Promoting computer literacy throug pyton

Science, Computer Science

PROMOTING COMPUTER LITERACY THROUGH PROGRAMMING PYTHON by John Alexander Miller A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Education) in The University of Michigan 2004 Doctoral Committee: Professor Frederick Goodman, Chair Emeritus Professor Carl Berger Professor Jay Lemke Professor John Swales Copyright 2004 by John Alexander Miller The Joys of the Craft Why is programming fun? What delights may its practitioner expect as his reward? First is the sheer joy of making things. As the child delights in his mud pie, so the adult enjoys building things, especially things of his own design. I think this delight must be an image of God's delight in making things, a delight shown in the distinctness and newness of each leaf and each snowflake. Second is the pleasure of making things that are useful to other people. Deep within, we want others to use our work and to find it helpful. In this respect the programming system is not essentially different from the child's first clay pencil holder “ for Daddy's office. " Third is the fascination of fashioning complex puzzle-like objects of interlocking moving parts and watching them work in subtle cycles, playing out the consequences of principles built in from the beginning. The programmed computer has all the fascination of the pinball machine or the jukebox mechanism, carried to the ultimate. Fourth is the joy of always learning, which springs from the nonrepeating nature of the task. In one way or another the problem is ever new, and its solver learns something: sometimes practical, sometimes theoretical, and sometimes both. Finally, there is the delight of working in such a tractable medium. The programmer, like the poet, works only slightly removed from pure thoughtstuff. He builds his

castles in the air, from air, creating by exertion of the imagination. Few media of creation are so flexible, so easy to polish and rework, so readily capable of realizing grand conceptual structures. ... Yet the program construct, unlike the poet's words, is real in the sense that it moves and works, producing visible outputs separate from the construct itself. It prints results, draws pictures, produces sounds, moves arms. The magic of myth and legend has come true in our time. One types the correct incantation on a keyboard, and a display screen comes to life, showing things that never were nor could be. Programming then is fun because it gratifies creative longings built deep within us and delights sensibilities we have in common with all men. (p. 7) Frederic Brooks, Jr. The Mythical Man-Month, 1975 DEDICATION Dedicated to my wife, Jane, and our children, Jessa and Jason. ii ACKNOWLEDGMENTS There were many people who encouraged me while I worked on this dissertation. First, I’d like to thank my parents, Mervil and Barbara Miller, who provided a loving, supportive and stable family environment conducive to intellectual pursuits. “ A good beginning is half the battle. " I’d also like to thank Charles and Genevieve Peterson, who extended an essential educational career opportunity to me, and continue to bless me with caring support to my family as it grows. I would also like to thank the numerous students I had in my ESL classroom: their kindness, generosity and respect revealed to me the true joy of teaching. Next, I’d like to thank my advisor, Frederick Goodman, for his practical wisdom and guidance. With our every conversation, I come away enriched and refreshed from the depths of his memory and experiences. He is an educator’s educator, and I am honored to have had the opportunity to work so closely with him. I would

also like to thank the other committee members, iii Carl Berger, Jay Lemke, and John Swales, for their suggestions and recommendations as the dissertation evolved. I would like to thank my colleagues at the Interactive Communications and Simulations group: Jeff Stanzler, Jeff Kupperman, Gary Weisserman, Roger Espinosa, Denise Conanan, Douglass Scott, Edgar Taylor, and Michael Fahy. Spending the early, heady days of the World Wide Web together engendered many stimulating and enjoyable conversations. I would also like to thank Gil Oswald for the time he took reviewing a draft of this manuscript, and for his gracious and thoughtful suggestions and feedback. Finally, I would like to thank my wife, Jane, who persevered knowing that I could indeed bring this dissertation to completion. iv TABLE OF CONTENTS Page

and inventions and technologies become collective efforts and cultural products. With your brain alone, with my brain alone (minus language and a language-based tradition), we would consider ourselves very lucky indeed to think of cracking nuts between a stone hammer and a stone anvil. Our greatest human creation is not the tool but the word, not the technology that we so treasure and depend on but the language that has allowed us to talk about it. Language, not technology, is the most compelling artifact of the human intellect. –Dale Peterson, Eating Apes 1. 0 Introduction Computers permeate our lives at work and at home, satisfying professional and recreational goals, enabling a kind of cybernetic1 activity that one could only imagine a few decades ago. There is a rich variety to these activities including: systems modeling, graphic designing, multimedia content creation and organizing, word processing, financial accounting and transacting, database processing of myriad types of information, scientific hypothesis testing, not to mention near ubiquitous e-mail corresponding, web 1 “ goal-oriented feedback mechanisms with learning" (Pickering, 1995, p. 31) 1 2 surfing and instant messaging activities that keep us in touch with each other. For many, if not most, there seems to be a natural affinity with these machines which gratify a cognitive desire to store, retrieve, and manipulate information that is important to each of us. However, it is important to realize that the cybernetic activities described above are mediated through computer applications that have been written. This particular form of writing is called computer programming and constitutes the domain of this dissertation. In order to communicate with a computer a programmer must use a language. Although there are many computer languages from which to

choose, there are few designed with the beginning programmer in mind. One language so designed is Python, and it currently enjoys a burgeoning popularity among computer programmers. Since computer programming must be learned, and since Python was designed to be easy to learn, this dissertation explores what considerations are most important in teaching Python as a first programming language in a secondary school setting. In order to address this issue, a large corpus consisting of over three years worth of messages posted to a public newsgroup was analyzed using innovative techniques to reduce the data and isolate the relevant portions. This dissertation, then, is also a study of what techniques are efficacious in reducing large, textual datasets and their applicability to other researchers engaged with similarly large, unstructured textual data. 3 In order to address the first question, it will be worthwhile framing it in a larger context of literacy, that is, what considerations are most important in teaching people to become increasingly literate in the age of the computer? We shall see that understanding literacy as a way of knowing enables us to understand computer literacy as an alternative and supplementary way of knowing; and furthermore, that computer programming is an essential component of computer literacy, analogous to the way writing is essential to traditional literacy. This first chapter is a comparison of traditional print literacy with the much newer computer literacy. We will see how computer programming is related to computer literacy, and how computer literacy is more related to the rhetorical function of using computers to express oneself than to knowing how computers themselves work. We then look at how print and computer literacies enable learning and why computer programming is an

essential component of that learning process. The second chapter discusses what skills and processes are involved in writing a computer program, focuses on some connectionist considerations surrounding the teaching of programming, takes a look at the Python programming language itself, and describes the genesis of the data source. The third chapter explicates the methods and procedures used to analyze the data, distinguishes what was considered relevant from irrelevant 4 in the data, and describes the eight categories that emerged and served as a framework for the analysis. The fourth chapter provides the results of the analysis. Going category by category, the concerns of the posters that addressed this dissertation’s topic are detailed and summarized. Also, the results of using the methods and procedures delineated in the third chapter are discussed. The fifth and final chapter presents the conclusions drawn from the results and considers the larger role of programming as a literate practice. 1. 1 Literacies The term ‘ literacy’ is somewhat charged with contested meanings. A sense of this can be gleaned from the Usage Note for the word ‘ literate’ in the American Heritage Dictionary (1992): For most of its long history in English, literate has meant only “ familiar with literature, " or more generally, “ well-educated, learned"; it is only during the last hundred years that it has also come to refer to the basic ability to read and write. … More recently, the meanings of the words literacy and illiteracy have been extended from their original connection with reading and literature to any body of knowledge. For example, “ geographic illiterates" cannot identify the countries on a map, and “ computer illiterates" are unable to use a word-processing system. Certainly, the use of the term ‘ literacy’ in this work is governed more by the

later, more modern sense indicated above. In other words, I take computer literacy to be more inclusive than just ‘ familiarity with computers’; I explicitly include a ‘ reading and writing’ component. In the context of computer usage, ‘ reading’ roughly corresponds to the ability to use a computer’s operating system and applications productively (which may 5 include the reading and writing skills of traditional print literacy), while ‘ writing’ roughly corresponds to the ability to program a computer. The opposite of literacy is illiteracy (keeping in mind that these are measured in degrees rather than existing as categorical states), and it is often claimed that print illiteracy rates are rising in this country. Mihai Nadin (1997) expresses this complaint in The civilization of illiteracy: We notice that literate language use does not work as we assume or were told it should, and wonder what can be done to make things fit our expectations. Parents hope that better schools with better teachers will remedy the situation. Teachers expect more from the family and suggest that society should invest more in order to maintain literacy skills. Professors groan under the prospect of ill-prepared students entering college. Publishers redefine their strategies as new forms of expression and communication vie for public attention and dollars. Lawyers, journalists, the military, and politicians worry about the role and functions of language in society. … The major accomplishment of analyzing illiteracy so far has been the listing of symptoms: the decrease in functional literacy; a general degradation of writing skills and reading comprehension; an alarming increase of packaged language (clichés used in speeches, canned messages); and a general tendency to substitute visual media (especially television and video) for written language. (pp. 3-4)

However, instead of bemoaning the decline of literacy, Nadin embraces the notion that its decline is inevitable, and seeks to describe human life and interactions that emerge in an ‘ illiterate’ world: The decline of literacy is an encompassing phenomenon impossible to reduce to the state of education, to a nation’s economic rank, to the status of social, ethnic, religious, or racial groups, to a political system, or to cultural history. There was life before literacy, and there will be life after it. … My position in the discussion is one of questioning historic continuity as a premise for literacy. If we can understand what the end of literacy as we know it means in practical terms, we will avoid further lamentation and initiate a course of action from which all can benefit. Moreover, if we can get an idea of what to expect beyond the safe haven now fading on the horizon, then we will be able to come up 6 with improved, more effective models of education. … This leads me to state from the outset–almost as self-encouragement–that literacy, whose end I discuss, will not disappear. … For the majority, it will continue in literacies that facilitate the use and integration of new media and new forms of communication and interpretation. The utopian in me says that we will find ways to reinvent literacy, if not save it. … We give life to images, sounds, textures, to multimedia and virtual reality involving ourselves in new interactions. Transcending boundaries of literacy in practical experiences for which literacy is no longer appropriate means, ultimately, to grow into a new civilization. (pp. 5-7) It is within this sense of ‘ transcending boundaries of literacy’ that I frame my argument for computer programming. I do not argue that the rise of illiteracy must be quelled, rather, I believe that computer literacy will arise as one of the new literacies Nadin speaks of, and

that active promotion of computer programming will lead to computer fluency, which, due to its reliance on specific written syntaxes, will also reinvigorate traditional print literacy skills. This reinvigoration is implied by Knuth (1992) and his thoughts on Literate Programming: Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do. The practitioner of literate programming can be regarded as an essayist, whose main concern is with exposition and excellence of style. Such an author, with thesaurus in hand, chooses the names of variables carefully and explains what each variable means. He or she strives for a program that is comprehensible because its concepts have been introduced in an order that is best for human understanding, using a mixture of formal and informal methods that reinforce each other. (p. 99) A similar vision of intertwining literacies is offered by Olson (1985): 7 Computers simply raise, by an order of magnitude, the requirement of making meanings explicit–a requirement that was begun by giving communication systems a semantics and by then making that semantics more explicit and elaborate through literacy. Far from being obsolete, literate competencies are basic to computational ones. To be intelligent in the society of computer users is to be skilled in making one’s meanings explicit. In the 20th century, the world of computing, we are succeeding in doing what the 16th century, the world of literacy, aspired to do–make meanings clear and explicit. (p. 7) Knuth challenges programmers to write programs that are ‘ explicit’ in two ways, explicit to the computer, in the sense Olson describes, but also explicit to other human readers of

software programs. In a sense, I’m saying we can have our cake and eat it too: Computer literacy is emerging as an alternative literacy in the way Nadin describes, but as such, computer literacy (especially programming) is so strongly dependent upon traditional print literacy skills that promoting computer programming also strengthens those traditional skills. Furthermore, we can understand this strengthening, this interdependence of literacies in terms of semiotics as illustrated by Barthes and Lavers (1993) in Mythologies where they define myth as a second-order semiotic system. Using their system, I showed how computer literacy and print literacy are each second-order semiotic systems where mastery of the first-order semiotic system is a prerequisite for mastery of the second-order one (Miller, 2001). In this case, computer programming strongly depends on a specific written syntax, so at least minimal competence in print literacy skills is necessary for the budding programmer. 8 Besides extending print literacy skills, computer literacy can also extend scientific literacy skills. In A New Kind of Science (Wolfram, 2002) we are offered a vision of ways computer literate scientists can conduct experiments and perform research using computers. Wolfram suggests that much research in the future may be done this way: For with calculus there was finally real success in taking abstract rules created by human thought and using them to reproduce all sorts of phenomena in the natural world. But the particular rules that were found to work were fairly sophisticated ones based on particular kinds of mathematical equations. And from seeing the sophistication of these rules there began to develop an implicit belief that in almost no important cases would simpler rules be useful in reproducing the behavior of natural systems.

During the 1700s and 1800s there was ever-increasing success in using rules based on mathematical equations to analyze physical phenomena. And after the spectacular results achieved in physics in the early 1900s with mathematical equations there emerged an almost universal belief that absolutely every aspect of the natural world would in the end be explained by using such equations. Needless to say, there were many phenomena that did not readily yield to this approach, but it was generally assumed that if only the necessary calculations could be done, then an explanation in terms of mathematical equations would eventually be found. Beginning in the 1940s, the development of electronic computers greatly broadened the range of calculations that could be done. But disappointingly enough, most of the actual calculations that were tried yielded no fundamentally new insights. And as a result many people came to believe–and in some cases still believe today–that computers could never make a real contribution to issues of basic science. But the crucial point that was missed is that computers are not just limited to working out consequences of mathematical equations. And indeed, what we have seen in this chapter is that there are fundamental discoveries that can be made if one just studies directly the behavior of even some of the very simplest computer programs. (pp. 44-45)

9 Thus, scientific computer simulations are not just the working out of mathematical equations, but rather, the execution of algorithms that may model physical phenomena that cannot be represented by equations. And it is the conceptualization and embodiment in code of those algorithms that will more and more constitute the life of the mind of future scientists.

Another way scientific literacy intersects with computer literacy is the

growing field of genetic programming. In “ Evolving Inventions" (Koza, Keane et al. 2003) write: The first practical commercial area for genetic programming will probably be design. In essence, design is what engineers do eight hours a day and is what evolution does. Design is especially well suited to genetic programming because it presents tough problems for which people seek solutions that are very good but not mathematically perfect. Generally there are complex trade-offs between competing considerations, and the best balance among the various factors is difficult to foresee. Finally, design usually involves discovering topological arrangements of things (as opposed to merely optimizing a set of numbers), a task that genetic programming is very good at. (p. 54) Furthermore, as such computer literate scientists perform algorithm-based, simulation-based and genetic programming-based experiments with their computers (rather than, or alongside, mathematical equation-based ones) they can help educate a new generation of student/scientists who are ‘ looking over their shoulders’ via internet-based legitimate peripheral participation (Lave and Wegner, 1991). The rise of computer-mediated communication systems enables a kind of collaboration-at-a-distance not only between scientists who are peers, but also between master scientists and student apprentices. 10 1. 2 Literacy Analogy In order to make the connection between print literacy and computer literacy, I would like to elaborate on an analogy: Reading : Writing :: Using a computer : Programming a computer In order to precisely discriminate what is meant by each term in the analogy, it helps to define each of them as having two aspects: an external function and an internal function. Reading, for example, has the external, or mechanical, function of

seeing text on a surface (or, more generally, perceiving signs in an environment) and the internal function of interpreting that text to assign meaning(s) to it (usually, but not always, with the aim of reproducing meaning(s) intended by the author in the reader's mind). Writing also has an external, or mechanical, function of manipulating a writing instrument (pen, typewriter, word processor) on a surface (paper, screen) to create text and an internal function of strategizing which words, phrases, ideas, and rhetorical devices to ' textualize' in order to create the written artifact. Likewise, the other two, computer-based terms in the analogy have a dual aspect akin to reading and writing, so that we may say ' using a computer' has the external, mechanical function of seeing signs on a surface and the internal function of assigning meaning to those signs, while ' programming a computer' has the external function of manipulating a device (computer keyboard and mouse) and the internal function of devising and 11 designing data structures and algorithms using a computer language's syntax and vocabulary to create a programming artifact. Although both sets of literacy skills, reading and writing text, and using and programming computers, involve the encoding and decoding of signs (discussed shortly), the salient difference between them is the object of their attention. With written artifacts the object of attention is what is said (written), whereas with programmatic artifacts, the object of attention is what is done. Suppose, for example, we are examining a PDF (Portable Document Format) document on a computer screen. To the extent that our attention is focused on the textual content of the document, we are engaged in a traditional print literacy skill: reading. However, to the extent that our attention is focused on jumping to

page 25 of that document, or adding a bookmark to it, or adding a personal comment, or copying a passage, or zooming in for an enlarged view, or clicking on a hyperlink, or emailing the document to a colleague, or printing selected pages, or even using the Help menu to find out how to do any of these actions with the document, we are engaged in a computer literacy skill: using a computer. 1. 3 Computer Literacy Conventions In the past two decades, computers have become almost ubiquitous (in the US and other developed countries). Computers enhance and extend traditional print (and other kinds of) literacy skills by greatly augmenting one’s ability to manipulate the symbols of literacy: alphabetic characters, 12 words, paragraphs, documents, references, numerals, equations, charts, maps, images, graphics, sounds, video, etc. However, these machines also Table 1 Alphabet Spelling Punctuation Grammar Sentences Paragraphs Rhetorical Devices Chapters Quoting Stories Poetry Novels Typical literacy conventions Footnotes Endnotes Indexes Tables of Contents Parenthetical comments Authors Textbooks Reference Works Periodicals Letters Publishers Anthologies impose an additional cognitive load on the various literacy skills they enhance. For example, before computers, being literate consisted of learning a wide-ranging set of conventions applied to written text (see Table 1). The pervasive expectation threading throughout all these literary conventions was that the symbols being manipulated were either immutably printed, or relatively fixed, on paper. This, in turn, imposed certain limitations on the degree of interactivity with those symbols by both writer and reader. With computers and word-processing applications and the Internet, these limitations have greatly diminished, and expectations of what

the literacy skill set consists of have risen concomitantly. For instance, cutting and pasting text from one document to another, formatting a document for publication in a journal, finding and replacing all occurrences of the word ‘ skill’ with the word ‘ ability’ throughout a document, inserting a bar 13 chart into a document, and posting a version of an article on a webpage are all examples of computer-assisted literacy skills that can reasonably be Table 2 Files Undo Find and Replace ‘ Desktop’ Attachment Typical computer literacy conventions Folders Copy and Paste Multimedia Presentation ‘ Clipboard’ Spreadsheet expected of a literate person these days. In other words, these procedures and their artifacts constitute additional conventions that could be added to a list like the one above (see Table 2). This raises a question. Do these additional operating system and wordprocessing skills constitute an extension of what it means to be literate, or do they, along with other computer-enhanced symbol-manipulation skills such as using a spreadsheet and presentation software and digital photo editing, constitute a new kind of literacy, computer literacy? I argue that facility with computers is indeed a new kind of literacy. The earlier literacies (print literacy, numeric literacy, media literacy) represent ways of communicating with others where ‘ others’ in those cases are usually assumed to be ‘ other humans’ (with the rare exception being chimpanzees or dolphins). However, in this case, computer literacy represents a way of communicating with, not another human (directly), but with a machine (or perhaps more accurately, a machine system, if the computer is attached to a 14 network)2. Even though it is probably most often the case that the reason the user is interacting with a machine is to create an artifact (or message)

which will be communicated to another human, the concepts involved in accomplishing this task are different enough from those of traditional print literacy that it constitutes a qualitatively different experience which deserves a ‘ literacy’ appellation. We expect computer users to design artifacts, retrieve information, configure operating systems, program and use applications, maintain hard drives, as well as have some sort of understanding of how their computers, networks, applications, file systems and digital objects work. That is, there now exists a rich set of abstract symbols constituting operations on ‘ computable resources’ which manipulators (‘ writers’ and ‘ readers’) of those symbols need to learn in order for meaningful communication (between human and machine) to occur and which we can refer to as computer literacy. For example, sending an e-mail attachment to a newsgroup constitutes a symbol manipulation that requires a certain amount of care and understanding that is different from what is involved with sending the same information through postal mail. There are a host of variables to consider when performing this seemingly simple act, and becoming computer literate involves anticipating and dealing with those variables. First there is the It could be argued that it is humans who created the computer and the communication protocols for interacting with it, so communication with a machine is indirectly communication with the person (or committee, or company) that created the machine and its protocols. 2 15 attachment’s file format. Most any recipient will likely be able to open a simple text file. However other text documents created by proprietary word processors require the same or similar application on each recipient’s machine (or at least some way of getting it opened remotely).

There is also the issue of how the newsgroup server handles attachments: some let them through; many others discard them. How is the attachment to be compressed, if at all? What about language encoding issues? What about the perceived risk of viruses hidden in attachments? Is there another way to distribute the information? Understanding these and other issues means understanding in some way how computers operate, and how our interactions with computers affect the way our messages and other artifacts are handled. Understanding many of these issues involves understanding conventions which, when consistently invoked by a critical mass of users, becomes part of the knowledge base for interacting with computers. These conventions, in turn, create expectations about the process of creating, sending and receiving messages and artifacts mediated through computers. Three kinds of signs Computer code consists of sequences of signs generated by humans in a computing environment. Charles Sanders Peirce postulated that there were three kinds of signs: icons, indexes, and symbols. These three kinds are distinguished by the relationship they hold with their ‘ dynamic object,’ or 16 that which the sign refers to (Boyarin, 1992, p. 115-116). He identified icons as signs that are determined by their dynamic object by virtue of their own internal nature and are characterized by qualities of feeling and unity. For example, icons easily identify many corporations, flags serve as iconic representations of countries, and many traffic signs are iconic rather than textual. In this context, we note that computer interactions utilizing a graphical user interface heavily rely on the use of icons to identify applications and document types (for example, . pdf, . doc, . qt, . psd and . zip files); eventually, these icons become conventional.

Peirce says that the second kind of sign, indexes, are determined by their dynamic object by virtue of being in a real relation to it and are characterized by the experience of effort and action. For example, proper names serve as indexes to the people they refer to, as do symptoms of diseases. In computer literacy, the naming of the documents we create is a primary example of indexical signs. Often, the extensions at the end of the filename are used to identify the kind of file it is, and these indexical extensions also become conventional (for example, . jpg, . pdf, . doc, . gif, . txt, etc.) A similar convention exists for the naming of network domains. However, it is the third kind of sign we are most interested in, symbols, which are signs that are determined by their dynamic object only in the sense that they will be so interpreted. In other words, a symbol’s meaning is wholly arbitrary, and as such, depends on convention, habit, or a natural disposition 17 of the interpretant. This, of course, includes programming languages such as Python where the symbols used are wholly arbitrary (yet resonant with the written English language, a much larger set of arbitrary symbols) and where the programmer learns the conventions of the language in order to write expressions that the Python interpreter can properly interpret. So, we see that computer literacy consists of learning a set of conventions arising out of the use of different kinds of signs. This applies not only to the use of computers, but also encompasses the programming of computers. Three pillars of literacy Andrea diSessa (2000), in Changing Minds: Computers, Learning and Literacy describes what he calls the three pillars of literacy. The first is the material pillar involving “ external, materially based signs, symbols, depictions, or representations. " The

components of interacting with a computer, including computer code, certainly consist of these kinds of materials as we just saw in the discussion of signs. DiSessa also says that the material pillar of literacy has two important features: they are technologically dependent, and they are designed. Both of these features are true of computer artifacts and computer languages. The second pillar of literacy is mental or cognitive, that is, the inscriptions of the material pillar are meaningless without a corresponding consciousness to bring meaning to them. “ Clearly the material basis of literacy stands only in conjunction with what we think and do with our minds 18 in the presence of inscriptions" (p. 8). Indeed, this cognitive pillar constitutes much of the effort of learning a programming language, as detailed in the five kinds of programming knowledge in the second chapter (the knowledge framework for teaching programming). But it also presupposes much auxiliary knowledge associated with computer literacy, that is, knowing how to use a computer. The third pillar of literacy is social, that is, a literacy does not exist solely with one person. Its conventions are shared among a group of people who use it to communicate or calculate or perform some other function of importance to the group. Computer use is certainly social in that many people share the conventions of using their machine’s operating system, and of the applications that run on it. Programming languages, such as Python, are also included in that social milieu by having their own set of conventions that are shared by the programmers who learn that language. Having described these three pillars, diSessa (2000), defines a central hypothesis of literacy: “ A literacy is the convergence of a large number of genres/social niches on a common,

underlying representational form" (p. 24). We see this definition in action by considering the multitude of ways people use their computers. Thus, one can send and receive a variety of text-based messages such as e-mail, word processing documents, newsletters, etc. using a computer (‘ a common, underlying representational form’). One can read and create static graphic-based messages such as spreadsheet documents, 19 graphing calculator equations, drawings, presentations etc. using a computer (‘ a common, underlying representational form’). One can read and create dynamic graphics such as, movies, slideshows, virtual reality scenes, etc. using a computer (‘ a common, underlying representational form’). Across all of these activities (genres, social niches), is a quickly evolving (not yet universal) set of conventions that enable users to ‘ read’ the computer applications being employed for their tasks, ‘ write’ the messages they wish to convey, and ultimately, create new message-creating tools themselves. Increasingly, too, this ‘ body of knowledge’ that computer literacy is comprised of is coming to resemble the cognitive structures that computer programmers use to construct their programs and applications, as if these artifacts that computer users create were ‘ programmed’. Harrell (2003) details this evolution: The tools used to present and create media art lie behind every media artwork. The theory of programming languages is a useful means by which to characterize these media. Formal languages offer broad insight into the nature of computational manipulation and specific organizational structures of imperative languages reveal reflections of these structures in media software. This is a natural reflection because the theory of languages expresses organized models for executing algorithms and

structuring data, which are the types of manipulations human creators perform on media when treating it as computational data. Here we see again an example of diSessa’s definition of a literacy: “ the convergence of a large number of genres/social niches on a common, underlying representational form, " that is, a large number of media arts 20 niches are converging on an underlying representational form that look and behave very much like programming languages. Computer Fluency Just as there are degrees of print literacy, ranging from barely literate to highly literate, there are also degrees of computer literacy. Lawrence Snyder, chairman of the Committee on Information Technology Literacy that produced “ Being Fluent With Information Technology, " a report published in June 1999 by the National Research Council, in an interview with Florence Olsen (2000), discusses the differences between computer literacy and computer fluency: Think of fluency as having three kinds of knowledge–skills , concepts, and logical reasoning. Skills are knowing how to use e-mail, browse the Web, and so forth. The basic concepts that students need to know for fluency are such things as how does a computer work, what is a network, how do we represent information digitally, algorithmic thinking, things like that. The intellectual capabilities needed for fluency include logical reasoning, the ability to manage complexity, to troubleshoot and debug information systems. One way to promote such computer fluency is to engage students in computer-based projects: The report proposed teaching fluency in the context of projects that require students to use those three kinds of knowledge. So let me give you an example: formulating an H. I. V.-tracking system for a hospital or doctor's office. It's a great project. It is a database

project, because you need to record clients coming into the clinic, to keep track of the specimens they give, and where the specimens are sent out for testing. A project gives you a chance to learn and practice three or four skills, three or four concepts, and three or four capabilities. 21 One place to embed such computer projects is in programming classes. This wouldn’t necessarily mean teaching C++ or Java or other ‘ heavy-duty’ compiled languages. Guido van Rossum, originator of the Python programming language, initiated a general computer literacy research effort called Computer Programming for Everybody (CP4E) (1999) which purports to “ improve the state of the art of computer use, not by introducing new hardware, nor even (primarily) through new software, but simply by empowering all users to be computer programmers. " He goes on to describe the goals and motivation of this research effort: Our plan has three components: - Develop a new computing curriculum suitable for high school and college students. - Create better, easier to use tools for program development and analysis. - Build a user community around all of the above, encouraging feedback and self-help. … In the future, we envision that computer programming will be taught in elementary school, just like reading, writing and arithmetic. We really mean computer programming–not just computer use (which is already being taught). The Logo project, for example, has shown that young children can benefit from a computing education. Of course, most children won't grow up to be skilled application developers, just as most people don't become professional authors–but reading and writing skills are useful for everyone, and so (in our vision) will be general programming skills. … Even if most users do not program regularly, a

familiarity with programming and the structure of software will make them more effective users of computers. For example, when something goes wrong, they will be able to make a better mental model of the likely failure, which will allow them to fix or work around the problem. 22 By making computer literacy more commonplace, van Rossum hopes to answer the question, “ What will happen if users can program their own computer? " In other words, what changes to individuals and society, analogous to the changes wrought by mass print literacy, will occur when programming computers is as ubiquitous as, say, using a word processor is now? Clark (1997) puts this process in the context of becoming more cyborglike: We see some of the 'cognitive fossil trail' of the Cyborg [cybernetic organism] trait in the historical procession of potent Cognitive Technologies that begins with speech and counting, morphs first into written text and numerals, then into early printing (without moveable typefaces), on to the revolutions of moveable typefaces and the printing press, and most recently to the digital encodings that bring text, sound and image into a uniform and widely transmissible format. Such technologies, once up-and-running in the various appliances and institutions that surround us, do far more than merely allow for the external storage and transmission of ideas. They constitute, I want to say, a cascade of 'mindware upgrades': cognitive upheavals in which the effective architecture of the human mind is altered and transformed. We understand the word ‘ cyborg’ in the sense defined by Haraway (1992) where she says “ the cyborg is the figure born of the interface of automaton and autonomy" (p. 139, in Aarseth, 1997, p. 54); we can think of cyborg as the symbiosis of mechanism and organism. This points to one possible

direction that society may evolve as programming approaches the ubiquity mass print literacy has currently achieved. Computer Literacy as the New Rhetoric Why study computers? Is there any value to such study? One suggestion, which we might call the joy of proficiency, comes from Ong (1991): 23 Technologies are artificial, but–paradox again–artificiality is natural to human beings. Technology, properly interiorized, does not degrade human life but on the contrary, enhances it. The modern orchestra, for example, is the result of high technology. A violin is an instrument, which is to say a tool. An organ is a huge machine, with sources of power — pumps, bellows, electric generators — totally outside its operator. Beethoven's score for his Fifth Symphony consists of very careful directions to highly trained technicians, specifying exactly how to use their tools. … The fact is that by using a mechanical contrivance, a violinist or an organist can express something poignantly human that cannot be expressed without the mechanical contrivance. To achieve such expression of course the violinist or organist has to have interiorized the technology, made the tool or machine a second nature, a psychological part of himself or herself. This calls for years of ' practice', learning how to make the tool do what it can do. Such shaping of a tool to oneself, learning a technological skill, is hardly dehumanizing. The use of a technology can enrich the human psyche, enlarge the human spirit, intensify its interior life. (p. 83) Ong is comparing the technology of musical performance with the technology of writing and arguing that competence in either enhances our lives. To achieve a similar effect of enrichment, enlargement and intensification in our lives through the technology of computer literacy requires a similar process of "

interiorization". This dissertation explores the practices of programming teachers that can guide us to imbue students with at least a taste of the satisfaction that comes from becoming computer literate. To achieve a similar competency with the technology of traditional print literacy, we learned what was traditionally called rhetoric: We have in the West a venerable tradition of studying how human attention is created and allocated: the “ art of persuasion" which the Greeks called rhetoric. A better definition of rhetoric, in fact, might be “ the economics of human attention-structures, " for whenever we “ persuade" someone, we do so by getting that person to “ look at things 24 from our point of view, " share our attention-structure. (Lanham, 1991, p. 227) According to Michael Goldhaber, attention is the currency of cyberspace: So a key question arises: Is there something else that flows through cyberspace [besides information], something that is scarce and desirable? There is. No one would put anything on the Internet without the hope of obtaining some. It's called attention. And the economy of attention - not information - is the natural economy of cyberspace. … In his online book Virtual Community, Howard Rheingold lays out two guidelines: “ Rule Number One is to pay attention. Rule Number Two might be: Attention is a limited resource, so pay attention to where you pay attention. " (1997a; see also Goldhaber, 1997b) Thus, some of these modern expressions of rhetoric, or “ the science of human attention-structures" (Lanham, 1991, p. 134), such as web sites and various other activities emerging on the Internet, are growing in importance. Computer literacy will be essential if students are to succeed in the process of “ interiorizing" web skills that enable competency in these rhetorical activities. We note that the term

rhetoric has a somewhat negative connotation, and for Lanham, unjustifiably so: The intellectual structures of formal rhetoric have formed part of Western culture for so long, and yet we have for so long suspected and despised rhetoric as simply hypocrisy and deception, that it is very difficult to recognize it for what it is–an information system. Systems, at least for humanists, have never escaped from the Platonic orbit; they are closed patterns organized like human society in the Republic. Everyone has a single job; every element a fixed place; the aim is perfect stasis, with an emphasis on both “ perfect" and “ stasis. " Our notion of systems is Platonic philosophy on the one hand and physics on the other. What Plato wanted above all to exile from his utopia, like Thomas More after him, was style, the unabridged range of ornament, of purposeless play. Rhetoric defines itself as a counter-system to the Platonic political order by admitting stylistic, ornamental behavior, by acknowledging that such behavior lies at the heart of human life, is what human politics is all about. If stylistic behavior is acknowledged as part of the complex human “ reason, " then rhetoric becomes the 25 systematic attempt to account for this complex “ reason, " and find agreements within it. (Lanham, 1993; p. 57) Lanham’s major thesis is that philosophy (think ‘ science’) and rhetoric (think ‘ humanities’ or ‘ arts’) are interestingly antagonistic, that philosophy has been ascendant for the past three hundred years or so, that rhetoric is making a comeback (through much of what Ong refers to as ‘ secondary orality’), and that digital technology is making these new rhetorical devices available to the masses: The quarrel between the philosophers and the rhetoricians constitutes the quarrel in Western culture. McLuhan’s argument for electronic media

reintroduced the rhetorician’s conception of language, and of human self and society, after three hundred years dominated by the philosophers, with their strongly opposed conceptions of language and social reality. … The rhetorical/philosophical distinction, though it grows from the technological distinction between oral and literate cultures, concerns more than technology. It debates opposed theories of human motive, human selfhood, and human society. (pp. 202-203) Classical rhetoric, and hence all of classical education, was built on a single dominant exercise: modeling. The key form was the oration, and it was rehearsed again and again in every possible form and context. Declamatio, as the modeling of speeches came to be called, stood at the hub of Western education, just as computer modeling is coming to do today. The world of electronic text has reinstated this centrality of modeled reality. The computer has adopted once again, as the fundamental educational principle, the dramatizing of experience. (p. 47) This ‘ dramatizing of experience’ extends beyond language; we can also, says Richard Buchanan (1989), think of design as rhetoric: Communication is usually considered to be the way a speaker discovers arguments and presents them in suitable words and gestures to persuade an audience. The goal is to induce in the audience some belief about the past (as in legal rhetoric), the present (as in ceremonial rhetoric), or the future (as in deliberative or political rhetoric). The speaker seeks to provide the audience with the reasons for adopting a 26 new attitude or taking a new course of action. In this sense, rhetoric is an art of shaping society, changing the course of individuals and communities, and setting patterns for new action. However, with the rise of technology in the twentieth century, the

remarkable power of man-made objects to accomplish something very similar has been discovered. By presenting an audience of potential users with a new product–whether as simple as a plow or a new form of hybrid seed corn, or as complex as an electric light bulb or a computer–designers have directly influenced the actions of individuals and communities, changed attitudes and values, and shaped society in surprisingly fundamental ways. This is an avenue of persuasion not previously recognized, a mode of communication that has long existed but that has never been entirely understood or treated from a perspective of human control such as rhetoric provides for communication in language. (p. 93) And just as Lanham seeks to reunite philosophy and rhetoric into a more dynamic oscillation than existed previously through the use of computer technology, Buchanan does also for science and design through the use of technology writ large: The primary obstacle to such understanding is the belief that technology is essentially part of science, following all of t