Note: What follows is a slightly longer version of a paper that was presented at the 22nd International Conference on Improving University Learning and Teaching, Rio de Janeiro, Brazil, July 1997. The shorter version of the paper appears in the proceedings of that conference, pages 427 - 444.

THE PERSONAL FONEMATE & OTHER CAUTIONARY TALES

David M. Harrison[†]

UPSCALE: Undergraduate Physics Students'
Computing and Learning Environment

Department of Physics
University of Toronto

ABSTRACT

We have been using information technology in our undergraduate program for over twenty years. Over that span of time, we have had some successes and many failures. In this paper we examine both. The measurement of 'success' and 'failure' is fairly simple: if students use a system when not required it is a success, otherwise it is a failure.

We have had universal failures in attempting to either construct or use other people's self-paced tutorials and similar traditional *Computer Assisted Instruction* or *Computer Based Instruction* tools. We have also had failures in bulletin board systems that allow students to interact with each other as well as with teaching staff. Successes center on delivery of encapsulated information, such as laboratory instrument specifications or the results of fitting a dataset to a line. We have also had great success with delivery of materials over the network which the student can then use on-screen or print themselves.

One of the big pitfalls is that too often the staff developing computer materials devote great resources to an application for little reason other than it is technologically possible to do it. This resonates with pressure from administrators, colleagues and the students themselves for a program to be perceived as 'modern' and 'up to date'.

The common denominator in our successes seems to be applications that do not require large quantities of information and/or the context of that information to be read on a screen.

† Information on contacting us appears towards the end of this paper.

I. INTRODUCTION

The Department of Physics has been providing computational support to our undergraduate program for over twenty years. In that time the breadth of services and usage by our students has grown almost every year. For the 1995-96 academic year, we provided 1720 undergraduate students with over 17,000 hours of computation.[1] Essentially, every student taking an undergraduate physics course uses the facility.

We can divide our user community into two major categories:

1. About 1600 first year students taking their one and only physics course, often to satisfy admission requirements to a professional faculty such as medicine.
2. About 100 upper year physics majors and specialists.

The group of upper year students, despite their relatively small numbers, make much heavier demands on our computational facility than our first year students. It is our work with our first year students that has the broadest applicability to fields other than physics, and we largely concentrate on this group below.

Our efforts through the years have had some successes and many failures. The measurement of ‘success’ and ‘failure’ is fairly simple for most of our applications: a success is a non-required program that the students continue to use long after it is first introduced. We have some ideas on why some projects have succeeded and others have failed, and will be sharing these in this paper.

We do not discuss here our large and growing use of networked PC’s as data acquisition and process control instruments in our upper year laboratories.

Our successes have led to large numbers of hours of computation delivered to our students, with a median connect time of 4.5 hours per student per year for first year students and a median connect time 36.3 hours per student per year for our upper year specialists. This is largely because we have concentrated our efforts into extending our successes and dropping our failures.

1. This is over 17,000 **connect hours** from nearly 42,000 logins.

II. SOME TALES

II.A Self-paced Tutorials

Personal Fonemate allows one to use the World Wide Web to enter a phone number and receive back the tones necessary to dial your telephone. Thus instead of just picking up your phone and dialing the number, you may fire up your web browser, enter the URL for the *Personal Fonemate*, wait for the page to come up, type in the phone number, wait for it to travel over the web, and then hold your phone receiver up to the speaker on your computer.[2]

Although the *Personal Fonemate* may be useless, the process that led to its creation is not: there is a long and honorable history of useless computer programs and games. For example, a still popular text editor under UNIX (*vi*), was based on a game (*rogue*). By writing games and other useless applications, programmers not only have lots of fun but often learn how to use new technology in ways that ultimately prove to be useful. However, what drives such explorations is more often the realisation that one *can* write a particular program rather than whether one *should*.

Similarly, a computer-philic academic can have lots of fun developing self-paced course tutorials using some new and exciting technology such as *Java*. Some years ago we also had a lot of fun implementing such physics tutorials using now-ancient *Tektronix* storage tube technology.

Sadly, after a few explorations the students didn't use our materials. Undoubtedly part of the problem was in our own lack of creativity. However, we are still not aware of any such software using any technological base by anybody that our students have found to be useful in their studies.

The problem can be somewhat more subtle than the fact that the developer becomes blinded by the technological issues and loses sight of the pedagogical ones. For example, Bork [Bork 1981, 1985] developed many tutorials for physics by assembling a group of people to do the design in a hot tub. He hired programmers

2. The URL is: `http://www.ulcc.ac.uk/Sound/dial.html`. The page was inducted into the *Hall of Fame* of *The Useless WWW Pages* as a founder member in February 1995.

to implement the designs, and to a first order approximation the design was independent of the implementation. Nonetheless the results of these efforts were in our view underwhelming.

We will discuss in the conclusion some ideas about the difficulties in this sort of software, based in part on results from the following subsections.

II.B Problem Set Solutions

The heart of traditional physics education is having the students solve problems. Particularly in beginning courses these problem sets are usually collected and marked, largely to require the students to do the problems. After being collected, the solutions to the problems are posted so that students may see how to do the problems that they were unable to solve themselves.

At first glance, problem set solutions would seem to have little applicability to disciplines outside the sciences, engineering and perhaps the social sciences. We shall argue in the conclusion that this is not entirely true.

Our largest first year course has nearly 1000 students, and just before a test the students were crammed a dozen deep in front of the problem set bulletin board trying to see and copy down the solutions. We thought this was fairly inhumane, and began putting the solved problem sets on-line before posting. We now have a bank of five years of problem set, test and examination solutions for this course on-line, and other courses are now using the facility. In the 1995-96 academic year this application was used 22,229 times by 996 different students.

The solutions themselves are hand written, and are simply scanned to produce the on-line version. Given sufficient human resources, it would be much better to have them word-processed; this would make the solutions more readable, indexable, and accessible to students with visual disabilities.

Originally this application used our in-house X-terminals, connected to our central UNIX server. Over the summer of 1996 we converted to a series of World Wide Web pages. This has relieved the pressure on our X-terminals just before tests: about two-thirds of the access to the solutions now occurs from other University locations or from students dialing in from home.

According to a survey, 86% of our first year students used the solutions either from the problem set bulletin board or on-line. 80% of those students used the on-line version for the majority of their accesses. About a third of our first year students who use the on-line version print all or virtually all the solutions for later reference, while about 25% do not print a significant fraction at all.

Independent of format, usage of problem set solutions falls into two broad categories. Some students look at the solutions only after they have tried a problem, and want to check or confirm the solution: this is usually very educational for the student and is always recommended by teaching staff. The second group of students reads the solutions as substitutes for doing the problems themselves: this is seldom educational and is strongly discouraged by teaching staff. Of course, recommendations by teaching staff are routinely ignored by some students, particularly by beginning students.

Regardless of the educational benefit, it should be noted that when a student "reads" a problem set solution, these two categories of student users are doing something quite different. Students who are checking a particular problem with which they are familiar typically are looking at only a part of a solution for a particular problem; these students are also familiar with the overall context of the problem. Students who are looking at solutions for problems they have not actually done are "reading" in a way similar to the way they should be studying the worked examples in their textbook.

We shall be arguing that these two types of reading activity have large implications in terms of the delivery of computer based instructional materials.

So far we have concentrated on our first year students in this subsection. A recent general strike in Toronto meant that it was difficult to distribute a problem set to a much more experienced group of science students: a class of 47 third year engineers. We put the problem set up on our web site *without* solutions as an alternative distribution mechanism. With one exception, all students printed the problem set. When asked whether they would have still printed the problem set if they had a screen on the desk where they were doing the problems, with one exception they were unanimous that they found it easier to read a problem set on paper than on a screen, and would have printed the problem set regardless of whether it was possible to read it on-line at their desk.

Certainly the author of this paper finds it easier to read many types of material on paper than on-line. A personal question has been whether this is because of my age and the available technology when learning to read. Evidently our current twenty year old students still prefer paper.

In contrast, when our inexperienced first year students were asked whether they would print fewer problem set solutions if they had a screen on their desk, about half said they would. We believe this is a good choice for those students checking a solution to a problem that they have already tried, but is yet another poor choice for students reading problems they have not attempted to solve.

II.C Using Technology As a Distribution Medium

We have been suggesting that on-line reading is most effective when the information is fairly well *encapsulated*, and that the role of paper is still important for other types of information. By encapsulated we mean that it contains only a few quanta of information and, perhaps more important, the *context* of that information is already known. We shall explore other types of encapsulated information in succeeding subsections.

However, the technology allows a solution to a long standing problem in traditional courses. Virtually every academic discipline hands out large quantities of paper. These contain curricula, calendars and sometimes lengthy supplementary course notes. One always duplicates enough for all students with extra copies, which means that every term a quantity of paper is not picked up by students and ends up in the recycling bins.

More and more in our department, we post such information on our web site and leave it up the student to print the document if they wish.[3] Although *we* are concerned about the appropriateness of shifting the costs of printing to the students, there have been no complaints from the *students*.

Using technology for up-to-date delivery of course materials also makes some forms of distance education much more practical than using the rightly named ‘snail mail’.

Another example of electronic delivery is fairly simple, but perhaps instructive in yet another way. In the third year engineering course discussed in the previous subsection, it is standard for the author of this paper to keep the teaching assistants in the course informed of what is going on in class by weekly email summaries, which are typically on the order of 60 lines long. The course itself has a

3. We are currently using Adobe’s *Acrobat* and *Portable Document Format*.

"home page", containing office, telephone and office hours for staff, a curriculum, calendar, marking scheme etc. A few weeks into the term it struck me that including these weekly summaries into the course home page would be a trivial effort on my part and perhaps of use to the students. I was disappointed in myself for taking so long to come to an obvious idea. The instruction in this example, then, is that the technology allows us to do new and useful things, but our natural tendency to get into a rut makes many simple applications difficult for us to think of.

II.D Numerical Applications

So far, we have concentrated our discussion largely to areas with the greatest relevance for all academic disciplines. Here we shall concentrate more on areas of computational pedagogy with particular applicability to the sciences, engineering and perhaps the social sciences. This is computational "number crunching", which is the heart of much of the way physics is carried out in the real world.

Historically physics has been a bifurcated discipline, with theoretical physics and experimental physics being somewhat distinct from each other. Over time the interaction between theory and experiment has become well defined, and the natural tension between experimentalists (who claim they "don't spend all day pushing symbols around a piece of paper with a pencil") and theoreticians (who proudly state that they "don't roll around on a dirty lab floor with a screwdriver in their teeth") has had a largely positive effect on physics. The effect of computers on the way physics is carried out has led some to suggest that the discipline is now *trifurcated* into theoretical, experimental, and computational physics. One of my experimentalist colleagues said to me recently that he doesn't "need to have theorists any more because [he] can do the theory with a computer." In fact, whole new fields of physics such as chaos, non-linear dynamics, and more are possible only with the assistance of computers.

The last time I conversed with other educators at an *I.U.T.* conference, our computational efforts were concentrated on our first year laboratory program [Harrison 1988a]. Our user interfaces have been greatly improved since that report, but we still provide least-squares curve fitters, on-line instrument specifications, unit conversions and a test on error analysis [Harrison 1983, 1988b].

In the context of the discussion of this paper, these all provide encapsulated information. Our curve fitters, for example, return a few numbers: the fit parameters (values of the slope and intercept for example), errors in those fit parameters, the chi-squared or sum-of-squares statistic and the number of degrees of freedom; they also provide a graph of the data, the results of the fit to the data and a plot of the residuals of the fit. The user interface is a "point and click" one, which we intend to convert to a World Wide Web base in the near future.

Similarly, when a student accesses our on-line instrument specifications they are typically looking for one or two numbers in a table, such as the accuracy of a voltmeter on the 10 volt scale. The specification sheet also includes a color photograph of the instrument, primarily so the student is sure they are accessing the instrument they want (such as a Hewlett-Packard 3466A multimeter) instead of another with a similar name (such as the very different Hewlett-Packard 3476A multimeter). The sheet also includes instructions for use of the instrument, which we now think may not be useful to the students in an on-line form. The underlying technology is SGML, which is filtered to a World Wide Web page.

In the past few years, there has been a revolution in the capability of computerised mathematical systems.[4] This software has had an important impact in spreading numerical computation throughout the upper years of our undergraduate program.

Although we offer these tools to our beginning students, their usage usually does not go much beyond using the programs as advanced calculators, which only scratches the surface of their capabilities. However, there is another large impact on our first year pedagogy just because such software exists. Previously, a fair amount of class time was spent in developing the mathematical skills necessary to solve a particular physics problem. The problem-solving process, then, was logically divided into three steps:

1. Setting up the problem.
2. Solving the mathematics.
3. Interpreting the result.

The first and third steps are physics, but the second step is just an exercise in pushing symbols around on a piece of paper with a pencil, and we often stressed to students that once the problem was set up the equations could be taken to a mathematician for solution without telling the mathematician what the problem was about.

4. *Mathematica* and *Maple* are two well-known examples of this type of software. We believe both are excellent, but have done most of our work with *Mathematica*.

Now we expect that later versions of this type of software will be sitting on the desks of our students when they get out into the real world, and that the software can do the second step in solving the problem. The implication is that we no longer need to devote significant class time to developing the mathematics, which allows an increase in course content without increasing student workload.

In our upper year program, we increasingly have problem sets which require the use of this type of software [see Douglas 1995 for example]. In addition we now offer three half-year courses in computational physics at the third and fourth year level.[5] Growing numbers of our upper year students choose to use the software to do the mathematics for *all* of their problem sets.

The software allows an electronic document consisting of text, computer commands, output, and graphics to be mixed. The input commands can be edited by the user, which means that the document is "alive" when it is connected to the "kernel" that actually carries out the processing. These *notebooks* are the basis for our courses in computational physics. A typical notebook contains introductory material, references, sample commands, input fragments, and code listings. The students edit an electronic copy of the notebook to do their work, and submit the electronic file for marking when complete.

Almost all students in these courses work from a hardcopy listing of the original notebook, which they keep beside the terminal as they are doing the assignment. Similarly, we print the student's notebook and typically have it on our desk beside a screen running the electronic version as we mark the assignment. We have found it preferable to make our comments and corrections on the hardcopy version, which is then handed back to the student. We have never observed a student in these courses produce a hardcopy version of their own notebook, although we do observe them produce listings of computer programs that they are developing or debugging.

With reference to our discussion in this paper, the notebook usage in our computational physics courses is consistent with the suitability of encapsulated information for on-screen representation. A student's output from an input command is typically only a few lines of numbers and/or symbols and/or a graph; since the

5. A half-year course is roughly equivalent to a one semester course.

output is from the student's own input, the context is also known in advance. Longer expositions are read in paper form.

However, there are other issues about on-screen representation which are perhaps unique to mathematical disciplines. First is the fact that traditional mathematical notation has the ambiguities of all natural languages. For example, consider an x with a bar over it: $\bar{x}$. Does this indicate the *mean* of a collection of measurements x, or a vector quantity x or .... The notation is ambiguous. This leads to considerable difficulty when one is trying to represent mathematics in a traditional format and still have the document be alive and interpretable by a kernel in an unambiguous way. Fortunately, this problem seems to be near a solution.

Another problem with on-screen representations of mathematics compared to paper involves the differences in resolution and perhaps some other human factors. For example, often one has a superscript to a superscript: X^{Y^Z}. Hopefully even after duplication, the Z superscript to the Y is readable. If one tries to represent this page on a computer screen, that Z becomes almost totally unreadable unless it is magnified. There are similar problems with serif characters losing their serifs, and difficulties in representing things like arrow symbols over characters as an indicator of a vector, such as $\vec{x}$. Thus, considerable work on fonts and basic changes in typesetting rules such as the appropriate decrease in point size for superscripts and subscripts is currently under way.

Meanwhile, authors trying to represent mathematics over the World Wide Web have two alternatives, neither very satisfactory.

1. Represent the document as a normal page, using some technology such as Adobe's *Acrobat* and *PDF*. The on-line reader will then likely have to spend considerable time magnifying various parts of the document.

2. Incorporate the mathematics into a HTML document by turning it into a gif image, with suitable magnifications (typically 150 to 200%) included. This requires a large effort by the author, and generally looks awful.

Note that neither of these approaches allows the document to be alive and interacting with a kernel.

II.E A Failed Bulletin Board

For some years we have felt that a course based computer bulletin board system along the lines of Usenet news could be useful to our students. The only thing that has kept us from implementing such a facility has been that the significant learning curve of the user interface of traditional news readers (such as *rn* or *trn*) gave us images of hundreds of students lined up outside our office because they were stuck.

Over the summer of 1996, we implemented a World Wide Web based bulletin board system called *HyperNews*.[6] We felt the user interface was simple and effective, and in September 1996 installed "groups" for eight undergraduate courses and labs. Now, in late December 1996, not a single student has used the facility to "post" a message.

Although certainly a failure, we are still not sure why. Note that it probably does not relate to the encapsulation issue that has dominated this section.

III. CONCLUSIONS

University students are sometimes very smart about assessing what will and will not help them to do well in their courses. When we first introduce a software tool in support of our undergraduate program, the newness plus the enthusiasm of the developers means that often a fair amount of initial exploration occurs by the students. However, they will continue to use the facility only if they perceive that it is helping their studies. Thus, continued student usage is one of the primary means by which we evaluate the success of our efforts.

As a result of our experiences we are increasingly making a distinction between materials which can be effectively read on a computer screen and materials which are distributed electronically but are intended to be printed and read on paper. The former type of material we have been calling *encapsulated*, by which we mean a fairly brief piece of information within a known context. We see no indication that the current generation of students are any better at effectively reading non-encapsulated information on-screen than are their more elderly professors.

This distinction has specific consequences in terms of our assessment of the lack of effectiveness of traditional self-paced tutorials for *Computer Assisted Instruction* or *Computer Based Instruction*. Such software involves the delivery of non-encapsulated information. If we are correct, it also has ramifications in the

6. The software was developed by Daniel LaLiberte et al. at the National Center for Supercomputing Applications, and has been locally modified.

continuing controversy about the very future of the book [see Nunberg 1996, for example]. Another exciting consequence is that an electronic document has the potential to become ‘alive’, and we have had great success in using such documents in our courses; such a document also blurs the distinction between the author and the reader.

There is nothing particularly revolutionary in our conclusion, and the study of the distinction between on-screen and paper documents has a huge and somewhat overwhelming literature. We intend to continue to monitor our students to watch for changes in attitudes to reading non-encapsulated information on-screen. The fact that our current students are no better than we are at this task may not be surprising and may change in time; as Soetaert and Van Belle wrote:

> If we’re indeed witnessing a revolution comparable to the Gutenberg revolution, we should be able to learn something from the past. We find, for example, that transformations in thinking, reading, writing, and acting under the influence of a new medium don’t occur from one day to the next. We’re talking of hundreds of years of evolution, the results of which today we consider "natural." [Soetaert 1993].

Although not explicitly discussed but implied in much of the above discussion is the importance of hypertext in accessing information, particularly encapsulated information. This is one of the reasons why we intend over the next few years to have the capability of offering *all* instructional materials, with the exception of physical laboratory apparatus, in a network accessible form. Also, the effect of multimedia on the "readability" that has been the principle focus of this paper is still an unanswered question.

CONTACTING US

The author may be reached at *harrison@physics.utoronto.ca*. The ‘snail mail’ address is:

Dr. David M. Harrison
Department of Physics
University of Toronto
Toronto, Ontario M5S 1A7 CANADA

The *UPSCALE* home page’s URL is *http://www.upscale.utoronto.ca*.

ACKNOWEDGEMENTS

Many people in the Department of Physics have invested significant effort into the projects described here. These include R.C. Bailey, J.R Drummond, B. Holdom, W.R. Peltier, J.M Pitre, T.G. Shepherd, P. Sinervo and S.S.M. Wong. Our efforts have been financially supported by the Department, the Faculty of Arts and Science and the Provost at the University.

In addition, we have greatly benefited by donations of equipment and funding from Hewlett-Packard (Canada) and Northern Telecom (NorTel) Ltd.

REFERENCES

Bork 1981	Alfred M. Bork, **Learning With Computers** (Digital Press, Bedford Mass., 1981).
Bork 1985	Alfred M. Bork, **Personal Computers for Education** (Harper & Row, New York, 1985).
Douglas 1995	S.C. Douglas, D.M. Harrison and T.G. Shepherd, "The Physical Pendulum in an Advanced Undergraduate Course in Mechanics," Computers in Physics **8**, 416 (1995).
Harrison 1983	D. Harrison and J.M. Pitre, "Computerised lab test on error analysis," The Physics Teacher **21,** 588 (1983).
Harrison 1988a	David Harrison, "A computer is just another piece of apparatus: *CAI* in a Science Laboratory", **I.U.T XIV** (July 1988, Umeå Sweden) 719.
Harrison 1988b	D. Harrison and J.M. Pitre, "Computers in a teaching laboratory: just another piece of apparatus," Computers and Education **12,** 261 (1988).
Nunberg 1996	Geoffrey Nunberg, ed., **The Future of The Book** (Univ. of California Press, 1996).
Soetaert 1993	Ronald Soetaert & Guy Van Belle, "Screening Screens for Literacy", the text of a lecture delivered at the General Conference of the Dutch Language and Literatures in Dutch, "An Image of Literacy," which was organized by the Dutch Language Union at the Antwerp Statehouse on November 5, 1993. The URL is *http://dewey.rug.ac.be/CAMERspace/Camera/Screens.html*.