SIMULATION AND ITS DISCONTENTS

SHERRY TURKLE

With additional essays by
William J. Clancey, Stefan Helmreich, Yanni A. Loukissas,
and Natasha Myers

The MIT Press
Cambridge, Massachusetts
London, England

For information about special quantity discounts, please email special_sales@mitpress.mit.edu.

This book was set in Scala and Scala Sans by Graphic Composition, Inc., Bogart, Georgia. Printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Simulation and its discontents / Sherry Turkle ; with additional essays by William J. Clancey . . . [et al.]. ; foreword by John Maeda
p. cm. — (Simplicity—design, technology, business, life)
Includes bibliographical references and index.
ISBN 978-0-262-01270-6 (hbk. : alk. paper)
1. Computer simulation. 2. Visualization. 3. Technology—History. 4. Technology—Social aspects. I. Turkle, Sherry.
QA76.9.C65T87 2009 003′.3—dc22 2008035982

10 9 8 7 6 5 4 3 2 1

CONTENTS

THE VIEW FROM THE 1980s

Winston Churchill once said, "We make our buildings and afterwards they make us. They regulate the course of our lives."[1] We make our technologies and our technologies shape us. They change the way we think. Recently, during a student conference, an MIT professor of architecture spoke of designing his first building, a small lakeside cottage, when he was fresh out of graduate school in the early 1960s: "The connection between drawing and thinking, of sketching until the building was in your blood—that is what made a great design." Later that week, in a workshop on simulation, a senior architect remarked on being taken aback when he realized that his current design students did not know how to draw. They had never learned; for them, drawing was something the computer could do. His response was to insist that all his students take drawing classes. But he had no illusions. This new generation of

students, all fluent users of simulation, is well on its way to leaving behind the power of thinking with a pencil.

From one point of view, the story of an architect who sketches with a computer program rather than with pencil and paper is a story about individual creative experience. It might be about loss or new possibilities. It might be about both. But professional life takes place in public as well as private spheres; the story of the architect and the pencil is about social consequences and aesthetics. Life in simulation has ethical and political as well as artistic dimensions.[2]

Individuals become immersed in the beauty and coherency of simulation; indeed simulations are built to capture us in exactly this way. A thirteen-year-old caught up in *SimCity,* a game which asks its users to play the role of urban developers, told me that among her "Top Ten Rules of *Sim*" was rule number 6: "Raising taxes leads to riots." And she thought that this was not only a rule of the game but a rule in life.[3] What may charm in this story becomes troubling when professionals lose themselves in life on the screen. Professional life requires that one live with the tension of using technology and remembering to distrust it.

PROJECT ATHENA: DESIGN IN THE GARDEN

Project Athena, launched in 1983, brought personal computing to an MIT education.[4] Across all disciplines, faculty were encouraged, equipped, and funded to write educational software to use in their classes. For many, the Athena experience marked a first brush with simulation and visualization technologies.

The experience could be heady. In the School of Architecture and Planning, some faculty saw computing as a window onto new

ways of seeing. To architects accustomed to thinking with pencils in hand, designing on screens suggested novel ways of envisioning space and thinking about the process of design. Students could, as one design professor put it, "go inside programs" to "construct, change, interpret, and understand" how their models worked. Members of the architecture faculty dreamed about students moving from an aerial to a "worm's-eye" view of their sites, of moving from two-dimensional representations into virtual buildings that could come to life.

In the 1980s, this kind of talk was visionary. At that time, faced with slow, clunky equipment, it was not obvious that computers would drive new design epistemologies. A generation had grown up using "batch processing"—a style of computing where you dropped off data at the computer center and picked it up a day later. It was a time when screens were most associated with television, and indeed the same professors who imagined computers changing the basic elements of design also worried that computers might lull designers into the passive habits that came from TV viewing. Interactivity was a word that was only starting to become associated with computers.

The Athena story brings us back to a moment when educators spoke as though they had a choice about using computers in the training of designers and scientists.[5] Most of their students knew better. While his professors were still typing journal articles on typewriters, an MIT architecture student admitted that once he was given access to word processing and a fancy laser printer, not using them made him feel "stupid . . . behind." Another surprised her professors by sharing her idea that computers would soon "be like phones" and part of an architect's daily life.

Students also understood that although the Athena project was cast as an "experiment" and commonly discussed that way on campus, it was never really an experiment. One architecture student said that when she asked her professor why no one was trying to compare how structures designed "manually" actually "stacked up" against structures designed on the screen, she just got a shrug. As she put it: "It was just assumed that the computer would win." From the time it was introduced, simulation was taken as the way of the future.

In the School of Architecture and Planning it was clear from the early 1980s that graduates needed fluency with computer-aided design to compete for jobs. Athena's resources would make it possible to build a computer resource laboratory to meet their needs. This laboratory was called the "Garden," a name that celebrated and was celebrated by its inclusion of ceiling-high ficus trees.

A style of mutual support flourished under the ficus. Those students known to be experts were sought out by those who needed help. Everyone took turns; courtesy was cultivated; students came to know each other's projects. The Garden, open twenty-four hours a day, was one of the most heavily used computer facilities on campus. Even the early computer-aided design programs of the mid-1980s made it possible to move rapidly through a series of design alternatives and tinker with form, shape, and volume. Software presented "the defaults," predrawn architectural elements to manipulate. Faculty were amazed at how students, using them, came up with plans that were novel even to experienced designers.

Design students spoke glowingly of how easy it was to do multiple versions of any one design. Multiple passes meant that mistakes generated less anxiety. Mistakes no longer meant going back to the

drawing board. Now mistakes simply needed to be "debugged" in simulation. In debugging, errors are seen not as false but as fixable. This is a state of mind that makes it easy to learn from mistakes.[6] Multiple passes also brought a new feel for the complexity of design decisions. Civil engineers as well as architects said that simulation let them go beyond simple classroom exercises to problems that gave them a sense of "how an actual building behaves under many conditions." For some, computer-aided design made theory come alive. One student observed that when he studied engineering principles in class, "I knew them to be facts, but I didn't know them to be actual and true." But when he saw the principles at work in virtual structures, he experienced them in a more immediate way. He expressed a paradox that will become familiar: the virtual makes some things seem more real.

But there were troubles in Eden. Most faculty in Architecture and Planning did not like to teach the details of software and programming in their classrooms. While this was in line with an MIT tradition of expecting students to master the technology required for coursework on their own,[7] it also reflected faculty ambivalence about bringing computers into the design curriculum in the first place. Many took the position that computer-aided design was only acceptable if practitioners were steeped in its shortcomings. Faculty referred to this as having a "critical stance." One professor didn't believe that students were in a position to achieve this balance because learning how to use simulation demanded a suspension of disbelief: "They can't be doubting and doing at the same time." And indeed, most students were overwhelmed by how hard it was to use the new technology; getting the details right took their total concentration. In general, students felt that if the faculty wanted them

to have a critical stance toward simulation, then the faculty should teach it to them.

What emerged in Architecture and Planning was a division of labor. The students took up the doing; the faculty took up the doubting. For students, computer literacy became synonymous with technical mastery; for faculty, it meant using simulation even as one appreciated its limitations.

"OWNING" DESIGN

From the beginning of Athena, architecture faculty noted that computer fluency had its price, for example, design programs seemed to draw some students into sloppy, unconsidered work—making changes for change's sake. For their part, students admitted feeling captivated by software, at times so much so that they felt "controlled" by it. For one: "It is easy to let the simulation manipulate you instead of the other way around." Another student felt that the computer's suite of "defaults" constrained her imagination. When she drew designs by hand, she had a set of favorite solutions that she was likely to "plug in." Yet using preset computer defaults seemed different from these. For one thing, she was not their author. For another, they were too easy to implement. She said: "To get to the default you just have to hit return. It makes me feel too secure. If you use the default, it is as though the computer is telling you, 'this is right.'" One student asked of a default setting: "Why would it be in the computer if it wasn't a good solution?" Defaults offered feelings of instant validation that could close down conversation.

A student accustomed to planning her designs with cardboard models felt hemmed in by the computer programs she was forced to use in class. The cardboard had provided an immediate, tactile feedback she enjoyed; designing on the screen felt rigid. In 1984, this student was asked to view the design tools of her "youth" as relics. She was allowed to bring cardboard, plywood, paste, and dowels into her studio classes, but she was discouraged from using them until her designs had been "finalized" in simulation.

Another student summed up his experience of designing in simulation by saying that it had its good points but demanded "constant vigilance": "If you just look at the fine, perfect, neat, and clean sketches that come out of the machine and don't keep changing things . . . going back and forth between hand drawings and the computer, you might forget that there are other ways of connecting to the space."

Ted Randall, a professor of urban planning, lamented, "You love things that are your own marks. In some primitive way, marks are marks. . . . I can lose this piece of paper in the street and if [a day later] I walk on the street and see it, I'll know that I drew it. With a drawing that I do on the computer . . . I might not even know that it's mine. . . . People do analyses of their plan [on the computer] but they only fall in love with the marks they make themselves."

In the 1980s, the final products of computer-aided design programs lacked the artistry and personality of hand-drawn work. Some faculty were so demoralized by the aesthetics of computer printouts that they encouraged students in artisanal "compensations." For example, faculty suggested that students enhance the appearance of their computer projects by using colored pencils to "soften" the

printouts. Softening played an emotional as well as aesthetic role. By making things seem more attractive and handcrafted, softening helped young designers feel more connected to their work. But for Randall, this attempt to bridge drawing and printout only provided ornamentation. What was being lost was the deep connection between hand and design.

Randall was wary of simulation because it encouraged detachment:

> Students can look at the screen and work at it for a while without learning the topography of a site, without really getting it in their head as clearly as they would if they knew it in other ways, through traditional drawing for example. . . . When you draw a site, when you put in the contour lines and the trees, it becomes ingrained in your mind. You come to know the site in a way that is not possible with the computer.

He went on to tell the story of a student who, working on the computer, drew a road on a slope that was too steep to support it. The student had made this mistake because he had left out a contour line in his onscreen drawing, a contour line that represented twenty-five feet on the site. When Randall asked the student to explain what had happened, the student replied, "Well, it's only one contour." Given the screen resolution of the technology in the Garden, it would have been impossible to distinguish the lines on the screen if the student had put in the missing contour line. The student's screen could not accommodate more contour lines and the student had deferred to the technology. Randall commented:

> It was only one contour, but [in physical space] that contour was twenty-five feet. The computer had led to a distortion of the site in the student's mind. He couldn't put more contour lines on the drawing, he said, because it was

"too confusing." He said, "I couldn't work with that many contours. . . . I can't tell the lines apart any more."

In one reading of this vignette about an omitted contour line, it is a story about the limitations of a particular technology. The program was not able to "zoom in" to sufficient resolution; there was not "screen space" for the necessary number of contour lines. But Randall told the story to illustrate a larger concern. He believes that there will always be something in the physical real that will not be represented on the screen. Screen versions of reality will always leave something out, yet screen versions of reality may come to seem like reality itself. We accept them because they are compelling and present themselves as expressions of our most up-to-date tools. We accept them because we have them. They become practical because they are available. So, even when we have reason to doubt that screen realities are true, we are tempted to use them all the same.

Randall saw screens as shifting our attention from the true to what we might think of as the "true-here," the true in the simulation.

To Randall, his students' already noticeable allegiance to the "true-here," their lack of interest in questioning what was on the screen, was facilitated by the opacity of the tools used in computer-aided design. Despite early hopes that students would study programming and become fluent with the underlying structure of their digital tools, most were designing on systems whose inner workings they did not understand. If a tool is opaque to you, verification is not possible; in time, Randall worried, it may come to seem unnecessary. He was not alone in his concerns. One of his colleagues

commented that in simulation, students "got answers without really knowing what's happened." Another felt that students using opaque tools would conceptualize buildings or cities "only in ways that can be embedded in a computer system." Students had no choice but to trust the simulations, which meant they had to trust the programmers who wrote the simulations.

In the early days of Athena, architecture and planning faculty tended to take sides for and against the computer. Critics talked about the danger of architecture becoming "mere engineering" and about the opacity of design programs. Proponents focused on new creative possibilities. In their view, simulation would enable design to be reborn.

Both critics and proponents felt the ground shifting; the fundamentals of design practice were being called into question. It was a thrilling, yet confusing time. And so, it is not surprising that many found themselves in conflict, defending hard-to-reconcile positions, for example, that computers would revolutionize design epistemology *and* that the heart of design would remain in the human domain. Faced with daily conflict, many designers found ways to dismiss the enormity of the change they expected. One strategy was to declare problematic effects inevitable but to relegate them to a far-distant future. Another was to minimize changes through rhetoric: for a while it seemed that the more a designer argued that the computer would bring revolutionary change, the more he or she invoked the idea of the computer as "just a tool." This phrase did a lot of work. Calling the computer "just a tool," even as one asserted that tools shape thought, was a way of saying that a big deal was no big deal.

In the early 1980s, even faculty who were most invested in the future of computer-aided design began to argue that an aspect of

design education should be left untouched by the machine. This was drawing. Design software was being introduced that could serve as an alternative to drawing. Some faculty were pleased that now design professions would not exclude those without a gift for drawing. But others objected. They feared that if young designers relied on the computer, they might never achieve the intimate understanding of a design that comes from tracing it by hand. Designers described drawing as a sacred space, an aspect of design that should be kept inviolate. They pointed out that the practice of drawing, pencil on paper, made it clear that the touchstone for design was close to the body. They made it clear that even as new tools enable new ways of knowing, they also lead to new ways of forgetting.

DESIGN AND THE PROGRAMMER

In the 1980s, alternate visions of computers and the future of design were expressed in competing views about programming. Some architects believed that designers needed to learn advanced programming. If designers did not understand how their tools were constructed, they would not only be dependent on computer experts but less likely to challenge screen realities. Other architects disagreed. They argued that, in the future, creativity would not depend on understanding one's tools but on using them with finesse; the less one got tied up in the technical details of software, the freer one would be to focus exclusively on design. They saw programming and design as at odds; they discussed them as though the technicity of the first would impinge on the artistry of the second. Such views were influential: architecture students peppered

their conversation with phrases such as, "I will never be a computer hacker" and "I don't consider myself a hacker." One graduate student in architecture explained how it was not possible to be both a "computer person" and a good designer at the same time: "Can you be a surgeon and a psychiatrist? I don't think you can. I think you have to make a choice." Another worried about becoming too competent at programming: "I don't want to become so good at it that I'm stuck in front of a computer forty hours a week. It's a matter of selective ignorance."

Civil engineering students were similarly divided on the question of programming. Some made it clear that if they had wanted to become programmers, they would have gone into computer science. They were happy that simulation tools, even if opaque, made it possible for them to attack complex problems from early on in their careers. But civil engineering students and faculty, like their colleagues in architecture, were also concerned about their growing dependency on the computer scientists who built their tools. A design program called Growltiger became a lightning rod for these concerns.

From 1986 on, Growltiger made it possible for civil engineers to create a visual representation of a structure, make a local modification to it, and explore the consequences of that move for the behavior of the structure as a whole. Students appreciated Growltiger because it released them from the burden of tedious calculation. They used it for theory testing, intuition building, and playful exploration. But, as when architects considered computer-aided design, enthusiasm for what this opaque program offered was tempered by anxieties about what it might take away. In particular,

there was fear that ready-made software would blind engineers to crucial sources of error and uncertainty. Just as drawing became a sacred space for many MIT architects, something they wanted to maintain as a "simulation-free zone," so, too, civil engineers frequently talked about the analysis of structures as off-limits for the computer.

Civil engineering faculty shared design lore with their students by telling stories. In the mid-1980s, many of these stories had a common moral: the danger of using computers in structural analysis. One professor described a set of projects where young engineers used complex programs "not only without judgment but sometimes also without even analyzing the program's possibilities," producing "results [that] really are garbage." In a preemptive move, he insisted his students learn to program, but his effort had unhappy results. Those students who saw themselves primarily as designers thought programming was a waste of time. Others became so involved with programming that they drifted away from engineering. They spent more time debugging software than working on design.

This is a complex vignette. It points toward a future that would link the destinies of engineers and designers, even as many designers would be content to navigate the surface of their software, calling in technical "experts" when something went wrong.

Design is a volatile combination of the aesthetic and the technical. In the early 1980s, design students felt pressured to learn more about computers, but worried that this new mastery would turn their professional identities away from art and toward engineering. They needed to be computer fluent; they wanted to keep their distance. One way to accommodate the competing claims of fluency

and distance was to form designer/hacker couples that divided the artistic and technical labor. The hacker members of these couples saw themselves as the midwives of the design future. The designers felt lucky to have found a strategy that allowed them to exploit simulation at a distance. These new collaborations prefigured how an architect such as Frank Gehry could later note that his designs were dependent on the most complex simulations, but that he himself did not get close to the computer.[8] More generally, they prefigured what would become commonplace in design offices—a senior designer, a principal of the firm, working on plans with a computationally sophisticated colleague at his or her side.

The simulations of the 1980s were fragile: programs crashed, data were lost. It took a lot of time to get anything done. These technical troubles obscured the reality that month-by-month, year-by-year, design software was overcoming its doubters. With some resistance, students were learning to live simulation as a new reality. It was a reality that subverted traditional hierarchies. As one civil engineering professor put it, in the new world order of design, the freshmen and sophomores were the "wizards" who could "run circles around the older kids." The faculty, once brilliant and revered, were "way behind, out of this." Faculty would have to "climb on board as best they can," once the field had been "turned around."

In classes, it was often graduate student teaching assistants, not faculty, who pressed students to learn the new technology, or as one undergraduate said, "If she [the graduate student teaching assistant] hadn't pushed, I don't know that we would have used it." Some teaching assistants became well known for their vigilance in pointing out bugs in the technical systems. One famously reminded

undergraduates that "in industry you really need to use hand calculations to check the computer results."

THE SURPRISE OF STYLES

One of the hallmarks of the Athena project was that faculty were asked to build their own educational software. Most worked with a set of assumptions about learning: students begin by learning "fundamental concepts"; formal, mathematical representation is always the best approach; students would use simulations the way designers intended. As Athena unfolded, none of these assumptions proved true. To begin with, students approached simulation with a wide range of personal intellectual styles. Some used the highly organized style of the top-down planner, what many think of as the canonical engineer's style. These students spoke about the importance of beginning any project with the big picture. One student, working with AutoCAD, a computer-aided design program, explained how he used it. He said that he needed to start with "a diagram that gives you direction, a framework for your decisions. Then you can zoom in onto the details, work on them to reinforce the bigger idea, which you always have to keep in mind."

But other students found that, contrary to faculty expectations, AutoCAD in architecture and Growltiger in civil engineering were good environments for working in a bottom-up style, with little initial "framing." From their point of view, the programs let them take a design element and play with it, letting one idea lead them to the next. In simulation, design did not require precise plans; solutions could be "sculpted."

A graduate student in architecture was surprised that she could use the computer to work in this style, one I have called "soft mastery" or "tinkering."[9] When working this way, she approached design as an exercise "where I randomly . . . digitize, move, copy, erase the elements—columns, walls, and levels—without thinking of it as a building, but rather as a sculpture . . . and then take a fragment and work on it in more detail." This student saw design as a conversation rather than a monologue. She took the simulation as a design partner, an "other" that helped her shape ideas. She enjoyed the fact that the program seemed to "push back": "It pushed you to play," she said. In the same spirit, another student remarked that when he used the computer to sketch, he saw himself as a passive observer of spontaneous developments. This sense of the simulation as an "other" with a life of its own made it easier for him to edit, to "discover the structure within."

But even as simulation offered the opportunity to tinker and play, it could also encourage premature closure. As one student put it, "Even if you are just starting out, when you are faced with a computer printout, it is easy to feel as though the design has already been completed." Another described being "mesmerized by the hard-line quality and definition of the output and not want[ing] to change anything. And even if you do [make] changes, you do it in chunks, rather than making little moves, fine tuning, the kind of thing that comes from overlaying the tracing paper on an existing drawing and working on it." While some emphasized how simulation facilitated flexibility, this student focused on the deliberateness of every gesture in simulation. At least for her, "Everything you do is a very definite thing. You can't partially erase things, smudge

them, make them imprecise." From the beginning it was dramatic: simulation offered different things to different people.

In the mid-1980s, what were designers thinking about when they were thinking about simulation? They were thinking about programming and design, doubting and doing, about attachment to the real through tracing with the hand. Having discovered the pleasures of tinkering with simulations, they wondered how wrong their first assumptions about computers in design might be. History had presented computers to them as instruments for "top-down" calculations; now they appeared as facilitators of "bottom-up" investigations. Designers had anticipated that computers would speed things up, freeing them from tedious calculations. What they had not fully anticipated were new ways of seeing sites and structures as they stepped through the looking glass.

INSECURITY

For architects, drawing was the sacred space that needed to be protected from the computer. The transparency of drawing gave designers confidence that they could retrace their steps and defend their decisions. In the 1980s, apart from all aesthetic considerations, the prospect of having to give up drawing made designers feel insecure. What of scientists? As with designers, scientists' insecurity about simulation was tied to concerns about opacity.

At MIT, chemists and physicists complained that when students used computers they entered data and did calculations without understanding their meaning. One chemistry professor said that even the most basic data-analysis programs led students to "simply

enter a set of numbers, getting out an analysis, a number for which they had no clue if it was reasonable or not. They didn't even know whether they were working with a straight line or a curve." Most troubling, his students didn't seem to care. In response, he frantically rewrote computer code so that students would have to specify all analytic steps for any program they used. But even as this chemist worked to reintroduce transparency in software tools, his department was moving in the other direction. It began to use a computer program called Peakfinder that automatically analyzed the molecular structure of compounds, something that students had previously explored through painstaking hours at the spectrometer.

Peakfinder was a time saver, but using it meant that analytic processes once transparent were now *black boxed,* the term engineers use to describe something that is no longer open to understanding. When students used Peakfinder, they didn't know how the program worked but simply read results from a screen. One resentful student summed up the Peakfinder experience by saying, "A monkey could do this." Another likened it to a cookbook, "I simply follow the recipes without thinking."

Most of the chemistry faculty divided the work of science into the routine and the exciting and there was consensus that Peakfinder belonged to the humdrum. One said: "You see the same thing that you would on graph paper, but you're able to manipulate it more easily." From this perspective, simulation simply presented old wine in new bottles. But students could see what many faculty did not: computational speed and accuracy were quantitative changes that had a dramatic qualitative effect. With Peakfinder, the science on the screen began to feel more compelling than any representations that had come before.

So even as Peakfinder took chemistry students away from what many MIT scientists called "the messiness of the real," students said that being able to manipulate data on the screen made working with computer-mediated molecules feel more hands-on than working with "wet ones." Software made it easier to see patterns in data, to feel closer to what some students referred to as "the real science." They said that not having to worry about the mechanics of analyzing compounds with graph paper and plastic printer's rulers left more time for "thinking about fundamentals." Students noted that Peakfinder opened up chemistry to visual intuition. For one chemistry undergraduate: "There was the understanding that comes when you see things actually happen. The lines on the spectral graph were like seeing the molecule moving." For another, "We've always been told how molecules are moving, but it was the first time we actually saw what happens." These students' use of the word *actually* is telling. From the earliest days of Athena we have seen the paradox that simulation often made people feel most in touch with the real.

All science students have limited laboratory time and this makes it hard to collect good data. Before the computer entered the chemistry laboratory, if a student's one run of an experiment yielded only anomalous data, the student could not bring the result of an experiment into relationship with theory. To make this connection, the student was then given prepackaged data, data "certified" as correct. But Peakfinder saved so much laboratory time that students could afford many runs of their experiments. They were no longer so dependent on pre-packaged data. So, in addition to providing compelling visuals that illuminated theory, Peakfinder made students more likely to be collecting their own data, giving them a feeling of ownership over their experiments.

In physics, too, where computers were used to relieve the tedium of data collection and plotting, relatively mundane applications had significant effects. When calculation was automated and its results instantaneously translated into screen visualizations, patterns in data became more apparent. Physics students described feeling "closer to science" and "closer to theory" when their laboratory classes began to use software for visualization and analysis. As in chemistry, messy data no longer spoiled an experiment because you could afford the time to make extra passes until you had a good data set. One student was sure that doing all of these extra passes was what got her "hooked on physics" because "seeing that the data fits in spite of the variation is part of the allure of physics." Another physics student reflected, "If you do the same things 1,000 times by hand, you lose the sense of what you're doing. It takes so long, you forget the goal. You lose the forest in the trees." Another commented that for him "theory came alive" when a simulation demonstrated time and voltage in relation to each another. "You're not just seeing the curve drawn, you're seeing it actually happen . . . created point by point."

These students were experiencing another of computation's "paradoxical effects." Computers, usually associated with precision and rules, brought students closer to what is messy and irregular about nature. Computers made it possible for students to confront anomalous data with confidence. With multiple iterations, patterns became clear. As one student put it, "irregularities could be embraced."

Yet even with such benefits, and these benefits were substantial, MIT physics and chemistry faculty fretted that computer visualization put their students at an unacceptable remove from the real.

When students claimed to be "seeing it *actually* happen" on a screen, their teachers were upset by how a representation had taken on unjustified authority. Faculty began conversations by acknowledging that in any experiment, one only sees nature through an apparatus, but here, there were additional dangers: the users of this apparatus did not understand its inner workings and indeed, visualization software was designed to give the impression that it offered a direct window onto nature.

When chemistry students talked about feeling close to the "real molecule," faculty were at pains to point out the many, many levels of programming that stood between students and what they were taking as "reality." The most seductive visualizations could be described as a proverbial house of cards. For example, in chemical simulations, molecules were built from computer code that students never examined. Simulations of chemical reactions were built on top of these simulations, which were built on top of simulations of atomic particles. Professors tried to respect student enthusiasms while objecting to any uncritical immersion in the screen world. Many faculty began to argue the strategic importance of there being a place where they felt more in control of their students' development: this was the lecture hall. As drawing was for designers, as the analysis of structures was for civil engineers, the lecture hall became a sacred space for chemists.

In the lecture hall, chemists were confident that they could communicate what they considered the nonnegotiables. These included both things to know and how to know them. In their view, a computer could never teach theory, no matter what its usefulness in the laboratory. And a computer could never teach how science and the body are enmeshed. In the lecture hall, alone with their

students, chemists felt they could do these things. In elaborate demonstrations, playing with models, turning and twisting their limbs to dramatize molecular reactions, chemists showed students how to think aloud and how to think with the body. And, in the lecture hall, chemists felt sure that they could show students how to put first things first. It was a place where faculty presented themselves as curators of their fields. There was similar sentiment among physicists. Lectures were a place where one could teach students how to be scientists. Although mentorship happened in laboratories, faculty acknowledged that there they would increasingly be competing with the seduction of screens. In lectures, one could hope for students' undivided attention.[10]

REVERENCE FOR THE REAL

At MIT, physicists became voices of resistance to Athena. When asked to be part of the project, their department several times declined. At the heart of its objection was the feeling that simulation stood in the way of the most direct experience of nature. Physics faculty acknowledged that some students found computers helpful for seeing patterns in the laboratory but then went on to speak reverently of the power of direct experience in their own introductions to science, or as one put it, "of learning Newton's laws by playing baseball." "Simulations," he said, "are not the real world. . . . There is no substitute for knowing what a kilogram feels like or knowing what a centimeter is or a one-meter beach ball. But these things have to be taught to students, and we faculty should teach it to them." For physicists, using simulation when you could be in direct touch with the physical world was close to blasphemy.

Physics faculty were concerned that students who understood the theoretical difference between representation and reality lost that clarity when faced with compelling screen graphics. The possibility for such blurring existed even when students used simple programs that visualized data. Indeed, the faculty who introduced the first generation of such programs were shocked at how little it took for students to get lost in what would come to be known as "the zone." It was a time when a simple computer game such as *Space Invaders* mesmerized millions. As simulations became more complex, faculty anxiety grew. For one physicist:

> My students know more and more about computer reality, but less and less about the real world. And they no longer even really know about computer reality because the simulations have become so complex that people don't build them any more. They just buy them and can't get beneath the surface. If the assumptions behind some simulation were flawed, my students wouldn't even know where or how to look for the problem. So I'm afraid that where we are going here is towards *Physics: The Movie*.

In physics, student opinion mirrored faculty concerns. For one physics student, "Using computers as a black box isn't right. For scientists who are interested in understanding phenomena and theorizing, it's important to know what the program is doing. You can't just use it [a program] to measure everything." Another student admitted that it was easy to "be mindless on the computer. You put worthless data in and get worthless results out, but think it's noble somehow just because it has been through the machine." Although some students spoke about how automatic data collection enabled them to see new patterns, others extolled the virtues of taking data by hand, even though this might mean getting only a fraction of the data points. For one, "Doing it by hand forces you to think about

the data as it is being taken. You have to know what you are looking for." In his view, working through a long series of hand calculations put him in a position to keep track of the kinds of errors he made. His dedication to the aesthetic of transparency was total: "When you work on the computer it is hard to tell what data is 'lose-able' and what is essential. The computer makes these choices for you. That's why I do it by hand as much as I can. Doing it by hand, point by point, you can see subtle variations in the phenomenon you are studying." Another student described "chart[ing] tables by hand" as providing "an intuitive sense of an experiment." One "live[s] with these numbers so much that you begin to understand what they mean. It's nice to be able to enjoy your graph and take your ruler to it. With the computer, it's a mental process further removed from reality. The problem is in getting the feeling of moving the ruler on the paper with the computer."

Just as this physics student spoke about the pleasures of ruler and paper, another admitted that even though doing things by hand was "drudgery," it was useful drudgery:

> There is some truth to the importance of getting a feel for the data by hand. When you plot it with the computer, you just see it go "ZUK" and there it is. You have to look at it and think about it for a long time before you know what it all means. Whereas when you draw it, you live with it and you think about it as you go along.

No member of the MIT physics faculty expressed greater commitment to transparency and the direct experience of nature than William Malven. Of direct experience he said, "I like physical objects that I touch, smell, bite into. The idea of making a simulation . . . excuse me, but that's like masturbation." On the need for transparency, Malven was a purist. Ideally, for Malven, every

piece of equipment that a student finds in a laboratory "should be simple enough . . . [to] open . . . and see what's inside." In his view, students should be able to design and build their own equipment. If they don't do so for practical reasons, they should feel that they could have. Malven saw his role as a teacher and a scientist as "fighting against the black box." He admitted that this required a continual "rear-guard action" because, as he put it, "as techniques become established, they naturally become black boxes." But, he added, "it's worth fighting at every stage, because wherever you are in the process there is a lot to be learned."

Yet even as Malven understood simulation as a potentially threatening black box, he was dedicated to the idea that scientists could create visualization programs that would improve scientific training. The key was for scientists to take control of their own pedagogy. For example, Malven envisaged software that would teach transparent understanding by requiring researchers to make explicit the procedures they were demanding of the computer. To this end, in the early 1980s Malven designed a laboratory course for all juniors majoring in physics. He wrote all of the course software himself. All of his programs, designed to help with data collection and analysis, had their inner workings transparent to the user.

Malven's junior laboratory began with a classic experiment that used computers to increase students' sensitivity to experimental error. When a source of electrons is placed first in a magnetic and then in an electric field, it is possible to determine its specific charge. Malven set up this experiment using a computer program that would not accept a data point without a student specifying an error factor. Malven described this as an effort to use simulation to bring students closer to "the real data."

A second experiment, known as the Stern-Gerlach experiment, used computers to similar effect.[11] Here, a beam of silver atoms is passed through a magnetic field and then onto a film plate. The atoms form smudges from which their orientation can be determined. Originally in a mixed state, the silver atoms collapse into one of two orientations—as quantum physics would predict—once they enter the magnetic field.

Before the computer was used to collect data in the Stern-Gerlach experiment, students had to manually move magnets, read a meter, and take measurements with a painfully slow pen-recording device. Malven said that in those precomputer days, "you could never quite figure out what was going on because it would take fifteen minutes to do any one thing." But Malven devised a new way of doing the experiment: he attached a computer to an apparatus that collected data from the film smear and he wrote software that did the calculations necessary to determine the emergent shape. Because of the computer's speed, two peaks appeared almost instantaneously, indicating the two magnetic moments. For Malven, the dramatic appearance of the two peaks made the experiment come alive, an example of "something profound and something mundane—that combination, it's like love! The mundane part: something as simple as remembering to degauss the magnet in the experiment. The profound part: the space quantization. That the mundane and profound go together—like washing dishes and love—it's one of those things that is hard to learn."[12]

Malven said that using the computer brought his students closer to science because they could play with data and tinker with variables. Like Peakfinder in chemistry, Growltiger in civil engineering, and computer-aided design programs in architecture, the software

in the junior physics lab encouraged students, in Malven's words, to "ask a question and say, 'Let's try it.'"

The instrument that Malven used most in the laboratory was his Swiss Army knife, a simple, all-purpose tool. He thought that the best computational environments were—like the knife—transparent and general purpose. He was not happy when the physics department purchased some "fancy data analyzers." In his view, only general-purpose computers gave students sufficient opportunity to write their own programs, to learn from "the ground up." For Malven, knowing how to program was crucial to making simulations transparent. Other people wanted to use the most up-to-date tools. Malven was interested in only the most transparent ones.

Malven's involvement won over some of his skeptical colleagues to the idea of using computers in their classes under the banner of transparent understanding. Physics professors Barry Niloff and David Gorham designed an Athena-sponsored freshman seminar that began by having all students learn to program. Then, the seminar turned to computers to engage students in problem solving that used numerical rather than analytical methods. Professor Niloff, summed up the seminar's purpose—to demonstrate the power of "just plotting stuff," using the computer to get students closer to direct observation. Only a small subset of real-world physics problems are solvable by purely analytical methods. Most require experimentation, where you do trials, evaluate forces, and fit data to a curve. The computer makes such numerical solutions easier to do. And in a practical sense, it makes many of them possible for the first time.[13]

Over time, the freshman seminar turned its focus to problems of physical measurement. It was something about which Niloff and

Gorham were passionate. Gorham noted that from the time students began to use calculators, they suffered from an insufficient understanding of scale and of "what it means to make a physical measurement," in particular, what it means to have a margin of error. Students with calculators had gotten lazy; they didn't want to do things by hand, and they didn't want to include units in their calculations. The stakes were high. As Gorham put it, for a physics student, not understanding error during college could translate into a "space shuttle blowing up" after the student entered the workforce.

Gorham believed that computers were making students' problems with scale and error worse but that since calculators and computers were not going away, good pedagogy demanded that students be forced to do "back of the envelope calculations." These calculations require an understanding of the scale you are working in, the units you are using, the number of significant digits that make sense. Gorham saw himself as on a mission: to take a technology that had caused problems and turn it to the good. Computers had played a part in encouraging students to be sloppy about scale and error. But now teachers could use computers to become more effective "proselytizers for statements of error terms."

In one freshman seminar assignment, Niloff and Gorham set up a computer program that simulated a ball dropping in space. Students could vary the ball's weight and the height from which it was dropped. The program displayed the falling ball, recorded relevant data, and then allowed students to analyze it. Students were asked to consider the influence of wind drag and inertia on the measurements. Gorham said:

> In the ball dropping exercise, the problem was simply stated. Measure the acceleration due to gravity and then state the uncertainty. And this blows

their [the students'] minds totally. People are used to textbooks. They are used to computers. They don't understand that the computer is marvelous to take data and reduce it, but what does it mean to have an error? If someone said, "*g* is 9.81," that's totally meaningless. 9.81 plus or minus what? 9.81 plus or minus 10 is not a very good measurement. And so we try to drill into them the whole idea of error analysis.

Like Malven, Niloff and Gorham believed that students needed a deep knowledge of laboratory software that could only come from knowing how to program it. Niloff went beyond this: students needed to understand how computers worked, down to the physics of the processor and the graphics screen. Understanding these deeper levels would ultimately bring students closer to problems of estimation, scale, and error. Said Niloff, "When students plot points for the first time, they literally understand what the physical screen is, what the graphics screen is, how to actually put points on the screen." A curve drawn on a screen and a theoretical curve might look the same, but a student who understood the screen's resolution might find a difference at "the tenth of a pixel level, which you can't see. . . . And this tenth of a pixel level may be of critical importance. It may be the margin of error that makes all the difference."

SIMULATION AND DEMONSTRATION: SCIENCE AND ENGINEERING

The junior laboratory and freshman seminar were notable exceptions to a general "antisimulation" culture in the physics department. But in one instance, even those faculty most hostile to simulation were willing to accept it as a necessary evil. This was when simulation made the invisible visible—that is, when it provided access to quantum-level phenomena.

In the 1960s, MIT physicists Harold Rabb and Burt Fallon were part of a research group whose focus was innovation in physics education. Rabb and Fallon continued this work and, among other things, collaborated on short films that illustrated principles of relativity. One of these films demonstrated wave packets propagating as a function of time and fragmenting upon collision. It made quantum mechanics come alive. During the 1980s, Fallon used interactive computing to continue his visualization work in the areas of relativity and quantum physics. He wanted to go beyond the passive presentations possible in film to provide an experience of *living* in the quantum world, including the ability to perform experiments within it. Fallon's computer programs were designed to demonstrate the invisible physics, to help students develop intuitions about the quantum world the way they developed intuitions about classical physics: by manipulating its materials.

One of Fallon's programs simulated what it looks like to travel down a road at nearly the speed of light. Shapes are distorted; they twist and writhe. Objects change color and intensity. All of this can be described through the laws of physics, but Fallon pointed out that "you can't experience them directly, except through the computer." When MIT physicists in the 1980s looked at Fallon's work, they saw it as an exception to a cherished rule. They were willing to accept a simulation when no real-world experience could possibly be substituted, but when Fallon used simulation to demonstrate something that could be done in a traditional laboratory setting, one saw the full force of his colleagues' hostility.

The following exchange on simulation between physicist Barry Niloff and his colleague, Arthur Richman, spoke directly to this

issue. Here Niloff and Richman are eloquent about the importance of keeping simulation in its place.

Richman: One of the problems that a physicist has to come to grips with is that sometimes light behaves like a particle and sometimes it behaves like a wave. If you have a dike and two little openings, the waves of water will propagate through those two little openings. They'll form little rings, which will then interfere with one another, and you'll see the results of these two wave fronts coming out, interfering with one another. That's a very clear wave front. If you think about shooting bullets through these two holes, you know the bullet goes through one or through the other. Now, you're being told as a student of quantum mechanics that sometimes you're supposed to think about light in one way and sometimes you're supposed to think about it the other way. And so a very important experiment comes to mind. You take this case of two slits and decrease the level of illumination so you're very sure photons are only going through one at a time. You would be tempted to say, "Well, by gosh, this is going through one at a time. It's like a bullet, it goes through there, or it goes through there." And there's a dramatic demonstration that can be done to show in a way that just hits you over the head in a beautiful way, that even though they're going through one at a time, they are managing to diffuse. It's a fantastic experience for a physicist who is beginning to think about quantum mechanics. And I think many of us have the same reaction, that to simulate that on the computer . . .

Niloff: It's a cheat.

Richman: It's almost sacrilege.

William Malven worried that students, drawn to substituting models for reality, are then tempted to believe that something will happen in the real world because they have seen it in a simulation. Richman and Niloff feared that when their colleague Fallon demonstrated something on the computer that could be shown without it, students would have a feeling of understanding without true

insight. On the most general level, in the 1980s, MIT physics faculty opposed anything that smacked of demonstration rather than experiment. A demonstration takes what we know and shows it in powerful relief. An experiment, in ideal terms, turns to nature ready to be surprised. But if experiments are done "in simulation," then by definition, nature is presumed to be "known in advance," for nature would need to be embedded in the program.

Physicists acknowledged that, in their own way, simulations can surprise. Complexity in simulation can lead to "emergent effects."[14] But in simulated experiments, they insisted, the programmer has always been there before. Said one student, commenting on an experiment presented virtually, "The 'experiment' is prewired," and it was this prewiring that had turned it into a demonstration, or as Richman summed it up, "close to sacrilege." Physicists constructed nature as untamed and unruly. Simulation sorts it out.[15] At MIT the debate about simulation and demonstration was fueled by faculty anxiety that if students performed enough "experiments" in simulation, they would become accustomed to looking for nature in representations they did not fully understand.

Simply put, for MIT physicists in the 1980s, simulation provoked discontents. Physicists used criticism of simulation as a way of asserting core values defined in opposition to it: the importance of transparent understanding, direct experience, and a clear distinction between science and engineering. For them, simulation was dangerously close to demonstration, the stuff of engineering education. They conceded that engineers might be satisfied with simulation, but as scientists they were bound to resist it as something that might taint scientific culture with engineering values. And indeed, physics faculty felt that they could already see signs

that simulation was eroding critical sensibilities. For one thing, when students found something on the computer, they tended to assume that someone in authority had thought it was correct. For another, students who had grown up with video games experienced screen objects as, if not real, then real enough. This was a generation more likely than their elders to take the screen world "at interface value."[16]

If physicists worried that the computer was a Trojan horse that might introduce engineering values into the scientific enterprise, they were also concerned that it would make physicists too dependent on engineers. One student put it bluntly: "Of course you can't know everything about the computer—as a computer science major might—but you should know enough that you don't have to hand your work over to a Course 6 [Electrical Engineering and Computer Science] guy."

Another physics student said he respected engineers but drew a sharp distinction between his field and theirs: "A goal of engineering is to create new devices of human significance." In physics, however, "one is dealing with the universe. It elevates you to an eternal sense . . . in physics you're relating to absolute truth as opposed to practical truth." In his view, simulation did not help with absolute truth.

Engineers were rather less conflicted. They contented themselves with personalizing software (even if some elements of it were black boxed) by tweaking it in a style MIT students called *customization*. In the 1980s, customization referred to making opaque software do something quixotic, often "against its grain." So, for example, civil engineering students took a program that was supposed to drill them with questions and got it to do their homework for them.

Although some joked that they didn't have a "physicist's" understanding of the simulations they modified, this kind of playful interaction gave a feeling of making technology one's own. Customizing was a strategy, as one civil engineering student described it, "to feel less at a program's mercy than at its command."

Physicists were critical of customization. In their view, there was no true intellectual ownership without full transparent understanding. For them, customization exemplified the kind of compromise that engineers and designers might be willing to make, but that a scientist should never consider.

In the 1980s, scientists were comforted by the idea that building simulations was the province of computer scientists and that they were in a very different field. Not too much time would pass before this division became elusive. More and more, science was done on the computer and a migration of computer scientists to the natural sciences would soon make it hard to say where one left off and the other began.

The View from the 1980s

1. Winston Churchill. Remarks to the English Architectural Association, 1924, available at <http://www.icf-cebe.com/quotes/quotes.html> (accessed October 20, 2008).

2. See Max Weber, "Science as a Vocation" and "Politics as a Vocation," *From Max Weber: Essays in Sociology,* trans., ed., and with an introduction by H. H. Gerth and C. W. Mills (New York: Oxford University Press, 1946).

3. The programmers of *SimCity* could have made raising taxes lead to more social services and increased social harmony—but they didn't. What is important here is that this young woman did not know how to program in her game environment. She did not know how to change the rules of the game, nor did she consider questioning the rules of the game, or asking if these rules applied beyond the game. She did not know how to measure the program against the history of real cities. In my view, citizenship in the twenty-first century calls for readership skills in the culture of simulation, the digital equivalent to knowing the "Who, What, Where, Why, and How?" of print media. Such readership skills enable people to be critical of simulation, not simply immersed in it. And what is true of citizens can be no less true when citizens act as professionals. For more on this example, see Sherry Turkle, "Virtuality and Its Discontents," *The American Prospect,* no. 24 (Winter 1996): 50–57.

4. Athena was launched in 1983 with a $70 million dollar gift from the IBM and Digital Equipment Corporations for the purpose of using computers in the undergraduate curriculum. Athena's first priority was to create a "coherent network" for educational computing, including hardware, operating systems, and programming languages. The goal was to have a student's one-time investment in learning the system provide access to the full range of educational software developed at MIT. Building on this technically sophisticated network, Athena's architects hoped to integrate "modern computer and computational facilities into all phases of the educational process" in order to "help students learn more creatively and fully in a wide range of disciplines" by developing "new conceptual and intuitive understanding

and [improving and refining] our teaching methods." "Project Athena, Faculty/Student Projects," *MIT Bulletin* (March 1985).

5. In our 1983–1987 Athena study, many MIT faculty believed they could avoid having much to do with computers. This study included interviews with students and faculty as well as observation of courses that had introduced computers into the teaching experience. Four fields were intensively studied. Interviews in civil engineering were conducted by Wim Overmeer and Donald Schön, in chemistry by M. Stella Orsini and Brenda Nielsen; in physics and architecture and planning by Brenda Nielsen and Sherry Turkle.

6. See Seymour Papert, *Mindstorms: Computers, Children, and Powerful Ideas* (New York, Basic Books, 1981), 23.

7. MIT tradition favors teaching programming as an "aside" to courses. At the time of the Athena launch, even the main programming course for computer science majors, 6.001, expected students to teach themselves the LISP programming language and focused on larger issues about the structure of programs. This strategy of expecting students to acquire programming skills on their own worked within the Department of Electrical Engineering and Computer Science. It worked less well in the School of Architecture and Planning. There, anxiety ran high when students realized that they were being left to their own devices on technical matters.

8. On Gehry, see Michael Schrage, "Nice Building, but the Real Innovation Is in the Process," *Fortune,* July 10, 2000. Available at <http://money.cnn.com/magazines/fortune/fortune_archive/2000/07/10/283746/index.htm> (accessed July 21, 2008).

9. For a more complete description of styles of mastery and how computers facilitated diverse styles, see Sherry Turkle, *The Second Self: Computers and the Human Spirit* (Cambridge, Mass.: MIT Press, 2005 [1984]), especially chapter 3; and Sherry Turkle and Seymour Papert, "Epistemological Pluralism: Styles and Voices in the Computer Culture," *Signs: Journal of Women in Culture and Society* 16, no. 1 (1990): 128–157.

10. These days, the sanctity of the lecture hall in the 1980s has a distinct irony. Students now enter lectures and flip open laptops that are directly connected to the Internet.

11. As an electron travels around an atomic nucleus, a magnetic field is set up. Classical physics predicts that, if influenced by another magnetic field, the orientation of the electron will be deflected and there should be a continuum of deflections depending on the strength of the external magnetic field and the magnetic momentum of the atom itself. But quantum physics predicts only two positions; the Stern-Gerlach experiment demonstrates this space quantization.

12. A third experiment in Professor Malven's junior laboratory, the Mossbauer experiment, demonstrated resonance fluorescence, a technique used to infer the mean lifetime of the excited state of nuclei. The technique works in a fairly straightforward way for atomic transitions, but a number of problems arise when you try to use it for nuclear transitions: crystal structures magnify the movement of individual nuclei and widen the energy curve, masking its natural width; when nuclei emit a photon they recoil and this motion shifts the spectrum. Rudolf Mossbauer discovered that by reducing the temperature one could eliminate the factors that skewed the curve. And it was later found that iron (^{57}Fe) could give similarly clear results without lowering temperatures. In the junior lab, the nuclei in a sample of iron were raised to an excited state. As photons were emitted and the energy states dropped, the spectrum was measured by a photometer and the computer was used to record its data and fit it to a curve whose width related to the mean lifetime of the excited state.

13. Physics faculty explained that problems that could be solved by analytical methods (and those methods themselves) had defined "high physics," prestigious physics. The computer made it possible to solve classes of problems that were only accessible through numerical methods, what Professor Niloff referred to as "just plotting stuff."

14. On using emergent phenomena in science pedagogy, see Mitchel Resnick, *Termites, Turtles, and Traffic Jams: Explorations in Massively Parallel Computing* (Cambridge, Mass., MIT Press, 1997).

15. Every laboratory, in some sense, creates an enclosed, artificial environment, a "simulation" necessary to perform experiments. Simulation on the computer takes things further: in their attempts at rigorous internal consistency, simulations become increasingly difficult to calibrate against an external world. See Harry Collins, *Changing Order: Replication and Induction in Scientific Practice,* 2nd ed. (Chicago: University of Chicago Press, 1992 [1985]). Collins and others have used the term "experimental regress" to describe how the calibration of a scientific instrument can become entangled within the highly contrived environment of an experimental framework. In the end, there is no outside source against which to tune the machine. See Andrew Pickering, "The Mangle of Practice: Agency and Emergence in the Sociology of Science," *American Journal of Sociology* 99, no. 3 (1993): 559–589.

16. See Sherry Turkle, *Life on the Screen: Identity in the Age of the Internet* (New York: Simon and Schuster, 1995).

Design and Science at the Millennium

1. In the pages that follow, I report on an NSF-funded study of simulation and visualization conducted from 2003-2005. See Sherry Turkle, Joseph Dumit, David Mindell, Hugh Gusterson, Susan Silbey, Yanni A. Loukissas, and Natasha Myers, "Information Technologies and Professional Identity: A Comparative Study of the Effects of Virtuality," in *A Report to the National Science Foundation on Grant No. 0220347* (Cambridge, Mass.: Massachusetts Institute of Technology, 2005). Two pre-doctoral research assistants were integral to that study, Yanni A. Loukissas, working in architecture, and Natasha Myers, working in structural biology.

Citations from practicing architects and architecture faculty and students in the 2000s draw on interviews conducted by Loukissas and Turkle except where I specify that the citations are from designers participating in one