Project Number: 51-JQW-0201

Software for Science and Math Education

An Interactive Qualifying Project Report

Submitted to the Faculty

Of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Bachelor of Science

by

Charles J. Martin

Date: August 21, 2002

1. Education
2. Computers
3. Science

Table of Contents

List of Illustrations

Abstract

The purpose of this IQP is to study simulation-based software to aid in teaching middle and high school science and math. Specifically, we researched three software packages StarLogo, Stella, and ModelIt. We researched:

1. What skill and knowledge are required of teachers to use these programs and
2. When each program would be best used, and what combinations of programs would be best and easiest to implement lessons in standard science and math curriculum.

1.0 – Background

1.1 – General Background

Researchers have often looked at the introduction of technology into elementary and high school education. As technology expands forward, students are aided both by the technology-improved curriculum and by learning to make use of that technology in their lives. The intent of this project is to focus on modeling and simulation-based programs for math and science education in middle and high school. Each of three programs, "StarLogo", "ModelIt", and "Stella", was researched and compared.

The research and comparison came in a few major forms. First, I researched a candidate set of programs and read about them from literature and web sites. I looked into the workings of each program, running through the tutorials of each and building simple models in each. I researched models and lessons prepared by teachers and researchers that are available on line. From doing that work, I came to understand the general capabilities of each program, and then decided which science and math topics can be reasonably taught through each program. The final result of that can be seen in Table 3.

Finally, I took the programs to a compiled list of difficulty requirements to see how difficult each program was based upon multiple criteria (See Tables 1 and 2). I researched several educational standards for technology education. The criteria were built from two sets of rubrics and standards, customizing them to better fit the programs in this research. The preliminary list of rules can be found in Tables 1 and 2.

The simulation-based programs researched were chosen based upon appropriateness and general similarity in their value for 6-12th grade math and science lessons. The initial list of programs chosen was taken from the MIT project collaboration [1]. The list contained programs for Complex Adaptive Systems, or advanced simulations set up in such a way that only certain parts need be seen by the user, allowing the more complex parts to be left hidden. Every program in the list could be used for lessons by students in the classroom.

1.2 – Specific Programs

Three programs were researched in this project: ModelIt, StarLogo, and Stella. While there were striking similarities between ModelIt and Stella, the three programs differed greatly. ModelIt is the best of the three for simplifying and generalizing complex systems. StarLogo, which creates complex systems through the use of many turtles, is the best for simulating models and decentralized systems. Finally, Stella is the best for calculating models accurately based upon potentially complex relationships.

The original list of programs includes Matlab. It was removed from consideration because it is too complex and it is not widely (as compared with the other three) used for modeling and simulation in middle and high school. The program itself was baffling to some college students, and even Computer Science students. It is impossible to assume an average high-school teacher, or high-school students, could use it for teaching or learning.

Each program has its own uses, advantages and disadvantages. Each program is of varying difficulty that alters the usability of the program. The most powerful programs tend to be the most complex, for which StarLogo seems a likely case. The simplest programs tend to be the least versatile, as can be seen in ModelIt. That said, each program still needs to be explored more deeply for its usefulness.

1.2.1 – ModelIt

"ModelIt allows students to easily build, test, and evaluate qualitative models." [2] Effectively, ModelIt is a simple WYSIWYG modeling program, which lets the user see every piece of the model, starting by building the most general aspects and ending with the specific variables and relationships between them. It claims to allow students to "create models that represent their theories about the scientific phenomena being investigated and run simulations in order to test their models." [2] It allows the creation of simple models consisting of broad objects whose parameters are linked by general graph-based relationships.

The simplicity of ModelIt is one of its selling points. According to the official homepage: "students at any mathematical level will be able to… build, test, and evaluate qualitative models, without knowledge of the underlying calculus." [3] This statement is true on multiple angles. Instead of having complex equations defining variables in terms of relationships between them and derivatives thereof, each connected relationship shows a graph of an equation type, be it linear, constant, bell-curved, or exponential. Each relationship is connected by pictures and lines, not complex growth equations and their derivatives (see Figure 1). On the other hand, it is almost (if not completely) impossible to get numerically accurate answers for most complex systems since only minimum, maximum, and general curve can be set.

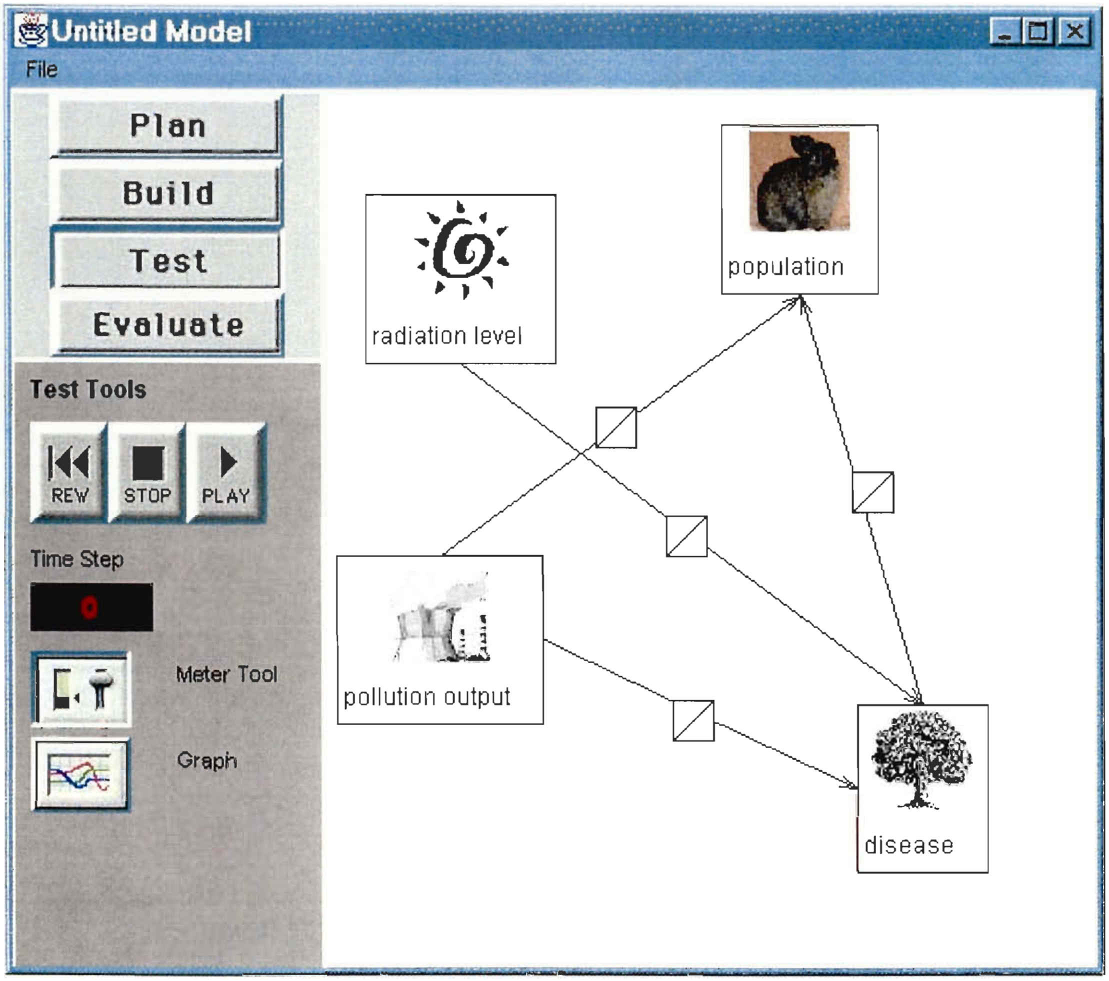

Figure 1: A diagram from the ModelIt web site indicating a test of linear relationships of four inter-related factors.

Another important feature of ModelIt is its ability to "run simulations to test and analyze" [3] models created by students. Once the model has been built, the user can switch into test mode. ModelIt then provides a set of sliders in front of the model. The user can slide the independent variables (those that are not pointed to by other variables in a relationship) and watch the changes in the dependent ones, both visually on the sliders and graphed on a separate window.

From these features, the most obvious use of ModelIt would be population and food models, though any cause-and-effect model would work as long as exact calculations are not expected. In population models, the change of animal population can affect food supply, which in turn can affect survivability of the animal, etc.

1.2.2 – StarLogo

Instead of modeling a graphical representation of relationships between objects, then calculating those relationships based upon internally or externally included math equations, StarLogo creates a simulation of the system to be modeled. To model a simulation, StarLogo creates a grid of "turtles", or active participants, and "nodes", or passive participants, giving a set of typically simple instructions for each turtle to act out. For example, a model of termites collecting woodchips gives a simple repeated set of commands that a "termite" would do to move about randomly, find a woodchip, carry that woodchip to a nearby pile, and drop it. Each step, and even the entire set of steps, is very simple, but when run repeatedly, the termites end up moving all the woodchips successfully to a set of piles. Figure 2 depicts the termite simulation.

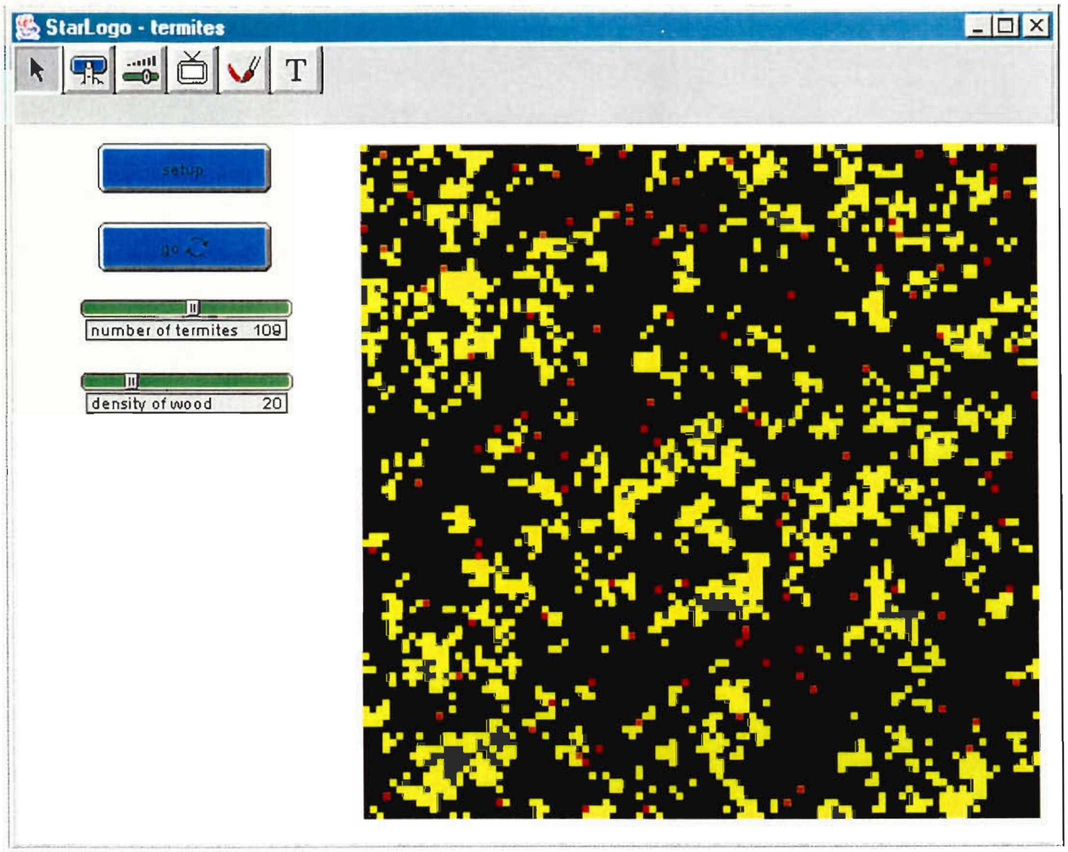

Figure 2: Termites in StarLogo with Complete Front-End

A Complex Adaptive System is a system in which "many independent elements or agents interact, leading to emergent outcomes that are often difficult (or impossible) to predict simply by looking at the individual interactions" [1] The termite model is an example of a decentralized system. "In decentralized systems, orderly patterns can arise without centralized control." [4] The main concept of StarLogo is to model decentralized systems, and a significant majority of science experiments can be conceptualized as decentralized systems. Many mathematical models can, with some creative coding, be shown in a decentralized system as well.

An example of this creative coding is creating graphs out of turtles. A set of turtles can be placed horizontally, one at each x location, and then told to place their y locations based upon a function. The effect is that the turtles will fall into place along the points of a graph, mimicking that graph.

StarLogo, unlike the other two programs researched, is freeware. Teachers need not pay to make use of it, but might not be able to expect the same kind of technical support that would be expected of the other two programs. Several mailing lists and forums exist for StarLogo users to discuss and share ideas. However, there are no technical support lines and support staff to fix everyone's problems.

1.2.3 – Stella

The third program researched was Stella. Stella advertises itself to be "a tool for supporting learner-directed learning." [5] It is, like ModelIt, a tool for modeling by drawing relationships in a WYSIWYG environment between variables. Unlike ModelIt, Stella allows the user to define exact equations, instead of simply basic curves. The user can even manually move the curve, or convert to a step-based graph.

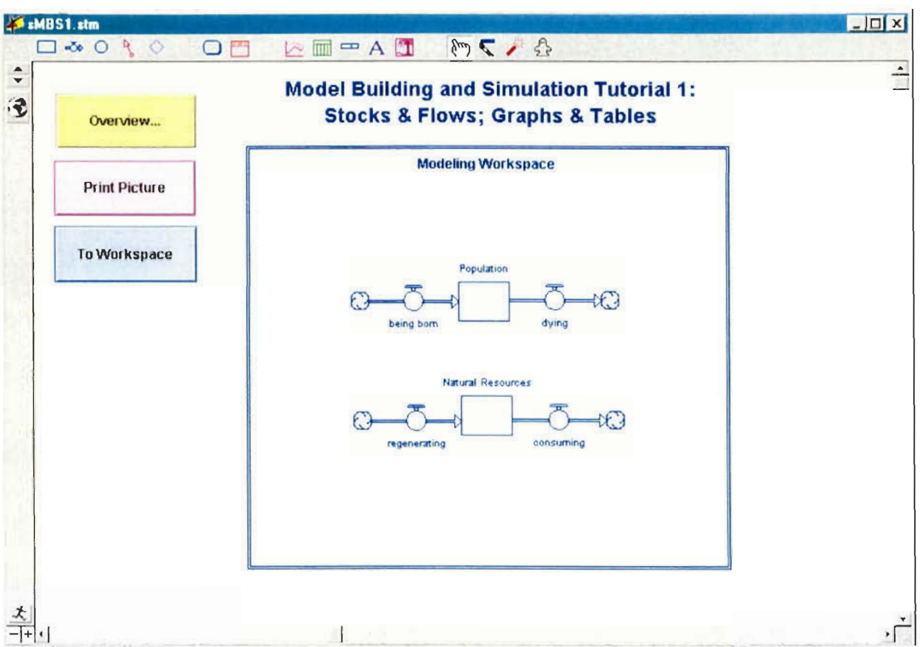

Figure 3: Stella Population Model

Stella can conceivably run anything that ModelIt can and more. This does not make ModelIt useless. It means Stella could be argued to be a more complete program as long as the user is capable of and willing to use the obviously more complex interface. In return for that more complex interface, there are many features that Stella has that ModelIt lacks.

As was previously mentioned, Stella has more accurate relationships, with definable functions instead of choosing from a list of six different types of curves. Additionally, Stella supports feedback relationships, allowing a value of a variable to be affected mathematically by the previous value of that variable. This feature is necessary for accurately modeling changes in population, which is invariably based on the previous population.

Stella also allows the designer to create a user-friendly front-end. This front-end makes building the model and simulation a bit more complex, but when completed, it allows a user to learn from the model with significantly less knowledge than the designer, both in knowledge of the subject and knowledge of computers. With a front end built, Stella becomes easier for a student to use than ModelIt, while being more complex to the teacher to build. The front end can be used for professional-looking reports. Features such as charts, fields, buttons to change data, and a one-click way to get a walkthrough of the entire underlying model, combine to create an easy and quick-to-read summary of the work done. In part, the increased complexity of Stella is due to the fact that Stella is used for business and scientific research, as well as student lessons.

The most important feature Stella has over ModelIt is the ability to create an animated walkthrough. While this walkthrough is perhaps the most complex aspect of the program, properly done, a click of a button will run a slide-show of the model, showing a section of the model at a time, with captions explaining the meaning of each section. As an addition to a presentation, the walkthrough can be used to explain, quickly and efficiently, to those who are technically adept how the underlying structure works. The structure is also kept from those who do not need to know it.

1.3 – Education Standards

This research made use of two sets of technology standard for educators (a performance indicator and a set of rubrics) to define a list of expected requirements from which I could define the difficulty levels of each program. How the data was used is explained in section 2.2.0, but this section intends to detail the two sets of rubrics.

1.3.1 – Nebraska Department of Education

The Nebraska Department of Education derives its standards and successful performance from the standards and performance indicators found in the ISTE (International Society for Technology in Education) standards. The format of the Nebraska standards document is: the ISTE standard, a set of ISTE indicators, and finally a list of examples of Nebraska performance indicators. Table 1 shows the list of criteria that are used for this study. For each criteria, the table shows the Nebraska and the ISTE standards, then the criteria derived.

"ISTE Standard…Teachers demonstrate a sound understanding of technology operations and concepts." [6] "Educators . . . Use basic computer operations such as editing, file management, printing, e-mail, multi-tasking, and networking. " [6]
Difficulty Rules Created: 1. Use basic computer operations. Computer Literate.
"Educators… Utilize help or support resources in solving problems. " [6]
Difficulty Rules Created: 1. Capable of solving problems with included support.
"ISTE Indicators… Teachers design developmentally appropriate learning opportunities that apply technology-enhanced instructional strategies to support the diverse needs of learners." [6] "Educators… Know how to assess, select, and use a variety of tool-based and content-based software to support learning. " [6]
Difficulty Rules Created: 3. Capable of deciding which software to use for a given lesson. 4. Capable of designing lessons to be used with specific software tools.
"Educators… Correlate the use of technology in learning environments to the application of technology in society." [6]
Difficulty Rules Created: 5. Able to use technology to a business level and translate to an educational one
"Educators… Understand and apply the characteristics of learners and the nature of the learning task to the selection and use of technology-based instructional strategies and presentation techniques. " [6] "Educators…Use technology to facilitate effective learner-centered instruction." [6]
Difficulty Rules Created: 6. Design interactive lessons in a program
"Educators… Use technology to communicate with others, including educators, administrators, parents, and experts." [6] "Educators… Use multimedia, hypermedia, and telecommunications to support effective instructional activities for lessons, presentations, demonstrations, and student projects." [6]
Difficulty Rules Created: 7. Design lessons specifically used in the high-tech venue
"Educators… Identify how the utilization of technology enhances student achievement. " [6]
Difficulty Rules Created: 8. Able to gauge enhanced achievement through use of technology

Table 1: Difficulty Rules from NDE Standards List

1.3.2 - Scott County, Kentucky Board of Education

The Scott County Board of Education "developed a rubric under each of the criteria required by the state with the hope that this will help teachers achieve [the Kentucky teaching] standard." [7] The rubric contains a list of 16 items, with sets of 4 quality levels defining where a teacher is in reaching the standard. The four levels, "Orientation", "Preparation", "Application/Implementation", and "Refinement", represent a scale of values that the teacher can attempt to attain to fulfill the standard. Assumedly, reaching the third rank is sufficient to reach any given standard.

"Operates a multimedia computer and peripherals to install and use a variety of software… Preparation: When given step-by-step instructions, I will attempt to install software on my own. Application/Implementation: When presented with new software, I can install it myself. " [7]
Difficulty Rules Created: 9. Basic ability to install software without support.
"3. Demonstrates knowledge of the use of technology in business, industry, and society… Application/Implementation: I stay informed on technology issues in society via various types of media. " [7]
Difficulty Rules Created: 10. Keep up to date on advancements related to the tool, and read online newsgroups regarding the tool's use in your field
"11. Facilitates the lifelong learning of self and others through the use of technology… Application/Implementation: I provide opportunities for my students to explore/practice new technologies. " [7] "13. Applies research-based instructional practices that use computers and other technology… Application/Implementation: I use research-based instructional practices. " [7]
Difficulty Rules Created: 11. Able to research online for lesson plans.

Table 2: Difficulty Rules from Scott County Standards List

The above table shows the criteria extracted from the Scott County rubric. While only three of the 11 difficulty criteria were obtained from it, those three were valuable and important. There were fewer criteria obtained from Scott County's standard in part because of the redundant points were taken from the Nebraska standard. The greater reason is perhaps that the list of rubrics was too specific and tended to touch topics unrelated to the use of computer tools in math and science lessons.

2.0 – Research

2.1 – Program Capabilities

While some simulation tools may be better or worse than others, usually the choice of tool is based upon the particular lesson a teacher wants to present. Each tool has a vastly different array of features that cater to different types of lessons. A teacher would probably do best to make use of multiple tools based upon the lessons to be taught. Table 3 enumerates a list of Complex Adaptive System topics represented in Math and Science standards taken from the MIT Project Statement [4] and how well each program can be used to teach the topic.

KEY
able: defines whether or not the program can illustrate a given
y = yes, it is within the capabilities of the program
Y = yes, this is part of the definition of the program's workings
n = no, the simulation cannot be built in the program
s = some, but not all, aspects of the simulation can be illustrated in the program
p = yes, but only by using the workings of the program in a unique way that is not in the original intent of the program

Curriculum Topics	Programs		
	STELLA	**STARLOGO**	**MODELIT**
<u>**SCIENCE**</u>	able?	able?	able?
MIDDLE SCHOOL			
Resources In Ecosystems	y	y	y
Population Dynamics	y	y	y
The Food Web	y	y	s
Chemistry of Living Systems	n	y	n
HIGH SCHOOL			
Genetics	n	s	n
Ecology	y	y	s
Evolution	y	y	s
Dynamic Earth Processes	y	y	y
Gas Laws	y	y	y
Diffusion and Osmosis	y	y	S
Chemical Reactions	y	y	s
Atomic Interactions	y	y	s
Energy in the Earth system	y	y	s
	STELLA	**STARLOGO**	**MODELIT**
<u>**MATH**</u>	able?	able?	able?
MIDDLE SCHOOL			
Rates and Proportions	Y	p	y
Dependent and Independent Events	Y	y	Y
Models to explain mathematical reasoning	s	y	n
Making and testing conjectures using inductive and deductive reasoning	n	Y	n
HIGH SCHOOL			
Graphing functions	p	p	n
Plotting and interpreting graphs	y	p	y
Asymptotic behavior	n	p	n
Patterns and sequences	n	y	n
Exponential growth	s	y	s
Estimation	y	p	Y

Table 3: Programs and Ability to Teach Certain Lessons

The table shows that there are many similarities between Stella and ModelIt. Both are similar styles of program, but differ in their complexity. Stella can potentially do more, but ModelIt can do things more easily. Sometimes, the easier ModelIt turns out to be better for both teacher and students. Other times, when exact numbers are needed, Stella is a more useful tool. For teachers and students not using complex equations and formulas, ModelIt is a much better choice.

StarLogo can do almost everything on the list of topics, in part because the MIT report was centered on StarLogo, but it sometimes requires creative use of the code. StarLogo works much like a language, capable of doing most things if a user takes the time and has the skills to code them. This fact is why there are so many ‘p’ ratings under StarLogo, or “capable only by using the workings of the program in a unique way that is not part of the original intent of the program”. Even though StarLogo’s primary intent is to create decentralized systems, many systems that are not defined as decentralized can be modeled and simulated with decentralized systems. Supporting such varied operations, StarLogo is most like a language, both in complexity and potential.

Table 3 above is split into two sections. The distinction between those two sections is very important. None of these programs are made to run purely mathematical lessons. For this reason, most of the math lessons are only somewhat doable in the programs. The only math lesson supported by ModelIt and Stella is interpreting graphs. StarLogo, while not a math program, did much better, but mostly through creative use of its language power. A conclusion easily realized is that none of these programs should really be used for purely mathematical lessons, except perhaps StarLogo in very rare circumstances requiring creative programming. ModelIt has no use of numbers, and the only access to mathematical functions in Stella is in relationship curves.

A more interesting topic, then, becomes the top section of the table. Obviously, it is noted that StarLogo can be used for almost all experiment types in Science. This still does not rule out Stella and ModelIt. As is seen in Table 4, Stella and ModelIt are much simpler programs for the teacher, and while most scientific systems are decentralized, Stella and ModelIt show relationships between different variables as expressions and numbers, instead of conditional statements in steps of a turtle's life. It is very difficult to extract the numeric equations (or model the system accurately based upon numeric equations) in StarLogo.

For example, a StarLogo project of animals and food may include a growth rate created by a certain probability that an animal would have a child in each time interval, and certain percent of food reproduction. Death rate, too, would be modeled by the ability of each particular individual animal to reach food. In a Stella model for animals and food, growth rate and death rate are equations based upon current population and food, and can be mathematically created and solved. Nevertheless, properly setup, both programs could accurately teach the population dynamic, showing the same lesson from different angles.

2.2 – Program Difficulty

The simulation programs researched were wildly different in skills required to use them proficiently (See Table 4). On the surface, ModelIt was simplest, Stella moderate, and StarLogo the most difficult. Looking deeper, though, it can be seen that each program requires particular independent skills. Admittedly, Stella requires almost all the skills ModelIt does, and more, since they are based on similar modeling concepts with Stella being more complex.

It can be noted that the required skills for Stella and StarLogo showed Stella having nothing more difficult than StarLogo. Our initial supposition, that StarLogo really is more difficult than Stella which in turn is more difficult than ModelIt, is true.

Table 4 shows a rating of how much skill and knowledge is required to run each program. I obtain the rating by running through the programs with each skill and knowledge statement in mind, noting how important each of the knowledge and skills is. While the uses of the programs differ, none of the programs was particularly more difficult in any one of the knowledge statements.

Key
1 = this knowledge does not apply to using this program 2 = this knowledge is of little real help to using this program 3 = this knowledge is as useful to this program as to most programs 4 = this knowledge is vitally important to using the program 5 = it is impossible to make any use of the program without this knowledge

	Programs		
KNOWLEDGE	**STELLA**	**STARLOGO**	**MODELIT**
Computer Literate	2	5	2
Problem Solving with external support.	2	4	2
Capable of deciding which software to use for a given lesson	4	4	4
Capable of designing lessons to be used with specific software tools	4	5	2
Able to use technology to a business level and translate to an educational one	4	4	2
Design interactive lessons in a program	4	5	2
Design lessons specifically used in the high-tech venue	2	4	2
Able to gauge enhanced achievement through use of technology	2	2	2
Ability to install software without support	3	4	3
Keep up to date on advancements related to the tool	4	5	2
Able to research online for lesson plans	3	5	2
Experienced with a programming language	2	4	1
Experienced with object oriented design	4	5	3
Total Difficulty	40	56	29
Average Difficulty (out of 10, with hardest being 10)	7.1	10	5.2

Table 4: Programs and Required Skills

Since we came into this table already assuming a program-difficulty hierarchy, it would be easy to pass this table off as obvious. There is, though, some important information gleaned. Not only does the table have a compiled list of knowledge requirements, it shows that some requirements are not strictly more difficult in the more "difficult" programs. For example, even though Stella is more difficult than ModelIt in general, both can be used reasonably with about the same amount of computer literacy. Stella also shares with StarLogo a large amount of knowledge to translate business or science knowledge to lessons in a classroom. Both require a teacher to be able to translate knowledge on a complex and specific level.

It is reasonable to assume that teachers could rate themselves on how well they have achieved the knowledge in Table 4, and therefore decide how capable they would be at using that program. Each of the three programs has aspects unique to it unrelated to the set of standards, and each program requires the teacher to look at the lesson to be taught from a different perspective. For example, mimicking graphs in StarLogo requires a teacher to be able to recognize the graph as a set of dots, with each dot calculating its Y position from its X position. Nevertheless, any teacher meeting all of the requirements should, with minimal assistance, be able to use the programs for lessons in math and science. However, the skills and knowledge required might be different from that required to teach the lesson the traditional way.

2.3 – Survey

The plan for this study included a survey. However, because the study was conducted in the summer and very few responses were attained, and none of those responses were by teachers who had experience with all three programs, it was not feasible to include the results of the survey in this report.

The survey, seen in Appendix 1, was hosted on University of Virginia's SurveySuite for four weeks spanning July and August. We announced the survey on the StarLogo teacher mailing lists of 1999 to 2001.

2.4 – Summary

2.4.1 – What Was Learned

Through the research, several things were learned, and several assumptions were proved in numbers. Firstly, it was realized that it is possible, but not practical, to use the researched programs in math lessons. Second, each program's ability to be used for a lesson can be selected, taking the best programs to solve the lesson, and choosing the simplest among them. Programs are picked for Table 5 entirely on the easiest program that can solve the given problem effectively.

SCIENCE	*Simplest Program*
MIDDLE SCHOOL	
Resources In Ecosystems	ModelIt
Population Dynamics	ModelIt
The Food Web	Stella
Chemistry of Living Systems	StarLogo
HIGH SCHOOL	
Genetics	StarLogo
Ecology	Stella
Evolution	Stella
Dynamic Earth Processes	ModelIt
Gas Laws	ModelIt
Diffusion and Osmosis	Stella
Chemical Reactions	Stella
Atomic Interactions	Stella
Energy in the Earth system	Stella

MATH	
MIDDLE SCHOOL	
Rates and Proportions	Stella
Dependent and Independent Events	ModelIt
Models to explain mathematical reasoning	StarLogo
Making and testing conjectures using inductive and deductive reasoning	StarLogo
HIGH SCHOOL	
Graphing functions	Stella
Plotting and interpreting graphs	ModelIt
Asymptotic behavior	StarLogo
Patterns and sequences	StarLogo
Exponential growth	StarLogo
Estimation	ModelIt

Table 5: "Best Simplest" Program for Implementing Each Lesson

This above table lists the "simplest" program that can effectively be used for each lesson. It does not take into account that one program may in fact be better for the particular lesson based upon its features. Realistically speaking, StarLogo would likely be best for every lesson, but since StarLogo is the most "difficult" of the programs, it only appears when it is the only program that can sufficiently show the lesson at all.

The table does show that one can structure a set of lessons that can integrate several science and math concepts. Additionally, more complex programs can completely replace the simpler ones in some cases, especially using StarLogo to allow for more graphical examples, as long as the teacher is sufficiently capable of creating lessons with the harder program.

2.4.2 – Future Work Suggested

This completed project leaves several openings for future work. First, the information found in this project could be used to create lesson plans for one or more subjects. Second, the program suggestion information could be used to show which lessons can effectively be taught by modeling the math and science topics as specific Complex Adaptive Systems. Finally, the validity of the difficulty levels used could be verified or altered with additional surveys.

First, the information in the project, especially the list in Table 5, could be very valuable in designing lesson plans with specific modeling and simulation lessons. A given course's requirements tend to fulfill several of the lesson topics in Table 5. Especially in science courses, those lessons could be created in several models, and used by students to understand as well as to give hands-on learning. For example, a life-science course would include at least: "Food Web", "Population Dynamics", and "Resources". A teacher or researcher could look at the program suggested for each topic, and build a lab experiment for each program by finding a way to put important knowledge standards into a lab.

Second, MIT's project statement mentions that instead of breaking each topic into a unit, a teacher could "combine several interrelated standards into project-based and inquiry-based activities so that students can understand the dynamic and adaptive nature of the phenomena." [1] Effectively, several required lessons could be made into single project units. The project statement gives an example that "...one project might be for students to build their own models of an adaptive ecosystem, combining principles from genetics, ecology, evolution, diseases and earth science." [1] Perhaps all of the lessons can in some way be combined to create a set of projects that cross over each standard topics one or more times, creating realistic projects that teach all important lessons within the subject.

Finally, it would be best to conduct the survey again at a better time and with a larger number of participants. A survey with sufficient responses, and with responses from people who have used all three programs, would validate the chosen numbers in Tables 3 and 4, and if necessary, change them. In order to truly tell how many teachers can use the programs, it is necessary to find out:

1. What the average teacher actually knows,
2. Whether there are any pieces of knowledge missing that are important, and
3. Whether any of this knowledge needs to be weighted as more or less important in deciding the difficulty.

With surveys and research used to study these three items, the difficulty levels could be more effectively numbered. From these numbers, educators can develop better training and education material to "train-the-teachers". The difficulty numbers can also be used by the modeling and simulation software developer to improve the usability of their programs.

Appendices

Appendix 1 – Survey

An on-line survey was conducted to study how teachers view the three software tools. The survey created to gain more information appears below in its entirety. The numbering system in the below survey has been changed for the web-version, to support the workings of the University of Virginia SurveySuite site.

Introduction

Simulation-based software is software whose purpose is to simulate mathematical or scientific systems, usually in some way linking the interactions between objects such as animals and food sources. Such software can potentially be used in a lab environment in schools to teach students concepts that may have aspects above their heads, like population growth dynamics that require calculus knowledge to actually solve completely. The simulation-based software allows most of the difficult math to be buried beneath the surface so the students can learn at their own levels, even on complex systems.

Survey Questions:

1. Which field or fields do you teach? Please be as specific as possible. (Examples: 9-10th grade biology, 12th grade physics, 9th grade algebra, etc)

2. On a scale from 1 to 5, how do you rate your level of computer proficiency? (1 = computer-illiterate,

   3 = average computer user, and

   5 = computer expert)

3. Have you ever contemplated using simulation-based computer software to aid in student learning? Why or why not?

4. Have you ever been to a seminar to learn a how to use simulation-based computer software to aid in student learning? If so, did you make use of the software afterwards? Why or why not?

5. Have you ever used StarLogo?
    a. On a scale from 1 to 10, what would you rate StarLogo, overall?
    b. How do you feel about StarLogo's worth in the classroom?
    c. What do you think is the best use for StarLogo, if any?
    d. How difficult, on a scale from 1 to 10, do you feel it is for a teacher to implement StarLogo in a classroom environment?
    e. Would you recommend StarLogo to other science or math teachers who may not have advanced knowledge of computers?

6. Have you ever used Stella?
    a. On a scale from 1 to 10, what would you rate Stella, overall?
    b. How do you feel about Stella's worth in the classroom?
    c. What do you think is the best use for Stella, if any?
    d. How difficult, on a scale from 1 to 10, do you feel it is for a teacher to implement Stella in a classroom environment?
    e. Would you recommend Stella to other science or math teachers who may not have advanced knowledge of computers?

7. Have you ever used ModelIt?
    f. On a scale from 1 to 10, what would you rate ModelIt, overall?
    g. How do you feel about ModelIt's worth in the classroom?
    h. What do you think is the best use for ModelIt, if any?
    i. How difficult, on a scale from 1 to 10, do you feel it is for a teacher to implement ModelIt in a classroom environment?

j. Would you recommend ModelIt to other science or math teachers who may not have advanced knowledge of computers?

If you have used more than one of the above three programs:

8. Do you find the usability of any of them to be eclipsed by one of the other two?
9. How would you define the differences in usability between the programs you have used?

Bibliography

[1] MIT Project Proposal, Collaboration Regarding Complex Adaptive Systems (unpublished)

This paper is a proposal from MIT to NSF. The purpose of the proposal is to expand the use of - StarLogo in the educational environment. It was the first piece of information that started my project, even though the final project was very different from this proposal.

[2] Investigation Station: Science Laboratory – Software – Model It. University of Michigan, 2000. {http://hi-ce.eecs.umich.edu/sciencelaboratory/modelit/}

This is a college homepage for ModelIt. It contains informative information regarding it, as well as providing a demo of it.

[3] Model-It by GoKnow. GoKnow, LLC, 2002 {http://www.goknow.com/modelit.htm}

This is the official homepage for ModelIt. It contains features and advertisements for ModelIt, aimed at classroom environments.

[4] StarLogo on the Web. Massachusetts Institute of Technology. {http://www.media.mit.edu/starlogo/}

This is the homepage for StarLogo. It contains informative information regarding it, as well as providing a free downloadable copy of it, a web-community, and several sample projects.

[5] High Performance Systems, Inc. “Stella 7.0”. High Performance Systems, {http://www.hps-inc.com/Education/new_Stella.htm}

This is the homepage for Stella. It contains informative information regarding it, as well as providing a demo of it.

[6] ISTE – The Leading Organization for Educational Technology Professionals. {http://www.iste.org}.

ISTE, or the International Society for Technology in Educations, is home to some of the more universally accepted standards. The standards I found, though, were unsuitable for the research I needed, and ended up being unused.

[7] NDE Education Technology Center. {http://www.nde.state.ne.us/TECHCEN/comp/ntct.html}

This is the Nebraska Department of Education’s performance indicator for technology knowledge for teachers, and based on the ISTE standard. It was used to create part of the table defining how difficult each program is for a teacher to use.

[8] Scott County Board of Education. “Scott Co. Technology Assistance Tool “. {http://www.scott.k12.ky.us/technology/techtool.html}

This is the Scott County, Kentucky rubric for technology knowledge teachers. It was used to create part of the table defining how difficult each program is for a teacher to use.