>Flight dynamics, flight planning</p> |
| | <p>Interactivity</p> | <p>Highly interactive</p> |
| | <p>Grounding</p> | <p>Well grounded in avionics</p> |
| | <p>Unpredictability</p> | <p>Weather; highly sensitive to user input</p> |

Learning Technology Review

A Taxonomy of Simulation Software

| | | |
|--|---|---|
| <p>Cosmopolitan Virtual Makeover</p>  <p style="text-align: center;">Not a simulation</p> | <p>Phenomena, Environment, or Experience</p> | Experience different hairstyles, make-up, fashion accessories |
| | <p>Opportunity for Understanding</p> | Little |
| | <p>Interactivity</p> | Highly interactive |
| | <p>Grounding</p> | Little |
| | <p>Unpredictability</p> | Very little |

| | | |
|--|---|--|
| <p>Marathon</p>  <p style="text-align: center;">A simulation</p> | <p>Phenomena, Environment, or Experience</p> | Multi-player, first-person, three-dimensional combat |
| | <p>Opportunity for Understanding</p> | Little (some strategy, but mostly fun and gore) |
| | <p>Interactivity</p> | Highly interactive |
| | <p>Grounding</p> | Some grounding in motion and mechanics |
| | <p>Unpredictability</p> | Actions of computer-based characters, number of characters, effectiveness of weapons, actions of human-based players |

| | | |
|---|---|--|
| <p>Myth</p>  <p style="text-align: center;">A simulation</p> | <p>Phenomena, Environment, or Experience</p> | Medieval combat between groups |
| | <p>Opportunity for Understanding</p> | Strategy, placement of forces |
| | <p>Interactivity</p> | Interactive |
| | <p>Grounding</p> | Some grounding in motion and mechanics |
| | <p>Unpredictability</p> | Actions of both armies |

Learning Technology Review

A Taxonomy of Simulation Software

Taxonomy methodology

The taxonomy described in this article was built from the “bottom up” by examining, using, and watching the use of many Mac OS– and Windows-based simulation packages, and attempting to organize and categorize them in a way that might be useful to others. This taxonomy is techno-centric in that it is organized around the features of the various simulation programs (for example, immersive simulations such as a flight simulator versus an observational simulation such as SimCity; a fixed functionality simulation such as The Incredible Machine versus a programmable environment such as Stella or Extend). Other ways of organizing a simulation taxonomy are both possible and useful. Bruce, for example, presents a taxonomy of educational technology applications organized around the pedagogical principles embodied in the application [Bruce], and Taylor presents a different taxonomy with main categories that differentiate the roles of the computer: as a tutor, as a tool, and as a tutee [Taylor]. Still others have used other organizing principles: instructional methodologies [Alessi and Trollip] and evaluation criteria [Persichitte].

While the taxonomy in this article is meant to be a complete categorization of software simulation packages, the examples listed here are meant merely to be a representative, not an exhaustive, listing of Mac OS– and Windows-based simulation packages currently available. If, for example, 10 software packages examined were all examples of the same type of simulation category, only one or two of the packages might be listed in the figures and tables of this article.

In addition, many of the examples shown in the taxonomy could be considered as examples in several different categories. Yet, with one exception, each exemplar has been used only once, with the “main intent” of the application used to determine in which category it should be placed. The category boundaries are also somewhat fuzzy [Schmucker, 1983]. For example, the difference between a complete and full-featured simulation and a single-concept simulation is often just a matter of perspective.

Learning Technology Review

A Taxonomy of Simulation Software

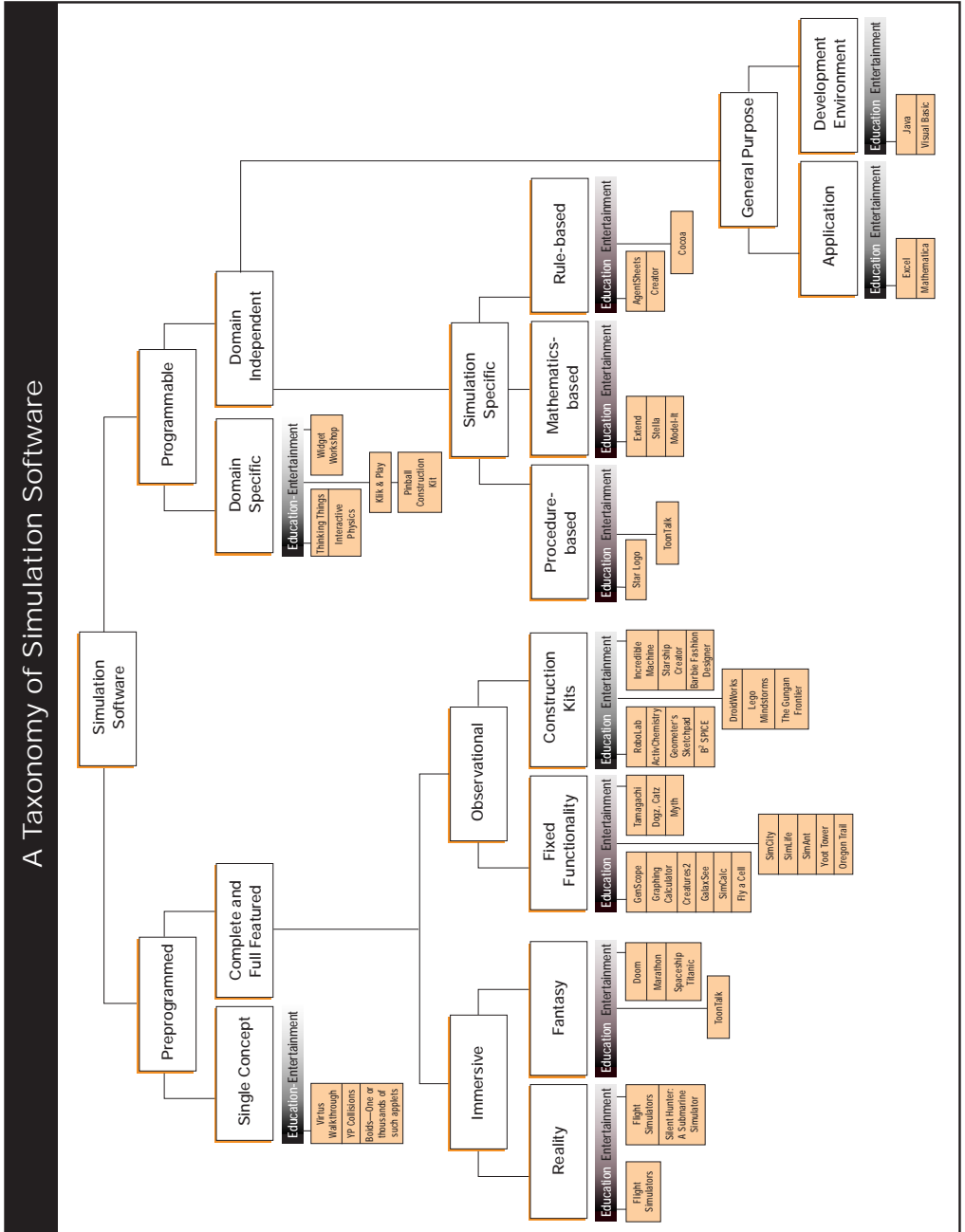


Figure 4. A techno-centric taxonomy of Mac OS- and Windows-based simulation software, with examples of every category.

Learning Technology Review

A Taxonomy of Simulation Software

As a first step in assisting educators with simulations, the chart in Figure 4 presents a taxonomy of current simulation software, categorizing some of the features of each type of simulation that might be used in a K–12 environment. Many representative examples of Mac OS– and Windows-based simulation packages that typify the various simulation categories are also listed. While the taxonomy is meant to be a complete categorization of simulation packages, the examples listed here are meant merely to be a representative, not an exhaustive, listing of all Macintosh and Windows simulation packages currently available.

The Taxonomy

In categorizing this information about simulation software packages, several interesting observations were made. Some distinctions became clearer and underlying trends emerged, while some issues seemed to require a closer look. Here are some of the most important points about the taxonomy:

Education-Entertainment

The education-entertainment spectrum occurred for almost every category in the taxonomy. *ActivChemistry* (Figure 2) is an example of an educational simulation that is a complete and full-featured, observational, construction kit; *The Incredible Machine* (Figure 5) is another example of this same type of simulation, but it is purely for entertainment; and *Star Wars DroidWorks* (Figure 6) is another example of this same type of simulation, but one that is delicately balanced to lie in the middle of the education-entertainment spectrum.

Learning Technology Review

A Taxonomy of Simulation Software

58

The taxonomy takes this spectrum into account by having a continuous education-entertainment dimension as a property of every final node in the taxonomy tree.

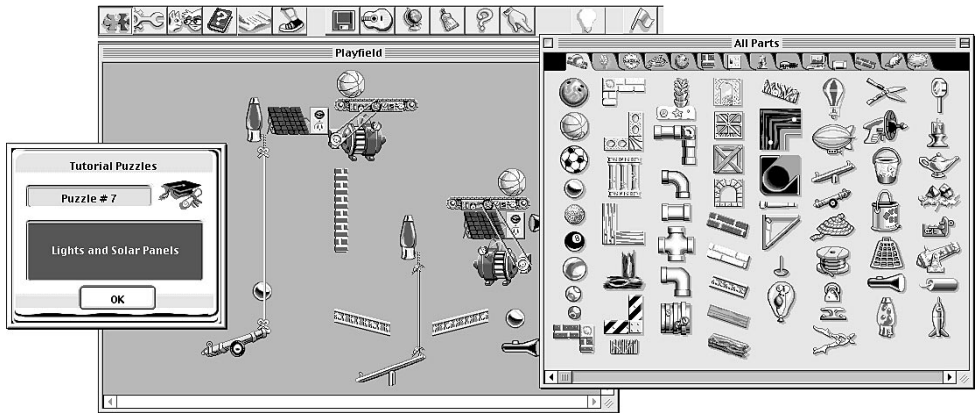


Figure 5. The Incredible Machine. A simulation construction kit that while not always firmly grounded in correct physics principles, is nevertheless a compelling problem-solving environment that is extremely entertaining.

Learning Technology Review

A Taxonomy of Simulation Software

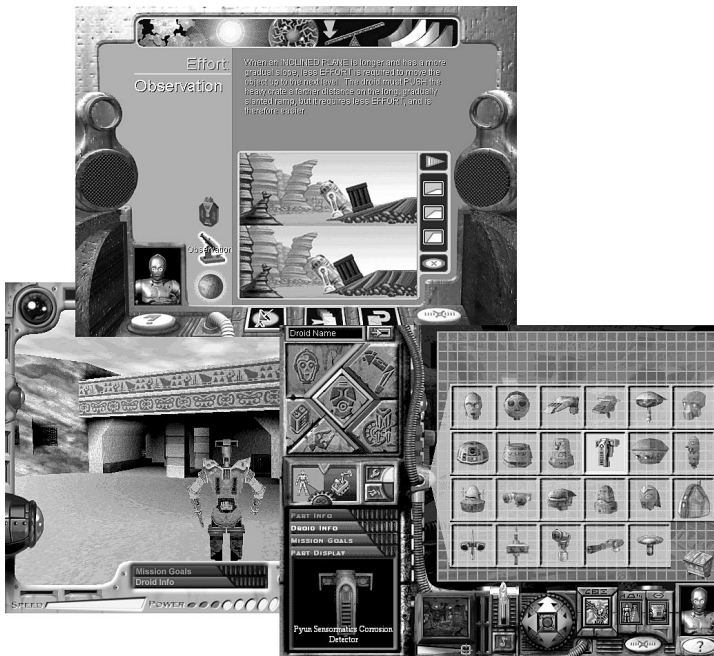


Figure 6. Star Wars DroidWorks. A simulation construction kit that successfully achieves the difficult balance between entertainment and education. Students build droids to accomplish various missions, but to do so they must apply a variety of principles in math, physics, and biology. The application is self-contained with an integrated "Information & Data Expert" that explains the necessary principles and often provides interactive animations to assist the student in understanding a new concept. A fine example of "just in time" learning as well as situated learning in a project environment. And yet it still remains entertaining.

Programmable or preprogrammed

The first division in the taxonomy splits actual simulations from software packages that enable users to build their own simulations. Some simulations are well designed and highly functional preprogrammed packages—the user cannot add any new functionality to the simulation, nor even examine its inner workings. GenScope, a genetics simulation (Figure 1), is an excellent example of a preprogrammed simulation. Other applications are simulation construction kits that enable the user to build a new simulation. The famous Pinball Construction Set is one of the first examples of such a kit. Another example, and one designed specifically for middle school students, is Cocoa [Smith, Cypher, and Spohrer; Smith, Cypher, and Schmucker] (Figure 7). Cocoa has been used to author simulations as diverse as:

- An interactive simulation of the plant life cycle, complete with falling rain, germinating seeds, pollinating bees, blossoming flowers, and pesky weeds. Clicking the ground plants more seeds, and clicking a weed removes it.
- A simulation of the spontaneous emission of light in a laser. [Schmucker 98a]
- A simulation of how rumors spread through a population.
- A simulation of the generation of Chinese characters.
- A simulation of pressure and volume of a gas at the molecular level.
- Lots of games, especially adventure games, mazes, and the re-implementation of some classics (Breakout, PacMan, and so on) (Figure 8) [Schmucker, October 1998].

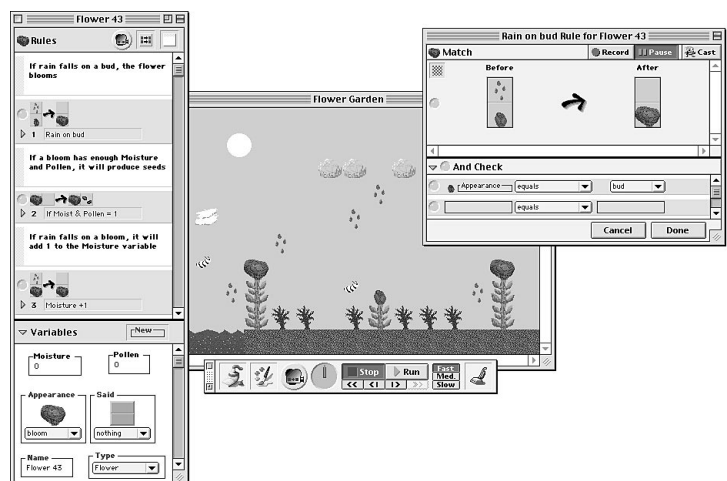


Figure 7. Cocoa. A simulation construction kit designed specifically for middle school students. The Cocoa user does not program in textual programming language, but rather constructs graphical rewrite rules by demonstration. The Cocoa program fragment shown in the right window states that when rain falls on a flower bud, the rain should be absorbed and the bud should change to a full blossom. The full collection of such fragments for the flower character is shown in the left window. The simulation itself plays out in the center window. Cocoa has been used successfully by tens of thousands of students around the world.

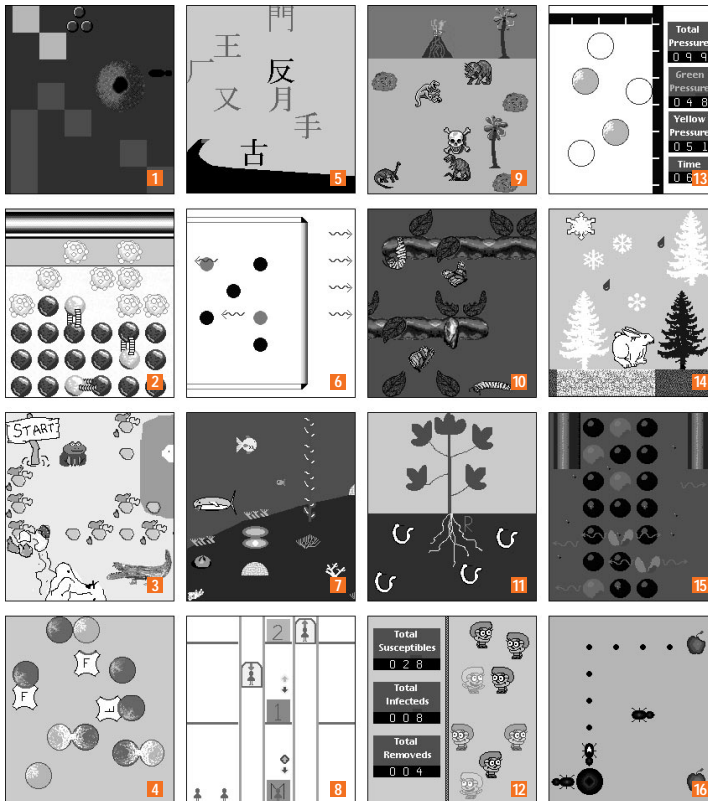


Figure 8. Examples of some of the best simulations written with Cocoa:

- | | | | |
|-------------------------|-------------------------------------|------------------------------------|----------------------------------|
| 1. Ant foraging | 5. Generation of Chinese characters | 9. The extinction of the dinosaurs | 13. Pressure and volume of a gas |
| 2. Evaporation models | 6. Inner workings of a laser | 10. Butterfly life cycle | 14. Why does it snow? |
| 3. Ecology of the swamp | 7. The coral reef | 11. Nematodes (a potato parasite) | 15. Nuclear reactor |
| 4. Catalysts | 8. Elevator queuing | 12. How a rumor spreads | 16. Ant foraging |

This factor of programmable versus preprogrammed is primarily one of focus and intent. All simulations take input, but for some simulations the input is program “code” or other nonnumerical input that will somehow be evaluated or executed as part of the computational model underlying the simulation. The input of preprogrammed simulations is usually numerical parameters that are used in the model but that don’t alter the model’s code.

Learning Technology Review

A Taxonomy of Simulation Software

Many consider the construction of a new simulation (or the modification of the inner workings of an existing one) to be an admirable educational goal in itself. However, even these constructivist proponents would agree that the amount of effort to produce a complete and full-featured, domain-specific observational simulation such as GenScope or Creatures2 is too large an effort for even a team of high school students. Programming a simulation would seem to be a more useful K–12 activity for single-topic simulations of limited depth, and in fact, this activity would probably prepare the student for a better, more critical use of a large, preprogrammed simulation.

Programmable simulation generators can be further divided into those that are designed specifically to construct simulations and those that provide general programmability. Microsoft Excel, for example, is being used as a simulation engine in both high school and university courses. Programmable simulation generators that are specific to the generation of simulations can be classified on the basis of the type of programming that the student uses. In Cocoa and AgentSheets (Figure 9) [Repenning and Ambach], for example, the student constructs individual rules for the objects that make up the simulation. In a procedurally based system, the student constructs subroutines that form the basis of the underlying computational model. Here is one example of such a subroutine from Star Logo, a procedurally based simulation generator that can be used to build simulations with hundreds or thousands of objects, all operating in parallel. This example is from an ant-foraging simulation, complete with pheromones marking the trail to diminishing food supplies:

```
to return-to-nest
  if nest?
    [setcarrying-food? false
     rt 180 fd 1 stop]
  tsetchemical chemical + drop-size
  setdrop-size (drop-size - 1.5)
  if drop-size < 1 [setdrop-size 1]
  uphill-nest-scent
  wiggle
  grid-step
end
```

Learning Technology Review

A Taxonomy of Simulation Software

In mathematically based generators, the student describes the behavior of the simulation by constructing difference equations that relate the value of some aggregate quantity at time t from that at time $t-1$ (Figures 10 and 11).

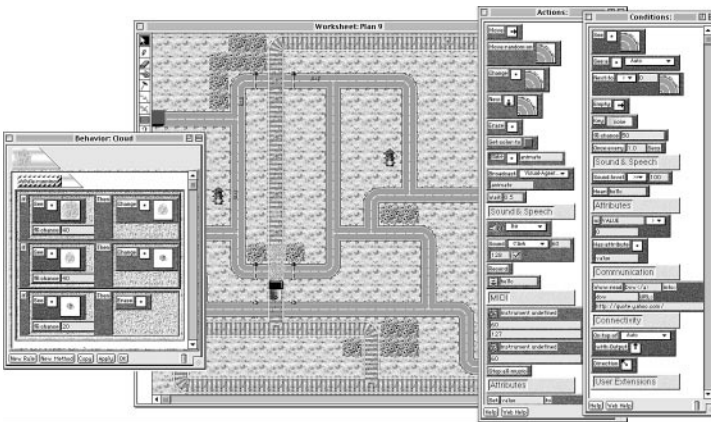


Figure 9. AgentSheets. In AgentSheets, as in Cocoa, the user constructs rules for each of the objects in the simulation being programmed. Unlike Cocoa, the rules are constructed by dragging conditions and actions from palettes (shown above on the right) into the rule set for the object. The simulation shown above is a simple pollution model diffusing pollution over time and space. Cars and factories are pollution sources contributing to pollution of the city of Sustainopolis. Forests decrease pollution. The rule set for the clouds of exhaust from the train is shown on the left. An English version of the exhaust cloud rules would be: "If I have the large appearance, then 40% of the time, change to the medium appearance. If I have the medium appearance, then 40% of the time, change to the small appearance. If I have the small appearance, then 20% of the time, disappear." The speed at which AgentSheets executes is truly astounding, and thus larger, more complex simulations are possible with AgentSheets than with Cocoa.

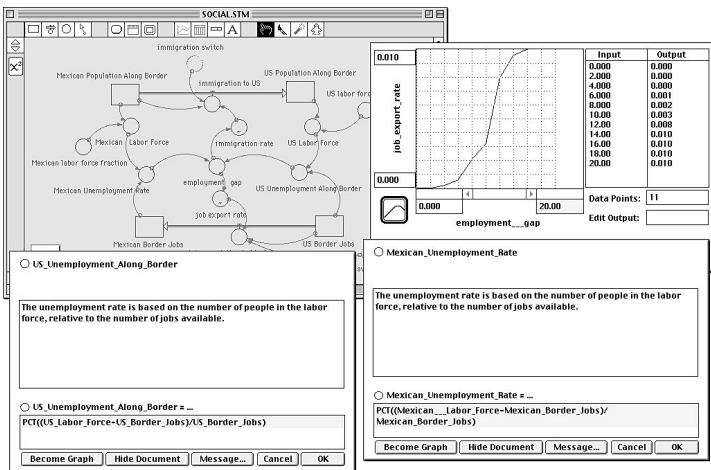


Figure 10. Stella. A simulation construction environment in which equations such as $US_Unemployment_Along_Border = Percentage((US_Labor_Force - US_Border_Jobs) / US_Border_Jobs)$ are used as difference equations to relate the values of aggregate measurements from one time increment to the next.

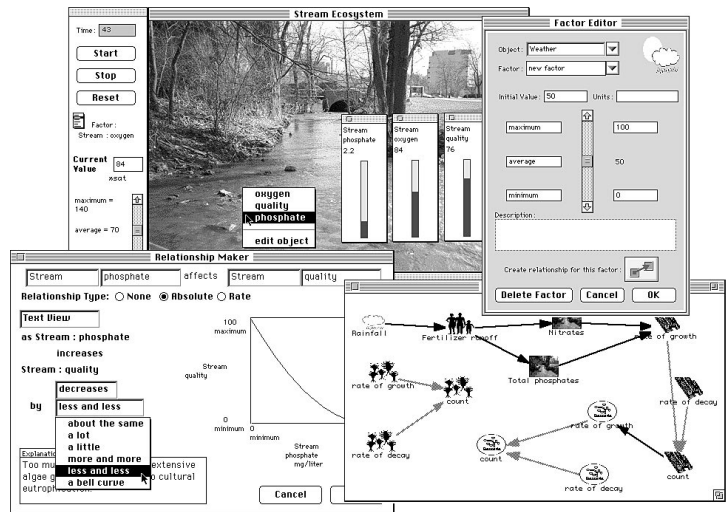


Figure 11. Model-It. Model-It presents a more approachable, user-friendly interface for a mathematically based simulation generator. As can be seen in the Relationship Maker window above, the equations can easily be entered with pop-up menus that have an English-like syntax. In this window, the student is entering the following "equation:" "As stream phosphate increases, stream quality decreases by less and less." The various objects and factors in the model are represented with unique icons, and the model can execute on a picture backdrop suggestive of the study being performed.

Domain-specific or domain-independent

The most important difference between software packages that enable users to build their own simulations is whether the package restricts users to building simulations only in certain domains. The factor of domain specificity is primarily one of scope. (Note that all preprogrammed simulations are necessarily domain-specific, so this factor applies only to programmable simulations.) Domain-specific packages such as the Pinball Construction Set or Interactive Physics have deliberately limited their scope (to the construction of pinball games or certain types of physics simulations, in the case of the these two examples) in order to gain performance, memory utilization, ease of learning, or ease of use. Domain-independent packages such as Cocoa, Model-It, and Star Logo (or even very general-purpose packages such as Microsoft Excel or Visual Basic) attempt to cover a wider range of possible simulations, with their own trade-offs in ease of use, ease of learning, and performance.

Immersive or observational

The biggest differentiating factor among complete and full-featured simulations was the point of view given to the user. How involved is the user in the simulation? Is the user an observer on the side, or is the user actually a part of the simulation, as in a flight simulator or a first-person adventure game such as Doom or Marathon? When the user is given the impression of actually being

Learning Technology Review

A Taxonomy of Simulation Software

in the simulation, the simulation is immersive. If the user is an observer (albeit perhaps with god-like powers to control the simulation), the simulation is observational. Most of the simulations mentioned so far are observational (GenScope, SimCity, ActivChemistry, The Incredible Machine, and most simulations built with Cocoa). Immersive simulations such as first-person “shoot-em up” games (Doom, Marathon, Quake, and so on) and flight simulators are usually entertainment simulations. ToonTalk, a programming environment (Figure 12) has many aspects of an immersive simulation construction kit [Kahn].

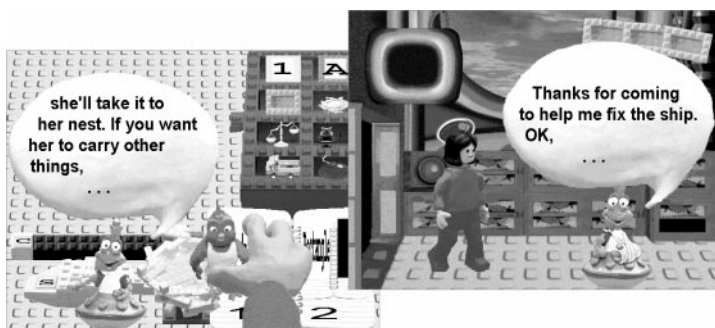


Figure 12. ToonTalk. An immersive programming environment. In ToonTalk, the user is sometimes represented by a hand that comes out from the bottom of the screen (left screen shot) and that is controlled by the mouse and keyboard. The hand is used to grasp and manipulate the ToonTalk objects, which represent the elements of a program: numbers, letters, procedures (robots), subroutines (houses), and inter-process communication (birds that fly between the houses). At other times the user is represented by a customizable, animated figurine (right screen shot). When the user is confused, Marty the Martian (in both screen shots) is always available to help and Marty's assistance is both spoken and shown in textual balloons.

Single concept or complete and full-featured

While many of the simulation packages presented so far are large applications with dozen or hundreds of features, a simulation does not have to be so feature-full (or occupy hundreds of megabytes on disk, or consume tens of megabytes of RAM) to be useful. Small, single-concept simulations, with their smaller learning curve, can be extremely useful educationally. Yves Pelletier has created many such small, focused simulations, of which YP Collisions (Figure 13), which models a two-dimensional collision between two particles, is one example. In YP Collisions, the user chooses the mass, the initial velocity (magnitude and direction), and the coefficient of restitution of the particles. An animation shows the motions of the particles before and after the collision, as well as computed quantities (final velocity of each particle, the initial and final momentum, and so on). YP Collisions is a very useful simulation for physics students, yet it uses just 1.2 megabytes of RAM and 0.23 megabytes of hard disk space.

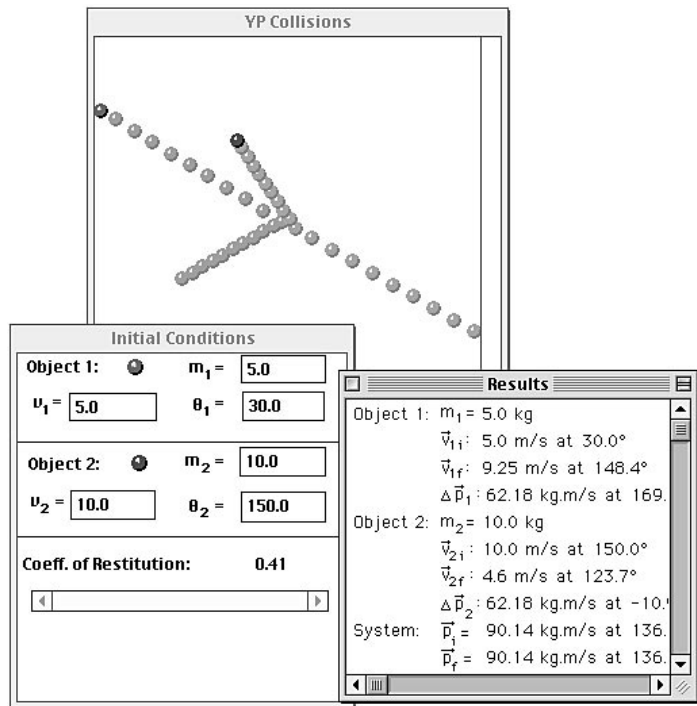


Figure 13. YP Collisions. A single-concept simulation that models two-dimensional collisions between two particles. In YP Collisions, the user chooses the mass, the initial velocity (magnitude and direction), and the coefficient of restitution of the particles. The application then shows the positions of the particles before and after the collision, as well as computed quantities (final velocity of each particle, the initial and final momentum, and so on).

Java is particularly well suited for building single-concept simulations and delivering these over the Internet as applets. (The Boids applet shown earlier is one of the best-known simulation applets.) There are literally thousands of such applets; so many, in fact, that the task of organizing these applets is a large job in itself. Fortunately, organizations such as the Educational Object Economy (www.eoe.org) assist both teachers and students in finding the appropriate applet for a variety of educational purposes.

Programmable or construction kit

The distinction between a programmable simulation and a simulation construction kit is difficult and is usually an even finer distinction than between programmable and preprogrammed simulations. By their nature, construction kits allow their users to build new simulations from parts, and so do programmable simulations. The difference is in the nature of the parts and in the types of data the user is allowed to input while building the simulation. As was done for preprogrammed versus programmable simulations, if the input is only numerical parameters to be used by the constructed parts in the

simulation, then the application was classified as a construction kit. If the input was some sort of code that would become part of the simulation under construction, then the application was classified as a programmable simulation generator. In Interactive Physics and the Pinball Construction Set, for example, the user can input equations that become part of the simulation, so these are programmable. In The Incredible Machine and ActivChemistry, only numerical parameters or simple choices are available to the user, so these were classified as construction kits.

One very difficult case to classify was the Widget Workshop (Figure 14). In this application, the user constructs code in a graphical way, by wiring together small building blocks, most of which are programming elements. While many programming environments, such as Visual Basic or Star Logo, use text to construct new programs, not all programming environments are textual. In some programming environments, the programmer constructs new programs by moving, connecting, or manipulating pictures that represent objects, actions, or programming elements. Such graphical programming systems are often extraordinarily productive programming environments [Schmucker, 1993; Schmucker, 1994]. For this reason, Widget Workshop is classified as a programmable simulation generator.

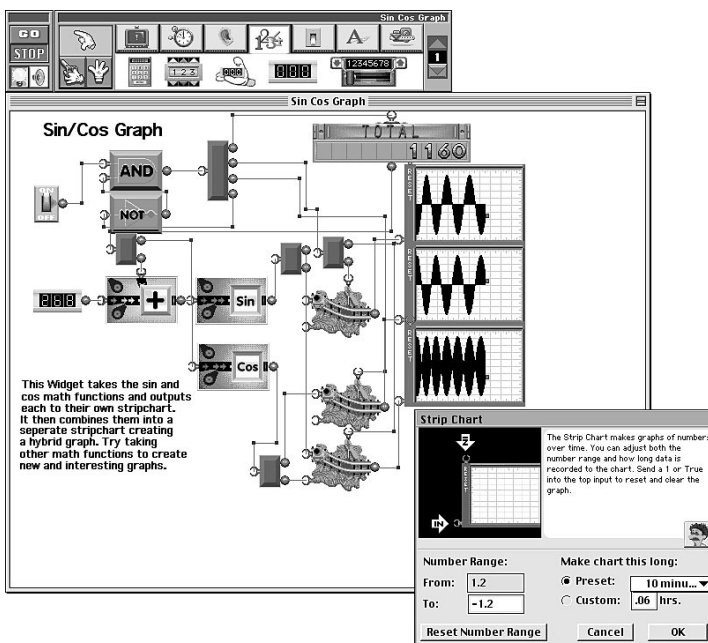


Figure 14. Widget Workshop. A programmable simulation generator in which the user programs with a graphical, not a textual, language.

Learning Technology Review

A Taxonomy of Simulation Software

Physical phenomena or intellectual discipline

While many simulations present the virtual experience of some physical reality (ActivChemistry, Flight Unlimited) or fantasy world (Quake, Marathon), others model a body of knowledge, such as algebraic geometry (Graphing Calculator [Figure 15]), geometry (Geometer's Sketchpad [Figure 16]), or programming (ToonTalk [Figure 11]). These "knowledge simulations" are full simulations in the sense defined in this article.

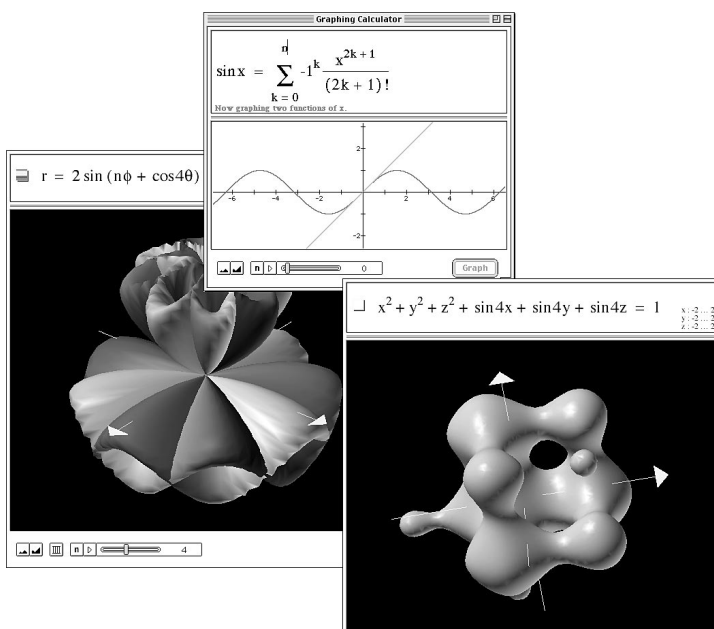


Figure 15. Graphing Calculator. A simulation for algebra and geometry with a delightfully elegant user interface. In the top window, the convergence of the Taylor Series is being interactively simulated; in the middle window, a family of equations in polar coordinates is being explored; and in the bottom window, a complex trigonometric equation in three dimensions is being visualized.

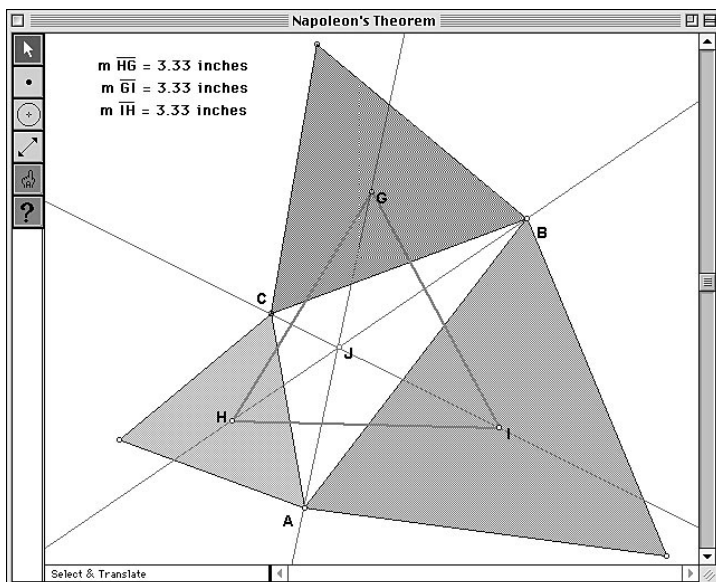


Figure 16. Geometer's Sketchpad. A construction kit for "simulations" of geometric theorems. The true power of Geometer's Sketchpad is that the drawings can be manipulated by the user and yet still maintain their geometric "knowledge." The above figure demonstrates Napoleon's Theorem, "The centroids of equilateral triangles constructed on the sides on an arbitrary triangle form an equilateral triangle." (That is, triangle GHI remains an equilateral triangle regardless of how the vertices of triangle ABC are moved.) A "proof" of Napoleon's Theorem in Geometer's Sketchpad is an interactive drawing in which points A, B, and C can be moved around the plane, and regardless of the new positions of A, B, and C, triangle GHI remains equilateral. A rich simulation of geometry such as Geometer's Sketchpad enables students to explore and discover mathematics, and several high school students have published original mathematical results derived from Geometer's Sketchpad work. [Key Curriculum Press]

Why bother?

There is anecdotal and experimental evidence that simulation use and simulation construction are valuable educational experiences. Anecdotally:

- "Modeling can often help students see relationships that are masked by the complexities of the real world. For example, models can ignore friction and air resistance. They can slow down or speed up time and back up or repeat an event. Models can expand the scale of investigation to include molecular level or a star system. Modeling can explore areas where equipment is too expensive and experiments are dangerous.
- Mathematics, science, and technical skills are taught in an applied and integrated manner. A limited number of topics are explored at depth.
- Modeling supports inquiry learning and promotes student collaboration. Students work with new methods of problem solving.
- Modeling provides skills for emerging career areas. Modeling is used extensively in science and industries." [Envision It!]

Learning Technology Review

A Taxonomy of Simulation Software

As another example, middle school students who used Cocoa for a semester noticeably improved the quality and the quantity of their hypothesis formation in science class [Lewis].

By studying National Assessment of Educational Progress (NAEP) scores for U.S. students, Wenglinsky found that classroom simulation use was associated with academic achievement in math and also with social improvements in areas such as student motivation, tardiness, absenteeism, vandalism of school property, and so on [Wenglinsky; Milken Exchange]. In drawing these conclusions about simulation use, Wenglinsky's study took into account factors such as gender, socio-economic status, region (Northeast, West, Central, and Southeast), community status (urban, suburban, and rural), and governance of the school (public or private), among other variables.

I hope this simulation taxonomy will be of some assistance to educators who want to begin or continue simulation use in the classroom, or for developers of educational software who want to see where the "holes" are in the product landscape.

References

Alessi, S. M., and Trollip, S. R. *Computer-based Instruction: Methods and Development*. Englewood Cliffs, NJ: Prentice-Hall, 1991.

Bruce, Bertram C. "Educational Technology: Tools for Inquiry, Communication, Construction, and Expression," College of Education, University of Illinois at Urbana-Champaign (<http://www.ed.uiuc.edu/courses/edpsy-387/Taxonomy-Graph/Ed-Tech-Taxonomy.html>).

Envision It! project description, an NSF-funded project to train secondary school teachers in the Minneapolis-St. Paul area to use the methods and tools of computational science to enrich math and science teaching (http://www.ties.k12.mn.us/envision/97/project_description.html).

Horwitz, Paul. "Designing Computer Models that Teach" in *Models that Teach*, Wally Feurzeig and Nancy Roberts (eds.), in preparation.

Kahn, Ken. "ToonTalk—An animated programming environment for children." *Journal of Visual Languages and Computing*, Volume 7, June 1995, pp. 197–217.

Key Curriculum Press. http://www.keypress.com/product_info/sketch30.html

Learning Technology Review

A Taxonomy of Simulation Software

Krauss, Lawrence M. *The Physics of Star Trek*. New York: Basic Books/HarperCollins, 1995.

Lewis, Clayton, University of Colorado at Boulder, personal communication.

Milken Exchange. <http://www.milkenexchange.org/>

Oppenheimer, Todd. "The Computer Delusion," *Atlantic Monthly*, July 1997.

Persichitte, Kay. "Basic Criteria for Selecting and Evaluating Instructional Software," University of Northern Colorado (http://www.coe.uh.edu/insite/elec_pub/html1995/1011.htm).

Repenning, Alex, and Ambach, James. "Tactile Programming: A Unified Manipulation Paradigm Supporting Program Comprehension, Composition, and Sharing," *Proceedings of Visual Languages 1996*, Boulder, CO: IEEE Computer Society, 1996.

Schmucker, Kurt. *Fuzzy Sets, Natural Language Computations, and Risk Analysis*. Computer Science Press, 1983.

Schmucker, Kurt. "Serius—Re-implementing the MacApp Samples without Programming," *FrameWorks—The Journal for Software Developers Using Object Technology*, Volume 7, Number 4, July/August 1993, pp. 28–35.

Schmucker, Kurt. "Prograph CPX—A Tutorial," *MacTech Magazine*, Volume 10, Number 11, November 1994, pp. 69–82. (Reprinted in *Object Oriented Application Frameworks*, Ted Lewis [editor], Manning Press and Prentice Hall, 1995, pp. 291–331.)

Schmucker, Kurt. "The Cocoa 'Worlds of Science' Contest," *Learning Technology Review*, Spring 1998 (<http://www.apple.com/education/LTReview/spring98/contest.html>).

Schmucker, Kurt. "The Cocoa Collection," Version 3.0, Apple Computer, Inc., October 1, 1998.

Smith, David Cantfield, Cypher, Allen, and Schmucker, Kurt. "Making Programming Easier for Children," *ACM Interactions*, Volume 3, Number 5, Sept./Oct. 1996, pp. 58–67.

Learning Technology Review

A Taxonomy of Simulation Software

Smith, David Cantfield; Cypher, Allen; and Spohrer, Jim. "KidSim: Programming Agents without a Programming Language," *Communications of the ACM*, Volume 37, Number 7, July 1994, pp. 54–67.

Smith, Thomas. A review of simulated and microworld environments (<http://coyote.ultralab.anglia.ac.uk/simulation/report.htm>).

Taylor, R. P. *The Computer in the School: Tutor, Tool, Tutee*. New York: Teachers College Press, 1980.

Wenglinsky, Harold. *Does It Compute? The Relationship between Educational Technology and Student Achievement in Mathematics*. Educational Testing Service, September 1998.

URLs

Listed below are URLs for most of the simulations studied in building the taxonomy described in this article.

| | |
|-------------------------|--|
| ActivChemistry | http://www.salamanderinteractive.com/ACgetAC.html http://heg-school.awl.com/bc/companion/actchem/ac.htm |
| AgentSheets | http://www.agentsheets.com/ |
| AquaZone | http://www.aquazone.com/home.html |
| AutoSim | http://www.interpretive.com |
| Barbie Fashion Designer | http://www.mattelmedia.com/barbie/fashiondesigner/shopping/index.htm |
| B ² Logic | http://www.beigebag.com |
| B ² Spice | |
| BioLab Fly | http://www.pierian.com |
| BioLab Frog | |
| BioLab Pig | |
| BioSIGHT | http://biosight.usc.edu |
| Boids | http://hmt.com/cwr/boids.html |
| BuildSim | http://www.tritera.com/products/web_buildsim/bs_page1.html |
| Catz | http://www.petz.com/central/default.asp |
| Cocoa | http://www.crim.ca/~hayne/Cocoa http://www.kidsdomain.com/games/cocoa.html |
| Creator | http://www.stagecast.com |
| Creatures2 | http://www.creatures2.com http://www.creatures.co.uk http://www.creatures-lab.com |
| Data Desk | http://www.datadesk.com/main.htm |
| Dogz | http://www.petz.com/central/default.asp |
| Doom | http://www.idsoftware.com |
| Excel | http://www.microsoft.com/excel?RLD=41 |
| Extend | http://www.imaginetthatinc.com/home.html |

Learning Technology Review

A Taxonomy of Simulation Software

| | |
|------------------------------|--|
| Fatal Abyss | http://www.fatalabyss.com |
| Flight Unlimited | http://www.lglass.com |
| Fly a Cell! | http://www.hps-inc.com/products/le/education.html |
| GalaxSee | http://www.shodor.org/master |
| Galapagos: Mendel's Revenge: | http://www.anark.com/Galapagos/index.shtml |
| Gazillionaire | http://www.lavamind.com http://www.gazillionaire.com/gaz.html |
| GenScope | http://genscope.concord.org |
| Geometer's Sketchpad | http://www.keypress.com |
| Graphing Calculator | http://www.nucalc.com |
| Gravity Simulator | http://groovysoft.com |
| Incredible Machine | http://www.presage.com/pTEMIM.html http://www.sierra.com |
| Interactive Physics | http://www.knowledgerevolution.com |
| Java | http://www.javasoft.com |
| Klik & Play | http://www.maxis.com/games/index.html |
| Klingon Honor Guard | http://www.wizworks.com/macsoft |
| LabView | http://www.natinst.com/labview |
| LEGO MINDSTORMS | http://www.legomindstorms.com http://www.pitsco-legodacta.com/html/mindstorms.html |
| Lode Runner | http://www.wizworks.com/macsoft/ode_r2/lode_r2.html |
| Marathon | http://www.bungie.com |
| Mathematica | http://www.wolfram.com |
| Milky Way Cafe | http://www.illawarra.net.au/rush |
| Model-It | http://www.cogitomedia.com |
| Myst | http://www.cyan.com/ |
| OpRat | http://www.thecraft.com |
| PharmSim | http://www.interpretive.com |
| PolyLife | http://members.home.net/facticsmax/polylife.html |
| Profitania | http://www.lavamind.com/edu.html http://www.profitania.com |
| Quake | http://www.idsoftware.com/ |
| RoboLab | http://www.cceo.tufts.edu/graphics/robolab.html http://www.lego.com/dacta/robolab/defaultjava.htm http://www.pitsco-legodacta.com/html/robolab.html http://www.natinst.com/robolab/ |
| Salmon Odyssey | http://ingenuity.com |
| ServiceSim | http://www.interpretive.com |
| Silent Hunter | http://www.learningco.com/products/default.htm |
| SimCalc | http://www.simcalc.umassd.edu/simcalc/ |
| SimAnt | http://www.maxis.com/games/index.html |
| SimCity | |
| SimsIsle | |
| SimLife | |
| SimMars | |
| SimTown | |
| Simulink | http://mathworks.com |

Learning Technology Review

A Taxonomy of Simulation Software

| | |
|-----------------------------|---|
| Sniffy, the Virtual Rat | http://ic-unix.ic.utoronto.ca/inst_tech/itp/html_files/file_52.html http://icubed.com/users/mfichten/rats/cnn04.html http://www.thomson.com/brookscole/technology/products/sniffy/4.5/mtsnf.html |
| Star Logo | http://www.media.mit.edu/~starlogo |
| Star Wars DroidWorks | http://www.droidworks.com |
| Starship Creator | http://www.simonsays.com/startrek/library/creator |
| Stella | http://www.hps-inc.com |
| StratSim | http://www.interpretive.com |
| Tamagachi | http://www.bandai.com |
| Thinking Things | http://www.edmark.com/prod/tt |
| ToonTalk | http://www.toontalk.com |
| Virtual Frog Dissection Kit | http://www.mirrors.org.sg/vfrog http://www-itg.lbl.gov/vfrog |
| Virtual Lizzy Borden House | http://www.halfmoon.org/borden |
| Virtus WalkThrough | http://www.VIRTUS.com |
| Visual Basic | http://msdn.microsoft.com/vbasic |
| Widget Workshop | http://www.maxis.com/games/index.html |
| YP Collisions | http://www.kagi.com/pelletier |
| Yoot Tower | http://www.yoot.com http://www.yootmacfirst.com |
| Zapitalism | http://www.zapitalism.com http://www.lavamind.com/edu.html |
| 7th Guest | http://www.vie.com/ |

Acknowledgments

The author gratefully acknowledges the assistance of many colleagues who commented on earlier drafts of this article. Among these, Kristina Hooper Woolsey, Peter Jensen, Bard Williams, and Nora Roa were especially helpful both for their comments and for the many spirited discussions about simulations and their use.

Learning Technology Review

A Taxonomy of Simulation Software

About the author

Kurt Schmucker has worked at Apple Computer (kurts@apple.com) for more than 11 years in a variety of capacities, including research manager and engineering manager for Cocoa, Apple's simulation authoring toolkit for kids. Currently he is the senior manager for leadership initiatives in Apple's K-12 Education Marketing group, as well as the editor of the Learning Technology Review.

Kurt is the author of Object-Oriented Programming for the Macintosh (Hayden, 1986), The Complete Book of Lisa (Harper & Row, 1984), and Fuzzy Sets, Natural Language Computations, and Risk Analysis (Computer Science Press, 1983), as well as many articles in technical and government journals.

Kurt has master's degrees in both mathematics (Michigan State University) and computer science (The Johns Hopkins University). He has also completed all requirements for his Ph.D. in computer science at George Washington University, with the exception of his dissertation (ABD). While he has no education credentials, Kurt teaches fourth and fifth graders a style of Japanese fencing one afternoon each week at an elementary school in San Jose, California.